模式

AWS 方案指引

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS 方案指引: 模式

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商標和商業外觀不得用於任何非 Amazon 的產品或服務，也不能以任何可能造成客戶混淆、任何貶低或使 Amazon 名譽受損的方式使用 Amazon 的商標和商業外觀。所有其他非 Amazon 擁有的商標均為其各自擁有者的財產，這些擁有者可能附屬於 Amazon，或與 Amazon 有合作關係，亦或受到 Amazon 贊助。

Table of Contents

AWS 規範性指引模式 1
雲端基礎 2
使用登陸區域加速器自動建立帳戶 3
總結 3
先決條件和限制 3
Architecture 3
工具 5
最佳實務 7
史詩 7
相關資源 30
其他資訊 30
自動清查 AWS 資源 32
摘要 32
先決條件和限制 32
Architecture 33
工具 34
最佳實務 35
史詩 35
疑難排解 40
相關資源 41
使用 MongoDB Atlas 建置 AWS 登陸區域 42
總結 42
先決條件和限制 42
Architecture 43
工具 44
史詩 45
相關資源 52
跨帳戶設定 VPC 流程日誌 53
總結 53
先決條件和限制 53
Architecture 53
工具 54
最佳實務 54
史詩 56

相關資源 57
其他資訊 58
使用 Terraform 管理 AWS 許可集 60
摘要 60
先決條件和限制 60
Architecture 61
工具 62
最佳實務 64
史詩 64
疑難排解 74
相關資源 74
其他資訊 75
自動標記 Transit Gateway 附件 77
總結 77
先決條件和限制 77
Architecture 77
工具 79
史詩 80
相關資源 84
更多模式 85
AI 與機器學習 86
跨帳戶將 AWS CodeCommit 儲存庫與 Amazon SageMaker AI Studio Classic 建立關聯 87
總結 87
先決條件和限制 87
Architecture 87
工具 88
史詩 89
其他資訊 93
從 PDF 檔案自動擷取內容 96
總結 96
先決條件和限制 96
Architecture 97
工具 98
史詩 98
相關資源 102
附件 103

使用 SageMaker AI DeepAR 建置冷啟動預測模型 104
摘要 104
先決條件和限制 104
Architecture 105
工具 105
最佳實務 106
史詩 106
相關資源 108
使用 SageMaker AI 和 Azure DevOps 建置 MLOps 工作流程 109
總結 109
先決條件和限制 109
Architecture 110
工具 111
最佳實務 112
史詩 112
疑難排解 119
相關資源 119
使用 CloudFormation 在 Amazon Bedrock 中設定記錄 121
總結 121
先決條件和限制 121
Architecture 121
工具 122
史詩 123
相關資源 125
在 SageMaker 中建立 Docker 容器，以在 Step Functions 中進行模型訓練 126
總結 126
先決條件和限制 126
Architecture 127
工具 127
史詩 128
相關資源 139
使用 Amazon Bedrock 代理程式在 Amazon EKS 中建立存取控制 140
總結 140
先決條件和限制 140
Architecture 141
工具 141

最佳實務 142
史詩 142
疑難排解 158
相關資源 158
在 上部署 RAG 使用案例 AWS 160
總結 160
先決條件和限制 160
Architecture 161
工具 162
最佳實務 163
史詩 164
相關資源 167
其他資訊 167
在單一 SageMaker 端點中部署多個管道模型物件 169
總結 169
先決條件和限制 169
Architecture 170
工具 170
史詩 171
相關資源 180
使用 Kiro 部署即時編碼安全驗證 181
摘要 181
先決條件和限制 182
Architecture 182
工具 184
最佳實務 185
史詩 186
疑難排解 194
相關資源 194
其他資訊 195
使用 RAG 和 ReAct 提示來開發 AI 聊天助理 198
摘要 198
先決條件和限制 198
Architecture 199
工具 200
最佳實務 202

史詩 202
疑難排解 208
相關資源 208
其他資訊 208
使用 Amazon Bedrock 開發聊天式助理 210
摘要 210
先決條件和限制 210
Architecture 211
工具 212
最佳實務 213
史詩 214
相關資源 217
其他資訊 218
從語音輸入記錄機構知識 220
總結 220
先決條件和限制 220
Architecture 221
工具 221
最佳實務 223
史詩 223
相關資源 228
使用 Amazon Personalize 產生個人化建議 230
總結 230
先決條件和限制 230
Architecture 230
工具 231
史詩 232
相關資源 235
其他資訊 235
使用 SageMaker AI 和 hydra 簡化 ML 工作流程 239
摘要 239
先決條件和限制 239
Architecture 240
工具 240
最佳實務 241
史詩 242

疑難排解 246
相關資源 247
其他資訊 247
..... 248
總結 248
先決條件和限制 248
Architecture 251
工具 252
最佳實務 253
史詩 253
相關資源 259
其他資訊 259
使用 Amazon Q Developer 作為編碼助理 264
總結 264
先決條件和限制 264
工具 265
最佳實務 265
史詩 265
疑難排解 272
相關資源 273
使用 SageMaker Processing 對 TB 級 ML 資料集進行分散式特徵工程 274
總結 274
先決條件和限制 274
Architecture 275
工具 277
史詩 277
相關資源 288
附件 288
使用 Flask 和 Elastic Beanstalk 視覺化 AI/ML 模型結果 289
摘要 289
先決條件和限制 289
Architecture 290
工具 291
史詩 292
相關資源 299
其他資訊 299

更多模式 303
分析 304
在 Microsoft SQL Server Analysis Services 中分析 Amazon Redshift 資料 306
總結 306
先決條件和限制 306
Architecture 306
工具 307
史詩 307
相關資源 309
..... 310
摘要 310
先決條件和限制 310
Architecture 310
工具 311
史詩 312
相關資源 316
自動化從 AWS Data Exchange 到 Amazon S3 的資料擷取 317
總結 317
先決條件和限制 317
Architecture 317
工具 318
史詩 319
相關資源 320
附件 320
在 AWS Glue 中自動化加密強制執行 321
總結 321
先決條件和限制 321
Architecture 321
工具 322
最佳實務 323
史詩 323
相關資源 325
建置資料管道以使用 AWS DataOps 開發套件處理 Google Analytics 資料 326
總結 326
先決條件和限制 326
Architecture 326

工具 327
史詩 328
疑難排解 330
相關資源 330
其他資訊 330
建置影片處理管道 333
總結 333
先決條件和限制 333
Architecture 334
工具 334
史詩 335
相關資源 341
其他資訊 341
附件 342
建置從 Amazon S3 到 Amazon Redshift 的 ETL 管道 343
總結 343
先決條件和限制 343
Architecture 344
工具 344
史詩 345
相關資源 350
其他資訊 350
使用 Amazon DataZone 建置企業資料網格 351
摘要 351
先決條件和限制 351
Architecture 352
工具 353
史詩 354
相關資源 364
其他資訊 365
使用 AWS 服務計算風險值 (VaR) 367
總結 367
先決條件和限制 367
Architecture 368
工具 369
最佳實務 369

史詩 370
相關資源 372
設定共用 AWS Glue Data Catalog 的跨帳戶存取權 373
總結 373
先決條件和限制 373
Architecture 373
工具 374
史詩 375
相關資源 386
其他資訊 386
將 NORMALIZE 轉換為 Amazon Redshift SQL 387
總結 387
先決條件和限制 387
Architecture 387
工具 388
史詩 392
相關資源 393
將 RESET WHEN 轉換為 Amazon Redshift SQL 394
總結 394
先決條件和限制 394
Architecture 394
工具 395
史詩 398
相關資源 398
在 AWS 上部署和管理無伺服器資料湖 400
總結 400
先決條件和限制 400
Architecture 400
工具 402
史詩 403
相關資源 404
..... 406
總結 406
先決條件和限制 406
Architecture 407
工具 407

史詩 408
相關資源 411
附件 411
確保 Amazon EMR 記錄到 Amazon S3 412
總結 412
先決條件和限制 412
Architecture 413
工具 413
史詩 414
相關資源 416
附件 416
使用 AWS Glue 產生測試資料 417
總結 417
先決條件和限制 417
Architecture 417
工具 418
最佳實務 418
史詩 419
相關資源 427
其他資訊 427
將 IoT 資料直接擷取至 Amazon S3 432
總結 432
先決條件和限制 432
Architecture 433
工具 433
最佳實務 434
史詩 434
疑難排解 440
相關資源 441
其他資訊 441
使用 Lambda 函數在 Amazon EMR 中啟動 Spark 任務 445
總結 445
先決條件和限制 445
Architecture 445
工具 446
史詩 447

相關資源 449
其他資訊 450
附件 452
將 Apache Cassandra 工作負載遷移至 Amazon Keyspaces 453
總結 453
先決條件和限制 453
Architecture 453
工具 454
最佳實務 455
史詩 455
疑難排解 467
相關資源 467
其他資訊 467
將 Oracle Business Intelligence 12C 遷移至 AWS 雲端 469
總結 469
先決條件和限制 469
Architecture 470
工具 471
史詩 472
相關資源 482
其他資訊 482
將 ELK 堆疊遷移至 AWS 雲端 487
總結 487
先決條件和限制 487
Architecture 488
工具 490
史詩 491
相關資源 497
其他資訊 498
AWS 使用 Starburst 將資料遷移至 499
總結 499
先決條件和限制 499
Architecture 499
工具 501
史詩 501
相關資源 503

最佳化輸入檔案大小的 ETL 擷取 505
總結 505
先決條件和限制 505
Architecture 505
工具 506
史詩 506
相關資源 509
其他資訊 509
使用 Step Functions 協調 ETL 管道 510
總結 510
先決條件和限制 510
Architecture 511
工具 511
史詩 513
疑難排解 518
相關資源 518
其他資訊 518
使用 Amazon Redshift ML 執行 ML 分析 519
總結 519
先決條件和限制 519
Architecture 520
工具 520
史詩 521
相關資源 524
使用 Amazon Athena 查詢 Amazon DynamoDB 526
總結 526
先決條件和限制 526
Architecture 526
工具 527
史詩 527
疑難排解 530
相關資源 530
使用 Athena 查詢 DynamoDB 資料表 531
總結 531
先決條件和限制 531
Architecture 531

工具 532
史詩 532
相關資源 540
其他資訊 540
設定最少的可行資料空間 542
摘要 542
先決條件和限制 543
Architecture 544
工具 544
最佳實務 545
史詩 546
疑難排解 590
相關資源 590
其他資訊 590
設定 Amazon Redshift 查詢結果的語言特定排序 595
總結 595
先決條件和限制 595
工具 595
史詩 596
相關資源 599
其他資訊 600
訂閱來自跨區域 S3 儲存貯體的事件通知的 Lambda 函數 604
總結 604
先決條件和限制 604
Architecture 604
工具 605
史詩 606
相關資源 609
用於轉換資料的三種 AWS Glue 任務類型 610
總結 610
先決條件和限制 610
Architecture 610
工具 611
史詩 612
相關資源 614
其他資訊 614

附件 620
使用 Athena 和 QuickSight 視覺化 Amazon Redshift 稽核日誌 621
總結 621
先決條件和限制 621
Architecture 621
工具 622
史詩 622
相關資源 625
附件 625
使用 Amazon Quick Sight 視覺化 IAM 登入資料報告 626
摘要 626
先決條件和限制 626
Architecture 627
工具 627
史詩 628
其他資訊 632
更多模式 634
運算 636
容器與微服務 637
從 Amazon EKS 存取 Amazon Neptune 639
存取 Amazon ECS 上的容器應用程式 650
使用 AWS Fargate 啟動類型存取 Amazon ECS 上的容器應用程式 663
在 Amazon EKS 上私下存取容器應用程式 674
自動化 Amazon RDS for PostgreSQL 資料庫執行個體的備份 681
自動化節點終止處理常式的部署 692
自動建置 Java 應用程式並將其部署至 Amazon EKS 705
跨帳戶和區域複製 Amazon ECR 容器映像 725
使用 Amazon EFS 在 EC2 執行個體上建立 Amazon ECS 任務定義 749
使用容器映像部署 Lambda 函數 755
使用 AWS Fargate 在 Amazon ECS 上部署 Java 微服務 762
使用 Amazon EKS 和 Helm 部署 Kubernetes 套件 767
使用 Terraform 在 Amazon EKS 中部署 CockroachDB 叢集 776
在 Amazon EKS 上部署 Java 微服務，並使用 Application Load Balancer 公開 785
在 Amazon EKS 上部署以 gRPC 為基礎的應用程式 796
在 上部署容器化應用程式 AWS IoT Greengrass V2 807
使用 Elastic Beanstalk 部署容器 830

使用 Lambda 和 Amazon VPC 產生靜態傳出 IP 地址 835
自動識別重複的容器映像 846
在 Amazon EKS 工作者節點上安裝 SSM 代理程式 872
使用 preBootstrapCommands 在 Amazon EKS 工作者節點上安裝 SSM 代理程式和 CloudWatch 代理程式 877
遷移 EKS Auto 模式的 NGINX 傳入控制器 883
將容器工作負載從 ARO 遷移至 ROSA 901
在 Amazon WorkSpaces 上執行 Amazon ECS 任務 913
在 AWS 上執行 ASP.NET Web API Docker 容器 922
使用持久性資料儲存體執行具狀態工作負載 933
使用 Amazon EKS Pod Identity 和 KEDA 設定自動擴展 955
..... 974
在 Amazon ECS 中使用交互 TLS 搭配 ALB 993
更多模式 1018
無伺服器 1019
使用 AWS Amplify 建置 React Native 應用程式 1020
跨多個 SaaS 產品集中管理租用戶 1037
合併 Amazon S3 預先簽章的 URL 產生和物件下載 1047
建立跨帳戶 Amazon EventBridge 連線 1061
將 DynamoDB 記錄交付至 Amazon S3 1073
使用 API Gateway 實作以路徑為基礎的 API 版本控制 1079
將 psycopg2 程式庫匯入至 AWS Lambda 1089
將 API Gateway 與 Amazon SQS 整合 1098
與 AWS Lambda 非同步處理 APIs 1111
與 Amazon DynamoDB Streams 非同步處理 APIs 1119
與 Amazon SQS 非同步處理 APIs 1128
從 Step Functions 同步執行 Systems Manager Automation 任務 1136
使用 執行 S3 物件的平行讀取 AWS Lambda 1148
將遙測資料從 Lambda 傳送至 OpenSearch 1159
設定無伺服器儲存格路由器 1171
設定 Amazon S3 儲存貯體的私有存取權 1187
疑難排解 Step Functions 狀態 1193
更多模式 1202
聯網 1204
自動化 AWS Transit Gateway 的對等互連 1205
使用 AWS Transit Gateway 集中網路連線 1210

使用 Application Load Balancer 設定 Oracle JD Edwards EnterpriseOne 的 HTTPS 加密 .. 1215
透過私有網路連線至 Application Migration Service 資料和控制平面 1225
使用 AWS CloudFormation 自訂資源建立 Infoblox 物件 1237
使用 Terraform 建立多區域 IPAM 架構 1249
自訂 Network Firewall 的 CloudWatch 提醒 1262
使用 Terraform 在 Wavelength 區域中部署資源 1278
將大量 DNS 記錄遷移至 Route 53 私有託管區域 1285
當您從 F5 遷移到 AWS 上的 Application Load Balancer 時修改 HTTP 標頭 1294
在多個 中報告 Network Access Analyzer 問題清單 AWS 帳戶 1299
在多帳戶 AWS 環境中設定混合網路的 DNS 解析 1321
..... 1333
使用 Splunk 檢視 AWS Network Firewall 日誌和指標 1338
更多模式 1348
內容交付 1349
使用 Firehose 將 AWS WAF 日誌傳送至 Splunk 1350
使用 CloudFront 透過 VPC 在 S3 儲存貯體中提供靜態內容 1357
更多模式 1365
資料庫和儲存體 1366
資料庫 1367
使用連結的伺服器存取內部部署 SQL Server 資料 1370
在 AWS 上將 HA 新增至 Oracle PeopleSoft 1375
分析從 Oracle 到 PostgreSQL 的部分遷移物件 1397
評估查詢效能以將 SQL Server 遷移至 MongoDB 1412
使用 IaC 自動化 Aurora 全域資料庫的藍/綠部署 1419
自動化 Amazon RDS 執行個體的跨帳戶複寫 1445
使用 AWS Lambda 和 任務排程器自動化資料庫任務 1458
使用 DR Orchestrator Framework 自動化容錯移轉和容錯回復 1467
自動備份 SAP HANA 資料庫 1492
自動為 DynamoDB 產生 PynamoDB 模型和 CRUD 函數 DynamoDB 1499
封鎖對 Amazon RDS 的公開存取 1506
使用 Microsoft AD 設定 Amazon RDS 的 Windows 身分驗證 1513
設定跨帳戶 Amazon DynamoDB 存取 1526
在 Always On 可用性群組中設定唯讀路由 1538
在 pgAdmin 中使用 SSH 通道進行連線 1545
將 JSON Oracle 查詢轉換為 PostgreSQL 資料庫 SQL 1550
跨帳戶複製 Amazon DynamoDB 資料表 1579

跨帳戶複製 Amazon DynamoDB 資料表 1585
建立 Amazon RDS 和 Amazon Aurora 的成本和用量報告 1595
在 Amazon EC2 和 Amazon FSx 上部署 SQL Server FCIs 1601
在 Amazon Aurora 和 Amazon RDS 中模擬 Oracle PL/SQL 關聯陣列 1614
啟用 PostgreSQL 資料庫執行個體的加密連線 1624
加密現有的 Amazon RDS for PostgreSQL 資料庫執行個體 1631
在啟動時強制執行 Amazon RDS 資料庫的自動標記 1637
估算 DynamoDB 成本 1642
Amazon DynamoDB 資料表的預估儲存成本 1651
使用 AWR 報告估計 Oracle 資料庫的 Amazon RDS 引擎大小 1655
將 Amazon RDS for SQL Server 資料表匯出至 S3 儲存貯體 1684
處理動態 SQL 陳述式中的匿名區塊 1693
協助強制執行 DynamoDB 標記 1700
實作跨區域 DR 1705
將超過 100 個引數的 Oracle 函數遷移至 PostgreSQL 1717
將 Redis 工作負載遷移至 AWS 上的 Redis Enterprise Cloud 1722
使用 HSR 將 SAP HANA 遷移至 AWS 1743
使用 AWS MGN 遷移 SQL Server Always On 可用性群組 1753
使用分散式可用性群組將 SQL Server 遷移至 AWS 1762
在 上將關聯式資料庫遷移至 MongoDB Atlas AWS 1771
將自我託管的 MongoDB 環境遷移至 MongoDB Atlas 1779
將 Oracle 資料庫遷移至 Amazon DynamoDB 1788
使用 SharePlex 和 AWS DMS 從 Oracle 8i 或 9i 遷移至 Amazon RDS for Oracle 1793
從內部部署 MySQL 遷移至 Amazon EC2 1800
監控 Amazon Aurora 進行加密 1807
使用 Amazon CloudWatch 監控 GoldenGate 日誌 1812
將 Oracle 資料庫 EE 轉換為 Amazon RDS for Oracle SE2 1824
使用 Precisely Connect 將大型主機資料庫複寫至 AWS 1833
排程 Amazon RDS 和 Aurora PostgreSQL 的任務 1846
使用 SMTP 伺服器傳送 RDS for SQL Server 的通知 1851
在 AWS 上設定 SAP on IBM Db2 的 DR 1862
使用 Terraform 設定資料庫遷移的 CI/CD 管道 1878
使用 FSx 在 Amazon EC2 上設定 Microsoft SQL Server FCI 1885
在 Amazon RDS Custom 上設定 Oracle E-Business Suite 的 HA/DR 架構 1898
在 上將資料從資料庫串流至 MongoDB Atlas AWS 1906
Oracle PeopleSoft 應用程式的轉換角色 1915

跨帳戶從 Amazon Redshift 卸載資料 1947
依工作負載的資料庫遷移模式 1970
更多模式 1979
儲存和備份 1983
允許 EC2 執行個體寫入存取 AMS 中的 S3 儲存貯體 1984
自動化資料串流擷取至 Snowflake 資料庫 1989
自動加密 EBS 磁碟區 1999
在 上備份 Charon-SSP 模擬器中的 Sun SPARC 伺服器 AWS 2008
使用 Veeam 備份資料並存檔至 Amazon S3 2026
跨帳戶和區域複製 Amazon S3 資料 2047
啟用 Amazon S3 的 DB2 日誌封存 2061
使用 DistCp 和適用於 Amazon S3 的 AWS PrivateLink 將 Hadoop 資料遷移至 Amazon S3 2077
更多模式 2090
開發人員工具 2091
DevOps 2092
使用後台和自助式 Amazon SageMaker AI 範本加速 MLOps 2095
自動化 AWS 基礎設施操作 2110
自動化負載平衡器端點中變更的 CloudFront 更新 2118
自動化 AWS CDK Python 應用程式的 CodeGuru 檢閱 2125
自動化 AWS Supply Chain 資料湖部署 2135
自動化 AWS 資源評估 2167
自動化 SAP 系統安裝 2179
使用 AWS CDK 自動化 Service Catalog 產品組合和產品部署 2189
自動化動態 Hotfix 管道 2204
自動化刪除 CloudFormation 堆疊 2228
自動化擷取 Amazon MWAA 自訂指標 2237
使用 AWS CodePipeline 和 AWS CodeBuild 自動化堆疊集部署 2252
自動將 Systems Manager 的受管政策連接至 EC2 執行個體描述檔 2277
為微服務自動建置 CI/CD 管道和 Amazon ECS 叢集 2291
建置 Docker 映像並將其推送至 Amazon ECR 2301
使用 AWS 服務建置和測試 iOS 應用程式 2306
為 Amazon EKS 上的應用程式設定交互 TLS 2311
使用 建立 Amazon WorkSpaces 應用程式資源 CloudFormation 2321
使用 Firelens 為 Amazon ECS 建立自訂日誌剖析器 2327
建立資源協同運作架構 2334
建立 AWS 基礎設施的自動提取請求 2345

為 Java 和 Python 專案建立動態 CI 管道 2355
部署 CloudWatch Synthetics Canary 2368
部署 ChatOps 解決方案來管理 SAST 掃描結果 2374
使用CrewAI 在 Amazon Bedrock 上部署代理系統 2388
..... 2402
在多個 AWS 區域中部署程式碼 2406
將工作負載從 Azure DevOps 管道部署到私有 Amazon EKS 叢集 2416
執行 Amazon Redshift SQL 查詢 2446
將 AWS Backup 報告匯出為 CSV 檔案 2457
將 Amazon EC2 執行個體標籤匯出至 CSV 檔案 2464
產生包含 AWS Config 受管規則的 AWS CloudFormation 範本 AWS Config 2470
讓 SageMaker 筆記本執行個體跨帳戶存取 CodeCommit 儲存庫 2476
實作 GitHub 流程分支策略 2486
實作 Gitflow 分支策略 2492
實作中繼線分支策略 2501
實作集中式自訂 Checkov 掃描 2506
使用 K8sGPT 和 Amazon Bedrock 實作診斷 2514
在偵測單一儲存庫中的變更後啟動不同的 CI/CD 管道 2527
將 Bitbucket 儲存庫與 AWS Amplify 整合 2541
使用 Lambda 跨 AWS 帳戶啟動 CodeBuild 專案 2548
使用應用程式復原控制器管理 EMR 叢集的多可用區域容錯移轉 2560
管理微服務在多個帳戶和區域的藍/綠部署 2572
監控 Amazon ECR 儲存庫是否有萬用字元許可 2605
最佳化多帳戶無伺服器部署 2609
使用 GitHub 動作佈建 AWS Service Catalog 產品 2626
佈建最低權限的 IAM 角色 2635
將 Amazon CloudWatch 指標發佈至 CSV 檔案 2645
跨 移除 Amazon EC2 項目 AWS 帳戶 2650
移除相同 中的 Amazon EC2 項目 AWS 帳戶 2664
在 中執行 Python ETL 任務的單元測試 AWS Glue 2672
使用 CodePipeline 設定 CI/CD 管道 2680
使用 Terraform 大規模設定集中式記錄 2692
在 end-to-end加密 2712
簡化 Amazon EKS 多租戶應用程式部署 2724
簡化 Amazon Lex 工作流程 2740
使用 AWS Fargate WaitCondition 勾點建構 2754

在 AWS CodePipeline 中使用第三方 Git 儲存庫 2764
使用 AWS CodePipeline 驗證 Terraform 組態 2771
更多模式 2784
基礎設施 2787
使用 Session Manager 和 Instance Connect 存取堡壘主機 2788
使用 集中 DNS 解析 AWS Managed Microsoft AD 2803
使用可觀測性存取管理員集中監控 2812
在啟動時檢查 EC2 執行個體是否有強制性標籤 2823
在狀態檔案遺失後安全地清除 AFT 資源 2828
在不支援 AWS CodePipeline 的 AWS 區域中建立管道 2838
使用 AWS CDK 層面和逃生艙自訂預設角色名稱 2845
使用私有靜態 IPs 在 Amazon EC2 上部署 Cassandra 叢集 2852
使用 Transit Gateway Connect VRFs 擴展至 AWS 2857
取得 AWS KMS 金鑰狀態變更的 Amazon SNS 通知 2870
在非工作負載子網路的多帳戶 VPC 設計中保留可路由 IP 空間 2876
從儲存庫在 Service Catalog 中佈建 Terraform 產品 2880
AWS 帳戶 使用單一電子郵件地址註冊多個 2896
在單一帳戶 AWS 環境中設定混合網路的 DNS 解析 2910
在 Amazon EC2 上自動設定 UiPath RPA 機器人 2914
在 AWS 上設定高度可用的 PeopleSoft 架構 2927
設定 Oracle JD Edwards EnterpriseOne 的災難復原 2950
設定偏離偵測和報告 2971
成功匯入 S3 儲存貯體做為 CloudFormation 堆疊 2976
同步不同區域中的 Amazon EFS 檔案系統 2986
使用 LocalStack 和 Terraform 測試來測試 AWS 基礎設施 2993
從 ENSA1 升級 SAP Pacemaker 叢集到 ENSA2 3000
在不同帳戶中使用 VPCs 中的一致可用區域 3021
在 IAM 政策中使用使用者 IDs 3026
在本機驗證帳戶工廠的 Terraform 程式碼 3035
更多模式 3049
Web 和行動應用程式 3052
驗證 React Web 應用程式使用者 3053
使用 AWS Amplify 和 Amazon Cognito 建立 React 應用程式 3062
建立微型前端的入口網站 3076
將以 React 為基礎的 SPA 部署至 Amazon S3 和 CloudFront 3103
使用私有端點部署 Amazon API Gateway API 3110

將 Amazon Quick Sight 視覺化元件內嵌至 Web 應用程式 3116
使用 Green Boost 探索 Web 應用程式開發 3134
使用 AWS CodeBuild 執行單位測試 3157
在六邊形架構中建構 Python 專案 3164
更多模式 3187
IoT 3189
擷取和查詢 AWS IoT SiteWise 中繼資料屬性 3190
總結 3190
先決條件和限制 3190
Architecture 3190
工具 3191
史詩 3192
相關資源 3194
其他資訊 3194
..... 3197
總結 3197
先決條件和限制 3197
Architecture 3198
工具 3199
最佳實務 3199
史詩 3199
疑難排解 3212
相關資源 3214
其他資訊 3214
更多模式 3216
遷移與現代化 3217
遷移 3218
建立 AWS DMS 的 AWS CloudFormation 範本 3219
開始使用自動化產品組合探索 3223
將內部部署 Cloudera 工作負載遷移至 AWS 3229
解決將 SQL Server 遷移至 AWS 後的連線錯誤 3241
自動重新啟動 AWS 複寫代理程式，而不停用 SELinux 3244
重新架構師 3250
重新託管 3570
重新定位 3765
平台重建 3802

依工作負載的遷移模式 4157
更多模式 4165
現代化 4167
自動將過期的 DynamoDB 資料封存至 Amazon S3 4168
在 Amazon OpenSearch Service 中建置多租戶無伺服器架構 4183
部署多堆疊應用程式 4226
使用 AWS SAM 部署巢狀應用程式 4234
使用 AWS Lambda TVM 實作 Amazon S3 的 SaaS 租用戶隔離 4242
使用 Step Functions 實作無伺服器 saga 模式 4265
使用 Amazon ECS Anywhere 管理內部部署容器應用程式 4277
在 AWS 上現代化 ASP.NET Web Forms 應用程式 4285
使用 AWS Fargate 執行事件驅動型工作負載 4297
SaaS 架構中的租戶加入 4305
使用 CQRS 和事件來源 4327
更多模式 4347
大型主機 4349
安裝 以 AWS 服務 從 IBM z/OS 存取 AWS CLI 4350
備份大型主機資料並存檔至 Amazon S3 4369
使用 AWS Mainframe Modernization 和 建置 COBOL Db2 程式 AWS CodeBuild 4391
建置 Micro Focus Enterprise Server PAC 4412
在 AWS 雲端中建置大型主機檔案檢視器 4429
容器化現代化 Blu Age 應用程式 4441
將 EBCDIC 資料轉換為 AWS 上的 ASCII 4449
使用 將大型主機 EBCDIC 檔案轉換為 ASCII 檔案 AWS Lambda 4466
使用複雜的記錄配置轉換大型主機資料檔案 4484
部署容器化應用程式的環境 4497
產生 Db2 z/OS 資料洞見 4505
使用 AWS Mainframe Modernization 和 Amazon Q 產生洞見 4546
實作適用於 AWS Blu Age 的 Microsoft Entra ID 身分驗證 4561
將stonebranch 通用控制器與 AWS 整合 4576
使用 Precisely 將 VSAM 檔案遷移和複寫至 AWS 雲端 4605
使用 現代化大型主機應用程式 AWS Transform 4620
使用 現代化大型主機應用程式 AWS Transform 4634
在 上現代化大型主機輸出管理 AWS 4665
在 上現代化大型主機批次列印工作負載 AWS 4705
使用 Rocket Software Enterprise Suite 現代化大型主機環境 4728

在 AWS 上現代化大型主機線上列印工作負載 4743
使用 Transfer 系列將大型主機檔案移至 Amazon S3 4770
最佳化 AWS Blu Age 現代化應用程式的效能 4781
Db2 聯合資料庫中的安全使用者存取 4800
將 Db2 z/OS 資料傳輸至 AWS 4807
更多模式 4831
管理 4832
成本管理 4833
建立 AWS Glue 任務的詳細成本和用量報告 4834
建立 Amazon EMR 叢集的詳細成本和用量報告 4839
更多模式 4843
最終使用者運算 4844
實作 Amazon WorkSpaces 的 SAML 2.0 身分驗證 4845
更多模式 4853
高效能運算 4854
使用 Terraform 和 DRA 部署 Lustre 檔案系統 4855
設定 AWS ParallelCluster 的 Grafana 監控儀表板 4862
更多模式 4873
混合雲端 4874
在 Amazon ECS Anywhere 上設定混合工作負載的管道 4875
更多模式 4891
管理與治理 4892
Amazon Data Firehose 資源未加密時發出警示 4893
自動化 Amazon VPC IPAM IPv4 CIDR 配置 4897
自動化新增或更新 Windows 登錄項目 4924
使用 Python 自動建立 RFC 4928
自動停止和啟動 Amazon RDS 資料庫執行個體 4934
使用 Terraform 集中 AWS Organizations 中的軟體套件分佈 4946
在 CloudWatch Logs 中設定 .NET 應用程式的記錄 4956
跨 AWS 帳戶和區域複製 AWS Service Catalog 產品 4964
為雲端操作建立 RACI 矩陣 4972
使用 CloudWatch 建立自訂指標的警示 4976
使用預設加密的 EBS 磁碟區建立 AWS Cloud9 IDE 4981
自動建立標籤型 CloudWatch 儀表板 4986
記錄您的登陸區域設計 4994
使用 AWS CDK 在整個組織中啟用 Amazon DevOps Guru 4997

使用 AFT 管理多個帳戶的許可集 5018
使用引導管道實作 AFT 5027
在多個 AWS 帳戶和區域中管理 AWS Service Catalog 產品 5047
監控 SAP RHEL Pacemaker 叢集 5054
使用 CloudWatch Logs Insights 監控應用程式活動 5070
監控跨 使用 AMI AWS 帳戶 5080
檢視 AWS 帳戶或組織的 EBS 快照詳細資訊 5093
更多模式 5099
訊息與通訊 5101
在 Amazon MQ 中自動化 RabbitMQ 組態 Amazon MQ 5102
改善 Amazon Connect 中客服人員工作站的通話品質 5108
更多模式 5119
多帳戶策略 5120
將 AWS 帳戶 從 遷移 AWS Organizations 至 AWS Control Tower 5121
在 AWS Organizations 中設定程式設計帳戶關閉提醒 5129
更多模式 5136
安全性、身分與合規 5137
自動化事件回應和鑑識 5139
總結 5139
先決條件和限制 5139
Architecture 5140
工具 5142
史詩 5143
相關資源 5145
其他資訊 5146
附件 5146
自動稽核來自公有 IP 地址的存取 5147
總結 5147
先決條件和限制 5147
Architecture 5148
工具 5149
最佳實務 5150
史詩 5150
疑難排解 5154
相關資源 5154
自動修復未加密的 Amazon RDS 資料庫執行個體和叢集 5155

總結 5155
先決條件和限制 5155
Architecture 5156
工具 5156
最佳實務 5158
史詩 5158
相關資源 5162
其他資訊 5163
自動驗證和部署 IAM 政策和角色 5164
摘要 5164
先決條件和限制 5164
Architecture 5165
工具 5166
史詩 5166
相關資源 5169
雙向整合 Security Hub CSPM 和 Jira 5171
總結 5171
先決條件和限制 5171
Architecture 5172
工具 5173
史詩 5174
相關資源 5181
其他資訊 5182
為強化的容器映像建置管道 5184
總結 5184
先決條件和限制 5184
Architecture 5185
工具 5187
史詩 5188
疑難排解 5194
相關資源 5195
使用 Terraform 在 AWS Organizations 中集中管理 IAM 存取金鑰 5196
總結 5196
先決條件和限制 5196
Architecture 5197
工具 5198

最佳實務 5199
史詩 5199
疑難排解 5206
相關資源 5206
檢查 Amazon CloudFront 分佈是否有存取記錄、HTTPS 和 TLS 版本 5207
總結 5207
先決條件和限制 5207
Architecture 5208
工具 5208
史詩 5209
相關資源 5211
附件 5211
選擇 Amazon Cognito 身分驗證流程 5212
總結 5212
先決條件和限制 5212
Architecture 5212
工具 5216
史詩 5217
相關資源 5219
其他資訊 5220
使用 Guard 建立 AWS Config 自訂規則 5221
總結 5221
先決條件和限制 5221
Architecture 5222
工具 5226
史詩 5227
疑難排解 5228
相關資源 5229
從多個 建立 Prowler 調查結果的報告 AWS 帳戶 5230
總結 5230
先決條件和限制 5230
Architecture 5231
工具 5232
史詩 5233
疑難排解 5250
相關資源 5250

其他資訊 5251
使用 部署 AWS Control Tower 控制項 AWS CDK 5253
總結 5253
先決條件和限制 5253
Architecture 5254
工具 5255
最佳實務 5256
史詩 5256
相關資源 5262
其他資訊 5263
使用 Terraform 部署 AWS Control Tower 控制項 5266
總結 5266
先決條件和限制 5266
Architecture 5267
工具 5268
最佳實務 5268
史詩 5269
疑難排解 5273
相關資源 5274
其他資訊 5275
AWS WAF 使用 Terraform 部署 的安全自動化 5278
總結 5278
先決條件和限制 5278
Architecture 5278
工具 5279
最佳實務 5279
史詩 5280
疑難排解 5282
相關資源 5282
部署可偵測程式碼中安全問題的管道 5284
總結 5284
先決條件和限制 5284
Architecture 5284
工具 5285
史詩 5286
疑難排解 5288

相關資源 5288
其他資訊 5289
部署公有子網路的偵測控制 5291
摘要 5291
先決條件和限制 5291
Architecture 5292
工具 5293
最佳實務 5293
史詩 5293
相關資源 5301
其他資訊 5301
部署公有子網路的預防性控制 5304
總結 5304
先決條件和限制 5304
Architecture 5305
工具 5306
史詩 5306
相關資源 5310
其他資訊 5311
使用即將過期的 CA 憑證偵測 Amazon RDS 執行個體 5313
總結 5313
先決條件和限制 5313
Architecture 5314
工具 5315
最佳實務 5316
史詩 5316
疑難排解 5320
相關資源 5320
使用 IAM Access Analyzer 動態產生 IAM 政策 5321
總結 5321
先決條件和限制 5321
Architecture 5322
工具 5322
史詩 5324
相關資源 5329
使用 CloudFormation 範本啟用 GuardDuty 5331

總結 5331
先決條件和限制 5331
Architecture 5332
工具 5332
史詩 5333
相關資源 5336
其他資訊 5336
在 Amazon RDS for SQL Server 中啟用透明資料加密 5338
總結 5338
先決條件和限制 5338
Architecture 5338
工具 5339
史詩 5339
相關資源 5341
協助防止排程的 KMS 金鑰刪除 5342
總結 5342
先決條件和限制 5342
Architecture 5343
工具 5343
史詩 5344
相關資源 5347
其他資訊 5347
附件 5348
在 中識別公有 Amazon S3 儲存貯體 AWS Organizations 5349
總結 5349
先決條件和限制 5349
Architecture 5350
工具 5350
史詩 5351
疑難排解 5354
相關資源 5354
其他資訊 5354
將 AWS 安全日誌擷取至 Microsoft Sentinel 5356
總結 5356
先決條件和限制 5356
Architecture 5357

工具 5358
最佳實務 5359
史詩 5359
相關資源 5367
以程式碼形式管理 AWS Organizations 政策 5368
總結 5368
先決條件和限制 5368
Architecture 5369
工具 5370
最佳實務 5371
史詩 5371
疑難排解 5386
相關資源 5387
其他資訊 5387
使用 CodePipeline 管理 IAM Identity Center 許可集 5389
總結 5389
先決條件和限制 5389
Architecture 5390
工具 5391
最佳實務 5392
史詩 5392
疑難排解 5399
相關資源 5399
使用 AWS Secrets Manager 管理登入資料 5400
總結 5400
先決條件和限制 5400
Architecture 5400
工具 5401
史詩 5401
相關資源 5402
其他資訊 5402
監控 Amazon ElastiCache 叢集的靜態加密 5405
總結 5405
先決條件和限制 5405
Architecture 5406
工具 5407

最佳實務 5407
史詩 5408
疑難排解 5411
相關資源 5412
監控 IAM 根使用者活動 5413
總結 5413
先決條件和限制 5413
Architecture 5414
工具 5414
史詩 5415
相關資源 5419
其他資訊 5419
建立 IAM 使用者時通知 5420
總結 5420
先決條件和限制 5420
Architecture 5420
工具 5421
史詩 5422
相關資源 5423
附件 5424
使用 SCP 防止網際網路存取 5425
總結 5425
先決條件和限制 5425
工具 5426
最佳實務 5426
史詩 5426
相關資源 5428
報告 IAM Identity Center 身分及其指派 5429
總結 5429
先決條件和限制 5429
Architecture 5430
工具 5431
史詩 5431
疑難排解 5432
相關資源 5433
其他資訊 5434

根據 IP 地址或地理位置限制存取 5436
總結 5436
先決條件和限制 5436
工具 5437
史詩 5438
相關資源 5442
掃描 Git 儲存庫以取得敏感資訊 5443
總結 5443
先決條件和限制 5443
Architecture 5443
工具 5443
最佳實務 5444
史詩 5444
相關資源 5448
使用 Transfer Family 和 GuardDuty 進行安全的檔案傳輸 5449
總結 5449
先決條件和限制 5449
Architecture 5450
工具 5451
最佳實務 5452
史詩 5453
疑難排解 5456
相關資源 5458
其他資訊 5458
使用 Macie 保護 CloudWatch Logs 中的敏感資料 5460
總結 5460
先決條件和限制 5460
Architecture 5461
工具 5462
最佳實務 5463
史詩 5463
疑難排解 5479
相關資源 5479
從 AWS Network Firewall 傳送提醒到 Slack 頻道 5480
總結 5480
先決條件和限制 5480

Architecture 5481
工具 5481
史詩 5482
相關資源 5487
其他資訊 5487
將自訂屬性傳送至 Amazon Cognito 權杖 5491
總結 5491
先決條件和限制 5491
Architecture 5492
工具 5492
最佳實務 5493
史詩 5494
疑難排解 5502
相關資源 5502
其他資訊 5502
使用 AWS Private CA 和 AWS RAM 簡化私有憑證管理 5506
總結 5506
先決條件和限制 5506
Architecture 5507
工具 5507
史詩 5508
相關資源 5514
其他資訊 5514
使用 Amazon Bedrock 代理程式和 簡化 Amazon EC2 合規管理 AWS Config 5515
總結 5515
先決條件和限制 5515
Architecture 5516
工具 5517
最佳實務 5517
史詩 5518
疑難排解 5520
相關資源 5520
使用 PowerShell 從 IAM Identity Center 更新憑證 5521
總結 5521
先決條件和限制 5521
Architecture 5522

工具 5522
最佳實務 5522
史詩 5523
疑難排解 5525
相關資源 5525
其他資訊 5525
使用 Network Firewall 從傳出流量擷取 DNS 名稱 5528
總結 5528
先決條件和限制 5528
Architecture 5528
工具 5529
史詩 5530
使用 Terraform 自動啟用 GuardDuty 5543
總結 5543
先決條件和限制 5543
Architecture 5545
工具 5546
史詩 5547
相關資源 5554
其他資訊 5554
驗證 PCI DSS 4.0 的最佳實務 5556
總結 5556
先決條件和限制 5556
工具 5557
史詩 5558
相關資源 5559
其他資訊 5560
附件 5560
更多模式 5561
..... 5563

AWS 規範性指引模式

Amazon Web Services (AWS) 方案指引模式提供step-by-step指示、架構、工具和程式碼。這些模式由主題專家在 審核 AWS，適用於計劃遷移至或正在進行遷移的建置人員和實作使用者 AWS。他們也支援已經在 上， AWS 並正在尋找方法來最佳化或現代化其雲端操作的使用者。

您可以使用這些模式將複雜程度不同的內部部署或雲端工作負載移至 AWS ，並加速雲端採用、最佳化和現代化工作，無論您是否處於專案的概念驗證、規劃或實作階段。例如，對於雲端遷移專案：

- 在規劃階段，您可以評估可遷移到的不同選項 AWS。您可以選擇符合您需求的正確模式，取決於您想要重新放置、重新託管、轉換或重新建構。您也可以了解可用於遷移的不同工具，並開始規劃購買授權或開始與廠商的初始對話。
- 在概念驗證和實作階段，您可以遵循模式中提供的step-by-step指示，將工作負載遷移至其中 AWS。每個模式都包含先決條件、目標參考架構、工具、step-by-step任務、最佳實務、疑難排解和程式碼等詳細資訊。
- 如果您已經在使用 AWS 雲端，您可以找到可協助您現代化、最佳化、擴展和保護雲端資源使用的模式。

若要依技術網域檢視模式清單，請使用以下連結或AWS 規範指引首頁上的篩選和搜尋選項。

- 雲端基礎
- AI 與機器學習
- 分析
- 運算
- 資料庫和儲存體
- 開發人員工具
- IoT
- 遷移與現代化
- 管理
- 安全性、身分與合規

若要檢視所有出版物，包括指南、策略和模式，請參閱 AWS 規範指引 首頁。

雲端基礎

主題

- 在 上使用登陸區域加速器自動建立帳戶 AWS
- 跨多個帳戶和區域自動清查 AWS 資源
- 建置包含 MongoDB Atlas 的 AWS 登陸區域
- 設定 VPC 流程日誌以進行跨 的集中化 AWS 帳戶
- 使用 Terraform 動態管理 AWS 許可集
- 使用 AWS Organizations 自動標記 Transit Gateway 連接
- 更多模式

在 上使用登陸區域加速器自動建立帳戶 AWS

Justin Kuskowski、Joe Behrens 和 Nathan Scott，Amazon Web Services

總結

此模式說明如何在 解決方案上使用登陸區域加速器 AWS，在授權使用者提交請求 AWS 帳戶 時自動部署新的 。它使用 AWS Step Functions 來協調許多 AWS Lambda 函數。Lambda 函數會將帳戶資訊新增至 Git 儲存庫、啟動 AWS CodePipeline 管道，並驗證是否已佈建必要的 AWS 資源。程序完成時，使用者會收到帳戶已建立的通知。

或者，您可以整合 Microsoft Entra ID 群組，並在帳戶建立程序期間指派 AWS IAM Identity Center 許可集。如果您的組織使用 Microsoft Entra ID 做為身分來源，此選用功能可協助您自動管理和設定新帳戶的存取權。

先決條件和限制

先決條件

- 在 中存取管理帳戶 AWS Organizations
- AWS Cloud Development Kit (AWS CDK) 2.118.0 版或更新版本，已安裝並設定
- Python 3.9 版或更新版本，已安裝
- AWS Command Line Interface (AWS CLI) 2.13.19 版或更新版本，已安裝
- Docker 24.0.6 版或更新版本，已安裝
- AWS 解決方案上的登陸區域加速器，部署在管理帳戶中
- （選用） Microsoft Entra ID 和 IAM Identity Center，整合

限制

帳戶建立工作流程支援循序執行來部署單一 AWS 帳戶。此限制可確保帳戶建立工作流程已成功完成，而無需在平行執行期間競爭資源。

Architecture

目標架構

下圖顯示 AWS 帳戶 使用 Landing Zone Accelerator on. AWS AWS Step Functions orchestrates 自動化自動化建立新 的高階架構。Step Functions 工作流程中的每個任務都由一或多個 AWS Lambda 函數執行。

該圖顯示以下工作流程：

1. 使用者透過執行 Python 指令碼或使用 Amazon API Gateway 來請求帳戶。
2. Account Creation Orchestrator 工作流程從 開始 AWS Step Functions。
3. 工作流程會更新來源碼儲存庫中的 `account-config.yaml` 檔案。它也會啟動 AWS 管道上的登陸區域加速器，並檢查管道的狀態。此管道會建立和設定新帳戶。如需此功能運作方式的詳細資訊，請參閱 上登陸區域加速器的架構概觀 AWS。
4. （選用）當管道完成時，工作流程會檢查群組是否存在於 Microsoft Entra ID 中。如果群組不存在於 Microsoft Entra ID 中，工作流程會將群組新增至 Microsoft Entra ID。
5. 工作流程會執行登陸區域加速器在 AWS 解決方案上無法執行的其他步驟。預設步驟包括：
   - 在 AWS Identity and Access Management (IAM) 中建立帳戶別名
   - 在 中將標籤連接至帳戶 AWS Organizations
   - 根據指派給帳戶的標籤在AWS Systems Manager 參數存放區中建立參數
6. （選用）工作流程會將一或多個許可集指派給您先前指定的 Microsoft Entra ID 群組。許可集允許群組中的使用者存取新帳戶，並允許他們執行您設定的動作。
7. AWS Lambda 函數會執行 QA 和驗證測試。它會驗證資源建立、檢查標籤是否已建立，並驗證是否已部署安全資源。
8. 工作流程會釋出帳戶，並使用 Amazon Simple Email Service (Amazon SES) 通知使用者程序已成功完成。

如需 Step Functions 工作流程的詳細資訊，請參閱此模式額外資訊區段中的 Step Functions 工作流程圖表。

Microsoft Entra ID 應用程式

如果您選擇與 Microsoft Entra ID 整合，請在部署此模式時建立下列兩個應用程式：

- 連結至 IAM Identity Center 的應用程式，並確保 Microsoft Entra ID 群組可在 IAM Identity Center 中使用。在此範例中，此 Microsoft Entra ID 應用程式名為 `LZA2`。
- 允許 Lambda 函數與 Microsoft Entra ID 通訊並呼叫 Microsoft Graph APIs的應用程式。在此模式中，此應用程式名為 `create_aws_account`。

這些應用程式會收集用來同步 Microsoft Entra ID 群組和指派許可集的資料。

工具

AWS 服務

- Amazon API Gateway 可協助您建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。在此模式中，您可以使用 API Gateway 來檢查 AWS 帳戶 名稱的可用性、啟動 AWS Step Functions 工作流程，以及檢查 Step Functions 執行的狀態。
- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端 基礎設施。
- AWS Control Tower 可協助您設定和管理 AWS 多帳戶環境，並遵循規範最佳實務。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如， AWS Lambda 函數、使用 API 目的地的 HTTP 呼叫端點，或其他事件匯流排 AWS 帳戶。如果 Step Functions 工作流程狀態變更為 `Failed`、 或 `Timed-out`，此解決方案會使用 EventBridge 規則來啟動 Lambda 函數`Aborted`。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS IAM Identity Center 可協助您集中管理所有 AWS 帳戶 和雲端應用程式的單一登入 (SSO) 存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。在此模式中， AWS KMS 金鑰用於加密資料，例如存放在 Amazon Simple Storage Service (Amazon S3) 中的資料、Lambda 環境變數，以及 Step Functions 中的資料。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。
- Amazon Simple Email Service (Amazon SES) 可協助您使用自己的電子郵件地址和網域來傳送和接收電子郵件。成功建立新帳戶後，您會透過 Amazon SES 收到通知。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。如果在帳戶建立過程中發生錯誤，Amazon SNS 會傳送通知到您設定的電子郵件地址。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務 來建置業務關鍵型應用程式。
- AWS Systems Manager 參數存放區為組態資料管理和秘密管理提供安全的階層式儲存。

其他工具

- awscurl 會將簽署 AWS API 請求的程序自動化，並協助您以標準 curl 命令的形式提出請求。
- Microsoft Entra ID，先前稱為 Azure Active Directory，是一種雲端型身分和存取管理服務。
- Microsoft Graph APIs可協助您存取 Microsoft 雲端服務中的資料和智慧，例如 Microsoft Entra 和 Microsoft 365。

程式碼儲存庫

此模式的程式碼可在 GitHub lza-account-creation-workflow 儲存庫中使用。

lambda_layer 目錄包含下列層，在多個 Lambda 函數中參考：

- account_creation_helper – 此 layer 包含擔任角色和檢查進度的模組 AWS Service Catalog。
- boto3 – 此 layer 包含 適用於 Python (Boto3) 的 AWS SDK模組，以確保 AWS Lambda 具有最新版本。
- identity_center_helper – 此層支援呼叫 IAM Identity Center。

lambda_src 目錄包含下列 Lambda 函數：

- AccountTagToSsmParameter – 此函數會使用連接至帳戶的標籤， AWS Organizations 以在參數存放區中建立參數。每個參數都以 `/account/tags/`字首開頭。
- AttachPermissionSet – 此函數會將許可集新增至 IAM Identity Center 群組。
- AzureADGroupSync – 此函數會將目標 Microsoft Entra ID 群組同步至 IAM Identity Center。
- CheckForRunningProcesses – 此函數會檢查`AWSAccelerator-Pipeline`管道目前是否正在執行。如果管道正在執行，則函數會延遲 AWS Step Functions 工作流程。
- CreateAccount – 此函數使用 AWS Service Catalog 和 AWS Control Tower 建立新的 AWS 帳戶。
- CreateAdditionalResources – 此函數會建立非由 Landing Zone Accelerator 或 管理 AWS 的資源 AWS CloudFormation，例如帳戶別名和 AWS Service Catalog 標籤。
- GetAccountStatus – 此函數會掃描 中佈建的產品 AWS Service Catalog ，以判斷帳戶建立程序是否已完成。
- GetExecutionStatus – 此函數會擷取執行中或已完成 AWS Step Functions 執行的狀態。
- NameAvailability – 此函數會檢查 AWS 帳戶 名稱是否已存在於其中 AWS Organizations。
- ReturnResponse – 如果帳戶建立成功，此函數會傳回新帳戶的 ID。如果帳戶建立失敗，則會傳回錯誤訊息。

- RunStepFunction – 此函數會執行建立帳戶的 AWS Step Functions 工作流程。
- SendEmailWithSES – 此函數會傳送電子郵件給等待帳戶建立完成的使用者。
- ValidateADGroupSyncToSSO – 此函數會檢查指定的 Microsoft Entra ID 群組是否與 IAM Identity Center 同步。
- ValidateResources – 此函數會驗證所有 AWS Control Tower 自訂是否已成功執行。

最佳實務

我們建議採用下列命名慣例 AWS CDK：

- 使用p字首啟動所有參數。
- 使用c字首啟動所有條件。
- 使用 r字首啟動所有資源。
- 使用 o字首啟動所有輸出。

史詩

部署用於驗證和標記的 IAM 角色

任務	Description	所需的技能
在 上準備登陸區域加速器 AWS 以進行自訂。	1. 在 AWS 程式碼儲存庫上的登陸區域加速器中，建立名為 的檔案customizations-config.yaml 。您可以使用此檔案來定義核心解決方案的自訂。如需詳細資訊，請參閱自訂解決方案。 2. 在 customizations-config.yaml 檔案中，建立名為 的區段cloudFormationStacks 。	AWS DevOps

任務	Description	所需的技能
準備部署lza-account-creation-validation 角色。	現在，您可以自訂解決方案，在管理帳戶以外的所有帳戶中部署 lza-account-creation-validation IAM 角色。此角色提供 ValidateResources Lambda 函數對新帳戶的唯讀存取權。 1. 從 GitHub 下載 account-creation-validation-role.yaml 檔案。 2. 將檔案儲存到customizations-config.yaml 檔案範本區段中指示的位置。 3. 開啟 customizations-config.yaml 檔案。 4. 在 cloudFormationStacks 區段中，新增下列程式碼。 AWS 區域 視需要更新登陸區域的目標： `- deploymentTargets: organizationalUnits: - Root excludedAccounts: - Management description: IAM Role to allow Account Validation`	AWS DevOps

任務	Description	所需的技能

	<pre>name: lza-account-creation-validation
	regions:
	- us-east-1
	template: cloudformation/account-creation-validation-role.yaml
	runOrder: 1
	terminationProtection: true
	parameters:
	- name: pManagementAccountId
	value: "{{ account Management }}"</pre>
	5. 儲存並關閉 `customizations-config.yaml` 檔案。	

任務	Description	所需的技能
準備部署account-tagging-to-ssm-parameter-role 角色。	現在，您可以自訂解決方案，在管理帳戶以外的所有帳戶中部署 account-tagging-to-ssm-parameter-role IAM 角色。此角色用於在 AWS Systems Manager 參數存放區中建立參數。 1. 從 GitHub 下載 account-tagging-to-ssm-parameter-role.yaml 檔案。 2. 將檔案儲存到customizations-config.yaml 檔案範本區段中指示的位置。 3. 開啟 customizations-config.yaml 檔案。 4. 在 cloudFormationStacks 區段中，新增下列程式碼。 AWS 區域 視需要更新登陸區域的目標： `- deploymentTargets: organizationalUnits: - Root excludedAccounts: - Management description: IAM Role to create SSM Parameters based on Account Tagging`	AWS DevOps

任務	Description	所需的技能

	<pre>name: lza-account-tagging-to-ssm-parameter
	regions:
	- us-east-1
	template: cloudformation/account-tagging-to-ssm-parameter-role.yaml
	runOrder: 1
	terminationProtection: true
	
	parameters:
	- name: pManagementAccountId
	value: "{{ account Management }}"</pre> 5. 儲存並關閉 `customizations-config.yaml` 檔案。	

任務	Description	所需的技能
準備部署config-log-validation-role 角色。	現在，您可以自訂解決方案，在日誌封存帳戶中部署 config-log-validation-role IAM 角色。此角色允許 ValidateResources Lambda 函數存取用於記錄和存取 AWS Config 規則的 Amazon S3 儲存貯體。 1. 從 GitHub 下載 config-log-validation-role.yaml 檔案。 2. 將檔案儲存到customizations-config.yaml 檔案範本區段中指示的位置。 3. 開啟 customizations-config.yaml 檔案。 4. 在 cloudFormationStacks 區段中，新增下列程式碼。 AWS 區域 視需要更新登陸區域的目標： `- deploymentTargets: accounts: - LogArchive description: IAM Role to validate Config and Logs name: lza-config-log-validation-role regions:`	AWS DevOps

任務	Description	所需的技能

	<pre>- us-
	east-1
	template:
	cloudformation/con
	fig-log-validation-
	role.yaml
	runOrder: 1
	terminati
	onProtection: true
	
	parameters:
	- name:
	pManagementAccount
	Id
	value:
	"{{ account Managemen
	t }}"</pre> 5. 儲存、關閉和遞交對 `customizations-con fig.yaml` 檔案所做的變更。	

（選用）從 Microsoft Entra ID 取得資料

任務	Description	所需的技能
建立允許 Lambda 函數與 Microsoft Entra ID 通訊的應用程式。	1. 在 Microsoft Entra ID 管理中心，註冊`create_aw s_account` 應用程式。如需說明，請參閱 Microsoft 文件中的註冊應用程式。 2. 遵循 Microsoft 文件中的更新應用程式請求的許可中的指示，為`create_aw s_account` 應用程式設	Microsoft Entra ID

任務	Description	所需的技能

	定下列 Microsoft Graph 許可：
	• Application.Read.All
	  • 類型：應用程式
	  • 需要管理員同意：是
	• AppRoleAssignment.ReadWrite.All
	  • 類型：應用程式
	  • 需要管理員同意：是
	• Group.ReadWrite.All
	  • 類型：應用程式
	  • 需要管理員同意：是
	• ServicePrincipalEndpoint.Read.All
	  • 類型：應用程式
	  • 需要管理員同意：是
	• Synchronization.ReadWrite.All
	  • 類型：應用程式
	  • 需要管理員同意：是
	• User.Read
	  • 類型：委派
	  • 需要管理員同意：否	

任務	Description	所需的技能
擷取`create_aws_account` 應用程式的值。	現在，您可以擷取`create_aws_account` 應用程式所需的值。 1. 在 Microsoft Entra ID 管理中心，導覽至應用程式註冊，然後選擇 `create_aws_account` 。 2. 在左側窗格中，選擇概觀。 3. 在概觀頁面上，記下下列值： • 應用程式（用戶端）ID • 目錄（租戶）ID 4. 在左側窗格中的管理下，選擇憑證和秘密。 5. 在憑證與秘密頁面上，選擇用戶端秘密索引標籤，然後記下下列值： • 用戶端秘密值 • 用戶端秘密 ID	Microsoft Entra ID
建立將 Microsoft Entra ID 與 IAM Identity Center 整合的應用程式。	在 Microsoft Entra ID 管理中心，註冊LZA2應用程式。如需說明，請參閱 Microsoft 文件中的註冊應用程式。	Microsoft Entra ID

任務	Description	所需的技能
擷取LZA2應用程式的值。	現在，您可以擷取LZA2應用程式所需的值。 1. 在 Microsoft Entra ID 管理中心，導覽至企業應用程式，然後選擇 LZA2。 2. 在左側窗格中，選擇概觀。 3. 在概觀頁面上，記下下列值： • Name • 物件 ID 4. 在左側窗格中的管理下，選擇資訊清單。 5. 在 JSON 檔案中的 `appRoles`區段中，找到名為 的應用程式角色`User`。 6. 請記下此應用程式角色`id`的值。	Microsoft Entra ID

任務	Description	所需的技能

建立秘密。	1. 在 中 AWS CLI，輸入下列命令來建立變數。使用您為 `create_aws_account` 和 LZA2 應用程式擷取的值：
	`# Variables for create_aws_account app`
	`TENANT_ID='<Directory ID>'`
	`CLIENT_ID='<Application ID>'`
	`SECRET_ID='<Client secret ID>'`
	`SECRET_VALUE='<Client secret value>'`
	
	`# Variables for LZA2 app`
	`OBJECT_ID='<Object ID>'`
	`APP_ROLE_ID='<App role ID>'`
	`ENTERPRISE_APP_NAME='<Name>'`
	2. 輸入下列命令以建立名為 `GraphApiSecret` 的秘密 AWS Secrets Manager：
	`aws secretsmanager create-secret \`
	`--name GraphApiSecret \`
	`--secret-string "{\"client_id\": \"${CLIENT_ID}\", \"tenant_id\": \"${TENANT_ID}\",`
		AWS DevOps

任務	Description	所需的技能

	<pre>\"object_id\":
	\"${OBJECT_ID}\",
	\"app_role_id\":
	\"${APP_ROLE_ID}\"
	, \"secret_value\":
	\"${SECRET_VALUE}\
	", \"secret_id\":
	\"${SECRET_ID}\"}"</pre>
	如果您未來需要更新秘密，您可以更新變數並執行下列命令：
	<pre>aws secretsmanager
	update-secret \
	--secret-id
	GraphApiSecret \
	--secret-string
	"{\"client_id\":
	\"${CLIENT_ID}\",
	\"tenant_id\":
	\"${TENANT_ID}\",
	\"object_id\":
	\"${OBJECT_ID}\",
	\"app_role_id\":
	\"${APP_ROLE_ID}\"
	, \"secret_value\":
	\"${SECRET_VALUE}\
	", \"secret_id\":
	\"${SECRET_ID}\"}"</pre>	

部署解決方案

任務	Description	所需的技能
複製原始程式碼。	1. 輸入下列命令以複製 lza-account-creation-workflow 儲存庫：	DevOps 工程師

任務	Description	所需的技能
	``` git clone https://github.com/aws-samples/lza-account-creation-workflow ``` 2. 輸入下列命令以導覽至複製儲存庫的 config 目錄： ``` cd lza-account-creation-workflow/config ```	

任務	Description	所需的技能
更新 deploy-config.yaml 檔案。	1. 開啟 deploy-config.yaml 檔案。 2. 檢閱範本並視需要更新值，以便在您的 AWS 環境中進行部署。例如，更新下列項目的值： • accountCreationFailure • accountCompletionFromEmail • ssoLoginUrl • rootEmailPrefix • rootEmailDomain 3. 如果您要與 Microsoft Entra ID 整合，請執行下列動作： • 將 enableAzureADIntegration 設定為 true。 • 針對graphApiSecretName 值，輸入您先前建立的秘密 (GraphApiSecret)。 4. 儲存並關閉 deploy-config.yaml 檔案。	AWS DevOps

任務	Description	所需的技能

在您的 AWS 環境中部署解決方案。	1. 輸入以下命令：
	`cdk bootstrap <account-number>/<Region>`
	如需詳細資訊，請參閱 AWS CDK 文件中的引導。
	2. 輸入下列命令來合成 CloudFormation 範本：
	`cdk synth`
	如需詳細資訊，請參閱 AWS CDK 文件中的設定和執行 AWS CDK 堆疊合成。
	3. 輸入下列命令來部署解決方案。
	`cdk deploy`
	如需詳細資訊，請參閱 AWS CDK 文件中的部署 AWS CDK 應用程式。
	Note
	此解決方案使用 Amazon S3 儲存貯體來存放此解決方案的原始程式碼。您可以使用 upload_to_source_bucket.py 指令碼來建立
		AWS DevOps

任務	Description	所需的技能
	原始碼的封存，並上傳更新的版本。	

選項 1 – 使用 Python 建立帳戶

任務	Description	所需的技能
識別要使用的引數。	選擇當您執行啟動 Step Functions 工作流程的 Python 指令碼時要使用的引數。如需引數的完整清單，請參閱此模式的其他資訊???一節。	AWS DevOps、Python

啟動 Python 指令碼。	1. 輸入下列命令以導覽至複製的儲存庫：
	`cd lza-account-creation-workflow`
	2. 執行啟動 Step Functions 工作流程的 Python 指令碼。以下是包含範例引數和值的範例命令：
	`python ./run-stepfunction.py \ --account-name "lza-test-01" \ --support-dl "johnsmith@example.com" \ --managed-org-unit "Workloads/Workload-1" \ --purpose "Testing new micro service" \`
		DevOps 工程師，Python

任務	Description	所需的技能
	--force-update true \ --ad-integration "{\"CustomerAccountAdmin\": \"platform-admin\", \"CustomerAccountDev\": \"workload1-app1\"}" \ --bypass-creation true \ --tags APPLICATION=TestingMicroService 3. 等到您收到已成功建立帳戶的通知。 **Note** 如果帳戶建立程序失敗，Amazon SNS 會傳送通知到您在 accountCreationFailure deploy-config.yaml 檔案中定義的電子郵件地址。帳戶請求者不會收到通知。	

選項 2 – 使用 API Gateway 和 awscurl 建立帳戶

任務	Description	所需的技能
設定 awscurl 的變數。	1. 輸入下列命令以導覽至來源碼目錄：	AWS DevOps

任務	Description	所需的技能
	``` cd lza-account-creati on-workflow ``` 2. 輸入下列命令來設定 AWS 存取金鑰變數。您可以從AWS 存取入口網站複製變數，然後將其貼入 shell。以下是範例： ``` export AWS_ACCES S_KEY_ID="<id>" export AWS_SECRE T_ACCESS_KEY="<key >" export AWS_SESSI ON_TOKEN="<token>" ``` 3. 輸入下列命令來設定 API 呼叫 AWS 區域 的 ： ``` export AWS_REGION= $(aws configure get region) ``` 4. 輸入下列命令，從 `lza-account-creation-workflow-application` CloudFormation 輸出擷取 API Gateway 端點： ``` export API_GATEW AY_ENDPOINT=$(aws cloudformation describe-stacks -- stack-name "lza-acco unt-creation-workf low-application" --query 'Stacks[* ```	

任務	Description	所需的技能
	```].Outputs[?OutputKey==`oApiGatewayCreateAccountEndpoint`].OutputValue' --output text)```	
檢查名稱可用性。	輸入下列命令來驗證名稱是否可供 使用 AWS 帳戶。<AWS_ACCOUNT_NAME> 將 取代為目標帳戶的名稱： ```awscurl --service execute-api \ --region ${AWS_REGION} \ --access_key ${AWS_ACCESS_KEY_ID} \ --secret_key ${AWS_SECRET_ACCESS_KEY} \ --security_token ${AWS_SESSION_TOKEN} \ -X POST ${API_GATEWAY_ENDPOINT}check_name?account_name=<AWS_ACCOUNT_NAME>```	AWS DevOps

任務	Description	所需的技能
執行帳戶建立工作流程。	1. 在複製儲存庫的根資料夾中，開啟 `api_example.json` 檔案。 2. 使用組態值更新參數： `{ "account_name": "lza-test-01", "account_email": "johnsmith+lzatest01@example.com", "support_dl": "johnsmith@example.com", "managed_org_unit": "Workloads/Workload-1", "ad_integration": [{ "PermissionSetName": "CustomerAccountAdmin", "ActiveDirectoryGroupName": "platform-admin" }, { "PermissionSetName": "CustomerAccountDev", "ActiveDirectoryGroupName": "workload1-app1" }], "force_update": "true",`	AWS DevOps

任務	Description	所需的技能

	<pre> "bypass_c
	reation": "false",
	    "account_tags": [
	        {
	            "Key":
	 "Environment",
	            "Value":
	 "Dev"
	        }, {
	            "Key":
	 "DeploymentMethod",
	            "Value":
	 "ApiGateway"
	        }
	    ]
	}</pre>	

	3. 儲存並關閉 `api_examp le.json` 檔案。	

	4. 輸入下列命令以啟動 Step Functions 工作流程：
	<pre>awscurl --service
	 execute-api \
	    --data @api-exam
	ple.json \
	    --region ${AWS_REG
	ION} \
	    --access_key
	 ${AWS_ACCESS_KEY_I
	D} \
	    --secret_key
	 ${AWS_SECRET_ACCES
	S_KEY} \
	    --security_token
	 ${AWS_SESSION_TOKE
	N} \
	    -X POST ${API_GAT
	EWAY_ENDPOINT}exec
	ute</pre>	

任務	Description	所需的技能
	5. 在上一個命令的輸出中，記下 Step Functions 執行 Amazon Resource Name (ARN)。 6. 如果您想要檢查 Step Functions 工作流程的狀態，請輸入下列命令。<STEP_FUNCTION_EXECUTION_NAME> 將 取代為 Step Functions 執行 ARN 或名稱： `awscurl --service execute-api \ --region ${AWS_REGION} \ --access_key ${AWS_ACCESS_KEY_ID} \ --secret_key ${AWS_SECRET_ACCESS_KEY} \ --security_token ${AWS_SESSION_TOKEN} \ -X GET ${API_GATEWAY_ENDPOINT}get_execution_status?execution=<STEP_FUNCTION_EXECUTION_NAME>` 7. 等到您收到已成功建立帳戶的通知。	

任務	Description	所需的技能
	Note 如果帳戶建立程序失敗，Amazon SNS 會傳送通知到您在 `accountCreationFailure deploy-config.yaml` 檔案中定義的電子郵件地址。帳戶請求者不會收到通知。	

（選用）清除解決方案

任務	Description	所需的技能
從 Amazon S3 儲存貯體移除物件。	移除下列 Amazon S3 儲存貯體中的任何物件： • `lza-account-creation-work-<CDK_UNIQUE_ID>` • `lza-account-creation-workflow-src-<AWS_REGION>-<AWS_ACCOUNT>` • `lza-account-creation-workflow-<AWS_REGION>-<AWS_ACCOUNT>`	AWS DevOps

任務	Description	所需的技能

刪除 CloudFormation 堆疊。	輸入下列命令來刪除 CloudFormation 堆疊：
	`aws cloudformation delete-stack \ --stack-name lza-account-creation-workflow-application`
	`aws cloudformation wait stack-delete-complete \ --stack-name lza-account-creation-workflow-application`
		AWS DevOps

刪除管道。	輸入下列命令來刪除`lza-account-creation-workflow-pipeline` 管道： `cdk destroy lza-account-creation-workflow-pipeline --force`	AWS DevOps

相關資源

- 上的登陸區域加速器 AWS (AWS 解決方案程式庫)
- 疑難排解 AWS CDK 常見問題 (AWS CDK 文件)

其他資訊

Step Functions 工作流程圖表

下圖顯示 Step Functions 工作流程中的狀態。

Arguments (引數)

以下是您在執行啟動 Step Functions 工作流程的 Python 指令碼時可以使用的引數。

下列是必要引數：

- `account-name (-a)`（字串）– 新 的名稱 AWS 帳戶。
- `support-dl (-s)`（字串）– 帳戶建立程序完成時收到通知的電子郵件地址。
- `managed-org-unit (-m)`（字串）– 將包含新帳戶的受管組織單位 (OU)。

下列引數為選用：

- `ad-integration (-ad)`（字串字典）– Microsoft Entra ID 群組和指派的許可集。以下是如何使用此引數的範例：

```
--ad-integration "{\"<PermissionSetName>\": \"<EntraIdGroupName>\"}"
```

- `account-email (-e)`（字串）– 新 根使用者的電子郵件地址 AWS 帳戶。

> **Note**
>
> 如果未使用此引數，則會使用 `rootEmailDomain` 檔案中的值 `rootEmailPrefix`和 產生電子郵件地址`configs/deploy-config.yaml`。如果未提供電子郵件地址，則會使用下列格式產生電子郵件地址：`rootEmailPrefix+accountName@rootEmailDomain`。

- `region (-r)`（字串）– 已部署 Step Functions 工作流程 AWS 區域 的 。預設值為 `us-east-1`。
- `force-update (-f)`（字串布林值）– 輸入 `true` 強制 AWS Service Catalog 更新佈建的產品。
- `bypass-creation (-b)`（字串布林值）– 輸入 `true` 以略過將帳戶新增至 `accounts-config.yaml` 檔案，並略過執行`AWSAccelerator-Pipeline`管道。此引數通常用於測試帳戶建立工作流程程序，或在`Landing Zone Accelerator`管道發生錯誤時執行其餘的 Step Functions 步驟。
- `tags (-t)`（字串）– 您要新增至 的其他標籤 AWS 帳戶。根據預設，會新增下列標籤：`account-name`、 `support-dl`和 `purpose`。以下是如何使用此引數的範例：

```
--tags TEST1=VALUE1 TEST2=VALUE2
```

跨多個帳戶和區域自動清查 AWS 資源

Matej Macek，Amazon Web Services

摘要

此模式概述自動化方法，以維護跨多個帳戶和 的完整 AWS 資源庫存 AWS 區域。它旨在協助基礎設施和安全工程師改善其資源管理實務。它使用 AWS Config 來追蹤資源變更、用於查詢的 Amazon Athena，以及用於互動式儀表板的 Amazon Quick Sight。您可以透過部署 AWS CloudFormation 堆疊來實作此解決方案。

此解決方案類似於使用 Amazon Athena 和 Amazon Quick Sight 視覺化 AWS Config 資料中所呈現的解決方案 (AWS 部落格文章）。此模式擴展了該解決方案，以解決下列常見需求，並提供下列主要優點：

- 以合規為重心 – 此方法可協助您符合法規要求，例如 PCI DSS、NIST SP 800-53、ISO/IEC 27001、HIPAA、GDPR 和其他要求正確資產庫存的法規要求。
- 自訂架構 – 它提供為各種 AWS 資源建立 Quick Sight 儀表板的基礎，讓您可以根據特定需求自訂解決方案。
- 使用者驅動的增強功能 – 此方法納入來自真實世界使用案例的意見回饋，並處理對更全面解決方案的請求。

基礎設施、安全和財務團隊通常會在動態、多帳戶或多區域環境中面臨可見性和協作挑戰。此解決方案旨在解決這些挑戰，並大幅減少建立和維護資源庫存所需的時間和精力。結果是資源的集中檢視，可協助您改善資源配置決策、識別和降低風險、最佳化成本，以及改善整體可見性和協同合作。此方法可彌補概念性解決方案與實際實作需求之間的差距，以達成安全性、合規性和營運目的。

先決條件和限制

先決條件

- 下列作用中 AWS 帳戶：
  - 管理帳戶 - 用於計費、建立帳戶和控制整個組織的存取的集中式帳戶
  - 稽核帳戶 – 用於安全監控、合規檢查和偏離通知的集中式中樞
  - 日誌封存帳戶 – 用於儲存和分析收集的資料的集中式帳戶
- 在稽核帳戶中， AWS Config 彙總工具會從目標帳戶和區域收集和彙總組態資料

- 在日誌封存帳戶中，設定下列項目：
  - Amazon Simple Storage Service (Amazon S3) AWS Config 儲存貯體，您可以在其中存放來自彙總工具的資料
  - Amazon Quick 訂閱
  - Quick Sight 與 Amazon Athena 之間的授權連線
  - 透過 Athena 查詢存取 Amazon S3 儲存貯體的許可
- AWS Command Line Interface (AWS CLI)，已安裝並設定
- 部署佈建下列資源之 CloudFormation 堆疊的許可：
  - AWS Lambda 函數
  - Amazon S3 通知組態
  - Athena 資料庫、資料表和檢視
  - Quick Sight 資料集和資料來源
- 在 中執行自動化的許可 AWS Systems Manager
- 快速存取的許可

限制

- 解決方案倚賴 AWS Config。 AWS Config 通常在偵測到變更後立即記錄資源的組態變更，或依您指定的頻率記錄。不過，這需要盡最大努力，有時可能需要更長的時間。
- 此解決方案只會追蹤 AWS Config 支援的資源類型。
- 解決方案不會追蹤其他雲端提供者或內部部署環境的資源庫存。
- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱 AWS 文件中的服務端點和配額頁面，然後選擇服務的連結。

Architecture

下圖顯示簡化的流程，用於收集、組織、分析和視覺化 AWS 組織中多個帳戶的組態和合規資料。

該圖顯示以下工作流程：

1. AWS Config 彙總器會定期收集目標帳戶和區域中資源的組態和合規資料，然後將資料交付至日誌封存帳戶中的 Amazon S3 儲存貯體。
2. 將新 AWS Config 資料新增至 Amazon S3 儲存貯體會叫用 AWS Lambda 函數。

3. Lambda 函數透過使用對應於每個快照檔案的區域和日期的值來設定索引鍵來分割資料。這有助於 AWS Glue 有效率地查詢和處理組態和合規資料。
4. Amazon Athena 使用 AWS Glue 結構描述，針對存放在 Amazon S3 儲存貯體中的資料執行 SQL 查詢。它利用來自 的結構描述中繼資料 AWS Glue 來了解資料的結構。
5. Athena 中的檢視會定義和擷取目標資料集。
6. Quick Sight 中的儀表板可協助您視覺化和分析資料集。

工具

AWS 服務

- Amazon Athena 是一種互動式查詢服務，可協助您使用標準 SQL 直接在 Amazon S3 中分析資料。
- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。
- AWS Config 提供 中資源的詳細檢視 AWS 帳戶 及其設定方式。它可協助您識別資源彼此之間的關係，以及其組態如何隨著時間而改變。 AWS Config 彙整工具會從多個 和 區域收集 AWS Config 組態 AWS 帳戶 和合規資料。
- AWS Glue 是一種全受管的擷取、轉換和載入 (ETL) 服務。它可協助您可靠地分類、清理、擴充和移動資料存放區和資料串流之間的資料。此模式使用 AWS Glue 資料目錄和結構描述登錄檔。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。
- Amazon Quick Sight 是一項商業智慧 (BI) 服務，可協助您透過互動式視覺化、儀表板和報告，將原始資料轉換為有意義的洞見。Quick Sight 是 Amazon Quick 的核心元件。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Systems Manager 可協助您管理在 中執行的應用程式和基礎設施 AWS 雲端。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。AWS Systems Manager 自動化可簡化許多 的常見維護、部署和修復任務 AWS 服務。

程式碼儲存庫

此模式的 AWS CloudFormation 範本可在AWS Config 視覺化 GitHub 儲存庫中使用。此 CloudFormation 範本會部署 AWS Systems Manager 自動化 Runbook，該 Runbook AWS Config 設

定為與 Amazon Athena 搭配使用。此自動化 AWS Glue 準備與指定的 Amazon S3 儲存貯體連線、在 Amazon Athena 中建立檢視，以及設定儀表板視覺化的 Quick Sight。

最佳實務

- 建議您遵循 AWS 規範指引中的設定和管理安全、多帳戶 AWS 環境 AWS Control Tower的最佳實務。
- 我們建議您建立 AWS Config 彙總工具，以收集整個 AWS 組織的組態和合規資料。如需詳細資訊，請參閱 AWS Config 文件中的多帳戶多區域資料彙總。
- 部署此解決方案之前，建議您檢閱 Amazon S3、、AWS ConfigAthena 和 Quick 的目前定價資訊。

史詩

部署 CloudFormation 堆疊

任務	Description	所需的技能
下載 CloudFormation 範本。	下載 Config-QuickSight-Visualization-SSM-Automation.yaml CloudFormation 範本。	AWS 管理員、雲端管理員、DevOps 工程師
修改 CloudFormation 範本。	只有在您使用 AWS Control Tower並由 AWS Config 管理時，才完成此步驟 AWS Control Tower。您需要修改 CloudFormation 範本。 1. 登入 管理帳戶。 2. 開啟 AWS Organizations 主控台。 3. 瀏覽至 Settings (設定) 頁面。此頁面會顯示組織的詳細資訊，包括組織 ID。 4. 複製組織 ID。 5. 在您偏好的文字編輯器中，開啟 Config-Qu	DevOps 工程師、AWS 管理員

任務	Description	所需的技能

	ickSight-Visualization-SSM-Automation.yaml 檔案。
	6. 尋找以下行：
	`return re.match( '^AWSLogs/(\d+)/Config/([\w-]+)/(\d+)/(\d+)/(\d+)/(\d+)/ConfigSnapshot/[^\\\]+$', object_key)`
	7. 將此行取代為以下內容，其中 <ORGANIZATION_ID> 是您先前複製的 ID：
	`return re.match( '^<ORGANIZATION_ID>/AWSLogs/(\d+)/Config/([\w-]+)/(\d+)/(\d+)/(\d+)/(\d+)/ConfigSnapshot/[^\\\]+$', object_key)`
	8. 儲存並關閉 Config-QuickSight-Visualization-SSM-Automation.yaml 檔案。	

任務	Description	所需的技能
建立 CloudFormation 堆疊。	遵循從 CloudFormation 主控台建立堆疊中的指示。注意下列事項： 1. 選擇 上傳範本檔案，然後選擇您下載的 YAML 檔案。 2. 針對堆疊名稱 輸入 `Config-QuickSight-Visualization-SSM-Automation` 。 3. 選擇提交。	AWS 管理員、雲端管理員、DevOps 工程師

在 Systems Manager 中執行自動化

任務	Description	所需的技能
尋找您的快速使用者名稱。	1. 開啟 Quick 主控台。 2. 開啟設定檔選單。 3. 請記下使用者名稱。您稍後將需要這個值。	AWS 管理員、雲端管理員、DevOps 工程師
尋找交付管道名稱和 Amazon S3 儲存貯體名稱。	1. 在 中 AWS CLI，輸入下列命令： `aws configservice describe-delivery-channels` 2. 請記下 Amazon S3 儲存貯體名稱和AWS Config 交付管道的名稱。您稍後需要這些值。	AWS 管理員、雲端管理員、DevOps 工程師
在 Systems Manager 中執行自動化。	1. 開啟 AWS Systems Manager 主控台。	AWS 管理員、雲端管理員、DevOps 工程師

任務	Description	所需的技能
	2. 在導覽窗格中，選擇 Documents (文件)。 3. 選擇 Owned by me (自有)。 4. 選擇 Config-QuickSight-Visualization。 5. 選擇 Execute automation (執行自動化)。 6. 在輸入參數區段中，輸入下列參數的值： • `ConfigDeliveryChannelName` – 輸入您的 AWS Config 交付管道名稱。此為必要參數。 • `ConfigS3BucketLocation` – 輸入您存放 AWS Config 組態資料的 Amazon S3 儲存貯體名稱。此為必要參數。 • `QuickSightUserName` – 輸入具有 Quick 管理存取權的使用者名稱。此為必要參數。 • `AutomationAssumeRole` – (IAM) 角色的 Amazon Resource Name AWS Identity and Access Management (ARN)，允許 Systems Manager Automation 代表您執行動作。此為選用參數。將此參數保留空白。	

任務	Description	所需的技能
	• `DeleteConfigVisualization` – 選擇 `false`。 7. 選擇 Execute (執行)。	

在 Quick Sight 中視覺化資料

任務	Description	所需的技能
重新整理資料。	若要根據您的特定需求排程資料集重新整理，請遵循重新整理 SPICE 資料中的指示。	AWS 管理員、DevOps 工程師、雲端管理員
建立 分析。	若要在 Quick Sight 中建立可協助您視覺化資源的儀表板，請遵循在 Quick Sight 中開始分析中的指示。	Quick Suite 管理員
建立儀表板。	1. 修改 Quick Sight 分析完成後，請遵循發佈儀表板中的指示來建立儀表板。儀表板是您可以與其他 Quick 使用者共用的分析。 2. 遵循授予儀表板存取權中的指示，與您的目標 Quick 使用者共用儀表板。	Quick Suite 管理員

(選用) 清除

任務	Description	所需的技能
刪除 Systems Manager 自動化建立的資源。	1. 開啟 AWS Systems Manager 主控台。	AWS 管理員、雲端管理員、DevOps 工程師

任務	Description	所需的技能
	2. 在導覽窗格中，選擇 Documents (文件)。 3. 選擇 Owned by me (自有)。 4. 選擇 Config-QuickSight-Visualization。 5. 選擇 Execute automation (執行自動化)。 6. 在輸入參數區段中，針對 `DeleteConfigVisualization` 參數輸入 `true`。 7. 選擇 Execute (執行)。	
刪除 CloudFormation 堆疊。	若要刪除`Config-QuickSight-Visualization-SSM-Automation` 堆疊中的資源，請遵循從 CloudFormation 主控台刪除堆疊中的指示。	AWS 管理員、雲端管理員、DevOps 工程師

疑難排解

問題	解決方案
Amazon Quick 正在嘗試連線至 `us-east-1` AWS 區域，但不允許在該區域中建立資源。	服務控制政策限制您在此區域中訂閱 Amazon Quick。在服務控制政策中，手動指定目標 AWS 區域。`<REGION_ID>` 將 取代為適當的區域識別符： `https://<REGION_ID>.quicksight.aws.amazon.com/sn/start/dashboards` 以下是範例：

問題	解決方案
	`https://eu-central-1.quicksight.aws.amazon.com/sn/start/dashboards`
在 Amazon Athena 中，您遇到下列訊息： `Before you run your first query, you need to set up a query result location in Amazon S3.`	請確定您已準備好要存放 Amazon Athena 查詢結果的 Amazon S3 儲存貯體。 Amazon Athena 然後遵循使用 Amazon Athena 主控台指定查詢結果位置中的指示。

相關資源

AWS 文件

- AWS Config 文件
- Amazon Quick 文件

AWS 部落格文章

- 使用 自動化 AWS Config 資料視覺化 AWS Systems Manager
- 如何使用 定期記錄資源組態變更 AWS Config

其他資源

- Amazon Quick Community Learning Center
- Amazon Quick Community Gallery

建置包含 MongoDB Atlas 的 AWS 登陸區域

Igor Alekseev，Amazon Web Services

Anuj Panchal，MongoDB

總結

此模式說明如何建置與 MongoDB Atlas 叢集整合的 AWS 登陸區域。基礎設施會使用 Terraform 指令碼自動部署。

結構良好的多帳戶 AWS 環境稱為登陸區域，可提供可擴展性和安全性，特別是企業。它可作為工作負載和應用程式快速部署的基礎，並有助於確保對安全性和基礎設施的信心。建置登陸區域需要仔細考慮技術和商業因素，包括帳戶結構、聯網、安全和存取管理。這些考量事項應與組織的未來成長和業務目標保持一致。

此模式的使用案例包括下列項目。

- 企業 SaaS 和 PaaS 平台：在 上執行的多租戶軟體即服務 (SaaS) 應用程式和平台即服務 (PaaS) 平台 AWS 可以使用此設定，協助提供對 MongoDB Atlas 的安全私有存取，而無需透過公有網際網路公開資料。
- 高度監管的產業：銀行、金融服務、醫療保健和政府工作負載需要嚴格遵守健康保險流通與責任法案 (HIPAA)、支付卡產業資料安全標準 (PCI DSS)、系統和組織控制 2 (SOC2) 和一般資料保護法規 (GDPR) 等標準，受益於：
  - 透過 加密的私有連線 AWS PrivateLink
  - MongoDB 複本集的異地同步備份高可用性
- 安全的 AI/ML 工作負載：Amazon Bedrock、Amazon SageMaker AI 或自訂 AI 模型中的訓練或推論管道，可以透過 PrivateLink 安全地擷取和存放 MongoDB Atlas 中的資料。
- 災難復原和業務持續性：多可用區設計可確保單一可用區域故障不會中斷工作負載。跨可用區域的 Atlas 複本集可確保自動容錯移轉。這對金融技術 （ 金融技術 ） 應用程式、數位銀行或醫療保健監控等全年無休的服務至關重要。

先決條件和限制

先決條件

- 組織擁有者可存取 MongoDB Atlas，因此您可以建立 Atlas API 金鑰。如需此需求的相關資訊，請參閱 MongoDB 文件中的管理組織存取。

- 作用中的 AWS 帳戶。
- Terraform，已安裝並設定。
- 使用 MongoDB 6.0 版或更新版本建立的 MongoDB Atlas 叢集。
- 熟悉 MongoDB 和 MongoDB Atlas。如需詳細資訊，請參閱 MongoDB Atlas 文件。

限制

- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

下列參考架構圖說明與 MongoDB Atlas 私有端點整合之 AWS 登陸區域的部署設定。此參考架構示範如何建立與 MongoDB Atlas 整合的安全、可擴展且高可用性的 AWS 登陸區域。透過結合多可用區部署、最低權限安全控制和私有連線等 AWS 最佳實務，此設計可讓組織為現代應用程式佈建強大的環境。

此架構包含下列項目：

VPC

- 單一虛擬私有雲端 (VPC) 跨越三個可用區域。
- VPC 會細分為與每個可用區域對齊的子網路。這些子網路會分配工作負載以獲得高可用性。

網際網路存取

- 網際網路閘道可為所需的資源提供傳出網際網路連線，例如應用程式或堡壘主機。
- 公有子網路可以容納 NAT 閘道，允許私有子網路工作負載下載更新、修補程式和其他必要的套件，而不會直接公開到公有網際網路。

私有子網路和路由表

- 應用程式元件、微服務或其他敏感資源通常位於私有子網路中。
- 專用路由表控制流量流程。將來自私有子網路的傳出流量路由到 NAT 閘道，以實現安全、僅限輸出的網際網路存取。

- 來自網際網路的傳入請求會流經公有子網路中的彈性負載平衡器或堡壘主機（如果使用的話），然後適當地路由至私有子網路資源。

透過 PrivateLink 的 MongoDB Atlas 連線

- 此架構使用 PrivateLink（透過 VPC 端點）安全地連線至 MongoDB Atlas，而不會將您的資料暴露至公有網際網路。
- 請求會保留在 AWS 骨幹網路上。傳輸中的資料受益於 PrivateLink 加密，且絕不會透過公有網際網路路由。
- MongoDB Atlas 專用 VPC 託管您的主要和次要節點，並為受管資料庫叢集提供安全、隔離的環境。

多可用區域部署

- 關鍵基礎設施元件（例如 NAT 閘道和應用程式子網路）會分佈在至少三個可用區域。如果可用區域發生中斷，此架構可確保剩餘可用區域中的工作負載保持運作。
- 根據預設，MongoDB Atlas 透過複本集提供高可用性，並確保您的資料庫層保持容錯能力。關鍵基礎設施會分散到至少三個可用區域，以提供彈性。

工具

AWS 服務

- AWS Secrets Manager 可協助您以 API 呼叫取代程式碼中的硬式編碼登入資料，包括密碼，以程式設計方式擷取秘密。

其他產品和工具

- MongoDB Atlas 是全受管資料庫即服務 (DbaaS)，用於在雲端中部署和管理 MongoDB 資料庫。
- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。在此模式中，您會使用 Terraform 執行指令碼，以協助在 AWS 和 MongoDB Atlas 上部署所需的資源。

程式碼儲存庫

此模式的程式碼可在 AWS 和 MongoDB Atlas 登陸區域 GitHub 儲存庫中使用。

史詩

完成探索和評估

任務	Description	所需的技能
識別關鍵利益相關者。	識別參與登陸區域專案的所有主要利益相關者和團隊成員。這可能包括下列角色： • 資料庫管理員 (DBAs) • DevOps 工程師 • 應用程式開發人員 • 應用程式架構師	遷移潛在客戶
建立結構藍圖。	建立藍圖，概述 AWS 和啟用 MongoDB Atlas 的登陸區域所需的結構。	遷移潛在客戶
建立架構計劃。	與您的應用程式架構師合作，以分析需求並設計容錯且具彈性的架構。此模式提供入門架構範本供您參考。您可以自訂此範本，以滿足組織的安全和基礎設施需求。	雲端架構師
規劃設定和部署。	與所有利益相關者一起決定如何部署架構、如何實作安全措施，以及任何其他層面，以確保與組織和請求團隊的利益保持一致。	Migration Lead，DevOps 工程師，DBA

設定 MongoDB Atlas 環境

任務	Description	所需的技能
複製儲存庫。	執行 命令，從 GitHub 儲存庫複製程式碼： `git clone https://github.com/mongodb-partners/AWS-MongoDB-Atlas-Landing-Zone`	應用程式開發人員、DevOps 工程師
取得您的 Atlas 組織 ID。	1. 如果您沒有 MongoDB Atlas 帳戶，請註冊一個帳戶。 2. 遵循 MongoDB 文件中的步驟來建立組織。 3. 複製組織 ID。	DBA
產生 Atlas 組織層級 API 金鑰。	若要在 Atlas 中產生組織層級 API 金鑰，請遵循 MongoDB 文件中的指示。	DBA
在 中建立秘密 AWS Secrets Manager。	將上一個步驟中產生的 MongoDB Atlas API 金鑰儲存為 Secrets Manager 中的金鑰值秘密。如需說明，請參閱 Secrets Manager 文件。	DevOps 工程師
選取 Atlas 叢集層。	若要選取正確的 Atlas 叢集層，請遵循 MongoDB 文件中的指示。	DBA

設定 AWS 環境

任務	Description	所需的技能
修改 Terraform 指令碼。	在 GitHub 儲存庫的本機副本中，更新 module/mongodb-atlas/main.tf 檔案（第 12 行）中的秘密名稱，因此 Terraform 可以在部署期間從 Secrets Manager 擷取登入資料。	DevOps 工程師
建立 AWS 存取金鑰 ID 和私密金鑰。	若要建立您的 AWS 存取金鑰 ID 和私密金鑰，請遵循 AWS re：Post 文章中的指示如何建立 AWS 存取金鑰？ 最佳實務是指派具有所需最低權限的政策，但在此案例中，請選取`AdministratorAccess` 政策。 建立存取金鑰後，請檢閱 IAM 中的安全最佳實務，以了解管理存取金鑰的最佳實務。	DevOps 工程師
配置彈性 IP 地址。	配置至少兩個彈性 IP IDs。如需說明，請參閱 Amazon Virtual Private Cloud (Amazon VPC) 文件。	DevOps 工程師
建立 S3 儲存貯體。	遵循 Amazon Simple Storage Service (Amazon S3) 文件中的指示，建立 S3 儲存貯體以存放 Terraform 部署的狀態。Amazon S3	DevOps 工程師
更新 S3 儲存貯體以進行儲存。	更新本機版本 environments/development/main.tf 中的 S3	DevOps 工程師

任務	Description	所需的技能
	儲存貯體資訊，以符合您在上一個步驟中建立的儲存貯體名稱和區域，並指定金鑰字首。例如： `terraform { ... backend "s3" { bucket = "startup-name-product-terraform" key = "network/dev" region = "ap-southeast-1" } }` 在此範例中，您可以將 Terraform 設定為使用金鑰字首`network/dev` 來組織 Terraform 狀態檔案。您可以將值變更為 `prod`或 `staging`，以符合您要建立的環境。如需使用多個環境的資訊，請參閱本節中的最後一個步驟。 如需 Amazon S3 金鑰字首的詳細資訊，請參閱 Amazon S3 文件中的使用字首組織物件。	

任務	Description	所需的技能
設定 Terraform 變數。	範例登陸區域使用 Terraform 變數定義檔案來定義輸入變數值。 變數檔案位於 environments/development/variables.tf。您可以在 environments/development/terraform.tfvars 檔案中設定變數值。如 GitHub 儲存庫的讀我檔案所述設定這些變數。	DevOps 工程師
設定環境變數。	如果您打算在本機電腦上執行 Terraform 指令碼，請設定下列環境變數： • `AWS_ACCESS_KEY_ID` ：AWS 存取金鑰 ID • `AWS_SECRET_ACCESS_KEY` ：AWS secret 存取金鑰 • `AWS_DEFAULT_REGION` ：AWS 區域 • `TF_LOG`：Terraform 日誌層級 (`DEBUG` 或 `INFO`) 如需設定環境變數的詳細資訊，請參閱 AWS Command Line Interface (AWS CLI) 文件。	DevOps 工程師

任務	Description	所需的技能
檢查 VPC 組態。	若要遵循 建議的最佳實務 AWS，請在 Terraform 指令碼中設定 VPC 和子網路 CIDRs、NAT 閘道、路由和路由表的設定，以符合組織的需求。如需詳細資訊，請參閱 GitHub 儲存庫的讀我檔案。	DevOps 工程師
標記 資源。	您可以標記 AWS 資源，以便在 Terraform 指令碼部署它們時對其進行監控。如需範例，請參閱 GitHub 儲存庫的讀我檔案。如需透過標籤監控資源的成本、用量等資訊，請參閱 AWS Billing 文件中的啟用使用者定義的成本分配標籤。	DevOps 工程師
使用多個環境。	GitHub 儲存庫提供`development` 環境資料夾。您也可以在環境資料夾中新增自己的環境。 若要新增環境，請將 `development` 資料夾複製到 下的新資料夾 （例如，`prod`或 `staging`)`environments` 。然後，您可以使用新值更新`terraform.tfvars` 檔案。	DevOps 工程師

部署登陸區域

任務	Description	所需的技能
初始化 Terraform 工作目錄。	若要初始化工作目錄並下載必要的套件，請執行 命令： `terraform init`	DevOps 工程師
建立執行計畫。	若要建立執行計畫並將 Terraform 對基礎設施所做的變更視覺化，請執行 命令： `terraform plan`	DevOps 工程師
部署變更。	若要如程式碼所述實作基礎設施的變更，請執行 命令： `terraform apply`	DevOps 工程師
驗證部署。	驗證 Terraform 在基礎設施中建立或修改的元件。 若要測試設定，請在 中佈建運算資源 （例如 Amazon EC2 執行個體或 AWS Lambda 函數） 或連接到 VPC。	DevOps 工程師，應用程式開發人員

移除資源

任務	Description	所需的技能
清除。	完成測試後，請執行下列命令來銷毀 Terraform 部署在基礎設施中的資源：	DevOps 工程師

任務	Description	所需的技能
	`terraform destroy`	

相關資源

探索和評估

- 登陸區域設定的管理秘訣 (AWS Control Tower 文件）
- 對登陸區域組態的期望 (AWS Control Tower 文件）
- 登陸區域更新的最佳實務 (AWS Control Tower 文件）

設定 MongoDB Atlas 和 AWS 環境

- 取得 MongoDB Atlas (AWS Marketplace)
- 記憶體 (MongoDB Atlas 文件）
- 使用 Atlas 範例資料集調整大小範例 (MongoDB Atlas 文件）
- 行動應用程式的大小範例 (MongoDB Atlas 文件）
- 網路流量 (MongoDB Atlas 文件）
- 叢集自動擴展 (MongoDB Atlas 文件）
- Atlas 大小調整範本 (MongoDB Atlas 文件）
- 設定網路對等互連 (MongoDB Atlas 文件）
- Atlas 中的私有端點 (MongoDB Atlas 文件）
- 用戶端欄位層級加密 (MongoDB 資料庫文件）
- 自動加密 (MongoDB 資料庫文件）
- 選取叢集層 (MongoDB Atlas 文件）

部署登陸區域

- 上的 Terraform AWS （白皮書上適用於 5G 網路的 CI/CD AWS)
- MongoDB Atlas 搭配 Terraform (MongoDB 文件）

設定 VPC 流程日誌以進行跨 的集中化 AWS 帳戶

Benjamin Morris 和 Aman Kaur Gandhi，Amazon Web Services

總結

在 AWS 虛擬私有雲端 (VPC) 中，VPC 流程日誌功能可以提供有用的資料，以進行操作和安全性疑難排解。不過，在多帳戶環境中使用 VPC 流程日誌會有限制。具體而言，不支援來自 Amazon CloudWatch Logs 的跨帳戶流程日誌。反之，您可以使用適當的儲存貯體政策設定 Amazon Simple Storage Service (Amazon S3) 儲存貯體，以集中管理日誌。

> **Note**
>
> 此模式討論將流程日誌傳送至集中位置的需求。不過，如果您也希望日誌可在成員帳戶中於本機使用，您可以為每個 VPC 建立多個流程日誌。無法存取 Log Archive 帳戶的使用者可以看到流量日誌以進行疑難排解。或者，您可以為將日誌傳送至 CloudWatch Logs 的每個 VPC 設定單一流程日誌。然後，您可以使用 Amazon Data Firehose 訂閱篩選條件，將日誌轉送至 S3 儲存貯體。如需詳細資訊，請參閱相關資源一節。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 具有 帳戶 AWS Organizations 的組織，用於集中日誌 （ 例如，Log Archive)

限制

如果您使用 AWS Key Management Service (AWS KMS) 受管金鑰`aws/s3`來加密您的中央儲存貯體，則不會收到來自不同帳戶的日誌。反之，您會看到`Unsuccessful`錯誤代碼 400，其中包含訊息，例如指定 `"LogDestination: <bucketName> is undeliverable"` 的 `ResourceId`。這是因為帳戶的 AWS 受管金鑰無法跨帳戶共用。解決方案是使用 Amazon S3 受管加密 (SSE-S3) 或可與成員帳戶共用 AWS KMS 的客戶受管金鑰。

Architecture

目標架構

在下圖中，每個 VPC 部署了兩個流程日誌。一個 會將日誌傳送至本機 CloudWatch Logs 群組。另一個 會將日誌傳送至集中式日誌帳戶中的 S3 儲存貯體。儲存貯體政策允許日誌交付服務將日誌寫入儲存貯體。

> **Note**
>
> 截至 2023 年 11 月， AWS 現在支援 aws：SourceOrgID 條件金鑰。此條件可讓您拒絕寫入 AWS Organizations 組織外部帳戶的集中式儲存貯體。

自動化和擴展

每個 VPC 都設定為將日誌傳送至中央記錄帳戶中的 S3 儲存貯體。使用下列其中一個自動化解決方案，以協助確保流程日誌設定正確：

- CloudFormation StackSets
- AWS Control Tower Terraform 帳戶工廠 (AFT)
- 具有修復的 AWS Config 規則

工具

工具

- Amazon CloudWatch Logs 可協助您集中所有系統、應用程式的日誌， AWS 服務 以便您可以監控日誌並將其安全地存檔。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。此模式使用 VPC 流程日誌功能來擷取進出 VPC 網路介面之 IP 流量的相關資訊。

最佳實務

使用基礎設施做為程式碼 (IaC) 可大幅簡化 VPC 流程日誌部署程序。抽象 VPC 部署定義以包含流程日誌資源建構，會自動使用流程日誌部署您的 VPCs。這會在下一節中示範。

集中式流程日誌

將集中式流程日誌新增至 HashiCorp Terraform 中 VPC 模組的範例語法：此程式碼會建立流程日誌，將日誌從 VPC 傳送至集中式 S3 儲存貯體。請注意，此模式不包含 S3 儲存貯體的建立。如需建議的儲存貯體政策陳述式，請參閱其他資訊一節。

```
variable "vpc_id" { type = string }
locals { custom_log_format_v5 = "$${version} $${account-id} $${interface-id} $
${srcaddr} $${dstaddr} $${srcport} $${dstport} $${protocol} $${packets} $${bytes}
 $${start} $${end} $${action} $${log-status} $${vpc-id} $${subnet-id} $${instance-
id} $${tcp-flags} $${type} $${pkt-srcaddr} $${pkt-dstaddr} $${region} $${az-id} $
${sublocation-type} $${sublocation-id} $${pkt-src-aws-service} $${pkt-dst-aws-service}
 $${flow-direction} $${traffic-path}" }
resource "aws_flow_log" "centralized_flow_log" {
  log_destination      = "arn:aws:s3:::centralized-vpc-flow-logs-
<log_archive_account_id>" # Optionally, a prefix can be added after the ARN.
  log_destination_type = "s3"
  traffic_type         = "ALL"
  vpc_id               = var.vpc_id
  log_format           = local.custom_log_format_v5 # If you want fields from VPC Flow
 Logs v3+, you will need to create a custom log format.
}
```

如需自訂日誌格式的詳細資訊，請參閱 Amazon VPC 文件。

本機流程日誌

將本機流程日誌新增至 Terraform 中具有必要許可的 VPC 模組的範例語法：此程式碼會建立流程日誌，將日誌從 VPC 傳送至本機 CloudWatch Logs 群組。

```
data "aws_region" "current" {}
variable "vpc_id" { type = string }
resource "aws_iam_role" "local_flow_log_role" {
  name = "flow-logs-policy-${var.vpc_id}"
  assume_role_policy = <<EOF
{
  "Version": "2012-10-17",
  "Statement": [{
      "Effect": "Allow",
      "Principal": {"Service": "vpc-flow-logs.amazonaws.com"},
      "Action": "sts:AssumeRole"
  }]
}
```

```
EOF
}
resource "aws_iam_role_policy" "logs_permissions" {
  name = "flow-logs-policy-${var.vpc_id}"
  role = aws_iam_role.local_flow_log_role.id
  policy = <<EOF
{
  "Version": "2012-10-17",
  "Statement": [{
      "Action": ["logs:CreateLog*", "logs:PutLogEvents", "logs:DescribeLog*",
 "logs:DeleteLogDelivery"],
      "Effect": "Allow",
      "Resource": "arn:aws:logs:${data.aws_region.current.name}:*:log-group:vpc-flow-
logs*"
  }]
}
EOF
}
resource "aws_cloudwatch_log_group" "local_flow_logs" {
  name              = "vpc-flow-logs/${var.vpc_id}"
  retention_in_days = 30
}
resource "aws_flow_log" "local_flow_log" {
  iam_role_arn    = aws_iam_role.local_flow_log_role.arn
  log_destination = aws_cloudwatch_log_group.local_flow_logs.arn
  traffic_type    = "ALL"
  vpc_id          = var.vpc_id
}
```

史詩

部署 VPC 流程日誌基礎設施

任務	Description	所需的技能
決定加密策略，並建立中央 S3 儲存貯體的政策。	中央儲存貯體不支援 AWS KMS aws/s3金鑰，因此您必須使用 SSE-S3 或 AWS KMS 客戶受管金鑰。如果您使用 AWS KMS 金鑰，金鑰政策必須允許成員帳戶使用 金鑰。	合規

任務	Description	所需的技能
建立中央流程日誌儲存貯體。	建立流程日誌將傳送到的中央儲存貯體，並套用您在上一個步驟中選擇的加密策略。這應該位於 Log Archive 或類似用途的帳戶中。 從其他資訊區段取得儲存貯體政策，並在使用環境特定值更新預留位置後將其套用至您的中央儲存貯體。	一般 AWS
設定 VPC 流程日誌將日誌傳送至中央流程日誌儲存貯體。	將流程日誌新增至您要從中收集資料的每個 VPC。最可擴展的方式是使用 AFT 或 等 IaC 工具 AWS Cloud Development Kit (AWS CDK)。例如，您可以建立 Terraform 模組，將 VPC 與流程日誌一起部署。如有必要，您可以手動新增流程日誌。	網路管理員
設定要傳送至本機 CloudWatch Logs 的 VPC 流程日誌。	（選用）如果您希望流程日誌顯示在產生日誌的帳戶中，請建立另一個流程日誌，將資料傳送至本機帳戶中的 CloudWatch Logs。或者，您可以將資料傳送至本機帳戶中的帳戶特定 S3 儲存貯體。	一般 AWS

相關資源

- 如何使用集中式流程日誌資料來促進資料分析和滿足安全需求 (AWS 部落格文章）
- 如何使用 AWS Config 規則自動啟用 VPC 流程日誌 (AWS 部落格文章）

其他資訊

儲存貯體政策

在您新增預留位置名稱的值之後，此儲存貯體政策範例可以套用至流量日誌的中央 S3 儲存貯體。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSLogDeliveryWrite",
            "Effect": "Allow",
            "Principal": {
                "Service": "delivery.logs.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::<BUCKET_NAME>/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:SourceOrgID": "<ORG_ID>"
                }
            }
        },
        {
            "Sid": "AWSLogDeliveryCheck",
            "Effect": "Allow",
            "Principal": {
                "Service": "delivery.logs.amazonaws.com"
            },
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::<BUCKET_NAME>",
            "Condition": {
                "StringEquals": {
                    "aws:SourceOrgID": "<ORG_ID>"
                }
            }
        },
        {
            "Sid": "DenyUnencryptedTraffic",
            "Effect": "Deny",
            "Principal": {
                "AWS": "*"
```

```
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::<BUCKET_NAME>/*",
                "arn:aws:s3:::<BUCKET_NAME>"
            ],
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "false"
                }
            }
        }
    ]
}
```

使用 Terraform 動態管理 AWS 許可集

Vinicius Elias 和 Marcos Vinicius Pinto Jordao，Amazon Web Services

摘要

AWS IAM Identity Center enhances AWS Identity and Access Management (IAM) 提供集中式中樞，用於管理 AWS 帳戶 和雲端應用程式的單一登入存取。不過，隨著組織的成長，手動管理 IAM Identity Center 許可集可能會變得越來越複雜且容易出錯。這種複雜性可能會導致潛在的安全漏洞和管理開銷。

此解決方案可讓您使用以原生建置的持續整合和持續交付 (CI/CD) 管道，透過基礎設施將許可集管理為程式碼 (IaC) AWS 服務。它可讓許可集指派機制與 AWS Control Tower 生命週期事件或 Account Factory for Terraform (AFT) 環境無縫整合。此方法為新的和現有的 提供動態身分組態 AWS 帳戶。

Amazon EventBridge 規則會監控 AWS 帳戶 建立和更新，這有助於您的身分組態與您的組織結構保持同步。在 或 AFT 中建立 AWS Control Tower 或更新帳戶之後，就會觸發管道。它使用許可集定義和指派規則來評估一組 JSON 檔案。然後，管道會套用並同步所有帳戶的設定。

此方法提供下列優勢：

- 一致性 – 消除整個 AWS 組織的手動組態偏離
- 可稽核性 – 維護所有身分管理變更的完整歷史記錄
- 可擴展性 – 隨著 AWS 環境的成長，自動套用組態
- 安全性 – 減少許可指派中的人為錯誤
- 合規 – 透過記錄的變更和指派規則，協助滿足法規要求

先決條件和限制

- 具有 AWS Control Tower 和 AWS Organizations 設定的多帳戶環境。或者，您可以搭配 使用 AFT AWS Control Tower。
- AWS 帳戶 接收解決方案的 IAM Identity Center 委派管理員。如需詳細資訊，請參閱 IAM Identity Center 文件中的委派管理。
- 處理主要程式碼的版本控制系統 (VCS) 儲存庫。如需範例，請參閱解決方案的 GitHub 儲存庫。
- Terraform 後端管理的必要 AWS 資源，例如 Amazon Simple Storage Service (Amazon S3) 儲存貯體和 Amazon DynamoDB 資料表。

限制

- 管道使用 AWS 原生資源和開放原始碼 Terraform。管道尚未準備好呼叫第三方生態系統。
- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

下圖顯示此模式的元件和工作流程。

AWS Control Tower 事件流程

解決方案從整合來自 AWS Control Tower 或 AFT 的事件開始。在實作時間，會透過變數定義，選擇一項或另一項服務。無論使用何種方法，只要建立或更新帳戶，就會觸發管道。管道會協調存放在許可集管理儲存庫中的政策。

以下是 AWS Control Tower 生命週期事件：

- `CreateManagedAccount` – 建立新帳戶時
- `UpdateManagedAccount` – 更新現有帳戶時

事件路由

EventBridge 做為中央事件處理服務，擷取 AWS Control Tower 帳戶中產生的事件。事件發生時，EventBridge 會以智慧方式將事件路由到解決方案帳戶中的集中式事件匯流排。 AWS Control Tower 生命週期事件遵循不同的路由模式。如果 AFT 定義為事件來源，則 AFT 管理帳戶會處理事件，而不是 AWS Control Tower 帳戶。此事件驅動型架構可自動回應組織變更，無需手動介入。

AFT 整合程序

當 AWS Control Tower 生命週期事件到達 AFT 管理帳戶時，它們會自動觸發多個 AFT 內部的下游程序。AFT 帳戶自訂工作流程完成後，它會將訊息發佈至專用 `aft-notifications` Amazon Simple Notification Service (Amazon SNS) 主題。該主題會觸發此解決方案實作的 `aft-new-account-forward-event` AWS Lambda 函數。Lambda 函數會將事件傳送至解決方案帳戶事件匯流排，用於啟動管道。

基礎設施即程式碼管道

解決方案管道以全自動化的部署機制運作。 AWS CodePipeline 服務會持續監控儲存庫的變更。偵測到新的遞交時，會自動啟動部署工作流程，並啟動包含驗證和執行階段的循序程序。系統會執行 Terraform `plan`操作來識別提議的變更，接著執行 Terraform `apply`命令以在 AWS 環境中實作這些變更。值得注意的是，管道在沒有任何手動核准閘道的情況下執行。此方法可讓您快速部署基礎設施變更，同時透過管道日誌和 Terraform 狀態檔案維持可稽核性。

管道會利用 AWS CodeBuild ，在具有適當許可的受控環境中執行 Terraform 操作。透過此 IaC 方法，管道可以執行全面的許可管理操作，包括：

- 建立新的許可集。
- 更新現有的許可集。
- 移除不必要的許可集。
- 管理組織中帳戶和群組之間這些許可的指派 AWS 。

為了維持基礎設施一致性並防止發生衝突的變更，解決方案會使用 Amazon S3 儲存貯體和專用 Amazon DynamoDB 資料表實作 Terraform 後端狀態管理系統。此方法為 Terraform 狀態檔案和狀態鎖定機制提供持久性儲存位置，以防止同時修改相同的資源。

主要 Terraform 程式碼使用官方 AWS `permission-sets` Terraform 模組。此模組可以根據許可集範本，在 IAM Identity Center 中動態管理許可集。

來源控制管理

許可集範本 (JSON 檔案） 位於外部版本控制系統中，例如 GitHub，為身分管理組態提供集中式儲存庫。此方法為許可集定義建立單一事實來源，同時透過標準程式碼檢閱實務進行協作開發。授權使用者可以在組織變更管理程序之後，將變更遞交至這些範本。這些遞交可做為自動化部署管道的主要觸發條件，啟動基礎設施更新程序。

如需如何使用儲存庫中的 JSON 檔案設定許可集的範例，請參閱其他資訊。

工具

AWS 服務

- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeConnections 可讓 CodePipeline 等 AWS 資源和服務連線至外部程式碼儲存庫，例如 GitHub。

- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 Shell 中的命令與 互動。
- AWS Control Tower 可協助您設定和管理 AWS 多帳戶環境，並遵循規範最佳實務。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如， AWS Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他事件匯流排 AWS 帳戶。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS IAM Identity Center 可協助您集中管理所有 AWS 帳戶 和雲端應用程式的單一登入 (SSO) 存取。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。它可啟用帳戶管理事件的推播通知，確保相關各方知道系統中的重要變更或動作。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他工具

- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。

程式碼儲存庫

此模式的程式碼可在 AWS samplesample-terraform-aws-permission-sets-pipeline儲存庫的 GitHub 上的 Samples 組織中取得。

最佳實務

- 一律鎖定用於在生產環境中執行程式碼的 Terraform 模組和提供者版本。
- 使用靜態程式碼分析工具，例如 Checkov，掃描您的程式碼，然後解決安全問題。
- 遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。

史詩

建立先決條件 （選用）

任務	Description	所需的技能

建立 Terraform 後端資源。	如果您尚未建立 Terraform 後端 AWS 資源，請使用下列步驟來建立 Amazon S3 儲存貯體 (`s3-tf-backend-{ACCOUNT_ID}`) 和 DynamoDB 資料表 ()`ddb-tf-backend` 。
	1. 登入 AWS 帳戶 您要部署解決方案的 ，並在AWS CloudShell AWS Control Tower 家中開啟 AWS 區域。
	2. 執行下列命令，以您的 AWS 帳戶 ID `{ACCOUNT_ID}` 取代預留位置：
	`aws s3api create-bucket --bucket s3-tf-backend-{ACCOUNT_ID}`
	`aws s3api put-bucket-versioning --bucket s3-tf-backend-{ACCOUNT_ID} --versioning-`
		AWS 管理員

任務	Description	所需的技能
	`configuration Status=Enabled` `aws dynamodb create-table --table-name ddb-tf-backend --attribute-definitions AttributeName=LockID,AttributeType=S --key-schema AttributeName=LockID,KeyType=HASH --provisioned-throughput ReadCapacityUnits=1,WriteCapacityUnits=1`	

任務	Description	所需的技能
建立跨帳戶角色。	您必須在 `event-source-account` Terraform AWS 提供者組態中提供跨帳戶 IAM 角色。如果您尚未建立此角色，請使用下列步驟來建立角色： 1. 登入將成為事件來源 AWS 帳戶 的 (AWS Control Tower 管理帳戶或 AFT 帳戶），然後開啟 AWS CloudShell。 2. 執行下列命令，將預留位置取代`{ACCOUNT_ID}` 為您用於此解決方案的 AWS 帳戶 ID，而非目前的 AWS 帳戶 ID： `aws iam create-role \ --role-name CrossAccountRole \ --assume-role-policy-document '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "AWS": "arn:aws:iam::{ACCOUNT_ID}:root" }, "Action": "sts:AssumeRole"`	AWS 管理員

任務	Description	所需的技能

	<pre> }
	        ]
	    }'</pre>
	1. 檢查命令輸出並複製角色 Amazon Resource Name (ARN)（例如 `arn:aws:iam::111122223333:role/CrossAccountRole`）。
	2. 若要將 IAM 政策連接至角色，請執行下列命令：
	
	<pre>aws iam attach-role-policy \
	    --role-name CrossAccountRole \
	    --policy-arn arn:aws:iam::aws:policy/AdministratorAccess</pre>
	此範例使用 AWS 受管 IAM 政策 AdministratorAccess。如果您願意，可以使用更具體的政策。	

準備許可集儲存庫

任務	Description	所需的技能
建立專用儲存庫。	此任務假設您使用的是 GitHub。建立專用儲存庫來存	DevOps 工程師

任務	Description	所需的技能
	放主要 Terraform 程式碼和許可集範本 JSON 檔案。	
準備許可集程式碼。	如需如何建構下列檔案的資訊，請參閱解決方案儲存庫上的範例程式碼： ├ – main.tf：// ├ – outputs.tf：// ├ – provider.jinja └ — 範本 複製內容、保留`providers.jinja` 值，並對其他檔案進行必要的調整。例如，將許可集範本檔案新增至 檔案，`templates` 或將`aws-ia/permission-sets/aws`模組版本釘選在 `main.tf` 檔案中。	DevOps 工程師
遞交您的變更。	遞交變更並推送至您先前建立的儲存庫。儲存儲存庫名稱及其 GitHub 組織，例如 `myorg/aws-ps-pipeline` 。	DevOps 工程師

準備部署程式碼

任務	Description	所需的技能
下載內容。	從解決方案儲存庫下載（複製）內容。	DevOps 工程師

任務	Description	所需的技能

履行變數。	建立`terraform.tfvars` 檔案並新增下列必要變數：
	• `repository_name` – 您先前建立的許可集儲存庫名稱
	• `branch_name` – 許可集儲存庫分支的名稱
	• `vcs_provider` – 您的 VCS 提供者
	• `account_lifecycle_events_source` – 觸發管道 (AWS Control Tower 或 AFT) 的事件來源
	`repository_name = "myorg/aws-ps-pipeline"`
	`branch_name = "main"`
	`vcs_provider = "github"`
	`account_lifecycle_events_source = "CT"`
	如需其他變數選項的詳細資訊，請參閱此模式的 GitHub 儲存庫中的 variables.tf 檔案。
		DevOps 工程師

任務	Description	所需的技能
調整 Terraform 後端組態。	在 `backend.tf` 檔案中，將預留位置取代為您自己的值。使用 AWS Control Tower 首頁 AWS 區域，並提供先前建立的 Amazon S3 儲存貯體和 DynamoDB 資料表的名稱。 `terraform { required_version = ">=1.6" backend "s3" { region = "{region}" bucket = "{bucket_name}" key = "terraform.tfstate" dynamodb_table = "{table_name}" encrypt = "true" } }` 如果您願意，可以使用自己的 Terraform 後端組態。	DevOps 工程師

任務	Description	所需的技能

調整 Terraform 提供者組態。	在 `providers.tf` 檔案中，將預留位置取代為您自己的資訊。使用 AWS Control Tower 主區域，並提供先前為`event-source-account` 提供者建立的跨帳戶 IAM 角色的 ARN。
	
	`provider "aws" {`
	`  region = "{region}"`
	`}`
	
	`provider "aws" {`
	`  alias  = "event-so`
	`urce-account"`
	`  region = "{region}"`
	`  assume_role {`
	`  role_arn = "{role_ar`
	`n}"`
	`  }`
	`}`
		DevOps 工程師

手動部署解決方案

任務	Description	所需的技能
選取 AWS 帳戶。	我們建議您在 IAM Identity Center 委派管理員帳戶中部署解決方案。不過，您也可以在 AWS Organizations 管理帳戶中部署它。 若要登入與 IAM Identity Center 執行個體位於相同區域中的所選帳戶，請使用 AWS CLI。請確定您使用的 IAM	AWS 管理員

任務	Description	所需的技能
	角色具有許可，可擔任先前步驟中為`event-source-account` 提供者指定的角色。此外，此角色必須能夠存取 Terraform 後端組態中使用的 AWS 資源。	
手動執行 Terraform。	若要初始化、規劃和套用組態，請依所示順序執行下列 Terraform 命令： 1. `terraform init` 2. `terraform plan` 3. `terraform apply`	DevOps 工程師
檢查部署結果。	在 IAM Identity Center 委派管理員帳戶中，檢查`aws-ps-pipeline` 管道是否已建立。同時檢查是否有處於待定狀態的 AWS CodeConnections 連線。	AWS DevOps

任務	Description	所需的技能
完成 CodeConnections 組態。	若要完成 CodeConnections 組態，請使用下列步驟： 1. 前往 CodeConnections，選取已建立的連線，然後選擇更新待定連線。 2. 提供您的 GitHub 登入資料、安裝新的 GitHub 應用程式，或選取可存取許可集儲存庫的現有應用程式，然後選擇連線。 管道現在應該可以存取許可集儲存庫。 如需詳細說明，請參閱開發人員工具主控台文件中的更新待定連線。	AWS DevOps

選擇管道執行流程以測試解決方案

任務	Description	所需的技能
依 AWS Control Tower 或 AFT 更新執行管道。	使用 AWS Control Tower 或 AFT 建立或變更帳戶後（取決於您選擇的生命週期事件類型），管道就會啟動。	AWS 管理員
變更程式碼以執行管道。	在您變更程式碼並將其遞交至`main`分支之後，管道就會啟動。	AWS DevOps

任務	Description	所需的技能
手動執行管道。	若要手動啟動管道，請使用 中的發行變更功能 AWS CodePipeline。	AWS DevOps

疑難排解

問題	解決方案
存取遭拒	確認您具有部署解決方案所需的許可。
CodeConnections 問題	• 檢查連線狀態是否為可用，而非待定。 • 確認已使用 GitHub 登入資料正確完成 CodeConnections 組態。如需詳細資訊，請參閱開發人員工具主控台文件中的更新待定連線。 • 確定 GitHub 應用程式具有許可集儲存庫的適當存取許可。
管道執行問題	• 檢查 Amazon CloudWatch logs是否有管道執行錯誤。 • 確定已正確設定 IAM Identity Center 委派。否則，讀取 AWS 帳戶 中繼資料時可能會導致管道故障。
許可集部署問題	• 驗證許可集範本檔案中的 JSON 語法是否正確。 • 驗證範本中參考的 IAM 受管政策是否存在。 • 檢查指派中指定的群組和使用者是否存在於 IAM Identity Center 中。

相關資源

AWS 服務 文件

- AWS IAM Identity Center 使用者指南
- AWS 帳戶 使用許可集管理 (IAM Identity Center 文件)

其他資源

- AWS 許可集模組 (Terraform)

其他資訊

具有範例許可集的 JSON 檔案

下列範例示範如何使用儲存庫中的 JSON 檔案來設定許可集：

```
{
  "Name": "ps-billing", // Permission set identifier
  "Comment": "Sample permission set for billing access", // Comment to document the
 purpose of the permission set
  "Description": "Billing access in AWS", // Detailed description
  "SessionDuration": "PT4H", // Session duration = 4 hours (ISO 8601 format)
  "ManagedPolicies": [ // List of AWS IAM managed policies
    "arn:aws:iam::aws:policy/job-function/Billing",
    "arn:aws:iam::aws:policy/job-function/SupportUser",
    "arn:aws:iam::aws:policy/AWSSupportAccess",
    "arn:aws:iam::aws:policy/job-function/ViewOnlyAccess"
  ],
  "CustomerPolicies": [], // References to IAM policies previously created
  "CustomPolicy": {}, // Inline IAM policy defined directly in the permission set
  "PermissionBoundary": {  // AWS or customer managed IAM policy to be used as boundary
    "ManagedPolicy": "",
    "CustomerPolicy": ""
  },
  "Assignments": [ // Define the assignment rules
    {
      "all_accounts": true, // Apply to ALL active AWS accounts in organization
      "principal": "G_BILLING_USERS", // Group/user name in Identity Center
      "type": "GROUP", // Can be "GROUP" or "USER"
      "account_id": [], // List of AWS account ID (empty since all_accounts=true)
      "account_ou": [], // List of AWS Organizational Unit IDs with target AWS accounts
      "account_tag": [] // List of tags (key:value) to match AWS Organization accounts
 tags
    }
```

```
  ]
}
```

如需詳細資訊，請參閱 Terraform 網站上的AWS 許可集模組文件中的 JSON 結構描述。

提示

- 您可以使用 Terraform 匯入區塊，將現有的許可集匯入解決方案。
- 您可以使用 AFT 在委派帳戶中實作 AWS 許可集管道。如需詳細資訊，請參閱 AFT 藍圖。

使用 AWS Organizations 自動標記 Transit Gateway 連接

Richard Milner-Watts、Haris Bin Ayub 和 John Capps，Amazon Web Services

總結

在 Amazon Web Services (AWS) 上，您可以使用 AWS Resource Access Manager 跨AWS Transit Gateway AWS 帳戶 邊界共用。不過，當您跨帳戶邊界建立 Transit Gateway 連接時，會建立沒有名稱標籤的連接。這可能會讓識別連接耗時。

此解決方案提供自動化機制，可收集由 管理之組織內帳戶的每個 Transit Gateway 連接的相關資訊AWS Organizations。程序包括從 Transit Gateway 路由表中查詢無類別網域間路由 (CIDR) 範圍。解決方案接著會將 形式的 Name 標籤<CIDR-range>-<AccountName>套用至擁有傳輸閘道之帳戶內的附件。

此解決方案可與解決方案程式庫中的 Serverless Transit Network Orchestrator 等 AWS 解決方案搭配使用。Serverless Transit Network Orchestrator 可大規模自動建立 Transit Gateway 附件。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 包含所有相關帳戶 AWS Organizations 的組織
- 在組織的根目錄下存取組織管理帳戶，以建立 required AWS Identity and Access Management (IAM) 角色
- 共用網路成員帳戶，其中包含與組織共用並具有附件的一或多個傳輸閘道

Architecture

的下列螢幕擷取畫面 AWS 管理主控台 顯示沒有相關聯名稱標籤的 Transit Gateway 附件範例，以及具有此解決方案產生之名稱標籤的兩個 Transit Gateway 附件範例。產生的名稱標籤結構為 <CIDR-range>-<AccountName>。

此解決方案使用 AWS CloudFormation 部署AWS Step Functions工作流程，管理所有已設定之 Transit Gateway Name 標籤的建立 AWS 區域。工作流程會叫用 AWS Lambda函數，以執行基礎任務。

解決方案從中取得帳戶名稱後 AWS Organizations，Step Functions 狀態機器會取得所有 Transit Gateway 連接 IDs。這些是由區域平行處理。此處理包括查詢每個附件的 CIDR 範圍。CIDR 範圍是透過搜尋區域內的 Transit Gateway 路由表來取得相符的 Transit Gateway 連接 ID。如果所有必要資訊都可用，解決方案會將名稱標籤套用至附件。解決方案不會覆寫任何現有的名稱標籤。

解決方案會按照由 Amazon EventBridge 事件控制的排程執行。事件會在 UTC 每天上午 6：00 啟動解決方案。

目標技術堆疊

- Amazon EventBridge
- AWS Lambda
- AWS Organizations
- AWS Transit Gateway
- Amazon Virtual Private Cloud (Amazon VPC)
- AWS X-Ray

目標架構

下圖顯示解決方案架構和工作流程。

1. 排程事件會啟動規則。
2. EventBridge 規則會啟動 Step Functions 狀態機器。
3. 狀態機器會叫用 `tgw-tagger-organizations-account-query` Lambda 函數。
4. `tgw-tagger-organizations-account-query` Lambda 函數會擔任組織管理帳戶中的角色。
5. `tgw-tagger-organizations-account-query` Lambda 函數會呼叫 Organizations API 來傳回 AWS 帳戶 中繼資料。
6. 狀態機器會叫用 `tgw-tagger-attachment-query` Lambda 函數。
7. 對於每個區域，狀態機器會並行叫用 `tgw-tagger-rtb-query` Lambda 函數來讀取每個附件的 CIDR 範圍。
8. 對於每個區域，狀態機器會平行叫用 `tgw-tagger-attachment-tagger`Lambda 函數。
9. 系統會為共用網路帳戶中的 Transit Gateway 附件建立名稱標籤。

自動化和擴展

解決方案會平行處理每個區域，以減少執行的總持續時間。

工具

AWS 服務

- AWS CloudFormation 透過將基礎設施視為程式碼，提供建立相關 AWS 和第三方資源集合模型、快速一致地佈建它們，以及在整個生命週期中管理它們的方法。
- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您執行的應用程式。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可用來將應用程式與來自各種來源的資料連線。EventBridge 會收到事件、環境變更的指標，並套用規則將事件路由至目標。規則會根據事件的結構、稱為事件模式或排程，將事件與目標配對。
- AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只會在需要時執行程式碼，並自動擴展，從每天幾個請求擴展到每秒數千個請求。您只需為使用的運算時間支付費用。程式碼未執行時無須付費。
- AWS Organizations 隨著資源的成長和擴展， 可協助您集中管理和控管您的環境 AWS 。使用 Organizations，您可以透過程式設計方式建立新的資源 AWS 帳戶 並配置資源、將 帳戶分組以組織您的工作流程、將政策套用到帳戶或群組以進行控管，以及為所有帳戶使用單一付款方式來簡化計費。
- AWS Step Functions 是一種低程式碼視覺化工作流程服務，用於協調 AWS 服務、自動化業務流程和建置無伺服器應用程式。工作流程會管理故障、重試、平行化、服務整合和可觀測性，讓開發人員可以專注於更高價值的商業邏輯。
- AWS Transit Gateway 透過中央中樞連接 VPCs 和內部部署網路。這可簡化您的網路，並結束複雜的互連關係。它充當雲端路由器，因此每個新連線只會進行一次。
- Amazon Virtual Private Cloud (Amazon VPC) 是一種在您定義的邏輯隔離虛擬網路中啟動 AWS 資源的服務。
- AWS X-Ray 會收集應用程式提供的請求相關資料，並提供可用來檢視、篩選和深入了解該資料的工具，以識別問題和最佳化的機會。

Code

此解決方案的原始程式碼可在 Transit Gateway Attachment Tagger GitHub 儲存庫中使用。儲存庫包含下列檔案：

- `tgw-attachment-tagger-main-stack.yaml` 會在共用網路帳戶中建立所有資源以支援此解決方案。

- tgw-attachment-tagger-organizations-stack.yaml在組織的管理帳戶中建立角色。

史詩

部署主要解決方案堆疊

任務	Description	所需的技能
收集必要的先決條件資訊。	若要設定從 Lambda 函數到 AWS Organizations API 的跨帳戶存取權，您需要組織管理帳戶的帳戶 ID。 **Note** 建立兩個 CloudFormation 堆疊的順序很重要。您必須先將資源部署到共用網路帳戶。在將資源部署到組織的管理帳戶中之前，共用網路帳戶中的角色必須已存在。如需詳細資訊，請參閱 AWS 文件。	DevOps 工程師
啟動主要解決方案堆疊的 CloudFormation 範本。	主要解決方案堆疊的範本將部署 IAM 角色、Step Functions 工作流程、Lambda 函數和 Amazon CloudWatch 事件。 開啟 AWS 管理主控台 共用網路帳戶的 ，然後開啟 ：&CFN 主控台。 使用tgw-attachment-tagger-main-stack.yaml 範本和下列值建立堆疊：	DevOps 工程師

任務	Description	所需的技能
	• 堆疊名稱 – tgw-attachment-tagger-main-stack • awsOrganizationsRootAccountId – 組織的管理帳戶的帳戶 ID • TGWRegions 參數 – AWS 區域 針對解決方案，以逗號分隔字串輸入 • TGWList 參數 – 要從解決方案中排除的傳輸閘道 IDs，以逗號分隔字串輸入 如需啟動 CloudFormation 堆疊的詳細資訊，請參閱 AWS 文件。	
確認解決方案已成功啟動。	等待 CloudFormation 堆疊達到 CREATE_COMPLETE 狀態。這應該需要不到一分鐘的時間。 開啟 Step Functions 主控台，並確認已建立名為 tgw-attachment-tagger-state-machine 的新狀態機器。	DevOps 工程師

部署 AWS Organizations 堆疊

任務	Description	所需的技能
收集必要的先決條件資訊。	若要設定從 Lambda 函數到 AWS Organizations API 的跨帳戶存取權，您需要共用網路帳戶的帳戶 ID。	DevOps 工程師

任務	Description	所需的技能
啟動 Organizations 堆疊的 CloudFormation 範本	AWS Organizations 堆疊的範本將在組織的管理帳戶中部署 IAM 角色。 存取組織管理帳戶的 AWS 主控台。然後開啟 CloudFormation 主控台。 使用`tgw-attachment-tagger-organizations-stack.yaml` 範本和下列值建立堆疊： • 堆疊名稱 – tgw-attachment-tagger-organizations-stack • NetworkingAccountId 參數 – 共用網路帳戶的帳戶 ID 對於其他堆疊建立選項，請使用預設值。	DevOps 工程師
確認解決方案已成功啟動。	等待 CloudFormation 堆疊達到 CREATE_COMPLETE 狀態。這應該需要不到一分鐘的時間。 開啟 AWS Identity and Access Management (IAM) 主控台，並確認已建立名為 tgw-attachment-tagger-organization-query-role 的新角色。	DevOps 工程師

驗證解決方案

任務	Description	所需的技能
執行狀態機器。	開啟共用網路帳戶的 Step Functions 主控台，然後在導覽窗格中選擇狀態機器。 選取狀態機器 tgw-attachment-tagger-state-machine，然後選擇開始執行。 由於解決方案不會使用此狀態機器的輸入，因此您可以使用預設值。 `{ "Comment": "Insert your JSON here" }` 選擇 Start Execution (開始執行)。	DevOps 工程師
觀看狀態機器直到完成。	在開啟的新頁面上，您可以觀看狀態機器執行。持續時間取決於要處理的 Transit Gateway 附件數量。 在此頁面上，您可以檢查狀態機器的每個步驟。您可以在狀態機器中檢視各種任務，並遵循 Lambda 函數的 CloudWatch 日誌連結。對於在映射中平行執行的任務，您可以使用索引下拉式清單來檢視每個區域的特定實作。	DevOps 工程師

任務	Description	所需的技能
驗證 Transit Gateway 連接標籤。	開啟共用網路帳戶的 VPC 主控台，然後選擇傳輸閘道附件。 在 主控台上，為符合條件的附件提供名稱標籤（附件會傳播到 Transit Gateway 路由表，而資源擁有者是組織的成員）。	DevOps 工程師
驗證 CloudWatch 事件啟動。	等待 CloudWatch 事件啟動。這是排程為 06：00 UTC。 然後開啟共用網路帳戶的 Step Functions 主控台，然後在導覽窗格中選擇狀態機器。 選取狀態機器 tgw-attachment-tagger-state-machine。確認解決方案在 UTC 的 06：00 執行。	DevOps 工程師

相關資源

- AWS Organizations
- AWS Resource Access Manager
- 無伺服器傳輸網路協調器
- 建立 IAM 角色
- 在 AWS CloudFormation 主控台上建立堆疊

更多模式

- AWS 帳戶 使用 AFT 自動化新 的 Amazon VPC IPAM IPv4 CIDR 配置
- 使用 Account Factory for Terraform 管理多個帳戶的許可集
- 使用 GitHub 動作根據 AWS CloudFormation 範本佈建 AWS Service Catalog 產品
- 透過部署角色販賣機解決方案來佈建最低權限的 IAM 角色

AI 與機器學習

主題

- 將儲存 AWS CodeCommit 庫與另一個帳戶中 AWS 帳戶 的 Amazon SageMaker AI Studio Classic 建立關聯
- 使用 Amazon Textract 從 PDF 檔案自動擷取內容
- 在 Amazon SageMaker AI Studio Lab 中將 DeepAR 用於時間序列，以建置冷啟動預測模型
- 使用 Amazon SageMaker AI 和 Azure DevOps 建置 MLOps 工作流程
- 使用 在 Amazon Bedrock 中設定模型調用記錄 AWS CloudFormation
- 為 SageMaker 建立自訂 Docker 容器映像，並將其用於 AWS Step Functions 中的模型訓練
- 使用 Amazon Bedrock 代理程式，透過文字型提示在 Amazon EKS 中自動建立存取項目控制項
- AWS 使用 Terraform 和 Amazon Bedrock 在 上部署 RAG 使用案例
- 使用 Amazon SageMaker 中的推論管道，將預先處理邏輯部署到單一端點中的 ML 模型
- 搭配 Kiro 和其他編碼助理使用 MCP 伺服器部署即時編碼安全驗證
- 使用 RAG 和 ReAct 提示，開發進階生成式 AI 聊天式助理
- 使用 Amazon Bedrock 代理程式和知識庫開發全自動聊天式助理
- 使用 Amazon Bedrock 和 Amazon Transcribe 從語音輸入記錄機構知識
- 使用 Amazon Personalize 產生個人化和重新排名的建議
- 使用 SageMaker AI 和 hydra 簡化從本機開發到可擴展實驗的機器學習工作流程
- 將自然語言轉換為查詢 DSL for OpenSearch 和 Elasticsearch 查詢
- 使用 Amazon Q Developer 作為編碼助理，以提高您的生產力
- 使用 SageMaker Processing 對 TB 級 ML 資料集進行分散式特徵工程
- 使用 Flask 和 AWS Elastic Beanstalk 視覺化 AI/ML 模型結果
- 更多模式

將儲存 AWS CodeCommit 庫與另一個帳戶中 AWS 帳戶 的 Amazon SageMaker AI Studio Classic 建立關聯

Laurens van der Maas 和 Aubrey Oosthuizen，Amazon Web Services

總結

此模式提供如何將一個 AWS 帳戶 （帳戶 A) 中的 AWS CodeCommit 儲存庫與另一個 AWS 帳戶 （帳戶 B) 中的 Amazon SageMaker AI Studio Classic 建立關聯的說明和程式碼。若要設定關聯，您必須在帳戶 A 中建立 AWS Identity and Access Management (IAM) 政策和角色，並在帳戶 B 中建立 IAM 內嵌政策。然後，您可以使用 shell 指令碼，將 CodeCommit 儲存庫從帳戶 A 複製到帳戶 B 中的 Amazon SageMaker AI Classic。

先決條件和限制

先決條件

- 兩個 AWS 帳戶，一個包含 CodeCommit 儲存庫，另一個包含具有使用者的 SageMaker AI 網域
- 佈建的 SageMaker AI 網域和使用者，可透過網際網路存取或透過虛擬私有網路 AWS Security Token Service (VPC AWS STS) 端點存取 CodeCommit 和 ()
- 對 IAM 的基本了解
- 對 SageMaker AI Studio Classic 的基本了解
- 對 Git 和 CodeCommit 的基本了解

限制

此模式僅適用於 SageMaker AI Studio Classic，不適用於 Amazon SageMaker AI 上的 RStudio。

Architecture

技術堆疊

- Amazon SageMaker AI
- Amazon SageMaker AI Studio Classic
- AWS CodeCommit
- AWS Identity and Access Management (IAM)
- Git

目標架構

下圖顯示將 CodeCommit 儲存庫從帳戶 A 與帳戶 B 中的 SageMaker AI Studio Classic 建立關聯的架構。

該圖顯示以下工作流程：

1. 使用者透過角色擔任帳戶 A 中`MyCrossAccountRepositoryContributorRole`的角色`sts:AssumeRole`，同時使用帳戶 B 中 SageMaker AI Studio Classic 中的 SageMaker AI 執行角色。擔任的角色包含 CodeCommit 許可，可複製並與指定的儲存庫互動。
2. 使用者從 SageMaker AI Studio Classic 中的系統終端機執行 Git 命令。

自動化和擴展

此模式包含可使用 AWS Cloud Development Kit (AWS CDK)、 AWS CloudFormation或 Terraform 自動化的手動步驟。

工具

AWS 工具

- Amazon SageMaker AI 是一種受管機器學習 (ML) 服務，可協助您建置和訓練 ML 模型，然後將模型部署到生產就緒的託管環境中。
- Amazon SageMaker AI Studio Classic 是適用於機器學習的 Web 型整合開發環境 (IDE)，可讓您建置、訓練、偵錯、部署和監控機器學習模型。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。

  注意： AWS CodeCommit 不再提供給新客戶。的現有客戶 AWS CodeCommit 可以繼續正常使用服務。進一步了解
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。

其他工具

- Git 是一種分散式版本控制系統，可在軟體開發期間追蹤原始程式碼的變更。

史詩

在帳戶 A 中建立 IAM 政策和 IAM 角色

任務	Description	所需的技能
在帳戶 A 中建立儲存庫存取的 IAM 政策。	1. 登入 AWS 管理主控台 並開啟 IAM 主控台。 2. 在導覽窗格中，選擇政策，然後選擇建立政策。 3. 選擇 JSON 標籤。 4. 複製此模式額外資訊區段中範例 IAM 政策的政策陳述式，然後將陳述式貼到 JSON 編輯器中。請務必取代政策中的所有預留位置值。 5. 選擇下一步：標籤，然後選擇下一步：檢閱。 6. 針對 Name (名稱)，輸入政策的名稱。注意：在此模式中，IAM 政策稱為 `CrossAccountAccessForMySharedDemoRepo` ，但您可以選擇任何您偏好的政策名稱。 7. 選擇建立政策。 **Tip** 最佳實務是將 IAM 政策的範圍限制為使用案例所需的最低許可。	AWS DevOps

任務	Description	所需的技能
在帳戶 A 中建立儲存庫存取的 IAM 角色。	1. 在 IAM 主控台的導覽窗格中，選擇角色，然後選擇建立角色。 2. 針對信任的實體類型，選取 AWS 帳戶。 3. 在 AWS 帳戶區段中，選取另一個 AWS 帳戶。 4. 針對帳戶 ID，輸入帳戶 B 的帳戶 ID。 5. 在新增許可頁面上，搜尋並選擇您先前建立`CrossAccountAccessForMySharedDemoRepo` 的政策。 6. 選擇下一步。 7. 在 Role name (角色名稱) 中，輸入名稱。注意：在此模式中，IAM 角色名稱稱為 `MyCrossAccountRepositoryContributorRole` ，但您可以選擇您偏好的任何角色名稱。 8. 選擇建立角色，然後複製新角色的 Amazon Resource Name (ARN)。	AWS DevOps

在帳戶 B 中建立 IAM 內嵌政策

任務	Description	所需的技能
將內嵌政策連接至帳戶 B 中連接至 SageMaker 網域使用者的執行角色。	1. 在 IAM 主控台的導覽窗格中，選擇角色。	AWS DevOps

任務	Description	所需的技能
	2. 搜尋並選擇連接到帳戶 B 中 SageMaker AI 網域使用者的執行角色。 3. 選擇新增許可，然後選擇建立內嵌政策。 4. 選擇 JSON 標籤。 5. 複製下列政策陳述式，然後將其貼到 JSON 編輯器中。 `{ "Version": "2012-10-17", "Statement": [ { "Sid": "VisualEditor0", "Effect": "Allow", "Action": "sts:AssumeRole", "Resource": "arn:aws:iam::<Account_A_ID>:role/<Account_A_Role_Name>" } ] }` 6. 將 取代<Account_A_ID> 為帳戶 A 的帳戶 ID。將 取代<Account_A_Role_Name> 為您先前建立的 IAM 角色名稱。 7. 選擇檢閱政策。 8. 在名稱中，輸入內嵌政策的名稱。	

任務	Description	所需的技能
	9. 選擇建立政策。	

在帳戶 B 的 SageMaker AI Studio Classic 中複製儲存庫

任務	Description	所需的技能
在帳戶 B 的 SageMaker AI Studio Classic 中建立 shell 指令碼。	1. 在 SageMaker 主控台的導覽窗格中，選擇 Studio。 2. 選取您的使用者設定檔，然後選擇開啟 Studio。 3. 在首頁區段中，選擇開啟啟動器。 4. 在公用程式和檔案區段中，選擇文字檔案。 5. 從此模式的其他資訊區段中的範例 SageMaker shell 指令碼複製指令碼，然後將陳述式貼到新檔案中。請務必取代指令碼中的所有預留位置值。 6. 在新檔案的 untitled.txt 索引標籤上按一下滑鼠右鍵，然後選擇重新命名文字。對於新名稱，輸入 cross_account_git_clone.sh，然後選擇重新命名。	AWS DevOps
從系統終端機叫用 shell 指令碼。	1. 在 SageMaker 主控台的主區段中，選擇開啟啟動器。 2. 在公用程式和檔案區段中，選擇系統終端機。	AWS DevOps

任務	Description	所需的技能
	3. 在終端機中，執行下列命令： `chmod u+x ./cross_account_git_clone.sh && ./cross_account_git_clone.sh` 您已在 SageMaker AI Studio 跨帳戶中複製 CodeCommit 儲存庫。您現在可以從系統終端機執行所有 Git 命令。	

其他資訊

IAM 政策範例

如果您使用此範例政策，請執行下列動作：

- <CodeCommit_Repository_Region> 將 取代 AWS 區域 為 儲存庫的 。
- <Account_A_ID> 將 取代為帳戶 A 的帳戶 ID。
- <CodeCommit_Repository_Name> 將 取代為帳戶 A 中 CodeCommit 儲存庫的名稱。

```
{
"Version": "2012-10-17",
"Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "codecommit:BatchGet*",
            "codecommit:Create*",
            "codecommit:DeleteBranch",
            "codecommit:Get*",
            "codecommit:List*",
            "codecommit:Describe*",
            "codecommit:Put*",
```

```
            "codecommit:Post*",
            "codecommit:Merge*",
            "codecommit:Test*",
            "codecommit:Update*",
            "codecommit:GitPull",
            "codecommit:GitPush"
        ],
        "Resource": [

 "arn:aws:codecommit:<CodeCommit_Repository_Region>:<Account_A_ID>:<CodeCommit_Repository_Name>
        ]
    }
]
}
```

SageMaker AI shell 指令碼範例

如果您使用此範例指令碼，請執行下列動作：

- <Account_A_ID>將 取代為帳戶 A 的帳戶 ID。
- <Account_A_Role_Name>將 取代為您先前建立的 IAM 角色名稱。
- <CodeCommit_Repository_Region>將 取代 AWS 區域 為 儲存庫的 。
- <CodeCommit_Repository_Name>將 取代為帳戶 A 中 CodeCommit 儲存庫的名稱。

```
#!/usr/bin/env bash
#Launch from system terminal
pip install --quiet git-remote-codecommit

mkdir -p ~/.aws
touch ~/.aws/config

echo "[profile CrossAccountAccessProfile]
region = <CodeCommit_Repository_Region>
credential_source=EcsContainer
role_arn = arn:aws:iam::<Account_A_ID>:role/<Account_A_Role_Name>
output = json" > ~/.aws/config

echo '[credential "https://git-
codecommit.<CodeCommit_Repository_Region>.amazonaws.com"]
        helper = !aws codecommit credential-helper $@ --profile
 CrossAccountAccessProfile
```

```
        UseHttpPath = true' > ~/.gitconfig

git clone codecommit::<CodeCommit_Repository_Region>://
CrossAccountAccessProfile@<CodeCommit_Repository_Name>
```

使用 Amazon Textract 從 PDF 檔案自動擷取內容

Tianxia Jia，Amazon Web Services

總結

許多組織需要從上傳到其商業應用程式的 PDF 檔案擷取資訊。例如，組織可能需要準確從稅務或醫療 PDF 檔案擷取資訊，以進行稅務分析或醫療索賠處理。

在 Amazon Web Services (AWS) 雲端上，Amazon Textract 會自動從 PDF 檔案擷取資訊（例如，列印的文字、表單和資料表），並產生 JSON 格式的檔案，其中包含原始 PDF 檔案的資訊。您可以在 AWS 管理主控台中或透過實作 API 呼叫來使用 Amazon Textract。我們建議您使用程式設計 API 呼叫來擴展和自動處理大量 PDF 檔案。

當 Amazon Textract 處理檔案時，會建立下列`Block`物件清單：頁面、行和文字、表單（索引鍵/值對）、資料表和儲存格，以及選取元素。還包括其他物件資訊，例如週框方塊、可信度間隔、IDs和關係。Amazon Textract 會將內容資訊擷取為字串。正確識別和轉換的資料值是必要的，因為下游應用程式可以更輕鬆地使用這些值。

此模式描述使用 Amazon Textract 自動從 PDF 檔案擷取內容並將其處理為乾淨輸出的step-by-step工作流程。模式使用範本比對技術來正確識別必要欄位、金鑰名稱和資料表，然後將後置處理更正套用至每個資料類型。您可以使用此模式來處理不同類型的 PDF 檔案，然後您可以擴展和自動化此工作流程，以處理格式相同的 PDF 檔案。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 現有的 Amazon Simple Storage Service (Amazon S3) 儲存貯體，用於儲存轉換為 JPEG 格式供 Amazon Textract 處理後的 PDF 檔案。如需 S3 儲存貯體的詳細資訊，請參閱 Amazon S3 文件中的儲存貯體概觀。
- 已安裝和設定的 `Textract_PostProcessing.ipynb` Jupyter 筆記本（已連接）。如需 Jupyter 筆記本的詳細資訊，請參閱 Amazon SageMaker 文件中的建立 Jupyter 筆記本。 Amazon SageMaker
- 具有相同格式的現有 PDF 檔案。
- 了解 Python。

限制

- 您的 PDF 檔案必須品質良好且清晰可讀。建議使用原生 PDF 檔案，但如果所有個別單字都清晰，您可以使用轉換為 PDF 格式的掃描文件。如需詳細資訊，請參閱 AWS Machine Learning 部落格上的使用 Amazon Textract：視覺效果偵測和移除進行 PDF 文件預先處理。
- 對於多頁檔案，您可以使用非同步操作，或將 PDF 檔案分割成單一頁面並使用同步操作。如需這兩個選項的詳細資訊，請參閱 Amazon Textract 文件中的偵測和分析多頁文件中的文字，以及偵測和分析單頁文件中的文字。

Architecture

此模式的工作流程會先在範例 PDF 檔案上執行 Amazon Textract (第一次執行)，然後在與第一個 PDF 格式相同的 PDF 檔案上執行它 (重複執行)。下圖顯示合併的第一次執行和重複執行工作流程，可自動且重複地從具有相同格式的 PDF 檔案擷取內容。

圖表顯示此模式的下列工作流程：

1. 將 PDF 檔案轉換為 JPEG 格式，並將其存放在 S3 儲存貯體中。
2. 呼叫 Amazon Textract API 並剖析 Amazon Textract 回應 JSON 檔案。
3. 透過為每個必要欄位新增正確的`KeyName:DataType`配對來編輯 JSON 檔案。建立重複執行階段`TemplateJSON`的檔案。
4. 定義每個資料類型 （ 例如浮點數、整數和日期 ） 的後製處理更正函數。
5. 準備與您第一個 PDF 檔案格式相同的 PDF 檔案。
6. 呼叫 Amazon Textract API 並剖析 Amazon Textract 回應 JSON。
7. 比對剖析的 JSON 檔案與 `TemplateJSON` 檔案。
8. 實作後製處理更正。

每個必要欄位的最終 JSON `Value` 輸出檔案都有正確的 `KeyName`和 。

目標技術堆疊

- Amazon SageMaker
- Amazon S3
- Amazon Textract

自動化和擴展

您可以使用 AWS Lambda 函數來自動化重複執行工作流程，該函數會在新的 PDF 檔案新增至 Amazon S3 時啟動 Amazon Textract。然後，Amazon Textract 會執行處理指令碼，並將最終輸出儲存到儲存位置。如需詳細資訊，請參閱 Lambda 文件中的使用 Amazon S3 觸發叫用 Lambda 函數。

工具

- Amazon SageMaker 是一項全受管 ML 服務，可協助您快速輕鬆地建置和訓練 ML 模型，然後將模型直接部署到生產就緒的託管環境中。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Amazon Textract 可讓您輕鬆地將文件文字偵測和分析新增至應用程式。

史詩

第一次執行

任務	Description	所需的技能
轉換 PDF 檔案。	將 PDF 檔案分割為單一頁面並將其轉換為 JPEG 格式以進行 Amazon Textract 同步操作 ()，以準備初次執行的 PDF 檔案Syn API。 **Note** 您也可以將 Amazon Textract 非同步操作 (Asyn API) 用於多頁 PDF 檔案。	資料科學家、開發人員
剖析 Amazon Textract 回應 JSON。	開啟 Textract_PostProcessing.ipynb Jupyter 筆記本（已連接），並使用下列	資料科學家、開發人員

任務	Description	所需的技能

	程式碼呼叫 Amazon Textract API：
	
	`response = textract.analyze_document( Document={ 'S3Object': { 'Bucket': BUCKET, 'Name': '{}'.format(filename) } }, FeatureTypes=["TABLES", "FORMS"])`
	
	使用下列程式碼，將回應 JSON 剖析為表單和資料表：
	
	`parseformKV=form_kv_from_JSON(response) parseformTables=get_tables_fromJSON(response)`	

編輯 TemplateJSON 檔案。	編輯每個 KeyName和對應 DataType（例如，字串、浮點數、整數或日期）和資料表標頭 （例如，ColumnNames 和）的剖析 JSONRowNames。 此範本用於每個個別的 PDF 檔案類型，這表示範本可以重複使用於格式相同的 PDF 檔案。	資料科學家、開發人員

任務	Description	所需的技能
定義後置處理校正函數。	Amazon Textract 對 `TemplateJSON` 檔案的回應中的值為字串。日期、浮點數、整數或貨幣沒有差異。這些值必須轉換為下游使用案例的正確資料類型。 使用下列程式碼，根據`TemplateJSON` 檔案修正每個資料類型： `finalJSON=postprocessingCorrection(parsedJSON,templateJSON)`	資料科學家、開發人員

重複執行

任務	Description	所需的技能
準備 PDF 檔案。	將 PDF 檔案分割成單一頁面，並將其轉換為 JPEG 格式以進行 Amazon Textract 同步操作 ()，以準備 PDF 檔案Syn API。 **Note** 您也可以將 Amazon Textract 非同步操作 (Asyn API) 用於多頁 PDF 檔案。	資料科學家、開發人員

任務	Description	所需的技能
呼叫 Amazon Textract API。	使用下列程式碼呼叫 Amazon Textract API： `response = textract.analyze_document( Document={ 'S3Object': { 'Bucket': BUCKET, 'Name': '{}'.format(filename) } }, FeatureTypes=["TABLES", "FORMS"])`	資料科學家、開發人員
剖析 Amazon Textract 回應 JSON。	使用下列程式碼，將回應 JSON 剖析為表單和資料表： `parseformKV=form_kv_from_JSON(response) parseformTables=get_tables_fromJSON(response)`	資料科學家、開發人員

任務	Description	所需的技能
載入 TemplateJSON 檔案，並將其與剖析的 JSON 比對。	使用以下命令，使用 TemplateJSON 檔案擷取正確的鍵值對和資料表： `form_kv_corrected= form_kv_correction (parseformKV,templ ateJSON) form_table_correct ed=form_Table_corr ection(parseformTa bles, templateJSON) form_kv_table_correcte d_final={**form_kv _corrected , **form_ta ble_corrected}`	資料科學家、開發人員
後置處理更正。	在 TemplateJSON 檔案和後置處理函數DataType中使用 ，透過使用下列程式碼來更正資料： `finalJSON=postproc essingCorrection(f orm_kv_table_corre cted_final,templat eJSON)`	資料科學家、開發人員

相關資源

- 使用 Amazon Textract 從文件中自動擷取文字和結構化資料
- 使用 Amazon Textract 擷取文字和結構化資料
- Amazon Textract 資源

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

在 Amazon SageMaker AI Studio Lab 中將 DeepAR 用於時間序列，以建置冷啟動預測模型

Ivan Cui 和 Eyal Shacham，Amazon Web Services

摘要

無論您是更有效率地為 Web 流量配置資源、預測患者對人員需求的需求，還是預測公司產品的銷售，預測都是不可或缺的工具。冷啟動預測會針對歷史資料很少的時間序列建置預測，例如剛進入零售市場的新產品。此模式使用 Amazon SageMaker AI DeepAR 預測演算法來訓練冷啟動預測模型，並示範如何對冷啟動項目執行預測。

DeepAR 是一種監督式學習演算法，可使用遞歸神經網路 (RNN) 預測純量 （ 一維 ） 時間序列。DeepAR 會針對相關產品的所有時間序列，採取聯合訓練單一模型的方法。

傳統的時間序列預測方法，例如自動迴歸整合移動平均值 (ARIMA) 或指數平滑 (ETS)，很大程度上依賴每個個別產品的歷史時間序列。因此，這些方法對冷啟動預測無效。當您的資料集包含數百個相關時間序列時，DeepAR 的執行效能會優於標準 ARIMA 和 ETS 方法。您也可以使用已訓練的模型，針對與其已訓練之時間序列類似的新時間序列產生預測。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- Amazon SageMaker AI 網域。
- Amazon SageMaker AI Studio Lab 或 Jupiter 實驗室應用程式。
- 具有讀取和寫入許可的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。
- Python 程式設計的知識。
- 使用 Jupyter 筆記本的知識。

限制

- 叫用預測模型而沒有任何歷史資料點，將會傳回錯誤。使用最少的歷史資料點調用模型將傳回不準確的預測，並具有高可信度。此模式建議解決冷啟動預測這些已知限制的方法。
- 有些 AWS 服務 無法全部使用 AWS 區域。如需區域可用性，請參閱依區域的 AWS 服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

- Python 3.10 版或更新版本。
- 模式的筆記本已在 Amazon SageMaker AI Studio 中使用 Python 3 （資料科學） 核心在 ml.t3.medium 執行個體上進行測試。

Architecture

下圖顯示此模式的工作流程和架構元件。

工作流程會執行下列任務：

1. 訓練和測試資料的輸入檔案會合成，然後上傳至 Amazon S3 儲存貯體。此資料包含具有分類和動態功能的多個時間序列，以及目標值 （要預測）。Jupyter 筆記本可視覺化資料，以進一步了解訓練資料的需求和預期的預測值。
2. 建立超參數調校器任務是為了訓練模型，並根據預先定義的指標尋找最佳模型。
3. 輸入檔案會從 Amazon S3 儲存貯體下載到超參數調校任務的每個執行個體。
4. 在調校器任務根據調校器的預先定義閾值選取最佳模型後，模型會部署為 SageMaker AI 端點。
5. 然後，部署的模型已準備好被叫用，其中它的預測會根據測試資料進行驗證。

筆記本示範當有足夠數量的歷史資料點可用時，模型預測目標值的能力。不過，當我們調用具有較少歷史資料點的模型時 （代表冷產品），即使模型的可信度範圍內，模型的預測也不符合原始測試資料。在 模式中，新模型是針對冷產品所建置，其中其初始內容長度 （預測點） 定義為可用的歷史點數量，而新模型會在取得新資料點時進行反覆訓練。筆記本顯示，只要歷史資料點的數量接近其內容長度，模型就會有準確的預測。

工具

AWS 服務

- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon SageMaker AI 是一種受管機器學習 (ML) 服務，可協助您建置和訓練 ML 模型，然後將模型部署到生產就緒的託管環境中。
- Amazon SageMaker AI Studio 是適用於 ML 的 Web 型整合開發環境 (IDE)，可讓您建置、訓練、偵錯、部署和監控 ML 模型。

- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他工具

- Python 是一種一般用途的電腦程式設計語言。

程式碼儲存庫

此模式的程式碼可在 GitHub DeepAR-ColdProduct-Pattern 儲存庫中使用。

最佳實務

- 在虛擬環境中訓練您的模型，並一律使用版本控制進行最高的重現性工作。
- 包含盡可能多的高品質分類功能，以獲得最高的預測模型。
- 請確定中繼資料包含類似的分類項目，以便模型充分推斷冷啟動產品預測。
- 執行超參數調校任務以取得最高的預測模型。
- 在此模式中，您建置的模型內容長度為 24 小時，這表示它會預測接下來的 24 小時。如果您嘗試在歷史資料少於 24 小時時預測接下來的 24 小時，模型的預測準確性會根據歷史資料點數量線性下降。若要緩解此問題，請為每個歷史資料點組建立新的模型，直到此數字達到所需的預測（內容）長度。例如，從內容長度模型 2 小時開始，然後逐步將模型增加到 4 小時、8 小時、16 小時和 24 小時。

史詩

啟動 SageMaker AI Studio Classic 應用程式

任務	Description	所需的技能
啟動您的筆記本環境。	1. 登入 AWS 管理主控台，然後開啟 SageMaker AI Studio 首頁。然後選擇開啟 Studio。 2. 在左側導覽窗格中，選擇應用程式中的 Studio Classic	資料科學家

任務	Description	所需的技能
	圖示。然後，選擇應用程式清單上的開啟按鈕。 如需詳細資訊，請參閱 SageMaker AI 文件中的啟動 Amazon SageMaker AI Studio。 SageMaker	

建立和啟用筆記本

任務	Description	所需的技能
設定虛擬環境以進行模型訓練。	若要設定虛擬環境以進行模型訓練，請執行下列動作： 1. 將`deepar_synthetic.ipynb` 筆記本從此模式的 GitHub 儲存庫下載到本機電腦。 2. 在 Amazon SageMaker AI Studio Classic 中，從 Studio Classic 選單列中選擇上傳檔案圖示，然後選取下載的筆記本。 3. 在左側導覽窗格中的檔案瀏覽器中選擇筆記本。依照提示設定筆記本環境。選取資料科學 3.0 映像和 Python 3 核心。 如需詳細資訊，請參閱 SageMaker AI 文件中的將檔案	資料科學家

任務	Description	所需的技能
	上傳至 SageMaker AI Studio Classic。 SageMaker	
建立和驗證預測模型。	• 遵循筆記本中的指示來建立訓練和測試資料、訓練模型，然後調用模型。 • 觀察在提供足夠的歷史資料點時，模型的預測有多準確。	資料科學家

相關資源

- DeepAR 超參數
- 使用 AWS 機器學習服務預測新產品簡介的需求
- 啟動 Amazon SageMaker AI Studio Classic
- 使用 SageMaker AI DeepAR 預測演算法

使用 Amazon SageMaker AI 和 Azure DevOps 建置 MLOps 工作流程

Deepika Kumar、Sara van de Moosdijk 和 Amazon Web Services 的 PHP Kokoh Prasetyo

總結

機器學習操作 (MLOps) 是一組可自動化和簡化機器學習 (ML) 工作流程和部署的實務。MLOps 著重於自動化 ML 生命週期。它有助於確保模型不僅經過開發，而且還會以系統化和重複的方式部署、監控和重新訓練。它將 DevOps 原則帶入 ML。MLOps 可加快 ML 模型的部署速度、提高一段時間內的準確度，以及更強大的保證，確保它們提供真正的商業價值。

在開始 MLOps 旅程之前，組織通常會有現有的 DevOps 工具和資料儲存解決方案。此模式展示如何利用 Microsoft Azure 和 的優勢 AWS。它可協助您將 Azure DevOps 與 Amazon SageMaker AI 整合，以建立 MLOps 工作流程。

解決方案可簡化 Azure 和 之間的工作 AWS。您可以使用 Azure 進行開發和 AWS 機器學習。它可提升從頭到尾建立機器學習模型的有效程序，包括資料處理、訓練和部署 AWS。為了提高效率，您可以透過 Azure DevOps 管道管理這些程序。此解決方案適用於生成式 AI 中的基礎模型操作 (FMOps) 和大型語言模型操作 (LLMOps)，其中包括微調、向量資料庫和提示管理。

先決條件和限制

先決條件

- Azure 訂閱 – 存取 Azure DevOps 等 Azure 服務，用於設定持續整合和持續部署 (CI/CD) 管道。
- 作用中 AWS 帳戶 – 在此 AWS 服務 模式中使用 的許可。
- 資料 – 存取歷史資料以訓練機器學習模型。
- 熟悉 ML 概念 – 了解 Python、Jupyter 筆記本和機器學習模型開發。
- 安全組態 – 在 Azure 和 AWS 之間正確設定角色、政策和許可，以確保安全的資料傳輸和存取。
- （選用）向量資料庫 – 如果您為向量資料庫使用擷取增強生成 (RAG) 方法和第三方服務，則需要存取外部向量資料庫。

限制

- 本指南不會討論安全的跨雲端資料傳輸。如需跨雲端資料傳輸的詳細資訊，請參閱AWS 混合多雲端解決方案。

- 多雲端解決方案可能會增加即時資料處理和模型推論的延遲。
- 本指南提供多帳戶 MLOps 架構的一個範例。必須根據您的機器學習和 AWS 策略進行調整。
- 本指南不會描述使用 Amazon SageMaker AI 以外的 AI/ML 服務。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

Architecture

目標架構

目標架構整合 Azure DevOps 與 Amazon SageMaker AI，建立跨雲端 ML 工作流程。它使用 Azure for CI/CD 流程和 SageMaker AI for ML 模型訓練和部署。它概述了透過模型建置和部署取得資料 （來自 Amazon S3、Snowflake 和 Azure Data Lake 等來源） 的程序。主要元件包括用於模型建置和部署、資料準備、基礎設施管理的 CI/CD 管道，以及用於訓練和微調、評估和部署 ML 模型的 Amazon SageMaker AI。此架構旨在跨雲端平台提供高效、自動化和可擴展的 ML 工作流程。

架構包含下列元件：

1. 資料科學家在開發帳戶中執行 ML 實驗，透過使用各種資料來源探索 ML 使用案例的不同方法。資料科學家執行單元測試和試驗，並追蹤他們的實驗，他們可以將 Amazon SageMaker AI 與 MLflow 搭配使用。在生成式 AI 模型開發中，資料科學家從 Amazon SageMaker AI JumpStart 模型中微調基礎模型。在模型評估之後，資料科學家會將程式碼推送並合併到託管在 Azure DevOps 上的模型建置儲存庫。此儲存庫包含多步驟模型建置管道的程式碼。
2. 在 Azure DevOps 上，提供持續整合 (CI) 的模型建置管道可以在程式碼合併至主要分支時自動或手動啟動。在自動化帳戶中，這會啟用 SageMaker AI 管道，以根據準確性進行資料預先處理、模型訓練和微調、模型評估和條件式模型註冊。
3. 自動化帳戶是跨 ML 平台的中央帳戶，可託管 ML 環境 (Amazon ECR)、模型 (Amazon S3)、模型中繼資料 (SageMaker AI Model Registry)、功能 (SageMaker AI Feature Store)、自動化管道 (SageMaker AI Pipelines) 和 ML 日誌洞察 (CloudWatch)。對於生成式 AI 工作負載，您可能需要對下游應用程式中的提示進行額外的評估。提示管理應用程式有助於簡化和自動化程序。此帳戶允許 ML 資產的可重複使用性，並強制執行最佳實務來加速 ML 使用案例的交付。
4. 最新的模型版本會新增至 SageMaker AI 模型登錄檔以供檢閱。它會追蹤模型版本和個別成品 （系列和中繼資料）。它也會管理模型的狀態 （核准、拒絕或待定），並管理下游部署的 版本。
5. 在模型登錄檔中經過訓練的模型透過 Studio 界面或 API 呼叫核准後，事件可以分派至 Amazon EventBridge。EventBridge 會在 Azure DevOps 上啟動模型部署管道。

6. 模型部署管道提供持續部署 (CD)，可從模型部署儲存庫檢查來源。來源包含程式碼、模型部署的組態，以及品質基準測試的測試指令碼。模型部署管道可以根據您的推論類型量身打造。
7. 質量控制檢查後，模型部署管道會將模型部署到預備帳戶。預備帳戶是生產帳戶的副本，用於整合測試和評估。對於批次轉換，模型部署管道可以自動更新批次推論程序，以使用最新核准的模型版本。對於即時、無伺服器或非同步推論，它會設定或更新個別模型端點。
8. 在預備帳戶中成功測試後，可以透過模型部署管道手動核准，將模型部署到生產帳戶。此管道會在部署至生產步驟中佈建生產端點，包括模型監控和資料意見回饋機制。
9. 模型進入生產環境後，請使用 SageMaker AI Model Monitor 和 SageMaker AI Clarify 等工具來識別偏差、偵測偏離，並持續監控模型的效能。

自動化和擴展

使用基礎設施做為程式碼 (IaC) 自動部署到多個帳戶和環境。透過自動化設定 MLOps 工作流程的程序，可以區分 ML 團隊在不同專案中使用的環境。 AWS CloudFormation 會將基礎設施視為程式碼，協助您建立、佈建和管理 AWS 資源。

工具

AWS 服務

- Amazon SageMaker AI 是一種受管 ML 服務，可協助您建置和訓練 ML 模型，然後將模型部署到生產就緒的託管環境中。
- AWS Glue 是一種全受管的擷取、轉換和載入 (ETL) 服務。它可協助您可靠地分類、清理、擴充和移動資料存放區和資料串流之間的資料。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。在此模式中，Amazon S3 用於資料儲存，並與 SageMaker AI 整合，用於模型訓練和模型物件。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。在此模式中，Lambda 用於資料預處理和後製處理任務。
- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。在此模式中，它會存放 SageMaker AI 用於訓練和部署環境的 Docker 容器。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。在此模式中，EventBridge 會協調事件驅動型或時間型工作流程，以啟動自動模型重新訓練或部署。

- Amazon API Gateway 可協助您建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。 在此模式中，它會用來為 SageMaker AI 端點建立面向外部的單一進入點。
- 對於 RAG 應用程式，您可以使用 Amazon OpenSearch Service 和 Amazon RDS for PostgreSQL AWS 服務等 來存放向 LLM 提供內部資料的向量內嵌。

其他工具

- Azure DevOps 可協助您管理 CI/CD 管道，並促進程式碼建置、測試和部署。
- Azure Data Lake Storage 或 Snowflake 是 ML 模型訓練資料的可能第三方來源。
- Pinecone、Milvus 或 ChromaDB 是儲存向量內嵌的可能第三方向量資料庫。

最佳實務

在實作此多雲端 MLOps 工作流程的任何元件之前，請完成下列活動：

- 定義並了解機器學習工作流程和支援它所需的工具。不同的使用案例需要不同的工作流程和元件。例如，功能存放區在個人化使用案例中可能需要重複使用功能和低延遲推論，但其他使用案例可能不需要。需要了解資料科學團隊的目標工作流程、使用案例要求和偏好的協作方法，才能成功自訂架構。
- 為架構的每個元件建立明確的責任分離。將資料儲存分散到 Azure Data Lake Storage、Snowflake 和 Amazon S3 可能會增加複雜性和成本。如果可能，請選擇一致的儲存機制。同樣地，請避免使用 Azure 和 AWS DevOps 服務的組合，或 Azure 和 AWS ML 服務的組合。
- 選擇一或多個現有模型和資料集，以執行 MLOps 工作流程的end-to-end測試。測試成品應反映資料科學團隊在平台進入生產環境時所開發的實際使用案例。

史詩

設計您的 MLOps 架構

任務	Description	所需的技能
識別資料來源。	根據目前和未來的使用案例、可用的資料來源和資料類型 （例如機密資料），記錄需要與 MLOps 平台整合的資料來源。資料可以存放在	資料工程師、資料科學家、雲端架構師

任務	Description	所需的技能
	Amazon S3、Azure Data Lake Storage、Snowflake 或其他來源。對於生成式 AI 工作負載，資料也可能包含基於所產生回應的知識庫。此資料會以向量內嵌形式存放在向量資料庫中。建立整合這些來源與平台的計劃，並保護對正確資源的存取。	
選擇適用的服務。	根據資料科學團隊所需的工作流程、適用的資料來源和現有的雲端架構，新增或移除服務，以自訂架構。例如，資料工程師和資料科學家可能會在 SageMaker AI AWS Glue或 Amazon EMR 中執行資料預先處理和特徵工程。不太可能需要這三種服務。	AWS 管理員、資料工程師、資料科學家、ML 工程師

任務	Description	所需的技能
分析安全需求。	收集並記錄安全需求。這包括判斷： • 哪些團隊或工程師可以存取特定資料來源 • 哪些團隊或工程師可以存取預先訓練的基礎模型 • 是否允許團隊存取其他團隊的程式碼和模型 • 團隊成員應為非開發帳戶擁有哪些許可（如果有的話） • 跨雲端資料傳輸需要實作哪些安全措施 如需保護生成式 AI 工作負載的詳細資訊，請參閱保護生成式 AI：生成式 AI 安全範圍矩陣簡介 (AWS 部落格文章）。	AWS 管理員、雲端架構師

設定 AWS Organizations

任務	Description	所需的技能
設定 AWS Organizations。	在根 AWS Organizations 目錄上設定 AWS 帳戶。這可協助您管理在多帳戶 MLOps 策略中建立的後續帳戶。如需詳細資訊，請參閱 AWS Organizations 文件。	AWS 管理員

設定開發環境和版本控制

任務	Description	所需的技能
建立 AWS 開發帳戶。	建立 AWS 帳戶 ，讓資料工程師和資料科學家具有實驗和建立 ML 模型的許可。如需說明，請參閱 AWS Organizations 文件中的在組織中建立成員帳戶。	AWS 管理員
建立模型建置儲存庫。	在 Azure 中建立 Git 儲存庫，資料科學家可以在實驗階段完成後推送其模型建置和部署程式碼。如需說明，請參閱 Azure DevOps 文件中的設定 Git 儲存庫。	DevOps 工程師、ML 工程師
建立模型部署儲存庫。	在 Azure 中建立 Git 儲存庫，以存放標準部署程式碼和範本。它應該包含組織使用的每個部署選項的程式碼，如設計階段中所識別。例如，它應該包含即時端點、非同步端點、無伺服器推論或批次轉換。如需說明，請參閱 Azure DevOps 文件中的設定 Git 儲存庫。	DevOps 工程師、ML 工程師
建立 Amazon ECR 儲存庫。	設定 Amazon ECR 儲存庫，將核准的 ML 環境儲存為 Docker 影像。允許資料科學家和 ML 工程師定義新的環境。如需說明，請參閱 Amazon ECR 文件中的建立私有儲存庫。	ML 工程師
設定 SageMaker AI Studio。	根據先前定義的安全需求、偏好的資料科學工具（例如	資料科學家、ML 工程師、Prompt 工程師

任務	Description	所需的技能
	MLflow) 和偏好的整合式開發環境 (IDE)，在開發帳戶上設定 SageMaker AI Studio。使用生命週期組態來自動化關鍵功能的安裝，並為資料科學家建立統一的開發環境。如需詳細資訊，請參閱 SageMaker AI 文件中的 Amazon SageMaker AI Studio 和 MLflow 追蹤伺服器。 SageMaker	

整合 CI/CD 管道

任務	Description	所需的技能
建立自動化帳戶。	建立自動化管道和任務執行 AWS 帳戶 所在的 。您可以讓資料科學團隊讀取此帳戶的存取權。如需說明，請參閱 AWS Organizations 文件中的在組織中建立成員帳戶。	AWS 管理員
設定模型登錄檔。	在自動化帳戶中設定 SageMaker AI 模型登錄檔。此登錄檔會存放 ML 模型的中繼資料，並協助特定資料科學家或團隊核准或拒絕模型。如需詳細資訊，請參閱 SageMaker AI 文件中的使用模型登錄檔註冊和部署模型。	ML 工程師
建立模型建置管道。	在 Azure 中建立 CI/CD 管道，此管道會在程式碼推送至模型建置儲存庫時手動或自動啟動。管道應該檢查原始程式	DevOps 工程師、ML 工程師

任務	Description	所需的技能
	碼，並在自動化帳戶中建立或更新 SageMaker AI 管道。管道應將新模型新增至模型登錄檔。如需建立管道的詳細資訊，請參閱 Azure Pipelines 文件。	

建置部署堆疊

任務	Description	所需的技能
建立 AWS 預備和部署帳戶。	AWS 帳戶 為 ML 模型的預備和部署建立 。這些帳戶應該完全相同，以便在移至生產環境之前，允許在預備階段中準確測試模型。您可以讓資料科學團隊讀取臨時帳戶。如需說明，請參閱 AWS Organizations 文件中的在組織中建立成員帳戶。	AWS 管理員
設定 S3 儲存貯體以進行模型監控。	如果您想要為模型部署管道建立的已部署模型啟用模型監控，請完成此步驟。建立 Amazon S3 儲存貯體以存放輸入和輸出資料。如需建立 S3 儲存貯體的詳細資訊，請參閱 Amazon S3 文件中的建立儲存貯體。設定跨帳戶許可，讓自動化模型監控任務在自動化帳戶中執行。如需詳細資訊，請參閱 SageMaker AI 文件中的監控資料和模型品質。	ML 工程師

任務	Description	所需的技能
建立模型部署管道。	在 Azure 中建立 CI/CD 管道，該管道會在模型註冊表中核准模型時開始。管道應檢查原始程式碼和模型成品、建置基礎設施範本以在預備和生產帳戶中部署模型、在預備帳戶中部署模型、執行自動化測試、等待手動核准，以及將核准的模型部署到生產帳戶中。如需建立管道的詳細資訊，請參閱 Azure Pipelines 文件。	DevOps 工程師、ML 工程師

（選用）自動化 ML 環境基礎設施

任務	Description	所需的技能
組建 AWS CDK 或 CloudFormation 範本。	為所有需要自動部署的環境定義 AWS Cloud Development Kit (AWS CDK) 或 AWS CloudFormation 範本。這可能包括開發環境、自動化環境，以及預備和部署環境。如需詳細資訊，請參閱 AWS CDK和 CloudFormation 文件。	AWS DevOps
建立 基礎設施管道。	在 Azure 中為基礎設施部署建立 CI/CD 管道。管理員可以啟動此管道來建立新的 ， AWS 帳戶 並設定 ML 團隊所需的環境。	DevOps 工程師

疑難排解

問題	解決方案
監控不足和偏離偵測 – 監控不足可能會導致錯過模型效能問題或資料偏離的偵測。	使用 Amazon CloudWatch、SageMaker AI Model Monitor 和 SageMaker AI Clarify 等工具強化監控架構。針對已識別的問題設定立即動作的提醒。
CI 管道觸發錯誤– 由於組態錯誤，可能不會在程式碼合併時觸發 Azure DevOps 中的 CI 管道。	檢查 Azure DevOps 專案設定，以確保 Webhook 已正確設定並指向正確的 SageMaker AI 端點。
控管 – 中央自動化帳戶可能無法跨 ML 平台強制執行最佳實務，導致工作流程不一致。	稽核自動化帳戶設定，確保所有 ML 環境和模型都符合預先定義的最佳實務和政策。
模型登錄檔核准延遲 – 當檢查和核准模型有延遲時會發生這種情況，因為人員需要一些時間來檢閱模型，或是因為技術問題。	實作通知系統，以提醒利益相關者待核准的模型，並簡化審核程序。
模型部署事件失敗 – 傳送至啟動模型部署管道的事件可能會失敗，導致部署延遲。	確認 Amazon EventBridge 具有正確的許可和事件模式，以成功叫用 Azure DevOps 管道。
生產部署瓶頸 – 手動核准程序可能會產生瓶頸，進而延遲模型的生產部署。	最佳化模型部署管道中的核准工作流程，促進及時審核和清晰的溝通管道。

相關資源

AWS 文件

- Amazon SageMaker AI 文件
- Machine Learning Lens (AWS Well Architected Framework)
- 規劃成功的 MLOps (AWS 方案指引)

其他 AWS 資源

- 使用 Amazon SageMaker AI 的企業 MLOps 基礎藍圖 (AWS 部落格文章)

- AWS Summit ANZ 2022 - End-to-end MLOps (YouTube 影片)
- FMOps/LLMOps：操作生成式 AI 和 MLOps 的差異 (AWS 部落格文章)
- 使用 Amazon SageMaker AI Clarify 和 MLOps 服務大規模操作化 LLM 評估 (AWS 部落格文章)
- 向量資料庫在生成式 AI 應用程式中的角色 (AWS 部落格文章)

Azure 文件

- Azure DevOps 文件
- Azure 管道文件

使用 在 Amazon Bedrock 中設定模型調用記錄 AWS CloudFormation

Vikramaditya Bhatnagar，Amazon Web Services

總結

您可以設定 Amazon Bedrock 為 中的所有模型調用收集調用日誌、模型輸入資料和模型輸出資料 AWS 帳戶。這是使用 Amazon Bedrock 建置強大生成式 AI 應用程式的最佳實務。您可以將模型調用日誌存放在 Amazon CloudWatch Logs 日誌群組、Amazon Simple Storage Service (Amazon S3) 儲存貯體或兩者中。在 CloudWatch Logs 中擁有日誌資料可協助您建立自訂指標篩選條件、警示和儀表板。Amazon S3 非常適合用於跨資料複寫， AWS 區域 或用於長期儲存，這取決於您組織的政策。

此模式提供範例 AWS CloudFormation 範本，使用基礎設施做為程式碼 (IaC) 方法來設定 Amazon Bedrock 的模型調用記錄。範本會在 CloudWatch Logs 和 Amazon S3 中設定日誌儲存。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 下列許可：
  - 建立 CloudFormation 堆疊的許可
  - 存取 Amazon Bedrock 的許可
  - 建立和存取 Amazon S3 儲存貯體的許可
  - 建立和存取 CloudWatch Logs 日誌群組的許可
  - 建立和存取 AWS Lambda 函數的許可
  - 建立和存取 AWS Key Management Service (AWS KMS) 金鑰的許可

限制

此模式會將模型調用記錄到 CloudWatch Logs 和 Amazon S3。它不支援僅選擇這兩個服務之一。

Architecture

目標架構

CloudFormation 範本會在您的目標中佈建下列資源 AWS 帳戶：

- 用於儲存模型調用日誌的 CloudWatch Logs 日誌群組
- 用於儲存模型調用日誌的 Amazon S3 儲存貯體和對應的儲存貯體政策
- Amazon S3 儲存貯體，用於存放伺服器端存取日誌和對應的儲存貯體政策
- 在 Amazon Bedrock 中設定記錄設定的 AWS Lambda 函數
- AWS KMS key 和對應的金鑰別名
- Amazon Bedrock 的 AWS Identity and Access Management (IAM) 服務角色

下圖顯示在您部署與此模式相關聯的 CloudFormation 堆疊後，如何存放叫用日誌。當基礎模型交付文字、影像、影片或內嵌資料時，Amazon Bedrock 會發佈日誌資料。如圖所示，Amazon S3 儲存貯體和 CloudWatch Logs 日誌群組會使用 加密 AWS KMS key。

該圖顯示以下工作流程：

1. 使用者向 Amazon Bedrock 中的基礎模型提交查詢。
2. Amazon Bedrock 擔任 IAM 服務角色。
3. Amazon Bedrock 會產生日誌資料，並將其存放在 CloudWatch Logs 日誌群組和 Amazon S3 儲存貯體中。
4. 如果使用者讀取、上傳或刪除 Amazon S3 儲存貯體中包含模型叫用日誌的任何檔案，這些活動會記錄在另一個 Amazon S3 儲存貯體中做為伺服器端存取日誌。

自動化和擴展

若要擴展此解決方案，您可以將 CloudFormation 範本部署為堆疊設定為多個 AWS 區域 和 AWS 帳戶。如需詳細資訊，請參閱 CloudFormation 文件中的使用 StackSets 管理跨帳戶和區域的堆疊。

工具

AWS 服務

- Amazon Bedrock 是一項全受管服務，可讓您透過統一 API 使用來自領導 AI 公司和 Amazon 的高效能基礎模型 (FMs)。
- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。

- Amazon CloudWatch Logs 可協助您集中所有系統、應用程式的日誌， AWS 服務 以便您可以監控日誌並將其安全地存檔。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- Amazon Simple Storage Service (Amazon S3) 是一種物件儲存服務，可提供業界領先的可擴展性、資料可用性、安全性和效能。

其他工具

- Git 是一種開放原始碼的分散式版本控制系統。

程式碼儲存庫

此模式的程式碼可在 GitHub enable-bedrock-logging-using-cloudformation 儲存庫中使用。

史詩

建立 CloudFormation 堆疊

任務	Description	所需的技能
下載 CloudFormation 範本。	從 GitHub 儲存庫下載 CloudFormation 範本。	雲端架構師
部署 範本。	在目標帳戶和區域中建立堆疊。在參數區段中，指定範本中定義的參數值。如需說明，請參閱 CloudFormation 文件中的建立堆疊。	雲端架構師

測試解決方案

任務	Description	所需的技能
啟用模型存取。	在 Amazon Bedrock 中，新增基礎模型的存取權。如需說明，請參閱 Amazon Bedrock 文件中的新增或移除對 Amazon Bedrock 基礎模型的存取權。	雲端架構師
執行範例提示。	在 Amazon Bedrock 遊樂場中，執行範例提示。如需說明，請參閱《Amazon Bedrock 文件》中的使用遊樂場在主控台中產生回應。	雲端架構師
檢閱記錄組態。	1. 登入 Amazon Bedrock 主控台。 2. 在導覽列中，選擇您部署 CloudFormation 堆疊 AWS 區域 的 。 3. 在左側導覽窗格中，於 Bedrock 組態下，選擇設定。 4. 請確認以下內容： • 模型調用記錄已啟用。 • 已選取所有資料類型。 • 對於記錄目的地，會同時選取 S3 和 CloudWatch Logs。	雲端架構師
檢閱 Amazon S3 儲存貯體。	1. 在 S3 組態區段中，選擇瀏覽 S3。這會在 Amazon S3 主控台中開啟目標儲存貯體。	雲端架構師

任務	Description	所需的技能
	2. 確認您先前執行的範例提示有記錄資料。	
檢閱日誌群組。	1. 導覽回 Amazon Bedrock 主控台中的設定頁面。 2. 在 CloudWatch Logs 組態區段中，檢閱 CloudWatch Logs 日誌群組的設定。請記下日誌群組名稱。 3. 開啟 CloudWatch 主控台。 4. 在導覽窗格中，選擇日誌下方的日誌群組。 5. 選擇 Amazon Bedrock 發佈日誌資料的日誌群組名稱。 6. 確認您先前執行的範例提示有記錄資料。	雲端架構師

相關資源

AWS 文件

- 存取 Amazon S3 儲存貯體 (Amazon S3 文件)
- 建立和管理堆疊 (CloudFormation 文件)
- 監控模型調用 (Amazon Bedrock 文件)
- 使用日誌群組和日誌串流 (CloudWatch Logs 文件)

AWS 部落格文章

- 使用 Amazon Bedrock 和 Amazon CloudWatch 整合監控生成式 AI 應用程式
- 使用 Amazon Bedrock Agents 建置強大生成式 AI 應用程式的最佳實務 – 第 1 部分
- 使用 Amazon Bedrock Agents 建置強大生成式 AI 應用程式的最佳實務 – 第 2 部分

為 SageMaker 建立自訂 Docker 容器映像，並將其用於 AWS Step Functions 中的模型訓練

Julia Bluszcz、Aubrey Oosthuizen、Mohan Gowda Purushothama、Neha Sharma 和 Mateusz Zaremba，Amazon Web Services

總結

此模式說明如何為 Amazon SageMaker 建立 Docker 容器映像，並將其用於 AWS Step Functions 中的訓練模型。透過在容器中封裝自訂演算法，您可以在 SageMaker 環境中執行幾乎任何程式碼，無論程式設計語言、架構或相依性為何。

在提供的 SageMaker 筆記本範例中，自訂 Docker 容器映像存放在 Amazon Elastic Container Registry (Amazon ECR) 中。Step Functions 接著會使用存放在 Amazon ECR 中的容器來執行 SageMaker 的 Python 處理指令碼。然後，容器會將模型匯出至 Amazon Simple Storage Service (Amazon S3)。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 具有 Amazon S3 S3 許可的 SageMaker AWS Identity and Access Management (IAM) 角色
- Step Functions 的 IAM 角色
- 熟悉 Python
- 熟悉 Amazon SageMaker Python SDK
- 熟悉 AWS Command Line Interface (AWS CLI)
- 熟悉適用於 Python 的 AWS 開發套件 (Boto3)
- 熟悉 Amazon ECR
- 熟悉 Docker

產品版本

- AWS Step Functions 資料科學 SDK 2.3.0 版
- Amazon SageMaker Python SDK 2.78.0 版

Architecture

下圖顯示為 SageMaker 建立 Docker 容器映像的範例工作流程，然後將其用於 Step Functions 中的訓練模型：

該圖顯示以下工作流程：

1. 資料科學家或 DevOps 工程師使用 Amazon SageMaker 筆記本來建立自訂 Docker 容器映像。
2. 資料科學家或 DevOps 工程師會將 Docker 容器映像存放在私有登錄檔中的 Amazon ECR 私有儲存庫中。
3. 資料科學家或 DevOps 工程師使用 Docker 容器在 Step Functions 工作流程中執行 Python SageMaker 處理任務。

自動化和擴展

此模式中的範例 SageMaker 筆記本使用`ml.m5.xlarge`筆記本執行個體類型。您可以變更執行個體類型以符合您的使用案例。如需 SageMaker 筆記本執行個體類型的詳細資訊，請參閱 Amazon SageMaker 定價。

工具

- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- Amazon SageMaker 是一種受管機器學習 (ML) 服務，可協助您建置和訓練 ML 模型，然後將模型部署到生產就緒的託管環境中。
- Amazon SageMaker Python SDK 是一個開放原始碼程式庫，用於在 SageMaker 上訓練和部署機器學習模型。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務來建置業務關鍵應用程式。
- AWS Step Functions 資料科學 Python SDK 是一個開放原始碼程式庫，可協助您建立 Step Functions 工作流程來處理和發佈機器學習模型。

史詩

建立自訂 Docker 容器映像並將其存放在 Amazon ECR 中

任務	Description	所需的技能
設定 Amazon ECR 並建立新的私有登錄檔。	如果您尚未設定 Amazon ECR，請遵循《Amazon ECR 使用者指南》中的使用 Amazon ECR 設定中的指示來設定 Amazon ECR。 每個 AWS 帳戶都會提供預設的私有 Amazon ECR 登錄檔。	DevOps 工程師
建立 Amazon ECR 私有儲存庫。	遵循《Amazon ECR 使用者指南》中建立私有儲存庫的指示。 **Note** 您建立的儲存庫是您存放自訂 Docker 容器映像的位置。	DevOps 工程師
建立 Dockerfile，其中包含執行 SageMaker 處理任務所需的規格。	建立 Dockerfile，其中包含透過設定 Dockerfile 執行 SageMaker 處理任務所需的規格。如需說明，請參閱《Amazon SageMaker 開發人員指南》中的調整您自己的訓練容器。 如需 Dockerfiles 的詳細資訊，請參閱 Docker 文件中的 Dockerfile 參考。 建立 Dockerfile 的 Jupyter 筆記本程式碼儲存格範例	DevOps 工程師

任務	Description	所需的技能

	儲存格 1
	`# Make docker folder`
	`!mkdir -p docker`
	儲存格 2
	`%%writefile docker/Dockerfile`
	`FROM python:3.7-slim-buster`
	`RUN pip3 install pandas==0.25.3 scikit-learn==0.21.3`
	`ENV PYTHONUNBUFFERED=TRUE`
	`ENTRYPOINT ["python3"]`	

任務	Description	所需的技能
建置 Docker 容器映像並將其推送至 Amazon ECR。	1. 使用您在 AWS CLI 中執行`docker build`命令所建立的 Dockerfile 建置容器映像。 2. 執行`docker push`命令，將容器映像推送至 Amazon ECR。 如需詳細資訊，請參閱在 GitHub 上建置您自己的演算法容器中的建置和註冊容器。GitHub 建置和註冊 Docker 映像的 Jupyter 筆記本程式碼儲存格範例 **Important** 在執行下列儲存格之前，請確定您已建立 Dockerfile，並將其存放在名為 的目錄中`docker`。此外，請確定您已建立 Amazon ECR 儲存庫，並將第一個儲存格中的`ecr_repository` 值取代為儲存庫的名稱。 儲存格 1 `import boto3`	DevOps 工程師

任務	Description	所需的技能
	tag = ':latest' account_id = boto3.client('sts').get_caller_identity().get('Account') region = boto3.Session().region_name ecr_repository = 'byoc' image_uri = '{}.dkr.ecr.{}.amazonaws.com/{}'.format(account_id, region, ecr_repository + tag) 儲存格 2 # Build docker image !docker build -t $image_uri docker 儲存格 3 # Authenticate to ECR !aws ecr get-login-password --region {region} | docker login --username AWS --password-stdin {account_id}.dkr.ecr.{region}.amazonaws.com 儲存格 4 # Push docker image !docker push $image_uri	

任務	Description	所需的技能
	Note 您必須向私有登錄檔驗證 Docker 用戶端，才能使用 `docker push`和 `docker pull`命令。這些命令會在登錄檔中的儲存庫中推送和提取映像。	

建立使用自訂 Docker 容器映像的 Step Functions 工作流程

任務	Description	所需的技能

建立 Python 指令碼，其中包含您的自訂處理和模型訓練邏輯。	撰寫要在資料處理指令碼中執行的自訂處理邏輯。然後，將其儲存為名為 的 Python 指令碼`training.py` 。
	如需詳細資訊，請參閱使用 GitHub 上的 SageMaker 指令碼模式使用您自己的模型。GitHub
	包含自訂處理和模型訓練邏輯的範例 Python 指令碼
	`%%writefile training.py`
	`from numpy import empty`
	`import pandas as pd`
	`import os`
	`from sklearn import datasets, svm`
	`from joblib import dump, load`
		資料科學家

任務	Description	所需的技能

	`if __name__ == '__main__':`
	`    digits = datasets.load_digits()`
	`    #create classifier object`
	`    clf = svm.SVC(gamma=0.001, C=100.)`
	
	`    #fit the model`
	`    clf.fit(digits.data[:-1], digits.target[:-1])`
	
	`    #model output in binary format`
	`    output_path = os.path.join('/opt/ml/processing/model', "model.joblib")`
	`    dump(clf, output_path)`	

任務	Description	所需的技能

建立 Step Functions 工作流程，其中包含您的 SageMaker Processing 任務作為其中一個步驟。	安裝並匯入 AWS Step Functions 資料科學 SDK，並將 training.py 檔案上傳至 Amazon S3。然後，使用 Amazon SageMaker Python SDK 在 Step Functions 中定義處理步驟。
	
	Important
	請確定您已在 AWS 帳戶中為 Step Functions 建立 IAM 執行角色。
	
	要上傳至 Amazon S3 的環境設定範例和自訂訓練指令碼
	
	`!pip install stepfunct ions`
	
	`import boto3`
	`import stepfunctions`
	`import sagemaker`
	`import datetime`
	
	`from stepfunctions import steps`
	`from stepfunct ions.inputs import ExecutionInput`
	`from stepfunct ions.steps import ( Chain )`
	`from stepfunct ions.workflow import Workflow`
		資料科學家

任務	Description	所需的技能
	from sagemaker .processing import ScriptProcessor, ProcessingInput, ProcessingOutput sagemaker_session = sagemaker.Session() bucket = sagemaker _session.default_b ucket() role = sagemaker .get_execution_role() prefix = 'byoc-tra ining-model' # See prerequisites section to create this role workflow_execution_rol e = f"arn:aws:iam:: {account_id}:role/Ama zonSageMaker-StepF unctionsWorkflowEx ecutionRole" execution_input = ExecutionInput(schema={ "Preproce ssingJobName": str}) input_code = sagemaker _session.upload_data("training.py", bucket=bucket, key_prefix="prepro cessing.py",)	

任務	Description	所需的技能
	使用自訂 Amazon ECR 映像和 Python 指令碼的 SageMaker 處理步驟定義範例 **Note** 請務必使用 `execution_input` 參數來指定任務名稱。每次任務執行時，參數的值必須是唯一的。此外，training.py 檔案的程式碼會做為 `input` 參數傳遞至 `ProcessingStep`，這表示它將在容器內複製。`ProcessingInput` 程式碼的目的地與 中的第二個引數相同`container_entrypoint`。 `script_processor = ScriptProcessor(command=['python3'], image_uri=image_uri, role=role, instance_count=1, instance_type='ml.m5.xlarge')`	

任務	Description	所需的技能

	processing_step =
	 steps.ProcessingStep(
	    "training-step",
	    processor=script_processor,
	    job_name=execution_input["PreprocessingJobName"],
	    inputs=[
	        ProcessingInput(
	            source=input_code,
	            destination="/opt/ml/processing/input/code",
	            input_name="code",
	        ),
	    ],
	    outputs=[
	        ProcessingOutput(
	            source='/opt/ml/processing/model',
	            destination="s3://{}/{}".format(bucket, prefix),
	            output_name='byoc-example')
	    ],
	    container_entrypoint=["python3", "/opt/ml/processing/input/code/training.py"],
	)
	
	執行 SageMaker 處理任務的 Step Functions 工作流程範例	

任務	Description	所需的技能

	Note
	此範例工作流程僅包含 SageMaker 處理任務步驟，而非完整的 Step Functions 工作流程。如需完整的工作流程範例，請參閱 AWS Step Functions 資料科學 SDK 文件中的 SageMaker 中的範例筆記本。
	
	`workflow_graph = Chain([processing_step])`
	
	`workflow = Workflow( name="ProcessingWorkflow", definition=workflow_graph, role=workflow_execution_role )`
	
	`workflow.create()`
	`# Execute workflow`
	`execution = workflow.execute( inputs={ "PreprocessingJobName": str(datetime.datetime.now().strftime("%Y%m%d%H%M-%SS")), # Each pre processin`	

任務	Description	所需的技能
	g job (SageMaker processing job) requires a unique name, }) execution_output = execution.get_outp ut(wait=True)	

相關資源

- 程序資料 (Amazon SageMaker 開發人員指南)
- 調整您自己的訓練容器 (Amazon SageMaker 開發人員指南)

使用 Amazon Bedrock 代理程式，透過文字型提示在 Amazon EKS 中自動建立存取項目控制項

Keshav Ganesh 和 Sudhanshu Saurav，Amazon Web Services

總結

當多個團隊需要使用共用的 Amazon Elastic Kubernetes Service (Amazon EKS) 叢集時，組織在管理存取控制和資源佈建方面面臨挑戰。Amazon EKS 等受管 Kubernetes 服務可簡化叢集操作。不過，管理團隊存取和資源許可的管理開銷仍然複雜且耗時。

此模式顯示 Amazon Bedrock 代理程式如何協助您自動化 Amazon EKS 叢集存取管理。此自動化可讓開發團隊專注於其核心應用程式開發，而不是處理存取控制設定和管理。您可以自訂 Amazon Bedrock 代理程式，透過簡單的自然語言提示對各種任務執行動作。

透過使用 AWS Lambda 函數作為動作群組，Amazon Bedrock 代理程式可以處理任務，例如建立使用者存取項目和管理存取政策。此外，Amazon Bedrock 代理程式可以設定 Pod 身分關聯，允許存取叢集中執行之 Pod 的 AWS Identity and Access Management (IAM) 資源。使用此解決方案，組織可以使用簡單的文字提示來簡化其 Amazon EKS 叢集管理、減少手動額外負荷，並改善整體開發效率。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 建立部署程序的 IAM 角色和許可。這包括存取 Amazon Bedrock 基礎模型 (FM)、建立 Lambda 函數，以及目標中任何其他必要資源的許可 AWS 帳戶。
- 在作用中啟用 AWS 帳戶 對這些 Amazon Bedrock FMs存取：Amazon Titan Text Embeddings V2 和 Anthropic Claude 3 Haiku。
- AWS Command Line Interface (AWS CLI) 2.9.11 版或更新版本，已安裝並設定。
- eksctl 0.194.0 或更新版本，已安裝。

限制

- 可能需要訓練和文件，以協助確保這些技術的順利採用和有效使用。使用 Amazon Bedrock、Amazon EKS、Lambda、Amazon OpenSearch Service 和 OpenAPI 為開發人員和 DevOps 團隊帶來了重要的學習曲線。

- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱依區域的 AWS 服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

下圖顯示此模式的工作流程和架構元件。

此解決方案會執行下列步驟：

1. 使用者透過提交提示或查詢來與 Amazon Bedrock 代理程式互動，做為代理程式處理和採取動作的輸入。
2. 根據提示，Amazon Bedrock 代理程式會檢查 OpenAPI 結構描述，以識別要鎖定的正確 API。如果 Amazon Bedrock 代理程式找到正確的 API 呼叫，請求會移至與實作這些動作的 Lambda 函數相關聯的動作群組。
3. 如果找不到相關 API，Amazon Bedrock 代理程式會查詢 OpenSearch 集合。OpenSearch 集合使用索引知識庫內容，這些內容來自包含 Amazon EKS 使用者指南的 Amazon S3 儲存貯體。
4. OpenSearch 集合會將相關內容資訊傳回給 Amazon Bedrock 代理程式。
5. 對於可行的請求（符合 API 操作的請求），Amazon Bedrock 代理程式會在虛擬私有雲端 (VPC) 內執行，並觸發 Lambda 函數。
6. Lambda 函數會根據使用者在 Amazon EKS 叢集內的輸入來執行動作。
7. Lambda 程式碼的 Amazon S3 儲存貯體會存放為 Lambda 函數撰寫程式碼和邏輯的成品。

工具

AWS 服務

- Amazon Bedrock 是一項全受管服務，可讓您透過統一 API 使用來自領導 AI 新創公司的高效能基礎模型 (FMs) 和 Amazon。
- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 上執行 Kubernetes， AWS 而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。

- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon OpenSearch Service 是一項受管服務，可協助您在 中部署、操作和擴展 OpenSearch 叢集 AWS 雲端。其集合功能可協助您整理資料，並建置 Amazon Bedrock 代理程式等 AI 助理可以使用的完整知識庫。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他工具

- eksctl 是一種命令列公用程式，用於在 Amazon EKS 上建立和管理 Kubernetes 叢集。

程式碼儲存庫

此模式的程式碼可在 GitHub eks-access-controls-bedrock-agent 儲存庫中使用。

最佳實務

- 實作此模式時，請盡可能維持最高的安全性。確定 Amazon EKS 叢集是私有的、具有有限的存取許可，且所有資源都在虛擬私有雲端 (VPC) 內。如需詳細資訊，請參閱 Amazon EKS 文件中的安全性最佳實務。
- 盡可能使用 AWS KMS 客戶受管金鑰，並授予他們有限的存取許可。
- 遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。

史詩

設定環境

任務	Description	所需的技能
複製儲存庫。	若要複製此模式的儲存庫，請在本機工作站中執行下列命令： `git clone https://github.com/aws-samp`	AWS DevOps

任務	Description	所需的技能
	`les/eks-access-con trols-bedrock-agen t.git`	
取得 AWS 帳戶 ID。	若要取得 AWS 帳戶 ID，請使用下列步驟： 1. 在複製儲存庫的根資料夾中開啟 Shell，`eks-acces s-controls-bedrock -agent` 。 2. 若要取得您的 AWS 帳戶 ID，請導覽至複製的目錄並執行下列命令： `AWS_ACCOUNT=$(aws sts get-caller-identity --query "Account" --output text)` 此命令會將您的 AWS 帳戶 ID 存放在 `AWS_ACCOUNT` 變數中。	AWS DevOps

任務	Description	所需的技能

建立 Lambda 程式碼的 S3 儲存貯體。	若要實作此解決方案，您必須建立三個提供不同用途的 Amazon S3 儲存貯體，如架構圖所示。S3 儲存貯體適用於 Lambda 程式碼、知識庫和 OpenAPI 結構描述。
	
	若要建立 Lambda 程式碼儲存貯體，請使用下列步驟：
	
	1. 若要為 Lambda 程式碼建立 S3 儲存貯體，請執行下列命令：
	
	`aws s3 mb s3://bedrock-agent-lambda-artifacts-${AWS_ACCOUNT} --region us-east-1`
	
	2. 若要安裝 Lambda 程式碼相依性，請執行下列命令：
	
	`cd eks-lambda`
	`npm install`
	`tsc`
	`cd .. && cd opensearch-lambda`
	`npm install`
	`tsc`
	`cd ..`
	
	3. 若要封裝程式碼並將其上傳至 Lambda 的 S3 儲存貯體，請執行下列命令：
	
	`aws cloudformation package \`
		AWS DevOps

任務	Description	所需的技能
	--template-file eks-access-controls.yaml \ --s3-bucket bedrock-agent-lambda-artifacts-${AWS_ACCOUNT} \ --output-template-file eks-access-controls-template.yaml \ --region us-east-1 套件命令會建立新的 CloudFormation 範本 (eks-access-controls-template.yaml)，其中包含： • 對存放在 S3 儲存貯體中的 Lambda 函數程式碼的參考。 • 所有必要 AWS 基礎設施的定義，包括 VPC、子網路、Amazon Bedrock 代理程式和 OpenSearch 集合。您可以使用此範本，透過 CloudFormation 部署完整的解決方案。	

任務	Description	所需的技能

建立知識庫的 S3 儲存貯體。	若要為知識庫建立 Amazon S3 儲存貯體，請使用下列步驟：
	
	1. 若要為知識庫建立 Amazon S3 儲存貯體，請執行下列命令：
	
	`aws s3 mb s3://eks-knowledge-base-${AWS_ACCOUNT} --region us-east-1`
	
	2. 若要下載 Amazon EKS 使用者指南並將其存放在目錄中，請執行下列命令：
	
	`mkdir dataSource`
	`cd dataSource`
	`curl https://docs.aws.amazon.com/pdfs/eks/latest/userguide/eks-ug.pdf -o eks-user-guide.pdf`
	
	3. 若要將使用者指南上傳至您在步驟 1 中建立的 S3 儲存貯體，請執行下列命令：
	
	`aws s3 cp eks-user-guide.pdf s3://eks-knowledge-base-${AWS_ACCOUNT} \`
	`--region us-east-1 \`
	
	4. 若要返回根目錄，請執行下列命令：
		AWS DevOps

任務	Description	所需的技能
	`cd ..`	

建立 OpenAPI 結構描述的 S3 儲存貯體。	若要為 OpenAPI 結構描述建立 Amazon S3 儲存貯體，請使用下列步驟：
	1. 若要建立 S3 儲存貯體，請執行下列命令：
	`aws s3 mb s3://eks-openapi-schema-${AWS_ACCOUNT} --region us-east-1`
	2. 若要將 OpenAPI 結構描述上傳至 S3 儲存貯體，請執行下列命令：
	`aws s3 cp openapi-schema.yaml s3://eks-openapi-schema-${AWS_ACCOUNT} --region us-east-1`
		AWS DevOps

部署 CloudFormation 堆疊

任務	Description	所需的技能
部署 CloudFormation 堆疊。	若要部署 CloudFormation 堆疊，請使用您先前建立`eks-access-controls-template.yaml` 的 CloudFormation 範本檔案。如需更多詳細資訊，請參閱 CloudFormation 文件中的從	AWS DevOps

任務	Description	所需的技能
	CloudFormation 主控台建立堆疊。 CloudFormation **Note** 使用 CloudFormation 範本佈建 OpenSearch 索引大約需要 10 分鐘。 建立堆疊之後，請記下 `VPC_ID`和 `PRIVATE_SUBNET_ID` 。	

任務	Description	所需的技能

建立 Amazon EKS 叢集。	若要在 VPC 內建立 Amazon EKS 叢集，請使用下列步驟：
	1. 建立`eks-config.yaml` 組態檔案的複本，並將複本命名為 `eks-deploy.yaml` 。
	2. 在文字編輯器中開啟 `eks-deploy.yaml` 。然後，將下列預留位置值取代為已部署堆疊中的值：`PRIVATE_SUBNET1` 、 `VPC_ID`和 `PRIVATE_SUBNET2`
	3. 若要使用 eksctl 公用程式建立叢集，請執行下列命令：
	`eksctl create cluster -f eks-deploy.yaml`
	Note
	此叢集建立程序最多可能需要 15-20 分鐘才能完成。
	4. 若要驗證叢集是否已成功建立，請執行下列命令：
	`aws eks describe-cluster --name --query "cluster.status"`
	`aws eks update-kubeconfig --name --region`
		AWS DevOps

任務	Description	所需的技能
	`kubectl get nodes` 預期的結果如下： • 叢集狀態為 ACTIVE。 • 命令`kubectl get nodes`顯示所有節點都處於 Ready 狀態。	

連接 Lambda 函數和 Amazon EKS 叢集

任務	Description	所需的技能
在 Amazon EKS 叢集和 Lambda 函數之間建立連線。	若要設定網路和 IAM 許可以允許 Lambda 函數與 Amazon EKS 叢集通訊，請使用下列步驟： 1. 若要識別連接至 Lambda 函數的 IAM 角色，請開啟 AWS 管理主控台 並找到名為 的 Lambda 函數`bedrock-agent-eks-access-control` 。記下 IAM 角色的 Amazon Resource Name (ARN)。 2. 若要在 Amazon EKS 叢集中為 Lambda 函數的 IAM 角色建立存取項目，請執行下列命令： `aws eks create-access-entry --cluster-name eks-testing-`	AWS DevOps

任務	Description	所需的技能

	`cluster --principal-arn <principal-Role-ARN>`
	
	3. 若要指派AmazonEKS ClusterAdminPolicy 許可給此角色，請執行下列命令：
	
	`aws eks associate-access-policy --cluster-name eks-testing-cluster --principal-arn <principal-Role-ARN> --policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy --access-scope type=cluster`
	
	如需詳細資訊，請參閱《AmazonEKSClusterAdminPolicy將存取政策與存取項目和 AmazonEKS ClusterAdminPolicy 建立關聯。
	
	4. 找出 Amazon EKS 叢集的安全群組。新增傳入規則，以允許從 Lambda 函數到 Amazon EKS 叢集的傳入網路流量。
	
	針對傳入規則使用下列值：
	
	• 類型 – HTTPS	

任務	Description	所需的技能
	• 連接埠範圍 – 443 • 來源 – Lambda 安全群組 如需詳細資訊，請參閱 Amazon VPC 文件中的設定安全群組規則。	

測試解決方案

任務	Description	所需的技能
測試 Amazon Bedrock 代理程式。	測試 Amazon Bedrock 代理程式之前，請務必執行下列動作： • 首先使用非生產角色進行測試。 • 記錄對叢集存取所做的任何變更。 • 視需要規劃還原變更。 若要存取 Amazon Bedrock 代理程式，請使用下列步驟： 1. AWS 管理主控台 使用具有 Amazon Bedrock 許可的 IAM 角色登入 ，然後開啟位於 https：//https://console.aws.amazon.com/bedrock/ 的 Amazon Bedrock 主控台。 2. 從左側導覽窗格選取代理程式。然後，在客服人員區段中選擇您設定的客服人員。	AWS DevOps

任務	Description	所需的技能
	3. 若要測試代理程式，請嘗試下列範例提示，在其中將 取代Principal-ARN-OF-ROLE 為實際的 IAM 角色 ARN： • 若要為您想要提供 EKS 叢集存取權的任何 IAM 角色建立存取項目，請使用下列提示：`Create an access entry in cluster eks-testing-new for a role whose principal arn is <Principal-ARN-OF-ROLE> with access policy as AmazonEKSAdminPolicy` 預期結果： • 代理程式應確認建立存取項目。 • 若要驗證，請使用 AWS 管理主控台 或使用 Amazon EKS API 進行檢查，並執行下列命令：`aws eks list-access-entries --cluster-name ekscluster` • 若要描述您建立的存取項目，請使用下列提示：`Describe an access entry in cluster eks-`	

任務	Description	所需的技能
	`testing-new whose principal arn is <Principal-ARN-OF-ROLE>` 預期結果： • 代理程式應傳回存取項目的詳細資訊。 • 詳細資訊應該符合您先前為存取項目設定的內容。 • 若要刪除您建立的存取項目，請使用下列提示：`Delete the access entry in cluster eks-testing-new whose principal arn is <Principal-ARN-OF-ROLE>` 預期結果： • 代理程式應確認刪除存取項目。 • 若要驗證，請使用 AWS 管理主控台 或使用 Amazon EKS API 進行檢查，並執行下列命令：`aws eks list-access-entries --cluster-name ekscluster` 您也可以要求代理程式對 EKS Pod 身分關聯執行動作。如需詳細資訊，請參閱《Amazon	

任務	Description	所需的技能
	EKS 文件》中的了解 EKS Pod Identity 如何授予 Pod 存取權 AWS 服務。	

清除

任務	Description	所需的技能
清除資源。	若要清除此模式建立的資源，請使用下列程序。等待每個刪除步驟完成，然後再繼續進行下一個步驟。 **Warning** 此程序將永久刪除這些堆疊建立的所有資源。在繼續之前，請確定您已備份任何重要資料。 1. 若要刪除 Amazon EKS 叢集，請執行下列命令： `eksctl delete cluster -f eks-deploy.yaml` **Note** 此操作可能需要 15-20 分鐘才能完成。 2. 若要刪除 Amazon S3 儲存貯體，請執行下列命令：	AWS DevOps

任務	Description	所需的技能

	• 若要清空 Lambda 儲存貯體：
	`aws s3 rm s3://bedrock-agent-lambda-artifacts-${AWS_ACCOUNT} --recursive`
	• 若要清空知識庫儲存貯體：
	`aws s3 rm s3://eks-knowledge-base-${AWS_ACCOUNT} –recursive`
	• 若要清空 OpenAPI 結構描述儲存貯體：
	`aws s3 rm s3://bedrock-agent-openapi-schema-${AWS_ACCOUNT} –recursive`
	• 若要刪除空儲存貯體：
	`aws s3 rb s3://bedrock-agent-lambda-artifacts-${AWS_ACCOUNT}`
	`aws s3 rb s3://eks-knowledge-base-${AWS_ACCOUNT}`
	`aws s3 rb s3://bedrock-agent-openapi-schema-${AWS_ACCOUNT}`
	3. 若要刪除 CloudFormation 堆疊，請執行下列命令：	

任務	Description	所需的技能

	`aws cloudformation delete-stack \ --stack-name`
	4. 若要驗證 Amazon EKS 叢集的刪除，請執行下列命令：
	`eksctl get clusters`
	5. 若要驗證 Amazon S3 儲存貯體的刪除，請執行下列命令：
	• 若要驗證 Lambda 儲存貯體的刪除：
	`aws s3 ls | grep "bedrock-agent-lambda-artifacts"`
	• 若要驗證刪除知識庫儲存貯體：
	`aws s3 ls | grep "eks-knowledge-base"`
	• 若要驗證 OpenAPI 結構描述儲存貯體的刪除：
	`aws s3 ls | grep "bedrock-agent-openapi-schema"`
	6. 若要驗證堆疊刪除，請執行下列命令：
	`aws cloudformation list-stacks \--query`	

任務	Description	所需的技能
	`'StackSummaries[?StackName==``]'` 如果堆疊無法刪除，請參閱故障診斷。	

疑難排解

問題	解決方案
環境設定期間會傳回非零錯誤代碼。	執行任何命令來部署此解決方案時，請確認您使用的是正確的資料夾。如需詳細資訊，請參閱此模式儲存庫中的 FIRST_DEPLOY.md 檔案。
Lambda 函數無法執行任務。	確定從 Lambda 函數到 Amazon EKS 叢集的連線設定正確。
客服人員提示無法辨識 APIs。	重新部署解決方案。如需詳細資訊，請參閱此模式儲存庫中的 RE_DEPLOY.md 檔案。
堆疊無法刪除。	刪除堆疊的初始嘗試可能會失敗。此失敗可能是因為針對 OpenSearch 集合所建立之自訂資源的相依性問題，而該資源會執行知識庫的索引。若要刪除堆疊，請保留自訂資源以重試刪除操作。

相關資源

AWS 部落格

- 深入了解簡化的 Amazon EKS 存取控制

Amazon Bedrock 文件

- 使用 AI 代理器自動化應用程式中的任務
- Amazon Bedrock 代理程式的運作方式

- 測試代理程式行為並進行疑難排解
- 使用動作群組來定義代理程式要執行的動作

Amazon EKS 文件

- 了解存取控制在 Amazon EKS 中的運作方式

AWS 使用 Terraform 和 Amazon Bedrock 在 上部署 RAG 使用案例

Martin Maritsch、Nicolas Jacob Baer、Olivier Brique、Julian Ferdinand Grueber、Alice Morano 和 Nicola D Orazio、Amazon Web Services

總結

AWS 提供各種選項來建置支援擷取增強生成 (RAG) 的生成式 AI 使用案例。此模式為您提供以 LangChain 和 Amazon Aurora PostgreSQL 相容作為向量存放區之 RAG 型應用程式的解決方案。您可以使用 Terraform 直接將此解決方案部署到 ， AWS 帳戶 並實作下列簡單的 RAG 使用案例：

1. 使用者手動將檔案上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體，例如 Microsoft Excel 檔案或 PDF 文件。（如需支援檔案類型的詳細資訊，請參閱非結構化文件。）
2. 檔案的內容會解壓縮並內嵌至以無伺服器 Aurora PostgreSQL 相容為基礎的知識資料庫中，該資料庫支援近乎即時地將文件擷取至向量存放區。此方法可讓 RAG 模型存取和擷取低延遲之使用案例的相關資訊。
3. 當使用者與文字產生模型互動時，它會透過從先前上傳的檔案擷取相關內容擴增來增強互動。

模式使用 Amazon Titan Text Embeddings v2 做為內嵌模型，而 Anthropic Claude 3 Sonnet 做為文字產生模型，兩者皆可在 Amazon Bedrock 上使用。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- AWS Command Line Interface (AWS CLI) 已安裝並使用 設定 AWS 帳戶。如需安裝說明，請參閱 AWS CLI 文件中的安裝或更新至最新版本的 AWS CLI 。若要檢閱您的 AWS 登入資料和對帳戶的存取，請參閱 AWS CLI 文件中的組態和登入資料檔案設定。
- 在 Amazon Bedrock 主控台中為所需的大型語言模型 (LLMs) 啟用的模型存取 AWS 帳戶。此模式需要下列 LLMs：
  - `amazon.titan-embed-text-v2:0`
  - `anthropic.claude-3-sonnet-20240229-v1:0`

限制

- 此範例架構不包含使用向量資料庫進行程式設計問題回答的界面。如果您的使用案例需要 API，請考慮使用執行擷取和問答任務的 AWS Lambda 函數新增 Amazon API Gateway。
- 此範例架構不包含已部署基礎設施的監控功能。如果您的使用案例需要監控，請考慮新增AWS 監控服務。
- 如果您在短時間內將大量文件上傳至 Amazon S3 儲存貯體，Lambda 函數可能會遇到速率限制。作為解決方案，您可以將 Lambda 函數與 Amazon Simple Queue Service (Amazon SQS) 佇列分離，您可以在其中控制 Lambda 調用速率。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

- AWS CLI 第 2 版或更新版本
- Docker 26.0.0 版或更新版本
- Poetry 1.7.1 版或更新版本
- Python 3.10 版或更新版本
- Terraform 1.8.4 版或更新版本

Architecture

下圖顯示此模式的工作流程和架構元件。

此圖表說明下列項目：

1. 在 Amazon S3 儲存貯體 中建立物件時`bedrock-rag-template-<account_id>`，Amazon S3 通知會叫用 Lambda 函數 `data-ingestion-processor`。
2. Lambda 函數`data-ingestion-processor`是以存放在 Amazon Elastic Container Registry (Amazon ECR) 儲存庫 中的 Docker 映像為基礎`bedrock-rag-template`。

   函數使用 LangChain S3FileLoader 將檔案讀取為 LangChain 文件。然後，LangChain RecursiveCharacterTextSplitter 會指定每個文件區塊，而 `CHUNK_SIZE`和 `CHUNK_OVERLAP`取決於 Amazon Titan Text Embedding V2 內嵌模型的最大字符大小。接著，Lambda 函數會叫用 Amazon Bedrock 上的內嵌模型，將區塊內嵌到數值向量表示法中。最後，這些向量會存放在 Aurora PostgreSQL 資料庫中。若要存取資料庫，Lambda 函數會先從中擷取使用者名稱和密碼 AWS Secrets Manager。

3. 在 Amazon SageMaker AI 筆記本執行個體 上aws-sample-bedrock-rag-template，使用者可以撰寫問題提示。此程式碼會在 Amazon Bedrock 上叫用 Claude 3，並將知識庫資訊新增至提示的內容。因此，Claude 3 會使用文件中的資訊提供回應。

此模式的聯網和安全性方法如下：

- Lambda 函數data-ingestion-processor位於虛擬私有雲端 (VPC) 內的私有子網路中。由於 Lambda 函數的安全群組，因此不允許將流量傳送至公有網際網路。因此，流向 Amazon S3 和 Amazon Bedrock 的流量只會透過 VPC 端點路由。因此，流量不會周遊公有網際網路，這可減少延遲，並在聯網層級增加額外的安全層。
- 適用時，所有資源和資料都會使用別名 的 AWS Key Management Service (AWS KMS) 金鑰進行加密aws-sample/bedrock-rag-template。

自動化和擴展

此模式使用 Terraform 將基礎設施從程式碼儲存庫部署到 AWS 帳戶。

工具

AWS 服務

- Amazon Aurora PostgreSQL 相容版本是完全受管的 ACID 相容關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。在此模式中，Aurora PostgreSQL 相容會使用 pgvector 外掛程式做為向量資料庫。
- Amazon Bedrock 是一項全受管服務，可讓您透過統一 API 使用來自領導 AI 新創公司的高效能基礎模型 (FMs) 和 Amazon。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。在此模式中，Amazon ECR 會託管 data-ingestion-processor Lambda 函數的 Docker 映像。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。

- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。在此模式中，Lambda 會將資料擷取至向量存放區。
- Amazon SageMaker AI 是一種受管機器學習 (ML) 服務，可協助您建置和訓練 ML 模型，然後將模型部署到生產就緒的託管環境中。
- AWS Secrets Manager 可協助您將程式碼中的硬式編碼憑證 (包括密碼) 取代為 Secrets Manager 的 API 呼叫，以便透過程式設計方法來擷取機密。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。VPC 包含子網路和路由表，以控制流量流程。

其他工具

- Docker 是一組平台即服務 (PaaS) 產品，可在作業系統層級使用虛擬化在容器中交付軟體。
- HashiCorp Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。
- Poetry 是一種在 Python 中管理相依性和封裝的工具。
- Python 是一種一般用途的電腦程式設計語言。

程式碼儲存庫

此模式的程式碼可在 GitHub terraform-rag-template-using-amazon-bedrock 儲存庫中使用。

最佳實務

- 雖然此程式碼範例可以部署到任何 AWS 區域，但我們建議您使用美國東部 （維吉尼亞北部） – `us-east-1`或美國西部 （加利佛尼亞北部） – `us-west-1`。此建議是根據此模式發佈時 Amazon Bedrock 中基礎模型和內嵌模型的可用性。如需 中 up-to-date清單 AWS 區域，請參閱 Amazon Bedrock 文件中的 模型支援 AWS 區域。如需將此程式碼範例部署到其他區域的詳細資訊，請參閱其他資訊。
- 此模式僅提供proof-of-concept(PoC) 或試行示範。如果您想要將程式碼帶入生產環境，請務必使用下列最佳實務：
  - 啟用 Amazon S3 的伺服器存取記錄。

  - 設定 Lambda 函數的監控和提醒。
  - 如果您的使用案例需要 API，請考慮使用執行擷取和問答任務的 Lambda 函數新增 Amazon API Gateway。
- 遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。

史詩

在 中部署解決方案 AWS 帳戶

任務	Description	所需的技能
複製儲存庫。	若要複製此模式隨附的 GitHub 儲存庫，請使用下列命令： `git clone https://github.com/aws-samples/terraform-rag-template-using-amazon-bedrock`	AWS DevOps
設定變數。	若要設定此模式的參數，請執行下列動作： 1. 在您的電腦的 GitHub 儲存庫中，使用下列命令來開啟 `terraform` 資料夾： `cd terraform` 2. 開啟 `commons.tfvars` 檔案，並根據您的需求自訂參數。	AWS DevOps
部署解決方案。	若要部署解決方案，請執行下列動作：	AWS DevOps

任務	Description	所需的技能

	1. 在 `terraform` 資料夾中，使用下列命令來執行 Terraform 並傳入您自訂的變數：
	
	`terraform init`
	`terraform apply -var-file=commons.tfvars`
	
	2. 確認架構圖中顯示的資源已成功部署。
	
	基礎設施部署會在 VPC 內佈建 SageMaker AI 執行個體，並具有存取 Aurora PostgreSQL 資料庫的許可。	

測試解決方案

任務	Description	所需的技能
執行示範。	先前的基礎設施部署成功後，請使用下列步驟在 Jupyter 筆記本中執行示範： 1. 登入部署基礎設施 AWS 管理主控台 之 AWS 帳戶的 。 2. 開啟 SageMaker AI 筆記本執行個體 `aws-sample-bedrock-rag-template` 。 3. 使用拖放將 `rag_demo.ipynb` Jupyter 筆記本移至	一般 AWS

任務	Description	所需的技能
	SageMaker AI 筆記本執行個體。 4. 在 SageMaker AI `conda_python3` 筆記本執行個體`rag_demo.ipynb` 上開啟，然後選擇核心。 5. 若要執行示範，請執行筆記本的儲存格。 Jupyter 筆記本會引導您完成下列程序： • 安裝需求 • 內嵌定義 • 資料庫連線 • 資料擷取 • 擷取擴增文字產生 • 相關文件查詢	

清除基礎設施

任務	Description	所需的技能
清除基礎設施。	若要移除您不再需要的所有資源，請使用下列命令： `terraform destroy -var-file=commons.tfvars`	AWS DevOps

相關資源

AWS resources

- 使用 Python 建置 Lambda 函數
- 基礎模型的推論參數
- 存取 Amazon Bedrock 基礎模型
- 向量資料庫在生成式 AI 應用程式中的角色 (AWS 資料庫部落格）
- 使用 Amazon Aurora PostgreSQL

其他資源

- pgvector 文件

其他資訊

實作向量資料庫

此模式使用 Aurora PostgreSQL 相容來實作 RAG 的向量資料庫。作為 Aurora PostgreSQL 的替代方案， 為 RAG AWS 提供其他功能和服務，例如 Amazon Bedrock 知識庫和 Amazon OpenSearch Service。您可以選擇最符合您特定需求的解決方案：

- Amazon OpenSearch Service 提供分散式搜尋和分析引擎，您可以用來存放和查詢大量資料。
- Amazon Bedrock 知識庫旨在建置和部署知識庫作為額外的抽象概念，以簡化 RAG 擷取和擷取程序。Amazon Bedrock 知識庫可以同時使用 Aurora PostgreSQL 和 Amazon OpenSearch Service。

部署到其他 AWS 區域

如架構中所述，我們建議您使用美國東部（維吉尼亞北部）– `us-east-1`或美國西部 `us-west-1`（加利佛尼亞北部）– 部署此程式碼範例。不過，有兩種可能的方法來將此程式碼範例部署到 `us-east-1`和 以外的區域`us-west-1`。您可以在 `commons.tfvars` 檔案中設定部署區域。對於跨區域基礎模型存取，請考慮下列選項：

- 周遊公有網際網路 – 如果流量可以周遊公有網際網路，請將網際網路閘道新增至 VPC。然後，調整指派給 Lambda 函數`data-ingestion-processor`和 SageMaker AI 筆記本執行個體的安全群組，以允許輸出流量到公有網際網路。

- 不周遊公有網際網路 – 若要將此範例部署到 us-east-1或 以外的任何區域us-west-1，請執行下列動作：

1. 在 us-east-1或 us-west-1區域中，建立額外的 VPC，包括 的 VPC 端點bedrock-runtime。
2. 使用 VPC 對等互連或傳輸閘道至應用程式 VPC 來建立對等互連。 https://docs.aws.amazon.com/vpc/latest/tgw/tgw-peering.html
3. 在 bedrock-runtime us-east-1或 之外的任何 Lambda 函數中設定 boto3 用戶端時us-west-1，請將 bedrock-runtime us-east-1或 us-west-1 中 VPC 端點的私有 DNS 名稱傳遞endpoint_url給 boto3 用戶端。

使用 Amazon SageMaker 中的推論管道，將預先處理邏輯部署到單一端點中的 ML 模型

Mohan Gowda Purushothama、Gabriel Rodriguez Garcia 和 Mateusz Zaremba，Amazon Web Services

總結

此模式說明如何在 Amazon SageMaker 中使用推論管道，在單一端點中部署多個管道模型物件。管道模型物件代表不同的機器學習 (ML) 工作流程階段，例如預先處理、模型推論和後製處理。為了說明序列連線管道模型物件的部署，此模式示範如何根據內建於 SageMaker 的線性學習程式演算法，部署預先處理 Scikit-learn 容器和迴歸模型。 https://docs.aws.amazon.com/sagemaker/latest/dg/linear-learner.html SageMaker 部署託管在 SageMaker 中的單一端點後方。

> **Note**
>
> 此模式中的部署會使用 ml.m4.2xlarge 執行個體類型。我們建議您使用符合您資料大小需求和工作流程複雜性的執行個體類型。如需更多資訊，請參閱 Amazon SageMaker 定價。此模式針對 Scikit-learn 使用預先建置的 Docker 映像，但您可以使用自己的 Docker 容器，並將其整合到您的工作流程中。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Python 3.9
- Amazon SageMaker Python SDK 和 Boto3 程式庫
- 具有基本 SageMaker 許可和 Amazon Simple Storage Service (Amazon S3) 許可的 AWS Identity and Access Management (AWS IAM) 角色 https://docs.aws.amazon.com/AmazonS3/latest/userguide/access-policy-language-overview.html

產品版本

- Amazon SageMaker Python SDK 2.49.2

Architecture

目標技術堆疊

- Amazon Elastic Container Registry (Amazon ECR)
- Amazon SageMaker
- Amazon SageMaker Studio
- Amazon Simple Storage Service (Amazon S3)
- Amazon SageMaker 的即時推論端點

目標架構

下圖顯示部署 Amazon SageMaker 管道模型物件的架構。

該圖顯示以下工作流程：

1. SageMaker 筆記本會部署管道模型。
2. S3 儲存貯體存放模型成品。
3. Amazon ECR 會從 S3 儲存貯體取得來源容器映像。

工具

AWS 工具

- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- Amazon SageMaker 是一項受管 ML 服務，可協助您建置和訓練 ML 模型，然後將模型部署到生產就緒的託管環境中。
- Amazon SageMaker Studio 是適用於 ML 的 Web 型整合式開發環境 (IDE)，可讓您建置、訓練、偵錯、部署和監控 ML 模型。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

Code

此模式的程式碼可在 GitHub 推論管道搭配 Scikit-learn 和線性學習程式儲存庫中使用。

史詩

準備資料集

任務	Description	所需的技能

為您的迴歸任務準備資料集。	在 Amazon SageMaker Studio 中開啟筆記本。
	
	若要匯入所有必要的程式庫並初始化您的工作環境，請在筆記本中使用下列範例程式碼：
	
	`import sagemaker`
	`from sagemaker import get_execution_role`
	
	`sagemaker_session = sagemaker.Session()`
	
	`# Get a SageMaker-compatible role used by this Notebook Instance.`
	`role = get_execution_role()`
	
	`# S3 prefix`
	`bucket = sagemaker_session.default_bucket()`
	`prefix = "Scikit-LinearLearner-pipeline-abalone-example"`
	
	若要下載範例資料集，請將下列程式碼新增至您的筆記本：
	
	`! mkdir abalone_data`
		資料科學家

任務	Description	所需的技能
	``` ! aws s3 cp s3://sage maker-sample-files /datasets/tabular/ uci_abalone/abalon e.csv ./abalone_data ``` **Note** 此模式中的範例使用 UCI Machine Learning 儲存庫中的鮑魚資料集。	

將資料集上傳至 S3 儲存貯體。	在您先前準備資料集的筆記本中，新增下列程式碼，將範例資料上傳至 S3 儲存貯體：
	
	``
	WORK_DIRECTORY =
	 "abalone_data"
	
	train_input = sagemaker
	_session.upload_data(
	    path="{}/{}".forma
	t(WORK_DIRECTORY,
	 "abalone.csv"),
	    bucket=bucket,
	    key_prefix="{}/
	{}".format(prefix,
	 "train"),
	)
	``
		資料科學家

使用 SKLearn 建立資料預處理器

任務	Description	所需的技能
準備 preprocessor.py 指令碼。	1. 從 GitHub sklearn_abalone_featurizer.py 儲存庫中的 Python 檔案複製預先處理邏輯，然後將程式碼貼到名為 的個別 Python 檔案中`sklearn_abalone_featurizer.py` 。您可以修改程式碼以符合您的自訂資料集和自訂工作流程。 2. 將`sklearn_abalone_featurizer.py` 檔案儲存在專案的根目錄中（也就是在您執行 SageMaker 筆記本的相同位置）。	資料科學家

建立 SKLearn 預處理器物件。	若要建立可納入最終推論管道的 SKLearn 預處理器物件（稱為 SKLearn 估算器），請在 SageMaker 筆記本中執行下列程式碼：
	`from sagemaker.sklearn.estimator import SKLearn`
	
	`FRAMEWORK_VERSION = "0.23-1"`
	`script_path = "sklearn_abalone_featurizer.py"`
	
	`sklearn_preprocessor = SKLearn(`
	`entry_point=script_path,`
		資料科學家

任務	Description	所需的技能
	role=role, framework_version= FRAMEWORK_VERSION, instance_type="ml. c4.xlarge", sagemaker_session= sagemaker_session,) sklearn_preproc essor.fit({"train": train_input})	

任務	Description	所需的技能

測試預處理器的推論。	若要確認您的預處理器已正確定義，請在 SageMaker 筆記本中輸入下列程式碼來啟動批次轉換任務：
	
	`# Define a SKLearn Transformer from the trained SKLearn Estimator`
	`transformer = sklearn_preprocessor.transformer(`
	`instance_count=1, instance_type="ml.m5.xlarge", assemble_with="Line", accept="text/csv"`
	`)`
	
	`# Preprocess training input`
	`transformer.transform(train_input, content_type="text/csv")`
	`print("Waiting for transform job: " + transformer.latest_transform_job.job_name)`
	`transformer.wait()`
	`preprocessed_train = transformer.output_path`	

建立機器學習模型

任務	Description	所需的技能
建立模型物件。	若要根據線性學習程式演算法建立模型物件，請在 SageMaker 筆記本中輸入下列程式碼： `import boto3 from sagemaker .image_uris import retrieve ll_image = retrieve("linear-learner", boto3.Session().region_name) s3_ll_output_key_prefix = "ll_training_output" s3_ll_output_location = "s3://{}/{}/{}/{}" .format(bucket, prefix, s3_ll_output_key_prefix, "ll_model") ll_estimator = sagemaker.estimator.Estimator(ll_image, role, instance_count=1, instance_type="ml.m4.2xlarge", volume_size=20, max_run=3600, input_mode="File", output_path=s3_ll_output_location,`	資料科學家

任務	Description	所需的技能

	<pre> sagemaker_session=
	sagemaker_session,
	)
	
	ll_estimator.s
	et_hyperparameters
	(feature_dim=10,
	 predictor_type="re
	gressor", mini_batc
	h_size=32)
	
	ll_train_data =
	 sagemaker.inputs.T
	rainingInput(
	    preprocessed_train
	,
	    distribution="Full
	yReplicated",
	    content_type="text
	/csv",
	    s3_data_type="S3Pr
	efix",
	)
	
	data_channels =
	 {"train": ll_train_
	data}
	ll_estimator.fit(inpu
	ts=data_channels,
	 logs=True)</pre>
	上述程式碼會從模型的公有 Amazon ECR 登錄檔擷取相關的 Amazon ECR Docker 影像、建立估算器物件，然後使用該物件來訓練迴歸模型。	

部署最終管道

任務	Description	所需的技能

部署管道模型。	若要建立管道模型物件（即預處理器物件）並部署物件，請在 SageMaker 筆記本中輸入下列程式碼：
	
	`from sagemaker.model import Model`
	`from sagemaker.pipeline import PipelineModel`
	`import boto3`
	`from time import gmtime, strftime`
	
	`timestamp_prefix = strftime("%Y-%m-%d-%H-%M-%S", gmtime())`
	
	`scikit_learn_inferencee_model = sklearn_preprocessor.create_model()`
	`linear_learner_model = ll_estimator.create_model()`
	
	`model_name = "inference-pipeline-" + timestamp_prefix`
	`endpoint_name = "inference-pipeline-ep-" + timestamp_prefix`
	`sm_model = PipelineModel(`
	`name=model_name, role=role, models=[scikit_learn_infe`
		資料科學家

任務	Description	所需的技能
	rencee_model, linear_learner_model]) sm_model.deploy(init ial_instance_count =1, instance_type="ml. c4.xlarge", endpoint_ name=endpoint_name) **Note** 您可以調整模型物件中使用的執行個體類型，以符合您的需求。	

任務	Description	所需的技能

測試推論。	若要確認端點是否正常運作，請在 SageMaker 筆記本中執行下列範例推論程式碼：
	
	`from sagemaker.predicto r import Predictor from sagemaker.serializ ers import CSVSerial izer`
	
	`payload = "M, 0.44, 0.365, 0.125, 0.516, 0.2155, 0.114, 0.155" actual_rings = 10 predictor = Predictor( endpoint_name=endp oint_name, sagemaker _session=sagemaker _session, serialize r=CSVSerializer() )`
	
	`print(predictor .predict(payload))`
		資料科學家

相關資源

- 使用 Amazon SageMaker 推論管道和 Scikit-learn 進行預測之前，請先預先處理輸入資料 (AWS Machine Learning 部落格）
- 使用 Amazon SageMaker (GitHub) 進行端對端Machine Learning GitHub

搭配 Kiro 和其他編碼助理使用 MCP 伺服器部署即時編碼安全驗證

Ivan Girardi 和 Iker Reina Fuente，Amazon Web Services

摘要

此模式說明如何實作模型內容通訊協定 (MCP) 伺服器，整合三種業界標準的安全性掃描工具，以提供全面的程式碼安全分析。伺服器可讓 AI 編碼助理 （例如 Kiro、Amazon Q Developer 和 Cline) 自動將程式碼片段和基礎設施掃描為程式碼 (IaC) 組態。透過這些掃描，編碼助理可協助識別安全漏洞、設定錯誤和違規。

在數百萬個程式碼片段上訓練的 AI 程式碼產生器會建立安全盲點，該訓練資料的安全性如何？ 此模式可在程式碼產生期間提供即時安全驗證，協助開發人員在程式碼過程中識別和了解潛在的安全問題。此方法可協助開發人員解決直接漏洞和來自相依性的繼承風險。透過彌補 AI 效率和安全合規之間的差距，此模式有助於安全採用 AI 驅動的開發工具。

此模式可協助組織透過 AI 輔助編碼工具增強其開發安全實務，提供跨多種程式設計語言和基礎設施定義的持續安全掃描功能。解決方案結合了下列工具的功能：

- 用於掃描 IaC 檔案的 Checkov，包括 Terraform AWS CloudFormation和 Kubernetes 資訊清單
- Semgrep 用於分析多種程式設計語言，例如 Python、JavaScript、Java 等
- 專用 Python 安全性掃描的 Bandit

此解決方案的主要功能包括下列項目：

- Delta 掃描新的程式碼區段，降低運算開銷
- 隔離的安全工具環境，防止跨工具污染
- 與 AI 編碼助理 (Kiro、Amazon Q Developer、Cline 等） 無縫整合
- 程式碼產生期間的即時安全意見回饋
- 適用於組織合規的自訂掃描規則

模式為具有標準化回應格式的安全掃描提供統一界面，讓您更輕鬆地將安全檢查整合到開發工作流程中。模式使用 Python 和 MCP 架構來提供自動化安全意見回饋。此方法可協助開發人員在開發程序初期識別和解決安全問題，同時透過詳細的調查結果了解安全最佳實務。

先決條件和限制

先決條件

- 如果您想要使用其中一個編碼助理， AWS 帳戶 則具有使用 Kiro 或 Amazon Q Developer 之存取權的作用中
- 已安裝 Python 3.10 版或更新版本
- uv 套件管理員已安裝
- 熟悉安全掃描工具和概念
- 對 IaC 和應用程式安全性的基本了解

限制

- Bandit 掃描僅限於 Python 檔案。
- 即時掃描可能會影響大型程式碼基底的效能。
- 工具特定的限制是以支援的檔案格式和語言為基礎。
- 需要手動檢閱才能驗證安全性問題清單。
- 安全掃描結果需要安全專業知識才能正確解譯。
- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

- Python 3.10 版或更新版本
- Checkov 3.0.0 版或更新版本
- Semgrep 1.45.0 版或更新版本
- Bandit 1.7.5 版或更新版本
- MCP【cli】 1.11.0 版或更新版本
- Pydantic 1.10.0 版或更新版本
- Loguru 0.6.0 版或更新版本

Architecture

下圖顯示此解決方案的架構。

該圖顯示以下工作流程：

1. 開發人員使用 AI 助理 （ 例如 Kiro、Cline、Amazon Q Developer 或 Roo Code) 來產生或分析程式碼。AI 助理會傳送程式碼以進行安全性掃描。
2. MCP 安全掃描器伺服器會透過將請求路由到適當的專用掃描器來處理請求：IaC 檔案的 Checkov、多個程式設計語言分析的 Semgrep，或 Python 特定安全性掃描的 Bandit。
3. 掃描器結果包含安全性調查結果、嚴重性等級、詳細說明和建議的修正，會透過 AI 助理傳回開發人員。
4. 建立持續的意見回饋迴圈，開發人員會在其中接收即時安全驗證，透過 AI 助理實現自動修正，並在開發期間提升安全最佳實務。

架構可降低下列常見的安全風險：

- 命令注入
- 提示注入
- 路徑周遊
- 相依性攻擊
- 資源耗盡

架構實作下列最佳實務，以降低這些常見的安全風險：

- 所有使用者和 AI 模型輸入都會寫入暫存檔案。
- 命令列界面 (CLI) 命令不會直接輸入。
- 檔案系統存取僅限於暫時目錄和檔案。
- 臨時檔案會自動清除。
- 掃描回應已淨化。
- 會強制執行限制程序功能的程序隔離。
- 會記錄所有掃描活動。

自動化和擴展

模式透過下列功能支援自動化：

- 與 AI 編碼助理整合，以進行自動程式碼掃描
- 用於自動化處理的標準化 API 回應
- 透過 MCP 組態檔案的組態
- 支援批次處理多個檔案
- 跨多種程式設計語言和 IaC 格式的可擴展掃描

掃描程序可以透過提供的 API 端點自動化：

- `scan_with_checkov` 用於 IaC 掃描
- `scan_with_semgrep` 適用於多語言程式碼掃描
- `scan_with_bandit` 適用於 Python 特定掃描
- `get_supported_formats` 用於格式驗證

擴充掃描工具時，請遵循本節先前所述的設計原則和最佳實務。另請參閱最佳實務。

工具

AWS 服務

- Kiro 是一種代理程式編碼服務，與開發人員一起將提示轉換為詳細規格，然後轉換為工作程式碼、文件和測試。Kiro 代理程式可協助開發人員解決具有挑戰性的問題，並自動化產生文件和單元測試等任務。
- Amazon Q Developer 是採用生成式 AI 技術的對話式助理，可協助您了解、建置、擴展和操作 AWS 應用程式。

其他工具

- Bandit 是一種專門的 Python 安全掃描器工具。它可偵測常見的 Python 安全問題，例如不安全的 函數、硬式編碼秘密和注入漏洞。Bandit 提供詳細的可信度和嚴重性評分。
- Checkov 是一種靜態程式碼分析工具，可檢查 IaC 的安全性和合規性設定錯誤。此外，Checkov 會偵測合規違規和安全最佳實務。
- Cline 是採用 AI 技術的編碼助理，可在 VS Code 中執行。
- Loguru 是 Python 的資料驗證程式庫。
- 模型內容通訊協定 (MCP) 是用於建置 AI 輔助開發工具的開放原始碼架構。

- Pydantic 是 Python 的資料驗證程式庫。
- Semgrep 會分析原始程式碼是否有安全漏洞和錯誤。它支援多種程式設計語言。Semgrep 使用以安全為重心的規則集進行全面分析。它提供詳細的可信度和嚴重性評分。

程式碼儲存庫

此模式的程式碼可在 GitHub MCP 安全掃描器：即時保護 AI 程式碼助理儲存庫中取得。儲存庫包含 MCP 伺服器實作、Kiro、Amazon Q Developer、Cline 等 MCP 組態的詳細資訊、組態範例和測試公用程式。

儲存庫結構包括：

- `security_scanner_mcp_server/` - 主要伺服器實作
- `docs/` - 文件和示範資料
- `tests/` - 測試檔案
- `mcp-config-example.json` - 範例 MCP 組態
- `requirements.txt` - 專案相依性

最佳實務

安全性掃描實作

- 檢閱安全性調查結果，以驗證問題並排定優先順序。
- 保持掃描工具 (Checkov、Semgrep 和 Bandit) 更新至最新版本。
- 將此模式的 MCP 安全工具與其他安全措施和工具搭配使用。
- 定期更新安全規則集和政策。

組態管理

- 在官方版本控制來源中使用 MCP 組態檔案。
- 記錄自訂規則和組態。

整合

- 在開發週期的早期整合安全性掃描。

- 在預先遞交勾點或持續整合和持續部署 (CI/CD) 管道中設定自動掃描。
- 為您的環境設定適當的嚴重性閾值。
- 建立明確的程序來處理安全問題清單。

營運考量事項

- 監控掃描效能和資源用量。
- 實作適當的錯誤處理和記錄。
- 維護自訂組態的文件。
- 建立檢閱和更新安全規則的程序。

此外，請記住下列最佳實務：

- 一律在特定內容中驗證安全調查結果。
- 將安全工具和相依性保持在最新狀態。
- 使用多個安全工具進行全面涵蓋。
- 遵循開發程序中的安全最佳實務。

史詩

(Kiro 使用者） 設定 MCP 安全掃描器伺服器

任務	Description	所需的技能
設定 MCP 設定。	您可以使用 Kiro IDE，透過（選項 1) 手動尋找組態檔案或（選項 2) 在 Kiro 中編輯組態檔案。 • （選項 1) 手動尋找和編輯組態檔案，如下所示： • 路徑： `.kiro/settings/mcp.json` （本機）或	應用程式開發人員

任務	Description	所需的技能
	~/.kiro/settings/mcp.json （全域） • （選項 2) 使用 Kiro IDE 修改組態檔案，如下所示： • 選擇左側導覽選單中的 Kiro 符號。 • 導覽至「MCP SERVERS」區段。 • 編輯組態設定。使用 Kiro IDE 所做的變更將儲存至下列其中一個組態檔案：.kiro/settings/mcp.json （本機）或 ~/.kiro/settings/mcp.json （全域） • （兩個選項）在您完成選項 1 或選項 2 之後，新增 security-scanner 組態 .kiro/settings/mcp.json （本機）或 ~/.kiro/settings/mcp.json （全域）： `{ "mcpServers": { "security-scanner" : { "command": "uvx", "args": [ "--from", "git+https://github.com/aws-samples/sample-mcp-security-scanner.git@main",`	

任務	Description	所需的技能

	<pre>"security
	_scanner_mcp_server"
	],
	"env": {
	"FASTMCP_
	LOG_LEVEL": "ERROR"
	},
	"disabled":
	false,
	"autoApprove": []
	}
	}
	}</pre>	

(Amazon Q Developer 使用者） 設定 MCP 安全掃描器伺服器

任務	Description	所需的技能

設定 MCP 設定。	若要手動設定 MCP 設定，請使用下列步驟：
	1. 在 ~/.aws/amazonq/mcp.json （全域）或 .amazonq/mcp.json （本機）編輯 MCP 組態檔案。
	2. 新增下列 security-scanner 組態 ~/.aws/amazonq/mcp.json （全域）或 .amazonq/mcp.json （本機）：
	<pre>{
	"mcpServers": {
	"security-scanner"
	: {</pre>
		應用程式開發人員

任務	Description	所需的技能

	"command": "uvx",
	"args": [
	"--from",
	"git+http
	s://github.com/aws
	-samples/sample-mc
	p-security-scanner
	.git@main",
	"security
	_scanner_mcp_server"
	],
	"env": {
	"FASTMCP_
	LOG_LEVEL": "ERROR"
	}
	}
	}
	}	

(Cline 使用者） 設定 MCP 安全掃描器伺服器

任務	Description	所需的技能
設定 MCP 設定。	若要手動設定 MCP 設定，請使用下列步驟： 1. 選擇要開啟的延伸模組，然後選取 MCP 伺服器。 2. 選擇已安裝索引標籤，然後選擇設定 MCP 伺服器以開啟`cline_mcp_settings.json` 檔案。 3. 新增下列 security-scanner 組態`cline_mcp_settings.json` ：	應用程式開發人員

任務	Description	所需的技能

	{
	  "mcpServers": {
	  "security-scanner"
	: {
	  "command": "uvx",
	  "args": [
	  "--from",
	  "git+http
	s://github.com/aws
	-samples/sample-mc
	p-security-scanner
	.git@main",
	  "security
	_scanner_mcp_server"
	  ],
	  "env": {
	  "FASTMCP_
	LOG_LEVEL": "ERROR"
	  },
	  "disabled":
	 false,
	  "autoApprove": []
	  }
	  }
	}	

使用 Python 和 Bandit 的程式碼分析範例

任務	Description	所需的技能
執行程式碼分析。	若要使用 Python 和 Bandit 執行程式碼分析，請使用下列步驟： 1. `bandit_test.py` 使用下列內容建立名為 的新 Python 檔案：	應用程式開發人員

任務	Description	所需的技能

	<pre>import pickle
	import yaml
	import subprocess
	from flask import
	 Flask, request
	import hashlib
	
	app = Flask(__n
	ame__)
	
	@app.route("/unsafe
	_pickle", methods=[
	"POST"])
	def unsafe_pi
	ckle_usage():
	    data = request.g
	et_data()
	    return pickle.lo
	ads(data)
	
	@app.route("/com
	mand_injection",
	 methods=["GET"])
	def command_i
	njection():
	    command =
	 request.args.get("
	cmd")
	    return subproces
	s.Popen(command,
	 shell=True)
	
	def weak_crypto():
	    password =
	 "secret_password"
	    return hashlib.m
	d5(password.encode
	()).hexdigest()
	
	PASSWORD = "super_se
	cret_password123"</pre>	

任務	Description	所需的技能
	2. 開啟編碼助理。在聊天中請求下列動作：掃描目前的指令碼並告訴我結果。	

使用 Terraform 和 Checkov 的程式碼分析範例

任務	Description	所需的技能
執行程式碼分析。	若要使用 Terraform 和 Checkov 執行程式碼分析，請使用下列步驟： 1. `terraform_test.tf` 使用下列內容建立名為 的新 Terraform 檔案： `resource "aws_s3_bucket" "insecure_bucket" { bucket = "my-insecure-bucket" acl = "public-read" } resource "aws_security_group" "wide_open" { name = "allow_all" description = "Allow all inbound traffic" ingress { from_port = 0`	應用程式開發人員

任務	Description	所需的技能

	<pre>to_port = 0
	protocol = "-1"
	cidr_blocks = ["0.0.0.0/0"]
	}
	}</pre>
	2. 開啟編碼助理。在聊天中請求下列動作：掃描目前的指令碼並告訴我結果。	

進階掃描功能的範例

任務	Description	所需的技能
執行目標掃描。	以下是可用來執行目標掃描的請求範例： • 掃描目前的專案並告訴我結果。 • 掃描目前指令碼中的第 6-10 行。	應用程式開發人員
使用安全掃描與程式碼產生。	若要使用程式碼產生迴圈來解決安全性問題清單，請使用下列步驟（此範例使用 Kiro 作為編碼助理）： 1. 在 Kiro 中，詢問：在 Terraform 中建立 DynamoDB 資料表，並使用安全掃描器掃描程式碼。	應用程式開發人員

任務	Description	所需的技能
	2. 在 Kiro 中，詢問：檢閱產生的程式碼和安全性掃描結果，並遵循解決安全性問題清單的程式碼產生迴圈。	

疑難排解

問題	解決方案
環境設定問題	• 確認已安裝 Python 3.10 版或更新版本。 • 確定套件uv管理員已安裝。
掃描器問題	• 驗證是否支援檔案格式。 • 檢查檔案語法是否有效。 • 確保使用適當的副檔名。 • 檢閱掃描逾時設定。
整合問題	• 確認 MCP 伺服器正在執行。 • 檢查組態檔案是否正確。 • 驗證 API 端點。 • 確保回應格式有效。
其他支援	• 檢閱日誌以取得詳細的錯誤訊息。 • 檢查工具特定的文件。 • 在儲存庫中建立問題。

相關資源

AWS 文件

- 基礎設施即程式碼 (AWS 白皮書介紹 DevOps on AWS)

其他 AWS 資源

- 安全性、身分與合規的最佳實務

其他資源

- Bandit 文件
- Checkov 文件
- 模型內容通訊協定 (MCP) 文件
- OWASP 安全編碼實務 (OWASP Foundation 網站）
- Semgrep 文件

其他資訊

啟用自動核准的範例 MCP 組態

如果未autoApprove設定，使用者必須授予許可，才能將程式碼傳送至 MCP 安全伺服器進行掃描。設定 autoApprove 時，程式碼助理無需使用者核准即可叫用工具。這些工具會在機器本機執行，不會傳送任何資料，而且只會執行程式碼掃描。

下列組態可自動執行所有安全掃描函數：

```
{
  "mcpServers": {
    "security-scanner": {
      "command": "uvx",
      "args": [
        "--from",
        "git+https://github.com/aws-samples/sample-mcp-security-scanner.git@main",
        "security_scanner_mcp_server"
      ],
      "env": {
        "FASTMCP_LOG_LEVEL": "ERROR"
      },
      "disabled": false,
      "autoApprove": [
        "scan_with_checkov",
        "scan_with_semgrep",
        "scan_with_bandit",
        "get_supported_formats"
```

```
      ]
    }
  }
}
```

若要啟用偵錯記錄，請將 "FASTMCP_LOG_LEVEL"設定為 "DEBUG"。

安全性掃描工具支援的檔案格式

此解決方案中的每個安全掃描工具都支援下列檔案格式：

Checkov (IaC)

- Terraform – .tf、.tfvars、.tfstate
- CloudFormation – .yaml、.yml、.json、.template
- Kubernetes – .yaml、.yml
- Dockerfile – Dockerfile
- ARM – .json (Azure Resource Manager)
- Bicep – .bicep
- 無伺服器 – .yml、.yaml
- Helm – .yaml、.yml、.tpl
- GitHub 動作 – .yml、.yaml
- GitLab_ci – .yml、.yaml
- Ansible – .yml、.yaml

Semgrep （原始程式碼）

- Python – .py
- JavaScript – .js
- TypeScript – .ts
- Java – .java
- Go – .go
- C – .c
- C++ – .cpp
- C# – .cs

- Ruby – .rb
- PHP – .php
- Scala – .scala
- Kotlin – .kt
- Rust – .rs

Bandit（僅限 Python）

- Python – .py

示範

針對程式碼掃描，請向 AI 助理嘗試下列範例提示：

- 「掃描目前的指令碼並告訴我結果。」
- 「掃描第 20-60 行並告訴我結果。」
- 「掃描此 Amazon DynamoDB 資料表資源並告訴我結果。」

如需詳細資訊，請參閱此模式 GitHub 儲存庫中的此程式碼掃描示範。

若要產生安全程式碼，請嘗試下列範例提示：

- 「產生 Terraform 組態以建立已啟用加密的 DynamoDB 資料表，並掃描是否有安全問題。」
- 「建立寫入 DynamoDB 的 Python Lambda 函數，並掃描是否有漏洞。」
- 「使用適當的安全性設定產生 S3 儲存貯體的 CloudFormation 範本，並確認它通過安全性檢查。」
- 「撰寫 Python 指令碼以使用分頁查詢 DynamoDB，並掃描安全性最佳實務。」
- 「為具有安全強化的微服務建立 Kubernetes 部署資訊清單並進行驗證。」

如需詳細資訊，請參閱此模式 GitHub 儲存庫中的此程式碼產生與安全性掃描示範。

使用 RAG 和 ReAct 提示，開發進階生成式 AI 聊天式助理

Praveen Kumar Jeyarajan、Shai Cao、Noah Hamilton、Kiowa Jackson、Jundong Qiao 和 Kara Yang、Amazon Web Services

摘要

典型的公司有 70% 的資料被困在孤立系統中。您可以使用生成式 AI 支援的聊天式助理，透過自然語言互動來釋放這些資料孤島之間的洞見和關係。為了充分利用生成式 AI，輸出必須值得信任、準確且包含可用的公司資料。成功的聊天式助理取決於下列項目：

- 生成式 AI 模型（例如 Anthropic Claude 2)
- 資料來源向量化
- 進階推理技巧，例如 ReAct 架構，用於提示模型

此模式提供來自資料來源的資料擷取方法，例如 Amazon Simple Storage Service (Amazon S3) 儲存貯體、AWS Glue 和 Amazon Relational Database Service (Amazon RDS)。透過將擷取增強生成 (RAG) 與chain-of-thought方法相交，從該資料中取得值。結果支援以聊天為基礎的複雜助理對話，這些對話會利用您公司儲存的所有資料。

此模式使用 Amazon SageMaker 手冊和定價資料表作為範例，以探索生成式 AI 聊天式助理的功能。您將建置聊天式助理，藉由回答有關定價和服務功能的問題，協助客戶評估 SageMaker 服務。解決方案使用 Streamlit 程式庫來建置前端應用程式，並使用 LangChain 架構來開發採用大型語言模型 (LLM) 的應用程式後端。

對聊天式助理的查詢會符合初始意圖分類，以路由至三個可能工作流程之一。最複雜的工作流程結合了一般諮詢指引與複雜的定價分析。您可以調整模式以符合企業、公司和工業使用案例。

先決條件和限制

先決條件

- 安裝並設定 AWS Command Line Interface (AWS CLI)
- 安裝和設定 AWS Cloud Development Kit (AWS CDK) Toolkit 2.114.1 或更新版本
- Python 和 AWS CDK 的基本熟悉度
- 已安裝 Git
- 已安裝 Docker
- Python 3.11 或更新版本已安裝並設定（如需詳細資訊，請參閱工具一節）

- 使用 AWS CDK 引導的作用中 AWS 帳戶 https://docs.aws.amazon.com/cdk/v2/guide/bootstrapping.html
- Amazon Bedrock 服務中啟用了 Amazon Titan 和 Anthropic Claude 模型存取
- 在終端機環境中正確設定 AWS 安全登入資料，包括 。 `AWS_ACCESS_KEY_ID`

限制

- LangChain 不支援每個 LLM 進行串流。支援 Anthropic Claude 模型，但不支援 AI21 實驗室的模型。
- 此解決方案會部署到單一 AWS 帳戶。
- 此解決方案只能在可使用 Amazon Bedrock 和 Amazon Kendra 的 AWS 區域中部署。如需可用性的詳細資訊，請參閱 Amazon Bedrock 和 Amazon Kendra 的文件。

產品版本

- Python 3.11 版或更新版本
- 串流 1.30.0 版或更新版本
- Streamlit-chat 0.1.1 版或更新版本
- LangChain 0.1.12 版或更新版本
- AWS CDK 2.132.1 版或更新版本

Architecture

目標技術堆疊

- Amazon Athena
- Amazon Bedrock
- Amazon Elastic Container Service (Amazon ECS)
- AWS Glue
- AWS Lambda
- Amazon S3
- Amazon Kendra
- Elastic Load Balancing

目標架構

AWS CDK 程式碼會部署在 AWS 帳戶中設定聊天式助理應用程式所需的所有資源。下圖中顯示的聊天式助理應用程式旨在回答來自使用者的 SageMaker 相關查詢。使用者透過 Application Load Balancer 連線至 VPC，其中包含託管 Streamlit 應用程式的 Amazon ECS 叢集。協同運作 Lambda 函數會連線至應用程式。S3 儲存貯體資料來源透過 Amazon Kendra 和 AWS Glue 將資料提供給 Lambda 函數。Lambda 函數會連線至 Amazon Bedrock，以回應聊天式助理使用者的查詢 （問題）。

1. 協同運作 Lambda 函數會將 LLM 提示請求傳送至 Amazon Bedrock 模型 (Claude 2)。
2. Amazon Bedrock 會將 LLM 回應傳回協調 Lambda 函數。

協同運作 Lambda 函數內的邏輯流程

當使用者透過 Streamlit 應用程式提出問題時，它會直接叫用協調 Lambda 函數。下圖顯示叫用 Lambda 函數時的邏輯流程。

- 步驟 1 – 輸入 `query`（問題） 分類為三個意圖之一：
  - 一般 SageMaker 指引問題
  - 一般 SageMaker 定價 （訓練/推論） 問題
  - 與 SageMaker 和定價相關的複雜問題
- 步驟 2 – 輸入會`query`啟動以下三種服務之一：
  - `RAG Retrieval service`，它會從 Amazon Kendra 向量資料庫擷取相關內容，並透過 Amazon Bedrock 呼叫 LLM，以摘要擷取的內容作為回應。
  - `Database Query service`，其使用 - 相關資料表的 LLM、資料庫中繼資料和範例資料列，將輸入轉換為 SQL `query` 查詢。資料庫查詢服務會透過 Amazon Athena 對 SageMaker 定價資料庫執行 SQL 查詢，並將查詢結果摘要為回應。
  - `In-context ReACT Agent service`，這會先將輸入細分為`query`多個步驟，再提供回應。代理程式使用 `RAG Retrieval service`和 `Database Query service`做為工具，在推理過程中擷取相關資訊。推理和動作程序完成後，客服人員會產生最終答案作為回應。
- 步驟 3 – 協同運作 Lambda 函數的回應會以輸出形式傳送至 Streamlit 應用程式。

工具

AWS 服務

- Amazon Athena 是一種互動式查詢服務，可協助您使用標準 SQL 直接在 Amazon Simple Storage Service (Amazon S3) 中分析資料。
- Amazon Bedrock 是一項全受管服務，可讓您透過統一 API 使用來自領導 AI 新創公司的高效能基礎模型 (FMs) 和 Amazon。
- AWS 雲端開發套件 (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- Amazon Elastic Container Service (Amazon ECS) 是快速、可擴展的容器管理服務，可協助您執行、停止和管理叢集上的容器。
- AWS Glue 是全受管的擷取、轉換和載入 (ETL) 服務。它可協助您可靠地分類、清理、擴充和移動資料存放區和資料串流之間的資料。此模式使用 AWS Glue 爬蟲程式和 AWS Glue Data Catalog 資料表。
- Amazon Kendra 是一種智慧型搜尋服務，使用自然語言處理和進階機器學習演算法，傳回從資料搜尋問題的特定答案。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Elastic Load Balancing (ELB) 會將傳入的應用程式或網路流量分散到多個目標。例如，您可以在一或多個可用區域中將流量分配到 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、容器和 IP 地址。

程式碼儲存庫

此模式的程式碼可在 GitHub genai-bedrock-chatbot 儲存庫中使用。

程式碼儲存庫包含下列檔案和資料夾：

- `assets` 資料夾 – 架構圖表和公有資料集的靜態資產
- `code/lambda-container` 資料夾 – 在 Lambda 函數中執行的 Python 程式碼
- `code/streamlit-app` 資料夾 – 在 Amazon ECS 中做為容器映像執行的 Python 程式碼
- `tests` 資料夾 – 執行以單元測試 AWS CDK 建構的 Python 檔案
- `code/code_stack.py` – 用來建立 AWS 資源的 AWS CDK 建構 Python 檔案
- `app.py` – 用於在目標 AWS 帳戶中部署 AWS 資源的 AWS CDK 堆疊 Python 檔案

- `requirements.txt` – 必須為 AWS CDK 安裝的所有 Python 相依性清單
- `requirements-dev.txt` – 必須安裝才能讓 AWS CDK 執行單元測試套件的所有 Python 相依性清單
- `cdk.json` – 輸入檔案，用來提供啟動資源所需的值

注意：AWS CDK 程式碼使用 L3 （第 3 層） 建構和 AWS Identity and Access Management (IAM) 政策來部署解決方案。

最佳實務

- 此處提供的程式碼範例僅適用於proof-of-concept(PoC) 或試行示範。如果您想要將程式碼帶入生產環境，請務必使用下列最佳實務：
  - Amazon S3 存取記錄已啟用。
  - VPC 流程日誌已啟用。
  - Amazon Kendra Enterprise Edition 索引已啟用。
- 設定 Lambda 函數的監控和提醒。如需詳細資訊，請參閱監控和疑難排解 Lambda 函數。如需使用 Lambda 函數的一般最佳實務，請參閱 AWS 文件。

史詩

在本機電腦上設定 AWS 登入資料

任務	Description	所需的技能

匯出要部署堆疊之帳戶和 AWS 區域的變數。	若要使用環境變數為 AWS CDK 提供 AWS 登入資料，請執行下列命令。
	`export CDK_DEFAU LT_ACCOUNT=<12 Digit AWS Account Number>`
	`export CDK_DEFAU LT_REGION=<region>`
		DevOps 工程師，AWS DevOps

任務	Description	所需的技能
設定 AWS CLI 設定檔。	若要設定帳戶的 AWS CLI 設定檔，請遵循 AWS 文件中的指示。	DevOps 工程師，AWS DevOps

設定您的環境

任務	Description	所需的技能
在本機電腦上複製儲存庫。	若要複製儲存庫，請在終端機中執行下列命令。 `git clone https://github.com/awslabs/genai-bedrock-chatbot.git`	DevOps 工程師，AWS DevOps

設定 Python 虛擬環境並安裝必要的相依性。	若要設定 Python 虛擬環境，請執行下列命令。
	`cd genai-bedrock-chatbot`
	`python3 -m venv .venv`
	`source .venv/bin/activate`
	若要設定所需的相依性，請執行下列命令。
	`pip3 install -r requirements.txt`
		DevOps 工程師，AWS DevOps

設定 AWS CDK 環境並合成 AWS CDK 程式碼。	1. 若要在您的 AWS 帳戶中設定 AWS CDK 環境，請執行下列命令。	DevOps 工程師，AWS DevOps

任務	Description	所需的技能
	`cdk bootstrap aws://ACCOUNT-NUMBER/REGION` 2. 若要將程式碼轉換為 AWS CloudFormation 堆疊組態，請執行命令 `cdk synth`。	

設定和部署聊天式助理應用程式

任務	Description	所需的技能
佈建 Claude 模型存取。	若要為您的 AWS 帳戶啟用 Anthropic Claude 模型存取，請遵循 Amazon Bedrock 文件中的指示。	AWS DevOps
在帳戶中部署資源。	若要使用 AWS CDK 在 AWS 帳戶中部署資源，請執行下列動作： 1. 在複製儲存庫的根目錄中，在 `cdk.json` 檔案中提供`logging`參數的輸入。範例值為 `INFO`、`WARN`、`DEBUG`和 `ERROR`。 這些值定義 Lambda 函數和 Streamlit 應用程式的日誌層級訊息。 2. 複製儲存庫根中的 `app.py` 檔案包含用於部署的 AWS CloudFormation 堆疊名稱。預設堆疊名稱為 `chatbot-stack` 。	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	3. 若要部署資源，請執行命令 `cdk deploy`。 `cdk deploy` 命令使用 L3 建構模組建立多個 Lambda 函數，將文件和 CSV 資料集檔案複製到 S3 儲存貯體。 4. 命令完成後，請登入 AWS 管理主控台，開啟 CloudFormation 主控台，並檢閱堆疊是否已成功部署。 成功部署後，您可以使用 CloudFormation Outputs 區段中提供的 URL 來存取聊天式助理應用程式。	

任務	Description	所需的技能
執行 AWS Glue 爬蟲程式並建立資料目錄資料表。	AWS Glue 爬蟲程式用於保持資料結構描述動態。解決方案會隨需執行爬蟲程式，在 AWS Glue Data Catalog 資料表中建立和更新分割區。將 CSV 資料集檔案複製到 S3 儲存貯體後，請執行 AWS Glue 爬蟲程式並建立 Data Catalog 資料表結構描述以進行測試： 1. 導覽至 AWS Glue 主控台。 2. 在導覽窗格中的資料目錄下，選擇爬蟲程式。 3. 選取字尾為 的爬蟲程式`sagemaker-pricing-crawler` 。 4. 執行爬蟲程式。 5. 爬蟲程式成功執行後，會建立 AWS Glue Data Catalog 資料表。 **Note** AWS CDK 程式碼會將 AWS Glue 爬蟲程式設定為隨需執行，但您也可以排定它定期執行。	DevOps 工程師，AWS DevOps

任務	Description	所需的技能
啟動文件索引。	將檔案複製到 S3 儲存貯體後，請使用 Amazon Kendra 來編目和編製索引： 1. 導覽至 Amazon Kendra 主控台。 2. 選取尾碼為 的索引`chatbot-index` 。 3. 在導覽窗格中，選擇資料來源，然後使用尾碼 選取資料來源連接器`chatbot-index` 。 4. 選擇立即同步以啟動索引程序。 **Note** AWS CDK 程式碼會將 Amazon Kendra 索引同步設定為隨需執行，但您也可以使用排程參數定期執行。	AWS DevOps，DevOps 工程師

清除解決方案中的所有 AWS 資源

任務	Description	所需的技能
移除 AWS 資源。	測試解決方案之後，請清除資源： 1. 若要移除解決方案部署的 AWS 資源，請執行命令`cdk destroy`。	DevOps 工程師，AWS DevOps

任務	Description	所需的技能
	2. 刪除兩個 S3 儲存貯體中的所有物件，然後移除儲存貯體。 如需詳細資訊，請參閱刪除儲存貯體。	

疑難排解

問題	解決方案
AWS CDK 傳回錯誤。	如需 AWS CDK 問題的協助，請參閱疑難排解常見的 AWS CDK 問題。

相關資源

- Amazon Bedrock：
  - 模型存取
  - 基礎模型的推論參數
- 使用 Python 建置 Lambda 函數
- 開始使用 AWS CDK
- 在 Python 中使用 AWS CDK
- AWS 上的生成式 AI 應用程式建置器
- LangChain 文件
- 串流文件

其他資訊

AWS CDK 命令

使用 AWS CDK 時，請記住下列有用的命令：

- 列出應用程式中的所有堆疊

```
cdk ls
```

- 發出合成的 AWS CloudFormation 範本

```
cdk synth
```

- 將堆疊部署到您的預設 AWS 帳戶和區域

```
cdk deploy
```

- 比較已部署堆疊與目前狀態

```
cdk diff
```

- 開啟 AWS CDK 文件

```
cdk docs
```

- 刪除 CloudFormation 堆疊並移除 AWS 部署的資源

```
cdk destroy
```

使用 Amazon Bedrock 代理程式和知識庫開發全自動聊天式助理

Jundong Qiao、Shai Cao、Noah Hamilton、Kiowa Jackson、Praveen Kumar Jeyarajan 和 Kara Yang、Amazon Web Services

摘要

許多組織在建立能夠協調各種資料來源以提供完整答案的聊天式助理時，都會面臨挑戰。此模式提供一種解決方案，用於開發聊天式助理，能夠以直接的部署來回應文件和資料庫的查詢。

從 Amazon Bedrock 開始，這項全受管生成式人工智慧 (AI) 服務提供廣泛的進階基礎模型 (FMs)。這有助於高效建立生成式 AI 應用程式，並高度重視隱私權和安全性。在文件擷取的內容中，擷取增強生成 (RAG) 是一種樞紐功能。它使用知識庫，透過來自外部來源的內容相關資訊來增強 FM 提示。Amazon OpenSearch Serverless 索引做為 Amazon Bedrock 知識庫後方的向量資料庫。此整合是透過仔細的提示詞工程來增強，以將不正確的情況降至最低，並確保回應錨定在實際文件中。對於資料庫查詢，Amazon Bedrock FMs 會將文字查詢轉換為結構化 SQL 查詢，並結合特定參數。這可讓您從資料庫管理的AWS Glue 資料庫精確擷取資料。Amazon Athena 用於這些查詢。

為了處理更複雜的查詢，實現全面的答案需要來自文件和資料庫的資訊。Amazon Bedrock 的代理程式是一項生成式 AI 功能，可協助您建置自主代理程式，了解複雜的任務並將其分解為更簡單的協同運作任務。由 Amazon Bedrock 自動代理程式協助，從簡化任務中擷取的洞見組合可增強資訊的合成，進而產生更徹底且詳盡的答案。此模式示範如何在自動化解決方案中使用 Amazon Bedrock 和相關的生成式 AI 服務和功能來建置聊天式助理。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- Docker，已安裝
- AWS Cloud Development Kit (AWS CDK)，已安裝並引導至 `us-east-1`或 `us-west-2` AWS 區域
- AWS CDK 已安裝 Toolkit 2.114.1 版或更新版本
- AWS Command Line Interface (AWS CLI)，已安裝並設定
- Python 3.11 版或更新版本，已安裝
- 在 Amazon Bedrock 中，啟用對 Claude 2、Claude 2.1、Claude Instant 和 Titan Embeddings G1 的存取 – 文字 G1

限制

- 此解決方案會部署到單一 AWS 帳戶。
- 此解決方案只能在支援 Amazon Bedrock 和 Amazon OpenSearch Serverless AWS 區域 的 中部署。如需詳細資訊，請參閱 Amazon Bedrock 和 Amazon OpenSearch Serverless 的文件。

產品版本

- Llama-index 0.10.6 版或更新版本
- Sqlalchemy 2.0.23 版或更新版本
- Opensearch-py 2.4.2 版或更新版本
- Requests_aws4auth 1.2.3 版或更新版本
- 適用於 Python (Boto3) 的 AWS SDK 1.34.57 版或更新版本

Architecture

目標技術堆疊

AWS Cloud Development Kit (AWS CDK) 是開放原始碼軟體開發架構，用於在程式碼中定義雲端基礎設施並透過其佈建 AWS CloudFormation。此模式中使用的 AWS CDK 堆疊會部署下列 AWS 資源：

- AWS Key Management Service (AWS KMS)
- Amazon Simple Storage Service (Amazon S3)
- AWS Glue Data Catalog，適用於 AWS Glue 資料庫元件
- AWS Lambda
- AWS Identity and Access Management (IAM)
- Amazon OpenSearch Serverless
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon Elastic Container Service (Amazon ECS)
- AWS Fargate
- Amazon Virtual Private Cloud (Amazon VPC)
- Application Load Balancer

目標架構

圖表顯示使用多個 AWS 區域的單一 AWS 中全面的雲端原生設定 AWS 服務。聊天式助理的主要界面是在 Amazon ECS 叢集上託管的 Streamlit 應用程式。 Application Load Balancer 會管理可存取性。透過此界面進行的查詢會啟用 `Invocation` Lambda 函數，然後與 Amazon Bedrock 的代理程式連接。此代理程式會透過諮詢 Amazon Bedrock 的知識庫或叫用 `Agent executor` Lambda 函數來回應使用者查詢。此函數會在預先定義的 API 結構描述之後，觸發與代理程式相關聯的一組動作。Amazon Bedrock 的知識庫使用 OpenSearch Serverless 索引作為向量資料庫基礎。此外，`Agent executor`函數會產生透過 Amazon Athena 針對 AWS Glue 資料庫執行的 SQL 查詢。

工具

AWS 服務

- Amazon Athena 是一種互動式查詢服務，可協助您使用標準 SQL 直接在 Amazon Simple Storage Service (Amazon S3) 中分析資料。
- Amazon Bedrock 是一項全受管服務，可讓您透過統一 API 使用來自領導 AI 新創公司的高效能基礎模型 (FMs) 和 Amazon。
- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- Amazon Elastic Container Service (Amazon ECS) 是快速、可擴展的容器管理服務，可協助您執行、停止和管理叢集上的容器。
- Elastic Load Balancing 會將傳入的應用程式或網路流量分散到多個目標。例如，您可以將流量分散到一或多個可用區域中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、容器和 IP 地址。
- AWS Glue 是全受管擷取、轉換和載入 (ETL) 服務。它可協助您可靠地分類、清理、擴充和移動資料存放區和資料串流之間的資料。此模式使用 AWS Glue 爬蟲程式和 AWS Glue Data Catalog 資料表。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon OpenSearch Serverless 是 Amazon OpenSearch Service 的隨需無伺服器組態。在此模式中，OpenSearch Serverless 索引可做為 Amazon Bedrock 知識庫的向量資料庫。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他工具

- Streamlit 是一種開放原始碼 Python 架構，可用來建立資料應用程式。

程式碼儲存庫

此模式的程式碼可在 GitHub genai-bedrock-agent-chatbot 儲存庫中使用。程式碼儲存庫包含下列檔案和資料夾：

- `assets` 資料夾 – 靜態資產，例如架構圖表和公有資料集。
- `code/lambdas/action-lambda` 資料夾 – Lambda 函數的 Python 程式碼，可做為 Amazon Bedrock 代理程式的動作。
- `code/lambdas/create-index-lambda` 資料夾 – 建立 OpenSearch Serverless 索引之 Lambda 函數的 Python 程式碼。
- `code/lambdas/invoke-lambda` 資料夾 – 叫用 Amazon Bedrock 代理程式的 Lambda 函數的 Python 程式碼，該代理程式直接從 Streamlit 應用程式呼叫。
- `code/lambdas/update-lambda` 資料夾 – Lambda 函數的 Python 程式碼，會在透過 部署資源之後更新或刪除 AWS 資源 AWS CDK。
- `code/layers/boto3_layer` 資料夾 – 建立 Boto3 layer 的 AWS CDK 堆疊，在所有 Lambda 函數之間共用。
- `code/layers/opensearch_layer` 資料夾 – 建立 OpenSearch Serverless layer 的 AWS CDK 堆疊，會安裝所有相依性來建立索引。
- `code/streamlit-app` 資料夾 – 在 Amazon ECS 中做為容器映像執行的 Python 程式碼。
- `code/code_stack.py` – AWS CDK 建立 AWS 資源的建構 Python 檔案。
- `app.py` – 在目標 AWS 帳戶中部署 AWS 資源的 AWS CDK 堆疊 Python 檔案。
- `requirements.txt` – 必須為 安裝的所有 Python 相依性清單 AWS CDK。
- `cdk.json` – 輸入檔案，用來提供建立資源所需的值。此外，您可以在`context/config`欄位中相應地自訂解決方案。如需自訂的詳細資訊，請參閱其他資訊一節。

最佳實務

- 此處提供的程式碼範例僅用於proof-of-concept(PoC) 或試行目的。如果您想要將程式碼帶入生產環境，請務必使用下列最佳實務：
  - 啟用 Amazon S3 存取記錄

  - 啟用 VPC 流程日誌
- 設定 Lambda 函數的監控和提醒。如需詳細資訊，請參閱監控和疑難排解 Lambda 函數。如需最佳實務，請參閱使用 AWS Lambda 函數的最佳實務。

史詩

在本機工作站上設定 AWS 登入資料

任務	Description	所需的技能
匯出帳戶和區域的變數。	若要使用環境變數提供 的 AWS 登入 AWS CDK 資料，請執行下列命令。 `export CDK_DEFAU LT_ACCOUNT=<12-digit AWS account number> export CDK_DEFAU LT_REGION=<Region>`	AWS DevOps，DevOps 工程師
設定 AWS CLI 具名設定檔。	若要為帳戶設定 AWS CLI 具名設定檔，請遵循組態和登入資料檔案設定中的指示。	AWS DevOps，DevOps 工程師

設定您的環境

任務	Description	所需的技能
將儲存庫複製到您的本機工作站。	若要複製儲存庫，請在終端機中執行下列命令。 `git clone https://g ithub.com/awslabs/ genai-bedrock-agent- chatbot.git`	DevOps 工程師，AWS DevOps

任務	Description	所需的技能

設定 Python 虛擬環境。	若要設定 Python 虛擬環境，請執行下列命令。
	`cd genai-bedrock-agent-chatbot`
	`python3 -m venv .venv`
	`source .venv/bin/activate`
	若要設定所需的相依性，請執行下列命令。
	`pip3 install -r requirements.txt`
		DevOps 工程師，AWS DevOps

設定 AWS CDK 環境。	若要將程式碼轉換為 AWS CloudFormation 範本，請執行命令 `cdk synth`。	AWS DevOps，DevOps 工程師

設定和部署應用程式

任務	Description	所需的技能
在帳戶中部署資源。	若要 AWS 帳戶 使用 在 中部署資源 AWS CDK，請執行下列動作： 1. 在複製儲存庫的根目錄中，在 `cdk.json` 檔案中，提供記錄參數的輸入。範例值為 `INFO`、`WARN`、 `DEBUG`和 `ERROR`。	DevOps 工程師，AWS DevOps

任務	Description	所需的技能
	這些值定義 Lambda 函數和 Streamlit 應用程式的日誌層級訊息。 2. 複製儲存庫根中的 `cdk.json` 檔案包含用於部署的 AWS CloudFormation 堆疊名稱。預設堆疊名稱為 `chatbot-stack` 。預設 Amazon Bedrock 代理程式名稱為 `ChatbotBedrockAgent` ，預設 Amazon Bedrock 代理程式別名為 `Chatbot_Agent` 。 3. 若要部署資源，請執行命令 `cdk deploy`。此`cdk deploy`命令使用 layer-3 建構模組來建立多個 Lambda 函數，以將文件和 CSV 資料集檔案複製到 S3 儲存貯體。它也會為 Amazon Bedrock 代理程式部署 Amazon Bedrock 代理程式、知識庫和 `Action group` Lambda 函數。 4. 登入 AWS 管理主控台，然後開啟 CloudFormation 主控台。 5. 確認堆疊已成功部署。如需說明，請參閱在 AWS CloudFormation 主控台上檢閱您的堆疊。	

任務	Description	所需的技能
	成功部署後，您可以使用 CloudFormation 主控台的輸出索引標籤上提供的 URL 來存取聊天式助理應用程式。	

清除解決方案中的所有 AWS 資源

任務	Description	所需的技能
移除 AWS 資源。	測試解決方案之後，若要清除資源，請執行命令 `cdk destroy`。	AWS DevOps，DevOps 工程師

相關資源

AWS 文件

- Amazon Bedrock 資源：
  - 模型存取
  - 基礎模型的推論參數
  - Amazon Bedrock 的代理程式
  - Amazon Bedrock 的知識庫
- 使用 Python 建置 Lambda 函數
- AWS CDK 資源：
  - 開始使用 AWS CDK
  - 對 AWS CDK 常見問題進行故障診斷
  - 在 Python AWS CDK 中使用
- 上的生成式 AI 應用程式建置器 AWS

其他 AWS 資源

- Amazon OpenSearch Serverless 的向量引擎

其他資源

- LlamaIndex 文件
- 串流文件

其他資訊

使用您自己的資料自訂聊天式助理

若要整合自訂資料以部署解決方案，請遵循這些結構化準則。這些步驟旨在確保無縫且有效率的整合程序，讓您能夠使用自訂資料有效地部署解決方案。

用於知識庫資料整合

資料準備

1. 找到 `assets/knowledgebase_data_source/`目錄。
2. 將資料集放在此資料夾中。

組態調整

1. 開啟 `cdk.json` 檔案。
2. 導覽至 `context/configure/paths/knowledgebase_file_name` 欄位，然後相應地更新。
3. 導覽至 `bedrock_instructions/knowledgebase_instruction` 欄位，然後更新它，以準確反映新資料集的細微差別和內容。

用於結構資料整合

資料組織

1. 在 `assets/data_query_data_source/`目錄中，建立子目錄，例如 `tabular_data`。
2. 將您的結構化資料集 （ 可接受的格式包括 CSV、JSON、ORC 和 Parquet) 放入這個新建立的子資料夾。
3. 如果您要連線到現有的資料庫，請在 `create_sql_engine()`中更新 函數`code/lambda/action-lambda/build_query_engine.py`以連線到您的資料庫。

組態和程式碼更新

1. 在 cdk.json 檔案中，更新 context/configure/paths/athena_table_data_prefix 欄位以符合新的資料路徑。
2. 透過整合與資料集對應的新text-to-SQL範例code/lambda/action-lambda/dynamic_examples.csv進行修訂。
3. 修改code/lambda/action-lambda/prompt_templates.py以鏡像結構化資料集的屬性。
4. 在 cdk.json 檔案中，更新 context/configure/bedrock_instructions/action_group_description 欄位以解釋 Action group Lambda 函數的目的和功能。
5. 在 assets/agent_api_schema/artifacts_schema.json 檔案中，說明 Action group Lambda 函數的新功能。

一般更新

在 cdk.json 檔案中，在 context/configure/bedrock_instructions/agent_instruction區段中，提供 Amazon Bedrock 代理程式預期功能和設計用途的完整描述，並將新整合的資料納入考量。

使用 Amazon Bedrock 和 Amazon Transcribe 從語音輸入記錄機構知識

Praveen Kumar Jeyarajan、Jundong Qiao、Rajiv Upadhyay 和 Megan Wu，Amazon Web Services

總結

擷取機構知識對於確保組織成功和彈性至關重要。機構知識代表員工隨著時間累積的集體智慧、洞見和經驗，通常在性質上隱含，並以非正式方式傳遞。這種豐富的資訊包含獨特的方法、最佳實務和解決方案，可處理可能未記錄在其他地方的複雜問題。透過正式化和記錄這些知識，公司可以保留機構記憶體、促進創新、增強決策過程，並加速新員工的學習曲線。此外，它可促進協作、賦予個人能力，並培養持續改善的文化。最後，利用機構知識可協助公司使用其最寶貴的資產，也就是人力資源的集體智慧，在動態商業環境中應對挑戰、推動成長並維持競爭優勢。

此模式說明如何透過資深員工的語音錄音來擷取機構知識。它使用 Amazon Transcribe 和 Amazon Bedrock 進行系統性文件和驗證。透過記錄這種非正式知識，您可以保留它並與後續的員工群組共用。此工作支援卓越營運，並透過整合透過直接體驗取得的實際知識來改善訓練計劃的有效性。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Docker，已安裝
- AWS 雲端開發套件 (AWS CDK) 2.114.1 版或更新版本，已安裝並引導至 `us-east-1` 或 `us-west-2` AWS 區域
- AWS CDK Toolkit 2.114.1 版或更新版本，已安裝
- AWS 命令列界面 (AWS CLI)，已安裝並設定
- Python 3.12 版或更新版本，已安裝
- 建立 Amazon Transcribe、Amazon Bedrock、Amazon Simple Storage Service (Amazon S3) 和 AWS Lambda 資源的許可

限制

- 此解決方案會部署到單一 AWS 帳戶。
- 此解決方案只能在可使用 Amazon Bedrock 和 Amazon Transcribe 的 AWS 區域中部署。如需可用性的相關資訊，請參閱 Amazon Bedrock 和 Amazon Transcribe 的文件。

- 音訊檔案的格式必須是 Amazon Transcribe 支援的格式。如需支援的格式清單，請參閱轉錄文件中的媒體格式。

產品版本

- 適用於 Python (Boto3) 的 AWS 開發套件 1.34.57 版或更新版本
- LangChain 0.1.12 版或更新版本

Architecture

架構代表 AWS 上的無伺服器工作流程。AWS Step Functions 協調 Lambda 函數以進行音訊處理、文字分析和文件產生。下圖顯示 Step Functions 工作流程，也稱為狀態機器。

狀態機器中的每個步驟都由不同的 Lambda 函數處理。以下是文件產生程序中的步驟：

1. `preprocess` Lambda 函數會驗證傳遞至 Step Functions 的輸入，並列出所提供 Amazon S3 URI 資料夾路徑中的所有音訊檔案。工作流程中的下游 Lambda 函數會使用檔案清單來驗證、摘要和產生文件。
2. `transcribe` Lambda 函數使用 Amazon Transcribe 將音訊檔案轉換為文字記錄。此 Lambda 函數負責啟動轉錄程序，並將語音準確轉換為文字，然後存放以供後續處理。
3. `validate` Lambda 函數會分析文字文字記錄，判斷回應與初始問題之間的相關性。透過 Amazon Bedrock 使用大型語言模型 (LLM)，可以識別主題上的答案，並將其與主題外回應分開。
4. `summarize` Lambda 函數使用 Amazon Bedrock 產生主題上答案的一致性和簡潔摘要。
5. `generate` Lambda 函數會將摘要組合成結構良好的文件。它可以根據預先定義的範本格式化文件，並包含任何其他必要的內容或資料。
6. 如果任何 Lambda 函數失敗，您會透過 Amazon Simple Notification Service (Amazon SNS) 收到電子郵件通知。

在此過程中，AWS Step Functions 會確保以正確的順序啟動每個 Lambda 函數。此狀態機器具有可平行處理的容量，可提高效率。Amazon S3 儲存貯體做為中央儲存儲存庫，透過管理涉及的各種媒體和文件格式來支援工作流程。

工具

AWS 服務

- Amazon Bedrock 是一項全受管服務，可讓您透過統一 API 使用來自領導 AI 新創公司的高效能基礎模型 (FMs) 和 Amazon。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務來建置業務關鍵應用程式。
- Amazon Transcribe 是一種自動語音辨識服務，使用機器學習模型將音訊轉換為文字。

其他工具

- LangChain 是一種架構，用於開發採用大型語言模型 (LLMs的應用程式。

程式碼儲存庫

此模式的程式碼可在 GitHub genai-knowledge-capture 儲存庫中使用。

程式碼儲存庫包含下列檔案和資料夾：

- `assets` 資料夾 – 解決方案的靜態資產，例如架構圖表和公有資料集
- `code/lambdas` 資料夾 – 所有 Lambda 函數的 Python 程式碼
  - `code/lambdas/generate` 資料夾 - 從 S3 儲存貯體中的摘要資料產生文件的 Python 程式碼
  - `code/lambdas/preprocess` 資料夾 - 處理 Step Functions 狀態機器輸入的 Python 程式碼
  - `code/lambdas/summarize` 資料夾 - 使用 Amazon Bedrock 服務摘要轉錄資料的 Python 程式碼
  - `code/lambdas/transcribe` 資料夾 - 使用 Amazon Transcribe 將語音資料 （ 音訊檔案 ） 轉換為文字的 Python 程式碼
  - `code/lambdas/validate` 資料夾 - 驗證所有答案是否與相同主題相關的 Python 程式碼
- `code/code_stack.py` – 用來建立 AWS 資源的 AWS CDK 建構 Python 檔案
- `app.py` – 用於在目標 AWS 帳戶中部署 AWS 資源的 AWS CDK 應用程式 Python 檔案
- `requirements.txt` – 必須為 AWS CDK 安裝的所有 Python 相依性清單

- `cdk.json` – 提供建立資源所需的值的輸入檔案

最佳實務

提供的程式碼範例僅用於proof-of-concept(PoC) 或試行目的。如果您想要將解決方案用於生產環境，請使用下列最佳實務：

- 啟用 Amazon S3 存取記錄
- 啟用 VPC 流程日誌

史詩

在本機工作站上設定 AWS 登入資料

任務	Description	所需的技能
匯出帳戶和 AWS 區域的變數。	若要使用環境變數提供 AWS CDK 的 AWS 登入資料，請執行下列命令。 `export CDK_DEFAU LT_ACCOUNT=<12-digit AWS account number> export CDK_DEFAU LT_REGION=<Region>`	AWS DevOps，DevOps 工程師
設定名為 的 AWS CLI 設定檔。	若要設定帳戶的 AWS CLI 命名設定檔，請遵循組態和登入資料檔案設定中的指示。	AWS DevOps，DevOps 工程師

設定您的環境

任務	Description	所需的技能
將儲存庫複製到您的本機工作站。	若要複製 genai-knowledge-capture 儲存庫，請在終端機中執行下列命令。 `git clone https://github.com/aws-samples/genai-knowledge-capture`	AWS DevOps，DevOps 工程師
（選用）取代音訊檔案。	若要自訂範例應用程式以整合您自己的資料，請執行下列動作： 1. 導覽至複製儲存庫中的 `assets/audio_samples` 資料夾。 2. 刪除包含範例音訊檔案的資料夾。 3. 為您要分析的每個主題建立資料夾。 4. 將您的音訊檔案傳輸到各自的資料夾。	AWS DevOps，DevOps 工程師

設定 Python 虛擬環境。	若要設定 Python 虛擬環境，請執行下列命令。
	`cd genai-knowledge-capture`
	`python3 -m venv .venv`
	`source .venv/bin/activate`
	`pip install -r requirements.txt`
		AWS DevOps，DevOps 工程師

任務	Description	所需的技能
合成 AWS CDK 程式碼。	若要將程式碼轉換為 AWS CloudFormation 堆疊組態，請執行下列命令。 `cdk synth`	AWS DevOps，DevOps 工程師

設定和部署解決方案

任務	Description	所需的技能
佈建基礎模型存取。	啟用存取您 AWS 帳戶的 Anthropic Claude 3 Sonnet 模型。如需說明，請參閱 Bedrock 文件中的新增模型存取權。	AWS DevOps
在帳戶中部署資源。	若要使用 AWS CDK 在 AWS 帳戶中部署資源，請執行下列動作： 1. （選用）在複製儲存庫的根目錄中，於 `app.py` 檔案中更新 AWS CloudFormation 堆疊名稱。預設堆疊名稱為 `genai-knowledge-capture-stack` 。 2. 若要部署資源，請執行命令 `cdk deploy`。 `cdk deploy` 命令使用 layer-3 建構來建立一組 Lambda 函數、S3 儲存貯體、Amazon SNS 主題和 Step Functions 狀態機器。	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	在部署期間，`assets/audio_samples` 資料夾中的音訊檔案會複製到 S3 儲存貯體。 3. 登入 AWS 管理主控台，然後開啟位於 https：//https://console.aws.amazon.com/cloudformation/ 的 CloudFormation 主控台。 4. 確認堆疊已成功部署。如需說明，請參閱 AWS CloudFormation 主控台上的檢閱堆疊。	
訂閱 Amazon SNS 主題。	若要訂閱 Amazon SNS 主題以進行通知，請執行下列動作： 1. 在 CloudFormation 主控台的導覽窗格中，選擇 Stacks。 2. 選擇`genai-knowledge-capture-stack` 堆疊。 3. 選擇 Output (輸出) 索引標籤。 4. 使用金鑰 尋找 Amazon SNS 主題名稱`SNSTopicName` 。 5. 依照訂閱電子郵件地址至 Amazon SNS 主題中的指示，設定電子郵件地址以接收通知。	一般 AWS

測試解決方案

任務	Description	所需的技能
執行狀態機器。	1. 開啟 Step Functions 主控台。 2. 在狀態機器頁面上，選擇 genai-knowledge-capture-stack-state-machine。 3. 選擇 Start execution (開始執行)。 4. （選用）在名稱方塊中，輸入執行的名稱。 5. 在輸入區域中，透過取代預留位置文字來輸入下列 JSON 物件，其中： • <Name> 是您想要命名文件的名稱。 • <S3 bucket name> 是包含音訊檔案的 Amazon S3 儲存貯體名稱。 • <Folder path> 是包含音訊檔案的目錄。 `{ "documentName": "<Name>", "audioFileFolderUri": "s3://<S3 bucket name>/<Folder path>" }` 6. 選擇 Start Execution (開始執行)。 7. 在執行詳細資訊頁面上，檢閱結果並等待執行完成。	應用程式開發人員，一般 AWS

清除解決方案中的所有 AWS 資源

任務	Description	所需的技能
移除 AWS 資源。	測試解決方案之後，請清除資源： 1. 從 S3 儲存貯體刪除所有物件，然後刪除儲存貯體。如需詳細資訊，請參閱刪除儲存貯體。 2. 從複製的儲存庫中，執行命令 `cdk destroy`。	AWS DevOps，DevOps 工程師

相關資源

AWS 文件

- Amazon Bedrock 資源：
  - 模型存取
  - 基礎模型的推論參數
- AWS CDK 資源：
  - 開始使用 AWS CDK
  - 在 Python 中使用 AWS CDK
  - 疑難排解常見的 AWS CDK 問題
  - 工具組命令
- AWS Step Functions 資源：
  - AWS Step Functions 入門
  - 疑難排解
- 使用 Python 建置 Lambda 函數
- AWS 上的生成式 AI 應用程式建置器

其他資源

- LangChain 文件

使用 Amazon Personalize 產生個人化和重新排名的建議

Mason Cahill、Matthew Chasse 和 Tayo Olajide，Amazon Web Services

總結

此模式說明如何使用 Amazon Personalize，根據從這些使用者擷取即時使用者互動資料，為您的使用者產生個人化建議，包括重新排名的建議。此模式中使用的範例案例是以寵物採用網站為基礎，該網站會根據其互動為其使用者產生建議（例如，使用者造訪哪些寵物）。遵循範例案例，您將學習如何使用 Amazon Kinesis Data Streams 擷取互動資料、AWS Lambda 產生建議並重新排名建議，以及 Amazon Data Firehose 將資料存放在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中。您也會學習使用 AWS Step Functions 來建置狀態機器，以管理產生建議的解決方案版本（即訓練過的模型）。

先決條件和限制

先決條件

- 具有引導式 AWS 雲端開發套件 (AWS CDK) 的作用中 AWS 帳戶
- 具有已設定登入資料的 AWS Command Line Interface (AWS CLI)
- Python 3.9

產品版本

- Python 3.9
- AWS CDK 2.23.0 或更新版本
- AWS CLI 2.7.27 或更新版本

Architecture

技術堆疊

- Amazon Data Firehose
- Amazon Kinesis Data Streams
- Amazon Personalize
- Amazon Simple Storage Service (Amazon S3)

- AWS 雲端開發套件 (AWS CDK)
- AWS 命令列界面 (AWS CLI)
- AWS Lambda
- AWS Step Functions

目標架構

下圖說明將即時資料擷取至 Amazon Personalize 的管道。然後，管道會使用該資料為使用者產生個人化和重新排名的建議。

該圖顯示以下工作流程：

1. Kinesis Data Streams 會擷取即時使用者資料（例如，造訪寵物等事件），以供 Lambda 和 Firehose 處理。
2. Lambda 函數會處理來自 Kinesis Data Streams 的記錄，並發出 API 呼叫，將記錄中的使用者互動新增至 Amazon Personalize 中的事件追蹤器。
3. 以時間為基礎的規則會叫用 Step Functions 狀態機器，並使用 Amazon Personalize 中事件追蹤器的事件，為建議產生新的解決方案版本並重新排序模型。
4. 狀態機器會更新 Amazon Personalize 行銷活動，以使用新的解決方案版本。
5. Lambda 透過呼叫 Amazon Personalize 重新排名行銷活動來重新排名建議項目清單。
6. Lambda 會透過呼叫 Amazon Personalize 建議行銷活動來擷取建議項目的清單。
7. Firehose 會將事件儲存到 S3 儲存貯體，以做為歷史資料存取。

工具

AWS 工具

- AWS 雲端開發套件 (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- Amazon Data Firehose 可協助您將即時串流資料交付至其他 AWS 服務、自訂 HTTP 端點，以及受支援的第三方服務供應商所擁有的 HTTP 端點。

- Amazon Kinesis Data Streams 可協助您即時收集和處理大型資料記錄串流。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而不需要佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Personalize 是一項全受管機器學習 (ML) 服務，可協助您根據您的資料為使用者產生項目建議。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 Lambda 函數和其他 AWS 服務來建置業務關鍵應用程式。

其他工具

- pytest 是一種 Python 架構，用於撰寫小型且可讀取的測試。
- Python 是一種一般用途的電腦程式設計語言。

Code

此模式的程式碼可在 GitHub Animal Recommender 儲存庫中使用。您可以從此儲存庫使用 AWS CloudFormation 範本來部署範例解決方案的資源。

> **Note**
>
> Amazon Personalize 解決方案版本、事件追蹤器和行銷活動由在原生 CloudFormation 資源上擴展的自訂資源（基礎設施內）提供支援。

史詩

建立基礎設施

任務	Description	所需的技能
建立隔離的 Python 環境。	Mac/Linux 設定 1. 若要手動建立虛擬環境，請從終端機執行 `$ python3 -m venv .venv`命令。 2. 初始化程序完成後，請執行 `$ source .venv/bin/`	DevOps 工程師

任務	Description	所需的技能
	`activate` 命令來啟用虛擬環境。 Windows 設定 若要手動建立虛擬環境，請從終端機執行 `% .venv\Scripts\activate.bat` 命令。	

任務	Description	所需的技能
合成 CloudFormation 範本。	1. 若要安裝必要的相依性，請從終端機執行 `$ pip install -r requirements.txt` 命令。 2. 在 AWS CLI 中，設定下列環境變數： • `export ACCOUNT_ID=123456789` • `export CDK_DEPLOY_REGION=us-east-1` • `export CDK_ENVIRONMENT=dev` 3. 在 `config/{env}.yml` 檔案中，更新`vpcId` 以符合您的虛擬私有雲端 (VPC) ID。 4. 若要合成此程式碼的 CloudFormation 範本，請執行 `$ cdk synth`命令。 **Note** 在步驟 2 中，`CDK_ENVIRONMENT` 是指 `config/{env}.yml` 檔案。	DevOps 工程師

任務	Description	所需的技能
部署資源並建立基礎設施。	若要部署解決方案資源，請從終端機執行 `./deploy.sh` 命令。 此命令會安裝所需的 Python 相依性。Python 指令碼會建立 S3 儲存貯體和 AWS Key Management Service (AWS KMS) 金鑰，然後新增初始模型建立的種子資料。最後，指令碼會執行 `cdk deploy`來建立剩餘的基礎設施。 **Note** 初始模型訓練會在堆疊建立期間進行。堆疊最多可能需要兩個小時才能完成建立。	DevOps 工程師

相關資源

- Animal Recommender (GitHub)
- AWS CDK 參考文件
- Boto3 文件
- 使用 Amazon Personalize 最佳化您選擇的業務指標的個人化建議 (AWS Machine Learning 部落格)

其他資訊

範例承載和回應

建議 Lambda 函數

若要擷取建議，請使用下列格式的承載向建議 Lambda 函數提交請求：

```
{
  "userId": "3578196281679609099",
  "limit": 6
}
```

下列範例回應包含動物群組的清單：

```
[{"id": "1-domestic short hair-1-1"},
{"id": "1-domestic short hair-3-3"},
{"id": "1-domestic short hair-3-2"},
{"id": "1-domestic short hair-1-2"},
{"id": "1-domestic short hair-3-1"},
{"id": "2-beagle-3-3"},
```

如果您離開 `userId` 欄位，函數會傳回一般建議。

重新排序 Lambda 函數

若要使用重新排名，請提交請求至重新排名的 Lambda 函數。承載包含要重新排名的所有項目 IDs`userId`的 及其中繼資料。下列範例資料使用適用於 `animal_species_id`(1=貓，2=狗） 的 Oxford Pets 類別，以及適用於 `animal_age_id`和 的整數 1-5`animal_size_id`：

```
{
   "userId":"12345",
   "itemMetadataList":[
      {
         "itemId":"1",
         "animalMetadata":{
            "animal_species_id":"2",
            "animal_primary_breed_id":"Saint_Bernard",
            "animal_size_id":"3",
            "animal_age_id":"2"
         }
      },
      {
         "itemId":"2",
         "animalMetadata":{
            "animal_species_id":"1",
            "animal_primary_breed_id":"Egyptian_Mau",
            "animal_size_id":"1",
```

```
            "animal_age_id":"1"
         }
      },
      {
         "itemId":"3",
         "animalMetadata":{
            "animal_species_id":"2",
            "animal_primary_breed_id":"Saint_Bernard",
            "animal_size_id":"3",
            "animal_age_id":"2"
         }
      }
   ]
}
```

Lambda 函數會重新排序這些項目，然後傳回排序清單，其中包含項目 IDs 和來自 Amazon Personalize 的直接回應。這是項目所在的動物群組及其分數的排名清單。Amazon Personalize 使用使用者個人化和個人化排名配方，在建議中包含每個項目的分數。這些分數代表 Amazon Personalize 針對使用者接下來將選擇的項目所擁有的相對確定性。分數越高代表確定性越高。

```
{
   "ranking":[
      "1",
      "3",
      "2"
   ],
   "personalizeResponse":{
      "ResponseMetadata":{
         "RequestId":"a2ec0417-9dcd-4986-8341-a3b3d26cd694",
         "HTTPStatusCode":200,
         "HTTPHeaders":{
            "date":"Thu, 16 Jun 2022 22:23:33 GMT",
            "content-type":"application/json",
            "content-length":"243",
            "connection":"keep-alive",
            "x-amzn-requestid":"a2ec0417-9dcd-4986-8341-a3b3d26cd694"
         },
         "RetryAttempts":0
      },
      "personalizedRanking":[
         {
            "itemId":"2-Saint_Bernard-3-2",
            "score":0.8947961
```

```
         },
         {
            "itemId":"1-Siamese-1-1",
            "score":0.105204
         }
      ],
      "recommendationId":"RID-d97c7a87-bd4e-47b5-a89b-ac1d19386aec"
   }
}
```

Amazon Kinesis 承載

要傳送至 Amazon Kinesis 的承載格式如下：

```
{
    "Partitionkey": "randomstring",
    "Data": {
        "userId": "12345",
        "sessionId": "sessionId4545454",
        "eventType": "DetailView",
        "animalMetadata": {
            "animal_species_id": "1",
            "animal_primary_breed_id": "Russian_Blue",
            "animal_size_id": "1",
            "animal_age_id": "2"
        },
        "animal_id": "98765"

    }
}
```

> **Note**
>
> 未驗證的使用者會移除 `userId` 欄位。

使用 SageMaker AI 和 hydra 簡化從本機開發到可擴展實驗的機器學習工作流程

David Sauerwein、Marco Geiger 和 Julian Ferdinand Grueber，Amazon Web Services

摘要

此模式提供統一的方法，以設定和執行從本機測試到 Amazon SageMaker AI 生產的機器學習 (ML) 演算法。ML 演算法是此模式的重點，但其方法延伸到特徵工程、推論和整個 ML 管道。此模式示範透過範例使用案例，從本機指令碼開發轉換至 SageMaker AI 訓練任務。

典型的 ML 工作流程是在本機機器上開發和測試解決方案、在雲端執行大規模實驗（例如，使用不同的參數），以及在雲端部署已核准的解決方案。然後，必須監控和維護部署的解決方案。如果沒有統一的工作流程方法，開發人員通常需要在每個階段重構程式碼。如果解決方案依賴於此工作流程的任何階段可能變更的大量參數，則保持組織和一致性可能會變得越來越困難。

此模式可解決這些挑戰。首先，它提供統一的工作流程，無論在本機機器、容器還是 SageMaker AI 上執行，都不需要在環境之間進行程式碼重構。其次，它透過 Hydra 的組態系統簡化參數管理，其中參數是在不同的組態檔案中定義，可輕鬆修改和合併，並自動記錄每個執行的組態。如需此模式如何處理這些挑戰的詳細資訊，請參閱其他資訊。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 部署和啟動 SageMaker AI 訓練任務的 AWS Identity and Access Management (IAM) 使用者角色
- AWS Command Line Interface (AWS CLI) 2.0 版或更新版本已安裝並設定
- 已安裝 Poetry 1.8 版或更新版本，但早於 2.0 版
- 已安裝 Docker
- Python 3.10.x 版

限制

- 此程式碼目前僅鎖定 SageMaker AI 訓練任務。將其擴展到處理任務和整個 SageMaker AI 管道非常簡單。

- 若要進行完全生產的 SageMaker AI 設定，必須備妥其他詳細資訊。範例可以是用於運算和儲存的自訂 AWS Key Management Service (AWS KMS) 金鑰，或聯網組態。您也可以在 `config` 資料夾的專用子資料夾中使用 hydra 來設定這些額外的選項。
- 有些 AWS 服務 無法全部使用 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

下圖說明解決方案的架構。

該圖顯示以下工作流程：

1. 資料科學家可以在本機環境中以小規模迭代演算法、調整參數，以及快速測試訓練指令碼，而不需要 Docker 或 SageMaker AI。（如需詳細資訊，請參閱《Epics》中的「在本機執行以進行快速測試」任務。）
2. 滿足演算法後，資料科學家會建置 Docker 映像並將其推送至名為 的 Amazon Elastic Container Registry (Amazon ECR) 儲存庫`hydra-sm-artifact`。（如需詳細資訊，請參閱《Epics》中的「在 SageMaker AI 上執行工作流程」。）
3. 資料科學家使用 Python 指令碼啟動 SageMaker AI 訓練任務或超參數最佳化 (HPO) 任務。對於一般訓練任務，調整後的組態會寫入名為 的 Amazon Simple Storage Service (Amazon S3) 儲存貯體`hydra-sample-config`。對於 HPO 任務，會套用位於 `config` 資料夾中的預設組態集。
4. SageMaker AI 訓練任務會提取 Docker 映像、從 Amazon S3 儲存貯體 讀取輸入資料`hydra-sample-data`，以及從 Amazon S3 儲存貯體擷取組態`hydra-sample-config`或使用預設組態。訓練後，任務會將輸出資料儲存至 Amazon S3 儲存貯體 `hydra-sample-data`。

自動化和擴展

- 對於自動訓練、重新訓練或推論，您可以將 AWS CLI 程式碼與 AWS Lambda、 AWS CodePipeline或 Amazon EventBridge 等服務整合。
- 可以透過變更執行個體大小的組態或新增分散式訓練的組態來實現擴展。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。對於此模式， AWS CLI 適用於初始資源組態和測試。
- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- Amazon SageMaker AI 是一種受管機器學習 (ML) 服務，可協助您建置和訓練 ML 模型，然後將模型部署到生產就緒的託管環境中。SageMaker AI Training 是 SageMaker AI 內的全受管 ML 服務，可大規模訓練 ML 模型。該工具可以有效地處理訓練模型的運算需求，利用內建的可擴展性和與其他模型的整合 AWS 服務。SageMaker AI Training 也支援自訂演算法和容器，因此可靈活用於各種 ML 工作流程。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他工具

- Docker 是一組平台即服務 (PaaS) 產品，可在作業系統層級使用虛擬化在容器中交付軟體。它用於此模式，以確保從開發到部署，以及可靠地封裝相依性和程式碼等各種階段的環境一致。允許 Docker 的容器化，以便在整個工作流程中輕鬆擴展和控制版本。
- Hydra 是一種組態管理工具，可為處理多個組態和動態資源管理提供彈性。它有助於管理環境組態，允許在不同環境中無縫部署。如需 Hydra 的詳細資訊，請參閱其他資訊。
- Python 是一種一般用途的電腦程式設計語言。Python 用於撰寫 ML 程式碼和部署工作流程。
- Poetry 是一種在 Python 中管理相依性和封裝的工具。

程式碼儲存庫

此模式的程式碼可在 GitHub configuring-sagemaker-training-jobs-with-hydra 儲存庫中使用。

最佳實務

- 選擇 IAM 角色以部署和啟動遵循最低權限原則的 SageMaker AI 訓練任務，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。
- 使用暫時登入資料來存取終端機中的 IAM 角色。

史詩

設定環境

任務	Description	所需的技能

建立和啟用虛擬環境。	若要建立和啟用虛擬環境，請在儲存庫的根目錄中執行下列命令：
	`poetry install`
	`poetry shell`
		一般 AWS

部署基礎設施。	若要使用 CloudFormation 部署基礎設施，請執行下列命令： `aws cloudformation deploy --template-file infra/hydra-sagemaker-setup.yaml --stack-name hydra-sagemaker-setup --capabilities CAPABILITY_NAMED_IAM`	一般 AWS，DevOps 工程師
下載範例資料。	若要將輸入資料從 openml 下載到本機電腦，請執行下列命令： `python scripts/download_data.py`	一般 AWS
在本機執行 以進行快速測試。	若要在本機執行訓練程式碼進行測試，請執行下列命令： `python mypackage/train.py data.train_data_path=data/t`	資料科學家

任務	Description	所需的技能
	`rain.csv evaluatio n.base_dir_path=data` 所有執行的日誌會依執行時間儲存在名為 的資料夾中outputs。如需詳細資訊，請參閱 GitHub 儲存庫中的「輸出」一節。 您也可以使用 `--multiru n` 功能，以不同的參數平行執行多個訓練。如需詳細資訊，請參閱 Hydra 文件。	

在 SageMaker AI 上執行工作流程

任務	Description	所需的技能

設定環境變數。	若要在 SageMaker AI 上執行任務，請設定下列環境變數，並提供您的 AWS 區域 和 AWS 帳戶 ID：
	
	`export ECR_REPO_ NAME=hydra-sm-arti fact`
	`export image_tag =latest`
	`export AWS_REGIO N="<your_aws_regio n>" # for instance, us- east-1`
	`export ACCOUNT_I D="<your_account_id>"`
	`export BUCKET_NA ME_DATA=hydra-sample- data-$ACCOUNT_ID`
		一般 AWS

任務	Description	所需的技能

	`export BUCKET_NAME_CONFIG=hydra-sample-config-$ACCOUNT_ID`
	`export AWS_DEFAULT_REGION=$AWS_REGION`
	`export ROLE_ARN=arn:aws:iam::${ACCOUNT_ID}:role/hydra-sample-sagemaker`
	`export INPUT_DATA_S3_PATH=s3://$BUCKET_NAME_DATA/hydra-on-sm/input/`
	`export OUTPUT_DATA_S3_PATH=s3://$BUCKET_NAME_DATA/hydra-on-sm/output/`	

建立並推送 Docker 映像。	若要建立 Docker 映像並將其推送至 Amazon ECR 儲存庫，請執行下列命令：
	`chmod +x scripts/create_and_push_image.sh`
	`scripts/create_and_push_image.sh $ECR_REPO_NAME $image_tag $AWS_REGION $ACCOUNT_ID`
	此任務假設您在環境中擁有有效的登入資料。Docker 映像會推送到先前任務中環境變數中指定的 Amazon ECR 儲存庫，並用於啟用訓練任務將在其中執行的 SageMaker AI 容器。
		ML 工程師，一般 AWS

任務	Description	所需的技能
將輸入資料複製到 Amazon S3。	SageMaker AI 訓練任務需要接收輸入資料。若要將輸入資料複製到 Amazon S3 儲存貯體以取得資料，請執行下列命令： `aws s3 cp data/train.csv "${INPUT_DATA_S3_PATH}train.csv"`	資料工程師，一般 AWS
提交 SageMaker AI 訓練任務。	若要簡化指令碼的執行，請在 `default.yaml` 檔案中指定預設組態參數。除了確保跨執行的一致性之外，此方法還提供視需要輕鬆覆寫預設設定的彈性。請參閱下列範例： `python scripts/start_sagemaker_training_job.py sagemaker.role_arn=$ROLE_ARN sagemaker.config_s3_bucket=$BUCKET_NAME_CONFIG sagemaker.input_data_s3_path=$INPUT_DATA_S3_PATH sagemaker.output_data_s3_path=$OUTPUT_DATA_S3_PATH`	一般 AWS、ML 工程師、資料科學家

任務	Description	所需的技能
執行 SageMaker AI 超參數調校。	執行 SageMaker AI 超參數調校類似於提交 SageMaker AII 訓練任務。不過，執行指令碼在一些重要方面有所不同，如您在 start_sagemaker_hpo_job.py 檔案中所見。要調校的超參數必須透過 boto3 承載傳遞，而不是訓練任務的通道。 若要啟動超參數最佳化 (HPO) 任務，請執行下列命令： `python scripts/start_sagemaker_hpo_job.py sagemaker.role_arn=$ROLE_ARN sagemaker.config_s3_bucket=$BUCKET_NAME_CONFIG sagemaker.input_data_s3_path=$INPUT_DATA_S3_PATH sagemaker.output_data_s3_path=$OUTPUT_DATA_S3_PATH`	資料科學家

疑難排解

問題	解決方案
權杖過期	匯出新的 AWS 登入資料。
缺少 IAM 許可	請務必匯出具備所有必要 IAM 許可的 IAM 角色憑證，以部署 CloudFormation 範本並啟動 SageMaker AI 訓練任務。

相關資源

- 使用 Amazon SageMaker AI 訓練模型 (AWS 文件)
- 什麼是超參數調校？

其他資訊

此模式可解決下列挑戰：

從本機開發到大規模部署的一致性 – 透過此模式，開發人員可以使用相同的工作流程，無論他們是使用本機 Python 指令碼、執行本機 Docker 容器、在 SageMaker AI 上執行大型實驗，還是在 SageMaker AI 的生產環境中部署。由於下列原因，此一致性很重要：

- 更快速的反覆運算 – 它允許快速的本機實驗，而不需要在擴展時進行重大調整。
- 不重構 – 在 SageMaker AI 上轉換到更大的實驗是無縫的，不需要對現有設定進行大修。
- 持續改進 – 開發新功能並持續改進演算法非常簡單，因為程式碼在環境中保持不變。

組態管理 – 此模式使用組態管理工具 hydra 來提供下列優點：

- 參數是在組態檔案中定義，與程式碼分開。
- 您可以輕鬆交換或組合不同的參數集。
- 實驗追蹤會簡化，因為每個執行的組態都會自動記錄。
- 雲端實驗可以使用與本機執行相同的組態結構，以確保一致性。

使用 Hydra，您可以有效管理組態，啟用下列功能：

- 分割組態 – 將您的專案組態分成可獨立修改的較小、可管理的部分。這種方法可讓您更輕鬆地處理複雜的專案。
- 輕鬆調整預設值 – 快速變更基準組態，讓您更輕鬆地測試新想法。
- 對齊 CLI 輸入和組態檔案 – 順暢地將命令列輸入與您的組態檔案合併。這種方法可減少雜亂和混淆，讓您的專案隨著時間更易於管理。

將自然語言轉換為查詢 DSL for OpenSearch 和 Elasticsearch 查詢

Tabby Ward、Nicholas Switzer 和 Breanne Warner，Amazon Web Services

總結

此模式示範如何使用大型語言模型 (LLMs) 將自然語言查詢轉換為查詢網域特定的語言 （ 查詢 DSL)，讓使用者更輕鬆地與搜尋服務互動，例如 OpenSearch 和 Elasticsearch，而無需對查詢語言有廣泛的了解。對於想要使用自然語言查詢功能增強以搜尋為基礎的應用程式，最終改善使用者體驗和搜尋功能的開發人員和資料科學家來說，此資源特別有用。

模式說明了提示性工程、反覆精簡和納入專業知識的技術，這些技術對於合成資料產生至關重要。雖然此方法主要著重於查詢轉換，但隱含地展示了資料擴增和可擴展性合成資料生產的可能性。此基礎可以擴展到更全面的合成資料產生任務，以強調 LLMs使用結構化、應用程式特定的輸出橋接非結構化自然語言輸入的能力。

此解決方案在傳統意義上不涉及遷移或部署工具。相反地，它著重於示範使用 LLMs概念驗證 (PoC)。

- 模式使用 Jupyter 筆記本做為step-by-step指南。 text-to-query
- 它使用 Amazon Bedrock 存取 LLMs，這對於解譯自然語言和產生適當的查詢至關重要。
- 解決方案旨在與 Amazon OpenSearch Service 搭配使用。您可以遵循 Elasticsearch 的類似程序，而產生的查詢可能會適應類似的搜尋引擎。

查詢 DSL 是一種靈活的 JSON 型搜尋語言，用於在 Elasticsearch 和 OpenSearch 中建構複雜的查詢。它可讓您在搜尋操作的查詢參數中指定查詢，並支援各種查詢類型。DSL 查詢包含分葉查詢和複合查詢。分葉查詢會搜尋特定欄位中的特定值，並包含全文、術語層級、地理、聯結、跨度和特殊化查詢。複合查詢可做為多個分葉或複合子句的包裝函式，並結合其結果或修改其行為。查詢 DSL 支援建立複雜的搜尋，範圍從簡單的所有相符查詢，到產生高度特定結果的複雜多子句查詢。對於需要進階搜尋功能、彈性查詢建構和 JSON 型查詢結構的專案，查詢 DSL 特別有價值。

此模式使用技術，例如少量擷取提示、系統提示、結構化輸出、提示鏈結、內容佈建，以及text-to-query DSL 轉換的任務特定提示。如需這些技術的定義和範例，請參閱其他資訊一節。

先決條件和限制

先決條件

若要有效地使用 Jupyter 筆記本將自然語言查詢轉換為查詢 DSL 查詢，您需要：

- 熟悉 Jupyter 筆記本。對如何在 Jupyter 筆記本環境中導覽和執行程式碼的基本了解。
- Python 環境。運作中的 Python 環境，最好是 Python 3.x，並已安裝必要的程式庫。
- Elasticsearch 或 OpenSearch 知識。Elasticsearch 或 OpenSearch 的基本知識，包括其架構以及如何執行查詢。
- AWS 帳戶。 AWS 帳戶 用於存取 Amazon Bedrock 和其他相關服務的作用中 。
- 程式庫和相依性。安裝筆記本中提及的特定程式庫，例如`boto3` AWS 互動，以及 LLM 整合所需的任何其他相依性。
- Amazon Bedrock 內的模型存取。此模式使用來自 Anthropic LLMs。開啟 Amazon Bedrock 主控台，然後選擇模型存取。在下一個畫面上，選擇啟用特定模型，然後選取這三個模型：
  - Claude 3 Sonnet
  - Claude 3.5 Sonnet
  - Claude 3 Haiku
- 適當的 IAM 政策和 IAM 角色。若要在 中執行筆記本 AWS 帳戶，您的 AWS Identity and Access Management (IAM) 角色需要 `SagemakerFullAccess` 政策以及其他資訊區段中提供的政策，您可以將其命名為 `APGtext2querydslpolicy`。此政策包含訂閱列出的三個 Claude 模型。

備妥這些先決條件可確保在使用筆記本時獲得順暢的體驗，並實作text-to-query功能。

限制

- 概念狀態證明。此專案主要用於概念驗證 (PoC)。它示範了使用 LLMs 將自然語言查詢轉換為查詢 DSL 的可能性，但它可能尚未完全最佳化或可供生產使用。
- 模型限制：

  內容視窗限制條件。使用 Amazon Bedrock 上可用的 LLMs 時，請注意內容時段限制：

  Claude 模型 （ 截至 2024 年 9 月 ）：
  - Claude 3 Opus：200，000 個字符
  - Claude 3 Sonnet：200，000 個字符
  - Claude 3 Haiku：200，000 個字符

  Amazon Bedrock 上的其他模型可能會有不同的內容視窗大小。請務必檢查最新的文件以取得最新資訊。

  模型可用性。 Amazon Bedrock 上特定模型的可用性可能會有所不同。在實作此解決方案之前，請確定您能夠存取所需的模型。

- 其他限制
  - 查詢複雜性。查詢 DSL 轉換的自然語言有效性可能會因輸入查詢的複雜性而有所不同。
  - 版本相容性。產生的查詢 DSL 可能需要根據您使用的 Elasticsearch 或 OpenSearch 特定版本進行調整。
  - 效能。此模式提供 PoC 實作，因此查詢產生速度和準確性可能不最適合大規模生產使用。
  - 成本：在 Amazon Bedrock 中使用 LLMs 會產生成本。請注意所選模型的定價結構。如需詳細資訊，請參閱 Amazon Bedrock 定價。
  - 維護。為了跟上 LLM 技術的進展和查詢 DSL 語法的變更，可能需要定期更新提示和模型選擇。

產品版本

此解決方案已在 Amazon OpenSearch Service 中測試。如果您想要使用 Elasticsearch，您可能需要進行一些變更，才能複寫此模式的確切功能。

- OpenSearch 版本相容性。 OpenSearch 在主要版本中維持回溯相容性。例如：
  - OpenSearch 1.x 用戶端通常與 OpenSearch 1.x 叢集相容。
  - OpenSearch 2.x 用戶端通常與 OpenSearch 2.x 叢集相容。

  不過，最好盡可能同時針對用戶端和叢集使用相同的次要版本。
- OpenSearch API 相容性。 OpenSearch 針對大多數操作維持與 Elasticsearch OSS 7.10.2 的 API 相容性。不過，存在一些差異，特別是在較新的版本中。
- OpenSearch 升級考量事項：
  - 不支援直接降級。視需要使用快照進行轉返。
  - 升級時，請檢查相容性矩陣和版本備註是否有任何重大變更。

Elasticsearch 考量事項

- Elasticsearch 版本。您使用的 Elasticsearch 主要版本至關重要，因為查詢語法和功能可能會在主要版本之間變更。目前，最新的穩定版本是 Elasticsearch 8.x。請確定您的查詢與您的特定 Elasticsearch 版本相容。
- Elasticsearch 查詢 DSL 程式庫版本。如果您使用的是 Elasticsearch 查詢 DSL Python 程式庫，請確定其版本與您的 Elasticsearch 版本相符。例如：
  - 對於 Elasticsearch 8.x，請使用大於或等於 8.0.0 但小於 9.0.0 的`elasticsearch-dsl`版本。
  - 對於 Elasticsearch 7.x，請使用大於或等於 7.0.0 但小於 8.0.0 的`elasticsearch-dsl`版本。

- 用戶端程式庫版本。無論您是使用官方 Elasticsearch 用戶端還是特定語言用戶端，請確定它與您的 Elasticsearch 版本相容。
- 查詢 DSL 版本。查詢 DSL 會隨著 Elasticsearch 版本而演進。某些查詢類型或參數可能會遭到取代，或在不同的版本中引入。
- 映射版本。定義索引映射和版本之間變更的方式。請務必檢查特定 Elasticsearch 版本的映射文件。
- 分析工具版本。如果您使用的是分析器、權杖化器或其他文字分析工具，其行為或可用性可能會在版本之間變更。

Architecture

目標架構

下圖說明此模式的架構。

其中：

1. 使用者輸入和系統提示，其中包含少量的提示範例。程序從提供自然語言查詢或產生結構描述請求的使用者開始。
2. Amazon Bedrock。輸入會傳送到 Amazon Bedrock，做為存取 Claude LLM 的界面。
3. Claude 3 Sonnet LLM。Amazon Bedrock 使用來自 Claude 3 系列 LLMs Claude 3 Sonnet 來處理輸入。它會解譯並產生適當的 Elasticsearch 或 OpenSearch 查詢 DSL。對於結構描述請求，它會產生合成 Elasticsearch 或 OpenSearch 映射。
4. 查詢 DSL 產生。對於自然語言查詢，應用程式會取得 LLM 的輸出，並將其格式化為有效的 Elasticsearch 或 OpenSearch Service 查詢 DSL。
5. 合成資料產生。應用程式也會採用結構描述來建立合成 Elasticsearch 或 OpenSearch 資料，以載入 OpenSearch Serverless 集合進行測試。
6. OpenSearch 或 Elasticsearch。產生的查詢 DSL 會針對所有索引上的 OpenSearch Serverless 集合進行查詢。JSON 輸出包含來自 OpenSearch Serverless 集合中資料的相關資料和命中次數。

自動化和擴展

此模式提供的程式碼專為 PoC 用途而建置。以下清單提供一些建議，讓您進一步自動化和擴展解決方案，並將程式碼移至生產環境。這些增強功能超出此模式的範圍。

- 容器化：

  - Dockerize 應用程式以確保不同環境的一致性。
  - 使用 Amazon Elastic Container Service (Amazon ECS) 或 Kubernetes 等容器協同運作平台進行可擴展的部署。
- 無伺服器架構：
  - 將核心功能轉換為 AWS Lambda 函數。
  - 使用 Amazon API Gateway 為自然語言查詢輸入建立 RESTful 端點。
- 非同步處理：
  - 實作 Amazon Simple Queue Service (Amazon SQS) 將傳入的查詢排入佇列。
  - 使用 AWS Step Functions 來協調處理查詢和產生查詢 DSL 的工作流程。
- 快取：
  - 實作機制以快取提示。
- 監控和記錄：
  - 使用 Amazon CloudWatch 進行監控和提醒。
  - 使用 Amazon CloudWatch Logs 或 Amazon OpenSearch Service 實作集中式日誌記錄，以進行日誌分析。
- 安全性增強功能：
  - 實作 IAM 角色以進行精細存取控制。
  - 使用 AWS Secrets Manager 安全地存放和管理 API 金鑰和登入資料。
- 多區域部署：
  - 請考慮在多個 之間部署解決方案 AWS 區域 ，以改善延遲和災難復原。
  - 使用 Amazon Route 53 進行智慧型請求路由。

透過實作這些建議，您可以將此 PoC 轉換為強大、可擴展且可供生產使用的解決方案。我們建議您在完全部署之前徹底測試每個元件和整個系統。

工具

工具

- Amazon SageMaker AI 筆記本是全受管的 Jupyter 筆記本，用於機器學習開發。此模式使用筆記本做為互動式環境，在 Amazon SageMaker AI 中進行資料探索、模型開發和實驗。筆記本可與其他 SageMaker AI 功能 和 無縫整合 AWS 服務。
- Python 是一種一般用途的電腦程式設計語言。此模式使用 Python 做為核心語言來實作解決方案。

- Amazon Bedrock 是一項全受管服務，可讓您透過統一 API 使用來自領導 AI 新創公司的高效能基礎模型 (FMs) 和 Amazon。Amazon Bedrock 提供 LLMs的存取權，以進行自然語言處理。此模式使用 Anthropic Claude 3 模型。
- 適用於 Python (Boto3) 的 AWS SDK 是一種軟體開發套件，可協助您整合 Python 應用程式、程式庫或指令碼 AWS 服務，包括 Amazon Bedrock。
- Amazon OpenSearch Service 是一項受管服務，可協助您在 AWS 雲端中部署、操作和擴展 OpenSearch Service 叢集。此模式使用 OpenSearch Service 做為產生查詢 DSL 的目標系統。

程式碼儲存庫

此模式的程式碼可在 GitHub Prompt Engineering Text-to-QueryDSL 中使用 Claude 3 Models 儲存庫。此範例使用運作狀態社交媒體應用程式，為與運作狀態應用程式相關聯的使用者和使用者設定檔建立貼文。

最佳實務

使用此解決方案時，請考慮下列事項：

- 需要適當的 AWS 登入資料和許可才能存取 Amazon Bedrock
- 與使用 AWS 服務 和 LLMs相關的潛在成本
- 了解查詢 DSL 驗證並可能修改所產生查詢的重要性

史詩

設定環境並準備資料

任務	Description	所需的技能
設定開發環境。	**Note** 如需此模式和此模式中其他步驟的詳細說明和程式碼，請參閱 GitHub 儲存庫中的完整演練。	Python、pip、AWS 開發套件

任務	Description	所需的技能
	1. `requests-aws4auth` 使用 pip 安裝必要的 Python 套件，包括 `boto3`、`opensearch-py` 、、 `numpy` `awscli`和 。 2. 從 `json`匯入必要的模組，例如 `boto3`、`os`、`opensearch` `opensearchpy` 、、`RequestsHttpConnection` 從 `Opensearchpy` 、`bulk`從 `opensearchpy.helpers` 、`sagemaker` 、`re`、`time` `random`和 `AWS4Auth` 從 匯入`requests_aws4auth` 。	
設定 AWS 存取權。	設定 Amazon Bedrock 用戶端和 SageMaker AI 工作階段。擷取 SageMaker AI 執行角色的 Amazon Resource Name (ARN)，以供日後建立 OpenSearch Serverless 集合時使用。	IAM、AWS CLI、Amazon Bedrock、Amazon SageMaker
載入運作狀態應用程式結構描述。	從預先定義的檔案讀取和剖析運作狀態文章和使用者設定檔的 JSON 結構描述。將結構描述轉換為字串以供稍後在提示中使用。	DevOps 工程師、一般 AWS、Python、JSON

產生合成資料

任務	Description	所需的技能

建立 LLM 型資料產生器。	實作 generate_data() 函數，以 Claude 3 模型呼叫 Amazon Bedrock Converse API。設定 Sonnet、Sonnet 3.5 和 Haiku IDs：
	`model_id_sonnet3_5 = "anthropic.claude-3-5-sonnet-20240620-v1:0"`
	`model_id_sonnet = "anthropic.claude-3-sonnet-20240229-v1:0"`
	`model_id_haiku = "anthropic.claude-3-haiku-20240307-v1:0"`
		Python、Amazon Bedrock API、LLM 提示

建立合成運作狀態文章。	使用 generate_data() 函數搭配特定訊息提示，根據提供的結構描述建立合成運作狀態文章項目。函數呼叫如下所示： `health_post_data = generate_data(bedrock_rt, model_id_sonnet, system_prompt, message_healthpost, inference_config)`	Python、JSON
建立合成使用者設定檔。	使用 generate_data() 函數搭配特定訊息提示，根據提供的結構描述建立合成使用者設定檔項目。這類似於產生運作狀態文章，但使用不同的提示。	Python、JSON

設定 OpenSearch 和擷取資料

任務	Description	所需的技能
設定 OpenSearch Serverless 集合。	使用 Boto3 建立具有適當加密、網路和存取政策的 OpenSearch Serverless 集合。集合建立如下所示： `collection = aoss_client.create_collection(name=es_name, type='SEARCH')` 如需 OpenSearch Serverless 的詳細資訊，請參閱 AWS 文件。	OpenSearch Serverless、IAM
定義 OpenSearch 索引。	根據預先定義的結構描述映射，使用 OpenSearch 用戶端建立運作狀態文章和使用者設定檔的 indiexe。索引建立如下所示： `response_health = oss_client.indices.create(healthpost_index, body=healthpost_body)`	OpenSearch、JSON
將資料載入 OpenSearch。	執行 ingest_data() 函數，將合成運作狀態文章和使用者設定檔大量插入各自的 OpenSearch 索引。函數使用來自 的大量協助程式opensearch-py ：	Python、OpenSearch API、大量資料操作

任務	Description	所需的技能
	`success, failed = bulk(oss_client, actions)`	

產生和執行查詢

任務	Description	所需的技能
設計少量擷取提示範例。	使用 Claude 3 模型產生範例查詢和對應的自然語言問題，做為產生查詢的少量範例。系統提示包含下列範例： `system_prompt_query_generation = [{"text": f"""You are an expert query dsl generator. ... Examples: {example_prompt} ..."""}]`	LLM 提示、查詢 DSL
建立text-to-query DSL 轉換器。	實作系統提示，其中包含結構描述、資料和少量擷取範例，以產生查詢。使用系統提示將自然語言查詢轉換為查詢 DSL。函數呼叫如下所示： `query_response = generate_data(bedrock_rt, model_id, system_prompt_query_generation, query, inference_config)`	Python、Amazon Bedrock API、LLM 提示
在 OpenSearch 上測試查詢 DSL。	執行 query_oss() 函數，針對 OpenSearch Serverless 集合	Python、OpenSearch API、查詢 DSL

任務	Description	所需的技能
	執行產生的查詢 DSL，並傳回結果。函數使用 OpenSearch 用戶端的搜尋方法： `response = oss_clien t.search(index="_a ll", body=temp)`	

測試和評估

任務	Description	所需的技能
建立測試查詢集。	使用 Claude 3 根據合成資料和結構描述產生一組不同的測試問題： `test_queries = generate_data(bedr ock_rt, model_id_ sonnet, query_sys tem_prompt, query_pro mpt, inference_config)`	LLM 提示
評估查詢 DSL 轉換的準確性。	對 OpenSearch 執行查詢，並分析傳回的結果相關性和準確性，以測試產生的查詢 DSL。這包括執行查詢和檢查命中： `output = query_oss (response1) print("Re sponse after running query against Opensearc h") print(output)`	Python、資料分析、查詢 DSL
基準 Claude 3 模型。	比較不同 Claude 3 模型 (Haiku、Sonnet、Sonnet 3.5)	Python、效能基準測試

任務	Description	所需的技能
	在準確性和延遲方面產生查詢的效能。若要比較，請在呼叫 generate_data() `model_id` 時變更 並測量執行時間。	

清除和記錄

任務	Description	所需的技能
開發清除程序。	使用後從 OpenSearch Serverless 集合中刪除所有索引。	Python、AWS 開發套件、OpenSearch API

相關資源

- 查詢 DSL (OpenSearch 文件)
- Amazon OpenSearch Service 文件
- OpenSearch Serverless 集合
- Amazon Bedrock 文件
- Amazon SageMaker AI 文件
- 適用於 Python (Boto3) 的 AWS SDK 文件

其他資訊

IAM 政策

以下是此模式中所用 IAM 角色`APGtext2querydslpolicy`的政策：

```
{
  "Version": "2012-10-17",
  "Statement": [
    { "Effect": "Allow",
      "Action": [
        "bedrock:InvokeModel",
```

```
        "bedrock:InvokeModelWithResponseStream"
      ],
      "Resource": "*"
    },
    { "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::sagemaker-*",
        "arn:aws:s3:::sagemaker-*/*"
      ]
    },
    { "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "arn:aws:logs:*:*:log-group:/aws/sagemaker/*"
    },
    { "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    },
    { "Effect": "Allow",
      "Action": [
        "aoss:*"
      ],
      "Resource": "*"
    },
    { "Effect": "Allow",
      "Action": [
        "iam:PassRole",
        "sagemaker:*"
      ],
      "Resource": [
        "arn:aws:iam::*:role/*", "*"
```

```
      ],
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": "sagemaker.amazonaws.com"
          }
      }
    },
    { "Effect": "Allow",
      "Action": [
        "codecommit:GetBranch",
        "codecommit:GetCommit",
        "codecommit:GetRepository",
        "codecommit:ListBranches",
        "codecommit:ListRepositories"
      ],
      "Resource": "*"
    },
    { "Effect": "Allow",
      "Action": [
        "aws-marketplace:Subscribe"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws-marketplace:ProductId": [
            "prod-6dw3qvchef7zy",
            "prod-m5ilt4siql27k",
            "prod-ozonys2hmmpeu"
          ]
        }
      }
    },
    { "Effect": "Allow",
      "Action": [
        "aws-marketplace:Unsubscribe",
        "aws-marketplace:ViewSubscriptions"
      ],
      "Resource": "*"
    },
    { "Effect": "Allow",
      "Action": "iam:*",
      "Resource": "*"
    }
  ]
```

```
}
```

Anthropic Claude 3 模型的提示技術

此模式示範使用 Claude 3 模型進行text-to-query DSL 轉換的下列提示技術。

- Few-shot 提示：Few-shot 提示是改善 Amazon Bedrock 上 Claude 3 模型效能的強大技術。此方法涉及為模型提供少量範例，示範所需的輸入/輸出行為，然後再要求它執行類似的任務。當您在 Amazon Bedrock 上使用 Claude 3 模型時，對於需要特定格式、推理模式或網域知識的任務，少量擷取提示特別有效。若要實作此技術，您通常會使用兩個主要元件來建構提示：範例區段和實際查詢。範例區段包含一或多個說明任務的輸入/輸出對，而查詢區段會顯示您想要回應的新輸入。此方法有助於 Claude 3 了解內容和預期的輸出格式，通常會導致更準確且一致的回應。

  範例：

```
"query": {
  "bool": {
    "must": [
      {"match": {"post_type": "recipe"}},
      {"range": {"likes_count": {"gte": 100}}},
      {"exists": {"field": "media_urls"}}
    ]
  }
}
Question: Find all recipe posts that have at least 100 likes and include media URLs.
```

- 系統提示：除了少量擷取提示之外，Amazon Bedrock 上的 Claude 3 模型也支援使用系統提示。系統提示是在向模型提供特定使用者輸入之前，提供模型整體內容、指示或指導方針的方法。它們特別適用於設定音調、定義模型的角色，或建立整個對話的限制。若要在 Amazon Bedrock 上使用 Claude 3 的系統提示，請在 API 請求的 `system` 參數中包含它。這與使用者訊息分開，並套用於整個互動。詳細的系統提示用於設定內容並提供模型的指導方針。

  範例：

```
You are an expert query dsl generator. Your task is to take an input question and
 generate a query dsl to answer the question. Use the schemas and data below to
 generate the query.

Schemas: [schema details]
Data: [sample data]
```

```
Guidelines:
- Ensure the generated query adheres to DSL query syntax
- Do not create new mappings or other items that aren't included in the provided
 schemas.
```

- 結構化輸出：您可以指示模型以特定格式提供輸出，例如 JSON 或在 XML 標籤內。

  範例：

  ```
  Put the query in json tags
  ```

- 提示鏈結：筆記本使用一個 LLM 呼叫的輸出做為另一個 LLM 呼叫的輸入，例如使用產生的合成資料來建立範例問題。
- 內容佈建：相關內容，包括結構描述和範例資料，會在提示中提供。

  範例：

  ```
  Schemas: [schema details]
  Data: [sample data]
  ```

- 任務特定的提示：針對特定任務製作不同的提示，例如產生合成資料、建立範例問題，以及將自然語言查詢轉換為查詢 DSL。

  產生測試問題的範例：

  ```
  Your task is to generate 5 example questions users can ask the health app based on
   provided schemas and data. Only include the questions generated in the response.
  ```

使用 Amazon Q Developer 作為編碼助理，以提高您的生產力

Ram Kandaswamy，Amazon Web Services

總結

此模式使用 tic-tac-toe 遊戲，示範如何在各種開發任務中套用 Amazon Q Developer。它會產生 tic-tac-toe 遊戲的程式碼做為單頁應用程式 (SPA)，增強其 UI，並建立指令碼來部署應用程式 AWS。

Amazon Q Developer 做為編碼助理，協助加速軟體開發工作流程，並增強開發人員和非開發人員的生產力。無論您的技術專業知識為何，都可協助您為業務問題建立架構和設計解決方案、引導您的工作環境、協助您實作新功能，以及產生測試案例以進行驗證。它使用自然語言指示和 AI 功能，以確保一致的高品質程式碼，並緩解編碼挑戰，無論您的程式設計技能為何。

Amazon Q Developer 的主要優點是能夠讓您擺脫重複的編碼任務。當您使用`@workspace`註釋時，Amazon Q Developer 會擷取整合開發環境 (IDE) 中的所有程式碼檔案、組態和專案結構，並提供量身打造的回應，協助您專注於創意問題解決。您可以花更多時間設計創新解決方案並增強使用者體驗。如果您不是技術，則可以使用 Amazon Q Developer 來簡化工作流程，並與開發團隊更有效地協作。Amazon Q Developer Explain 程式碼功能提供詳細說明和摘要，因此您可以導覽複雜的程式碼庫。

此外，Amazon Q Developer 提供與語言無關的方法，可協助初階和中階開發人員擴展技能集。您可以專注於核心概念和商業邏輯，而不是語言特定的語法。當您切換技術時，這會減少學習曲線。

先決條件和限制

先決條件

- 安裝 Amazon Q Developer 外掛程式的 IDE （例如 WebStorm 或 Visual Studio Code)。如需說明，請參閱《Amazon Q Developer 文件》中的在 IDE 中安裝 Amazon Q Developer 延伸模組或外掛程式。
- 使用 Amazon Q Developer 的作用中 AWS 帳戶 設定。如需說明，請參閱 Amazon Q Developer 文件中的入門。
- npm 已安裝。如需說明，請參閱 npm 文件。此模式已使用 npm 10.8 版進行測試。
- AWS Command Line Interface (AWS CLI) 已安裝。如需說明，請參閱 AWS CLI 文件。

限制

- Amazon Q Developer 一次只能執行一個開發任務。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

工具

- 此模式需要 IDE，例如 Visual Studio Code 或 WebStorm。如需支援的 IDEs 清單，請參閱 Amazon Q Developer 文件。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。

最佳實務

請參閱 AWS 方案指引中的使用 Amazon Q Developer 進行最佳編碼實務。此外：

- 當您向 Amazon Q Developer 提供提示時，請確定您的指示清楚明確。新增程式碼片段和註釋，例如 `@workspace`到提示中，為您的提示提供更多內容。
- 包含相關的程式庫並進行匯入，以避免系統發生衝突或不正確的猜測。
- 如果產生的程式碼不正確或如預期，請使用提供意見回饋和重新產生選項。嘗試將提示分成較小的指示。

史詩

設定工作環境

任務	Description	所需的技能
建立新專案。	若要在工作環境中建立新的專案，請執行下列命令，並接受所有問題的預設設定： `npx create-next-app@latest`	應用程式開發人員、程式設計師、軟體開發人員
測試基礎應用程式。	執行下列命令，並確認基礎應用程式在瀏覽器中成功載入：	應用程式開發人員、程式設計師、軟體開發人員

任務	Description	所需的技能
	`npm run dev`	
清除基本程式碼。	導覽至 `src/app` 資料夾中`page.tsx`的檔案，並刪除預設內容以取得空白頁面。刪除後，檔案應如下所示： `export default function Home() { return (<div></div> ); }`	應用程式開發人員、程式設計師、軟體開發人員

使用 Amazon Q Developer 設計 tic-tac-toe 遊戲專案

任務	Description	所需的技能
取得步驟概觀。	1. 在您的 IDE 中，開啟您的專案，然後選擇 Amazon Q 圖示以開啟聊天面板。 2. 在 Amazon Q Developer 聊天面板中，詢問建立單頁應用程式 (SPA) 的概觀。例如： `I would like to create a single-page application involving Next.js React framework for tic-tac-toe game. What are the steps?`	應用程式開發人員、程式設計師、軟體開發人員

任務	Description	所需的技能

產生 tic-tac-toe 的程式碼。	在聊天面板中，使用 /dev命令啟動開發任務，後面接著任務的描述。例如：
	
	`/dev Create a React-based single-page application written in TypeScript for a tic-tac-toe game with the following specifications:`
	`1. Design an aesthetically pleasing interface with the game grid centered vertically and horizontally on the page.`
	`2. Include a heading and clear instructions on how to play the game.`
	`3. Implement color-coding for X and O marks to distinguish them easily.`
	
	Amazon Q Developer 會根據您的指示產生程式碼。
		應用程式開發人員、程式設計師、軟體開發人員

檢查並接受產生的程式碼。	目測檢查程式碼，然後選擇接受程式碼以自動取代`page.tsx`檔案。 如果您遇到問題，請選擇提供意見回饋並重新產生並描述您遇到的問題。	應用程式開發人員、程式設計師、軟體開發人員

任務	Description	所需的技能
修正 lint 錯誤。	範例 tic-tac-toe 遊戲包含網格。Amazon Q Developer 產生的程式碼可能會使用預設類型 any。您可以提示 Amazon Q Developer 來新增類型安全，如下所示： `/dev Ensure proper TypeScript typing for the onSquare Click event handler to resolve any 'any' type issues.`	應用程式開發人員、程式設計師、軟體開發人員

任務	Description	所需的技能

新增視覺效果吸引力。	您可以將原始需求分成較小的片段。例如，您可以在開發任務中使用下列提示來改善遊戲 UI。此提示可增強串聯樣式表 (CSS) 樣式，並匯出應用程式以進行部署。
	
	`/dev Debug and fix any CSS issues to correctly display the game grid and overall layout.`
	
	`Simplify the code by removing game history functionality and related components.`
	
	`Implement static file export to an 'out' directory for easy deployment. The solution should be fully functional, visually appealing, and free of typing errors or layout issues.`
		應用程式開發人員、程式設計師、軟體開發人員

任務	Description	所需的技能

再次測試。	1. 現在您已完成開發生命週期，請測試程式碼以確認其如預期般運作。若要在本機執行應用程式，請使用 命令：
	`npm run dev`
	2. 如果應用程式如預期運作，請使用 build命令將整個應用程式匯出至輸出資料夾，以準備部署：
	`npm run build`
		應用程式開發人員、程式設計師、軟體開發人員

將應用程式部署到 AWS 雲端

任務	Description	所需的技能

建立要部署的資料夾和檔案。	在工作環境中的專案中，建立部署資料夾和其中的兩個檔案： `pushtos3.sh` 和 `cloudformation.yml` ：
	`mkdir deployment && cd deployment`
	`touch pushtos3.sh && chmod +x pushtos3.sh`
	`touch cloudformation.yml`
		應用程式開發人員、程式設計師、軟體開發人員

產生自動化程式碼。	1. 在 Amazon Q Developer 的聊天面板中，提供下列提示：	AWS 管理員、AWS DevOps、應用程式開發人員

任務	Description	所需的技能
	``` /dev Generate a Cloudformation template that creates two resources: tictactoe artifact bucket and CloudFron t. CloudFront should be configure d with this bucket as origin. Add cache policy appropriate for Amazon S3 and default root object as index.html. Ensure that origin access control is used and no public bucket is created. Output all the resources and their ARNs. ``` 2. 檢閱並接受產生的程式碼。您先前建立`cloudform ation.yml` 的檔案現在應該會填入為 建立資源的 CloudFormation 指令碼 AWS 雲端。	

任務	Description	所需的技能
產生指令碼內容。	若要建立部署指令碼，請使用下列提示： `/dev Modify the pushtos3 shell script so that it can use AWS CLI commands to create a CloudFormation stack named tictactoe-stack if it does not exist already, and use cloudformation.yml as the source template. Wait for the stack to complete and sync the contents from the out folder to the S3 bucket. Perform invalidation of the CloudFront origin.`	應用程式開發人員、程式設計師、軟體開發人員
將應用程式部署到 AWS 雲端。	1. 使用有效的 AWS 登入資料設定工作環境。 2. 執行 shell 指令碼，將功能完整的 tic-tac-toe 遊戲部署到 AWS 雲端。	AWS 管理員、AWS DevOps、雲端架構師、應用程式開發人員

疑難排解

問題	解決方案
組建不會建立單頁應用程式或將其匯出至輸出資料夾。	查看 `next.config.mjs` 檔案的內容。 如果程式碼具有下列預設組態：

問題	解決方案

	`const nextConfig = {};`
	
	修改如下：
	
	`const nextConfig = {`
	`  output: 'export',`
	`  distDir: 'out',`
	`};`

相關資源

- 建立新的 React 專案 (React 文件）
- Amazon Q Developer 概觀 (AWS 文件）
- Amazon Q Developer 最佳實務 (AWS 方案指引）
- 搭配 JetBrains IDEs 安裝、設定和使用 Amazon Q Developer (YouTube 影片）
- 為命令列安裝 Amazon Q (AWS 文件）

使用 SageMaker Processing 對 TB 級 ML 資料集進行分散式特徵工程

Chris Boomhower，Amazon Web Services

總結

許多 TB 級或更大的資料集通常由階層式資料夾結構組成，而且資料集中的檔案有時會共用相互依存性。因此，機器學習 (ML) 工程師和資料科學家必須做出深思熟慮的設計決策，以準備此類資料進行模型訓練和推論。此模式示範如何結合 Amazon SageMaker Processing 和虛擬 CPU (vCPU) 平行處理，使用手動巨集分片和微分技術，以有效率地擴展複雜大數據 ML 資料集的特徵工程程序。

此模式將巨集碎片定義為跨多部機器處理的資料目錄分割，並將微碎片定義為跨多個處理執行緒分割每部機器上的資料。模式透過使用 Amazon SageMaker 搭配 PhysioNet MIMIC-III 資料集中的範例時間序列波形記錄來示範這些技術。透過在此模式中實作技術，您可以將特徵工程的處理時間和成本降至最低，同時最大化資源使用率和輸送量效率。這些最佳化仰賴 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和 vCPUs 上的分散式 SageMaker Processing 進行類似的大型資料集，無論資料類型為何。

先決條件和限制

先決條件

- 如果您想要為自己的資料集實作此模式，請存取 SageMaker 筆記本執行個體或 SageMaker Studio。如果您是第一次使用 Amazon SageMaker，請參閱 AWS 文件中的開始使用 Amazon SageMaker。
- SageMaker Studio，如果您想要使用 PhysioNet MIMIC-III 範例資料實作此模式。
- 模式使用 SageMaker Processing，但不需要任何執行 SageMaker Processing 任務的經驗。

限制

- 此模式非常適合包含相互依存檔案的 ML 資料集。這些相互依存性受益於手動巨集分割和平行執行多個單一執行個體 SageMaker Processing 任務。對於不存在此類相互依存性的資料集，SageMaker Processing 中的`ShardedByS3Key`功能可能是巨集碎片的更佳替代方案，因為它會將碎片資料傳送至由相同處理任務管理的多個執行個體。不過，您可以在這兩種情況下實作此模式的微分策略，以充分利用執行個體 vCPUs。

產品版本

- Amazon SageMaker Python SDK 第 2 版

Architecture

目標技術堆疊

- Amazon Simple Storage Service (Amazon S3)
- Amazon SageMaker

目標架構

巨集控制和分散式 EC2 執行個體

此架構中表示的 10 個平行程序會反映 MIMIC-III 資料集的結構。（程序會以省略符號表示，以簡化圖表。）當您使用手動巨集分片時，類似的架構會套用至任何資料集。在 MIMIC-III 的情況下，您可以盡可能分別處理每個病患群組資料夾，藉此善用資料集的原始結構。在下圖中，記錄群組區塊會出現在左側 (1)。鑑於資料的分散式本質，依病患群組碎片是合理的。

不過，依病患群組手動分片表示每個病患群組資料夾都需要單獨的處理任務，如圖表 (2) 中間區段所示，而不是具有多個 EC2 執行個體的單一處理任務。由於 MIMIC-III 的資料同時包含二進位波形檔案和相符的文字型標頭檔案，而且需要依賴 wfdb 程式庫才能擷取二進位資料，因此必須在同一執行個體上提供特定病患的所有記錄。唯一可以確定每個二進位波形檔案的關聯標頭檔案也存在的方法，是實作手動碎片，在其自己的處理任務中執行每個碎片，並指定`s3_data_distribution_type='FullyReplicated'`何時定義處理任務輸入。或者，如果所有資料在單一目錄中可用，而且檔案之間不存在相依性，則更合適的選項可能是啟動具有多個 EC2 執行個體且`s3_data_distribution_type='ShardedByS3Key'`指定的單一處理任務。將 指定`ShardedByS3Key` 為 Amazon S3 資料分佈類型，會指示 SageMaker 自動管理跨執行個體的資料分片。

為每個資料夾啟動處理任務是一種經濟實惠的方式來預先處理資料，因為同時執行多個執行個體可節省時間。為了節省額外的成本和時間，您可以在每個處理任務中使用微分。

微分和平行 vCPUs

在每個處理任務中，分組的資料會進一步分割，以最大限度地使用 SageMaker 全受管 EC2 執行個體上所有可用的 vCPUs。圖表 (2) 中間區段中的區塊描述了每個主要處理任務中發生的情況。會根據執

行個體上可用的 vCPUs 數量，將病患記錄資料夾的內容扁平化並平均分割。分割資料夾內容後，平均大小的檔案集會分散到所有 vCPUs以進行處理。處理完成時，每個 vCPU 的結果會合併為每個處理任務的單一資料檔案。

在連接的程式碼中，這些概念會呈現在 `src/feature-engineering-pass1/preprocessing.py` 檔案的下一節中。

```
def chunks(lst, n):
    """
    Yield successive n-sized chunks from lst.

    :param lst: list of elements to be divided
    :param n: number of elements per chunk
    :type lst: list
    :type n: int
    :return: generator comprising evenly sized chunks
    :rtype: class 'generator'
    """
    for i in range(0, len(lst), n):
        yield lst[i:i + n]


# Generate list of data files on machine
data_dir = input_dir
d_subs = next(os.walk(os.path.join(data_dir, '.')))[1]
file_list = []
for ds in d_subs:
    file_list.extend(os.listdir(os.path.join(data_dir, ds, '.')))
dat_list = [os.path.join(re.split('_|\.', f)[0].replace('n', ''), f[:-4]) for f in
 file_list if f[-4:] == '.dat']

# Split list of files into sub-lists
cpu_count = multiprocessing.cpu_count()
splits = int(len(dat_list) / cpu_count)
if splits == 0: splits = 1
dat_chunks = list(chunks(dat_list, splits))

# Parallelize processing of sub-lists across CPUs
ws_df_list = Parallel(n_jobs=-1, verbose=0)(delayed(run_process)(dc) for dc in
 dat_chunks)

# Compile and pickle patient group dataframe
ws_df_group = pd.concat(ws_df_list)
```

```
ws_df_group = ws_df_group.reset_index().rename(columns={'index': 'signal'})
ws_df_group.to_json(os.path.join(output_dir, group_data_out))
```

函數 chunks會先定義為透過將指定清單分割為平均大小的長度區塊n ，並將這些結果傳回為產生器，來使用指定的清單。接下來，透過編譯存在的所有二進位波形檔案清單，將資料扁平化到病患資料夾。完成後，會取得 EC2 執行個體上可用的 vCPUs 數量。透過呼叫 將二進位波形檔案的清單平均分割到這些 vCPUschunks，然後使用 joblib 的平行類別在自己的 vCPU 上處理每個波形子清單。處理任務會自動將結果合併為單一資料影格清單，然後 SageMaker 會在任務完成時進一步處理，再寫入 Amazon S3。在此範例中，處理任務寫入 Amazon S3 的檔案有 10 個 （每個任務一個）。

當所有初始處理任務完成時，次要處理任務會顯示在圖表 (3) 右側的 區塊中，結合每個主要處理任務產生的輸出檔案，並將合併的輸出寫入 Amazon S3 (4)。

工具

工具

- Python – 用於此模式的範例程式碼為 Python （第 3 版）。
- SageMaker Studio – Amazon SageMaker Studio 是適用於機器學習的 Web 型整合式開發環境 (IDE)，可讓您建置、訓練、偵錯、部署和監控機器學習模型。您可以在 SageMaker Studio 中使用 Jupyter 筆記本來執行 SageMaker Processing 任務。
- SageMaker Processing – Amazon SageMaker Processing 提供執行資料處理工作負載的簡化方法。在此模式中，使用 SageMaker Processing 任務大規模實作特徵工程程式碼。

Code

連接的 .zip 檔案提供此模式的完整程式碼。下一節說明為此模式建置架構的步驟。每個步驟都由附件中的範例程式碼說明。

史詩

設定 SageMaker Studio 環境

任務	Description	所需的技能
存取 Amazon SageMaker Studio。	遵循 Amazon SageMaker 文件中提供的指示，在您的	資料科學家、ML 工程師

任務	Description	所需的技能
	AWS 帳戶中加入 SageMaker Studio。Amazon SageMaker	
安裝 wget 公用程式。	如果您使用新的 SageMaker Studio 組態加入，或從未在 SageMaker Studio 中使用這些公用程式，請安裝 wget。 若要安裝，請在 SageMaker Studio 主控台中開啟終端機視窗，然後執行下列命令： `sudo yum install wget`	資料科學家、ML 工程師

下載並解壓縮範例程式碼。	在附件區段中下載 `attachments.zip` 檔案。在終端機視窗中，導覽至您下載檔案的資料夾，並解壓縮其內容：
	
	`unzip attachment.zip`
	
	導覽至您解壓縮 .zip 檔案的資料夾，並解壓縮`Scaled-Processing.zip` 檔案的內容。
	
	`unzip Scaled-Processing.zip`
		資料科學家、ML 工程師

任務	Description	所需的技能
從 physionet.org 下載範例資料集，並將其上傳至 Amazon S3。	在包含`Scaled-Processing` 檔案的資料夾中執行 `get_data.ipynb` Jupyter 筆記本。此筆記本會從 physionet.org 下載範例 MIMIC-III 資料集，並將其上傳至 Amazon S3 中的 SageMaker Studio 工作階段儲存貯體。	資料科學家、ML 工程師

設定第一個預先處理指令碼

任務	Description	所需的技能

扁平化所有子目錄的檔案階層。	在 MIMIC-III 等大型資料集中，檔案通常分佈於多個子目錄，即使在邏輯父群組內也是如此。您的指令碼應設定為在所有子目錄中扁平化所有群組檔案，如下列程式碼所示。
	`# Generate list of .dat files on machine`
	`data_dir = input_dir`
	`d_subs = next(os.walk(os.path.join(data_dir, '.')))[1]`
	`file_list = []`
	`for ds in d_subs:`
	`  file_list.extend(os.listdir(os.path.join(data_dir, ds, '.')))`
	`dat_list = [os.path.join(re.split('_|\.', f)[0].replace('n', ''), f[:-4]) for f in`
		資料科學家、ML 工程師

任務	Description	所需的技能
	`file_list if f[-4:] == '.dat']` **Note** 此範例程式碼片段的範例來自 檔案，該`src/feature-engineering-pass1/preprocessing.py` 檔案在附件中提供。	

根據 vCPU 計數將檔案分成子群組。	根據執行指令碼的執行個體上存在vCPUs 數量，檔案應分為大小均勻的子組或區塊。在此步驟中，您可以實作類似如下的程式碼。
	
	`# Split list of files into sub-lists`
	`cpu_count = multiprocessing.cpu_count()`
	`splits = int(len(dat_list) / cpu_count)`
	`if splits == 0: splits = 1`
	`dat_chunks = list(chunks(dat_list, splits))`
		資料科學家、ML 工程師

任務	Description	所需的技能

平行處理跨 vCPUs的子群組。	指令碼邏輯應設定為平行處理所有子群組。若要這樣做，請使用 Joblib 程式庫的`Parallel` 類別和`delayed` 方法，如下所示。
	`# Parallelize processing of sub-lists across CPUs`
	`ws_df_list = Parallel(n_jobs=-1, verbose=0)(delayed(run_process)(dc) for dc in dat_chunks)`
		資料科學家、ML 工程師

將單一檔案群組輸出儲存至 Amazon S3。	當平行 vCPU 處理完成時，每個 vCPU 的結果應合併並上傳至檔案群組的 S3 儲存貯體路徑。在此步驟中，您可以使用類似如下的程式碼。
	`# Compile and pickle patient group dataframe`
	`ws_df_group = pd.concat(ws_df_list)`
	`ws_df_group = ws_df_group.reset_index().rename(columns={'index': 'signal'})`
	`ws_df_group.to_json(os.path.join(output_dir, group_data_out))`
		資料科學家、ML 工程師

設定第二個預先處理指令碼

任務	Description	所需的技能

合併執行第一個指令碼的所有處理任務所產生的資料檔案。	先前的指令碼會為每個 SageMaker Processing 任務輸出單一檔案，該任務會從資料集處理一組檔案。 接著，您需要將這些輸出檔案合併為單一物件，並將單一輸出資料集寫入 Amazon S3。這是在 檔案中示範的，該`src/feature-engineering-pass1p5/preprocessing.py` 檔案在附件中提供，如下所示。
	
	`def write_parquet(wavs_df, path):`
	`    """`
	`    Write waveform summary dataframe to S3 in parquet format.`
	
	`    :param wavs_df: waveform summary dataframe`
	`    :param path: S3 directory prefix`
	`    :type wavs_df: pandas dataframe`
	`    :type path: str`
	`    :return: None`
	`    """`
	`    extra_args = {"ServerSideEncryption": "aws:kms"}`
	`    wr.s3.to_parquet(`
	`        df=wavs_df,`
	`        path=path,`
		資料科學家、ML 工程師

任務	Description	所需的技能
	compressi on='snappy', s3_additi onal_kwargs=extra_ args) def combine_data(): """ Get combined data and write to parquet. :return: waveform summary dataframe :rtype: pandas dataframe """ wavs_df = get_data( ) wavs_df = normalize _signal_names(wavs _df) write_parquet(wavs _df, "s3://{}/{}/ {}".format(buck et_xform, dataset_p refix, pass1p5ou t_data)) return wavs_df wavs_df = combine_d ata()	

執行處理任務

任務	Description	所需的技能

執行第一個處理任務。	若要執行巨集分割，請為每個檔案群組執行個別的處理任務。Microsharding 會在每個處理任務內執行，因為每個任務都會執行您的第一個指令碼。下列程式碼示範如何在下列程式碼片段（包含在中notebooks/FeatExtract_Pass1.ipynb ）中為每個檔案群組目錄啟動處理任務。
	
	`pat_groups = list(range(30,40))`
	`ts = str(int(time.time()))`
	
	`for group in pat_groups:`
	`  sklearn_processor = SKLearnProcessor(framework_version='0.20.0',`
	
	`  role=role,`
	
	`  instance_type='ml.m5.4xlarge',`
	
	`  instance_count=1,`
	
	`  volume_size_in_gb=5)`
	`  sklearn_processor.run(`
		資料科學家、ML 工程師

任務	Description	所需的技能
	code='../src/feature-engineering-pass1/preprocessing.py', job_name='-'.join(['scaled-processing-p1', str(group), ts]), arguments=["input_path", "/opt/ml/processing/input", "output_path", "/opt/ml/processing/output", "group_data_out", "ws_df_group.json"], inputs= [ProcessingInput(source=f's3://{sess.default_bucket()}/data_inputs/{group}', destination='/opt/ml/processing/input', s3_data_distribution_type='FullyReplicated')], outputs= [ProcessingOutput(	

任務	Description	所需的技能
	source='/opt/ml/processing/output', destination=f's3://{sess.default_bucket()}/data_outputs/{group}')], wait=False)	

任務	Description	所需的技能

執行第二個處理任務。	若要合併第一組處理任務產生的輸出並執行任何額外的預先處理運算，您可以使用單一 SageMaker Processing 任務來執行第二個指令碼。下列程式碼示範這一點（包含在中`notebooks/FeatExtract_Pass1p5.ipynb`）。
	
	`ts = str(int(time.time()))`
	`bucket = sess.default_bucket()`
	
	`sklearn_processor = SKLearnProcessor(framework_version='0.20.0',`
	
	`role=role,`
	
	`instance_type='ml.t3.2xlarge',`
	
	`instance_count=1,`
	
	`volume_size_in_gb=5)`
	`sklearn_processor.run(`
	`code='../src/feature-engineering-pass1p5/preprocessing.py',`
	`job_name='-'.join(['scaled-processing', 'p1p5', ts]),`
	`arguments=['bucket', bucket,`
		資料科學家、ML 工程師

任務	Description	所需的技能
	`'pass1out_prefix', 'data_outputs', 'pass1out_data', 'ws_df_group.json', 'pass1p5out_data', 'waveform_summary.parquet', 'statsdata_name', 'signal_stats.csv'], wait=True )`	

相關資源

- 使用 Quick Start 加入 Amazon SageMaker Studio (SageMaker 文件)
- 處理資料 (SageMaker 文件)
- 使用 scikit-learn 的資料處理 (SageMaker 文件)
- joblib.Parallel 文件
- Moody、B.、Moody、G.、Villanerroel、M.、Clifford、G. D. 和 Silva、I. (2020)。MIMIC-III 波形資料庫 (1.0 版)。PhysioNet。
- Johnson， A. E. W.， Pollard， T. J.， Shen， L.， Lehman， L. H.， Feng， M.， Ghassemi， M.， Moody， B.， Sezovits， P.， Celi， L. A.， & Mark， R. G. (2016)。MIMIC-III，可免費存取的關鍵護理資料庫。科學資料，3， 160035。
- MIMIC-III Waveform 資料庫授權

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Flask 和 AWS Elastic Beanstalk 視覺化 AI/ML 模型結果

Chris Caudill 和 Durga Sury，Amazon Web Services

摘要

視覺化人工智慧和機器學習 (AI/ML) 服務的輸出通常需要複雜的 API 呼叫，必須由您的開發人員和工程師自訂。如果您的分析師想要快速探索新的資料集，這可能是個缺點。

您可以使用 Web 型使用者介面 (UI)，讓使用者能夠上傳自己的資料，並在儀表板中視覺化模型結果，藉此增強服務的可存取性，並提供更互動式的資料分析形式。

此模式使用 Flask 和 Plotly 將 Amazon Comprehend 與自訂 Web 應用程式整合，並從使用者提供的資料視覺化情緒和實體。模式也提供使用 AWS Elastic Beanstalk 部署應用程式的步驟。您可以使用 Amazon Web Services (AWS) AI 服務或託管在端點（例如，Amazon SageMaker 端點) 上的自訂訓練模型來調整應用程式。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- AWS Command Line Interface (AWS CLI)，安裝在本機機器上並進行設定。如需詳細資訊，請參閱 AWS CLI 文件中的組態基本概念。您也可以使用 AWS Cloud9 整合開發環境 (IDE)；如需詳細資訊，請參閱 AWS Cloud9 文件中的 AWS Cloud9 的 Python 教學課程和在 AWS Cloud9 IDE 中預覽執行中的應用程式。 AWS Cloud9

  注意： AWS Cloud9 不再提供給新客戶。的現有客戶 AWS Cloud9 可以繼續正常使用服務。進一步了解
- 了解 Flask 的 Web 應用程式架構。如需 Flask 的詳細資訊，請參閱 Flask 文件中的 Quickstart。
- Python 3.6 版或更新版本，已安裝並設定。您可以依照 AWS Elastic Beanstalk 文件中設定 Python 開發環境的指示來安裝 Python。
- Elastic Beanstalk 命令列界面 (EB CLI)，已安裝並設定。如需詳細資訊，請參閱 AWS Elastic Beanstalk 文件中的安裝 EB CLI 和設定 EB CLI。

限制

- 此模式的 Flask 應用程式旨在使用使用單一文字資料欄且限制為 200 列的 .csv 檔案。應用程式程式碼可以調整為處理其他檔案類型和資料磁碟區。
- 應用程式不會考慮資料保留，並繼續彙總上傳的使用者檔案，直到手動刪除為止。您可以將應用程式與用於持久性物件儲存的 Amazon Simple Storage Service (Amazon S3) 整合，或使用 Amazon DynamoDB 等資料庫進行無伺服器金鑰值儲存。
- 應用程式只會考慮英文的文件。不過，您可以使用 Amazon Comprehend 來偵測文件的主要語言。如需每個動作支援語言的詳細資訊，請參閱 Amazon Comprehend 文件中的 API 參考。
- 其他資訊 區段提供包含常見錯誤及其解決方案的故障診斷清單。

Architecture

Flask 應用程式架構

Flask 是一種輕量型架構，可用於在 Python 中開發 Web 應用程式。它旨在結合 Python 的強大資料處理與豐富的 Web 使用者介面。模式的 Flask 應用程式說明如何建置 Web 應用程式，讓使用者上傳資料、將資料傳送至 Amazon Comprehend 進行推論，然後視覺化結果。 應用程式具有下列結構：

- `static` – 包含支援 Web UI 的所有靜態檔案 （ 例如 JavaScript、CSS 和映像 ）
- `templates` – 包含應用程式的所有 HTML 頁面
- `userData` – 存放上傳的使用者資料
- `application.py` – Flask 應用程式檔案
- `comprehend_helper.py` – 對 Amazon Comprehend 進行 API 呼叫的函數
- `config.py` – 應用程式組態檔案
- `requirements.txt` – 應用程式所需的 Python 相依性

`application.py` 指令碼包含 Web 應用程式的核心功能，其中包含四個 Flask 路由。下圖顯示這些 Flask 路由。

- `/` 是應用程式的根目錄，將使用者導向 `upload.html`頁面 （ 存放在 `templates`目錄中 ）。
- `/saveFile` 是在使用者上傳檔案後呼叫的路由。此路由會透過 HTML 表單接收`POST`請求，其中包含使用者上傳的檔案。檔案會儲存在 `userData`目錄中，而路由會將使用者重新導向至`/dashboard`路由。

- `/dashboard` 會將使用者傳送至 `dashboard.html`頁面。在此頁面的 HTML 中，它會在 中執行 JavaScript 程式碼`static/js/core.js`，從`/data`路由讀取資料，然後建置頁面的視覺化效果。
- `/data` 是一種 JSON API，提供要在儀表板中視覺化的資料。此路由會讀取使用者提供的資料，並使用 中的 函數將使用者資料`comprehend_helper.py`傳送至 Amazon Comprehend，以進行情緒分析和具名實體辨識 (NER)。Amazon Comprehend 的回應會格式化並傳回為 JSON 物件。

部署架構

設計考量事項

如需在 AWS 雲端上使用 Elastic Beanstalk 部署之應用程式的設計考量詳細資訊，請參閱 AWS Elastic Beanstalk 文件中的 。

技術堆疊

- Amazon Comprehend
- Elastic Beanstalk
- Flask

自動化和擴展

Elastic Beanstalk 部署會自動設定負載平衡器和自動擴展群組。如需更多組態選項，請參閱 AWS Elastic Beanstalk Elastic Beanstalk 文件中的設定 Elastic Beanstalk 環境。

工具

- AWS Command Line Interface (AWS CLI) 是一種統一的工具，可提供與 AWS 所有部分互動的一致界面。
- Amazon Comprehend 使用自然語言處理 (NLP) 擷取文件內容的洞見，而不需要特殊的預先處理。
- AWS Elastic Beanstalk 可協助您在 AWS 雲端中快速部署和管理應用程式，而無需了解執行這些應用程式的基礎設施。
- Elastic Beanstalk CLI (EB CLI) 是 AWS Elastic Beanstalk 的命令列界面，提供互動式命令，以簡化從本機儲存庫建立、更新和監控環境。
- Flask 框架使用 Python 執行資料處理和 API 呼叫，並透過 Plotly 提供互動式 Web 視覺化。

Code

此模式的程式碼可在使用 Flask 和 AWS Elastic Beanstalk 儲存庫的 GitHub 視覺化 AI/ML 模型結果中取得。 AWS Elastic Beanstalk

史詩

設定 Flask 應用程式

任務	Description	所需的技能
複製 GitHub 儲存庫。	執行下列命令，從使用 Flask 和 AWS Elastic Beanstalk 儲存庫的 GitHub 視覺化 AI/ML 模型結果提取應用程式程式碼： AWS Elastic Beanstalk `git clone git@github.com:aws-samples/aws-comprehend-elasticbeanstalk-for-flask.git` **Note** 請務必使用 GitHub 設定 SSH 金鑰。	開發人員
安裝 Python 模組。	複製儲存庫之後，會建立新的本機`aws-comprehend-elasticbeanstalk-for-flask` 目錄。在該目錄中，`requirements.txt` 檔案包含執行應用程式的 Python 模組和版本。使用下列命令來安裝模組： `cd aws-comprehend-elasticbeanstalk-for-flask`	Python 開發人員

任務	Description	所需的技能
	`pip install -r requirements.txt`	

在本機測試應用程式。	執行下列命令來啟動 Flask 伺服器：
	
	`python application.py`
	
	這會傳回執行中伺服器的相關資訊。您應該可以透過開啟瀏覽器並造訪 http：//localhost：5000 來存取應用程式
	
	Note
	如果您在 AWS Cloud9 IDE 中執行應用程式，則需要將 `application.py` 檔案中的 `application.run()` 命令取代為下列行：
	
	`application.run(host=os.getenv('IP', '0.0.0.0'),port=int(os.getenv('PORT', 8080)))`
	
	您必須在部署之前還原此變更。
		Python 開發人員

將應用程式部署至 Elastic Beanstalk

任務	Description	所需的技能
啟動 Elastic Beanstalk 應用程式。	若要以 Elastic Beanstalk 應用程式的形式啟動專案，請從應用程式的根目錄執行下列命令： `eb init -p python-3.6 comprehend_flask --region us-east-1` **⚠ Important** - `comprehend_flask` 是 Elastic Beanstalk 應用程式的名稱，可以根據您的需求進行變更。 - 您可以將 AWS 區域取代為您選擇的區域。如果您未指定區域，則會使用 AWS CLI 中的預設區域。 - 應用程式使用 Python 3.6 版建置。如果您使用其他 Python 版本，可能會遇到錯誤。 執行 `eb init -i`命令以取得更多部署組態選項。	架構師、開發人員
部署 Elastic Beanstalk 環境。	從應用程式的根目錄執行下列命令：	架構師、開發人員

任務	Description	所需的技能
	`eb create comprehend-flask-env` **Note** `comprehend-flask-env` 是 Elastic Beanstalk 環境的名稱，可根據您的需求進行變更。名稱只能包含字母、數字和破折號。	

任務	Description	所需的技能
授權您的部署以使用 Amazon Comprehend。	雖然您的應用程式可能已成功部署，但您也應該為部署提供 Amazon Comprehend 的存取權。 `ComprehendFullAccess` 是一種 AWS 受管政策，可為部署的應用程式提供對 Amazon Comprehend 進行 API 呼叫的許可。 執行下列命令，將`ComprehendFullAccess` 政策連接至 `aws-elasticbeanstalk-ec2-role` （此角色會自動為您部署的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體建立）： `aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/ComprehendFullAccess --role-name aws-elasticbeanstalk-ec2-role` **⚠ Important** `aws-elasticbeanstalk-ec2-role` 會在您的應用程式部署時建立。您必須先完成部署程序，才能連接 AWS Identity and Access	開發人員、安全架構師

任務	Description	所需的技能
	Management (IAM) 政策。	
造訪您部署的應用程式。	在應用程式成功部署後，您可以執行 `eb open`命令來造訪它。 您也可以執行 `eb status`命令來接收部署的詳細資訊。部署 URL 列在 下`CNAME`。	架構師、開發人員

（選用）根據您的 ML 模型自訂應用程式

任務	Description	所需的技能
授權 Elastic Beanstalk 存取新模型。	確定 Elastic Beanstalk 具有新模型端點所需的存取許可。例如，如果您使用 Amazon SageMaker 端點，您的部署需要具有叫用端點的許可。 如需詳細資訊，請參閱 Amazon SageMaker 文件中的 InvokeEndpoint。	開發人員、安全架構師
將使用者資料傳送至新模型。	若要變更此應用程式中的基礎 ML 模型，您必須變更下列檔案： • `comprehend_helper.py` – 這是與 Amazon Comprehend 連線、處理回應並將最終結果傳回給應用程式的 Python 指令碼。在此指令碼中，您可以將資料路	資料科學家

任務	Description	所需的技能
	由到 AWS 雲端上的另一個 AI 服務，也可以將資料傳送到自訂模型端點。我們建議您也為此指令碼中的結果格式化，以進行邏輯分離和此模式的可重複使用性。 • application.py – 如果您變更comprehend_helper.py 指令碼或函數的名稱，則需要更新應用程式application.py 指令碼以反映這些變更。	
更新儀表板視覺效果。	一般而言，整合新的 ML 模型表示必須更新視覺化以反映新的結果。這些變更會在下列檔案中進行： • templates/dashboard.html – 預先建置的應用程式僅考慮兩個基本視覺化。您可以在此檔案中調整頁面的整個配置。 • static/js/core.js – 此指令碼會擷取 Flask 伺服器/data路由的格式化輸出，並使用 Plotly 建立視覺化效果。您可以新增或更新頁面的圖表。	Web 開發人員

（選用）部署更新的應用程式

任務	Description	所需的技能
更新應用程式的需求檔案。	將變更傳送至 Elastic Beanstalk 之前，請在應用程式的根目錄中執行下列命令，來更新 `requirements.txt` 檔案以反映任何新的 Python 模組： `pip freeze > requirements.txt`	Python 開發人員
重新部署 Elastic Beanstalk 環境。	若要確保您的應用程式變更反映在您的 Elastic Beanstalk 部署中，請導覽至應用程式的根目錄並執行下列命令： `eb deploy` 這會將最新版的應用程式程式碼傳送至您現有的 Elastic Beanstalk 部署。	系統管理員、架構師

相關資源

- 使用 Amazon API Gateway 和 AWS Lambda 呼叫 Amazon SageMaker 模型端點 Amazon API Gateway AWS Lambda
- 將 Flask 應用程式部署至 Elastic Beanstalk
- EB CLI 命令參考
- 設定您的 Python 開發環境

其他資訊

故障診斷清單

以下是六個常見的錯誤及其解決方案。

錯誤 1

```
Unable to assume role "arn:aws:iam::xxxxxxxxxx:role/aws-elasticbeanstalk-ec2-role".
 Verify that the role exists and is configured correctly.
```

解決方案：如果在您執行 時發生此錯誤`eb create`，請在 Elastic Beanstalk 主控台上建立範例應用程式，以建立預設執行個體描述檔。如需詳細資訊，請參閱 AWS Elastic Beanstalk Elastic Beanstalk 文件中的建立 Elastic Beanstalk 環境。

錯誤 2

```
Your WSGIPath refers to a file that does not exist.
```

解決方案： 此錯誤發生在部署日誌中，因為 Elastic Beanstalk 預期 Flask 程式碼會命名為`application.py`。如果您選擇不同的名稱，請執行`eb config`並編輯 WSGIPath，如下列程式碼範例所示：

```
aws:elasticbeanstalk:container:python:
     NumProcesses: '1'
     NumThreads: '15'
     StaticFiles: /static/=static/
     WSGIPath: application.py
```

請確定您使用檔案名稱`application.py`取代 。

您也可以利用 Gunicorn 和 Procfile。如需此方法的詳細資訊，請參閱 AWS Elastic Beanstalk Elastic Beanstalk 文件中的使用 Procfile 設定 WSGI 伺服器。

錯誤 3

```
Target WSGI script '/opt/python/current/app/application.py' does not contain WSGI
 application 'application'.
```

解決方案： Elastic Beanstalk 預期代表 Flask 應用程式的變數會命名為 `application`。請確定`application.py`檔案使用 `application`做為變數名稱：

```
application = Flask(__name__)
```

錯誤 4

```
The EB CLI cannot find your SSH key file for keyname
```

解決方案：使用 EB CLI 指定要使用的金鑰對，或為部署的 EC2 執行個體建立金鑰對。若要解決錯誤，請執行 `eb init -i`，其中一個選項會詢問：

```
Do you want to set up SSH for your instances?
```

回應 Y 以建立金鑰對或指定現有的金鑰對。

錯誤 5

我已更新程式碼並重新部署，但我的部署並未反映我的變更。

解決方案：如果您在部署中使用 Git 儲存庫，請務必在重新部署之前新增並遞交變更。

錯誤 6

您正在從 AWS Cloud9 IDE 預覽 Flask 應用程式並發生錯誤。

解決方案： 如需詳細資訊，請參閱 AWS Cloud9 文件中的在 AWS Cloud9 IDE 中預覽執行中的應用程式。 AWS Cloud9

使用 Amazon Comprehend 的自然語言處理

透過選擇使用 Amazon Comprehend，您可以透過執行即時分析或非同步批次任務來偵測個別文字文件中的自訂實體。Amazon Comprehend 也可讓您透過建立端點，即時訓練自訂實體辨識和文字分類模型。

此模式使用非同步批次任務，從包含多個文件的輸入檔案中偵測情緒和實體。此模式提供的範例應用程式旨在讓使用者上傳包含單一資料欄的 .csv 檔案，每列包含一個文字文件。GitHub 中的 `comprehend_helper.py` 檔案使用 Flask 和 AWS Elastic Beanstalk 儲存庫視覺化 AI/ML 模型結果會讀取輸入檔案，並將輸入傳送至 Amazon Comprehend 進行處理。

BatchDetectEntities

Amazon Comprehend 會檢查一批文件的文字是否有具名實體，並傳回偵測到的實體、位置、實體類型，以及指出 Amazon Comprehend 可信度的分數。一次 API 呼叫最多可傳送 25 個文件，每個文

件的大小小於 5，000 個位元組。您可以篩選結果，根據使用案例僅顯示特定實體。例如，您可以略過‘quantity’實體類型，並為偵測到的實體設定閾值分數（例如 0.75)。建議您先探索特定使用案例的結果，再選擇閾值。如需詳細資訊，請參閱 Amazon Comprehend 文件中的 BatchDetectEntities。

BatchDetectSentiment

Amazon Comprehend 會檢查一批傳入文件，並傳回每個文件 (POSITIVE、MIXED、 NEUTRAL或) 的現行情緒NEGATIVE。一次 API 呼叫最多可傳送 25 個文件，每個文件的大小小於 5，000 個位元組。分析情緒非常簡單，您可以選擇要在最終結果中顯示最高分數的情緒。如需詳細資訊，請參閱 Amazon Comprehend 文件中的 BatchDetectSentiment。

Flask 組態處理

Flask 伺服器使用一系列的組態變數來控制伺服器的執行方式。這些變數可包含偵錯輸出、工作階段字符或其他應用程式設定。您也可以定義可在應用程式執行時存取的自訂變數。設定組態變數的方法有多種。

在此模式中，組態是在 中定義config.py，並在 中繼承application.py。

- **Note**

  config.py 包含在應用程式啟動時設定的組態變數。在此應用程式中，會定義DEBUG變數，指示應用程式以偵錯模式執行伺服器。：在生產環境中執行應用程式時，不應使用偵錯模式。 UPLOAD_FOLDER 是自訂變數，定義為稍後在應用程式中參考，並通知應存放上傳使用者資料的位置。

- application.py 會啟動 Flask 應用程式，並繼承 中定義的組態設定config.py。這由下列程式碼執行：

```
application = Flask(__name__)
application.config.from_pyfile('config.py')
```

更多模式

- 使用後台和自助式 Amazon SageMaker AI 範本加速 MLOps
- 使用 Amazon Bedrock 自動化 AWS 基礎設施操作
- 使用 Terraform 在 Amazon Bedrock 上部署代理程式系統，搭配CrewAI 架構
- 在聊天應用程式自訂動作和 中使用 Amazon Q Developer 部署 ChatOps 解決方案來管理 SAST 掃描結果 CloudFormation
- 在 Quick Sight 中使用 AWS Mainframe Modernization 和 Amazon Q 產生資料洞見
- 使用 Quick Sight 中的 AWS Mainframe Modernization 和 Amazon Q 產生 Db2 z/OS 資料洞見
- 讓 SageMaker 筆記本執行個體暫時存取另一個 AWS 帳戶中的 CodeCommit 儲存庫
- 使用 AWS CodePipeline 和 Amazon Bedrock 以程式碼形式管理 AWS Organizations 政策
- 使用 現代化 CardDemo 大型主機應用程式 AWS Transform
- 使用 和 Terraform 現代化 AWS Transform 和部署大型主機應用程式
- 使用 Amazon Redshift ML 執行進階分析
- 使用 Amazon Bedrock 代理程式和 簡化 Amazon EC2 合規管理 AWS Config
- 使用自動化工作流程簡化 Amazon Lex 機器人開發和部署
- AWS Step Functions 使用 Amazon Bedrock 對 中的狀態進行故障診斷

分析

主題

- 在 Microsoft SQL Server Analysis Services 中分析 Amazon Redshift 資料
- 使用 Amazon Athena 和 Amazon Quick Sight 分析並視覺化巢狀 JSON 資料
- 自動化從 AWS Data Exchange 到 Amazon S3 的資料擷取
- 使用 AWS CloudFormation 範本自動化 AWS Glue 中的加密強制執行 AWS CloudFormation
- 建置資料管道，以使用 AWS DataOps 開發套件擷取、轉換和分析 Google Analytics 資料
- 使用 Amazon Kinesis Video Streams 和 AWS Fargate 建置影片處理管道
- 建置 ETL 服務管道，使用 AWS Glue 從 Amazon S3 遞增載入資料至 Amazon Redshift
- 使用 Amazon DataZone 建置企業資料網格 AWS CDK，以及 AWS CloudFormation
- 使用 AWS 服務計算風險值 (VaR)
- 使用 Amazon Athena 設定共用 AWS Glue Data Catalog 的跨帳戶存取權
- 將 Teradata NORMALIZE 暫時功能轉換為 Amazon Redshift SQL
- 將 Teradata RESET WHEN 功能轉換為 Amazon Redshift SQL
- 使用基礎設施做為程式碼，在 AWS 雲端上部署和管理無伺服器資料湖
- 在啟動時強制標記 Amazon EMR 叢集
- 確保在啟動時啟用對 Amazon S3 的 Amazon EMR 記錄
- 使用 AWS Glue 任務和 Python 產生測試資料
- 使用 AWS IoT Greengrass，以經濟實惠的方式直接將 IoT 資料擷取至 Amazon S3 AWS IoT
- 使用 Lambda 函數在暫時性 EMR 叢集中啟動 Spark 任務
- 使用 AWS Glue 將 Apache Cassandra 工作負載遷移至 Amazon Keyspaces
- 從內部部署伺服器將 Oracle Business Intelligence 12c 遷移至 AWS 雲端
- 將 ELK 堆疊遷移至 AWS 上的彈性雲端
- AWS 雲端 使用 Starburst 將資料遷移至
- 在 AWS 上最佳化輸入檔案大小的 ETL 擷取
- 使用 透過驗證、轉換和分割來協調 ETL 管道 AWS Step Functions
- 使用 Amazon Redshift ML 執行進階分析
- 使用 Amazon Athena 查詢具有 SQL 的 Amazon DynamoDB 資料表
- 使用 Athena 存取、查詢和聯結 Amazon DynamoDB 資料表

- 設定最低可行的資料空間以在組織之間共用資料
- 使用純量 Python UDF 設定 Amazon Redshift 查詢結果的語言特定排序
- 訂閱來自不同 AWS 區域中 S3 儲存貯體的事件通知的 Lambda 函數
- 將資料轉換為 Apache Parquet 的三種 AWS Glue ETL 任務類型
- 使用 Amazon Athena 和 Amazon QuickSight 視覺化 Amazon Redshift 稽核日誌
- 使用 Amazon Quick Sight 視覺化所有 AWS 帳戶的 IAM 登入資料報告
- 更多模式

在 Microsoft SQL Server Analysis Services 中分析 Amazon Redshift 資料

Sunil Vora，Amazon Web Services

總結

此模式說明如何使用 Intellisoft OLE 資料庫提供者或 CData ADO.NET 提供者進行資料庫存取，來連接和分析 Microsoft SQL Server Analysis Services 中的 Amazon Redshift 資料。

Amazon Redshift 是一種在雲端中完全受管的 PB 級資料倉儲服務。SQL Server Analysis Services 是一種線上分析處理 (OLAP) 工具，可用來分析來自 Amazon Redshift 等資料區塊和資料倉儲的資料。您可以使用 SQL Server Analysis Services 從資料建立 OLAP 立方體，以進行快速、進階的資料分析。

先決條件和限制

假設

- 此模式說明如何在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上為 Amazon Redshift 設定 SQL Server Analysis Services 和 Intellisoft OLE 資料庫提供者或 CData ADO.NET 提供者。或者，您可以在公司資料中心的主機上安裝兩者。

先決條件

- 作用中的 AWS 帳戶
- 具有登入資料的 Amazon Redshift 叢集

Architecture

來源技術堆疊

- Amazon Redshift 叢集

目標技術堆疊

- Microsoft SQL Server Analysis Services

來源和目標架構

工具

- Microsoft Visual Studio 2019（社群版本）
- 適用於 Amazon Redshift 的 Intellisoft OLE 資料庫提供者（試用）或適用於 Amazon Redshift 的 CData ADO.NET 提供者（試用）

史詩

分析資料表

任務	Description	所需的技能
分析要匯入的資料表和資料。	識別要匯入的 Amazon Redshift 資料表及其大小。	DBA

設定 EC2 執行個體並安裝工具

任務	Description	所需的技能
設定 EC2 執行個體。	在您的 AWS 帳戶中，在私有或公有子網路中建立 EC2 執行個體。	系統管理員
安裝用於資料庫存取的工具。	下載並安裝適用於 Amazon Redshift 的 Intellisoft OLE 資料庫提供者（或適用於 Amazon Redshift 的 CData ADO.NET 提供者)。	系統管理員
安裝 Visual Studio。	下載並安裝 Visual Studio 2019 (Community Edition)。	系統管理員

任務	Description	所需的技能
安裝擴充功能。	在 Visual Studio 中安裝 Microsoft Analysis Services Projects 延伸模組。	系統管理員
建立專案。	在 Visual Studio 中建立新的表格式模型專案，以存放 Amazon Redshift 資料。在 Visual Studio 中，選擇建立專案時的 Analysis Services 表格式專案選項。	DBA

建立資料來源和匯入資料表

任務	Description	所需的技能
建立 Amazon Redshift 資料來源。	使用適用於 Amazon Redshift 的 Intellisoft OLE 資料庫提供者（或適用於 Amazon Redshift 的 CData ADO.NET 提供者）和您的 Amazon Redshift 憑證來建立 Amazon Redshift 資料來源。	Amazon Redshift、DBA
匯入資料表。	選取資料表和檢視，並將其從 Amazon Redshift 匯入 SQL Server Analysis Services 專案。	Amazon Redshift、DBA

遷移後清除

任務	Description	所需的技能
刪除 EC2 執行個體。	刪除您先前啟動的 EC2 執行個體。	系統管理員

相關資源

- Amazon Redshift (AWS 文件)
- 安裝 SQL Server Analysis Services (Microsoft 文件)
- 表格式模型設計工具 (Microsoft 文件)
- 進階分析的 OLAP 立方體概觀 (Microsoft 文件)
- Microsoft Visual Studio 2019 （社群版本）
- Amazon Redshift 的 Intellisoft OLE 資料庫提供者 （試驗）
- Amazon Redshift 的 CData ADO.NET 提供者 （試用）

使用 Amazon Athena 和 Amazon Quick Sight 分析並視覺化巢狀 JSON 資料

Anoop Singh，Amazon Web Services

摘要

此模式說明如何使用 Amazon Athena 將巢狀 JSON 格式的資料結構轉譯為表格式檢視，然後在 Amazon Quick Sight 中視覺化資料。

您可以將 JSON 格式的資料用於來自作業系統的 API 驅動資料饋送，以建立資料產品。此資料也可以協助您更了解客戶及其與產品的互動，因此您可以量身打造使用者體驗並預測結果。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 代表巢狀資料結構的 JSON 檔案 （此模式提供範例檔案）

限制：

- JSON 功能與 Athena 中現有的 SQL 導向函數完美整合。不過，它們與 ANSI SQL 不相容，而且 JSON 檔案預期會將每個記錄放在單獨的一行。您可能需要使用 Athena 中的 `ignore.malformed.json` 屬性來指出格式不正確的 JSON 記錄是否應該變成 null 字元或產生錯誤。 如需詳細資訊，請參閱 Athena 文件中的讀取 JSON 資料的最佳實務。
- 此模式只會考慮簡單和少量的 JSON 格式資料。如果您想要大規模使用這些概念，請考慮套用資料分割，並將您的資料合併成較大的檔案。

Architecture

下圖顯示此模式的架構和工作流程。巢狀資料結構會以 JSON 格式存放在 Amazon Simple Storage Service (Amazon S3) 中。在 Athena 中，JSON 資料會映射至 Athena 資料結構。然後，您可以建立檢視來分析資料，並在 Quick Sight 中視覺化資料結構。

工具

AWS 服務

- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。此模式使用 Amazon S3 來存放 JSON 檔案。
- Amazon Athena 是一種互動式查詢服務，可協助您使用標準 SQL 直接在 Amazon S3 中分析資料。此模式使用 Athena 來查詢和轉換 JSON 資料。透過 中的幾個動作 AWS 管理主控台，您可以將 Athena 指向 Amazon S3 中的資料，並使用標準 SQL 執行一次性查詢。Athena 是無伺服器，因此無需設定或管理基礎設施，您只需為執行的查詢付費。Athena 會自動擴展並平行執行查詢，因此即使使用大型資料集和複雜的查詢，結果也會很快。
- Amazon Quick Sight 是一種雲端規模的商業智慧 (BI) 服務，可協助您在單一儀表板上視覺化、分析和報告資料。Quick Sight 可讓您輕鬆建立和發佈互動式儀表板，其中包含機器學習 (ML) 洞見。您可以從任何裝置存取這些儀表板，並將其內嵌到您的應用程式、入口網站和網站。

範例程式碼

下列 JSON 檔案提供巢狀資料結構，您可以在此模式中使用。

```
{
  "symbol": "AAPL",
  "financials": [
    {
      "reportDate": "2017-03-31",
      "grossProfit": 20591000000,
      "costOfRevenue": 32305000000,
      "operatingRevenue": 52896000000,
      "totalRevenue": 52896000000,
      "operatingIncome": 14097000000,
      "netIncome": 11029000000,
      "researchAndDevelopment": 2776000000,
      "operatingExpense": 6494000000,
      "currentAssets": 101990000000,
      "totalAssets": 334532000000,
      "totalLiabilities": 200450000000,
      "currentCash": 15157000000,
      "currentDebt": 13991000000,
      "totalCash": 67101000000,
      "totalDebt": 98522000000,
      "shareholderEquity": 134082000000,
```

```
      "cashChange": -1214000000,
      "cashFlow": 12523000000,
      "operatingGainsLosses": null
    }
  ]
}
```

史詩

設定 S3 儲存貯體

任務	Description	所需的技能
建立 S3 儲存貯體。	若要建立儲存貯體以存放 JSON 檔案，請登入 AWS 管理主控台，開啟 Amazon S3 主控台，然後選擇建立儲存貯體。如需詳細資訊，請參閱 Amazon S3 文件中的建立儲存貯體。	系統管理員
新增巢狀 JSON 資料。	將您的 JSON 檔案上傳至 S3 儲存貯體。如需範例 JSON 檔案，請參閱上一節。如需說明，請參閱 Amazon S3 文件中的上傳物件。 Amazon S3	系統管理員

分析 Athena 中的資料

任務	Description	所需的技能
建立用於映射 JSON 資料的資料表。	1. 開啟 Athena 主控台。 2. 遵循 Athena 文件中的指示建立資料庫。 3. 從資料庫功能表中，選擇您建立的資料庫。	開發人員

任務	Description	所需的技能
	4. 在查詢編輯器中，輸入如下所示的CREATE TABLE陳述式： `CREATE EXTERNAL TABLE financials_json (symbol string, financials array< struct<reportdate: string, grossprofit: bigint, totalrevenue: bigint, totalcash: bigint, totaldebt: bigint, researchanddevelopment: bigint>>) ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe' LOCATION 's3://s3bucket-for-athena/'` 其中 LOCATION指定包含 JSON 檔案的 S3 儲存貯體位置。 5. 選擇執行以建立資料表。 如需建立資料表的詳細資訊，請參閱 Athena 文件。	

任務	Description	所需的技能
建立用於資料分析的檢視。	1. 開啟 Athena 主控台。 2. 遵循 Athena 文件中的指示建立資料庫。 3. 從資料庫功能表中，選擇您建立的資料庫。 4. 在查詢編輯器中，輸入如下所示的CREATE VIEW陳述式： 5. 選擇 Run (執行) 以建立檢視。 如需建立檢視的詳細資訊，請參閱 Athena 文件。	開發人員

```
CREATE OR REPLACE VIEW
 financial_json_view
 AS
  SELECT
  symbol,
  financials[1].repo
rtdate one_repor
t_date, -- indexes
 start with 1
  financials[1].tota
lrevenue one_total
_revenue,
  financials[1].repo
rtdate another_r
eport_date,
  financials[1].tota
lrevenue another_t
otal_revenue
FROM financials_json
 where symbol='AAPL'
ORDER BY
  1
```

任務	Description	所需的技能
分析和驗證資料。	1. 開啟 Athena 主控台。 2. 在查詢編輯器中，使用您在上一個步驟中建立的檢視來執行查詢。 3. 根據 JSON 檔案驗證資料，以確認資料欄名稱和資料類型已正確映射。	開發人員

在 Quick Sight 中視覺化資料

任務	Description	所需的技能
在 Quick Sight 中將 Athena 設定為資料來源。	1. 開啟 Quick 主控台。 2. 選擇資料集，再選擇新增資料集。 3. 選擇 Athena 做為資料來源。 4. 選擇包含您建立之檢視的資料庫。 5. 選擇您要為其建立資料集的檢視。 6. 在完成資料集建立頁面上，選擇直接查詢您的資料。 7. 選擇 Visualize (視覺化)。	系統管理員
在 Quick Sight 中視覺化資料。	1. 視覺化資料集之後，從左側窗格中選擇視覺效果，然後選擇資料集的欄位。如需詳細資訊，請參閱 Quick 文件中的教學課程。 2. 將變更儲存至分析。	資料分析

任務	Description	所需的技能
	3. 選擇發佈儀表板以發佈您建立的視覺效果。	

相關資源

- Amazon Athena 文件
- Amazon Quick Sight 教學課程
- 使用巢狀 JSON（部落格文章）

自動化從 AWS Data Exchange 到 Amazon S3 的資料擷取

Adnan Alvee 和 Manikanta Gona，Amazon Web Services

總結

此模式提供 CloudFormation 範本，可讓您在 Amazon Simple Storage Service (Amazon S3) 中自動將資料從 擷取 AWS Data Exchange 到資料湖。

AWS Data Exchange 是一項服務，可讓您輕鬆地在 AWS Cloud 中安全地交換檔案型資料集。 AWS Data Exchange 資料集是以訂閱為基礎。身為訂閱者，您也可以在供應商發佈新資料時存取資料集修訂。

CloudFormation 範本會在 Amazon CloudWatch Events 和 AWS Lambda 函數中建立事件。事件會監控您已訂閱之資料集的任何更新。如果有更新，CloudWatch 會啟動 Lambda 函數，將資料複製到您指定的 S3 儲存貯體。成功複製資料後，Lambda 會傳送 Amazon Simple Notification Service (Amazon SNS) 通知給您。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 在 中訂閱資料集 AWS Data Exchange

限制

- 範本 CloudFormation 必須針對其中的每個訂閱資料集分別部署 AWS Data Exchange。

Architecture

目標技術堆疊

- AWS Lambda
- Amazon S3
- AWS Data Exchange
- Amazon CloudWatch

- Amazon SNS

目標架構

自動化和擴展

您可以針對要擷取至資料湖的資料集多次使用 CloudFormation 範本。

工具

- AWS Data Exchange 可讓 AWS 客戶在 中安全地交換檔案型資料集 AWS 雲端。身為訂閱者，您可以從合格的資料提供者找到並訂閱數百種產品。然後，您可以快速下載資料集或將其複製到 Amazon S3，以用於各種 AWS 分析和機器學習服務。任何具有 的人 AWS 帳戶 都可以是 AWS Data Exchange 訂閱者。
- AWS Lambda 可讓您直接執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。您只需為使用的運算時間付費；程式碼未執行時無需付費。使用 Lambda，您可以為幾乎任何類型的應用程式或後端服務執行程式碼，無需管理。Lambda 會在高可用性運算基礎設施上執行您的程式碼，並管理所有運算資源，包括伺服器和作業系統維護、容量佈建和自動擴展、程式碼監控和記錄。
- Amazon S3 為網際網路提供儲存空間。您可以使用 Amazon S3 隨時從 Web 任何地方存放和擷取任意資料量。
- Amazon CloudWatch Events 提供近乎即時的系統事件串流，描述 AWS 資源的變更。使用您可以快速設定的簡單規則，您可以比對事件並將它們路由到一或多個目標函數或串流。CloudWatch Events 在操作變更時會查覺到。它會回應這些操作變更，並視需要採取修正動作，透過傳送訊息來回應環境、啟用 函數、進行變更，以及擷取狀態資訊。您也可以使用 CloudWatch Events 來排程使用 Cron 或 Rate 表達式在特定時間自行啟動的自動化動作。
- Amazon Simple Notification Service (Amazon SNS) 可讓應用程式、最終使用者和裝置立即從雲端傳送和接收通知。Amazon SNS 為高輸送量、以推送為基礎的many-to-many訊息提供主題（通訊管道）。使用 Amazon SNS 主題，發佈者可以將訊息分發給大量訂閱者以進行平行處理，包括 Amazon Simple Queue Service (Amazon SQS) 佇列、Lambda 函數和 HTTP/S Webhook。您也可以使用 Amazon SNS，使用行動推播、簡訊和電子郵件傳送通知給最終使用者。

史詩

訂閱資料集

任務	Description	所需的技能
訂閱資料集。	在 AWS Data Exchange 主控台中，訂閱資料集。如需說明，請參閱 AWS 文件中的在上訂閱資料產品 AWS Data Exchange。	一般 AWS
請記下資料集屬性。	請記下資料集的 、 AWS 區域 ID 和修訂 ID。在下一個步驟中，範本將需要此 CloudFormation 值。	一般 AWS

部署 CloudFormation 範本

任務	Description	所需的技能
建立 S3 儲存貯體和資料夾。	如果您在 Amazon S3 中已有資料湖，請建立資料夾來存放要擷取的資料 AWS Data Exchange。如果您要部署範本進行測試，請建立新的 S3 儲存貯體，並記下下一個步驟的儲存貯體名稱和資料夾字首。	一般 AWS
部署 CloudFormation 範本。	將做為附件提供的 CloudFormation 範本部署至此模式。如需說明，請參閱 CloudFormation 文件。 設定下列參數以對應至您的 AWS 帳戶、資料集和 S3 儲存貯體設定：資料集 AWS 區	一般 AWS

任務	Description	所需的技能
	域、資料集 ID、修訂 ID、S3 儲存貯體名稱（例如 DOC-EXAMPLE-BUCKET ）、資料夾字首（例如 myfolder/ ），以及 SNS 通知的電子郵件。您可以將資料集名稱參數設定為任何名稱。當您部署範本時，它會執行 Lambda 函數，以自動擷取資料集中可用的第一組資料。接著，當新資料抵達資料集時，會自動執行後續擷取。	

相關資源

- 訂閱 上的資料產品 AWS Data Exchange (AWS Data Exchange 文件）

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS CloudFormation 範本自動化 AWS Glue 中的加密強制執行 AWS CloudFormation

Diogo Guedes，Amazon Web Services

總結

此模式說明如何使用 AWS CloudFormation 範本在 AWS Glue 中設定和自動化加密強制執行。AWS CloudFormation 範本會建立強制執行加密所需的所有必要組態和資源。這些資源包括初始組態、Amazon EventBridge 規則建立的預防性控制，以及 AWS Lambda 函數。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 部署 CloudFormation 範本及其資源的許可

限制

此安全控制是區域性的。您必須在要在 AWS Glue 中設定加密強制執行的每個 AWS 區域中部署安全控制。

Architecture

目標技術堆疊

- Amazon CloudWatch Logs （ 來自 AWS Lambda)
- Amazon EventBridge 規則
- AWS CloudFormation 堆疊
- AWS CloudTrail
- AWS Identity and Access Management (IAM) 受管角色和政策
- AWS Key Management Service (AWS KMS)
- AWS KMS 別名
- AWS Lambda 功能

- AWS Systems Manager 參數存放區

目標架構

下圖顯示如何在 AWS Glue 中自動化加密強制執行。

該圖顯示以下工作流程：

1. CloudFormation 範本會建立所有資源，包括 AWS Glue 中加密強制執行的初始組態和偵測控制。
2. EventBridge 規則會偵測加密組態中的狀態變更。
3. 透過 CloudWatch Logs 叫用 Lambda 函數進行評估和記錄。對於不合規偵測，參數存放區會使用 AWS KMS 金鑰的 Amazon Resource Name (ARN) 復原。服務會修復為啟用加密的合規狀態。

自動化和擴展

如果您使用的是 AWS Organizations，則可以使用 AWS CloudFormation StackSets 在多個帳戶中部署此範本，其中您想要在 AWS Glue 中啟用加密強制執行。

工具

- Amazon CloudWatch 可協助您即時監控 AWS 資源的指標，以及您在 AWS 上執行的應用程式。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如，Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他 AWS 帳戶中的事件匯流排。
- AWS CloudFormation 可協助您設定 AWS 資源、快速一致地佈建資源，以及在整個 AWS 帳戶和區域的生命週期進行管理。
- AWS CloudTrail 可協助您啟用 AWS 帳戶的營運和風險稽核、控管和合規。
- AWS Glue 是全受管的擷取、轉換和載入 (ETL) 服務。它可協助您可靠地分類、清理、擴充和移動資料存放區和資料串流之間的資料。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Systems Manager 可協助您管理在 AWS 雲端中執行的應用程式和基礎設施。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。

Code

此模式的程式碼可在 GitHub aws-custom-guardrail-event-driven 儲存庫中使用。

最佳實務

AWS Glue 支援靜態資料加密，以便在 AWS Glue 中編寫任務，並使用開發端點開發指令碼。

請考慮下列最佳實務：

- 設定 ETL 任務和開發端點，以使用 AWS KMS 金鑰寫入靜態加密的資料。
- 使用您透過 AWS KMS 管理的金鑰，加密存放在 AWS Glue Data Catalog 中的中繼資料。
- 使用 AWS KMS 金鑰來加密任務書籤，以及爬蟲程式和 ETL 任務所產生的日誌。

史詩

啟動 CloudFormation 範本

任務	Description	所需的技能
部署 CloudFormation 範本。	從 GitHub 儲存庫下載`aws-custom-guardrail-event-driven.yaml` 範本，然後部署範本。`CREATE_COMPLETE` 狀態表示您的範本已成功部署。 **Note** 範本不需要輸入參數。	雲端架構師

驗證 AWS Glue 中的加密設定

任務	Description	所需的技能
檢查 AWS KMS 金鑰組態。	1. 登入 AWS 管理主控台，然後開啟 AWS Glue 主控台。	雲端架構師

任務	Description	所需的技能
	2. 在導覽窗格中的資料目錄下，選擇目錄設定。 3. 確認中繼資料加密和加密連線密碼設定已標記並設定為使用 KMSKeyGlue 。	

測試加密強制執行

任務	Description	所需的技能
識別 CloudFormation 中的加密設定。	1. 登入 AWS 管理主控台，然後開啟 CloudFormation 主控台。 2. 在導覽窗格中，選擇 Stacks，然後選擇您的堆疊。 3. 選擇 Resources (資源) 標籤。 4. 在資源表格中，依邏輯 ID 尋找加密設定。	雲端架構師
將佈建的基礎設施切換為不合規狀態。	1. 登入 AWS 管理主控台，然後開啟 AWS Glue 主控台。 2. 在導覽窗格中的資料目錄下，選擇目錄設定。 3. 清除中繼資料加密核取方塊。 4. 清除加密連線密碼核取方塊。 5. 選擇儲存。 6. 重新整理 AWS Glue 主控台。	雲端架構師

任務	Description	所需的技能
	在您清除核取方塊後，護欄會偵測 AWS Glue 中的不合規狀態，然後透過自動修復加密設定錯誤來強制執行合規。因此，在您重新整理頁面之後，應該再次選取加密核取方塊。	

相關資源

- 在 AWS CloudFormation 主控台上建立堆疊 (AWS CloudFormation 文件）
- 使用 AWS CloudTrail 建立在 AWS API 呼叫上觸發的 CloudWatch Events 規則 CloudTrail (Amazon CloudWatch 文件）
- 在 AWS Glue 中設定加密 (AWS Glue 文件）

建置資料管道，以使用 AWS DataOps 開發套件擷取、轉換和分析 Google Analytics 資料

Anton Kukushkin 和 Rudy Puig，Amazon Web Services

總結

此模式說明如何使用 AWS DataOps 開發套件 (AWS DDK) 和其他 建置資料管道來擷取、轉換和分析 Google Analytics 資料 AWS 服務。 AWS DDK 是一種開放原始碼開發架構，可協助您建置資料工作流程和現代資料架構 AWS。DDK AWS 的主要目標之一是節省您通常投入大量人力的資料管道任務的時間和精力，例如協調管道、建置基礎設施，以及在該基礎設施背後建立 DevOps。您可以將這些人力密集型任務卸載至 AWS DDK，以便專注於編寫程式碼和其他高價值活動。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 適用於 Google Analytics 的 Amazon AppFlow 連接器，已設定
- Python 和 pip (Python 的套件管理員)
- Git，已安裝和設定
- AWS Command Line Interface (AWS CLI)，已安裝並設定
- AWS Cloud Development Kit (AWS CDK)，已安裝

產品版本

- Python 3.7 或更高版本
- pip 9.0.3 或更新版本

Architecture

技術堆疊

- Amazon AppFlow
- Amazon Athena
- Amazon CloudWatch

- Amazon EventBridge
- Amazon Simple Storage Service (Amazon S3)
- Amazon Simple Queue Service (Amazon SQS)
- AWS DataOps 開發套件 (AWS DDK)
- AWS Lambda

目標架構

下圖顯示擷取、轉換和分析 Google Analytics 資料的事件驅動程序。

該圖顯示以下工作流程：

1. Amazon CloudWatch 排程事件規則會叫用 Amazon AppFlow。
2. Amazon AppFlow 會將 Google Analytics 資料擷取到 S3 儲存貯體。
3. 在 S3 儲存貯體擷取資料之後，EventBridge 中的事件通知就會產生、由 CloudWatch Events 規則擷取，然後放入 Amazon SQS 佇列。
4. Lambda 函數會使用來自 Amazon SQS 佇列的事件、讀取個別的 S3 物件、將物件轉換為 Apache Parquet 格式、將轉換的物件寫入 S3 儲存貯體，然後建立或更新 AWS Glue Data Catalog 資料表定義。
5. Athena 查詢會針對資料表執行。

工具

AWS 工具

- Amazon AppFlow 是一種全受管整合服務，可讓您在軟體即服務 (SaaS) 應用程式之間安全地交換資料。
- Amazon Athena 是一種互動式查詢服務，可協助您使用標準 SQL 直接在 Amazon S3 中分析資料。
- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您執行的應用程式。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如， AWS Lambda 函數、使用 API 目的地的 HTTP 呼叫端點，或其他事件匯流排 AWS 帳戶。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

- Amazon Simple Queue Service (Amazon SQS) 提供安全、耐用且可用的託管佇列，可協助您整合和分離分散式軟體系統和元件。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Cloud Development Kit (AWS CDK) 是在程式碼中定義雲端基礎設施並透過其佈建的架構 CloudFormation。
- AWS DataOps 開發套件 (AWS DDK) 是一種開放原始碼開發架構，可協助您在其中建置資料工作流程和現代資料架構 AWS。

Code

此模式的程式碼可在 GitHub AWS DataOps 開發套件 (AWS DDK) 和使用 Amazon AppFlow、Amazon Athena 和 AWS DataOps 開發套件儲存庫分析 Google Analytics 資料中找到。

史詩

準備環境

任務	Description	所需的技能
複製原始程式碼。	若要複製原始程式碼，請執行下列命令： `git clone https://github.com/aws-samples/aws-ddk-examples.git`	DevOps 工程師
建立虛擬環境。	導覽至原始程式碼目錄，然後執行下列命令來建立虛擬環境： `cd google-analytics-data-using-appflow/python && python3 -m venv .venv`	DevOps 工程師

任務	Description	所需的技能
安裝相依性。	若要啟用虛擬環境並安裝相依性，請執行下列命令： `source .venv/bin/activate && pip install -r requirements.txt`	DevOps 工程師

部署使用資料管道的應用程式

任務	Description	所需的技能
引導環境。	1. 確認 AWS CLI 已為您的 設定有效的登入資料 AWS 帳戶。如需詳細資訊，請參閱 AWS CLI 文件中的使用具名設定檔。 2. 執行 `cdk bootstrap --profile [AWS_PROFILE]` 命令。	DevOps 工程師
部署資料。	若要部署資料管道，請執行 `cdk deploy --profile [AWS_PROFILE]` 命令。	DevOps 工程師

測試部署

任務	Description	所需的技能
驗證堆疊狀態。	1. 開啟 CloudFormation 主控台。 2. 在堆疊頁面上，確認堆疊的狀態DdkAppflo	DevOps 工程師

任務	Description	所需的技能
	wAthenaStack 為 CREATE_COMPLETE 。	

疑難排解

問題	解決方案
部署在建立 AWS::AppFlow::Flow 來源期間失敗，您會收到下列錯誤： Connector Profile with name ga-connection does not exist	確認您已建立適用於 Google Analytics 的 Amazon AppFlow 連接器，並將其命名為 ga-connection 。 如需說明，請參閱 Amazon AppFlow 文件中的 Google Analytics。

相關資源

- AWS DataOps 開發套件 (AWS DDK) (GitHub)
- AWS DDK 範例 (GitHub)

其他資訊

AWS DDK 資料管道由一或多個階段組成。在下列程式碼範例中，您會使用 從 Google Analytics AppFlowIngestionStage 擷取資料、SqsToLambdaStage處理資料轉換，以及AthenaSQLStage執行 Athena 查詢。

首先，會建立資料轉換和擷取階段，如下列程式碼範例所示：

```
        appflow_stage = AppFlowIngestionStage(
            self,
            id="appflow-stage",
            flow_name=flow.flow_name,
        )
        sqs_lambda_stage = SqsToLambdaStage(
            self,
            id="lambda-stage",
```

```
            lambda_function_props={
                "code": Code.from_asset("./ddk_app/lambda_handlers"),
                "handler": "handler.lambda_handler",
                "layers": [
                    LayerVersion.from_layer_version_arn(
                        self,
                        id="layer",
                        layer_version_arn=f"arn:aws:lambda:
{self.region}:336392948345:layer:AWSDataWrangler-Python39:1",
                    )
                ],
                "runtime": Runtime.PYTHON_3_9,
            },
        )
        # Grant lambda function S3 read & write permissions
        bucket.grant_read_write(sqs_lambda_stage.function)
        # Grant Glue database & table permissions
        sqs_lambda_stage.function.add_to_role_policy(
            self._get_glue_db_iam_policy(database_name=database.database_name)
        )
        athena_stage = AthenaSQLStage(
            self,
            id="athena-sql",
            query_string=[
                (
                    "SELECT year, month, day, device, count(user_count) as cnt "
                    f"FROM {database.database_name}.ga_sample "
                    "GROUP BY year, month, day, device "
                    "ORDER BY cnt DESC "
                    "LIMIT 10; "
                )
            ],
            output_location=Location(
                bucket_name=bucket.bucket_name, object_key="query-results/"
            ),
            additional_role_policy_statements=[
                self._get_glue_db_iam_policy(database_name=database.database_name)
            ],
        )
```

接下來，建構會使用 EventBridge `DataPipeline` 規則將階段 "wire" 在一起，如下列程式碼範例所示：

```
        (
            DataPipeline(self, id="ingestion-pipeline")
            .add_stage(
                stage=appflow_stage,
                override_rule=Rule(
                    self,
                    "schedule-rule",
                    schedule=Schedule.rate(Duration.hours(1)),
                    targets=appflow_stage.targets,
                ),
            )
            .add_stage(
                stage=sqs_lambda_stage,
                # By default, AppFlowIngestionStage stage emits an event after the flow
run finishes successfully
                # Override rule below changes that behavior to call the the stage when
data lands in the bucket instead
                override_rule=Rule(
                    self,
                    "s3-object-created-rule",
                    event_pattern=EventPattern(
                        source=["aws.s3"],
                        detail={
                            "bucket": {"name": [bucket.bucket_name]},
                            "object": {"key": [{"prefix": "ga-data"}]},
                        },
                        detail_type=["Object Created"],
                    ),
                    targets=sqs_lambda_stage.targets,
                ),
            )
            .add_stage(stage=athena_stage)
        )
```

如需更多程式碼範例，請參閱 GitHub [使用 Amazon AppFlow、Amazon Athena 和 AWS DataOps 開發套件儲存庫分析 Google Analytics 資料]。

使用 Amazon Kinesis Video Streams 和 AWS Fargate 建置影片處理管道

Piotr Chotkowski 和 Pushparaju Thangavel，Amazon Web Services

總結

此模式示範如何使用 Amazon Kinesis Video Streams 和 AWS Fargate 從影片串流擷取影格，並將其儲存為影像檔案，以便在 Amazon Simple Storage Service (Amazon S3) 中進一步處理。

模式以 Java Maven 專案的形式提供範例應用程式。此應用程式使用 AWS 雲端開發套件 (AWS CDK) 定義 AWS 基礎設施。影格處理邏輯和基礎設施定義都會以 Java 程式設計語言撰寫。您可以使用此範例應用程式做為開發自己的即時影片處理管道的基礎，或建置機器學習管道的影片預先處理步驟。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Java SE 開發套件 (JDK) 11，已安裝
- Apache Maven，已安裝
- AWS 雲端開發套件 (AWS CDK)，已安裝
- AWS Command Line Interface (AWS CLI) 第 2 版，已安裝
- Docker（建置要在 AWS Fargate 任務定義中使用的 Docker 映像時需要），已安裝

限制

此模式旨在做為概念驗證，或做為進一步開發的基礎。它不應在生產部署中以目前的形式使用。

產品版本

- 此模式已使用 AWS CDK 1.77.0 版進行測試（請參閱 AWS CDK 版本）
- JDK 11
- AWS CLI 第 2 版

Architecture

目標技術堆疊

- Amazon Kinesis Video Streams
- AWS Fargate 任務
- Amazon Simple Queue Service (Amazon SQS) 佇列
- Amazon S3 儲存貯體

目標架構

使用者建立 Kinesis 影片串流、上傳影片，並將包含輸入 Kinesis 影片串流和輸出 S3 儲存貯體詳細資訊的 JSON 訊息傳送至 SQS 佇列。AWS Fargate 正在容器中執行主要應用程式，會從 SQS 佇列提取訊息並開始擷取影格。每個影格都會儲存在映像檔案中，並存放在目標 S3 儲存貯體中。

自動化和擴展

範例應用程式可以在單一 AWS 區域內水平和垂直擴展。水平擴展可以透過增加從 SQS 佇列讀取的已部署 AWS Fargate 任務數量來實現。垂直擴展可以透過增加應用程式中框架分割和影像發佈執行緒的數量來實現。這些設定會在 AWS CDK 的 QueueProcessingFargateService 資源定義中做為環境變數傳遞至應用程式。由於 AWS CDK 堆疊部署的本質，您可以在多個 AWS 區域和帳戶中部署此應用程式，無需額外努力。

工具

工具

- AWS CDK 是一種軟體開發架構，可透過使用 TypeScript、JavaScript、Python、Java 和 C#/ 等程式設計語言來定義您的雲端基礎設施和資源。淨額。
- Amazon Kinesis Video Streams 是一項全受管 AWS 服務，可用來將即時影片從裝置串流至 AWS 雲端，或建置應用程式以進行即時影片處理或批次導向影片分析。
- AWS Fargate 是容器的無伺服器運算引擎。Fargate 不需要佈建和管理伺服器，並可讓您專注於開發應用程式。
- Amazon S3 是一種物件儲存服務，可提供可擴展性、資料可用性、安全性和效能。
- Amazon SQS 是一種全受管訊息佇列服務，可讓您解耦和擴展微服務、分散式系統和無伺服器應用程式。

Code

- 已連接範例應用程式專案 (`frame-splitter-code.zip`) 的 .zip 檔案。

史詩

部署基礎設施

任務	Description	所需的技能
啟動 Docker 常駐程式。	在本機系統上啟動 Docker 協助程式。AWS CDK 使用 Docker 來建置 AWS Fargate 任務中使用的映像。您必須先執行 Docker，才能繼續下一個步驟。	開發人員、DevOps 工程師
建置專案。	下載`frame-splitter-code` 範例應用程式（已連接），並將其內容解壓縮至本機電腦上的資料夾。您必須先建置 Java Maven 專案，才能部署基礎設施。在命令提示字元中，導覽至專案的根目錄，然後執行 命令來建置專案： `mvn clean install`	開發人員、DevOps 工程師
引導 AWS CDK。	（僅限第一次使用 AWS CDK 的使用者）如果這是您第一次使用 AWS CDK，您可能需要執行 AWS CLI 命令來引導環境： `cdk bootstrap --profile "$AWS_PROFILE_NAME"`	開發人員、DevOps 工程師

任務	Description	所需的技能
	其中 `$AWS_PROFILE_NAME` 會保留來自您 AWS 登入資料的 AWS 設定檔名稱。或者，您可以移除此參數以使用預設設定檔。如需詳細資訊，請參閱 AWS CDK 文件。	

任務	Description	所需的技能
部署 AWS CDK 堆疊。	在此步驟中，您會在 AWS 帳戶中建立所需的基礎設施資源 (SQS 佇列、S3 儲存貯體、AWS Fargate 任務定義）、建置 AWS Fargate 任務所需的 Docker 映像，並部署應用程式。在命令提示字元中，導覽至專案的根目錄，然後執行 命令： `cdk deploy --profile "$AWS_PROFILE_NAME" --all` 其中 `$AWS_PROFILE_NAME` 會保留來自您 AWS 登入資料的 AWS 設定檔名稱。或者，您可以移除此參數以使用預設設定檔。確認部署。請注意 CDK 部署輸出中的 QueueUrl 和儲存貯體值；後續步驟中會需要這些值。AWS CDK 會建立資產、將其上傳至您的 AWS 帳戶，以及建立所有基礎設施資源。您可以在 AWS CloudFormation 主控台中觀察資源建立程序。如需詳細資訊，請參閱 AWS CloudFormation 文件和 AWS CDK 文件。	開發人員、DevOps 工程師

任務	Description	所需的技能
建立影片串流。	在此步驟中，您會建立 Kinesis 影片串流，做為影片處理的輸入串流。請確定您已安裝並設定 AWS CLI。在 AWS CLI 中，執行： `aws kinesisvideo --profile "$AWS_PROFILE_NAME" create-stream --stream-name "$STREAM_NAME" --data-retention-in-hours "24"` 其中 `$AWS_PROFILE_NAME` 會從您的 AWS 登入資料保留 AWS 設定檔的名稱（或移除此參數以使用預設設定檔），而且 `$STREAM_NAME` 是任何有效的串流名稱。 或者，您可以依照 Kinesis Video Streams 文件中的步驟，使用 Kinesis 主控台建立影片串流。請注意所建立串流的 AWS Resource Name (ARN)；稍後會需要它。	開發人員、DevOps 工程師

執行範例

任務	Description	所需的技能
將影片上傳至串流。	在範例`frame-splitter-code` 應用程式的專案資	開發人員、DevOps 工程師

任務	Description	所需的技能
	料夾中，開啟 `src/test/java/amazon/awscdk/examples/splitter` 資料夾中`ProcessingTaskTest.java` 的檔案。將 `profileName` 和 `streamName` 變數取代為您在先前步驟中使用的值。若要將範例影片上傳至您在上一個步驟中建立的 Kinesis 影片串流，請執行： `amazon.awscdk.examples.splitter.ProcessingTaskTest#testExample test` 或者，您可以使用 Kinesis Video Streams 文件中所述的其中一種方法上傳影片。	

任務	Description	所需的技能

啟動影片處理。	現在您已將影片上傳至 Kinesis 影片串流，您可以開始處理影片。若要啟動處理邏輯，您必須將包含詳細資訊的訊息傳送至 AWS CDK 在部署期間建立的 SQS 佇列。若要使用 AWS CLI 傳送訊息，請執行：
	
	`aws sqs --profile "$AWS_PROFILE_NAME" send-message --queue-url QUEUE_URL --message-body MESSAGE`
	
	其中 `$AWS_PROFILE_NAME` 保留來自您 AWS 登入資料的 AWS 設定檔名稱（移除此參數以使用預設設定檔）、 `QUEUE_URL` 是來自 AWS CDK 輸出的 QueueUrl 值，以及下列格式MESSAGE的 JSON 字串：
	
	`{ "streamARN": "STREAM_ARN", "bucket": "BUCKET_NAME", "s3Directory": "test-output" }`
	
	其中 `STREAM_ARN` 是您在先前步驟中建立之影片串流的 ARN，而 `BUCKET_NAME` 是來自 AWS CDK 輸出的儲存貯體值。
		開發人員、DevOps 工程師

任務	Description	所需的技能
	傳送此訊息會啟動影片處理。或者，您可以使用 Amazon SQS 主控台傳送訊息，如 Amazon SQS 文件所述。	
檢視影片影格的影像。	您可以在 S3 輸出儲存貯體中看到產生的映像，`s3://BUCKET_NAME/test-output` 其中 `BUCKET_NAME` 是來自 AWS CDK 輸出的儲存貯體值。	開發人員、DevOps 工程師

相關資源

- AWS CDK 文件
- AWS CDK API 參考
- AWS CDK 簡介研討會
- Amazon Kinesis Video Streams 文件
- 範例：使用 SageMaker 識別影片串流中的物件
- 範例：剖析和轉譯 Kinesis Video Streams 片段
- 使用 Amazon Kinesis Video Streams 和 Amazon SageMaker 即時大規模分析即時影片 (AWS Machine Learning 部落格文章）
- AWS Fargate 入門

其他資訊

選擇 IDE

我們建議您使用最愛的 Java IDE 來建置和探索此專案。

清除

執行完此範例後，請移除所有部署的資源，以避免產生額外的 AWS 基礎設施成本。

若要移除基礎設施和影片串流，請在 AWS CLI 中使用這兩個命令：

```
cdk destroy --profile "$AWS_PROFILE_NAME" --all
```

```
aws kinesisvideo --profile "$AWS_PROFILE_NAME" delete-stream --stream-arn "$STREAM_ARN"
```

或者，您可以使用 AWS CloudFormation 主控台來移除 AWS CloudFormation 堆疊，以及使用 Kinesis 主控台來移除 Kinesis 影片串流，以手動方式移除資源。請注意， `cdk destroy`不會移除輸出 S3 儲存貯體或 Amazon Elastic Container Registry (Amazon ECR) 儲存庫 () 中的映像`aws-cdk/assets`。您必須手動移除它們。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

建置 ETL 服務管道，使用 AWS Glue 從 Amazon S3 遞增載入資料至 Amazon Redshift

Rohan Jamadagni 和 Arunabha Datta，Amazon Web Services

總結

此模式提供如何設定 Amazon Simple Storage Service (Amazon S3) 以實現最佳資料湖效能的指引，然後使用 AWS Glue，將增量資料變更從 Amazon S3 載入 Amazon Redshift。

Amazon S3 中的來源檔案可以有不同的格式，包括逗號分隔值 (CSV)、XML 和 JSON 檔案。此模式說明如何使用 AWS Glue 將來源檔案轉換為成本最佳化和效能最佳化格式，例如 Apache Parquet。您可以直接從 Amazon Athena 和 Amazon Redshift Spectrum 查詢 Parquet 檔案。您也可以將 Parquet 檔案載入 Amazon Redshift、彙總它們，以及與消費者共用彙總資料，或使用 Amazon Quick Sight 視覺化資料。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 具有正確權限且包含 CSV、XML 或 JSON 檔案的 S3 來源儲存貯體。

假設

- CSV、XML 或 JSON 來源檔案已載入 Amazon S3，可從設定 AWS Glue 和 Amazon Redshift 的帳戶存取。
- 如 Amazon Redshift 文件所述，遵循載入檔案、分割檔案、壓縮和使用資訊清單的最佳實務。
- 來源檔案結構未變更。
- 來源系統能夠遵循 Amazon S3 中定義的資料夾結構，將資料擷取至 Amazon S3。
- Amazon Redshift 叢集跨越單一可用區域。（此架構是適當的，因為 AWS Lambda、AWS Glue 和 Amazon Athena 是無伺服器。）為了實現高可用性，叢集快照會定期拍攝。

限制

- 檔案格式僅限於 AWS Glue 目前支援的檔案格式。
- 不支援即時下游報告。

Architecture

來源技術堆疊

- 具有 CSV、XML 或 JSON 檔案的 S3 儲存貯體

目標技術堆疊

- S3 資料湖（具有分割 Parquet 檔案儲存）
- Amazon Redshift

目標架構

資料流程

工具

- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種高度可擴展的物件儲存服務。Amazon S3 可用於各種儲存解決方案，包括網站、行動應用程式、備份和資料湖。
- AWS Lambda – AWS Lambda 可讓您執行程式碼，而無需佈建或管理伺服器。AWS Lambda 是一種事件驅動的服務；您可以設定程式碼，以從其他 AWS 服務自動啟動。
- Amazon Redshift – Amazon Redshift 是全受管的 PB 級資料倉儲服務。使用 Amazon Redshift，您可以使用標準 SQL 查詢資料倉儲和資料湖中數 PB 的結構化和半結構化資料。
- AWS Glue – AWS Glue 是一種全受管 ETL 服務，可讓您更輕鬆地準備和載入資料以進行分析。AWS Glue 會探索您的資料，並將相關聯的中繼資料（例如資料表定義和結構描述）存放在 AWS Glue Data Catalog 中。您的目錄資料可立即搜尋、查詢，並可用於 ETL。
- AWS Secrets Manager – AWS Secrets Manager 有助於保護和集中管理應用程式或服務存取所需的秘密。服務會存放資料庫登入資料、API 金鑰和其他秘密，無需以純文字格式硬式編碼敏感資訊。Secrets Manager 也提供金鑰輪換，以滿足安全和合規需求。它具有 Amazon Redshift、Amazon Relational Database Service (Amazon RDS) 和 Amazon DocumentDB 的內建整合。您可以使用 Secrets Manager 主控台、命令列界面 (CLI) 或 Secrets Manager API 和 SDKs 來存放和集中管理秘密。

- Amazon Athena – Amazon Athena 是一種互動式查詢服務，可讓您輕鬆分析存放在 Amazon S3 中的資料。Athena 無伺服器並與 AWS Glue 整合，因此可以直接查詢使用 AWS Glue 編目的資料。Athena 會彈性擴展以提供互動式查詢效能。

史詩

建立 S3 儲存貯體和資料夾結構

任務	Description	所需的技能
分析來源系統的資料結構和屬性。	針對有助於 Amazon S3 資料湖的每個資料來源執行此任務。	資料工程師
定義分割區和存取策略。	此策略應以資料擷取的頻率、差異處理和耗用需求為基礎。請確定 S3 儲存貯體未向公眾開放，且存取權僅由特定服務角色型政策控制。如需詳細資訊，請參閱 Amazon S3 說明文件。	資料工程師
為每個資料來源類型建立單獨的 S3 儲存貯體，並為處理過的 (Parquet) 資料為每個來源建立單獨的 S3 儲存貯體。	為每個來源建立單獨的儲存貯體，然後根據來源系統的資料擷取頻率建立資料夾結構，例如 `s3://source-system-name/date/hour` 。對於已處理（轉換為 Parquet 格式）檔案，請建立類似的結構，例如 `s3://source-processed-bucket/date/hour` 。如需建立 S3 儲存貯體的詳細資訊，請參閱 Amazon S3 文件。	資料工程師

在 Amazon Redshift 中建立資料倉儲

任務	Description	所需的技能
使用適當的參數群組以及維護和備份策略啟動 Amazon Redshift 叢集。	建立 Amazon Redshift 叢集時，針對管理員使用者憑證使用 Secrets Manager 資料庫秘密。如需有關建立和調整 Amazon Redshift 叢集大小的資訊，請參閱 Amazon Redshift 文件和調整雲端資料倉儲大小白皮書。	資料工程師
建立 IAM 服務角色並將其連接至 Amazon Redshift 叢集。	AWS Identity and Access Management (IAM) 服務角色可確保存取 Secrets Manager 和來源 S3 儲存貯體。如需詳細資訊，請參閱授權和新增角色的 AWS 文件。	資料工程師
建立資料庫結構描述。	遵循資料表設計的 Amazon Redshift 最佳實務。根據使用案例，選擇適當的排序和分佈索引鍵，以及最佳的壓縮編碼。如需最佳實務，請參閱 AWS 文件。	資料工程師
設定工作負載管理。	根據您的需求設定工作負載管理 (WLM) 佇列、短查詢加速 (SQA) 或並行擴展。如需詳細資訊，請參閱《Amazon Redshift 文件》中的實作工作負載管理。	資料工程師

在 Secrets Manager 中建立秘密

任務	Description	所需的技能
建立新的秘密，將 Amazon Redshift 登入憑證存放在 Secrets Manager 中。	此秘密會存放管理員使用者以及個別資料庫服務使用者的登入資料。如需說明，請參閱 Secrets Manager 文件。選擇 Amazon Redshift 叢集做為秘密類型。此外，在秘密輪換頁面上，開啟輪換。這將在 Amazon Redshift 叢集中建立適當的使用者，並以定義的間隔輪換金鑰秘密。	資料工程師
建立 IAM 政策以限制 Secrets Manager 存取。	限制 Secrets Manager 只能存取 Amazon Redshift 管理員和 AWS Glue。	資料工程師

設定 AWS Glue

任務	Description	所需的技能
在 AWS Glue Data Catalog 中，新增 Amazon Redshift 的連線。	如需說明，請參閱 AWS Glue 文件。	資料工程師
建立並連接 IAM 服務角色，讓 AWS Glue 存取 Secrets Manager、Amazon Redshift 和 S3 儲存貯體。	如需詳細資訊，請參閱 AWS Glue 文件。	資料工程師
定義來源的 AWS Glue Data Catalog。	此步驟涉及在 AWS Glue Data Catalog 中建立資料庫和所需的資料表。您可以使用爬蟲程式來編目 AWS Glue 資料庫中的資料表，或將其定義為	資料工程師

任務	Description	所需的技能
	Amazon Athena 外部資料表。您也可以透過 AWS Glue Data Catalog 存取 Athena 中定義的外部資料表。如需在 Athena 中定義 Data Catalog 和建立外部資料表的詳細資訊，請參閱 AWS 文件。	
建立 AWS Glue 任務來處理來源資料。	AWS Glue 任務可以是 Python shell 或 PySpark，以標準化、刪除重複和清除來源資料檔案。若要最佳化效能並避免必須查詢整個 S3 來源儲存貯體，請依日期分割 S3 儲存貯體，並依年、月、日和小時細分，做為 AWS Glue 任務的下推述詞。如需詳細資訊，請參閱 AWS Glue 文件。將已處理和轉換的資料載入 Parquet 格式的已處理 S3 儲存貯體分割區。您可以從 Athena 查詢 Parquet 檔案。	資料工程師
建立 AWS Glue 任務以將資料載入 Amazon Redshift。	AWS Glue 任務可以是 Python shell 或 PySpark，透過維護資料載入資料，然後進行完整重新整理。如需詳細資訊，請參閱 AWS Glue 文件和其他資訊一節。	資料工程師

任務	Description	所需的技能
（選用）視需要使用觸發條件來排程 AWS Glue 任務。	增量資料載入主要是由導致 AWS Lambda 函數呼叫 AWS Glue 任務的 Amazon S3 事件所驅動。針對需要時間型而非事件型排程的任何資料載入，使用 AWS Glue 觸發型排程。	資料工程師

建立 Lambda 函數

任務	Description	所需的技能
建立並連接 AWS Lambda 的 IAM 服務連結角色，以存取 S3 儲存貯體和 AWS Glue 任務。	為 AWS Lambda 建立 IAM 服務連結角色，其中包含讀取 Amazon S3 物件和儲存貯體的政策，以及存取 AWS Glue API 以啟動 AWS Glue 任務的政策。如需詳細資訊，請參閱知識中心。	資料工程師
建立 Lambda 函數，根據定義的 Amazon S3 事件執行 AWS Glue 任務。	Lambda 函數應該透過建立 Amazon S3 資訊清單檔案來啟動。Lambda 函數應將 Amazon S3 資料夾位置（例如 source_bucket/year/month/date/hour) 做為參數傳遞至 AWS Glue 任務。AWS Glue 任務會使用此參數做為下推述詞，以最佳化檔案存取和任務處理效能。如需詳細資訊，請參閱 AWS Glue 文件。	資料工程師
建立 Amazon S3 PUT 物件事件以偵測物件建立，並呼叫個別的 Lambda 函數。	Amazon S3 PUT 物件事件只能透過建立資訊清單檔案來啟動。資訊清單檔案控制 Lambda 函數和 AWS Glue 任	資料工程師

任務	Description	所需的技能
	務並行，並以批次方式處理負載，而不是處理抵達 S3 來源儲存貯體特定分割區的個別檔案。如需詳細資訊，請參閱 Lambda 文件。	

相關資源

- Amazon S3 文件
- AWS Glue 文件
- Amazon Redshift 文件
- AWS Lambda
- Amazon Athena
- AWS Secrets Manager

其他資訊

upsert 和完整重新整理的詳細方法

Upsert：這適用於需要歷史彙總的資料集，取決於業務使用案例。根據您的業務需求，遵循更新和插入新資料 (Amazon Redshift 文件） 中所述的方法之一。

完整重新整理：這適用於不需要歷史彙總的小型資料集。請遵循下列其中一種方法：

1. 截斷 Amazon Redshift 資料表。
2. 從預備區域載入目前的分割區

或：

1. 使用目前的分割區資料建立暫存資料表。
2. 捨棄目標 Amazon Redshift 資料表。
3. 將暫存資料表重新命名為目標資料表。

使用 Amazon DataZone 建置企業資料網格 AWS CDK，以及 AWS CloudFormation

Dhrubajyoti Mukherjee、Ravi Kumar、WeiORDER Sun 和 Adjoa Taylor，Amazon Web Services

摘要

在 Amazon Web Services (AWS) 上，客戶了解資料是加速企業創新和推動商業價值的關鍵。若要管理此大量資料，您可以採用分散式架構，例如資料網格。資料網格架構可促進產品思維，這是一種將客戶、目標和市場納入考量的思維方式。資料網格也有助於建立聯合控管模型，以快速、安全地存取您的資料。

在 上建置資料網格型企業解決方案的策略 AWS討論了如何使用資料網格策略架構來為您的組織制定和實作資料網格策略。透過使用資料網格策略架構，您可以最佳化團隊組織及其互動，以加速您的資料網格之旅。

本文件提供如何使用 Amazon DataZone 建置企業資料網格的指引。Amazon DataZone 是一種資料管理服務，用於編目 AWS、探索、共用和管理跨內部部署和第三方來源存放的資料。模式包含程式碼成品，可協助您使用 AWS Cloud Development Kit (AWS CDK) 和 部署資料網格型資料解決方案基礎設施 AWS CloudFormation。此模式適用於雲端架構師和 DevOps 工程師。

如需此模式目標和解決方案範圍的相關資訊，請參閱其他資訊一節。

先決條件和限制

先決條件

- 至少兩個作用中 AWS 帳戶：一個用於中央控管帳戶，另一個用於成員帳戶
- AWS 開發環境中中央控管帳戶的管理員登入資料
- AWS Command Line Interface AWS 服務 安裝 https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html (AWS CLI) 從命令列管理
- 安裝 Node.js 和 Node Package Manager (npm) https://docs.npmjs.com/downloading-and-installing-node-js-and-npm來管理 AWS CDK 應用程式
- AWS CDK 使用 npm 在開發環境中全域安裝的工具組，以合成和部署 AWS CDK 應用程式

```
npm install -g aws-cdk
```

- 安裝在開發環境中的 Python 3.12 版

- 安裝在開發環境中或使用 npm 編譯器全域安裝的 TypeScript：

```
npm install -g typescript
```

- 安裝在開發環境中的 Docker
- 版本控制系統，例如 Git，以維護解決方案的原始程式碼 （ 建議 ）
- 支援 Python 和 TypeScript 的整合式開發環境 (IDE) 或文字編輯器 （ 強烈建議 ）

限制

- 解決方案僅在執行 Linux 或 macOS 的機器上進行測試。
- 在目前版本中，解決方案 AWS IAM Identity Center 預設不支援 Amazon DataZone 和 的整合。不過，您可以將其設定為支援此整合。

產品版本

- Python 3.12 版

Architecture

下圖顯示資料網格參考架構。架構是以 Amazon DataZone 為基礎，並使用 Amazon Simple Storage Service (Amazon S3) 和 AWS Glue Data Catalog 作為資料來源。根據組織的需求， AWS 服務 您在資料網格實作中與 Amazon DataZone 搭配使用的 可能會有所不同。

1. 在生產者帳戶中，原始資料適合目前形式的取用，或使用 進行轉換以供取用 AWS Glue。資料的技術中繼資料存放在 Amazon S3 中，並使用 AWS Glue 資料爬蟲程式進行評估。使用 Data Quality 測量資料AWS Glue 品質。Data Catalog 中的來源資料庫會註冊為 Amazon DataZone 目錄中的資產。Amazon DataZone 目錄使用 Amazon DataZone 資料來源任務託管在中央控管帳戶中。
2. 中央控管帳戶託管 Amazon DataZone 網域和 Amazon DataZone 資料入口網站。資料生產者和消費者 AWS 帳戶 的 與 Amazon DataZone 網域相關聯。資料生產者和消費者的 Amazon DataZone 專案是以對應的 Amazon DataZone 網域單位組織。
3. 資料資產的最終使用者使用其 AWS Identity and Access Management (IAM) 登入資料或單一登入 （ 透過 IAM Identity Center 整合 ） 登入 Amazon DataZone 資料入口網站。它們會在 Amazon DataZone 資料目錄中搜尋、篩選和檢視資產資訊 （ 例如，資料品質資訊或業務和技術中繼資料 ）。

4. 最終使用者找到所需的資料資產後，他們會使用 Amazon DataZone 訂閱功能來請求存取。生產者團隊的資料擁有者會收到通知，並在 Amazon DataZone 資料入口網站中評估訂閱請求。資料擁有者會根據訂閱請求的有效性來核准或拒絕訂閱請求。
5. 授予並履行訂閱請求後，會在消費者帳戶中存取資產以進行下列活動：
   - 使用 Amazon SageMaker AI 開發 AI/ML 模型
   - 使用 Amazon Athena 和 Amazon Quick 進行分析和報告

工具

AWS 服務

- Amazon Athena 是一種互動式查詢服務，可協助您使用標準 SQL 直接在 Amazon Simple Storage Service (Amazon S3) 中分析資料。
- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端 基礎設施。
- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。
- Amazon DataZone 是一項資料管理服務，可協助您編目、探索、共用和控管跨 AWS、內部部署和第三方來源存放的資料。
- Amazon Quick 是一種雲端規模的商業智慧 (BI) 服務，可協助您在單一儀表板中視覺化、分析和報告您的資料。
- Amazon SageMaker AI 是一種受管機器學習 (ML) 服務，可協助您建置和訓練 ML 模型，然後將模型部署到生產就緒的託管環境中。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Amazon Simple Queue Service (Amazon SQS) 提供安全、耐用且可用的託管佇列，可協助您整合和分離分散式軟體系統和元件。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

程式碼儲存庫

解決方案可在 GitHub data-mesh-datazone-cdk-cloudformation 儲存庫中使用。

史詩

設定環境

任務	Description	所需的技能
複製儲存庫。	若要複製儲存庫，請在本機開發環境 (Linux 或 macOS) 中執行下列命令： `git clone https://github.com/aws-samples/data-mesh-data zone-cdk-cloudformation`	雲端架構師、DevOps 工程師

建立環境。	若要建立 Python 虛擬環境，請執行下列命令：
	`python3 -m venv .venv`
	`source .venv/bin/activate`
	`pip install -r requirements.txt`
		雲端架構師、DevOps 工程師

引導帳戶。	若要使用 引導中央控管帳戶 AWS CDK，請執行下列命令： `cdk bootstrap aws://<GOVERNANCE_ACCOUNT_ID>/<AWS_REGION>` 登入 AWS 管理主控台，開啟中央控管帳戶主控台，並取得 AWS CDK 執行角色的 Amazon Resource Name (ARN)。	雲端架構師、DevOps 工程師

任務	Description	所需的技能
建構 DzDataMeshMemberStackSet.yaml 檔案。	若要建構DzDataMeshMemberStackSet.yaml 檔案，請從儲存庫的根目錄啟動下列 bash 指令碼： `./lib/scripts/create_dz_data_mesh_member_stack_set.sh`	雲端架構師、DevOps 工程師
確認範本建立。	確定 CloudFormation 範本檔案是在 lib/cfn-templates/DzDataMeshMemberStackSet.yaml 位置建立的。	雲端架構師、DevOps 工程師

在中央控管帳戶中部署資源

任務	Description	所需的技能

修改組態。	在 config/Config.ts 檔案中，修改下列參數：
	`DZ_APPLICATION_NAME - Name of the application.`
	`DZ_STAGE_NAME - Name of the stage.`
	`DZ_DOMAIN_NAME - Name of the Amazon DataZone domain`
	`DZ_DOMAIN_DESCRIPTION - Description of the Amazon DataZone domain`
	`DZ_DOMAIN_TAG - Tag of the Amazon DataZone domain`
		雲端架構師、DevOps 工程師

任務	Description	所需的技能
	DZ_ADMIN_PROJECT_NAM E - Name of the Amazon DataZone project for administrators DZ_ADMIN_PROJECT_DESC RIPTION - Description of the Amazon DataZone project for administr ators CDK_EXEC_ROLE_ARN - ARN of the cdk execution role DZ_ADMIN_ROLE_ARN - ARN of the administr ator role 將剩餘的參數保留空白。	

任務	Description	所需的技能
更新 Amazon DataZone 詞彙表組態。	若要更新 `lib/utils/glossary_config.json` 檔案中的 Amazon DataZone 詞彙表組態，請使用下列範例組態： `{ "GlossaryName": "PII Data", "GlossaryDescription": "If data source contains PII attributes", "GlossaryTerms": [{ "Name": "Yes", "ShortDescription": "Yes", "LongDescription": "Yes Glossary Term" }, { "Name": "No", "ShortDescription": "No", "LongDescription": "No Glossary Term" } ] }`	雲端架構師、DevOps 工程師

任務	Description	所需的技能
更新 Amazon DataZone 中繼資料表單組態。	若要更新 中的 Amazon DataZone 中繼資料表單組態`lib/utils/metadata_form_config.json` file，請使用下列範例組態： `{ "FormName": "ScheduleDataRefresh", "FormDescription": "Form for data refresh schedule", "FormSmithyModel": "@amazon.datazone#displayname(defaultName: \"Data Refresh Schedule\")\nstructure ScheduleDataRefresh {\n @documentation(\"Schedule of Data Refresh\")\n @required\n @amazon.datazone#searchable\n @amazon.datazone#displayname(defaultName: \"Data Refresh Schedule\")\n data_refresh_schedule: String\n}" }`	雲端架構師、DevOps 工程師

任務	Description	所需的技能
匯出 AWS 登入資料。	若要使用管理許可將 AWS 登入資料匯出至 IAM 角色的開發環境，請使用下列格式： `export AWS_ACCES S_KEY_ID= export AWS_SECRE T_ACCESS_KEY= export AWS_SESSI ON_TOKEN=`	雲端架構師、DevOps 工程師
合成範本。	若要合成 CloudFormation 範本，請執行下列命令： `npx cdk synth`	雲端架構師、DevOps 工程師
部署解決方案。	若要部署解決方案，請執行下列命令： `npx cdk deploy --all`	雲端架構師、DevOps 工程師

設定新的成員帳戶

任務	Description	所需的技能
部署 範本。	使用下列輸入參數部署位於成員帳戶中 `lib/cfn-t emplates/DzDataMes hCfnStackSetExecut ionRole.yaml` 的 CloudFormation 範本： • `GovernanceAccountID` – 控管帳戶的帳戶 ID	雲端架構師、DevOps 工程師

任務	Description	所需的技能
	• `DataZoneKMSKeyID` – 加密 Amazon DataZone 中繼資料的 AWS Key Management Service (AWS KMS) 金鑰 ID • `NotificationQueueName` – 控管帳戶中 Amazon SQS 通知佇列的名稱	
更新 ARNs。	若要更新成員帳戶的 CloudFormation StackSet 執行角色 ARNs清單，請使用下列程式碼： `DZ_MEMBER_STACK_SET_EXEC_ROLE_LIST - List of Stack set execution role arns for the member accounts.`	雲端架構師、DevOps 工程師

合成和部署。	若要合成 CloudFormation 範本並部署解決方案，請執行下列命令：
	`npx cdk synth`
	`npx cdk deploy --all`
		雲端架構師、DevOps 工程師

任務	Description	所需的技能
關聯成員帳戶。	若要將成員帳戶與中央控管帳戶建立關聯，請執行下列動作： 1. 登入中央控管帳戶的主控台，然後開啟位於 https：//https://console.aws.amazon.com/datazone/ 的 Amazon DataZone 主控台。 2. 選擇您建立的網域。 3. 捲動至關聯帳戶索引標籤，然後選擇請求關聯。 4. 提供 AWS 帳戶 ID，然後選擇 AWSRAMPermissionDataZonePortalReadWrite 做為 RAM 政策。 5. 選擇請求關聯。 6. 請等到您收到電子郵件通知，告知您的 帳戶已成功啟動。	雲端架構師、DevOps 工程師

任務	Description	所需的技能
更新參數。	若要更新 組態檔案中的成員帳戶特定參數config/Config.ts ，請使用下列格式： `export const DZ_MEMBER_ACCOUNT_CONFIG: memberAccountConfig = { '123456789012' : { PROJECT_NAME: 'TEST-PROJECT-123456789012', PROJECT_DESCRIPTION: 'TEST-PROJECT-123456789012', PROJECT_EMAIL: 'user@xyz.com' } }`	雲端架構師、DevOps 工程師

合成和部署範本。	若要合成 CloudFormation 範本並部署解決方案，請執行下列命令：
	
	`npx cdk synth`
	`npx cdk deploy --all`
		雲端架構師、DevOps 工程師

新增成員帳戶。	若要在資料解決方案中建立和設定其他成員帳戶，請為每個成員帳戶重複上述步驟。 此解決方案不會區分資料生產者和消費者。	雲端架構師、DevOps 工程師

清除

任務	Description	所需的技能
取消成員帳戶的關聯。	若要取消帳戶關聯，請執行下列動作： 1. 登入 主控台並開啟 Amazon DataZone 主控台。 2. 選擇檢視網域。 3. 選取您建立的網域。 4. 選擇帳戶關聯索引標籤。 5. 選取您要取消關聯的成員帳戶。 6. 選擇取消關聯，然後輸入 `disassociate` 進行確認。 7. 對所有成員帳戶重複步驟 3-6。	雲端架構師、DevOps 工程師
刪除堆疊執行個體。	若要刪除 CloudFormation 堆疊執行個體，請執行下列動作： 1. 在 https：//https://console.aws.amazon.com/cloudformation/ 開啟 CloudFormation 主控台。 2. 在導覽窗格中，選擇 StackSets。 3. 選擇名為 StackSet-DataZone-DataMesh-Member 的堆疊集，然後選擇堆疊執行個體索引標籤。 4. 複製您要從成員資格中移除 的成員帳戶 AWS 帳戶 ID。	雲端架構師、DevOps 工程師

任務	Description	所需的技能
	5. 選擇動作，選擇從 StackSet 刪除堆疊，然後保留預設選項。 6. 在帳戶號碼欄位中，輸入帳戶 ID。 7. 在指定區域下拉式清單中，選擇 AWS 區域。 8. 選擇下一步，然後選擇提交。 9. 在操作索引標籤上，確認操作已成功。堆疊刪除可能需要一些時間。 10對所有成員帳戶重複步驟 2-9。	
銷毀所有資源。	若要銷毀資源，請在本機開發環境 (Linux 或 macOS) 中實作下列步驟： 1. 導覽至儲存庫的根目錄。 2. 匯出您用來建立 AWS CDK 堆疊之 IAM 角色的 AWS 登入資料。 3. 若要銷毀雲端資源，請執行下列命令： `npx cdk destroy --all`	雲端架構師、DevOps 工程師

相關資源

- Amazon DataZone AWS Glue 資料快速入門
- 教學課程：建立您的第一個 AWS CDK 應用程式
- 入門 CloudFormation

- 在 上建置資料網格型企業解決方案的策略 AWS

其他資訊

目標

實作此模式可達成下列目標：

- 分散式資料擁有權：將資料擁有權從中央團隊轉移到代表組織來源系統、業務單位或使用案例的團隊。
- 產品思維 – 在考慮組織中的資料資產時，引進以產品為基礎的思維，其中包括客戶、市場和其他因素。
- 聯合控管 – 改善整個組織資料產品的安全防護機制、控制和合規性。
- 多帳戶和多專案支援 – 支援跨組織業務單位或專案的高效、安全的資料共用和協作。
- 集中監控和通知 – 使用 Amazon CloudWatch 監控資料網格的雲端資源，並在新成員帳戶相關聯時通知使用者。
- 可擴展性和可擴展性 – 隨著組織發展，將新的使用案例新增至資料網格。

解決方案範圍

當您使用此解決方案時，您可以啟動小型 並隨著資料網格旅程的進展進行擴展。通常，當成員帳戶採用資料解決方案時，它包含組織、專案或業務單位特定的帳戶組態。此解決方案支援下列功能，以容納這些不同的 AWS 帳戶 組態：

- AWS Glue Data Catalog 作為 Amazon DataZone 的資料來源
- Amazon DataZone 資料網域和相關資料入口網站的管理
- 在資料網格型資料解決方案中新增成員帳戶的管理
- Amazon DataZone 專案和環境的管理
- Amazon DataZone 詞彙表和中繼資料表單的管理
- 與資料網格型資料解決方案使用者對應的 IAM 角色管理
- 資料網格型資料解決方案使用者的通知
- 監控佈建的雲端基礎設施

  此解決方案使用 AWS CDK 和 CloudFormation 部署雲端基礎設施。它使用 CloudFormation 執行下列動作：

- 在較低的抽象層級定義和部署雲端資源。
- 從 部署雲端資源 AWS 管理主控台。透過使用此方法，您可以在沒有開發環境的情況下部署基礎設施。

資料網格解決方案使用 在較高的抽象層級 AWS CDK 定義資源。因此，解決方案透過選擇部署雲端資源的相關工具，提供解耦、模組化和可擴展的方法。

後續步驟

您可以聯絡 AWS專家，以取得使用 Amazon DataZone 建置資料網格 的指引。

此解決方案的模組化性質支援使用不同的架構來建置資料管理解決方案，例如資料結構和資料湖。此外，根據您的組織需求，您可以將解決方案擴展到其他 Amazon DataZone 資料來源。

使用 AWS 服務計算風險值 (VaR)

Sumon Samanta，Amazon Web Services

總結

此模式說明如何使用 AWS 服務實作風險值 (VaR) 計算系統。在內部部署環境中，大多數 VaR 系統使用大型的專用基礎設施，以及內部或商業網格排程軟體來執行批次程序。此模式提供簡單、可靠且可擴展的架構，以處理 AWS 雲端中的 VaR 處理。其建置的無伺服器架構使用 Amazon Kinesis Data Streams 做為串流服務、Amazon Simple Queue Service (Amazon SQS) 做為受管佇列服務、Amazon ElastiCache 做為快取服務，以及 AWS Lambda 來處理訂單和計算風險。

VaR 是交易者和風險經理用來估計其產品組合中潛在損失超出特定可信度水準的統計指標。大多數 VaR 系統涉及執行大量數學和統計計算並儲存結果。這些計算需要大量的運算資源，因此 VaR 批次程序必須分成較小的運算任務集。您可以將大型批次分割為較小的任務，因為這些任務大部分是獨立的（也就是說，一個任務的計算不會依賴其他任務）。

VaR 架構的另一個重要需求是運算可擴展性。此模式使用無伺服器架構，可根據運算負載自動縮減或縮減規模。由於批次或線上運算需求難以預測，因此需要動態擴展才能在服務層級協議 (SLA) 規定的時間內完成程序。此外，成本最佳化架構應該能夠在該資源上的任務完成後，立即縮減每個運算資源。

AWS 服務非常適合 VaR 計算，因為它們提供可擴展的運算和儲存容量、以成本最佳化的方式處理的分析服務，以及執行風險管理工作流程的不同類型排程器。此外，您只需為在 AWS 上使用的運算和儲存資源付費。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 輸入檔案，這取決於您的業務需求。典型的使用案例包含下列輸入檔案：
  - 市場資料檔案（輸入至 VaR 計算引擎）
  - 交易資料檔案（除非交易資料透過串流）。
  - 組態資料檔案（模型和其他靜態組態資料）
  - 計算引擎模型檔案（定量程式庫）
  - 時間序列資料檔案（適用於歷史資料，例如過去五年的股票價格）

- 如果市場資料或其他輸入透過串流傳入、Amazon Kinesis Data Streams 設定，以及設定為寫入串流的 Amazon Identity and Access Management (IAM) 許可。

此模式會建置一種架構，其中交易資料會從交易系統寫入 Kinesis 資料串流。您可以不使用串流服務，將交易資料儲存在小型批次檔案中、將其存放在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中，並叫用事件以開始處理資料。

限制

- 每個碎片都保證 Kinesis 資料串流排序，因此寫入多個碎片的交易訂單不保證會以與寫入操作相同的順序交付。
- AWS Lambda 執行時間限制目前為 15 分鐘。（如需詳細資訊，請參閱 Lambda 常見問答集。）

Architecture

目標架構

下列架構圖顯示風險評估系統的 AWS 服務和工作流程。

此圖展示了以下要點：

1. 從訂單管理系統傳入交易串流。
2. 票證位置網路 Lambda 函數會處理訂單，並將每個刻度的合併訊息寫入 Amazon SQS 中的風險佇列。
3. 風險計算引擎 Lambda 函數會處理來自 Amazon SQS 的訊息、執行風險計算，以及更新 Amazon ElastiCache 風險快取中的 VaR 損益 (PnL) 資訊。
4. 讀取 ElastiCache 資料 Lambda 函數會從 ElastiCache 擷取風險結果，並將其存放在資料庫和 S3 儲存貯體中。

如需這些服務和步驟的詳細資訊，請參閱 Epics 一節。

自動化和擴展

您可以使用 AWS Cloud Development Kit (AWS CDK) 或 AWS CloudFormation 範本來部署整個架構。架構可以同時支援批次處理和當日（即時）處理。

擴展內建於 架構中。隨著更多交易寫入 Kinesis 資料串流並等待處理，可以叫用其他 Lambda 函數來處理這些交易，然後在處理完成後縮減規模。透過多個 Amazon SQS 風險計算佇列進行處理也是一個選項。如果需要跨佇列嚴格排序或整合，則無法平行處理。不過，對於end-of-the-day批次或迷你當日批次，Lambda 函數可以平行處理並將最終結果存放在 ElastiCache 中。

工具

AWS 服務

- Amazon Aurora MySQL 相容版本是完全受管的 MySQL 相容關聯式資料庫引擎，可協助您設定、操作和擴展 MySQL 部署。此模式使用 MySQL 做為範例，但您可以使用任何 RDBMS 系統來存放資料。
- Amazon ElastiCache 可協助您在 AWS 雲端中設定、管理和擴展分散式記憶體內快取環境。
- Amazon Kinesis Data Streams 可協助您即時收集和處理大型資料記錄串流。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- Amazon Simple Queue Service (Amazon SQS) 提供安全、耐用且可用的託管佇列，可協助您整合和分離分散式軟體系統和元件。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

Code

此模式為 AWS 雲端中的 VaR 系統提供範例架構，並說明如何使用 Lambda 函數進行 VaR 計算。若要建立 Lambda 函數，請參閱 Lambda 文件中的程式碼範例。如需協助，請聯絡 AWS Professional Services。

最佳實務

- 讓每個 VaR 運算任務盡可能小且輕量。在每個運算任務中實驗不同數量的交易，以查看哪個交易最適合運算時間和成本。
- 在 Amazon ElastiCache 中存放可重複使用的物件。使用 Apache Arrow 等架構來減少序列化和還原序列化。
- 考慮 Lambda 的時間限制。如果您認為運算任務可能超過 15 分鐘，請嘗試將其分解為較小的任務，以避免 Lambda 逾時。如果無法做到這一點，您可能會考慮使用 AWS Fargate、Amazon Elastic Container Service (Amazon ECS) 和 Amazon Elastic Kubernetes Service (Amazon EKS) 的容器協同運作解決方案。

史詩

交易流程到風險系統

任務	Description	所需的技能
開始撰寫交易。	新的、已結清或部分結清的交易會從訂單管理系統寫入風險串流。此模式使用 Amazon Kinesis 作為受管串流服務。交易訂單代號的雜湊用於跨多個碎片下達交易訂單。	Amazon Kinesis

執行 Lambda 函數以進行訂單處理

任務	Description	所需的技能
使用 Lambda 開始處理風險。	為新訂單執行 AWS Lambda 函數。根據待定交易訂單的數量，Lambda 會自動擴展。每個 Lambda 執行個體都有一或多個訂單，並從 Amazon ElastiCache 擷取每個代號的最新位置。（您可以使用其他金融衍生產品的 CUSIP ID、曲線名稱或索引名稱做為金鑰，以存放並從 ElasticCache.) 在 ElastiCache 中，每個刻度的總位置（數量）和鍵值對 <ticker， net position>，其中 net position 是擴展係數，都會更新一次。	Amazon Kinesis、AWS Lambda、Amazon ElastiCache

將每個標記的訊息寫入佇列

任務	Description	所需的技能
將合併的訊息寫入風險佇列。	將訊息寫入佇列。此模式使用 Amazon SQS 做為受管佇列服務。單一 Lambda 執行個體可能在任何指定時間取得一批迷你交易訂單，但只會將每個代號的單一訊息寫入 Amazon SQS。系統會計算擴展係數：(舊的淨位置 + 目前位置) / 舊的淨位置。	Amazon SQS、AWS Lambda

叫用風險引擎

任務	Description	所需的技能
開始風險計算。	叫用風險引擎 Lambda 的 Lambda 函數。每個位置都由單一 Lambda 函數處理。不過，為了最佳化目的，每個 Lambda 函數都可以處理來自 Amazon SQS 的多個訊息。	Amazon SQS、AWS Lambda

從快取擷取風險結果

任務	Description	所需的技能
擷取和更新風險快取。	Lambda 會從 ElastiCache 擷取每個刻度的目前網路位置。它也會從 ElastiCache 擷取每個刻度的 VaR 損益 (PnL) 陣列。	Amazon SQS、AWS Lambda、Amazon ElastiCache

任務	Description	所需的技能
	如果 PnL 陣列已存在，Lambda 函數會使用 比例更新陣列和 VaR，該比例來自網路 Lambda 函數寫入的 Amazon SQS 訊息。如果 PnL 陣列不在 ElasticCache 中，則會使用模擬股票代號價格序列資料來計算新的 PnL 和 VaR。	

更新 Elastic Cache 中的資料並存放在資料庫中

任務	Description	所需的技能
存放風險結果。	在 ElastiCache 中更新 VaR 和 PnL 號碼後，每五分鐘會叫用新的 Lambda 函數。此函數會從 ElastiCache 讀取所有儲存的資料，並將其存放在 Aurora MySQL 相容資料庫和 S3 儲存貯體中。	AWS Lambda、Amazon ElastiCache

相關資源

- Basel VaR 架構

使用 Amazon Athena 設定共用 AWS Glue Data Catalog 的跨帳戶存取權

Denis Avdonin，Amazon Web Services

總結

此模式提供step-by-step指示，包括 AWS Identity and Access Management (IAM) 政策範例，以使用 AWS Glue Data Catalog 設定存放在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中之資料集的跨帳戶共用。您可以將資料集存放在 S3 儲存貯體中。中繼資料是由 AWS Glue 爬蟲程式收集，並放入 AWS Glue Data Catalog。S3 儲存貯體和 AWS Glue Data Catalog 位於稱為資料帳戶的 AWS 帳戶中。您可以為另一個稱為消費者帳戶的 AWS 帳戶中的 IAM 主體提供存取權。使用者可以使用 Amazon Athena 無伺服器查詢引擎來查詢消費者帳戶中的資料。

先決條件和限制

先決條件

- 兩個作用中的 AWS 帳戶
- 其中一個 AWS 帳戶中的 S3 儲存貯體
- Athena 引擎版本 2
- 安裝https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html並設定 AWS Command Line Interface (AWS CLI)（或執行 AWS CLI 命令的 AWS CloudShell)

產品版本

此模式僅適用於 Athena 引擎版本 2 和 Athena 引擎版本 3。我們建議您升級至 Athena 引擎版本 3。如果您無法從 Athena 引擎版本 1 升級到 Athena 引擎版本 3，請遵循 AWS 大數據部落格中跨帳戶 AWS Glue Data Catalog 存取 Amazon Athena 中討論的方法。

Architecture

目標技術堆疊

- Amazon Athena
- Amazon Simple Storage Service (Amazon S3)
- AWS Glue

- AWS Identity and Access Management (IAM)
- AWS Key Management Service (AWS KMS)

下圖顯示透過 AWS Glue Data Catalog，使用 IAM 許可將某個 AWS 帳戶 （資料帳戶） 中 S3 儲存貯體中的資料與另一個 AWS 帳戶 （消費者帳戶） 共用的架構。 AWS Glue

該圖顯示以下工作流程：

1. 資料帳戶中的 S3 儲存貯體政策會將許可授予取用者帳戶中的 IAM 角色，以及授予資料帳戶中的 AWS Glue 爬蟲程式服務角色。
2. 資料帳戶中的 AWS KMS 金鑰政策會將許可授予取用者帳戶中的 IAM 角色，以及授予資料帳戶中的 AWS Glue 爬蟲程式服務角色。
3. 資料帳戶中的 AWS Glue 爬蟲程式會探索存放在 S3 儲存貯體中的資料結構描述。
4. 資料帳戶中 AWS Glue Data Catalog 的資源政策會授予取用者帳戶中 IAM 角色的存取權。
5. 使用者使用 AWS CLI 命令，在消費者帳戶中建立具名目錄參考。
6. IAM 政策會授予取用者帳戶中的 IAM 角色對資料帳戶中資源的存取權。IAM 角色的信任政策允許取用者帳戶中的使用者擔任 IAM 角色。
7. 取用者帳戶中的使用者會擔任 IAM 角色，並使用 SQL 查詢存取資料目錄中的物件。
8. Athena 無伺服器引擎會執行 SQL 查詢。

> **Note**
>
> IAM 最佳實務建議您將許可授予 IAM 角色，並使用聯合身分。

工具

- Amazon Athena 是一種互動式查詢服務，可協助您使用標準 SQL 直接在 Amazon S3 中分析資料。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Glue 是全受管的擷取、轉換和載入 (ETL) 服務。它可協助您可靠地分類、清理、擴充和移動資料存放區和資料串流之間的資料。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。

- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以保護資料。

史詩

在資料帳戶中設定許可

任務	Description	所需的技能
授予對 S3 儲存貯體中資料的存取權。	根據下列範本建立 anS3bucket 政策，並將政策指派給儲存資料的儲存貯體。 `{ "Version": "2012-10-17", "Statement": [{ "Effect": "Allow", "Principal": { "AWS": ["arn:aws:iam::<consumer account id>:role/<role name>", "arn:aws:iam::<data account id>:role/service-role/AWSGlueServiceRole-data-bucket-crawler"] }, "Action": "s3:GetObject", "Resource": "arn:aws:s3:::data-bucket/*" }, {`	雲端管理員

任務	Description	所需的技能
	"Effect": "Allow", "Principal": { "AWS": ["arn:aws:iam::<consumer account id>:role/<role name>", "arn:aws:iam::<data account id>:role/service-role/AWSGlueServiceRole-data-bucket-crawler"] }, "Action": "s3:ListBucket", "Resource": "arn:aws:s3:::data-bucket" }] } 儲存貯體政策會授予許可給取用者帳戶中的 IAM 角色，以及資料帳戶中的 AWS Glue 爬蟲程式服務角色。	

任務	Description	所需的技能
（如果需要）授予資料加密金鑰的存取權。	如果 S3 儲存貯體是由 AWS KMS 金鑰加密，請將金鑰的`kms:Decrypt` 許可授予取用者帳戶中的 IAM 角色，以及授予資料帳戶中的 AWS Glue 爬蟲程式服務角色。 使用下列陳述式更新金鑰政策： `{ "Effect": "Allow", "Principal": { "AWS": [ "arn:aws:iam::<consumer account id>:role/<role name>", "arn:aws:iam::<data account id>:role/service-role/AWSGlueServiceRole-data-bucket-crawler" ] }, "Action": "kms:Decrypt", "Resource": "arn:aws:kms:<region>:<data account id>:key/<key id>" }`	雲端管理員

任務	Description	所需的技能
授予爬蟲程式對資料的存取權。	將下列 IAM 政策連接至爬蟲程式的服務角色： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "s3:GetObject", "Resource": "arn:aws:s3:::data-bucket/*" }, { "Effect": "Allow", "Action": "s3:ListBucket", "Resource": "arn:aws:s3:::data-bucket" } ] }`	雲端管理員

任務	Description	所需的技能

（如果需要）授予爬蟲程式對資料加密金鑰的存取權。	如果 S3 儲存貯體是由 AWS KMS 金鑰加密，請將下列政策連接至爬蟲程式的服務角色，以授予金鑰的kms:Decrypt 許可：
	
	`{`
	`  "Effect": "Allow",`
	`  "Action": "kms:Decrypt",`
	`  "Resource": "arn:aws:kms:<region>:<data account id>:key/<key id>"`
	`}`
		雲端管理員

任務	Description	所需的技能
授予取用者帳戶中的 IAM 角色和爬蟲程式對資料目錄的存取權。	1. 登入 AWS 管理主控台並開啟 AWS Glue 主控台。 2. 在導覽窗格中的資料目錄下，選擇設定。 3. 在許可區段中，新增下列陳述式，然後選擇儲存。 `{ "Version" : "2012-10-17", "Statement" : [{ "Effect" : "Allow", "Principal" : { "AWS" : ["arn:aws:iam::<consumer account id>:role/<role name>", "arn:aws:iam::<data account id>:role/service-role/AWSGlueServiceRole-data-bucket-crawler"] }, "Action" : "glue:*", "Resource" : ["arn:aws:glue:<region>:<data account id>:catalog",`	雲端管理員

任務	Description	所需的技能

	<pre>"arn:aws:glue:<region>:<data account id>:database/*",
	
	"arn:aws:glue:<region>:<data account id>:table/*"
	]
	}
	]
	}</pre>
	此政策允許資料帳戶中所有資料庫和資料表上的所有 AWS Glue 動作。您可以自訂政策，只將必要的許可授予消費者委託人。例如，您可以提供資料庫中特定資料表或檢視的唯讀存取權。	

從消費者帳戶存取資料

任務	Description	所需的技能
為資料目錄建立具名參考。	若要建立具名資料目錄參考，請使用 CloudShell 或本機安裝的 AWS CLI 來執行下列命令： <pre>aws athena create-data-catalog --name <shared catalog name> --type GLUE --parameters catalog-id=<data account id></pre>	雲端管理員

任務	Description	所需的技能
授予取用者帳戶中的 IAM 角色對資料的存取權。	將下列政策連接至取用者帳戶中的 IAM 角色，以授予角色跨帳戶的資料存取權： { "Version": "2012-10-17", "Statement": [{ "Effect": "Allow", "Action": "s3:GetObject", "Resource": "arn:aws:s3:::data-bucket/*" }, { "Effect": "Allow", "Action": "s3:ListBucket", "Resource": "arn:aws:s3:::data-bucket" }, { "Effect": "Allow", "Action": "glue:*", "Resource": ["arn:aws:glue:<region>:<data account id>:catalog", "arn:aws:glue:<region>:<data account id>:database/*",	雲端管理員

任務	Description	所需的技能

	<pre>"arn:aws:glue:<region>:<data account id>:table/*"
	            ]
	        }
	    ]
	}</pre>
	接著，使用下列範本指定哪些使用者可以在其信任政策中接受 IAM 角色：
	<pre>{
	    "Version": "2012-10-17",
	    "Statement": [
	       {
	            "Effect": "Allow",
	            "Principal": {
	                "AWS": "arn:aws:iam::<consumer account id>:user/<IAM user>"
	            },
	            "Action": "sts:AssumeRole"
	        }
	    ]
	}</pre>
	最後，將相同的政策連接至使用者所屬的使用者群組，授予使用者擔任 IAM 角色的許可。	

任務	Description	所需的技能
（如果需要）授予取用者帳戶中的 IAM 角色對資料加密金鑰的存取權。	如果 S3 儲存貯體是由 AWS KMS 金鑰加密，請將下列政策連接至取用者帳戶中的 IAM 角色，以授予金鑰的`kms:Decrypt` 許可： `{ "Effect": "Allow", "Action": "kms:Decrypt", "Resource": "arn:aws:kms:<region>:<data account id>:key/<key id>" }`	雲端管理員
切換到取用者帳戶中的 IAM 角色以存取資料。	身為資料取用者，請切換到 IAM 角色以存取資料帳戶中的資料。	資料取用者

任務	Description	所需的技能

存取資料。	使用 Athena 查詢資料。例如，開啟 Athena 查詢編輯器並執行下列查詢：
	
	`SELECT * FROM <shared catalog name>.<database name>.<table name>`
	
	您也可以依目錄的 Amazon Resource Name (ARN) 來參考目錄，而不是使用具名目錄參考。
	
	Note
	如果您在查詢或檢視中使用動態目錄參考，請使用逸出雙引號 (\") 括住參考。例如：
	
	`SELECT * FROM \"glue:arn:aws:glue:<region>:<data account id>:catalog\".<database name>.<table name>`
	
	如需詳細資訊，請參閱《Amazon Athena 使用者指南》中的 AWS Glue 資料目錄的跨帳戶存取權。
		資料取用者

相關資源

- 跨帳戶存取 AWS Glue 資料目錄 (Athena 文件）
- (AWS CLI) create-data-catalog (AWS CLI 命令參考）
- 使用 Amazon Athena 進行跨帳戶 AWS Glue Data Catalog 存取 (AWS 大數據部落格）
- IAM 中的安全最佳實務 (IAM 文件）

其他資訊

使用 Lake Formation 做為跨帳戶共用的替代方案

您也可以使用 AWS Lake Formation 跨帳戶共用對 AWS Glue 目錄物件的存取權。Lake Formation 在資料欄和資料列層級提供精細的存取控制、標籤型存取控制、ACID 交易的受管資料表，以及其他功能。雖然 Lake Formation 與 Athena 充分整合，但相較於此模式的僅限 IAM 方法，它確實需要額外的組態。我們建議您考慮在整體解決方案架構的更廣泛內容中使用 Lake Formation 或僅限 IAM 存取控制的決定。考量包括涉及哪些其他服務，以及它們如何與這兩種方法整合。

將 Teradata NORMALIZE 暫時功能轉換為 Amazon Redshift SQL

Po Hong，Amazon Web Services

總結

NORMALIZE 是 ANSI SQL 標準的 Teradata 擴充功能。當 SQL 資料表包含具有 PERIOD 資料類型的資料欄時，NORMALIZE 會結合在該資料欄中符合或重疊的值，以形成單一期間來合併多個個別期間值。若要使用 NORMALIZE，SQL SELECT 清單中至少有一個資料欄必須是 Teradata 的暫時 PERIOD 資料類型。如需 NORMALIZE 的詳細資訊，請參閱 Teradata 文件。

Amazon Redshift 不支援 NORMALIZE，但您可以在 Amazon Redshift 中使用原生 SQL 語法和 LAG 視窗函數來實作此功能。此模式著重於搭配 ON MEETS 或 OVERLAPS 條件使用 Teradata NORMALIZE 延伸模組，這是最受歡迎的格式。它說明此功能如何在 Teradata 中運作，以及如何將其轉換為 Amazon Redshift 原生 SQL 語法。

先決條件和限制

先決條件

- 基本 Teradata SQL 知識和經驗
- Amazon Redshift 知識和經驗

Architecture

來源技術堆疊

- Teradata 資料倉儲

目標技術堆疊

- Amazon Redshift

目標架構

如需將 Teradata 資料庫遷移至 Amazon Redshift 的高階架構，請參閱使用 AWS SCT 資料擷取代理程式將 Teradata 資料庫遷移至 Amazon Redshift 的模式。遷移不會自動將 Teradata NORMALIZE 片語轉換為 Amazon Redshift SQL。您可以遵循此模式中的準則來轉換此 Teradata 延伸。

工具

Code

若要說明 NORMALIZE 的概念和功能，請考慮 Teradata 中的下列資料表定義：

```
CREATE TABLE systest.project
     (    emp_id        INTEGER,
          project_name  VARCHAR(20),
          dept_id       INTEGER,
          duration      PERIOD(DATE)
     );
```

執行下列 SQL 程式碼，將範例資料插入資料表：

```
BEGIN TRANSACTION;

INSERT INTO systest.project VALUES (10, 'First Phase', 1000,  PERIOD(DATE '2010-01-10',
 DATE '2010-03-20') );
INSERT INTO systest.project VALUES (10, 'First Phase', 2000,  PERIOD(DATE '2010-03-20',
 DATE '2010-07-15') );

INSERT INTO systest.project VALUES (10, 'Second Phase', 2000,  PERIOD(DATE
 '2010-06-15', DATE '2010-08-18') );
INSERT INTO systest.project VALUES (20, 'First Phase', 2000,  PERIOD(DATE '2010-03-10',
 DATE '2010-07-20') );

INSERT INTO systest.project VALUES (20, 'Second Phase', 1000,  PERIOD(DATE
 '2020-05-10', DATE '2020-09-20') );

END TRANSACTION;
```

結果：

```
select * from systest.project order by 1,2,3;

 *** Query completed. 4 rows found. 4 columns returned.
 *** Total elapsed time was 1 second.

     emp_id  project_name              dept_id  duration
-----------  --------------------  -----------  ------------------------
         10  First Phase                  1000  ('10/01/10', '10/03/20')
```

```
         10  First Phase                  2000  ('10/03/20', '10/07/15')
         10  Second Phase                 2000  ('10/06/15', '10/08/18')
         20  First Phase                  2000  ('10/03/10', '10/07/20')
         20  Second Phase                 1000  ('20/05/10', '20/09/20')
```

Teradata NORMALIZE 使用案例

現在將 Teradata NORMALIZE SQL 子句新增至 SELECT 陳述式：

```
SELECT NORMALIZE ON MEETS OR OVERLAPS emp_id, duration
FROM systest.project
ORDER BY 1,2;
```

此 NORMALIZE 操作會在單一資料欄 () 上執行emp_id。對於 emp_id=10，持續時間中的三個重疊期間值會合併為單一期間值，如下所示：

```
     emp_id  duration
-----------  ------------------------
         10  ('10/01/10', '10/08/18')
         20  ('10/03/10', '10/07/20')
         20  ('20/05/10', '20/09/20')
```

下列 SELECT 陳述式會在 project_name和 上執行 NORMALIZE 操作dept_id。請注意，SELECT 清單只包含一個 PERIOD 資料欄 duration。

```
SELECT NORMALIZE project_name, dept_id, duration
FROM systest.project;
```

輸出：

```
project_name              dept_id  duration
--------------------  -----------  ------------------------
First Phase                  1000  ('10/01/10', '10/03/20')
Second Phase                 1000  ('20/05/10', '20/09/20')
First Phase                  2000  ('10/03/10', '10/07/20')
Second Phase                 2000  ('10/06/15', '10/08/18')
```

Amazon Redshift 對等 SQL

Amazon Redshift 目前不支援資料表中的 PERIOD 資料類型。相反地，您需要將 Teradata PERIOD 資料欄位分成兩個部分：start_date, end_date，如下所示：

```
CREATE TABLE systest.project
     (    emp_id        INTEGER,
          project_name  VARCHAR(20),
          dept_id       INTEGER,
          start_date  DATE,
          end_date    DATE
     );
```

將範例資料插入資料表：

```
BEGIN TRANSACTION;

INSERT INTO systest.project VALUES (10, 'First Phase', 1000,  DATE '2010-01-10', DATE
 '2010-03-20' );
INSERT INTO systest.project VALUES (10, 'First Phase', 2000,  DATE '2010-03-20', DATE
 '2010-07-15');

INSERT INTO systest.project VALUES (10, 'Second Phase', 2000,  DATE '2010-06-15', DATE
 '2010-08-18' );
INSERT INTO systest.project VALUES (20, 'First Phase', 2000,  DATE '2010-03-10', DATE
 '2010-07-20' );

INSERT INTO systest.project VALUES (20, 'Second Phase', 1000,  DATE '2020-05-10', DATE
 '2020-09-20' );

END TRANSACTION;
```

輸出：

```
 emp_id | project_name | dept_id | start_date |  end_date
--------+--------------+---------+------------+------------
     10 | First Phase  |    1000 | 2010-01-10 | 2010-03-20
     10 | First Phase  |    2000 | 2010-03-20 | 2010-07-15
     10 | Second Phase |    2000 | 2010-06-15 | 2010-08-18
     20 | First Phase  |    2000 | 2010-03-10 | 2010-07-20
     20 | Second Phase |    1000 | 2020-05-10 | 2020-09-20
(5 rows)
```

若要重寫 Teradata 的 NORMALIZE 子句，您可以在 Amazon Redshift 中使用 LAG 視窗函數。此函數會傳回分割區中目前資料列上方（之前）指定位移的資料列值。

您可以使用 LAG 函數來識別開始新期間的每個資料列，方法是判斷期間是否符合或與前一個期間重疊 (0 表示是，1 表示否）。當此旗標累積加總時，會提供群組識別符，可用於外部 Group By 子句，以到達 Amazon Redshift 中所需的結果。

以下是使用 LAG() 的 Amazon Redshift SQL 陳述式範例：

```
SELECT emp_id, start_date, end_date,
            (CASE WHEN start_date <= LAG(end_date) OVER (PARTITION BY emp_id ORDER BY
 start_date, end_date) THEN 0 ELSE 1 END) AS GroupStartFlag
FROM systest.project
ORDER BY 1,2;
```

輸出：

```
 emp_id | start_date |  end_date  | groupstartflag
--------+------------+------------+----------------
     10 | 2010-01-10 | 2010-03-20 |              1
     10 | 2010-03-20 | 2010-07-15 |              0
     10 | 2010-06-15 | 2010-08-18 |              0
     20 | 2010-03-10 | 2010-07-20 |              1
     20 | 2020-05-10 | 2020-09-20 |              1
(5 rows)
```

下列 Amazon Redshift SQL 陳述式只會在 emp_id 欄上標準化：

```
SELECT T2.emp_id, MIN(T2.start_date) as new_start_date, MAX(T2.end_date) as
 new_end_date
FROM
( SELECT T1.*, SUM(GroupStartFlag) OVER (PARTITION BY emp_id ORDER BY start_date ROWS
 UNBOUNDED PRECEDING) As GroupID
FROM ( SELECT emp_id, start_date, end_date,
            (CASE WHEN start_date <= LAG(end_date) OVER (PARTITION BY emp_id ORDER BY
 start_date, end_date) THEN 0 ELSE 1 END) AS GroupStartFlag
FROM systest.project ) T1
) T2
GROUP BY T2.emp_id, T2.GroupID
ORDER BY 1,2;
```

輸出：

```
 emp_id | new_start_date | new_end_date
```

```
--------+----------------+-------------------------------------
     10 | 2010-01-10     | 2010-08-18
     20 | 2010-03-10     | 2010-07-20
     20 | 2020-05-10     | 2020-09-20
(3 rows)
```

下列 Amazon Redshift SQL 陳述式會在 `project_name`和 `dept_id`資料欄上標準化：

```
SELECT T2.project_name, T2.dept_id, MIN(T2.start_date) as new_start_date,
 MAX(T2.end_date) as new_end_date
FROM
( SELECT T1.*, SUM(GroupStartFlag) OVER (PARTITION BY project_name, dept_id ORDER BY
 start_date ROWS UNBOUNDED PRECEDING) As GroupID
FROM ( SELECT project_name, dept_id, start_date, end_date,
            (CASE WHEN start_date <= LAG(end_date) OVER (PARTITION BY project_name,
 dept_id ORDER BY start_date, end_date) THEN 0 ELSE 1 END) AS GroupStartFlag
FROM systest.project ) T1
) T2
GROUP BY T2.project_name, T2.dept_id, T2.GroupID
ORDER BY 1,2,3;
```

輸出：

```
 project_name | dept_id | new_start_date | new_end_date
--------------+---------+----------------+--------------
 First Phase  |    1000 | 2010-01-10     | 2010-03-20
 First Phase  |    2000 | 2010-03-10     | 2010-07-20
 Second Phase |    1000 | 2020-05-10     | 2020-09-20
 Second Phase |    2000 | 2010-06-15     | 2010-08-18
(4 rows)
```

史詩

將 NORMALIZE 轉換為 Amazon Redshift SQL

任務	Description	所需的技能
建立 Teradata SQL 程式碼。	根據您的需求使用 NORMALIZE 片語。	SQL Developer

任務	Description	所需的技能
將程式碼轉換為 Amazon Redshift SQL。	若要轉換您的程式碼，請遵循此模式的「工具」區段中的準則。	SQL Developer
在 Amazon Redshift 中執行程式碼。	建立資料表、將資料載入資料表，並在 Amazon Redshift 中執行程式碼。	SQL Developer

相關資源

參考

- Teradata NORMALIZE 暫時功能 (Teradata 文件）
- LAG 視窗函數 (Amazon Redshift 文件）
- 遷移至 Amazon Redshift (AWS 網站）
- 使用 AWS SCT 資料擷取代理程式將 Teradata 資料庫遷移至 Amazon Redshift (AWS 方案指引）
- 將 Teradata RESET WHEN 功能轉換為 Amazon Redshift SQL (AWS 方案指引）

工具

- AWS Schema Conversion Tool (AWS SCT)

合作夥伴

- AWS 遷移能力合作夥伴

將 Teradata RESET WHEN 功能轉換為 Amazon Redshift SQL

Po Hong，Amazon Web Services

總結

RESET WHEN 是 SQL 分析視窗函數中使用的 Teradata 功能。它是 ANSI SQL 標準的延伸。RESET WHEN 會根據某些指定的條件，決定 SQL 視窗函數運作所在的分割區。如果條件評估為 TRUE，則會在現有視窗分割區內建立新的動態子分割區。如需 RESET WHEN 的詳細資訊，請參閱 Teradata 文件。

Amazon Redshift 在 SQL 視窗函數中不支援 RESET WHEN。若要實作此功能，您必須將 RESET WHEN 轉換為 Amazon Redshift 中的原生 SQL 語法，並使用多個巢狀函數。此模式示範如何使用 Teradata RESET WHEN 功能，以及如何將其轉換為 Amazon Redshift SQL 語法。

先決條件和限制

先決條件

- Teradata 資料倉儲及其 SQL 語法的基本知識
- 充分了解 Amazon Redshift 及其 SQL 語法

Architecture

來源技術堆疊

- Teradata 資料倉儲

目標技術堆疊

- Amazon Redshift

架構

如需將 Teradata 資料庫遷移至 Amazon Redshift 的高階架構，請參閱使用 AWS SCT 資料擷取代理程式將 Teradata 資料庫遷移至 Amazon Redshift 的模式。遷移不會自動將 Teradata RESET WHEN 片語轉換為 Amazon Redshift SQL。您可以遵循下一節中的準則來轉換此 Teradata 擴充功能。

工具

Code

若要說明 RESET WHEN 的概念，請考慮 Teradata 中的下列資料表定義：

```
create table systest.f_account_balance
( account_id integer NOT NULL,
  month_id integer,
  balance integer )
unique primary index (account_id, month_id);
```

執行下列 SQL 程式碼，將範例資料插入資料表：

```
BEGIN TRANSACTION;
Insert Into systest.f_account_balance values (1,1,60);
Insert Into systest.f_account_balance values (1,2,99);
Insert Into systest.f_account_balance values (1,3,94);
Insert Into systest.f_account_balance values (1,4,90);
Insert Into systest.f_account_balance values (1,5,80);
Insert Into systest.f_account_balance values (1,6,88);
Insert Into systest.f_account_balance values (1,7,90);
Insert Into systest.f_account_balance values (1,8,92);
Insert Into systest.f_account_balance values (1,9,10);
Insert Into systest.f_account_balance values (1,10,60);
Insert Into systest.f_account_balance values (1,11,80);
Insert Into systest.f_account_balance values (1,12,10);
END TRANSACTION;
```

範例資料表具有下列資料：

account_id	month_id	balance
1	1	60
1	2	99
1	3	94
1	4	90
1	5	80

1	6	88
1	7	90
1	8	92
1	9	10
1	10	60
1	11	80
1	12	10

對於每個帳戶，假設您想要分析連續的每月餘額增加序列。當一個月的餘額小於或等於上個月的餘額時，需要將計數器重設為零並重新啟動。

Teradata RESET 當使用案例

為了分析此資料，Teradata SQL 使用具有巢狀彙總和 RESET WHEN 片語的視窗函數，如下所示：

```
SELECT account_id, month_id, balance,
 ( ROW_NUMBER() OVER (PARTITION BY account_id ORDER BY month_id
RESET WHEN balance <= SUM(balance) over (PARTITION BY account_id ORDER BY month_id ROWS
 BETWEEN 1 PRECEDING AND 1 PRECEDING) ) -1 ) as balance_increase
FROM systest.f_account_balance
ORDER BY 1,2;
```

輸出：

account_id	month_id	balance	balance_increase
1	1	60	0
1	2	99	1
1	3	94	0
1	4	90	0
1	5	80	0

1	6	88	1
1	7	90	2
1	8	92	3
1	9	10	0
1	10	60	1
1	11	80	2
1	12	10	0

查詢在 Teradata 中的處理方式如下：

1. SUM（平衡）彙總函數會計算指定月份中指定帳戶的所有餘額總和。
2. 我們會檢查指定月份的餘額（針對指定帳戶）是否大於上個月的餘額。
3. 如果餘額增加，我們會追蹤累積計數值。如果 RESET WHEN 條件評估為 false，表示餘額已連續幾個月增加，我們會繼續增加計數。
4. ROW_NUMBER() 排序的分析函數會計算計數值。當我們達到餘額小於或等於上個月餘額的月份時，RESET WHEN 條件會評估為 true。若是如此，我們會啟動新的分割區，ROW_NUMBER() 會從 1 重新啟動計數。我們使用介於 1 PRECEDING 和 1 PRECEDING 之間的 ROWS 來存取上一列的值。
5. 我們減去 1，以確保計數值以 0 開頭。

Amazon Redshift 對等 SQL

Amazon Redshift 不支援 SQL 分析視窗函數中的 RESET WHEN 片語。 若要產生相同的結果，您必須使用 Amazon Redshift 原生 SQL 語法和巢狀子查詢重寫 Teradata SQL，如下所示：

```
SELECT account_id, month_id, balance,
   (ROW_NUMBER() OVER(PARTITION BY account_id, new_dynamic_part ORDER BY month_id) -1)
 as balance_increase
FROM
( SELECT account_id, month_id, balance, prev_balance,
SUM(dynamic_part) OVER (PARTITION BY account_id ORDER BY month_id ROWS BETWEEN
 UNBOUNDED PRECEDING AND CURRENT ROW) As new_dynamic_part
FROM ( SELECT account_id, month_id, balance,
```

```
SUM(balance) over (PARTITION BY account_id ORDER BY month_id ROWS BETWEEN 1 PRECEDING
 AND 1 PRECEDING) as prev_balance,
(CASE When balance <= prev_balance Then 1 Else 0 END) as dynamic_part
FROM systest.f_account_balance ) A
) B
ORDER BY 1,2;
```

由於 Amazon Redshift 在單一 SQL 陳述式的 SELECT 子句中不支援巢狀視窗函數，您必須使用兩個巢狀子查詢。

- 在內部子查詢（別名 A) 中，會建立並填入動態分割區指標 (dynamic_part)。如果一個月的餘額小於或等於上個月的餘額，則會將 dynamic_part 設定為 1；否則，會將 設定為 0。
- 在下一層（別名 B) 中，會產生新的_dynamic_part 屬性作為 SUM 視窗函數的結果。
- 最後，將 new_dynamic_part 作為新的分割區屬性 (動態分割區) 新增至現有的分割區屬性 (account_id)，並套用與 Teradata（和減一）相同的 ROW_NUMBER() 視窗函數。

在這些變更之後，Amazon Redshift SQL 會產生與 Teradata 相同的輸出。

史詩

將 RESET WHEN 轉換為 Amazon Redshift SQL

任務	Description	所需的技能
建立 Teradata 視窗函數。	根據您的需求使用巢狀彙總和 RESET WHEN 片語。	SQL Developer
將程式碼轉換為 Amazon Redshift SQL。	若要轉換您的程式碼，請遵循此模式的「工具」區段中的準則。	SQL Developer
在 Amazon Redshift 中執行程式碼。	建立資料表、將資料載入資料表，以及在 Amazon Redshift 中執行程式碼。	SQL Developer

相關資源

參考

- 片語時重設 (Teradata 文件）
- RESET WHEN 說明 （堆疊溢位）
- 遷移至 Amazon Redshift (AWS 網站）
- 使用 AWS SCT 資料擷取代理程式將 Teradata 資料庫遷移至 Amazon Redshift (AWS 方案指引）
- 將 Teradata NORMALIZE 暫時功能轉換為 Amazon Redshift SQL (AWS 方案指引）

工具

- AWS Schema Conversion Tool (AWS SCT)

合作夥伴

- AWS 遷移能力合作夥伴

使用基礎設施做為程式碼，在 AWS 雲端上部署和管理無伺服器資料湖

Kirankumar Chandrashekar 和 Abdel Jaidi，Amazon Web Services

總結

此模式說明如何使用無伺服器運算和基礎設施做為程式碼 (IaC)，在 Amazon Web Services (AWS) 雲端上實作和管理資料湖。此模式是以 AWS 開發的無伺服器資料湖架構 (SDLF) 研討會為基礎。

SDLF 是可重複使用的資源集合，可加速在 AWS 雲端上交付企業資料湖，並有助於更快速地部署至生產環境。它用於遵循最佳實務來實作資料湖的基礎結構。

SDLF 使用 AWS CodePipeline、AWS CodeBuild 和 AWS CodeCommit 等 AWS 服務，在整個程式碼和基礎設施部署中實作持續整合/持續部署 (CI/CD) 程序。

此模式使用多個 AWS 無伺服器服務來簡化資料湖管理。其中包括用於儲存的 Amazon Simple Storage Service (Amazon S3) 和 Amazon DynamoDB、用於運算的 AWS Lambda 和 AWS Glue，以及用於協同運作的 Amazon CloudWatch Events、Amazon Simple Queue Service (Amazon SQS) 和 AWS Step Functions。

AWS CloudFormation 和 AWS 程式碼服務可充當 IaC 層，以輕鬆操作和管理的方式提供可重複且快速的部署。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- AWS Command Line Interface (AWS CLI)，已安裝並設定。
- 安裝和設定的 Git 用戶端。
- SDLF 研討會，在 Web 瀏覽器視窗中開啟並準備使用。

Architecture

架構圖說明事件驅動的程序，步驟如下。

1. 將檔案新增至原始資料 S3 儲存貯體後，Amazon S3 事件通知會放置在 SQS 佇列中。每個通知會以 JSON 檔案形式傳送，其中包含中繼資料，例如 S3 儲存貯體名稱、物件金鑰或時間戳記。
2. Lambda 函數會使用此通知，此函數會根據中繼資料將事件路由至正確的擷取、轉換和載入 (ETL) 程序。Lambda 函數也可以使用存放在 Amazon DynamoDB 資料表中的內容式組態。此步驟可對資料湖中的多個應用程式進行解耦和擴展。
3. 事件會路由至 ETL 程序中的第一個 Lambda 函數，該函數會將資料從原始資料區域轉換和移動到資料湖的暫存區域。第一步是更新全面型錄。這是包含資料湖所有檔案中繼資料的 DynamoDB 資料表。此資料表中的每一列都會保留儲存在 Amazon S3 中單一物件的操作中繼資料。在 S3 物件上，同步呼叫執行輕度轉換的 Lambda 函數，這是運算上便宜的操作（例如將檔案從一種格式轉換為另一種格式）。由於已將新物件新增至預備 S3 儲存貯體，因此會更新完整目錄，並傳送訊息至等待 ETL 中下一個階段的 SQS 佇列。
4. CloudWatch Events 規則每 5 分鐘觸發一次 Lambda 函數。此函數會檢查訊息是否已從上一個 ETL 階段傳遞至 SQS 佇列。如果訊息已傳遞，Lambda 函數會從 ETL 程序的 AWS Step Functions 開始第二個函數。
5. 然後，大量轉換會套用至一批檔案。這種繁重的轉換是一種運算昂貴的操作，例如同步呼叫 AWS Glue 任務、AWS Fargate 任務、Amazon EMR 步驟或 Amazon SageMaker 筆記本。資料表中繼資料會使用更新 AWS Glue 目錄的 AWS Glue 爬蟲程式，從輸出檔案擷取。檔案中繼資料也會新增至 DynamoDB 中完整的目錄資料表。最後，也會執行利用 Deequ 的資料品質步驟。

技術堆疊

- Amazon CloudWatch Events
- AWS CloudFormation
- AWS CodePipeline
- AWS CodeBuild
- AWS CodeCommit
- Amazon DynamoDB
- AWS Glue
- AWS Lambda
- Amazon S3
- Amazon SQS
- AWS Step Functions

工具

- Amazon CloudWatch Events – CloudWatch Events 提供近乎即時的系統事件串流，說明 AWS 資源的變更。
- AWS CloudFormation – CloudFormation 有助於以可預測且重複的方式建立和佈建 AWS 基礎設施部署。
- AWS CodeBuild – CodeBuild 是一種全受管的建置服務，可編譯您的原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeCommit – CodeCommit 是由 AWS 託管的版本控制服務，可用來私下存放和管理資產（例如原始碼和二進位檔案）。
- AWS CodePipeline – CodePipeline 是一種持續交付服務，可用來建立模型、視覺化和自動化持續發佈軟體變更所需的步驟。
- Amazon DynamoDB – DynamoDB 是全受管的 NoSQL 資料庫服務，可提供快速且可預測的效能與可擴展性。
- AWS Glue – AWS Glue 是一種全受管 ETL 服務，可讓您更輕鬆地準備和載入資料以供分析。
- AWS Lambda – Lambda 支援執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種高度可擴展的物件儲存服務。Amazon S3 可用於各種儲存解決方案，包括網站、行動應用程式、備份和資料湖。
- AWS Step Functions - AWS Step Functions 是一種無伺服器函數協調器，可讓您輕鬆地將 AWS Lambda 函數和多個 AWS 服務排序為業務關鍵應用程式。
- Amazon SQS – Amazon Simple Queue Service (Amazon SQS) 是一種全受管訊息佇列服務，可協助您解耦和擴展微服務、分散式系統和無伺服器應用程式。
- Deequ – Deequ 是一種工具，可協助您運算大型資料集的資料品質指標、定義和驗證資料品質限制，並隨時掌握資料分佈的變更。

程式碼儲存庫

SDLF 的原始碼和資源可在 AWS Labs GitHub 儲存庫中使用。

史詩

設定 CI/CD 管道來佈建 IaC

任務	Description	所需的技能
設定 CI/CD 管道來管理資料湖的 IaC。	登入 AWS 管理主控台，並遵循 SDLF 研討會初始設定一節中的步驟。這會建立初始 CI/CD 資源，例如 CodeCommit 儲存庫、CodeBuild 環境，以及為資料湖佈建和管理 IaC 的 CodePipeline 管道。	DevOps 工程師

版本控制 IaC

任務	Description	所需的技能
在本機電腦上複製 CodeCommit 儲存庫。	請遵循 SDLF 研討會部署基礎一節中的步驟。這可協助您將託管 IaC 的 Git 儲存庫複製到本機環境。 如需詳細資訊，請參閱從 CodeCommit 文件連線至 CodeCommit 儲存庫。CodeCommit	DevOps 工程師
修改 CloudFormation 範本。	使用您的本機工作站和程式碼編輯器，根據您的使用案例或需求修改 CloudFormation 範本。將它們遞交至本機複製的 Git 儲存庫。 如需詳細資訊，請參閱 AWS CloudFormation 文件中的使用 AWS CloudFormation 範本。	DevOps 工程師

任務	Description	所需的技能
將變更推送至 CodeCommit 儲存庫。	您的基礎設施程式碼現在受到版本控制，並且會追蹤對程式碼基礎的修改。當您將變更推送至 CodeCommit 儲存庫時，CodePipeline 會自動將其套用至您的基礎設施，並將其交付至 CodeBuild。 **Important** 如果您在 CodeBuild 中使用 AWS SAM CLI，請執行 `sam package`和 `sam deploy`命令。如果您使用 AWS CLI，請執行 `aws cloudformation package` 和 `aws cloudformation deploy` 命令。	DevOps 工程師

相關資源

設定 CI/CD 管道以佈建 IaC

- SDLF 研討會 – 初始設定

版本控制 IaC

- SDLF 研討會 – 部署基礎
- 連線至 CodeCommit 儲存庫
- 使用 AWS CloudFormation 範本

其他資源

- AWS 無伺服器資料分析管道參考架構
- SDLF 文件

在啟動時強制標記 Amazon EMR 叢集

Priyanka Chaudhary，Amazon Web Services

總結

此模式提供安全控制，可確保在建立 Amazon EMR 叢集時對其進行標記。

Amazon EMR 是一種 Amazon Web Services (AWS) 服務，用於處理和分析大量資料。Amazon EMR 提供可擴展的低組態服務，是執行內部叢集運算的更簡單替代方案。您可以使用標記以不同方式分類 AWS 資源，例如依用途、擁有者或環境 。例如，您可以透過將自訂中繼資料指派給每個叢集來標記 Amazon EMR 叢集。標籤包含您定義的索引鍵和值。我們建議您建立一組一致的標籤，以符合組織的需求。當您將標籤新增至 Amazon EMR 叢集時，該標籤也會傳播到與叢集相關聯的每個作用中 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。同樣地，當您從 Amazon EMR 叢集移除標籤時，也會從每個相關聯的作用中 EC2 執行個體中移除該標籤。

偵測性控制項會監控 API 呼叫，並啟動 RunJobFlow、AddTags、RemoveTags 和 CreateTags APIs Amazon CloudWatch Events 事件。事件會呼叫執行 Python 指令碼的 AWS Lambda。Python 函數會從事件的 JSON 輸入取得 Amazon EMR 叢集 ID，並執行下列檢查：

- 檢查 Amazon EMR 叢集是否已使用您指定的標籤名稱設定。
- 如果沒有，請傳送 Amazon Simple Notification Service (Amazon SNS) 通知給使用者，並提供相關資訊：Amazon EMR 叢集名稱、違規詳細資訊、AWS 區域、AWS 帳戶和 Lambda 的 Amazon Resource Name (ARN)，此通知來源為 。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 用來上傳所提供 Lambda 程式碼的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。或者，您可以為此目的建立 S3 儲存貯體，如 Epics 章節所述。
- 您想要接收違規通知的作用中電子郵件地址。
- 您要檢查的必要標籤清單。

限制

- 此安全控制是區域性的。您必須將其部署到您要監控的每個 AWS 區域。

產品版本

- Amazon EMR 4.8.0 版及更新版本。

Architecture

工作流程架構

自動化和擴展

- 如果您使用的是 AWS Organizations，則可以使用 AWS Cloudformation StackSetsto，將此範本部署到您要監控的多個帳戶中。

工具

AWS 服務

- AWS CloudFormation –AWS CloudFormation 可協助您建立模型並設定 AWS 資源、快速一致地佈建資源，以及在整個生命週期中管理資源。您可以使用範本來描述資源及其相依性，並將它們一起啟動和設定為堆疊，而不是個別管理資源。您可以管理和佈建跨多個 AWS 帳戶和 AWS 區域的堆疊。
- Amazon CloudWatch Events - Amazon CloudWatch Events 提供近乎即時的系統事件串流，說明 AWS 資源的變更。
- Amazon EMR - Amazon EMR 是 Web 服務，可簡化大數據架構的執行，並有效率地處理大量資料。
- AWS Lambda – AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種物件儲存服務。您可以使用 Amazon S3 隨時從 Web 任何地方存放和擷取任意資料量。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 會協調和管理發佈者和用戶端之間的訊息傳遞或傳送，包括 Web 伺服器和電子郵件地址。訂閱者會收到發佈到所訂閱主題的所有訊息，且某一主題的所有訂閱者均會收到相同訊息。

Code

此模式包含下列附件：

- `EMRTagValidation.zip` – 用於安全控制的 Lambda 程式碼。
- `EMRTagValidation.yml` – 設定事件和 Lambda 函數的 CloudFormation 範本。

史詩

設定 S3 儲存貯體

任務	Description	所需的技能
定義 S3 儲存貯體。	在 Amazon S3 主控台上，選擇或建立 S3 儲存貯體以託管 Lambda 程式碼 .zip 檔案。此 S3 儲存貯體必須與您要監控的 Amazon EMR 叢集位於相同的 AWS 區域。Amazon S3 儲存貯體的名稱必須是全域唯一，且命名空間會由所有 AWS 帳戶共享。S3 儲存貯體名稱不能包含正斜線。	雲端架構師
上傳 Lambda 程式碼。	將附件區段中提供的 Lambda 程式碼 .zip 檔案上傳至 S3 儲存貯體。	雲端架構師

部署 AWS CloudFormation 範本

任務	Description	所需的技能
啟動 AWS CloudFormation 範本。	在與 S3 儲存貯體相同的 AWS 區域中開啟 AWS CloudFormation 主控台，並部署範本。如需部署 AWS CloudFormation 範本的詳細資訊，請參閱	雲端架構師

任務	Description	所需的技能
	CloudFormation 文件中的在 AWS CloudFormation 主控台上建立堆疊。 CloudFormation	

任務	Description	所需的技能
完成範本中的參數。	當您啟動範本時，系統會提示您輸入下列資訊： • S3 儲存貯體：指定您在第一個特徵中建立或選取的儲存貯體。這是您上傳連接的 Lambda 程式碼 (.zip 檔案）的位置。 • S3 金鑰：指定 S3 儲存貯體中 Lambda .zip 檔案的位置 S3（例如 filename.zip 或 control/filename.zip)。請勿包含正斜線。 • 通知電子郵件：提供您要接收 Amazon SNS 通知的作用中電子郵件地址。 • 標記金鑰名稱：在逗號分隔清單中提供您要檢查的標籤 `ApplicationID` （例如，、`Environment` 、`Owner`)。CloudWatch Events 事件會監控叢集是否有這些標籤，並在找不到它們時傳送通知。 • Lambda 記錄層級：指定 Lambda 函數的記錄層級和頻率。使用資訊記錄有關進度的詳細資訊訊息、仍允許繼續部署的錯誤事件，以及潛在有害情況的警告。	雲端架構師

確認訂閱

任務	Description	所需的技能
確認訂閱。	當 CloudFormation 範本成功部署時，它會傳送訂閱電子郵件到您提供的電子郵件地址。您必須確認此電子郵件訂閱，才能開始接收違規通知。	雲端架構師

相關資源

- AWS Lambda 開發人員指南
- 在 Amazon EMR 中標記叢集

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

確保在啟動時啟用對 Amazon S3 的 Amazon EMR 記錄

Priyanka Chaudhary，Amazon Web Services

總結

此模式提供安全控制，可監控在 Amazon Web Services (AWS) 上執行之 Amazon EMR 叢集的記錄組態。

Amazon EMR 是一種用於大數據處理和分析的 AWS 工具。Amazon EMR 提供可擴展的低組態服務，作為執行內部叢集運算的替代方案。Amazon EMR 提供兩種類型的 EMR 叢集。

- 暫時性 Amazon EMR 叢集：暫時性 Amazon EMR 叢集會在處理完成時自動關閉並停止產生成本。
- 持久性 Amazon EMR 叢集：持久性 Amazon EMR 叢集會在資料處理任務完成後繼續執行。

Amazon EMR 和 Hadoop 都會產生報告叢集狀態的日誌檔案。根據預設，這些會寫入 /mnt/var/log/ 目錄中的主節點。根據您在啟動叢集時設定叢集的方式，您也可以將這些日誌儲存至 Amazon Simple Storage Service (Amazon S3)，並透過圖形偵錯工具檢視它們。請注意，只有在叢集啟動時，才能指定 Amazon S3 記錄。使用此組態，日誌會每 5 分鐘從主節點傳送至 Amazon S3 位置。對於暫時性叢集，Amazon S3 記錄很重要，因為叢集會在處理完成時消失，而且這些日誌檔案可用來偵錯任何失敗的任務。

模式使用 AWS CloudFormation 範本來部署安全控制，以監控 API 呼叫，並在 "RunJobFlow" 上啟動 Amazon CloudWatch Events。RunJobFlow." 觸發程序會叫用執行 Python 指令碼的 AWS Lambda。Lambda 函數會從事件 JSON 輸入擷取 EMR 叢集 ID，並檢查 Amazon S3 日誌 URI。如果找不到 Amazon S3 URI，Lambda 函數會傳送 Amazon Simple Notification Service (Amazon SNS) 通知，詳細說明來自該通知的 EMR 叢集名稱、違規詳細資訊、AWS 區域、AWS 帳戶和 Lambda Amazon Resource Name (ARN)。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Lambda 程式碼 .zip 檔案的 S3 儲存貯體
- 您想要接收違規通知的電子郵件地址

限制

- 此偵測性控制是區域性控制，必須部署在您打算監控的 AWS 區域中。

產品版本

- Amazon EMR 4.8.0 版及更新版本

Architecture

目標技術堆疊

- Amazon CloudWatch Events 事件
- Amazon EMR
- Lambda 函式
- S3 bucket (S3 儲存貯體)
- Amazon SNS

目標架構

自動化和擴展

- 如果您使用的是 AWS Organizations，則可以使用 AWS CloudFormation StackSets，將此範本部署到您要監控的多個帳戶中。

工具

工具

- AWS CloudFormation – AWS CloudFormation 可協助您使用基礎設施做為程式碼來建立模型和設定 AWS 資源。
- AWS Cloudwatch Events – AWS CloudWatch Events 提供近乎即時的系統事件串流，說明 AWS 資源的變更。
- Amazon EMR – Amazon EMR 是一種受管叢集平台，可簡化大數據架構的執行。
- AWS Lambda – AWS Lambda 支援執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。

- Amazon S3 – Amazon S3 是一種 Web 服務介面，可用來從 Web 上的任何位置存放和擷取任意數量的資料。
- Amazon SNS – Amazon SNS 是一種 Web 服務，可協調和管理發佈者和用戶端之間的訊息傳遞或傳送，包括 Web 伺服器和電子郵件地址。

Code

- 專案的 .zip 檔案可作為附件使用。

史詩

定義 S3 儲存貯體

任務	Description	所需的技能
定義 S3 儲存貯體。	若要託管 Lambda 程式碼 .zip 檔案，請選擇或建立具有不包含正斜線之唯一名稱的 S3 儲存貯體。S3 儲存貯體名稱全域唯一，且命名空間由所有 AWS 帳戶共用。您的 S3 儲存貯體必須與正在評估的 Amazon EMR 叢集位於相同的 AWS 區域。	雲端架構師

將 Lambda 程式碼上傳至 S3 儲存貯體

任務	Description	所需的技能
將 Lambda 程式碼上傳至 S3 儲存貯體。	將「附件」區段中提供的 Lambda 程式碼 .zip 檔案上傳至 S3 儲存貯體。S3 儲存貯體必須與正在評估的 Amazon EMR 叢集位於相同的區域。	雲端架構師

部署 AWS CloudFormation 範本

任務	Description	所需的技能
部署 AWS CloudFormation 範本。	在 AWS CloudFormation 主控台上，在與 S3 儲存貯體相同的區域中，部署做為此模式附件提供的 AWS CloudFormation 範本。在下一個史詩中，提供參數的值。如需部署 AWS CloudFormation 範本的詳細資訊，請參閱「相關資源」一節。	雲端架構師

完成 AWS CloudFormation 範本中的參數

任務	Description	所需的技能
命名 S3 儲存貯體。	輸入您在第一個 epic 中建立的 S3 儲存貯體名稱。	雲端架構師
提供 Amazon S3 金鑰。	在您的 S3 儲存貯體中提供 Lambda 程式碼 .zip 檔案的位置，不帶正斜線（例如，<directory>/<file-name>.zip)。	雲端架構師
提供電子郵件地址。	提供作用中的電子郵件地址以接收 Amazon SNS 通知。	雲端架構師
定義記錄層級。	定義 Lambda 函數的記錄層級和頻率。"Info" 會指定應用程式進度的詳細資訊訊息。「錯誤」會指定仍然可以允許應用程式繼續執行的錯誤事件。「警告」會指定潛在的有害情況。	雲端架構師

確認訂閱

任務	Description	所需的技能
確認訂閱。	當範本成功部署時，它會傳送訂閱電子郵件訊息到提供的電子郵件地址。您必須確認此電子郵件訂閱，才能接收違規通知。	雲端架構師

相關資源

- AWS Lambda
- Amazon EMR 記錄
- 部署 AWS CloudFormation 範本

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS Glue 任務和 Python 產生測試資料

Moinul Al-Mamun，Amazon Web Services

總結

此模式說明如何建立以 Python 撰寫的 AWS Glue 任務，以快速且輕鬆地同時產生數百萬個範例檔案。範例檔案存放在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中。快速產生大量範例檔案的功能對於測試或評估 AWS 雲端中的服務至關重要。例如，您可以對 Amazon S3 字首中的數百萬個小型檔案執行資料分析，以測試 AWS Glue Studio 或 AWS Glue DataBrew 任務的效能。

雖然您可以使用其他 AWS 服務來產生範例資料集，但我們建議您使用 AWS Glue。您不需要管理任何基礎設施，因為 AWS Glue 是無伺服器資料處理服務。您可以攜帶程式碼，並在 AWS Glue 叢集中執行。此外，AWS Glue 會佈建、設定和擴展執行任務所需的資源。您只需為您的任務在執行時使用的資源付費。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- AWS Command Line Interface (AWS CLI)，已安裝並設定為使用 AWS 帳戶

產品版本

- Python 3.9
- AWS CLI 第 2 版

限制

每次觸發的 AWS Glue 任務數目上限為 50。如需詳細資訊，請參閱 AWS Glue 端點和配額。

Architecture

下圖說明以 AWS Glue 任務為中心的架構範例，該任務會將其輸出（即範例檔案）寫入 S3 儲存貯體。

圖表包含下列工作流程：

1. 您可以使用 AWS CLI、AWS 管理主控台或 API 來啟動 AWS Glue 任務。AWS CLI 或 API 可讓您自動化調用任務的平行化，並減少產生範例檔案的執行時間。
2. AWS Glue 任務會隨機產生檔案內容，將內容轉換為 CSV 格式，然後將內容儲存為通用字首下的 Amazon S3 物件。每個檔案小於 KB。AWS Glue 任務接受兩個使用者定義的任務參數：START_RANGE和 END_RANGE。您可以使用這些參數來設定檔案名稱，以及每個任務執行在 Amazon S3 中產生的檔案數目。您可以平行執行此任務的多個執行個體（例如 100 個執行個體）。

工具

- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- AWS Glue 是全受管的擷取、轉換和載入 (ETL) 服務。它可協助您可靠地分類、清理、擴充和移動資料存放區和資料串流之間的資料。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。

最佳實務

實作此模式時，請考慮下列 AWS Glue 最佳實務：

- 使用正確的 AWS Glue 工作者類型來降低成本。我們建議您了解工作者類型的不同屬性，然後根據 CPU 和記憶體需求，為您的工作負載選擇正確的工作者類型。對於此模式，我們建議您使用 Python shell 任務作為任務類型，以將 DPU 降至最低並降低成本。如需詳細資訊，請參閱《AWS Glue 開發人員指南》中的在 AWS Glue 中新增任務。 AWS Glue
- 使用正確的並行限制來擴展您的任務。我們建議您根據時間需求和所需的檔案數量來建立 AWS Glue 任務的最大並行數量。
- 首先開始產生少量檔案。若要在建置 AWS Glue 任務時降低成本並節省時間，請從少量檔案（例如 1，000) 開始。這可讓您更輕鬆地進行故障診斷。如果產生少量檔案成功，則可以擴展到更多檔案。
- 請先在本機執行 。若要在建置 AWS Glue 任務時降低成本並節省時間，請在本機開始開發並測試程式碼。如需設定 Docker 容器的說明，以協助您在 shell 和整合開發環境 (IDE) 中撰寫 AWS Glue 擷取、轉換和載入 (ETL) 任務，請參閱 AWS 大數據部落格上的使用容器文章在本機開發 AWS Glue ETL 任務。

如需更多 AWS Glue 最佳實務，請參閱 AWS Glue 文件中的最佳實務。

史詩

建立目的地 S3 儲存貯體和 IAM 角色

任務	Description	所需的技能
建立 S3 儲存貯體以存放檔案。	建立 S3 儲存貯體和其中的字首。 **Note** 此模式會將 `s3://{your-s3-bucket-name}/small-files/` 位置用於示範用途。	應用程式開發人員
建立和設定 IAM 角色。	您必須建立 AWS Glue 任務可用來寫入 S3 儲存貯體的 IAM 角色。 1. 建立 IAM 角色（例如，稱為 `"AWSGlueServiceRole-smallfiles"`）。 2. 選擇 AWS Glue 作為政策的信任實體。 3. 將名為 的 AWS 受管政策`"AWSGlueServiceRole"` 連接至角色。 4. 根據下列組態建立名為 `"s3-small-file-access"` 的內嵌政策或客戶受管政策。將 取	應用程式開發人員

任務	Description	所需的技能
	代"{bucket}" 為您的儲存貯體名稱。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "s3:GetObject", "s3:PutObject" ], "Resource": [ "arn:aws:s3:::{bucket}/small-files/input/*" ] } ] }` 5. 將"s3-small-file-access" 政策連接至您的角色。	

建立和設定 AWS Glue 任務以處理並行執行

任務	Description	所需的技能
建立 AWS Glue 任務。	您必須建立產生內容的 AWS Glue 任務，並將其存放在 S3 儲存貯體中。	應用程式開發人員

任務	Description	所需的技能
	建立 AWS Glue 任務，然後完成下列步驟來設定您的任務： 1. 登入 AWS 管理主控台並開啟 AWS Glue 主控台。 2. 在導覽窗格中的資料整合和 ETL 下，選擇任務。 3. 在建立任務區段中，選擇 Python Shell 指令碼編輯器。 4. 在選項區段中，選取使用樣板程式碼建立新指令碼，然後選擇建立。 5. 選擇任務詳細資訊。 6. 針對名稱，輸入 create_small_files。 7. 針對 IAM 角色，選取您先前建立的 IAM 角色。 8. 在此任務執行區段中，選擇您要撰寫的新指令碼。 9. 展開進階屬性。 10. 針對並行上限，輸入 100 做為示範之用。注意：並行上限定義您可以平行執行的任務執行個體數量。 11. 選擇儲存。	

任務	Description	所需的技能

更新任務代碼。	1. 開啟 AWS Glue 主控台。
	2. 在導覽窗格中，選擇 Jobs (任務)。
	3. 在任務區段中，選擇您先前建立的任務。
	4. 選擇指令碼索引標籤，然後根據下列程式碼更新指令碼。使用您的值更新 PREFIX、 BUCKET_NAME 和 text_str變數。
	`from awsglue.utils import getResolvedOptions`
	`import sys`
	`import boto3`
	`from random import randrange`
	
	`# Two arguments`
	`args = getResolvedOptions(sys.argv, ['START_RANGE', 'END_RANGE'])`
	
	`START_RANGE = int(args['START_RANGE'])`
	`END_RANGE = int(args['END_RANGE'])`
	
	`BUCKET_NAME = '{BUCKET_NAME}'`
	`PREFIX = 'small-files/input/'`
	`s3 = boto3.resource('s3')`
		應用程式開發人員

任務	Description	所需的技能

	for x in range(START_RANGE, END_RANGE):
	    # generate file name
	    file_name = f"input_{x}.txt"
	    # generate text
	    text_str = str(randrange(100000))+","+str(randrange(100000))+", " + str(randrange(10000000)) + "," + str(randrange(10000))
	    # write in s3
	s3.Object(BUCKET_NAME, PREFIX + file_name).put(Body=text_str)
	5. 選擇儲存。	

從命令列或主控台執行 AWS Glue 任務

任務	Description	所需的技能
從命令列執行 AWS Glue 任務。	若要從 AWS CLI 執行您的 AWS Glue 任務，請使用您的值執行下列命令： cmd:~$ aws glue start-job-run --job-name create_small_files --arguments '{"--START_RANGE":"0","--END_RANGE":"1000000"}'	應用程式開發人員

任務	Description	所需的技能

```
cmd:~$ aws glue start-
job-run --job-name
 create_small_files
 --arguments '{"--STAR
T_RANGE":"1000000"
,"--END_RANGE":"20
00000"}'
```

> **Note**
>
> 如需從 AWS 管理主控台執行 AWS Glue 任務的指示，請參閱此模式中 AWS 管理主控台案例的執行 AWS Glue 任務。

> **Tip**
>
> 如果您想要使用不同的參數一次執行多個執行，建議您使用 AWS CLI 來執行 AWS Glue 任務，如上述範例所示。

若要產生使用特定平行化因素產生定義數量的檔案所需的所有 AWS CLI 命令，請執行下列堡壘程式碼（使用您的值）：

```
# define parameters
NUMBER_OF_FILES=
10000000;
```

任務	Description	所需的技能

	PARALLELIZATION=50;
	
	# initialize
	_SB=0;
	
	# generate commands
	for i in $(seq 1
	$PARALLELIZATION);
	do
	echo aws glue
	start-job-run --
	job-name create_sm
	all_files --argumen
	ts "'"'{"--START_RANG
	E":"'$(((NUMBER_OF
	_FILES/PARALLELIZA
	TION) * (i-1) +
	_SB))'","--END_RAN
	GE":"'$(((NUMBER_O
	F_FILES/PARALLELIZ
	ATION) * (i)))'"}'"'";
	_SB=1;
	done
	
	如果您使用上述指令碼，請考慮下列事項：
	
	• 指令碼可簡化大規模呼叫和產生小型檔案的過程。
	• PARALLELIZATION 使用您的值更新 NUMBER_OF_FILES 和 。
	• 上述指令碼會列印您必須執行的命令清單。複製這些輸出命令，然後在終端機中執行它們。	

任務	Description	所需的技能
	• 如果您想要直接從指令碼中執行命令，請移除第 11 行中的 echo陳述式。 **Note** 若要查看上述指令碼的輸出範例，請參閱此模式的額外資訊區段中的 Shell 指令碼輸出。	
在 AWS 管理主控台中執行 AWS Glue 任務。	1. 登入 AWS 管理主控台並開啟 AWS Glue 主控台。 2. 在導覽窗格中的資料整合和 ETL 下，選擇任務。 3. 在任務區段中，選擇您的任務。 4. 在參數（選用）區段中，更新您的參數。 5. 選擇動作，然後選擇執行任務。 6. 視需要多次重複步驟 3-5。例如，若要建立 1，000 萬個檔案，請重複此程序 10 次。	應用程式開發人員

任務	Description	所需的技能
檢查 AWS Glue 任務的狀態。	1. 開啟 AWS Glue 主控台。 2. 在導覽窗格中，選擇 Jobs (任務)。 3. 在任務區段中，選擇您先前建立的任務 （ 也就是 `create_small_files`)。 4. 如需深入了解檔案的進度和產生，請檢閱執行 ID、執行狀態和其他資料欄。	應用程式開發人員

相關資源

參考

- AWS 上開放資料的登錄檔
- 用於分析的資料集
- 在 AWS 上開啟資料
- 在 AWS Glue 中新增任務
- AWS Glue 入門

指南和模式

- AWS Glue 最佳實務
- 負載測試應用程式

其他資訊

基準測試

此模式用於產生 1，000 萬個檔案，使用不同的平行化參數做為基準測試的一部分。下表顯示測試的輸出：

平行化	任務執行產生的檔案數目	任務持續時間	速度
10	1,000,000	6 小時 40 分鐘	非常慢
50	200,000	80 分鐘	適中
100	100,000	40 分鐘	快速

如果您想要讓程序更快，您可以在任務組態中設定更多並行執行。您可以根據您的需求輕鬆調整任務組態，但請記住，有 AWS Glue 服務配額限制。如需詳細資訊，請參閱 AWS Glue 端點和配額。

Shell 指令碼輸出

下列範例顯示此模式中從命令列故事執行 AWS Glue 任務的 shell 指令碼輸出。

```
user@MUC-1234567890 MINGW64 ~
  $ # define parameters
  NUMBER_OF_FILES=10000000;
  PARALLELIZATION=50;
  # initialize
  _SB=0;

  # generate commands
  for i in $(seq 1 $PARALLELIZATION);
   do
         echo aws glue start-job-run --job-name create_small_files --arguments
 "'"'{"--START_RANGE":"'$(((NUMBER_OF_FILES/PARALLELIZATION)  (i-1) + SB))'","--
ENDRANGE":"'$(((NUMBER_OF_FILES/PARALLELIZATION)  (i)))'"}'"'";
         _SB=1;
   done

  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"0","--END_RANGE":"200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"200001","--END_RANGE":"400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"400001","--END_RANGE":"600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"600001","--END_RANGE":"800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"800001","--END_RANGE":"1000000"}'
```

```
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"1000001","--END_RANGE":"1200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"1200001","--END_RANGE":"1400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"1400001","--END_RANGE":"1600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"1600001","--END_RANGE":"1800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"1800001","--END_RANGE":"2000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"2000001","--END_RANGE":"2200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"2200001","--END_RANGE":"2400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"2400001","--END_RANGE":"2600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"2600001","--END_RANGE":"2800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"2800001","--END_RANGE":"3000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"3000001","--END_RANGE":"3200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"3200001","--END_RANGE":"3400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"3400001","--END_RANGE":"3600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"3600001","--END_RANGE":"3800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"3800001","--END_RANGE":"4000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"4000001","--END_RANGE":"4200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"4200001","--END_RANGE":"4400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"4400001","--END_RANGE":"4600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"4600001","--END_RANGE":"4800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"4800001","--END_RANGE":"5000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"5000001","--END_RANGE":"5200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"5200001","--END_RANGE":"5400000"}'
```

```
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"5400001","--END_RANGE":"5600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"5600001","--END_RANGE":"5800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"5800001","--END_RANGE":"6000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"6000001","--END_RANGE":"6200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"6200001","--END_RANGE":"6400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"6400001","--END_RANGE":"6600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"6600001","--END_RANGE":"6800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"6800001","--END_RANGE":"7000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"7000001","--END_RANGE":"7200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"7200001","--END_RANGE":"7400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"7400001","--END_RANGE":"7600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"7600001","--END_RANGE":"7800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"7800001","--END_RANGE":"8000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"8000001","--END_RANGE":"8200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"8200001","--END_RANGE":"8400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"8400001","--END_RANGE":"8600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"8600001","--END_RANGE":"8800000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"8800001","--END_RANGE":"9000000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"9000001","--END_RANGE":"9200000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"9200001","--END_RANGE":"9400000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"9400001","--END_RANGE":"9600000"}'
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"9600001","--END_RANGE":"9800000"}'
```

```
  aws glue start-job-run --job-name create_small_files --arguments '{"--
START_RANGE":"9800001","--END_RANGE":"10000000"}'

  user@MUC-1234567890 MINGW64 ~
```

常見問答集

我應該使用多少個並行執行或平行任務？

並行執行和平行任務的數量取決於您的時間需求和所需的測試檔案數量。建議您檢查正在建立的檔案大小。首先，檢查 AWS Glue 任務產生所需檔案數量所需的時間。然後，使用正確數量的並行執行來滿足您的目標。例如，如果您假設 100，000 個檔案需要 40 分鐘才能完成執行，但目標時間為 30 分鐘，則必須增加 AWS Glue 任務的並行設定。

我可以使用此模式建立哪種類型的內容？

您可以建立任何類型的內容，例如具有不同分隔符號的文字檔案（例如 PIPE、JSON 或 CSV)。此模式使用 Boto3 寫入檔案，然後將檔案儲存在 S3 儲存貯體中。

在 S3 儲存貯體中，我需要什麼層級的 IAM 許可？

您必須擁有允許`Write`存取 S3 儲存貯體中物件的身分型政策。如需詳細資訊，請參閱 Amazon S3 文件中的 Amazon S3：允許對 S3 儲存貯體中的物件進行讀取和寫入存取。 Amazon S3

使用 AWS IoT Greengrass，以經濟實惠的方式直接將 IoT 資料擷取至 Amazon S3 AWS IoT

Sebastian Viviani 和 Rizwan Syed，Amazon Web Services

總結

此模式說明如何使用 AWS IoT Greengrass 第 2 版裝置，以經濟實惠的方式將物聯網 (IoT) 資料直接擷取至 Amazon Simple Storage Service (Amazon S3) 儲存貯體。裝置會執行自訂元件來讀取 IoT 資料，並將資料儲存在持久性儲存體（即本機磁碟區）中。然後，裝置會將 IoT 資料壓縮為 Apache Parquet 檔案，並定期將資料上傳至 S3 儲存貯體。

您擷取的 IoT 資料數量和速度受限於您的邊緣硬體功能和網路頻寬。您可以使用 Amazon Athena 以經濟實惠的方式分析擷取的資料。Athena 使用 Amazon Managed Grafana 支援壓縮的 Apache Parquet 檔案和資料視覺化。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在 AWS IoT Greengrass 第 2 版上執行並從感應器收集資料的邊緣閘道（資料來源和資料收集程序超出此模式的範圍，但您可以使用幾乎任何類型的感應器資料。 此模式使用本機 MQTT 代理程式搭配可在本機發佈資料的感應器或閘道。）
- AWS IoT Greengrass 元件、角色和 SDK 相依性
- 將資料上傳至 S3 儲存貯體的串流管理員元件
- 適用於 Java 的 AWS 開發套件、適用於 JavaScript 的 AWS 開發套件或適用於 Python 的 AWS 開發套件 (Boto3) APIs

限制

- 此模式中的資料不會即時上傳至 S3 儲存貯體。有延遲期間，您可以設定延遲期間。資料會在邊緣裝置中暫時緩衝，然後在期間過期後上傳。
- 開發套件僅適用於 Java、Node.js 和 Python。

Architecture

目標技術堆疊

- Amazon S3
- AWS IoT Greengrass
- MQTT 代理程式
- 串流管理員元件

目標架構

下圖顯示設計用來擷取 IoT 感應器資料的架構，並將該資料存放在 S3 儲存貯體中。

該圖顯示以下工作流程：

1. 多個感應器（例如，溫度和閥）更新會發佈至本機 MQTT 代理程式。
2. 訂閱這些感應器的 Parquet 檔案壓縮器會更新主題並接收這些更新。
3. Parquet 檔案壓縮器會在本機存放更新。
4. 期間經過之後，儲存的檔案會壓縮為 Parquet 檔案，並傳遞給串流管理員，以上傳到指定的 S3 儲存貯體。
5. 串流管理員會將 Parquet 檔案上傳至 S3 儲存貯體。

> **Note**
>
> 串流管理員 (`StreamManager`) 是受管元件。如需如何將資料匯出至 Amazon S3 的範例，請參閱 AWS IoT Greengrass 文件中的串流管理員。您可以使用本機 MQTT 代理程式做為元件或其他代理程式，例如 Eclipse Mosquitto。

工具

AWS 工具

- Amazon Athena 是一種互動式查詢服務，可協助您使用標準 SQL 直接在 Amazon S3 中分析資料。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

- AWS IoT Greengrass 是一種開放原始碼 IoT 邊緣執行期和雲端服務，可協助您在裝置上建置、部署和管理 IoT 應用程式。

其他工具

- Apache Parquet 是一種開放原始碼資料欄導向的資料檔案格式，專為儲存和擷取而設計。
- MQTT （訊息佇列遙測傳輸） 是一種輕量型傳訊通訊協定，專為受限裝置而設計。

最佳實務

針對上傳的資料使用正確的分割區格式

S3 儲存貯體 （例如 "myAwesomeDataSet/"或 "dataFromSource") 中沒有根字首名稱的特定要求，但我們建議您使用有意義的分割區和字首，以便輕鬆了解資料集的目的。

我們也建議您在 Amazon S3 中使用正確的分割，以便在資料集上以最佳方式執行查詢。在下列範例中，資料會以 HIVE 格式分割，以便最佳化每個 Athena 查詢掃描的資料量。這可改善效能並降低成本。

```
s3://<ingestionBucket>/<rootPrefix>/year=YY/month=MM/day=DD/
HHMM_<suffix>.parquet
```

史詩

設定您的環境

任務	Description	所需的技能
建立 S3 儲存貯體。	1. 建立 S3 儲存貯體或使用現有的儲存貯體。 2. 為您要擷取 IoT 資料的 S3 儲存貯體建立有意義的字首 （例如 `s3:\\<bucket>\<prefix>`)。 3. 記錄您的字首以供日後使用。	應用程式開發人員

任務	Description	所需的技能
將 IAM 許可新增至 S3 儲存貯體。	若要授予使用者對您先前建立的 S3 儲存貯體和字首的寫入存取權，請將下列 IAM 政策新增至您的 AWS IoT Greengrass 角色： `{ "Version": "2012-10-17", "Statement": [ { "Sid": "S3DataUpload", "Effect": "Allow", "Action": [ "s3:List*", "s3:Put*" ], "Resource": [ "arn:aws:s3:::<ingestionBucket>", "arn:aws:s3:::<ingestionBucket>/<prefix>/*" ] } ] }` 如需詳細資訊，請參閱 Aurora 文件中的建立 IAM 政策以存取 Amazon S3 資源。	應用程式開發人員

任務	Description	所需的技能
	接著，更新 S3 儲存貯體的資源政策（如有需要），以允許使用正確的 AWS 主體進行寫入存取。	

建置和部署 AWS IoT Greengrass 元件

任務	Description	所需的技能
更新 元件的配方。	根據下列範例，在建立部署時更新元件組態： `{ "region": "<region>", "parquet_period": <period>, "s3_bucket": "<s3Bucket>", "s3_key_prefix": "<s3prefix>" }` 將 取代`<region>`為您的 AWS 區域、`<period>`將 取代為您的週期性間隔、`<s3Bucket>` 將 取代為您的 S3 儲存貯體，並將 `<s3prefix>` 取代為您的字首。	應用程式開發人員
建立 元件。	執行以下任意一項： • 建立 元件。 • 將元件新增至 CI/CD 管道（如果有的話）。請務必將成品從成品儲存庫複製到 AWS IoT Greengrass 成品	應用程式開發人員

任務	Description	所需的技能
	儲存貯體。然後，建立或更新您的 AWS IoT Greengrass 元件。 • **Note** 新增 MQTT 代理程式做為元件，或稍後手動新增。：此決策會影響您可以與代理程式搭配使用的身分驗證機制。手動新增代理程式會將代理程式與 AWS IoT Greengrass 分離，並啟用代理程式的任何支援身分驗證機制。AWS 提供的代理程式元件具有預先定義的身分驗證機制。如需詳細資訊，請參閱 MQTT 3.1.1 代理程式 (Moquette) 和 MQTT 5 代理程式 (EMQX)。	

任務	Description	所需的技能
更新 MQTT 用戶端。	範例程式碼不使用身分驗證，因為元件會在本機連線至代理程式。如果您的案例不同，請視需要更新 MQTT 用戶端區段。此外，請執行下列動作： 1. 更新訂閱中的 MQTT 主題。 2. 視需要更新 MQTT 訊息剖析器，因為每個來源的訊息可能不同。	應用程式開發人員

將元件新增至 AWS IoT Greengrass 第 2 版核心裝置

任務	Description	所需的技能
更新核心裝置的部署。	如果 AWS IoT Greengrass 第 2 版核心裝置的部署已存在，請修改部署。如果部署不存在，請建立新的部署。 若要為元件提供正確的名稱，請根據下列項目更新新元件的日誌管理員組態（如果需要）： `{ "logsUploaderConfiguration": { "systemLogsConfiguration": { ... }, "componentLogsConfigurationMap": { "<com.iot.ingest.parquet>": {`	應用程式開發人員

任務	Description	所需的技能

	<pre>"minimumL
	ogLevel": "INFO",
	"diskSpac
	eLimit": "20",
	"diskSpac
	eLimitUnit": "MB",
	"deleteLo
	gFileAfterCloudUpl
	oad": "false"
	}
	...
	}
	},
	"periodicUploadInt
	ervalSec": "300"
	}</pre> 最後，完成 AWS IoT Greengrass 核心裝置的部署修訂。	

驗證資料擷取至 S3 儲存貯體

任務	Description	所需的技能
檢查 AWS IoT Greengrass 磁碟區的日誌。	檢查以下各項： • MQTT 用戶端已成功連線至本機 MQTT 代理程式。 • MQTT 用戶端已訂閱正確的主題。 • 感應器更新訊息即將傳送至 MQTT 主題上的代理程式。 • Parquet 壓縮在每個週期性間隔進行。	應用程式開發人員

任務	Description	所需的技能
檢查 S3 儲存貯體。	驗證資料是否正在上傳到 S3 儲存貯體。您可以在每個期間看到正在上傳的檔案。 您也可以在下一節中查詢資料，確認資料是否已上傳至 S3 儲存貯體。	應用程式開發人員

從 Athena 設定查詢

任務	Description	所需的技能
建立資料庫和資料表。	1. 建立 AWS Glue 資料庫（如有需要）。 2. 在 AWS Glue 中手動建立資料表，或在 AWS Glue 中執行爬蟲程式。	應用程式開發人員
授予 Athena 對資料的存取權。	1. 更新許可以允許 Athena 存取 S3 儲存貯體。如需詳細資訊，請參閱 Athena 文件中的 AWS Glue Data Catalog 中的精細存取資料庫和資料表。 2. 查詢資料庫中的資料表。	應用程式開發人員

疑難排解

問題	解決方案
MQTT 用戶端無法連線	• 驗證 MQTT 代理程式上的許可。如果您有來自 AWS 的 MQTT 代理程式，請參閱 MQTT

問題	解決方案
	3.1.1 代理程式 (Moquette) 和 MQTT 5 代理程式 (EMQX)。 • 驗證 MQTT 用戶端上的登入資料。如果您有來自 AWS 的 MQTT 代理程式，請參閱 MQTT 3.1.1 代理程式 (Moquette) 和 MQTT 5 代理程式 (EMQX)。
MQTT 用戶端無法訂閱	驗證 MQTT 代理程式上的許可。如果您有來自 AWS 的 MQTT 代理程式，請參閱 MQTT 3.1.1 代理程式 (Moquette) 和 MQTT 5 代理程式 (EMQX)。
不會建立 Parquet 檔案	• 驗證 MQTT 主題是否正確。 • 確認來自感應器的 MQTT 訊息格式正確。
物件不會上傳到 S3 儲存貯體	• 確認您有網際網路連線和端點連線。 • 驗證 S3 儲存貯體的資源政策是否正確。 • 驗證 AWS IoT Greengrass 第 2 版核心裝置角色的許可。

相關資源

- DataFrame (Pandas 文件)
- Apache Parquet 文件 (Parquet 文件)
- 開發 AWS IoT Greengrass 元件 (AWS IoT Greengrass 開發人員指南，第 2 版)
- 將 AWS IoT Greengrass 元件部署至裝置 (AWS IoT Greengrass 開發人員指南，第 2 版)
- 與本機 IoT 裝置互動 (AWS IoT Greengrass 開發人員指南，第 2 版)
- MQTT 3.1.1 代理程式 (Moquette) (AWS IoT Greengrass 開發人員指南，第 2 版)
- MQTT 5 代理程式 (EMQX) (AWS IoT Greengrass 開發人員指南，第 2 版)

其他資訊

成本分析

下列成本分析案例示範此模式中涵蓋的資料擷取方法如何影響 AWS 雲端中的資料擷取成本。此案例中的定價範例是以發佈時的價格為基礎。價格可能變動。此外，您的成本可能會根據您的 AWS 區域、AWS 服務配額以及與雲端環境相關的其他因素而有所不同。

輸入訊號集

此分析使用下列一組輸入訊號，做為比較 IoT 擷取成本與其他可用替代方案的基礎。

訊號數量	Frequency (頻率)	每個訊號的資料
125	25 Hz	8 位元組

在此案例中，系統會接收 125 個訊號。每個訊號都是 8 個位元組，每 40 毫秒 (25 Hz) 就會發生。這些訊號可以個別出現，或在常見的承載中分組。您可以選擇根據您的需求分割和封裝這些訊號。您也可以判斷延遲。延遲包含接收、累積和擷取資料的期間。

為了比較，此案例的擷取操作是以 `us-east-1` AWS 區域為基礎。成本比較僅適用於 AWS 服務。硬體或連線能力等其他成本不會納入分析。

成本比較

下表顯示每個擷取方法的每月成本，以美元 (USD) 為單位。

Method	每月成本
AWS IoT SiteWise*	331.77 USD
AWS IoT SiteWise Edge 搭配資料處理套件（將所有資料保留在邊緣）	200 USD
用於存取原始資料的 AWS IoT Core 和 Amazon S3 規則	84.54 USD
Parquet 檔案在邊緣壓縮並上傳至 Amazon S3	0.5 USD

*資料必須進行縮減取樣，以符合服務配額。這表示此方法有一些資料遺失。

替代方法

本節顯示下列替代方法的同等成本：

- AWS IoT SiteWise – 每個訊號都必須以個別訊息上傳。因此，每月的訊息總數為 125 × 25 × 3600 × 24 × 30，或每月 81 億則訊息。不過，AWS IoT SiteWise 每個屬性每秒只能處理 10 個資料點。假設資料縮減取樣至 10 Hz，則每月訊息數量會減少至 125 × 10 × 3600 × 24 × 30，或 32.4 億。如果您使用 發佈者元件，以 10 個（每百萬則訊息 1 USD) 的群組來封裝衡量值，則每月 324 USD 的費用。假設每則訊息為 8 個位元組 (1 Kb/125)，即 25.92 Gb 的資料儲存體。這會增加每月 7.77 USD 的成本。第一個月的總成本為 331.77 USD，每月增加 7.77 USD。
- AWS IoT SiteWise Edge 搭配資料處理套件，包括在邊緣完全處理的所有模型和訊號（即沒有雲端擷取）– 您可以使用資料處理套件做為替代方案，以降低成本並設定在邊緣計算的所有模型。即使未執行實際計算，這仍僅適用於儲存體和視覺化。在此情況下，邊緣閘道必須使用功能強大的硬體。每月固定費用為 200 USD。
- MQTT 直接擷取至 AWS IoT Core 和 IoT 規則，以將原始資料儲存在 Amazon S3 中 – 假設所有訊號都發佈在通用承載中，發佈至 AWS IoT Core 的訊息總數為每月 25×3600×24×30 或 6480 萬。每百萬則訊息 1 USD，即每月 64.8 USD 的成本。每百萬規則啟用 0.15 USD，每則訊息一個規則，每月增加 19.44 USD。在 Amazon S3 中，每 Gb 儲存的成本為 0.023 USD，每月增加 1.5 USD（每月增加以反映新資料）。第一個月的總成本為 84.54 USD，每月增加 1.5 USD。
- 在 Parquet 檔案中的邊緣壓縮資料並上傳至 Amazon S3（建議的方法) – 壓縮率取決於資料類型。使用針對 MQTT 測試的相同工業資料，整個月的總輸出資料為 1.2 Gb。此費用為每月 0.03 USD。其他基準測試中描述的壓縮率（使用隨機資料）的順序為 66%（較接近最壞情況）。總資料為 21 Gb，每月費用為 0.5 USD。

Parquet 檔案產生器

下列程式碼範例顯示以 Python 撰寫的 Parquet 檔案產生器結構。程式碼範例僅供說明之用，如果貼入您的環境，則無法運作。

```
import queue
import paho.mqtt.client as mqtt
import pandas as pd

#queue for decoupling the MQTT thread
messageQueue = queue.Queue()
client = mqtt.Client()
streammanager = StreamManagerClient()

def feederListener(topic, message):
    payload = {
        "topic" : topic,
        "payload" : message,
```

```
    }
    messageQueue.put_nowait(payload)

def on_connect(client_instance, userdata, flags, rc):
    client.subscribe("#",qos=0)

def on_message(client, userdata, message):
    feederListener(topic=str(message.topic),
 message=str(message.payload.decode("utf-8")))


filename = "tempfile.parquet"
streamname = "mystream"
destination_bucket= "amzn-s3-demo-bucket"
keyname="mykey"
period= 60

client.on_connect = on_connect
client.on_message = on_message
streammanager.create_message_stream(
            MessageStreamDefinition(name=streamname,
 strategy_on_full=StrategyOnFull.OverwriteOldestData)
        )


while True:
   try:
       message = messageQueue.get(timeout=myArgs.mqtt_timeout)
   except (queue.Empty):
       logger.warning("MQTT message reception timed out")

   currentTimestamp = getCurrentTime()
   if currentTimestamp >= nextUploadTimestamp:
       df = pd.DataFrame.from_dict(accumulator)
       df.to_parquet(filename)
       s3_export_task_definition = S3ExportTaskDefinition(input_url=filename,
 bucket=destination_bucket, key=key_name)
       streammanager.append_message(streamname,
 Util.validate_and_serialize_to_json_bytes(s3_export_task_definition))
       accumulator = {}
       nextUploadTimestamp += period
   else:
        accumulator.append(message)
```

使用 Lambda 函數在暫時性 EMR 叢集中啟動 Spark 任務

Dhrubajyoti Mukherjee，Amazon Web Services

總結

此模式使用 Amazon EMR RunJobFlow API 動作啟動暫時性叢集，從 Lambda 函數執行 Spark 任務。暫時性 EMR 叢集的設計會在任務完成或發生任何錯誤時立即終止。暫時性叢集可節省成本，因為它只會在運算時間執行，並在雲端環境中提供可擴展性和彈性。

在 Lambda 函數中使用 Boto3 API 和 Python 程式設計語言啟動暫時性 EMR 叢集。Lambda 函數以 Python 撰寫，提供在需要時啟動叢集的額外彈性。

為了示範範例批次運算和輸出，此模式會從 Lambda 函數在 EMR 叢集中啟動 Spark 任務，並根據虛構公司的範例銷售資料執行批次運算。Spark 任務的輸出將是 Amazon Simple Storage Service (Amazon S3) 中的逗號分隔值 (CSV) 檔案。用於執行運算的虛擬私有雲端 (VPC) 和 AWS Identity and Access Management (IAM) 角色的輸入資料檔案、Spark .jar 檔案、程式碼片段和 AWS CloudFormation 範本會以附件的形式提供。 AWS Identity and Access Management

先決條件和限制

先決條件

- 作用中的 AWS 帳戶

限制

- 一次只能從程式碼啟動一個 Spark 任務。

產品版本

- 在 Amazon EMR 6.0.0 上測試

Architecture

目標技術堆疊

- Amazon EMR

- AWS Lambda
- Amazon S3
- Apache Spark

目標架構

自動化和擴展

若要自動化 Spark-EMR 批次運算，您可以使用下列其中一個選項。

- 實作可在 Cron 排程中啟動 Lambda 函數的 Amazon EventBridge 規則。如需詳細資訊，請參閱教學課程：使用 EventBridge 排程 AWS Lambda 函數。
- 設定 Amazon S3 事件通知，以在檔案到達時啟動 Lambda 函數。
- 透過事件內文和 Lambda 環境變數，將輸入參數傳遞至 AWS Lambda 函數。

工具

AWS 服務

- Amazon EMR 是一種受管叢集平台，可簡化在 AWS 上執行大數據架構，以處理和分析大量資料。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而不需要佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他工具

- Apache Spark 是用於大規模資料處理的多語言分析引擎。

史詩

建立 Amazon EMR 和 Lambda IAM 角色和 VPC

任務	Description	所需的技能
建立 IAM 角色和 VPC。	如果您已有 AWS Lambda 和 Amazon EMR IAM 角色和 VPC，您可以略過此步驟。若要執行程式碼，EMR 叢集和 Lambda 函數都需要 IAM 角色。EMR 叢集還需要具有公有子網路的 VPC，或具有 NAT 閘道的私有子網路。若要自動建立所有 IAM 角色和 VPC，請依原樣部署連接的 AWS CloudFormation 範本，或者您可以依照其他資訊區段中的指定手動建立角色和 VPC。	雲端架構師
請注意 AWS CloudFormation 範本輸出金鑰。	在 CloudFormation 範本成功部署後，導覽至 AWS CloudFormation 主控台中的輸出索引標籤。請注意五個輸出金鑰： • `S3Bucket` • `LambdaExecutionRole` • `ServiceRole` • `JobFlowRole` • `Ec2SubnetId` 建立 Lambda 函數時，您會使用這些索引鍵中的值。	雲端架構師

上傳 Spark .jar 檔案

任務	Description	所需的技能
上傳 Spark .jar 檔案。	將 Spark .jar 檔案上傳至 AWS CloudFormation 堆疊建立的 S3 儲存貯體。儲存貯體名稱與輸出金鑰 相同`S3Bucket`。	一般 AWS

建立 Lambda 函數以啟動 EMR 叢集

任務	Description	所需的技能
建立 Lambda 函式。	在 Lambda 主控台上，使用執行角色建立 Python 3.9+ Lambda 函數。執行角色政策必須允許 Lambda 啟動 EMR 叢集。（請參閱連接的 AWS CloudFormation 範本。）	資料工程師、雲端工程師
複製並貼上程式碼。	將 `lambda_function.py` 檔案中的程式碼取代為此模式其他資訊區段中的程式碼。	資料工程師、雲端工程師
變更程式碼中的參數。	請遵循程式碼中的註解，變更參數值以符合您的 AWS 帳戶。	資料工程師、雲端工程師
啟動 函數以啟動叢集。	啟動 函數，以使用提供的 Spark .jar 檔案開始建立暫時性 EMR 叢集。它會執行 Spark 任務，並在任務完成時自動終止。	資料工程師、雲端工程師
檢查 EMR 叢集狀態。	EMR 叢集啟動後，會顯示在叢集索引標籤下的 Amazon EMR 主控台中。您可以相應地檢查	資料工程師、雲端工程師

任務	Description	所需的技能
	啟動叢集或執行任務時的任何錯誤。	

設定並執行示範範例

任務	Description	所需的技能
上傳 Spark .jar 檔案。	從附件區段下載 Spark .jar 檔案，並將其上傳至 S3 儲存貯體。	資料工程師、雲端工程師
上傳輸入資料集。	將連接`fake_sales_data.csv` 的檔案上傳至 S3 儲存貯體。	資料工程師、雲端工程師
貼上 Lambda 程式碼並變更參數。	從工具區段複製程式碼，並將程式碼貼到 Lambda 函數中，取代程式碼`lambda_function.py` 檔案。變更參數值以符合您的帳戶。	資料工程師、雲端工程師
啟動 函數並驗證輸出。	Lambda 函數使用提供的 Spark 任務啟動叢集後，會在 S3 儲存貯體中產生 .csv 檔案。	資料工程師、雲端工程師

相關資源

- 建置 Spark
- Apache Spark 和 Amazon EMR
- Boto3 文件 run_job_flow 文件
- Apache Spark 資訊和文件

其他資訊

Code

```
"""
Copy paste the following code in your Lambda function. Make sure to change the
 following key parameters for the API as per your account

-Name (Name of Spark cluster)
-LogUri (S3 bucket to store EMR logs)
-Ec2SubnetId (The subnet to launch the cluster into)
-JobFlowRole (Service role for EC2)
-ServiceRole (Service role for Amazon EMR)

The following parameters are additional parameters for the Spark job itself. Change the
 bucket name and prefix for the Spark job (located at the bottom).

-s3://your-bucket-name/prefix/lambda-emr/SparkProfitCalc.jar (Spark jar file)
-s3://your-bucket-name/prefix/fake_sales_data.csv (Input data file in S3)
-s3://your-bucket-name/prefix/outputs/report_1/ (Output location in S3)
"""
import boto3

client = boto3.client('emr')


def lambda_handler(event, context):
    response = client.run_job_flow(
        Name='spark_job_cluster',
        LogUri='s3://your-bucket-name/prefix/logs',
        ReleaseLabel='emr-6.0.0',
        Instances={
            'MasterInstanceType': 'm5.xlarge',
            'SlaveInstanceType': 'm5.large',
            'InstanceCount': 1,
            'KeepJobFlowAliveWhenNoSteps': False,
            'TerminationProtected': False,
            'Ec2SubnetId': 'subnet-XXXXXXXXXXXXX'
        },
        Applications=[{'Name': 'Spark'}],
        Configurations=[
            {'Classification': 'spark-hive-site',
             'Properties': {
```

```
                'hive.metastore.client.factory.class':
'com.amazonaws.glue.catalog.metastore.AWSGlueDataCatalogHiveClientFactory'}
            }
        ],
        VisibleToAllUsers=True,
        JobFlowRole='EMRLambda-EMREC2InstanceProfile-XXXXXXXXX',
        ServiceRole='EMRLambda-EMRRole-XXXXXXXXX',
        Steps=[
            {
                'Name': 'flow-log-analysis',
                'ActionOnFailure': 'TERMINATE_CLUSTER',
                'HadoopJarStep': {
                    'Jar': 'command-runner.jar',
                    'Args': [
                        'spark-submit',
                        '--deploy-mode', 'cluster',
                        '--executor-memory', '6G',
                        '--num-executors', '1',
                        '--executor-cores', '2',
                        '--class', 'com.aws.emr.ProfitCalc',
                        's3://your-bucket-name/prefix/lambda-emr/SparkProfitCalc.jar',
                        's3://your-bucket-name/prefix/fake_sales_data.csv',
                        's3://your-bucket-name/prefix/outputs/report_1/'
                    ]
                }
            }
        ]
    )
```

IAM 角色和 VPC 建立

若要在 Lambda 函數中啟動 EMR 叢集，需要 VPC 和 IAM 角色。您可以使用此模式附件區段中的 AWS CloudFormation 範本來設定 VPC 和 IAM 角色，也可以使用以下連結手動建立角色。

執行 Lambda 和 Amazon EMR 需要下列 IAM 角色。

Lambda 執行角色

Lambda 函數的執行角色會授予其存取 AWS 服務和資源的許可。

Amazon EMR 的服務角色

Amazon EMR 角色會在佈建資源和執行未在叢集內執行的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體內容中執行的服務層級任務時，定義 Amazon EMR 的允許動作。例如，服務角色用於在叢集啟動時佈建 EC2 執行個體。

EC2 執行個體的服務角色

叢集 EC2 執行個體的服務角色（也稱為 Amazon EMR 的 EC2 執行個體描述檔）是一種特殊類型的服務角色，會在執行個體啟動時指派給 Amazon EMR 叢集中的每個 EC2 執行個體。在 Apache Hadoop 上執行的應用程式程序會擔任此角色，以取得與其他 AWS 服務互動的許可。

VPC 和子網路建立

您可以從 VPC 主控台建立 VPC。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS Glue 將 Apache Cassandra 工作負載遷移至 Amazon Keyspaces

Nikolai Kolesnikov、Karthiga Priya Chandran 和 Samir Patel，Amazon Web Services

總結

此模式說明如何在 AWS Glue 上使用 CQLReplicator，將現有的 Apache Cassandra 工作負載遷移至 Amazon Keyspaces（適用於 Apache Cassandra)。您可以使用 AWS Glue 上的 CQLReplicator，將遷移工作負載的複寫延遲降到最低到幾分鐘。您也會了解如何使用 Amazon Simple Storage Service (Amazon S3) 儲存貯體來存放遷移所需的資料，包括 Apache Parquet 檔案、組態檔案和指令碼。此模式假設您的 Cassandra 工作負載託管在虛擬私有雲端 (VPC) 中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上。

先決條件和限制

先決條件

- 具有來源資料表的 Cassandra 叢集
- Amazon Keyspaces 中複寫工作負載的目標資料表
- S3 儲存貯體可存放包含增量資料變更的中繼 Parquet 檔案
- 存放任務組態檔案和指令碼的 S3 儲存貯體

限制

- AWS Glue 上的 CQLReplicator 需要一些時間來佈建 Cassandra 工作負載的資料處理單位 (DPUs)。AWS Glue Cassandra 叢集與 Amazon Keyspaces 中的目標金鑰空間和資料表之間的複寫延遲可能只會持續幾分鐘。

Architecture

來源技術堆疊

- Apache Cassandra
- DataStax 伺服器
- ScyllaDB

目標技術堆疊

- Amazon Keyspaces

遷移架構

下圖顯示範例架構，其中 Cassandra 叢集託管在 EC2 執行個體上，並分散在三個可用區域。Cassandra 節點託管在私有子網路中。

該圖顯示以下工作流程：

1. 自訂服務角色可讓您存取 Amazon Keyspaces 和 S3 儲存貯體。
2. AWS Glue 任務會讀取 S3 儲存貯體中的任務組態和指令碼。
3. AWS Glue 任務會透過連接埠 9042 連接，以從 Cassandra 叢集讀取資料。
4. AWS Glue 任務透過連接埠 9142 連接，將資料寫入 Amazon Keyspaces。

工具

AWS 服務和工具

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- AWS CloudShell 是一種瀏覽器型 shell，您可以使用 AWS Command Line Interface (AWS CLI) 和一系列預先安裝的開發工具來管理 AWS 服務。
- AWS Glue 是一項全受管 ETL 服務，可協助您可靠地分類、清理、擴充和移動資料存放區和資料串流之間的資料。
- Amazon Keyspaces （適用於 Apache Cassandra) 是一種受管資料庫服務，可協助您在 AWS 雲端中遷移、執行和擴展 Cassandra 工作負載。

Code

此模式的程式碼可在 GitHub CQLReplicator 儲存庫中使用。

最佳實務

- 若要判斷遷移所需的 AWS Glue 資源，請估計來源 Cassandra 資料表中的資料列數。例如，每 0.25 個 DPU 250 K 資料列 (2 個 vCPUs，4 GB 記憶體） 與 84 GB 磁碟。
- 在執行 CQLReplicator 之前預熱 Amazon Keyspaces 資料表。例如，八個 CQLReplicator 圖磚 (AWS Glue 任務） 每秒最多可寫入 22 K WCUs，因此目標應預先暖機至每秒最多 25-30 K WCUs。
- 若要啟用 AWS Glue 元件之間的通訊，請針對安全群組中的所有 TCP 連接埠使用自我參考傳入規則。
- 使用增量流量策略，隨時間分配遷移工作負載。

史詩

部署 CQLReplicator

任務	Description	所需的技能
建立目標金鑰空間和資料表。	1. 在 Amazon Keyspaces 中建立金鑰空間和資料表。 如需寫入容量的詳細資訊，請參閱此模式額外資訊區段中的寫入單位計算。 您也可以使用 Cassandra 查詢語言 (CQL) 來建立金鑰空間。如需詳細資訊，請參閱此模式額外資訊區段中的使用 CQL 建立金鑰空間。 **Note** 建立資料表之後，請考慮將資料表切換為隨需容量模式，以避免不必要的費用。	應用程式擁有者、AWS 管理員、DBA、應用程式開發人員

任務	Description	所需的技能
	2. 若要更新為輸送量模式，請執行下列指令碼： `ALTER TABLE target_keyspace.target_table WITH CUSTOM_PROPERTIES = { 'capacity_mode': { 'throughput_mode': 'PAY_PER_REQUEST'} }`	

任務	Description	所需的技能
設定 Cassandra 驅動程式以連線至 Cassandra。	使用下列組態指令碼： `Datastax-java-driver { basic.request.consistency = "LOCAL_QUORUM" basic.contact-points = ["127.0.0.1:9042"] advanced.reconnect-on-init = true basic.load-balancing-policy { local-datacenter = "datacenter1" } advanced.auth-provider = { class = PlainTextAuthProvider username = "user-at-sample" password = "S@MPLE=PASSWORD=" } }` **Note** 上述指令碼使用 Spark Cassandra 連接器。如需詳細資訊，請參閱 Cassandra 的參考組態。	DBA

任務	Description	所需的技能
設定 Cassandra 驅動程式以連線至 Amazon Keyspaces。	使用下列組態指令碼： `datastax-java-driver { basic { load-balancing-policy { local-datacenter = us-west-2 } contact-points = ["cassandra.us-west-2.amazonaws.com:9142"] request { page-size = 2500 timeout = 360 seconds consistency = LOCAL_QUORUM } } advanced { control-connection { timeout = 360 seconds } session-leak.threshold = 6 connection { connect-timeout = 360 seconds init-query-timeout = 360 seconds warn-on-init-error = false } auth-provider = { class = software.aws.mcs.auth.SigV4AuthProvider aws-region = us-west-2`	DBA

任務	Description	所需的技能

	<pre>}
	
	ssl-engine-factory {
	class = DefaultSs
	lEngineFactory
	}
	}
	}</pre>
	Note
	上述指令碼使用 Spark Cassandra 連接器。如需詳細資訊，請參閱 Cassandra 的參考組態。	

任務	Description	所需的技能
為 AWS Glue 任務建立 IAM 角色。	建立名為 的新 AWS 服務角色`glue-cassandra-migration` ，以 AWS Glue 做為信任的實體。 **Note** `glue-cassandra-migration` 應該提供對 S3 儲存貯體和 Amazon Keyspaces 的讀取和寫入存取權。S3 儲存貯體包含 .jar 檔案、Amazon Keyspaces 和 Cassandra 的組態檔案，以及中繼 Parquet 檔案。例如，它包含 `AWSGlueServiceRole` 、 `AmazonS3FullAccess` 和 `AmazonKeyspacesFullAccess` 受管政策。	AWS DevOps

任務	Description	所需的技能

在 AWS CloudShell 中下載 CQLReplicator。	執行下列命令，將專案下載到您的主資料夾：
	
	`git clone https://github.com/aws-samples/cql-replicator.git`
	`cd cql-replicator/glue`
	`# Only for AWS CloudShell, the bc package includes bc and dc. Bc is an arbitrary precision numeric processing arithmetic language`
	`sudo yum install bc -y`	

修改參考組態檔案。	將 `KeyspacesConnector.conf` `Cassandra Connector.conf` 和 複製到專案資料夾中的 `../glue/conf` 目錄。	AWS DevOps

任務	Description	所需的技能

啟動遷移程序。	下列命令會初始化 CQLReplicator 環境。初始涉及複製 .jar 成品，以及建立 AWS Glue 連接器、S3 儲存貯體、AWS Glue 任務、 `migration` 金鑰空間和 `ledger`資料表：
	`cd cql-replicator/glue/bin`
	`./cqlreplicator --state init --sg '"sg-1","sg-2"' \ --subnet "subnet-XXXXXXXXXXXX" \ --az us-west-2a --region us-west-2 \ --glue-iam-role glue-cassandra-migration \ --landing-zone s3://cql-replicator-1234567890-us-west-2`
	該指令碼包括下列參數：
	• `--sg` – 允許從 AWS Glue 存取 Cassandra 叢集的安全群組，並包含所有流量的自我參考傳入規則
	• `--subnet` – Cassandra 叢集所屬的子網路
	• `--az` – 子網路的可用區域
		AWS DevOps

任務	Description	所需的技能
	• `--region` – 部署 Cassandra 叢集的 AWS 區域 • `--glue-iam-role` – AWS Glue 代表您呼叫 Amazon Keyspaces 和 Amazon S3 時可擔任的 IAM 角色許可 • `--landing zone` – 用於重複使用 S3 儲存貯體的選用參數（如果您未提供`--landing zone`參數的值，`init`程序會嘗試建立新的儲存貯體，以存放組態檔案、.jar 成品和中繼檔案。）	
驗證部署。	在您執行先前的命令後，AWS 帳戶應包含下列項目： • AWS Glue 中的 CQLReplicator AWS Glue 任務和 AWS Glue 連接器 AWS Glue • 存放成品的 S3 儲存貯體 • Amazon Keyspaces 中的目標金鑰空間`migration`和`ledger`資料表	AWS DevOps

執行 CQLReplicator

任務	Description	所需的技能
啟動遷移程序。	若要在 AWS Glue 上操作 CQLReplicator，您需要使用`--state run`命令，後面接	AWS DevOps

任務	Description	所需的技能
	著一系列參數。這些參數的精確組態主要取決於您唯一的遷移需求。例如，如果您選擇複寫存留時間 (TTL) 值和更新，或將超過 1 MB 的物件卸載至 Amazon S3，這些設定可能會有所不同。 若要將工作負載從 Cassandra 叢集複寫到 Amazon Keyspaces，請執行下列命令： `./cqlreplicator --state run --tiles 8 \ --landing-zone s3://cql-replicator-1234567890-us-west-2 \ --region us-west-2 \ --src-keyspace source_keyspace \ --src-table source_table \ --trg-keyspace taget_keyspace \ --writetime-column column_name \ --trg-table target_table --inc-traffic` 您的來源金鑰空間和資料表位於 Cassandra 叢	

任務	Description	所需的技能
	集source_keyspace.source_table 中。您的目標金鑰空間和資料表位於 Amazon Keyspaces target_keyspace.target_table 中。參數--inc-traffic 有助於防止增量流量因大量請求而使 Cassandra 叢集和 Amazon Keyspaces 超載。 若要複寫更新，請將 --writetime-column regular_column_name 新增至命令列。一般資料欄將用作寫入時間戳記的來源。	

監控遷移程序

任務	Description	所需的技能
在歷史遷移階段驗證已遷移的 Cassandra 資料列。	若要取得回填階段期間複寫的資料列數，請執行下列命令： `./cqlreplicator --state stats \ --landing-zone s3://cql-replicator-1234567890-us-west-2 \ --src-keyspace source_keyspace --src-table source_table --region us-west-2`	AWS DevOps

停止遷移程序

任務	Description	所需的技能
使用 `cqlreplicator` 命令或 AWS Glue 主控台。	若要正常停止遷移程序，請執行下列命令： `./cqlreplicator --state request-stop --tiles 8 \ --landing-zone s3://cql-replicator-1234567890-us-west-2 \ --region us-west-2 \ --src-keyspace source_keyspace --src-table source_table` 若要立即停止遷移程序，請使用 AWS Glue 主控台。	AWS DevOps

清除

任務	Description	所需的技能
刪除已部署的資源。	下列命令會刪除 AWS Glue 任務、連接器、S3 儲存貯體和 Keyspaces 資料表 `ledger`： `./cqlreplicator --state cleanup --landing-zone s3://cql-replicato`	AWS DevOps

任務	Description	所需的技能
	`r-1234567890-us-we st-2`	

疑難排解

問題	解決方案
AWS Glue 任務失敗，並傳回記憶體不足 (OOM) 錯誤。	1. 變更工作者類型（向上擴展）。例如，`G0.25X`將 變更為 `G.1X`或 `G.1X`。 `G.2X`或者，增加 CQLReplicator 中每個 AWS Glue 任務（向外擴展） DPUs 數目。 2. 從中斷處開始遷移程序。若要重新啟動失敗的 CQLReplicator 任務，請使用相同的參數重新執行`--state run`命令。

相關資源

- CQLReplicator 搭配 AWS Glue https：//README.MD
- AWS Glue 文件
- Amazon Keyspaces 文件
- Apache Cassandra

其他資訊

遷移考量

您可以使用 AWS Glue 將 Cassandra 工作負載遷移至 Amazon Keyspaces，同時在遷移過程中保持 Cassandra 來源資料庫完全正常運作。複寫完成後，您可以選擇在 Cassandra 叢集和 Amazon Keyspaces 之間，以最小的複寫延遲（不到 分鐘）將應用程式切換到 Amazon Keyspaces。若要維持資料一致性，您也可以使用類似的管道，將資料從 Amazon Keyspaces 複寫回 Cassandra 叢集。

寫入單位計算

例如，假設您打算在一小時內寫入 500，000，000 且資料列大小為 1 KiB。您需要的 Amazon Keyspaces 寫入單位 WCUs) 總數是根據此計算：

```
(number of rows/60 mins 60s) 1 WCU per row = (500,000,000/(60*60s) * 1 WCU) = 69,444 WCUs required
```

每秒 69，444 WCUs 是 1 小時的速率，但您可以為額外負荷新增一些緩衝。 例如， `69,444 * 1.10 = 76,388` WCUs有 10% 的額外負荷。

使用 CQL 建立金鑰空間

若要使用 CQL 建立金鑰空間，請執行下列命令：

```
CREATE KEYSPACE target_keyspace WITH replication = {'class': 'SingleRegionStrategy'}
CREATE TABLE target_keyspace.target_table ( userid uuid, level text, gameid int,
 description text, nickname text, zip text, email text, updatetime text, PRIMARY KEY
 (userid, level, gameid) ) WITH default_time_to_live = 0 AND CUSTOM_PROPERTIES =
 {'capacity_mode':{ 'throughput_mode':'PROVISIONED', 'write_capacity_units':76388,
 'read_capacity_units':3612 }} AND CLUSTERING ORDER BY (level ASC, gameid ASC)
```

從內部部署伺服器將 Oracle Business Intelligence 12c 遷移至 AWS 雲端

Lanre (Lan-Ray) showunmi 和 Patrick Yellow，Amazon Web Services

總結

此模式說明如何使用 AWS CloudFormation，將 Oracle Business Intelligence Enterprise Edition 12c 從內部部署伺服器遷移至 AWS 雲端。 AWS CloudFormation 它也說明如何使用其他 AWS 服務來實作 Oracle BI 12c 元件，以提供高可用性、安全性、彈性和動態擴展的能力。

如需將 Oracle BI 12c 遷移至 AWS 雲端的相關最佳實務清單，請參閱此模式的其他資訊一節。

> **Note**
>
> 最佳實務是先執行多個測試遷移，再將現有的 Oracle BI 12c 資料傳輸至雲端。這些測試可協助您微調遷移方法、識別和修正潛在問題，以及更準確地預估停機時間需求。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 透過 AWS Virtual Private Network (AWS VPN) 服務或 AWS Direct Connect，保護內部部署伺服器與 AWS 之間的網路連線
- Oracle 作業系統、Oracle BI 12c、Oracle Database、Oracle WebLogic Server 和 Oracle HTTP Server 的軟體授權

限制

如需儲存大小限制的資訊，請參閱適用於 Oracledocumentation 的Amazon Relational Database Service (Amazon RDS)。

產品版本

- Oracle Business Intelligence Enterprise Edition 12c

- Oracle WebLogic Server 12c
- Oracle HTTP 伺服器 12c
- Oracle 資料庫 12c （或更新版本）
- Oracle Java SE 8

Architecture

下圖顯示在 AWS 雲端中執行 Oracle BI 12c 元件的範例架構：

此圖表顯示下列架構：

1. Amazon Route 53 提供網域名稱服務 (DNS) 組態。
2. Elastic Load Balancing (ELB) 會分配網路流量，以改善 Oracle BI 12c 元件跨多個可用區域的可擴展性和可用性。
3. Amazon Elastic Compute Cloud (Amazon EC2) Auto Scaling 群組跨多個可用區域託管 Oracle HTTP 伺服器、Weblogic Admin 伺服器和受管 BI 伺服器。
4. Oracle 資料庫的 Amazon Relational Database Service (Amazon RDS) 會跨多個可用區域存放 BI Server 中繼資料。
5. Amazon Elastic File System (Amazon EFS) 會掛載到每個 Oracle BI 12c 元件以進行共用檔案儲存。

技術堆疊

- Amazon Elastic Block Store (Amazon EBS)
- Amazon Elastic Compute Cloud (Amazon EC2)
- Amazon Elastic File System (Amazon EFS)
- Amazon RDS for Oracle
- AWS Certificate Manager (ACM)
- Elastic Load Balancing (ELB)
- Oracle BI 12c
- Oracle WebLogic Server 12c
- Oracle HTTP 伺服器 (OHS)

工具

- AWS CloudFormation 可協助您設定 AWS 資源、快速一致地佈建資源，以及在整個 AWS 帳戶和區域的生命週期進行管理。
- AWS Certificate Manager (ACM) 可協助您建立、存放和續約公有和私有 SSL/TLS X.509 憑證和金鑰，以保護 AWS 網站和應用程式。
- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，並快速向上或向下擴展。
- Amazon EC2 Auto Scaling 可協助您維持應用程式可用性，並可讓您根據您定義的條件自動新增或移除 Amazon EC2 執行個體。
- Amazon Elastic File System (Amazon EFS) 可協助您在 AWS 雲端中建立和設定共用檔案系統。
- Elastic Load Balancing 會將傳入的應用程式或網路流量分散到多個目標。例如，您可以在一或多個可用區域中跨 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、容器和 IP 地址分配流量。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路類似於您在自己的資料中心內操作的傳統網路，具有使用可擴展的 AWS 基礎設施的優勢。
- Oracle Data Pump 可協助您以高速將資料和中繼資料從一個資料庫移至另一個資料庫。
- Oracle Fusion Middleware 是一組應用程式開發工具和整合解決方案，可用於身分管理、協同合作和商業智慧報告。
- Oracle GoldenGate 可協助您在 Oracle Cloud Infrastructure 中設計、執行、協調和監控資料複寫和串流資料處理解決方案。
- Oracle WebLogic 指令碼工具 (WLST) 提供命令列界面，可協助您水平擴展 WebLogic 叢集。

史詩

評估來源環境

任務	Description	所需的技能
收集軟體庫存資訊。	識別每個來源技術堆疊軟體元件的版本和修補程式層級，包括下列項目： • Oracle 作業系統 • Oracle Database • Oracle BI 12c • Oracle WebLogic 伺服器 • Oracle HTTP 伺服器 • Java	Migration Architect、解決方案架構師、應用程式擁有者、Oracle BI 管理員
收集運算和儲存庫存資訊。	在您的來源環境中，檢閱下列項目的目前和歷史使用率指標： • CPU 用量 • 記憶體用量 • 儲存體用量 **Important** 請務必考慮用量的歷史尖峰。	遷移架構師、解決方案架構師、應用程式擁有者、Oracle BI 管理員、系統管理員
收集來源環境架構及其需求的相關資訊。	充分了解來源環境的架構及其需求，包括對下列項目的了解： • Oracle WebLogic Server 網域組態	Migration Architect、解決方案架構師、應用程式擁有者、Oracle BI 管理員

任務	Description	所需的技能
	• 叢集 • Load balancing • 連線能力 • 可用性 • 災難復原要求	
識別 Java Database Connectivity (JDBC) 資料來源。	收集來源環境 JDBC 資料來源及其使用之每個資料庫引擎的驅動程式的相關資訊。	遷移架構師、應用程式擁有者、Oracle BI 管理員、資料庫工程師或管理員
收集環境特定設定的相關資訊。	收集您的來源環境特有的設定和組態相關資訊，包括下列項目： • 自訂啟動和關閉指令碼 • Java 和其他環境變數 • 憑證	Migration Architect、解決方案架構師、應用程式擁有者、Oracle BI 管理員
識別其他應用程式的任何相依性。	收集來源環境中整合的相關資訊，以與其他應用程式建立相依性。 **Important** 請務必識別任何輕量型目錄存取通訊協定 (LDAP) 整合和其他聯網需求。	Migration Architect、解決方案架構師、應用程式擁有者、Oracle BI 管理員

設計您的目標環境

任務	Description	所需的技能
建立高階設計文件。	建立目標架構設計文件。請務必使用您在評估來源環境時所收集的資訊來通知設計文件。	解決方案架構師、應用程式架構師、資料庫工程師、遷移架構師
取得設計文件的核准。	與利益相關者一起檢閱設計文件，並取得必要的核准。	應用程式或服務擁有者、解決方案架構師、應用程式架構師

部署基礎設施

任務	Description	所需的技能
在 CloudFormation 中準備基礎設施程式碼。	建立 CloudFormation 範本以在 AWS 雲端中佈建 Oracle BI 12c 基礎設施。 如需詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的使用 AWS CloudFormation 範本。 **Note** 最佳實務是為每個 Oracle BI 12c 層建立模組化 CloudFormation 範本，而不是為所有資源建立大型範本。如需 CloudFormation 最佳實務的詳細資訊，請參閱 AWS 部落格上的使用 AWS CloudFormation 自動	Cloud Infrastructure Achitect，解決方案架構師，應用程式架構師

任務	Description	所需的技能
	化部署時的 8 個最佳實務。	
下載必要的軟體。	從 Oracle 網站DDownload下列軟體以及所需的版本和修補程式： • Java JDK8 • Oracle WebLogic Server 12c • Oracle BI 12c	遷移架構師、資料庫工程師、應用程式架構師
準備安裝指令碼。	建立執行無訊息安裝的軟體安裝指令碼。這些指令碼可簡化部署自動化。 如需詳細資訊，請參閱 Oracle Support 網站上的 OBIEE 12c：如何執行無提示安裝？。您需要 Oracle Support 帳戶才能檢視文件。	遷移架構師、資料庫工程師、應用程式架構師

任務	Description	所需的技能
為您的 Web 和應用程式層建立 Amazon EBS 支援的 Linux AMI。	1. 部署和設定 Web 和應用程式層的 Amazon EC2 執行個體。確定執行個體符合執行下列項目的先決條件： • Oracle 作業系統環境設定 • Oracle 作業系統使用者帳戶設定 • Java 軟體安裝 2. 建立執行個體的 Amazon Machine Image AMIs)，並儲存複本以供日後使用。如需說明，請參閱《Amazon EC2 Linux 執行個體使用者指南》中的建立 Amazon EBS 支援的 Linux AMI。Amazon EC2	遷移架構師、資料庫工程師、應用程式架構師
使用 CloudFormation 啟動您的 AWS 基礎設施。	使用您建立的 CloudFormation 範本，在模組中部署 Oracle BI 12c Web 和應用程式層。 如需說明，請參閱《AWS CloudFormation 使用者指南》中的 AWS CloudFormation 入門。	雲端基礎設施架構師、解決方案架構師、應用程式架構師

使用全新安裝將 Oracle BI 12c 遷移至 AWS

任務	Description	所需的技能
暫存所需的軟體。	將必要的軟體放置在可供 Amazon EC2 執行個體存取的位置。例如，您可以在 Amazon S3 或另一個可供	Migration Architect， Oracle BI Architect， Cloud Infrastructure Achitect， Solutions

任務	Description	所需的技能
	Web 和應用程式伺服器存取的 Amazon EC2 執行個體中暫存軟體。	Architect， Application Architect
準備您的儲存庫資料庫以進行 Oracle BI 12c 安裝。	透過在新的 Amazon RDS for Oracle 資料庫執行個體上執行 Oracle 儲存庫建立公用程式 (RCU) 來建立 Oracle BI 12c 結構描述。 https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_Oracle.html	Cloud Infrastructure Achitect、解決方案架構師、應用程式架構師、遷移架構師、Oracle BI 架構師

任務	Description	所需的技能
安裝 Oracle Fusion Middleware 12c 和 Oracle BI 12c。	1. 從一個 Amazon EC2 執行個體開始，安裝 Oracle Fusion Middleware 12c 基礎設施和 OBIEE 12c。如需詳細資訊，請參閱 Oracle Business Intelligence 的 Oracle Fusion Middleware Enterprise 部署指南中的下列章節： • 在 BIHOST1 上啟動基礎設施安裝程式 • 安裝 Oracle Business Intelligence 以準備企業部署 **Note** 使用 Amazon EFS 託管將在 Oracle BI 12c 叢集節點之間共用的目錄。 2. 將任何必要的修補程式套用至 安裝。 3. 建立執行個體AMIs，並儲存複本以供日後使用。	Migration Architect，Oracle BI Architect
設定 Oracle BI 12c 的 Oracle WebLogic Server 網域。	將您的 Oracle BI 12c 網域設定為非叢集部署。 如需詳細資訊，請參閱《Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Business Intelligence》中的設定 BI 網域。	Migration Architect，Oracle BI Architect

任務	Description	所需的技能
從 Oracle BI 12c 執行水平擴展。	水平擴展單一節點至所需的節點數量。 如需詳細資訊，請參閱《Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Business Intelligence》中的擴展 Oracle Business Intelligence。	Migration Architect，Oracle BI Architect
安裝 Oracle HTTP Server 12c。	1. 在 Oracle Web 層 Amazon EC2 執行個體上安裝 Oracle HTTP Server 12c。如需說明，請參閱安裝和設定 Oracle Access Management 12c 的安裝 Oracle HTTP Server 12c。 2. 將任何必要的修補程式套用至 安裝。 3. 建立執行個體AMIs，並儲存複本以供日後使用。	Migration Architect，Oracle BI Architect
設定 SSL 終止的負載平衡器。	1. 在 ACM 中建立或匯入 SSL 憑證。 2. 將 SSL 憑證與 ELB 建立關聯。	雲端基礎設施架構師、遷移架構師

任務	Description	所需的技能
將商業智慧中繼資料成品遷移至 AWS。	1. 從內部部署 Oracle BI 12c 安裝匯出 Oracle Business Intelligence Application Archive (BAR) 檔案。若要匯出 BAR 檔案，請使用 WebLogic 指令碼工具 (WLST) 來執行 exportServiceInstance 命令。 2. 將內部部署 BAR 檔案匯入 AWS Oracle BI 12c 安裝。若要匯入 BAR 檔案，請執行 importServiceInstance WLST 命令。	Migration Architect，Oracle BI Architect
執行遷移後任務。	匯入 BAR 檔案之後，請執行下列動作： • 設定任何其他 JDBC 資料來源。 • 為 PostgreSQL 或 Amazon Redshift 等其他資料來源安裝驅動程式。 • 設定 Oracle LDAP、SSL、單一登入 (SSO) 和 WebLogic 安全存放區。 • 設定 AWS Identity and Access Management (IAM) 政策。 • 啟用用量追蹤。 • 設定與其他系統的整合。 • 遷移任何自訂指令碼。	Migration Architect，Oracle BI Architect

測試新環境

任務	Description	所需的技能
測試新的 Oracle BI 12c 環境。	在新的 Oracle BI 12c 環境上執行end-to-end測試。盡可能使用自動化。 測試活動的範例包括下列項目： • 驗證儀表板、報告和 URLs • 使用者接受度測試 (UAT) • 操作接受測試 (OAT) **Note** 視需要執行額外的測試和驗證。	Migration Architect、解決方案架構師、應用程式擁有者、Oracle BI 管理員

切換到新環境

任務	Description	所需的技能
中斷內部部署 Oracle BI 12c 環境的流量。	在指定的切換視窗中，停止所有流向內部部署 Oracle BI 12c 環境的流量。	Migration Architect、解決方案架構師、應用程式擁有者、Oracle BI 管理員
重新同步新的 Oracle BI 12c 儲存庫資料庫與來源資料庫。	重新同步 Amazon RDS Oracle BI 12c 儲存庫資料庫與內部部署資料庫。 若要同步資料庫，您可以使用 Oracle Data Pump 重新整理或 AWS DMS 變更資料擷取 (CDC)。	Oracle BI 管理員、資料庫工程師/管理員

任務	Description	所需的技能
切換您的 Oracle BI 12c URLs 以指向新的 AWS 環境。	更新內部 DNS 伺服器上的 Oracle BI 12c URLs，使其指向新的 AWS 安裝。	Migration Architect、解決方案架構師、應用程式擁有者、Oracle BI 管理員
監控新環境。	使用下列任一工具監控新的 Oracle BI 12c 環境： • Amazon CloudWatch • Amazon RDS Performance Insights • Oracle Enterprise Manager	Oracle BI 管理員、資料庫工程師/管理員、應用程式管理員
取得專案的簽署。	與利益相關者一起檢閱測試結果，並取得必要的核准來包裝遷移。	應用程式擁有者、服務擁有者、雲端基礎設施架構師、遷移架構師、Oracle BI 架構師

相關資源

- 在 RDS for Oracle 上使用 Oracle 儲存庫建立公用程式 (Amazon RDS 使用者指南)
- Oracle on Amazon RDS (Amazon RDS 使用者指南)
- AWS 上的 Oracle WebLogic Server 12c (AWS 白皮書)
- 部署 Oracle Business Intelligence 以獲得高可用性 (Oracle 說明中心)
- Oracle Business Intelligence 應用程式封存 (BAR) 檔案 (Oracle 說明中心)
- 如何在環境之間遷移 OBI 12c (Oracle Support)

其他資訊

以下是與將 Oracle BI 12c 遷移至 AWS 雲端相關的最佳實務清單。

儲存庫資料庫

最佳實務是在 Amazon RDS for Oracle 執行個體上託管 Oracle BI 12c 資料庫結構描述。此執行個體類型提供經濟實惠且可調整大小的容量，同時自動化管理任務，例如硬體佈建、資料庫設定、修補和備份。

如需詳細資訊，請參閱《Amazon RDS 使用者指南》中的在 RDS for Oracle 上使用 Oracle 儲存庫建立公用程式。

Web 和應用程式層

記憶體最佳化的 Amazon EC2 執行個體通常非常適合 Oracle BI 12c 伺服器。無論您選擇何種執行個體類型，請確定您佈建的執行個體符合您系統的記憶體使用需求。此外，請確定您根據 Amazon EC2 執行個體的可用記憶體設定足夠的 WebLogic Java 虛擬機器 (JVM) 堆積大小。

本機儲存

I/O 在 Oracle BI 12c 應用程式的整體效能中扮演重要角色。Amazon Elastic Block Store (Amazon EBS) 提供針對不同工作負載模式最佳化的不同儲存類別。請務必選擇適合您使用案例的 Amazon EBS 磁碟區類型。

如需 EBS 磁碟區類型的詳細資訊，請參閱 Amazon EBS 文件中的 Amazon EBS 功能。

共用儲存

叢集 Oracle BI 12c 網域需要下列資源的共用儲存：

- 組態檔案
- Oracle BI 12c 單一資料目錄 (SDD)
- Oracle 全域快取
- Oracle BI 排程器指令碼
- Oracle WebLogic Server 二進位檔

您可以使用Amazon EFS 來滿足此共用儲存需求，該 EFS 提供可擴展、全受管的彈性網路檔案系統 (NFS) 檔案系統。

微調共用儲存效能

Amazon EFS 有兩種輸送量模式：佈建和爆量。服務也有兩種效能模式：一般用途和最大 I/O。

若要微調效能，請先在一般用途效能模式和佈建輸送量模式下測試工作負載。執行這些測試可協助您判斷這些基準模式是否足以滿足所需的服務水準。

如需詳細資訊，請參閱《Amazon EFS 使用者指南》中的 Amazon EFS 效能。 EFS

可用性和災難復原

最佳實務是在多個可用區域部署 Oracle BI 12c 元件，以便在可用區域故障時保護這些資源。以下是 AWS 雲端中託管之特定 Oracle BI 12c 資源的可用性和災難復原最佳實務清單：

- Oracle BI 12c 儲存庫資料庫：將多可用區域 Amazon RDS 資料庫執行個體部署至 Oracle BI 12crepository 資料庫。在多可用區域部署中，Amazon RDS 會自動在不同的可用區域中佈建和維護同步待命複本。跨可用區域執行 Oracle BI 12c 儲存庫資料庫執行個體，可以在計劃的系統維護期間增強可用性，並協助保護您的資料庫免於執行個體和可用區域故障。
- Oracle BI 12c 受管伺服器：為實現容錯能力，最佳實務是在設定為跨越多個可用區域的 Amazon EC2 Auto Scaling 群組中的受管伺服器上部署 Oracle BI 12c 系統元件。 Auto Scaling Auto Scaling 會根據Amazon EC2 運作狀態檢查取代故障的執行個體。如果發生可用區域故障，Oracle HTTP 伺服器會繼續將流量導向正常運作可用區域中的受管伺服器。然後，Auto Scaling 會啟動執行個體，以符合您的主機計數需求。建議啟用 HTTP 工作階段狀態複寫，以協助確保現有工作階段順利容錯移轉至正常運作的受管伺服器。
- Oracle BI 12c 管理伺服器：為了確保您的管理伺服器具有高可用性，請將它託管在設定為跨越多個可用區域的 Amazon EC2 Auto Scaling 群組中。然後，將群組的最小和最大大小設定為 1。如果發生可用區域故障，Amazon EC2 Auto Scaling 會在替代可用區域中啟動替代的管理伺服器。若要復原相同可用區域內任何失敗的基礎主機，您可以啟用Amazon EC2 Auto Recovery。
- Oracle Web 層伺服器：最佳實務是將 Oracle HTTP 伺服器與 Oracle WebLogic Server 網域建立關聯。為了獲得高可用性，請將 Oracle HTTP 伺服器部署在設定為擔任多個可用區域的 Amazon EC2 Auto Scaling 群組中。然後，將伺服器放在 ELB 彈性負載平衡器後方。若要提供額外的主機故障保護，您可以啟用 Amazon EC2 Auto Recovery。

可擴展性

AWS 雲端的彈性可協助您水平或垂直擴展應用程式，以回應工作負載需求。

垂直擴展

若要垂直擴展應用程式，您可以變更執行 Oracle BI 12c 元件的 Amazon EC2 執行個體的大小和類型。您不需要在部署開始時過度佈建執行個體，並產生不必要的成本。

水平擴展

Amazon EC2 Auto Scaling 會根據工作負載需求自動新增或移除受管伺服器，協助您水平擴展應用程式。

> **Note**
>
> 使用 Amazon EC2 Auto Scaling 水平擴展需要指令碼編寫技能和徹底測試才能實作。

備份和復原

以下是 AWS 雲端中託管之特定 Oracle BI 12c 資源的備份和復原最佳實務清單：

- Oracle Business Intelligence 中繼資料儲存庫：Amazon RDS 會自動建立和儲存資料庫執行個體的備份。這些備份會保留您指定的一段時間。請務必根據資料保護需求來設定 Amazon RDS 備份持續時間和保留設定。如需詳細資訊，請參閱 Amazon RDS 備份和還原。
- 受管伺服器、管理伺服器和 Web 層伺服器：請確定您根據資料保護和保留需求設定 Amazon EBS 快照。
- 共用儲存：您可以使用 AWS Backup Backup 管理存放在 Amazon EFS 中的檔案的備份和復原。AWS Backup 服務也可以部署，以集中管理其他服務的備份和復原，包括 Amazon EC2、Amazon EBS 和 Amazon RDS。如需詳細資訊，請參閱什麼是 AWS Backup？ 在 AWS Backup 開發人員指南中。

安全性與合規性

以下是安全最佳實務和 AWS 服務的清單，可協助您保護 AWS 雲端中的 Oracle BI 12c 應用程式：

- 靜態加密：Amazon RDS、Amazon EFS 和 Amazon EBS 都支援業界標準加密演算法。您可以使用 AWS Key Management Service (AWS KMS) 來建立和管理密碼編譯金鑰，並控制它們在 AWS 服務和應用程式中的使用。您也可以在託管 Oracle BI 12c 儲存庫資料庫的 Amazon RDS for Oracle 資料庫執行個體上設定 Oracle 透明資料加密 (TDE)。
- 傳輸中加密：最佳實務是啟用 SSL 或 TLS 通訊協定，以保護 Oracle BI 12c 安裝的各個層之間的傳輸中資料。您可以使用 AWS Certificate Manager (ACM) 為您的 Oracle BI 12c 資源佈建、管理和部署公有和私有 SSL 和 TLS 憑證。
- 網路安全：請確定您在已針對使用案例設定適當存取控制的 Amazon VPC 中部署 Oracle BI 12c 資源。設定您的安全群組，以篩選執行安裝之 Amazon EC2 執行個體的傳入和傳出流量。此外，請務必設定網路存取控制清單 (NACLs)，以根據定義的規則允許或拒絕流量。
- 監控和記錄：您可以使用 AWS CloudTrail 追蹤對 AWS 基礎設施的 API 呼叫，包括 Oracle BI 12c 資源。此功能在追蹤基礎設施的變更或執行安全分析時非常有用。您也可以使用 Amazon CloudWatch 來檢視操作資料，讓您深入了解 Oracle BI 12c 應用程式的效能和運作狀態。您也可以

設定警示，並根據這些警示採取自動化動作。Amazon RDS 提供額外的監控工具，包括增強型監控和效能詳情。

將 ELK 堆疊遷移至 AWS 上的彈性雲端

Battulga Purevragchaa 和 Antony Prasad Thevaraj，Amazon Web Services

uday reddy，無

總結

Elastic 已提供服務多年，其使用者和客戶通常會在內部部署中自行管理 Elastic。Elastic Cloud 是受管 Elasticsearch 服務，提供一種使用 Elastic Stack (ELK Stack) 和解決方案的方法，用於企業搜尋、可觀測性和安全性。您可以使用 Logs、Metrics、APM （應用程式效能監控） 和 SIEM （安全資訊和事件管理） 等應用程式來存取彈性解決方案。您可以使用整合功能，例如機器學習、索引生命週期管理、Kibana Lens （用於拖放視覺化）。

當您從自我管理的 Elasticsearch 移至 Elastic Cloud 時，Elasticsearch 服務會處理下列事項：

- 佈建和管理基礎基礎設施
- 建立和管理 Elasticsearch 叢集
- 向上和向下擴展叢集
- 升級、修補和拍攝快照

這可讓您有更多時間專注於解決其他挑戰。

此模式定義如何將內部部署 Elasticsearch 7.13 遷移至 Amazon Web Services (AWS) 上的 Elastic Cloud。其他版本可能需要稍微修改竊賊模式中描述的程序。如需詳細資訊，請聯絡您的 Elastic 代表。

先決條件和限制

先決條件

- 可存取快照的 Amazon Simple Storage Service (Amazon S3) 的作用中 AWS 帳戶
- 安全且足夠高頻寬的私有連結，可將快照資料檔案複製到 Amazon S3
- Amazon S3 Transfer Acceleration
- 彈性快照政策，以確保資料擷取定期封存至足夠大的本機資料存放區或遠端儲存 (Amazon S3)

您必須了解快照和隨附索引的生命週期政策在內部部署中的大小，才能啟動遷移。如需詳細資訊，請聯絡 Elastic。

角色和技能

遷移程序也需要下表所述的角色和專業知識。

Role	專業知識	責任
應用程式支援	熟悉現場部署的 Elastic Cloud 和 Elastic	所有彈性相關任務
系統管理員或 DBA	深入了解現場部署彈性環境及其組態	能夠佈建儲存體、安裝和使用 AWS 命令列界面 (AWS CLI)，並識別提供現場部署彈性的所有資料來源
網路管理員	了解內部部署到 AWS 網路連線能力、安全性和效能	了解連線頻寬，建立從內部部署到 Amazon S3 的網路連結

限制

- Elastic Cloud 上的 Elasticsearch 僅適用於支援的 AWS 區域 (2021 年 9 月) 。

產品版本

- Elasticsearch 7.13

Architecture

來源技術堆疊

內部部署 Elasticsearch 7.13 或更新版本：

- 叢集快照
- 索引快照
- Beats 組態

來源技術架構

下圖顯示具有不同擷取方法、節點類型和 Kibana 的典型內部部署架構。不同的節點類型反映 Elasticsearch 叢集、身分驗證和視覺化角色。

1. 從 Beats 擷取到 Logstash
2. 從 Beats 擷取到 Apache Kafka 訊息佇列
3. 從 Filebeat 擷取至 Logstash
4. 從 Apache Kafka 訊息佇列擷取至 Logstash
5. 從 Logstash 擷取至 Elasticsearch 叢集
6. Elasticsearch 叢集
7. 身分驗證和通知節點
8. Kibana 和 Blob 節點

目標技術堆疊

Elastic Cloud 透過跨叢集複寫部署到您的多個 AWS 區域中的軟體即服務 (SaaS) 帳戶。

- 叢集快照
- 索引快照
- Beats 組態
- 彈性雲端
- Network Load Balancer
- Amazon Route 53
- Amazon S3

目標架構

受管彈性雲端基礎設施為：

- 高可用性，存在於多個可用區域和多個 AWS 區域。
- 區域容錯，因為使用 Elastic Cloudcross-cluster 複寫 (CCR) 複寫資料（索引和快照）
- 封存，因為快照會封存在 Amazon S3 中
- 透過 Network Load Balancer 和 Route 53 的組合，可容忍網路分割區

- 源自（但不限於）Elastic APM、Beats、Logstash 的資料擷取

高階遷移步驟

Elastic 已開發自己的規範方法，將內部部署 Elastic Cluster 遷移至 Elastic Cloud。彈性方法直接符合 AWS 遷移指引和最佳實務，包括 Well-Architected Framework 和 AWS Migration Acceleration Program (MAP)。一般而言，三個 AWS 遷移階段如下：

- 評估
- 調動
- 遷移和現代化

Elastic 遵循具有互補術語的類似遷移階段：

- 啟動
- 計畫
- 實作
- 交付
- Close (關閉)

Elastic 使用彈性實作方法，協助交付專案成果。這在設計上具有包容性，以確保彈性、諮詢團隊和客戶團隊以清晰的方式合作，共同交付預期成果。

彈性方法在實作階段將傳統的瀑布假相與 Scrum 結合。技術需求的組態會以協作方式反覆交付，同時將風險降至最低。

工具

AWS 服務

- Amazon Route 53 – Amazon Route 53 是高可用性且可擴展的網域名稱系統 (DNS) Web 服務。您可以使用 Route 53 執行三個主要功能的任意組合：網域註冊、DNS 路由和運作狀態檢查。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種物件儲存服務。您可以使用 Amazon S3 隨時從 Web 任何地方存放和擷取任意資料量。此模式使用 S3 儲存貯體和 Amazon S3 Transfer Acceleration。

- Elastic Load Balancing – Elastic Load Balancing 會自動將傳入流量分散到一或多個可用區域中的多個目標，例如 EC2 執行個體、容器和 IP 地址。

其他工具

- Beats – Beats 從 Logstash 或 Elasticsearch 運送資料
- Elastic Cloud – Elastic Cloud 是一種託管 Elasticsearch 的受管服務。
- Elasticsearch – Elasticsearch 是一種搜尋和分析引擎，使用 Elastic Stack 集中存放您的資料，以進行可擴展的搜尋和分析。此模式也會使用快照建立和跨叢集複寫。
- Logstash – Logstash 是一種伺服器端資料處理管道，可從多個來源擷取資料、轉換資料，然後將其傳送至您的資料儲存體。

史詩

準備遷移

任務	Description	所需的技能
識別執行內部部署彈性解決方案的伺服器。	確認支援彈性遷移。	應用程式擁有者
了解內部部署伺服器組態。	若要了解在內部部署成功驅動工作負載所需的伺服器組態，請尋找目前正在使用的伺服器硬體足跡、網路組態和儲存特性	應用程式支援
收集使用者和應用程式帳戶資訊。	識別內部部署彈性環境所使用的使用者名稱和應用程式名稱。	系統管理員、應用程式支援
文件 Beats 和資料寄件人組態。	若要記錄組態，請查看現有的資料來源和目的地。如需詳細資訊，請參閱彈性文件。	應用程式支援
判斷資料的速度和數量。	建立叢集正在處理多少資料的基準。	系統管理員、應用程式支援

任務	Description	所需的技能
記錄 RPO 和 RTO 案例。	有關中斷和服務水準協議 (SLAs) 的文件復原點目標 (RPO) 和復原時間目標 (RTO) 案例。	應用程式擁有者、系統管理員、應用程式支援
決定最佳快照生命週期設定。	定義在遷移期間和之後使用彈性快照保護資料的頻率。	應用程式擁有者、系統管理員、應用程式支援
定義遷移後效能期望。	針對目前和預期的畫面重新整理、查詢執行期和使用者介面行為產生指標。	系統管理員、應用程式支援
記錄網際網路存取傳輸、頻寬和可用性需求。	確定將快照複製到 Amazon S3 的網際網路連線速度、延遲和彈性。	網路管理員
記錄 Elastic 內部部署執行時間的目前成本。	確保 AWS 目標環境的大小設計為高效能且符合成本效益。	DBA、系統管理員、應用程式支援
識別身分驗證和授權需求。	Elastic Stack 安全功能提供內建領域，例如輕量型目錄存取通訊協定 (LDAP)、安全聲明標記語言 (SAML) 和 OpenID Connect (OIDC)。	DBA、系統管理員、應用程式支援
根據地理位置了解特定法規要求。	確保資料根據您的要求和任何相關的國家要求匯出和加密。	DBA、系統管理員、應用程式支援

實作遷移

任務	Description	所需的技能
在 Amazon S3 上準備預備區域。	若要在 Amazon S3 上接收快照，請建立 S3 儲存貯體和具有新建立儲存貯體完整存取權的臨時 AWS Identity and	AWS 管理員

任務	Description	所需的技能
	Access Management (IAM) 角色。如需詳細資訊，請參閱建立角色以將許可委派給 IAM 使用者。使用 AWS Security Token Service 請求臨時安全登入資料。保護存取金鑰 ID、私密存取金鑰和工作階段字符的安全。 在儲存貯體上啟用 Amazon S3 Transfer Acceleration。	
在內部部署安裝 AWS CLI 和 Amazon S3 外掛程式。	在每個 Elasticsearch 節點上執行下列命令。 `sudo bin/elasticsearch-plugin install repository-s3` 然後重新啟動節點。	AWS 管理員

設定 Amazon S3 用戶端存取。	執行下列命令來新增先前建立的金鑰。
	
	`elasticsearch-keystore add s3.client.default.access_key`
	
	`elasticsearch-keystore add s3.client.default.secret_key`
	
	`elasticsearch-keystore add s3.client.default.session_token`
		AWS 管理員

任務	Description	所需的技能
註冊彈性資料的快照儲存庫	使用 Kibana 開發工具來告知內部部署本機叢集要寫入哪個遠端 S3 儲存貯體。	AWS 管理員
設定快照政策。	若要設定快照生命週期管理，請在 Kibana 政策索引標籤上，選擇 SLM 政策，並定義應包含的時間、資料串流或索引，以及要使用的名稱。 設定經常拍攝快照的政策。快照是增量式的，可有效利用儲存體。符合您的整備評估決策。政策也可以指定保留政策，並在不再需要快照時自動刪除快照。	應用程式支援
驗證快照是否正常運作。	在 Kibana 開發工具中，執行下列命令。 `GET _snapshot/<your_repo_name>/_all`	AWS 管理員、應用程式支援、
在 Elastic Cloud 上部署新的叢集。	登入 Elastic，並從準備度評估中的業務調查結果選擇「可觀測性、搜尋或安全性」叢集。	AWS 管理員、應用程式支援
設定叢集金鑰存放區存取權。	新叢集需要存取將存放快照的 S3 儲存貯體。在 Elasticsearch 服務主控台上，選擇安全性，然後輸入您先前建立的存取和秘密 IAM 金鑰。	AWS 管理員

任務	Description	所需的技能
設定 Elastic Cloud 託管叢集以存取 Amazon S3。	在 Amazon S3 中設定對先前建立的快照儲存庫的新叢集存取權。使用 Kibana，執行下列動作： 1. 選擇堆疊管理、快照設定、RegisterRepo。 2. 在別名欄位中，輸入儲存庫的名稱。 3. 針對 S3 用戶端名稱，選擇次要。 4. 將您先前建立的 S3 儲存貯體新增至儲存庫。 5. 選擇壓縮快照。 6. 對於加密設定，請保留預設值。	AWS 管理員、應用程式支援
驗證新的 Amazon S3 儲存庫。	請確定您可以存取 Elastic Cloud 叢集中託管的新儲存庫。	AWS 管理員

任務	Description	所需的技能

初始化 Elasticsearch 服務叢集。	在 Elasticsearch Service Console 上，從 S3 快照初始化 Elasticsearch 服務叢集。
	
	執行下列命令做為 POST。
	
	`*/_close?expand_wildcards=all`
	
	`/_snapshot/<your-repo-name>/ <your-snapshot-name>/_restore`
	
	`*/_open?expand_wildcards=all`
		應用程式支援

完成遷移

任務	Description	所需的技能
驗證快照還原是否成功。	使用 Kibana 開發工具，執行下列命令。 `GET _cat/indices`	應用程式支援
Redploy 擷取服務。	將 Beats 和 Logstash 的端點連接到新的 Elasticsearch 服務端點。	應用程式支援

測試叢集環境並清除

任務	Description	所需的技能
驗證叢集環境。	現場部署彈性叢集環境遷移至 AWS 之後，您就可以連線到該環境，並使用自己的使用者接受度測試 (UAT) 工具來驗證新環境。	應用程式支援
清除資源。	驗證叢集已成功遷移後，請移除 S3 儲存貯體和用於遷移的 IAM 角色。	AWS 管理員

相關資源

彈性參考

- 彈性雲端
- AWS 上的 Managed Elasticsearch 和 Kibana
- 彈性企業搜尋
- 彈性整合
- 彈性可觀測性
- 彈性安全性
- 打擊
- 彈性 APM
- 遷移至索引生命週期管理
- 彈性訂閱
- 聯絡彈性

彈性部落格文章

- 如何從自我管理的 Elasticsearch 遷移至 AWS 上的 Elastic Cloud （部落格文章）
- 遷移至 Elastic Cloud （部落格文章）

彈性文件

- 教學課程：使用 SLM 自動化備份
- ILM：管理索引生命週期
- Logstash
- 跨叢集複寫 (CCR)
- 擷取管道
- 執行 Elasticsearch API 請求
- 快照保留

彈性影片和網路研討會

- 彈性雲端遷移
- Elastic Cloud：為什麼客戶要遷移（網路研討會）

AWS 參考

- AWS Marketplace 上的彈性雲端
- AWS 命令列界面
- AWS Direct Connect
- Migration Acceleration Program (MAP)
- Network Load Balancer
- 區域與可用區域
- Amazon Route 53
- Amazon Simple Storage Service
- Amazon S3 Transfer Acceleration
- VPN 連線
- Well-Architected 架構

其他資訊

如果您打算遷移複雜的工作負載，請聯絡 Elastic Consulting Services。如果您有與組態和服務相關的基本問題，請聯絡 Elastic Support 團隊。

AWS 雲端 使用 Starburst 將資料遷移至

Antony Prasad Thevaraj 和 Suresh Vee Mirrori，Amazon Web Services

Shaun Van Staden，無

總結

Starburst 透過提供企業查詢引擎，在單一存取點中將現有資料來源整合在一起，協助加速資料遷移至 Amazon Web Services (AWS) 的旅程。您可以在完成任何遷移計畫之前，跨多個資料來源執行分析，以取得寶貴的洞見。在不中斷business-as-usual分析的情況下，您可以使用 Starburst 引擎或專用擷取、轉換和載入 (ETL) 應用程式來遷移資料。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 虛擬私有雲端 (VPC)
- Amazon Elastic Kubernetes Service (Amazon EKS) 叢集
- Amazon Elastic Compute Cloud (Amazon EC2) Auto Scaling 群組
- 需要遷移的目前系統工作負載清單
- 從 AWS 到內部部署環境的網路連線

Architecture

參考架構

下列高階架構圖顯示 中 Starburst Enterprise 的典型部署 AWS 雲端：

1. Starburst Enterprise 叢集會在您的 中執行 AWS 帳戶。
2. 使用者使用輕量型目錄存取協定 (LDAP) 或開放授權 (OAuth) 進行身分驗證，並直接與 Starburst 叢集互動。
3. Starburst 可以連線到數個 AWS 資料來源，例如 Amazon Simple Storage Service (Amazon S3) AWS Glue、Amazon Relational Database Service (Amazon RDS) 和 Amazon Redshift。Starburst 可在 、 AWS 雲端內部部署或其他雲端環境中跨資料來源提供聯合查詢功能。

4. 您可以使用 Helm Chart 在 Amazon EKS 叢集中啟動 Starburst Enterprise。
5. Starburst Enterprise 使用 Amazon EC2 Auto Scaling 群組和 Amazon EC2 Spot 執行個體來最佳化基礎設施。
6. Starburst Enterprise 會直接連線到現有的內部部署資料來源，以即時讀取資料。此外，如果您在此環境中有現有的 Starburst Enterprise 部署，則可以直接將 中的新 Starburst 叢集 AWS 雲端 連接到此現有的叢集。

請注意以下內容：

- Starburst 不是資料虛擬化平台。它是一種以 SQL 為基礎的大規模平行處理 (MPP) 查詢引擎，構成用於分析的整體資料網格策略的基礎。
- 當 Starburst 部署為遷移的一部分時，它可以直接連線至現有的內部部署基礎設施。
- Starburst 提供數個內建企業和開放原始碼連接器，可促進與各種舊版系統的連線。如需連接器及其功能的完整清單，請參閱 Starburst Enterprise 使用者指南中的連接器。
- Starburst 可以從內部部署資料來源即時查詢資料。這可防止資料遷移時中斷一般業務操作。
- 如果您要從現有的現場部署 Starburst Enterprise 部署遷移，您可以使用特殊連接器 Starburst Stargate，將 中的 Starburst Enterprise 叢集 AWS 直接連接到您的現場部署叢集。當商業使用者和資料分析師將查詢從 聯合 AWS 雲端 到內部部署環境時，這提供額外的效能優勢。

高階程序概觀

您可以使用 Starburst 來加速資料遷移專案，因為 Starburst 會在遷移所有資料之前啟用洞見。下圖顯示使用 Starburst 遷移資料的一般程序。

Roles (角色)

使用 Starburst 完成遷移通常需要下列角色：

- 雲端管理員 – 負責讓雲端資源可用於執行 Starburst Enterprise 應用程式
- Starburst 管理員 – 負責安裝、設定、管理和支援 Starburst 應用程式
- 資料工程師 – 負責：
  - 將舊版資料遷移至雲端
  - 建置語意檢視以支援分析

- 解決方案或系統擁有者 – 負責整體解決方案實作

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 上執行 Kubernetes， AWS 而無需安裝或維護您自己的 Kubernetes 控制平面或節點。

其他工具

- Helm – Helm 是 Kubernetes 的套件管理員，可協助您在 Kubernetes 叢集上安裝和管理應用程式。
- Starburst Enterprise – Starburst Enterprise 是以 SQL 為基礎的大量平行處理 (MPP) 查詢引擎，構成分析整體資料網格策略的基礎。
- Starburst Stargate – Starburst Stargate 會將一個 Starburst Enterprise 環境中的目錄和資料來源，例如內部部署資料中心的叢集，連結至另一個 Starburst Enterprise 環境中的目錄和資料來源，例如 中的叢集 AWS 雲端。

史詩

評估資料

任務	Description	所需的技能
識別您的資料並排定其優先順序。	識別您要移動的資料。大型內部部署舊版系統可以包含您想要與您不想要移動或由於合規原因而無法移動的資料一起遷移的核心資料。從資料庫存開始，可協助您排定應該優先鎖定哪些資料的優先順序。如需詳細資訊，請參閱開始使用自動化產品組合探索。	資料工程師，DBA

任務	Description	所需的技能
探索、清查和備份您的資料。	驗證使用案例資料的品質、數量和相關性。視需要備份或建立資料的快照，並完成資料的目標環境。	資料工程師，DBA

設定 Starburst Enterprise 環境

任務	Description	所需的技能
在 中設定 Starburst Enterprise AWS 雲端。	當資料編製目錄時，請在受管 Amazon EKS 叢集中設定 Starburst Enterprise。如需詳細資訊，請參閱 Starburst Enterprise 參考文件中的使用 Kubernetes 部署。這允許在資料遷移過程中進行business-as-usual分析。	AWS 管理員、應用程式開發人員
將 Starburst 連接到資料來源。	在您識別資料並設定 Starburst Enterprise 之後，請將 Starburst 連線到資料來源。Starburst 會直接從資料來源讀取資料做為 SQL 查詢。如需詳細資訊，請參閱 Starburst Enterprise 參考文件。	AWS 管理員、應用程式開發人員

遷移資料

任務	Description	所需的技能
建置並執行 ETL 管道。	開始資料遷移程序。此活動可以與business-as-usual分析同時進行。對於遷移，您可以使用第三方產品或 Starburst	資料工程師

任務	Description	所需的技能
	。Starburst 能夠跨不同來源讀取和寫入資料。如需詳細資訊，請參閱 Starburst Enterprise 參考文件。	
驗證資料。	遷移資料之後，請驗證資料，以確保所有必要的資料都已移動且完好無損。	資料工程師、DevOps 工程師

切換

任務	Description	所需的技能
剪下資料。	資料遷移和驗證完成後，您可以切換資料。這涉及變更 Starburst 中的資料連線連結。您可以指向新的雲端來源並更新語意檢視，而不是指向內部部署來源。如需詳細資訊，請參閱 Starburst Enterprise 參考文件中的連接器。	資料工程師，Cutover 主管
向使用者推出。	資料消費者開始處理遷移的資料來源。分析最終使用者看不到此程序。	Cutover 潛在客戶，資料工程師

相關資源

AWS Marketplace

- Starburst Galaxy
- Starburst Enterprise
- Starburst Data JumpStart
- Starburst Enterprise 與 Graviton

Starburst 文件

- Starburst Enterprise 使用者指南
- Starburst Enterprise 參考文件

其他 AWS 文件

- 開始使用自動化產品組合探索 (AWS 方案指引)
- 使用 Starburst on 最佳化雲端基礎設施成本和效能 AWS (AWS 部落格文章)

在 AWS 上最佳化輸入檔案大小的 ETL 擷取

Apoorva Patrikar，Amazon Web Services

總結

此模式說明如何在處理您的資料之前最佳化檔案大小，以最佳化 AWS Glue 上大數據和 Apache Spark 工作負載的擷取、轉換和載入 (ETL) 程序的擷取步驟。使用此模式來防止或解決小型檔案問題。也就是說，當大量小型檔案因為檔案的彙總大小而減慢資料處理速度時。例如，各僅數百 KB 的數百個檔案可以大幅降低 AWS Glue 任務的資料處理速度。這是因為 AWS Glue 必須在 Amazon Simple Storage Service (Amazon S3) 上執行內部清單函數，而 YARN (Yet Another Resource Negotiator) 必須存放大量中繼資料。若要改善資料處理速度，您可以使用分組讓 ETL 任務將一組輸入檔案讀取至單一記憶體內分割區。分割區會自動將較小的檔案分組在一起。或者，您可以使用自訂程式碼，將批次邏輯新增至現有的檔案。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 一或多個 AWS 黏附任務
- 一或多個大數據或 Apache Spark 工作負載
- S3 儲存貯體

Architecture

下列模式顯示 AWS Glue 任務如何處理不同格式的資料，然後存放在 S3 儲存貯體中以取得效能的可見性。

該圖顯示以下工作流程：

1.
> **Note**
>
> AWS Glue 任務會將 CSV、JSON 和 Parquet 格式的小型檔案轉換為動態影格。：輸入檔案的大小對 AWS Glue 任務的效能影響最大。

2. AWS Glue 任務會在 S3 儲存貯體中執行內部清單函數。

工具

- AWS Glue 是全受管 ETL 服務。它可協助您可靠地分類、清理、擴充和移動資料存放區和資料串流之間的資料。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

史詩

使用分組來最佳化讀取期間的 ETL 擷取

任務	Description	所需的技能
指定群組大小。	如果您有超過 50，000 個檔案，預設會完成分組。不過，您也可以在 connectionOptions 參數中指定群組大小，以使用少於 50，000 個檔案的分組。connectionOptions 參數位於 create_dynamic_frame.from_options 方法中。	資料工程師
撰寫分組程式碼。	使用 create_dynamic_frame 方法來建立動態影格。例如： `S3bucket_node1 = glueContext.create_dynamic_frame.from_options( format_options={"multiline": False}, connection_type="s3", format="json",`	資料工程師

任務	Description	所需的技能
	connection_options ={ "paths": ["s3://bucket/prefix/file.json"], "recurse": True, "groupFiles": 'inPartition', "groupSize": 1048576 }, transformation_ctx="S3bucket_node1",) **Note** 使用 groupFiles 將 Amazon S3 分割區群組中的檔案分組。使用 groupSize 設定要在記憶體中讀取之群組的目標大小。groupSize 以位元組 (1048576 = 1 MB 指定)。	
將程式碼新增至工作流程。	在 AWS Glue 中將分組程式碼新增至您的任務工作流程。	資料工程師

使用自訂邏輯來最佳化 ETL 擷取

任務	Description	所需的技能
選擇語言和處理平台。	選擇專為您的使用案例量身打造的指令碼語言和處理平台。	雲端架構師
撰寫程式碼。	撰寫自訂邏輯，將檔案批次處理在一起。	雲端架構師
將程式碼新增至工作流程。	在 AWS Glue 中將程式碼新增至您的任務工作流程。這可讓您的自訂邏輯在每次執行任務時套用。	資料工程師

在轉換後寫入資料時重新分割

任務	Description	所需的技能
分析消耗模式。	了解下游應用程式如何使用您寫入的資料。例如，如果他們每天查詢資料，而且您只有每個區域的分割區資料，或具有非常小的輸出檔案，例如每個檔案 2.5 KB，則這並非消耗的理想選擇。	DBA
寫入前重新分割資料。	在處理期間（根據處理邏輯）和處理後（根據耗用），根據聯結或查詢重新分割。例如，根據位元組大小重新分割，例如 `.repartition(100000)` ，或根據資料欄重新分割，例如 `.repartition("column_name")` 。	資料工程師

相關資源

- 讀取較大群組中的輸入檔案
- 監控 AWS Glue
- 使用 Amazon CloudWatch 指標監控 AWS Glue
- 任務監控與偵錯
- AWS Glue 上的無伺服器 ETL 入門

其他資訊

判斷檔案大小

無法直接判斷檔案大小是否太大或太小。檔案大小對處理效能的影響取決於叢集的組態。在核心 Hadoop 中，我們建議您使用 128 MB 或 256 MB 的檔案，以充分利用區塊大小。

對於 AWS Glue 上的大多數文字檔案工作負載，我們建議 5-10 DPU 叢集的檔案大小介於 100 MB 到 1 GB 之間。若要找出輸入檔案的最佳大小，請監控 AWS Glue 任務的預先處理區段，然後檢查任務的 CPU 使用率和記憶體使用率。

其他考量事項

如果早期 ETL 階段的效能是瓶頸，請考慮在處理之前分組或合併資料檔案。如果您完全控制檔案產生程序，在原始資料傳送至 AWS 之前，在來源系統本身彙總資料點會更有效率。

使用 透過驗證、轉換和分割來協調 ETL 管道 AWS Step Functions

Sandip Gangapadhyay，Amazon Web Services

總結

此模式說明如何建置無伺服器擷取、轉換和載入 (ETL) 管道，以驗證、轉換、壓縮和分割大型 CSV 資料集，以實現效能和成本最佳化。管道由 協調， AWS Step Functions 並包含錯誤處理、自動重試和使用者通知功能。

當 CSV 檔案上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體來源資料夾時，ETL 管道會開始執行。管道會驗證來源 CSV 檔案的內容和結構描述、將 CSV 檔案轉換為壓縮的 Apache Parquet 格式、依年、月和日分割資料集，並將其存放在單獨的資料夾中，以供分析工具處理。

自動執行此模式的程式碼可在具有儲存庫的 ETL 管道 AWS Step Functions的 GitHub 上取得。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- AWS Command Line Interface (AWS CLI) 已安裝並使用 設定 AWS 帳戶，因此您可以透過部署 AWS CloudFormation 堆疊來建立 AWS 資源。建議使用第 2 AWS CLI 版。如需說明，請參閱 AWS CLI 文件中的安裝或更新至最新版本的 AWS CLI。如需組態指示，請參閱 AWS CLI 文件中的組態和登入資料檔案設定。
- Amazon S3 儲存貯體。
- 具有正確結構描述的 CSV 資料集。（此模式隨附的程式碼儲存庫提供範例 CSV 檔案，其中包含您可以使用的正確結構描述和資料類型。）
- 支援 的 Web 瀏覽器 AWS 管理主控台。（請參閱支援的瀏覽器清單。）
- AWS Glue 主控台存取。
- AWS Step Functions 主控台存取。

限制

- 在 中 AWS Step Functions，保留歷史記錄日誌的限制上限為 90 天。如需詳細資訊，請參閱 AWS Step Functions 文件中的 Step Functions 服務配額。

產品版本

- 適用於 的 Python 3.13 AWS Lambda
- AWS Glue 4.0 版

Architecture

圖表中說明的工作流程包含下列高階步驟：

1. 使用者將 CSV 檔案上傳至 Amazon S3 中的來源資料夾。
2. Amazon S3 通知事件會啟動啟動 AWS Step Functions 狀態機器的 AWS Lambda 函數。
3. Lambda 函數會驗證原始 CSV 檔案的結構描述和資料類型。
4. 根據驗證結果：
   a. 如果驗證來源檔案成功，檔案會移至階段資料夾以進行進一步處理。
   b. 如果驗證失敗，檔案會移至錯誤資料夾，並透過 Amazon Simple Notification Service (Amazon SNS) 傳送錯誤通知。
5. AWS Glue 爬蟲程式會從 Amazon S3 中的階段資料夾建立原始檔案的結構描述。
6. AWS Glue 任務會將原始檔案轉換、壓縮和分割為 Parquet 格式。
7. AWS Glue 任務也會將檔案移至 Amazon S3 中的轉換資料夾。
8. AWS Glue 爬蟲程式會從轉換的檔案建立結構描述。產生的結構描述可供任何分析任務使用。您也可以使用 Amazon Athena 執行臨機操作查詢。
9. 如果管道完成時沒有發生錯誤，結構描述檔案會移至封存資料夾。如果遇到任何錯誤，檔案會改為移至錯誤資料夾。
10. Amazon SNS 會根據管道完成狀態傳送通知，指出成功或失敗。

此模式中使用的所有 AWS 資源都是無伺服器。沒有要管理的伺服器。

工具

AWS 服務

- AWS Glue – AWS Glue 是一種全受管 ETL 服務，可讓客戶輕鬆準備和載入其資料以進行分析。

- AWS Step Functions – AWS Step Functions 是一種無伺服器協同運作服務，可讓您結合 AWS Lambda 函數和其他 AWS 服務 來建置業務關鍵型應用程式。透過 AWS Step Functions 圖形主控台，您會將應用程式的工作流程視為一系列的事件驅動步驟。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種物件儲存服務，可提供業界領先的可擴展性、資料可用性、安全性和效能。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一種高可用性、耐用、安全、全受管的 pub/sub 訊息服務，可讓您解耦微服務、分散式系統和無伺服器應用程式。
- AWS Lambda – AWS Lambda 是一種運算服務，可讓您在不佈建或管理伺服器的情況下執行程式碼。 只會在需要時 AWS Lambda 執行程式碼，並自動擴展，從每天的幾個請求擴展到每秒數千個請求。

Code

此模式的程式碼可在具有儲存庫的 ETL 管道 AWS Step Functions的 GitHub 上取得。程式碼儲存庫包含下列檔案和資料夾：

- `template.yml` – AWS CloudFormation 使用 範本建立 ETL 管道 AWS Step Functions。
- `parameter.json` – 包含所有參數和參數值。您可以更新此檔案以變更參數值，如 Epics 一節中所述。
- `myLayer/python` 資料夾 – 包含建立此專案所需 AWS Lambda layer 所需的 Python 套件。
- `lambda` 資料夾 – 包含下列 Lambda 函數：
  - `move_file.py` – 將來源資料集移至封存、轉換或錯誤資料夾。
  - `check_crawler.py` – 在傳送失敗訊息之前，檢查 AWS Glue 爬蟲程式的狀態，其次數與`RETRYLIMIT` 環境變數所設定的次數相同。
  - `start_crawler.py` – 啟動 AWS Glue 爬蟲程式。
  - `start_step_function.py` – 開始 AWS Step Functions。
  - `start_codebuild.py` – 啟動 AWS CodeBuild 專案。
  - `validation.py` – 驗證輸入原始資料集。
  - `s3object.py` – 在 Amazon S3 儲存貯體內建立所需的目錄結構。
  - `notification.py` – 在管道結尾傳送成功或錯誤通知。

若要使用範例程式碼，請遵循 Epics 區段中的指示。

史詩

準備來源檔案

任務	Description	所需的技能
複製範本程式碼儲存庫。	1. 使用儲存庫開啟 ETL 管道 AWS Step Functions。 2. 在主要儲存庫頁面的檔案清單上方選擇程式碼，然後複製以 HTTPS 複製下列出的 URL。 3. 將工作目錄變更為您要存放範例檔案的位置。 4. 在終端機或命令提示字元中，輸入 命令： `git clone <repoURL>` 其中 `<repoURL>` 是指您在步驟 2 中複製的 URL。	開發人員
更新參數值。	在儲存庫的本機副本中，編輯 `parameter.json` 檔案並更新預設參數值，如下所示： • `pS3BucketName` - 用於存放資料集的 Amazon S3 儲存貯體名稱。範本會為您建立此儲存貯體。儲存貯體名稱必須是全域唯一的。 • `pSourceFolder` - Amazon S3 儲存貯體內將用於上傳來源 CSV 檔案的資料夾名稱。	開發人員

任務	Description	所需的技能
	• `pStageFolder` - Amazon S3 儲存貯體內的資料夾名稱，將在程序期間用作預備區域。 • `pTransformFolder` - Amazon S3 儲存貯體內的資料夾名稱，用於存放轉換和分割的資料集。 • `pErrorFolder` - Amazon S3 儲存貯體內的資料夾，如果無法驗證，來源 CSV 檔案將移至該資料夾。 • `pArchiveFolder` - Amazon S3 儲存貯體內用來封存來源 CSV 檔案的資料夾名稱。 • `pEmailforNotification` — 用於接收成功/錯誤通知的有效電子郵件地址。 • `pPrefix` — 將在 AWS Glue 爬蟲程式名稱中使用的字首字串。 • `pDatasetSchema` — 將驗證來源檔案的資料集結構描述。Cerberus Python 套件用於來源資料集驗證。如需詳細資訊，請參閱 Cerberus 網站。	

任務	Description	所需的技能
將原始碼上傳至 Amazon S3 儲存貯體。	部署自動化 ETL 管道的 AWS CloudFormation 範本之前，您必須封裝範本的來源檔案，並將其上傳至 Amazon S3 儲存貯體。若要執行此操作，請使用預先設定的設定檔執行下列 AWS CLI 命令： `aws cloudformation package --template-file template.yml --s3-bucket <bucket_name> --output-template-file packaged.template --profile <profile_name>` 其中： • `<bucket_name>` 是您要部署堆疊之 中現有 Amazon S3 儲存貯 AWS 區域 體的名稱。此儲存貯體用於存放 CloudFormation 範本的原始碼套件。 • `<profile_name>` 是您在設定 時預先設定的有效 AWS CLI 設定檔 AWS CLI。	開發人員

建立 堆疊。

任務	Description	所需的技能
部署 CloudFormation 範本。	若要部署 AWS CloudFormation 範本，請執行下列 AWS CLI 命令： `aws cloudformation deploy --stack-name <stack_name> --template-file packaged.template --parameter-overrides file://parameter.json --capabilities CAPABILITY_IAM --profile <profile_name>` 其中： • `<stack_name>` 是 CloudFormation 堆疊的唯一識別符。 • `<profile-name>` 是您預先設定的 AWS CLI 設定檔。	開發人員
檢查進度。	在 AWS CloudFormation 主控台上，檢查堆疊開發的進度。當狀態為 時`CREATE_COMPLETE` ，堆疊已成功部署。	開發人員
請記下 AWS Glue 資料庫名稱。	堆疊的輸出索引標籤會顯示 AWS Glue 資料庫的名稱。金鑰名稱為 `GlueDBOutput` 。	開發人員

測試管道

任務	Description	所需的技能
啟動 ETL 管道。	1. 導覽至 Amazon S3 儲存貯體內的來源資料夾 (`source`或您在 `parameter.json` 檔案中設定的資料夾名稱）。 2. 將範例 CSV 檔案上傳至此資料夾。（程式碼儲存庫提供名為 的範例檔案`Sample_Bank_Transaction_Raw_Dataset.csv` ，您可以使用。）上傳檔案會透過 Step Functions 啟動 ETL 管道。 3. 在 Step Functions 主控台上，檢查 ETL 管道狀態。	開發人員
檢查分割的資料集。	當 ETL 管道完成時，請確認 Amazon S3 轉換資料夾（`transform` 或您在 `parameter.json` 檔案中設定的資料夾名稱）中有可用的分割資料集。	開發人員
檢查分割的 AWS Glue 資料庫。	1. 在 AWS Glue 主控台上，選取堆疊建立的 AWS Glue 資料庫（這是您在上一個史詩中記下的資料庫）。 2. 確認分割的資料表在 中可用 AWS Glue Data Catalog。	開發人員
執行查詢。	（選用）使用 Amazon Athena 在分割和轉換的資料庫上執行臨機操作查詢。如需說	資料庫分析師

任務	Description	所需的技能
	明，請參閱 AWS 文件中的在 Amazon Athena 中執行 SQL 查詢。	

疑難排解

問題	解決方案
AWS Identity and Access Management AWS Glue 任務和爬蟲程式的 (IAM) 許可	如果您進一步自訂 AWS Glue 任務或爬蟲程式，請務必在 AWS Glue 任務使用的 IAM 角色中授予適當的 IAM 許可，或提供資料許可給 AWS Lake Formation。如需詳細資訊，請參閱 AWS 文件。

相關資源

AWS 服務 文件

- AWS Step Functions
- AWS Glue
- AWS Lambda
- Amazon S3
- Amazon SNS

其他資訊

下圖顯示來自 AWS Step Functions Inspector 面板的成功 ETL 管道 AWS Step Functions 工作流程。

下圖顯示 Step Functions Inspector 面板中，因輸入驗證錯誤而失敗之 ETL 管道的 AWS Step Functions 工作流程。

使用 Amazon Redshift ML 執行進階分析

Po Hong 和 Chyanna Antonio，Amazon Web Services

總結

在 Amazon Web Services (AWS) 雲端上，您可以使用 Amazon Redshift Machine Learning (Amazon Redshift ML) 對存放在 Amazon Redshift 叢集或 Amazon Simple Storage Service (Amazon S3) 中的資料執行 ML 分析。Amazon Redshift ML 支援監督式學習，通常用於進階分析。Amazon Redshift ML 的使用案例包括營收預測、信用卡詐騙偵測，以及客戶生命週期價值 (CLV) 或客戶流失預測。

Amazon Redshift ML 可讓資料庫使用者使用標準 SQL 命令輕鬆建立、訓練和部署 ML 模型。Amazon Redshift ML 使用 Amazon SageMaker Autopilot 根據您的資料自動訓練和調整分類或迴歸的最佳 ML 模型，同時保留控制和可見性。

Amazon Redshift、Amazon S3 和 Amazon SageMaker 之間的所有互動都會被抽象化和自動化。訓練並部署 ML 模型後，它在 Amazon Redshift 中會變成使用者定義函數 (UDF)，可用於 SQL 查詢。

此模式補充了 AWS 部落格中的使用 SQL 搭配 Amazon Redshift ML 在 Amazon Redshift 中建立、訓練和部署 ML 模型，以及入門資源中心的建置、訓練和部署 ML 模型 Amazon SageMaker教學。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Amazon Redshift 資料表中的現有資料

技能

- 熟悉 Amazon Redshift ML 所使用的術語和概念，包括機器學習、訓練和預測。如需詳細資訊，請參閱《Amazon Machine Learning (Amazon ML) 文件》中的訓練 ML 模型。
- 具有 Amazon Redshift 使用者設定、存取管理和標準 SQL 語法的經驗。如需詳細資訊，請參閱《Amazon Redshift 文件》中的 Amazon Redshift 入門。
- Amazon S3 和 AWS Identity and Access Management (IAM) 的知識和經驗。
- 在 AWS Command Line Interface (AWS CLI) 中執行命令的經驗也很有幫助，但並非必要。

限制

- Amazon Redshift 叢集和 S3 儲存貯體必須位於相同的 AWS 區域。
- 此模式的方法僅支援監督式學習模型，例如迴歸、二進位分類和多類別分類。

Architecture

下列步驟說明 Amazon Redshift ML 如何與 SageMaker 搭配使用來建置、訓練和部署 ML 模型：

1. Amazon Redshift 會將訓練資料匯出至 S3 儲存貯體。
2. SageMaker Autopilot 會自動預先處理訓練資料。
3. 叫用`CREATE MODEL`陳述式之後，Amazon Redshift ML 會使用 SageMaker 進行訓練。
4. SageMaker Autopilot 會搜尋並建議 ML 演算法和最佳化評估指標的最佳超參數。
5. Amazon Redshift ML 會將輸出 ML 模型註冊為 Amazon Redshift 叢集中的 SQL 函數。
6. ML 模型的 函數可用於 SQL 陳述式。

技術堆疊

- Amazon Redshift
- SageMaker
- Amazon S3

工具

- Amazon Redshift – Amazon Redshift 是一種企業級的 PB 級全受管資料倉儲服務。
- Amazon Redshift ML – Amazon Redshift Machine Learning (Amazon Redshift ML) 是一種強大的雲端服務，可讓所有技能水準的分析師和資料科學家輕鬆使用 ML 技術。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是網際網路的儲存體。
- Amazon SageMaker – SageMaker 是全受管 ML 服務。
- Amazon SageMaker Autopilot – SageMaker Autopilot 是一種功能集，可自動執行自動機器學習 (AutoML) 程序的關鍵任務。

Code

您可以使用下列程式碼，在 Amazon Redshift 中建立受監督的 ML 模型：

```
"CREATE MODEL customer_churn_auto_model
FROM (SELECT state,
             account_length,
             area_code,
             total_charge/account_length AS average_daily_spend,
             cust_serv_calls/account_length AS average_daily_cases,
             churn
      FROM customer_activity
      WHERE  record_date < '2020-01-01'
     )
TARGET churn
FUNCTION ml_fn_customer_churn_auto
IAM_ROLE 'arn:aws:iam::XXXXXXXXXXXX:role/Redshift-ML'
SETTINGS (
  S3_BUCKET 'your-bucket'
);")
```

> **Note**
>
> SELECT 狀態可以參考 Amazon Redshift 一般資料表、Amazon Redshift Spectrum 外部資料表或兩者。

史詩

準備訓練和測試資料集

任務	Description	所需的技能
準備訓練和測試資料集。	登入 AWS 管理主控台並開啟 Amazon SageMaker 主控台。遵循建置、訓練和部署機器學習模型教學中的指示，建立具有標籤欄 (監督式訓練) 且無標頭的 .csv 或 Apache Parquet 檔案。	資料科學家

任務	Description	所需的技能
	Note 我們建議您隨機播放並將原始資料集分割為模型訓練的訓練集 (70%)，以及模型效能評估的測試集 (30%)。	

準備和設定技術堆疊

任務	Description	所需的技能
建立和設定 Amazon Redshift 叢集。	在 Amazon Redshift 主控台上，根據您的需求建立叢集。如需詳細資訊，請參閱 Amazon Redshift 文件中的建立叢集。 **Important** Amazon Redshift 叢集必須使用`SQL_PREVIEW` 維護軌道建立。如需預覽追蹤的詳細資訊，請參閱 Amazon Redshift 文件中的選擇叢集維護追蹤。	DBA，雲端架構師
建立 S3 儲存貯體以存放訓練資料和模型成品。	在 Amazon S3 主控台上，為訓練和測試資料建立 S3 儲存貯體。如需建立 S3 儲存貯體的詳細資訊，請參閱從 AWS Quick Starts 建立 S3 儲存貯體。	DBA，雲端架構師

任務	Description	所需的技能
	Important 請確定您的 Amazon Redshift 叢集和 S3 儲存貯體位於相同的區域。	
建立 IAM 政策並將其連接至 Amazon Redshift 叢集。	建立 IAM 政策，以允許 Amazon Redshift 叢集存取 SageMaker 和 Amazon S3。如需指示和步驟，請參閱 Amazon Redshift 文件中的使用 Amazon Redshift ML 的叢集設定。	DBA，雲端架構師
允許 Amazon Redshift 使用者和群組存取結構描述和資料表。	授予許可，以允許 Amazon Redshift 中的使用者和群組存取內部和外部結構描述和資料表。如需步驟和說明，請參閱《Amazon Redshift 文件》中的管理許可和擁有權。	DBA

在 Amazon Redshift 中建立和訓練 ML 模型

任務	Description	所需的技能
在 Amazon Redshift 中建立和訓練 ML 模型。	在 Amazon Redshift ML 中建立和訓練您的 ML 模型。如需詳細資訊，請參閱 Amazon Redshift 文件中的 `CREATE MODEL`陳述式。	開發人員、資料科學家

在 Amazon Redshift 中執行批次推論和預測

任務	Description	所需的技能
使用產生的 ML 模型函數執行推論。	如需使用產生的 ML 模型函數執行推論的詳細資訊，請參閱 Amazon Redshift 文件中的預測。	資料科學家、商業智慧使用者

相關資源

準備訓練和測試資料集

- 使用 Amazon SageMaker 建置、訓練和部署機器學習模型

準備和設定技術堆疊

- 建立 Amazon Redshift 叢集
- 選擇 Amazon Redshift 叢集維護軌道
- 建立 S3 儲存貯體
- 設定 Amazon Redshift 叢集以使用 Amazon Redshift ML
- 管理 Amazon Redshift 中的許可和擁有權

在 Amazon Redshift 中建立和訓練 ML 模型

- Amazon Redshift 中的 CREATE MODEL 陳述式

在 Amazon Redshift 中執行批次推論和預測

- Amazon Redshift 中的預測

其他資源

- Amazon Redshift ML 入門
- 使用 SQL 搭配 Amazon Redshift ML 在 Amazon Redshift 中建立、訓練和部署 ML 模型
- Amazon Redshift 合作夥伴
- AWS 機器學習能力合作夥伴

使用 Amazon Athena 查詢具有 SQL 的 Amazon DynamoDB 資料表

Gavin Perrie、Ajit Ambike 和 Brad Yates，Amazon Web Services

總結

如果您的資料包含 Amazon Simple Storage Service (Amazon S3) 以外的來源，您可以使用聯合查詢來存取這些關聯式、非關聯式、物件或自訂資料來源。此模式說明如何使用 SQL 資料來源連接器，透過 Amazon Athena 設定對 Amazon DynamoDB 的聯合查詢存取。

使用此模式，您可以執行下列動作：

- 使用 SQL 查詢 DynamoDB 資料表。
- 在 Athena 中執行聯合 SQL 查詢，並將 DynamoDB 資料表與其他支援的資料來源聯結。

先決條件和限制

先決條件

- DynamoDB 資料表。
- Athena 工作群組設定為使用 Athena 引擎版本 2。如需說明，請參閱 Athena 文件。
- `AthenaDynamoDBConnector` AWS Lambda 函數可以溢出資料的 S3 儲存貯體。S3 儲存貯體和 Lambda 函數必須位於相同的 AWS 區域。

如果這是您第一次存取 Athena，您將需要額外的 S3 儲存貯體來做為查詢結果位置。如需說明，請參閱 Athena 文件。

限制

- 不支援 INSERT INTO 等寫入操作。

產品版本

- GitHub 上的 Athena 查詢聯合版本

Architecture

目標架構

下圖顯示建立模式後的連線流程。使用者連線到 Amazon Athena 以提供查詢。Athena 會將查詢和目標傳遞至 DynamoDB 資料來源連接器 Lambda 函數，該函數會擷取資料並將其傳回給 Athena。如果傳回大量資料，Athena 會在封裝和傳回完整資料集之前，將暫時結果存放在溢出儲存貯體中。

工具

AWS 服務

- Amazon Athena 是一種互動式查詢服務，可協助您使用標準 SQL 直接在 Amazon Simple Storage Service (Amazon S3) 中分析資料。此模式使用 Amazon Athena DynamoDB Connector，這是一種使用 Amazon Athena Query Federation SDK 建置並透過 安裝為 AWS Lambda 應用程式的工具 AWS Serverless Application Repository。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

程式碼儲存庫

此模式的程式碼可在 GitHub Athena 查詢聯合儲存庫中使用。

史詩

設定和測試 DynamoDB 資料來源連接器

任務	Description	所需的技能
部署 AthenaDynamoDBConnector 應用程式。	若要部署 AthenaDynamoDBConnecter，請執行下列動作： 1. 登入 AWS 管理主控台，然後選擇 AWS 區域 您用於 DynamoDB 資料表和溢出儲存貯體的 。	AWS DevOps

任務	Description	所需的技能
	2. 開啟位於 https：//https://console.aws.amazon.com/serverlessrepo/ 的 Serverless Application Repository。 3. 在導覽窗格中，選擇可用的應用程式。 4. 對於 AWS Identity and Access Management (IAM) 存取，在搜尋列下，選取顯示建立自訂 IAM 角色或資源政策的應用程式核取方塊。 5. 搜尋並選取 AthenaDynamoDBConnector，並確保列出的作者是 Amazon Athena Federation。 6. 在應用程式設定中，輸入下列值： • SpillBucket – 函數可以溢出資料的位置。 • AthenaCatalogName – 將要建立的 Lambda 函數名稱。名稱也會用作 Athena 中的資料來源名稱。 7. 選取核取方塊以確認建立 IAM 角色和政策。 8. 選擇部署。	

任務	Description	所需的技能
建立 Athena 的資料來源。	若要建立資料來源，請執行下列動作： 1. 在 開啟 Athena 主控台 https：//https://console.aws.amazon.com/athena/。 2. 展開導覽窗格，然後選擇資料來源。 3. 選擇 Create data source (建立資料來源)。 4. 選擇 Amazon DynamoDB。 5. 輸入資料來源名稱。 6. 選取您建立的 Lambda 函數。 7. 檢閱詳細資訊，然後選擇建立資料來源。	AWS DevOps
使用 Athena 查詢 DynamoDB 資料表。	若要查詢 DynamoDB 資料表，請執行下列動作： 1. 在 Athena 主控台上，展開導覽窗格，然後選擇查詢編輯器。 2. 在資料來源下拉式清單中，選擇您建立的資料來源。 3. 確認 DynamoDB 資料表列於資料表下。 4. 執行查詢。	應用程式開發人員

疑難排解

問題	解決方案
使用 查詢失敗GENERIC_INTERNAL_ERROR: The bucket is in this region: <region>。	請確定在相同的 中建立 Athena 溢出儲存貯體和 Lambda 函數 AWS 區域。
Athena 主控台上看不到新建立的資料來源。	Athena 資料目錄是區域性的。確定 AthenaDynamoDBConnector 已部署在您嘗試使用 Athena 的區域中。
您無法針對新建立的資料來源執行查詢。	檢查查詢結果位置是否已設定。

相關資源

- Amazon Athena DynamoDB 連接器
- Amazon Athena 聯合查詢

使用 Athena 存取、查詢和聯結 Amazon DynamoDB 資料表

Moinul Al-Mamun，Amazon Web Services

總結

此模式說明如何使用 Amazon Athena DynamoDB 連接器設定 Amazon Athena 和 Amazon DynamoDB 之間的連線。連接器使用 AWS Lambda 函數來查詢 DynamoDB 中的資料。您不需要撰寫任何程式碼來設定連線。建立連線後，您可以使用 Athena 聯合查詢從 Athena 執行 SQL 命令，快速存取和分析 DynamoDB 資料表。您也可以將一或多個 DynamoDB 資料表加入彼此或其他資料來源，例如 Amazon Redshift 或 Amazon Aurora。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶，具有管理 DynamoDB 資料表、Athena 資料來源、Lambda 和 AWS Identity and Access Management (IAM) 角色的許可
- Athena 可儲存查詢結果的 Amazon Simple Storage Service (Amazon S3) 儲存貯體
- Athena DynamoDB 連接器可以短期儲存資料的 S3 儲存貯體
- 支援 Athena 引擎第 2 版的 AWS 區域
- 存取 Athena 和所需 S3 儲存貯體的 IAM 許可
- Amazon Athena DynamoDB 連接器，已安裝

限制

查詢 DynamoDB 資料表需要付費。超過幾 GB (GBs資料表大小可能會產生高成本。建議您在執行任何完整資料表 SCAN 操作之前考慮成本。如需詳細資訊，請參閱 Amazon DynamoDB 定價。為了降低成本並實現高效能，我們建議您在查詢中一律使用 LIMIT （例如 `SELECT * FROM table1 LIMIT 10`)。此外，在生產環境中執行 JOIN 或 GROUP BY 查詢之前，請考慮資料表的大小。如果您的資料表太大，請考慮其他選項，例如將資料表遷移至 Amazon S3。

Architecture

下圖顯示使用者如何從 Athena 在 DynamoDB 資料表上執行 SQL 查詢。

該圖顯示以下工作流程：

1. 若要查詢 DynamoDB 資料表，使用者會從 Athena 執行 SQL 查詢。
2. Athena 啟動 Lambda 函數。
3. Lambda 函數會查詢 DynamoDB 資料表中請求的資料。
4. DynamoDB 會將請求的資料傳回至 Lambda 函數。然後，函數會透過 Athena 將查詢結果傳輸給使用者。
5. Lambda 函數會將資料存放在 S3 儲存貯體中。

技術堆疊

- Amazon Athena
- Amazon DynamoDB
- Amazon S3
- AWS Lambda

工具

- Amazon Athena 是一種互動式查詢服務，可協助您使用標準 SQL 直接在 Amazon S3 中分析資料。
- Amazon Athena DynamoDB Connector 是一種 AWS 工具，可讓 Athena 與 DynamoDB 連線，並使用 SQL 查詢存取您的資料表。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而不需要佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。

史詩

建立範例 DynamoDB 資料表

任務	Description	所需的技能
建立第一個範例資料表。	1. 登入 AWS 管理主控台並開啟 DynamoDB 主控台。 2. 選擇 Create Table (建立資料表)。	開發人員

任務	Description	所需的技能
	3. 針對資料表名稱，輸入 dydbtable1。 4. 對於分割區索引鍵，輸入 PK1。 5. 針對排序索引鍵，輸入 SK1。 6. 在資料表設定區段中，選擇自訂設定。 7. 在資料表類別區段中，選擇 DynamoDB Standard。 8. 在讀取/寫入容量設定區段中，針對容量模式選擇隨需。 9. 在靜態加密區段中，選擇 Amazon DynamoDB 所擁有。 10選擇 Create Table (建立資料表)。	

任務	Description	所需的技能

將範例資料插入第一個資料表。	1. 開啟 DynamoDB 主控台。
	2. 在導覽窗格中，選擇資料表，然後在名稱欄中選擇您的資料表。
	3. 選擇動作，然後選擇建立項目。
	4. 選擇 JSON 檢視。
	5. 在屬性編輯器的標題列中，關閉檢視 DynamoDB JSON。
	6. 在屬性編輯器中，逐一輸入下列範例資料：
	
	`{ "PK1": "1234", "SK1": "info", "Salary": "5000" }`
	
	`{ "PK1": "1235", "SK1": "info", "Salary": "5200" }`
		開發人員

任務	Description	所需的技能
建立第二個範例資料表。	1. 開啟 DynamoDB 主控台。 2. 選擇 Create Table (建立資料表)。 3. 針對資料表名稱，輸入 dydbtable2。 4. 針對分割區索引鍵，輸入 PK2。 5. 針對排序索引鍵，輸入 SK2。 6. 在資料表設定區段中，選擇自訂設定。 7. 在資料表類別區段中，選擇 DynamoDB Standard。 8. 在讀取/寫入容量設定區段中，針對容量模式選擇隨需。 9. 在靜態加密區段中，選擇 Amazon DynamoDB 所擁有。 10選擇 Create Table (建立資料表)。	開發人員

任務	Description	所需的技能

將範例資料插入第二個資料表。	1. 開啟 DynamoDB 主控台。
	2. 在導覽窗格中，選擇資料表，然後在名稱欄中選擇您的資料表。
	3. 選擇動作，然後選擇建立項目。
	4. 在屬性編輯器的標題列中，關閉檢視 DynamoDB JSON。
	5. 在屬性編輯器中，逐一輸入下列範例資料：
	`{ "PK2": "1234", "SK2": "bonus", "Bonus": "500" }`
	`{ "PK2": "1235", "SK2": "bonus", "Bonus": "1000" }`
		開發人員

在 Athena for DynamoDB 中建立資料來源

任務	Description	所需的技能
設定資料來源連接器。	建立 DynamoDB 的資料來源，然後建立 Lambda 函數以連線至該資料來源。	開發人員

任務	Description	所需的技能
	1. 登入 AWS 管理主控台並開啟 Athena 主控台。 2. 在導覽窗格中，選擇資料來源，然後選擇建立資料來源。 3. 選擇 Amazon DynamoDB 資料來源，然後選擇下一步。 4. 在資料來源詳細資訊區段中，針對資料來源名稱輸入 testDynamoDB。 5. 在連線詳細資訊區段中，選取已部署的 Lambda 函數，或者如果您沒有要用於此模式的 Lambda 函數，請選擇建立 Lambda 函數。注意：如需建立 Lambda 函數的詳細資訊，請參閱 Lambda 開發人員指南中的 Lambda 入門。 6. （選用）如果您選擇建立 Lambda 函數，則必須在部署堆疊之前設定 Java 應用程式包含的 AWS CloudFormation 範本。範本包含 ApplicationName、SpillBucket、AthenaCatalogName 和其他應用程式設定。注意：部署此以 Java 為基礎的應用程式後，堆疊會建立 Lambda 函數，讓 Athena 能夠與 DynamoDB 通訊。這可讓您的資料表透過 SQL 命令存取。	

任務	Description	所需的技能
	7. 部署您的 Lambda 函數。 8. 選擇下一步。	
確認 Lambda 函數可以存取 S3 溢出儲存貯體。	1. 開啟 Lambda 主控台。 2. 在導覽窗格中，選擇函數，然後選擇您先前建立的函數。 3. 選擇 Configuration (組態) 索引標籤。 4. 在左側窗格中，選擇環境變數，然後確認金鑰的值為 `spill_bucket` 。 5. 在左側窗格中，選擇許可，然後在執行角色區段中，選擇連接的 IAM 角色。注意：系統會將您導向至 IAM 主控台中連接至 Lambda 函數的 IAM 角色。 6. 確認您擁有儲存`spill_bucket` 貯體的寫入許可。 如果您遇到錯誤，請參閱此模式中的其他資訊一節以取得指引。	開發人員

從 Athena 存取 DynamoDB 資料表

任務	Description	所需的技能
查詢 DynamoDB 資料表。	1. 登入 AWS 管理主控台並開啟 Athena 主控台。	開發人員

任務	Description	所需的技能
	2. 在導覽窗格中，選擇資料來源，然後選擇建立資料來源。 3. 在導覽窗格中，選擇 Query Editor (查詢編輯器)。 4. 在編輯器索引標籤的資料來源區段中，選擇資料來源的資料來源。 5. 如需資料庫，請選擇您的資料庫。 6. 針對查詢 1，輸入下列查詢：`SELECT * FROM dydbtable1 t1;` 7. 選擇執行，然後驗證資料表中的輸出。 8. 針對查詢 2，輸入下列查詢：`SELECT * FROM dydbtable2 t2;` 9. 選擇執行，然後驗證資料表中的輸出。	

任務	Description	所需的技能
聯結兩個 DynamoDB 資料表。	DynamoDB 是 NoSQL 資料存放區，不支援 SQL 聯結操作。因此，您必須在兩個 DynamoDB 資料表上執行聯結操作： 1. 選擇加號圖示以建立另一個查詢。 2. 針對查詢 3，輸入下列查詢： `SELECT pk1, salary, bonus FROM dydbtable1 t1 JOIN dydbtable2 t2 ON t1.pk1 = t2.pk2;`	開發人員

相關資源

- Amazon Athena DynamoDB 連接器 (AWS 實驗室)
- 使用 Amazon Athena 的新聯合查詢查詢任何資料來源 (AWS 大數據部落格)
- Athena 引擎版本參考 (Athena 使用者指南)
- 使用 AWS Glue 和 Amazon Athena 簡化 Amazon DynamoDB 資料擷取和分析 (AWS 資料庫部落格)

其他資訊

如果您在 Athena 中使用 `{bucket_name}/folder_name/` 格式`spill_bucket`的 執行查詢，則您可能會收到下列錯誤訊息：

```
"GENERIC_USER_ERROR: Encountered an exception[java.lang.RuntimeException] from your
 LambdaFunction[arn:aws:lambda:us-east-1:xxxxxx:function:testdynamodb] executed in
```

```
 context[retrieving meta-data] with message[You do NOT own the spill bucket with the
 name: s3://amzn-s3-demo-bucket/athena_dynamodb_spill_data/]
This query ran against the "default" database, unless qualified by the query. Please
 post the error message on our forum  or contact customer support with Query Id:
 [query-id]"
```

若要解決此錯誤，請將 Lambda 函數的環境變數更新spill_bucket為 {bucket_name_only}，然後針對儲存貯體寫入存取更新下列 Lambda IAM 政策：

```
{
             "Action": [
                 "s3:GetObject",
                 "s3:ListBucket",
                 "s3:GetBucketLocation",
                 "s3:GetObjectVersion",
                 "s3:PutObject",
                 "s3:PutObjectAcl",
                 "s3:GetLifecycleConfiguration",
                 "s3:PutLifecycleConfiguration",
                 "s3:DeleteObject"
             ],
             "Resource": [
                 "arn:aws:s3:::spill_bucket",
                 "arn:aws:s3:::spill_bucket/*"
             ],
             "Effect": "Allow"
         }
```

或者，您可以移除先前建立的 Athena 資料來源連接器，並僅使用 {bucket_name}重新建立spill_bucket。

設定最低可行的資料空間以在組織之間共用資料

Ramy Hcini、Michael Miller 和 Jorge Hernandez Suarez，Amazon Web Services

Ismail Abdellaoui 和 Malte Gasseling，Think-it

摘要

資料空間是資料交換的聯合網路，可信任並控制資料做為核心原則。它們可讓組織透過提供經濟實惠且技術無關的解決方案，大規模共用、交換和協作資料。

透過使用資料驅動型問題解決方法與涉及所有相關利益相關者end-to-end方法，資料空間有可能大幅推動永續發展未來的努力。

此模式會引導您完成兩個公司如何在 Amazon Web Services (AWS) 上使用資料空間技術來推動其碳排放減少策略的範例。在此案例中，X 公司提供 Y 公司使用的碳排放資料。如需下列資料空間規格詳細資訊，請參閱其他資訊一節：

- 參與者
- 商業案例
- 資料空間授權機構
- 資料空間元件
- 資料空間服務
- 要交換的資料
- 資料模型
- Tractus-X EDC 連接器

模式包含下列步驟：

- 部署基本資料空間所需的基礎設施，其中有兩個參與者正在執行 AWS。
- 以安全的方式使用連接器來交換碳排放強度資料。

此模式會部署 Kubernetes 叢集，透過 Amazon Elastic Kubernetes Service (Amazon EKS) 託管資料空間連接器及其服務。

Eclipse Dataspace Components (EDC) 控制平面和資料平面都部署在 Amazon EKS 上。官方 Tractus-X Helm Chart 將 PostgreSQL 和 HashiCorp Vault 服務部署為相依性。

此外，身分服務會部署在 Amazon Elastic Compute Cloud (Amazon EC2) 上，以複寫最小可行資料空間 (MVDS) 的真實案例。

先決條件和限制

先決條件

- 在您選擇的 中 AWS 帳戶 部署基礎設施的作用中 AWS 區域
- 可存取 Amazon S3 的 AWS Identity and Access Management (IAM) 使用者，將暫時做為技術使用者使用 (EDC 連接器目前不支援使用 角色。 我們建議您特別為此示範建立一個 IAM 使用者，而且此使用者將擁有與其相關聯的有限許可。)
- 在您選擇的 中安裝和設定 AWS Command Line Interface (AWS CLI) AWS 區域
- AWS 安全登入資料
- 工作站上的 eksctl
- 工作站上的 Git
- kubectl
- Helm
- Postman
- AWS Certificate Manager (ACM) SSL/TLS 憑證
- 指向 Application Load Balancer 的 DNS 名稱 (DNS 名稱必須涵蓋在 ACM 憑證內)
- HashiCorp Vault （ 如需使用 AWS Secrets Manager 管理秘密的資訊，請參閱其他資訊一節。）

產品版本

- AWS CLI 第 2 版以上
- Postman 集合 2.1 版

限制

- 連接器選擇 – 此部署使用 EDC 型連接器。不過，請務必考慮 EDC 和 FIWARE True 連接器的優點和功能，以做出符合部署特定需求的明智決策。
- EDC 連接器建置 – 選擇的部署解決方案倚賴 Tractus-X EDC 連接器 Helm Chart，這是一個成熟且經過廣泛測試的部署選項。使用此圖表的決定取決於其常用和在提供的建置中包含基本延伸項目。雖然 PostgreSQL 和 HashiCorp Vault 是預設元件，但您可以視需要靈活地自訂自己的連接器建置。

- 私有叢集存取 – 對已部署 EKS 叢集的存取僅限於私有通道。與叢集的互動僅透過使用 `kubectl`和 IAM 來執行。您可以使用負載平衡器和網域名稱來啟用對叢集資源的公開存取，該名稱必須選擇性地實作，以便向更廣泛的網路公開特定服務。不過，我們不建議提供公開存取。
- 安全重點 – 強調將安全組態抽象化為預設規格，以便您可以專注於 EDC 連接器資料交換中涉及的步驟。雖然預設安全設定是維護的，但在您向公有網路公開叢集之前，請務必啟用安全通訊。此預防措施可確保安全資料處理的強大基礎。
- 基礎設施成本 – 您可以使用 找到基礎設施成本的估算AWS 定價計算工具。簡單的計算顯示，部署的基礎設施每月最多可達 162.92 USD。

Architecture

MVDS 架構包含兩個虛擬私有雲端 (VPCs)，一個用於動態屬性佈建系統 (DAPS) 身分服務，另一個用於 Amazon EKS。

DAPS 架構

下圖顯示 Auto Scaling 群組所控制之 EC2 執行個體上執行的 DAPS。Application Load Balancer 和路由表會公開 DAPS 伺服器。Amazon Elastic File System (Amazon EFS) 會在 DAPS 執行個體之間同步資料。

Amazon EKS 架構

資料空間設計成與技術無關的解決方案，並且存在多個實作。此模式使用 Amazon EKS 叢集來部署資料空間技術元件。下圖顯示 EKS 叢集的部署。工作者節點安裝在私有子網路中。Kubernetes Pod 存取也位於私有子網路中的 Amazon Relational Database Service (Amazon RDS) for PostgreSQL 執行個體。Kubernetes Pod 會將共用資料存放在 Amazon S3 中。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和區域的生命週期中管理資源。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。

- Amazon Elastic File System (Amazon EFS) 協助您在 AWS 雲端中建立和設定共用檔案系統。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 上執行 Kubernetes， AWS 而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Elastic Load Balancing (ELB) 會將傳入的應用程式或網路流量分散到多個目標。例如，您可以在一或多個可用區域中跨 EC2 執行個體、容器和 IP 地址分配流量。

其他工具

- eksctl 是一種命令列公用程式，用於在 Amazon EKS 上建立和管理 Kubernetes 叢集。
- Git 是一種開放原始碼的分散式版本控制系統。
- HashiCorp Vault 提供安全儲存，具有憑證和其他敏感資訊的受控存取。
- Helm 是 Kubernetes 的開放原始碼套件管理員，可協助您在 Kubernetes 叢集上安裝和管理應用程式。
- kubectl 是一種命令列界面，可協助您針對 Kubernetes 叢集執行命令。
- Postman 是 API 平台。

程式碼儲存庫

此模式的 Kubernetes 組態 YAML 檔案和 Python 指令碼可在 GitHub aws-patterns-edc 儲存庫中使用。模式也會使用 Tractus-X EDC 儲存庫。

最佳實務

Amazon EKS 和參與者基礎設施的隔離

Kubernetes 中的命名空間會以這種模式將公司 X 提供者的基礎設施與公司 Y 消費者的基礎設施分開。如需詳細資訊，請參閱 EKS 最佳實務指南。

在更逼真的情況下，每個參與者都會在自己的 Kubernetes 叢集中執行 AWS 帳戶。共用基礎設施 （此模式中的 DAPS) 可供資料空間參與者存取，同時與參與者的基礎設施完全分開。

史詩

設定環境，並佈建 EKS 叢集和 EC2 執行個體

任務	Description	所需的技能
複製儲存庫。	若要將儲存庫複製到您的工作站，請執行下列命令： `git clone https://github.com/Think-iT-Labs/aws-patterns-edc` 工作站必須能夠存取您的 AWS 帳戶。	DevOps 工程師

佈建 Kubernetes 叢集並設定命名空間。	若要在帳戶中部署簡化的預設 EKS 叢集，請在複製儲存庫的工作站上執行下列`eksctl`命令：
	`eksctl create cluster`
	命令會建立跨越三個不同可用區域的 VPC 和私有和公有子網路。建立網路層之後，命令會在 Auto Scaling 群組中建立兩個 `m5.large` EC2 執行個體。
	如需詳細資訊和輸出範例，請參閱 eksctl 指南。
	佈建私有叢集之後，請執行下列命令，將新的 EKS 叢集新增至本機 Kubernetes 組態：
	`aws eks update-kubeconfig --name <EKS`
		DevOps 工程師

任務	Description	所需的技能
	CLUSTER NAME> --region <AWS REGION> 此模式使用 eu-west-1 AWS 區域 來執行所有命令。不過，您可以在偏好的 中執行相同的命令 AWS 區域。 若要確認您的 EKS 節點正在執行且處於就緒狀態，請執行下列命令： kubectl get nodes	
設定命名空間。	若要為提供者和取用者建立命名空間，請執行下列命令： kubectl create ns provider kubectl create ns consumer 在此模式中，請務必使用 provider和 consumer做為命名空間，以符合後續步驟中的組態。	DevOps 工程師

部署身分服務

任務	Description	所需的技能
使用 部署 DAPS AWS CloudFormation。	為了輕鬆管理 DAPS 操作，DAPS 伺服器安裝在 EC2 執行個體上。	DevOps 工程師

任務	Description	所需的技能
	若要安裝 DAPS，請使用 AWS CloudFormation 範本。您將需要先決條件區段中的 ACM 憑證和 DNS 名稱。範本會部署並設定下列項目： • Application Load Balancer • Auto Scaling 群組 • 使用使用者資料設定的 EC2 執行個體，以安裝所有必要的套件 • IAM 角色 • DAPS 您可以登入 AWS 管理主控台 並使用 AWS CloudFormation 主控台來部署 AWS CloudFormation 範本。您也可以使用如下所示的 AWS CLI 命令來部署範本： `aws cloudformation create-stack --stack-name daps \ --template-body file://aws-patterns-edc/cloudformation.yml --parameters \ ParameterKey=CertificateARN,ParameterValue=<ACM Certificate ARN> \ ParameterKey=DNSName,ParameterValue=<DNS name> \ ParameterKey=InstanceType,Paramete`	

任務	Description	所需的技能
	rValue=<EC2 instance type> \ ParameterKey=EnvironmentName,ParameterValue=<Environment Name> --capabilities CAPABILITY_NAMED_IAM 環境名稱是您自己的選擇。我們建議您使用有意義的詞彙，例如 DapsInfrastructure ，因為它將反映在 AWS 資源標籤中。 對於此模式， t3.small 的大小足以執行具有三個 Docker 容器的 DAPS 工作流程。 範本會在私有子網路中部署 EC2 執行個體。這表示無法透過 SSH （安全殼層） 從網際網路直接存取執行個體。執行個體會佈建必要的 IAM 角色和 AWS Systems Manager 代理程式，以透過 Session Manager 的 功能存取執行中的執行個體 AWS Systems Manager。 建議使用 Session Manager 進行存取。或者，您可以佈建堡壘主機，以允許從網際網路存取 SSH。使用堡壘主機方法時，EC2 執行個體可能需要幾分鐘的時間才能開始執行。	

任務	Description	所需的技能

	成功部署 AWS CloudFormation 範本後，將 DNS 名稱指向 Application Load Balancer DNS 名稱。若要確認，請執行下列命令：
	
	`dig <DNS NAME>`
	
	輸出格式應類似以下內容：
	
	`; <<>> DiG 9.16.1-Ubuntu <<>> edc-pattern.think-it.io`
	`;; global options: +cmd`
	`;; Got answer:`
	`;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 42344`
	`;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 1`
	
	`;; OPT PSEUDOSECTION:`
	`; EDNS: version: 0, flags:; udp: 65494`
	`;; QUESTION SECTION:`
	`;edc-pattern.think-it.io. IN A`
	
	`;; ANSWER SECTION:`
	`edc-pattern.think-it.io. 276 IN CNAME daps-alb-iap9zmwy3kn8-1328773120.eu-west-1.elb.amazonaws.com.`
	`daps-alb-iap9zmwy3kn8-1328773120.eu-w`	

任務	Description	所需的技能
	est-1.elb.amazonaw s.com. 36 IN A 52.208.240.129 daps-alb-iap9zmwy3kn8 -1328773120.eu-wes t-1.elb.amazonaws. com. 36 IN A 52.210.15 5.124	

任務	Description	所需的技能

將參與者的連接器註冊至 DAPS 服務。	從為 DAPS 佈建的任何 EC2 執行個體中，註冊參與者：
	1. 使用根使用者在 EC2 執行個體上執行可用的指令碼：
	`cd /srv/mvds/omejdn-daps`
	2. 註冊供應商：
	`bash scripts/register_connector.sh <provider_name>`
	3. 註冊消費者：
	`bash scripts/register_connector.sh <consumer_name>`
	選擇名稱不會影響後續步驟。建議使用 `provider`和 `consumer`或 `companyx`和 `companyy`。
	註冊命令也會使用從建立的憑證和金鑰擷取的必要資訊，自動設定 DAPS 服務。
	當您登入 DAPS 伺服器時，請收集安裝中後續步驟所需的資訊：
	1. 從 `omejdn-daps/config/clients.yml` 取得`client id`供應商和
		DevOps 工程師

任務	Description	所需的技能
	消費者的 。這些client id值是十六進位數字的長字串。 2. 從 omejdn-daps/keys 目錄中，複製 consumer.cert 、provider.cert 、 consumer.key 和 provider.key 檔案的內容。 建議您將文字複製並貼到工作站daps-上字首為 的類似具名檔案。 您應該有提供者和消費者的用戶端 IDs，而且工作站上的工作目錄中應該有四個檔案： • 來源檔案名稱consumer.cert 會變成工作站檔案名稱 daps-consumer.cert 。 • 來源檔案名稱consumer.key 會變成工作站檔案名稱 daps-consumer.key 。 • 來源檔案名稱provider.cert 會變成工作站檔案名稱 daps-provider.cert 。 • 來源檔案名稱provider.key 會變成工作站檔案名稱 daps-provider.key 。	

部署參與者的連接器

任務	Description	所需的技能

複製 Tractus-X EDC 儲存庫並使用 0.4.1 版本。	Tractus-X EDC 連接器的建置需要部署和提供 PostgreSQL（資產資料庫）和 HashiCorp Vault（秘密管理）服務。
	
	Tractus-X EDC Helm Chart 有許多不同的版本。此模式指定 0.4.1 版，因為它使用 DAPS 伺服器。
	
	最新版本使用 Managed Identity Wallet (MIW) 搭配身分服務的分散式實作。
	
	在您建立兩個 Kubernetes 命名空間的工作站上，複製 tractusx-edc 儲存庫，然後查看`release/0.4.1` 分支。
	
	`git clone https://github.com/eclipse-tractusx/tractusx-edc`
	
	`cd tractusx-edc`
	
	`git checkout release/0.4.1`
		DevOps 工程師

設定 Tractus-X EDC Helm Chart。	修改 Tractus-X Helm Chart 範本組態，讓兩個連接器可以互動。 若要這樣做，您可以將 命名空間新增至服務的 DNS 名稱，以便叢集中的其他服務可以解	DevOps 工程師

任務	Description	所需的技能

	析。這些修改應該對 `charts/tractusx-connector/templates/_helpers.tpl` 檔案進行。此模式提供此檔案的最終修改版本供您使用。將其複製並放入檔案 的 `daps`區段`charts/tractusx-connector/templates/_helpers.tpl` 。
	
	請務必註解 中的所有 DAPS 相依性`charts/tractusx-connector/Chart.yaml` :
	
	`dependencies:`
	`  # IDS Dynamic Attribute Provisioning Service (IAM)`
	`  #  - name: daps`
	`  # version: 0.0.1`
	`  # repository: "file://./subcharts/omejdn"`
	`  # alias: daps`
	`  # condition: install.daps`	

任務	Description	所需的技能

設定連接器以在 Amazon RDS 上使用 PostgreSQL。	（選用）此模式不需要 Amazon Relational Database Service (Amazon RDS) 執行個體。不過，我們強烈建議使用 Amazon RDS 或 Amazon Aurora，因為它們提供高可用性、備份和復原等功能。
	
	若要將 Kubernetes 上的 PostgreSQL 取代為 Amazon RDS，請執行下列動作：
	
	1. 佈建 Amazon RDS for PostgreSQL 執行個體。
	2. 在 中`Chart.yaml` ，評論 `PostgreSQL` 區段。
	3. 在 `provider_values.yml` 和 中`consumer_values.yml` ，設定 `postgresql` 區段，如下所示：
	
	`postgresql:`
	`  auth:`
	`    database: edc`
	`    password: <RDS PASSWORD>`
	`    username: <RDS Username>`
	`  jdbcUrl: jdbc:postgresql://<RDS DNS NAME>:5432/edc`
	`  username: <RDS Username>`
	`  password: <RDS PASSWORD>`
		DevOps 工程師

任務	Description	所需的技能

	<pre>primary:
	  persistence:
	  enabled: false
	readReplicas:
	  persistence:
	  enabled: false</pre>	

任務	Description	所需的技能
設定和部署提供者連接器及其服務。	若要設定提供者連接器及其服務，請執行下列動作： 1. 若要將`provider_edc.yaml` 檔案從 `edc_helm_configs` 目錄下載到目前的 Helm Chart 資料夾，請執行下列命令： `wget -q https://raw.githubusercontent.com/Think-iT-Labs/aws-patterns-edc/main/edc_helm_configs/provider_edc.yaml> -P charts/tractusx-connector/` 2. 將下列變數（在 檔案中也標記）取代為其值： • `CLIENT_ID` – DAPS 產生的 ID。`CLIENT_ID` 應該在 DAPS 伺服器上`/srv/mvds/omejdn-daps/config/clients.yml/config/clients.yml` 的中。它應該是十六進位字元的字串。 • DAPS_URL – DAPS 伺服器的 URL。它應該`https://{DNS name}`使用您在執行 AWS CloudFormation 範本時設定的 DNS 名稱。	DevOps 工程師

任務	Description	所需的技能

	• VAULT_TOKEN – 用於保存庫授權的字符。選擇任何值。
	• vault.fullnameOverride – vault-provider 。
	• vault.hashicorp.url – http://vault-provider:8200/ 。
	先前的值假設部署名稱和命名空間名稱是提供者。
	3. 若要從工作站執行 Helm Chart，請使用下列命令：
	`cd charts/tractusx-connector`
	`helm dependency build`
	`helm upgrade --install provider ./ -f provider_edc.yaml -n provider`	

任務	Description	所需的技能

將憑證和金鑰新增至提供者保存庫。	為了避免混淆，請在 `tractusx-edc/charts` 目錄之外產生下列憑證。
	
	例如，執行下列命令以變更為您的主目錄：
	
	`cd ~`
	
	您現在需要將提供者所需的秘密新增至保存庫。
	
	保存庫中的秘密名稱是 `provider_edc.yml` 檔案 `secretNames:` 區段中金鑰的值。根據預設，它們的設定如下：
	
	`secretNames:`
	`  transferProxyTokenSignerPrivateKey: transfer-proxy-token-signer-private-key`
	`  transferProxyTokenSignerPublicKey: transfer-proxy-token-signer-public-key`
	`  transferProxyTokenEncryptionAesKey: transfer-proxy-token-encryption-aes-key`
	`  dapsPrivateKey: daps-private-key`
		DevOps 工程師

任務	Description	所需的技能

	<pre>dapsPublicKey: daps-public-key</pre>
	一開始會產生進階加密標準 (AES) 金鑰、私有金鑰、公有金鑰和自我簽署憑證。這些隨後會新增為保存庫的秘密。
	
	此外，此目錄應包含您從 DAPS 伺服器複製的 `daps-provider.cert` 和 `daps-provider.key` 檔案。
	
	1. 執行下列命令：
	
	<pre># generate a private key
	openssl ecparam -name prime256v1 -genkey -noout -out provider-private-key.pem
	# generate corresponding public key
	openssl ec -in provider-private-key.pem -pubout -out provider-public-key.pem
	# create a self-signed certificate
	openssl req -new -x509 -key provider-private-key.pem -out provider-cert.pem -days 360
	# generate aes key
	openssl rand -base64 32 > provider-aes.key</pre>	

任務	Description	所需的技能

	2. 將秘密新增至保存庫之前，請將換行符號取代為 ，將它們從多行轉換為單行\n：
	
	`cat provider-private-key.pem | sed 's/$/\\\n/'|tr -d '\\n' > provider-private-key.pem.line`
	`cat provider-public-key.pem | sed 's/$/\\\n/'|tr -d '\\n' > provider-public-key.pem.line`
	`cat provider-cert.pem | sed 's/$/\\\\n/'|tr -d '\\n' > provider-cert.pem.line`
	`cat provider-aes.key | sed 's/$/\\\\n/'|tr -d '\\n' > provider-aes.key.line`
	
	`## The following block is for daps certificate and key`
	`openssl x509 -in daps-provider.cert -outform PEM | sed 's/$/\\\\n/'|tr -d '\\n' > daps-provider.cert.line`
	`cat daps-provider.key | sed 's/$/\\\\n/'| tr -d '\\n' > daps-provider.key.line`
	
	3. 若要格式化要新增至保存庫的秘密，請執行下列命令：	

任務	Description	所需的技能
	<pre>JSONFORMAT='{"cont ent": "%s"}' #create a single line in JSON format printf "${JSONFO RMAT}\\n" "`cat  provider-private- key.pem.line`" > provider-private-k ey.json printf "${JSONFO RMAT}\\n" "`cat  provider-public- key.pem.line`" > provider-public-ke y.json printf "${JSONFO RMAT}\\n" "`cat  provider-cert.pem. line`" > provider- cert.json printf "${JSONFO RMAT}\\n" "`cat  provider-aes.key.l ine`" > provider- aes.json printf "${JSONFO RMAT}\\n" "`cat daps- provider.key.line`" > daps-provider.key. json printf "${JSONFO RMAT}\\n" "`cat daps- provider.cert.line`" > daps-provider.cert .json</pre> 秘密現在採用 JSON 格式，已準備好新增到保存庫。	

任務	Description	所需的技能

	4. 若要取得保存庫的 Pod 名稱，請執行下列命令：
	`kubectl get pods -n provider|egrep "vault|NAME"`
	Pod 名稱將類似於 `"vault-provider-0"` 。建立轉送至保存庫的連接埠時，會使用此名稱。連接埠轉送可讓您存取保存庫以新增秘密。您應該從已設定 AWS 登入資料的工作站執行此操作。
	5. 若要存取保存庫，請使用 `kubectl`設定連接埠轉送：
	`kubectl port-forward <VAULT_POD_NAME> 8200:8200 -n provider`
	您現在應該可以透過瀏覽器或 CLI 存取保存庫。
	瀏覽器
	1. 使用瀏覽器，導覽至 https：//http://127.0.0.1:8200。
	2. 使用您先前在 中設定的字符登入`provider_edc.yml` 。在秘密引擎中，建立三個秘密。每個秘密都會有一個`Path`	

任務	Description	所需的技能
	for this secret值，這是下列清單中顯示的秘密名稱。在 secret data區段中，金鑰的名稱將是 ，content而值將是來自名為 之個別檔案的單行文字.line。 3. 秘密名稱來自 provider_edc.yml 檔案的 secretNames 區段。 4. 建立下列秘密： • 具有檔案名稱transfer-proxy-token-signer-private-key 的秘密 provider-private-key.pem.line • 具有檔案名稱transfer-proxy-token-signer-public-key 的秘密 provider-cert.pem.line • 具有檔案名稱transfer-proxy-token-encryption-aes-key 的秘密 provider-aes.key.line • 具有檔案名稱daps-private-key 的秘密 daps-provider.key.line • 具有檔案名稱daps-public-key 的秘密	

任務	Description	所需的技能
	daps-provider.cert .line 保存庫 CLI CLI 也會使用您設定的連接埠轉送。 1. 在您的工作站上，遵循 HashiCorp Vault 文件中的指示安裝 Vault CLI。 2. 若要使用您在 中設定的字符登入保存庫provider_edc.yml ，請執行下列命令： vault login -address= http://127.0.0.1:8 200 使用正確的字符，您應該會看到訊息 "Success! You are now authenticated." 3. 若要使用您先前建立的 JSON 格式檔案來建立秘密，請執行下列程式碼： vault kv put -address= http://127.0.0.1:8 200 secret/transfer- proxy-token-signer-p rivate-key @provider -private-key.json vault kv put - address=http://12	

任務	Description	所需的技能
	7.0.0.1:8200 secret/transfer-proxy-token-signer-public-key @provider-cert.json vault kv put -address=http://127.0.0.1:8200 secret/transfer-proxy-token-encryption-aes-key @provider-aes.json vault kv put -address=http://127.0.0.1:8200 secret/daps-private-key @daps-provider.key.json vault kv put -address=http://127.0.0.1:8200 secret/daps-public-key @daps-provider.cert.json	

任務	Description	所需的技能
設定和部署消費者連接器及其服務。	設定和部署取用者的步驟與您為提供者完成的步驟類似： 1. 若要將 consumer_edc.yaml 從 aws-patterns-edc 儲存庫複製到 tractusx-edc/charts/tractusx-connector 資料夾，請執行下列命令： ``` cd tractusx-edc wget -q https://raw.githubusercontent.com/Think-iT-Labs/aws-patterns-edc/main/edc_helm_configs/consumer_edc.yaml -P charts/tractusx-connector/ ``` 2. 使用其實際值更新下列變數： • CONSUMER_CLIENT_ID – DAPS 產生的 ID。CONSUMER_CLIENT_ID 應該在 DAPS 伺服器上config/clients.yml 的 中。 • DAPS_URL – 您用於提供者的相同 DAPS URL。 • VAULT_TOKEN – 用於保存庫授權的字符。選擇任何值。	

任務	Description	所需的技能

	• `vault.fullnameOverride` – `vault-consumer`
	• `vault.hashicorp.url` – `http://vault-provider:8200/`
	
	先前的值假設部署名稱和命名空間名稱為 `consumer`。
	3. 若要執行 Helm Chart，請使用下列命令：
	
	`cd charts/tractusx-connector`
	
	`helm upgrade --install consumer ./ -f consumer_edc.yaml -n consumer`	

任務	Description	所需的技能

將憑證和金鑰新增至取用者保存庫。	從安全角度來看，我們建議為每個資料空間參與者重新產生憑證和金鑰。此模式會為消費者重新產生憑證和金鑰。
	
	這些步驟與提供者的步驟非常類似。您可以驗證 `consumer_edc.yml` 檔案中的秘密名稱。
	
	保存庫中的秘密名稱是 `secretNames:` 區段中金鑰的值`consumer_edc.yml file`。根據預設，它們的設定如下：
	
	`secretNames:`
	`  transferProxyTokenSignerPrivateKey: transfer-proxy-token-signer-private-key`
	`  transferProxyTokenSignerPublicKey: transfer-proxy-token-signer-public-key`
	`  transferProxyTokenEncryptionAesKey: transfer-proxy-token-encryption-aes-key`
	`  dapsPrivateKey: daps-private-key`
	`  dapsPublicKey: daps-public-key`
		DevOps 工程師

任務	Description	所需的技能

	您從 DAPS 伺服器複製的 `daps-consumer.cert` 和 `daps-consumer.key` 檔案應該已存在於此目錄中。
	1. 執行下列命令：
	`# generate a private key`
	`openssl ecparam -name prime256v1 -genkey -noout -out consumer-private-key.pem`
	`# generate corresponding public key`
	`openssl ec -in consumer-private-key.pem -pubout -out consumer-public-key.pem`
	`# create a self-signed certificate`
	`openssl req -new -x509 -key consumer-private-key.pem -out consumer-cert.pem -days 360`
	`# generate aes key`
	`openssl rand -base64 32 > consumer-aes.key`
	2. 手動編輯檔案以使用 取代換行符號\n，或使用類似下列的三個命令：
	`cat consumer-private-key.pem | sed 's/$/\\\n/'|tr -d '\\n' >`	

任務	Description	所需的技能
	consumer-private-key.pem.line cat consumer-public-key.pem | sed 's/$/\\\n/'\|tr -d '\\n' > consumer-public-key.pem.line cat consumer-cert.pem \| sed 's/$/\\\\n/'|tr -d '\\n' > consumer-cert.pem.line cat consumer-aes.key | sed 's/$/\\\\n/'\|tr -d '\\n' > consumer-aes.key.line cat daps-consumer.cert \| sed 's/$/\\\\n/'|tr -d '\\n' > daps-consumer.cert.line cat daps-consumer.key | sed 's/$/\\\\n/'\|tr -d '\\n' > daps-consumer.key.line 3. 若要格式化要新增至保存庫的秘密，請執行下列命令： JSONFORMAT='{"content": "%s"}' #create a single line in JSON format printf "${JSONFORMAT}\\n" "`cat consumer-private-key.pem.line`" >	

任務	Description	所需的技能
	consumer-private-key.json printf "${JSONFORMAT}\\n" "`cat consumer-public-key.pem.line`" > consumer-public-key.json printf "${JSONFORMAT}\\n" "`cat consumer-cert.pem.line`" > consumer-cert.json printf "${JSONFORMAT}\\n" "`cat consumer-aes.key.line`" > consumer-aes.json printf "${JSONFORMAT}\\n" "`cat daps-consumer.key.line`" > daps-consumer.key.json printf "${JSONFORMAT}\\n" "`cat daps-consumer.cert.line`" > daps-consumer.cert.json 秘密現在採用 JSON 格式，已準備好新增到保存庫。 4. 若要取得取用者保存庫的 Pod 名稱，請執行下列命令： kubectl get pods -n consumer | egrep "vault|NAME"	

任務	Description	所需的技能
	Pod 名稱將類似於 "vault-consumer-0" 。建立轉送至保存庫的連接埠時，會使用此名稱。連接埠轉送可讓您存取保存庫以新增秘密。您應該從已設定 AWS 登入資料的工作站執行此操作。 5. 若要存取保存庫，請使用 kubectl設定連接埠轉送： `kubectl port-forward <VAULT_POD_NAME> 8201:8200 -n consumer` 本機連接埠這次是 8201，因此您可以同時為生產者和消費者設定連接埠轉送。 瀏覽器 您可以使用瀏覽器連線至 http：//localhost：8201/ 以存取取用者保存庫，並依概述使用名稱和內容建立秘密。 包含內容的秘密和檔案如下： • 具有檔案名稱transfer-proxy-token-signer-private-key 的秘密consumer-private-key.pem.line • 具有檔案名稱transfer-proxy-token-signer	

任務	Description	所需的技能
	-public-key 的秘密 consumer-cert.pem. line • 具有檔案名稱transfer-proxy-token-encryption-aes-key 的秘密 consumer-aes.key.line 保存庫 CLI 使用保存庫 CLI，您可以執行下列命令來登入保存庫並建立秘密： 1. 使用您在 中設定的字符登入保存庫consumer_edc.yml ： ``` vault login -address=http://127.0.0.1:8201 ``` 使用正確的字符，您應該會看到訊息 "Success! You are now authenticated." 2. 若要使用您先前建立的 JSON 格式檔案建立秘密，請執行下列程式碼： ``` vault kv put -address=http://127.0.0.1:8201 secret/transfer-proxy-token-signer-p	

任務	Description	所需的技能
	rivate-key @consumer -private-key.json vault kv put - address=http://12 7.0.0.1:8201 secret/ transfer-proxy-token -signer-public-key @consumer-cert.json vault kv put -address= http://127.0.0.1:8 201 secret/transfer- proxy-token-encrypti on-aes-key @consumer -aes.json vault kv put -address= http://127.0.0.1:8 201 secret/daps- private-key @daps-con sumer.key.json vault kv put - address=http://12 7.0.0.1:8201 secret/ daps-public-key @daps-consumer.cer t.json	

設定 HTTP 用戶端以與連接器的管理 API 互動

任務	Description	所需的技能
設定連接埠轉送。	1. 若要檢查 Pod 的狀態，請執行下列命令： kubectl get pods -n provider kubectl get pods -n consumer	DevOps 工程師

任務	Description	所需的技能

	2. 若要確保 Kubernetes 部署成功，請執行下列命令來查看供應商和消費者 Kubernetes Pod 的日誌：
	`kubectl logs -n provider <producer control plane pod name>`
	`kubectl logs -n consumer <consumer control plane pod name>`
	叢集為私有，無法公開存取。若要與連接器互動，請使用 Kubernetes 連接埠轉送功能，將機器產生的流量轉送至連接器控制平面。
	1. 在第一個終端機上，透過連接埠 8300 將消費者的請求轉送至管理 API：
	`kubectl port-forward deployment/consumer-tractusx-connector-controlplane 8300:8081 -n consumer`
	2. 在第二個終端機上，透過連接埠 8400 將提供者的請求轉送至管理 API：
	`kubectl port-forward deployment/provider-tractusx-connect`	

任務	Description	所需的技能
	`or-controlplane 8400:8081 -n provider`	

任務	Description	所需的技能
為提供者和消費者建立 S3 儲存貯體。	EDC 連接器目前不使用臨時 AWS 登入資料，例如擔任角色提供的登入資料。EDC 僅支援使用 IAM 存取金鑰 ID 和私密存取金鑰組合。 後續步驟需要兩個 S3 儲存貯體。一個 S3 儲存貯體用於儲存提供者提供的資料。另一個 S3 儲存貯體用於消費者接收的資料。 IAM 使用者應具有僅在兩個具名儲存貯體中讀取和寫入物件的許可。 需要建立存取金鑰 ID 和私密存取金鑰對並保持安全。停用此 MVDS 之後，應該刪除 IAM 使用者。 下列程式碼是使用者的 IAM 政策範例： `{ "Version": "2012-10-17", "Statement": [ { "Sid": "Stmt1708699805237", "Action": [ "s3:GetObject", "s3:GetObjectVersion", "s3:ListAllMyBuckets",`	DevOps 工程師

任務	Description	所需的技能
	"s3:ListB ucket", "s3:ListB ucketMultipartUplo ads", "s3:ListB ucketVersions", "s3:PutObject"], "Effect": "Allow", "Resource": ["arn:aws: s3:::<S3 Provider Bucket>", "arn:aws: s3:::<S3 Consumer Bucket>", "arn:aws: s3:::<S3 Provider Bucket>/*", "arn:aws: s3:::<S3 Consumer Bucket>/*"] }] }	
設定 Postman 以與連接器互動。	您現在可以透過 EC2 執行個體與連接器互動。使用 Postman 做為 HTTP 用戶端，並為提供者和消費者連接器提供 Postman 集合。 將集合從aws-pattern-edc儲存庫匯入您的 Postman 執行個體。 此模式使用 Postman 集合變數，為您的請求提供輸入。	應用程式開發人員、資料工程師

透過連接器提供公司 X 碳排放足跡資料

任務	Description	所需的技能
準備要共用的碳排放強度資料。	首先，您需要決定要共用的資料資產。公司 X 的資料代表其機群的碳排放足跡。權重是以公噸為單位的總車輛權重 (GVW)，而根據 Wheel-to-Well (WTW) 測量，排放量是以每公噸公里 CO2 的克數 (g CO2 e/t-km) 為單位： • 車輛類型：Van；重量：< 3.5；排放：800 • 車輛類型：城市卡車；重量：3.5–7.5；排放：315 • 車輛類型：中型商品車輛 (MGV)；重量：7.5–20；排放量：195 • 車輛類型：重型貨物車輛 (HGV)；重量：> 20；排放：115 範例資料位於 `aws-patterns-edc` 儲存庫的 `carbon_emissions_data.json` 檔案中。 X 公司使用 Amazon S3 來存放物件。 建立 S3 儲存貯體，並將範例資料物件存放在該處。下列命令會使用預設安全設定建立 S3 儲存貯體。我們強烈建議您參	資料工程師、應用程式開發人員

任務	Description	所需的技能

	閱 Amazon S3 的安全最佳實務。
	
	`aws s3api create-bucket <BUCKET_NAME> --region <AWS_REGION>`
	`# You need to add '--create-bucket-configuration # LocationConstraint=<AWS_REGION>' if you want to create # the bucket outside of us-east-1 region`
	
	`aws s3api put-object --bucket <BUCKET_NAME> \ --key <S3 OBJECT NAME> \ --body <PATH OF THE FILE TO UPLOAD>`
	
	S3 儲存貯體名稱應該是全域唯一的。如需命名規則的詳細資訊，請參閱 AWS 文件。	

任務	Description	所需的技能
使用 Postman 將資料資產註冊到供應商的連接器。	EDC 連接器資料資產會保留資料的名稱及其位置。在此情況下，EDC 連接器資料資產會指向 S3 儲存貯體中建立的物件： • 連接器：供應商 • 請求：建立資產 • 集合變數：更新 ASSET_NAME 。選擇代表資產的有意義的名稱。 • 請求內文：使用您為提供者建立的 S3 儲存貯體更新請求內文。 `"dataAddress": { "edc:type": "AmazonS3", "name": "Vehicle Carbon Footprint", "bucketName": "<REPLACE WITH THE SOURCE BUCKET NAME>", "keyName": "<REPLACE WITH YOUR OBJECT NAME>", "region": "<REPLACE WITH THE BUCKET REGION>", "accessKeyId": "<REPLACE WITH YOUR ACCESS KEY ID>", "secretAccessKey": "<REPLACE WITH SECRET ACCESS KEY>" }`	應用程式開發人員、資料工程師

任務	Description	所需的技能
	• 回應：成功請求會傳回新建立資產的建立時間和資產 ID。 `{ "@id": "c89aa31c-ec4c-44ed-9e8c-1647f19d7583" }` • 集合變數 ASSET_ID：使用建立後由 EDC 連接器自動產生的 `ASSET_ID` ID 更新 Postman 集合變數。	

定義資產的使用政策。	EDC 資料資產必須與明確的使用政策相關聯。首先，在供應商連接器中建立政策定義。
	
	公司 X 的政策是允許資料空間的參與者使用碳排放足跡資料。
	
	• 請求內文：
	  • 連接器：供應商
	  • 請求：建立政策
	  • 集合變數：使用政策的名稱更新`Policy Name`變數。
	• 回應：成功請求會傳回新建立政策的建立時間和政策 ID。使用 EDC 連接器在建立後產生的政策 `POLICY_ID` ID 更新集合變數。
		應用程式開發人員、資料工程師

任務	Description	所需的技能
定義資產的 EDC 合約優惠及其使用政策。	若要允許其他參與者請求存取您的資料，請在指定使用條件和許可的合約中提供資料： • 連接器：供應商 • 請求：建立合約定義 • 集合變數：使用合約優惠或定義的名稱更新`Contract Name`變數。	應用程式開發人員、資料工程師

探索資產並就定義的合約達成協議

任務	Description	所需的技能
請求公司 X 共用的資料目錄。	身為資料空間中的資料取用者，Y 公司需要先探索其他參與者共用的資料。 在此基本設定中，您可以要求取用者連接器直接從提供者連接器請求可用資產的目錄來執行此操作。 • 連接器：消費者 • 請求：請求目錄 • 回應：來自供應商的所有可用資料資產及其連接的用量政策。身為資料消費者，請尋找您感興趣的合約，並相應地更新下列集合變數。 • `CONTRACT_OFFER_ID` – 消費者想要協商的合約優惠 ID	應用程式開發人員、資料工程師

任務	Description	所需的技能
	• ASSET_ID – 消費者想要交涉的資產 ID • PROVIDER_CLIENT_ID – 要與 交涉的提供者連接器 ID	
從公司 X 啟動碳排放強度資料的合約溝通。	現在您已識別要使用的資產，請在消費者和供應商連接器之間啟動合約溝通程序。 • 連接器：消費者 • 請求：合約溝通 • 集合變數：使用消費者連接器的 ID 更新CONSUMER_CLIENT_ID 變數以進行交涉。 程序可能需要一些時間才能達到 VERIFIED 狀態。 您可以使用 Get Negotiation 請求來檢查合約協商的狀態和對應的協議 ID。	應用程式開發人員、資料工程師

使用合約協議來使用資料

任務	Description	所需的技能
使用來自 HTTP 端點的資料。	（選項 1) 若要使用 HTTP 資料平面來取用資料空間中的資料，您可以使用 webhook.site 模擬 HTTP 伺服器，並在取用者連接器中啟動傳輸程序：	應用程式開發人員、資料工程師

任務	Description	所需的技能

	• 連接器：消費者
	• 請求：合約溝通
	• 集合變數：使用 EDC 連接器產生的合約協議 ID 更新`Contract Agreement ID`變數。
	• 請求內文：更新請求內文，將 指定`HTTP`為 Webhook URL `dataDestination` 旁的 ：
	`{ "dataDestination": { "type": "HttpProxy" }, "privateProperties": { "receiverHttpEndpoint": "<WEBHOOK URL>" } }`
	連接器會將下載檔案所需的資訊直接傳送到 Webhook URL。
	收到的承載類似下列內容：
	`{ "id": "dcc90391-3819-4b54-b401-1a005a029b78", "endpoint": "http://consumer-tractusx-connector-`	

任務	Description	所需的技能

	<pre>dataplane.consumer
	:8081/api/public",
	  "authKey":
	 "Authorization",
	  "authCode": "<AUTH
	 CODE YOU RECEIVE IN
	 THE ENDPOINT>",
	  "properties": {
	  "https://
	w3id.org/edc/v0.0.
	1/ns/cid": "vehicle-
	carbon-footprint-c
	ontract:4563abf7-5
	dc7-4c28-bc3d-97f4
	5e32edac:b073669b-
	db20-4c83-82df-46b
	583c4c062"
	  }
	}</pre>
	使用收到的登入資料來取得提供者共用的 S3 資產。
	
	在此最後一個步驟中，您必須將請求傳送至取用者資料平面（正確轉送連接埠），如承載 () 中所述endpoint。	

任務	Description	所需的技能
直接使用來自 S3 儲存貯體的資料。	（選項 2) 使用 Amazon S3 與 EDC 連接器整合，並直接指向消費者基礎設施中的 S3 儲存貯體做為目的地： • 請求內文：更新請求內文，將 S3 儲存貯體指定為 dataDestination。 這應該是您先前為儲存消費者接收的資料而建立的 S3 儲存貯體。 `{ "dataDestination": { "type": "AmazonS3", "bucketName": "{{ REPLACE WITH THE DESTINATION BUCKET NAME }}", "keyName": "{{ REPLACE WITH YOUR OBJECT NAME }}", "region": "{{ REPLACE WITH THE BUCKET REGION }}", "accessKeyId": "{{ REPLACE WITH YOUR ACCESS KEY ID }}", "secretAccessKey": "{{ REPLACE WITH SECRET ACCESS KEY }}" } } }`	資料工程師、應用程式開發人員

疑難排解

問題	解決方案
連接器可能會引發憑證 PEM 格式的問題。	新增 ，將每個檔案的內容串連至單一行\n。

相關資源

- DSSC
- 為永續性使用案例建置資料空間 （採用 Think-it 的 AWS 規範指引策略）
- 資料空間的 AWS
- Tractus-X 文件
- DAPS
- 透過資料空間和 AWS 啟用資料共用 （部落格文章）

其他資訊

資料空間規格

參與者

參與者	公司的描述	公司的重點
公司 X	營運橫跨歐洲和南美洲的機群，以運送各種貨物。	旨在做出資料驅動型決策，以減少其碳排放足跡強度。
公司 Y	環境監管機構	強制執行環境法規和政策，旨在監控和減輕企業和產業的環境影響，包括碳排放強度。

商業案例

X 公司使用資料空間技術與合規稽核人員 Y 公司共用碳足跡資料，以評估和解決 X 公司物流營運對環境的影響。

資料空間授權機構

資料空間授權機構是管理資料空間之組織的聯盟。在此模式中，公司 X 和公司 Y 都會形成控管機構，並代表聯合資料空間授權單位。

資料空間元件

元件	選擇實作	其他資訊
資料集交換通訊協定	資料空間通訊協定 0.8 版	• JSON-LD • Data Catalog Vocabulary (DCAT)
資料空間連接器	Tractus-X EDC Connector 0.4.1 版	• EDC 擴充功能
資料交換政策	預設使用政策	• 開啟數位權利語言 (ODRL)

資料空間服務

服務	實作	其他資訊
身分服務	動態屬性佈建系統 (DAPS)	「動態屬性佈建系統 (DAPS) 旨在確定組織和連接器的特定屬性。因此，第三方不需要信任後者，前提是他們信任 DAPS 聲明。」 — DAPS 為了專注於連接器的邏輯，資料空間會使用 Docker Compose 部署在 Amazon EC2 機器上。
探索服務	Gaia-X 聯合目錄	「聯合目錄構成 Gaia-X Self-Descriptions 的索引儲存庫，以啟用供應商及其服務項目的探索和選擇。自我描述是參與者以屬性和宣告的形式提供有

關自己及其服務的資訊。」 — Gaia-X 生態系統 Kickstarter

要交換的資料

資料資產	Description	格式
碳排放資料	來自整個機群之指定區域（歐洲和南美洲）中不同車輛類型的強度值	JSON 檔案

資料模型

```
{
  "region": "string",
  "vehicles": [
// Each vehicle type has its Gross Vehicle Weight (GVW) category and its emission
 intensity in grams of CO2 per Tonne-Kilometer (g CO2 e/t-km) according to the "Well-
to-Wheel" (WTW) measurement.
    {
      "type": "string",
      "gross_vehicle_weight": "string",
      "emission_intensity": {
        "CO2": "number",
        "unit": "string"
      }
    }
  ]
}
```

Tractus-X EDC 連接器

如需每個 Tractus-X EDC 參數的文件，請參閱原始值檔案。

下表列出所有 服務，以及其對應的公開連接埠和端點以供參考。

服務名稱	連接埠和路徑

控制平面	• 管理：– 連接埠：8081 路徑： `/management` 控制 – 連接埠：8083 路徑： `/control` • 通訊協定連接埠：8084 路徑： `/api/v1/dsp` • 指標 – 連接埠：9090 路徑： `/metrics` • 可觀測性 – 連接埠：8085 路徑： `/observability`
資料平面	預設 – 連接埠：8080 路徑： `/api` public – 連接埠：8081 路徑： `/api/dataplane/control` Proxy – 連接埠：8186 路徑： `/proxy` 指標 – 連接埠：9090 路徑： `/metrics` 可觀測性 – 連接埠：8085 路徑： `/observability`
保存庫	連接埠：8200
PostgreSQL	連接埠：5432

使用 AWS Secrets Manager Manager

您可以使用 Secrets Manager 而非 HashiCorp Vault 作為秘密管理員。若要這樣做， 必須使用或建置 AWS Secrets Manager EDC 延伸模組。

您將負責建立和維護自己的映像，因為 Tractus-X 不支援 Secrets Manager。

若要達成此目的，您需要透過引入 AWS Secrets Manager EDC 延伸來修改控制平面和連接器資料平面的建置 Gradle 檔案（請參閱此平面成品的範例），然後建置、維護和參考 Docker 映像。

如需重構 Tractus-X 連接器 Docker 映像的更多洞見，請參閱重構 Tractus-X EDC Helm Chart。

為了簡單起見，我們避免在此模式中重建連接器映像，並使用 HashiCorp Vault。

使用純量 Python UDF 設定 Amazon Redshift 查詢結果的語言特定排序

Ethan Stark，Amazon Web Services

總結

此模式提供使用純量 Python UDF （使用者定義的函數） 設定 Amazon Redshift 查詢結果不區分大小寫語言排序的步驟和範例程式碼。必須使用純量 Python UDF，因為 Amazon Redshift 會根據二進位 UTF-8 排序傳回結果，且不支援語言特定的排序。Python UDF 是非 SQL 處理程式碼，以 Python 程式為基礎，並在資料倉儲中執行。您可以在單一查詢中使用 SQL 陳述式執行 Python UDF 程式碼。如需詳細資訊，請參閱 Amazon Redshift 大數據部落格文章中的 Python UDFs 簡介 AWS 。

此模式中的範例資料是根據土耳其字母進行示範。此模式中的純量 Python UDF 旨在使 Amazon Redshift 的預設查詢結果符合土耳其文字元的語言順序。如需詳細資訊，請參閱此模式額外資訊區段中的土耳其文語言範例。您可以針對其他語言，以此模式修改純量 Python UDF。

先決條件和限制

先決條件

- 具有資料庫、結構描述和資料表的 Amazon Redshift 叢集
- 具有 `CREATE TABLE`和 `CREATE FUNCTION`許可的 Amazon Redshift 使用者
- Python 2.7 或更新版本

限制

此模式中查詢所使用的語言排序不區分大小寫。

工具

AWS 服務

- Amazon Redshift 是 中的受管 PB 級資料倉儲服務 AWS 雲端。Amazon Redshift 已與您的資料湖整合，可讓您使用資料為您的企業和客戶取得新的洞見。

其他工具

- Python (UDFs) 使用者定義的函數是您可以在 Python 中撰寫，然後在 SQL 陳述式中呼叫 的函數。

史詩

開發程式碼以語言順序排序查詢結果

任務	Description	所需的技能

為您的範例資料建立資料表。	若要在 Amazon Redshift 中建立資料表並將範例資料插入資料表，請使用下列 SQL 陳述式：
	`CREATE TABLE my_table (first_name varchar(30));`
	
	`INSERT INTO my_table (first_name) VALUES ('ali'), ('Ali'), ('ırmak'), ('IRMAK'), ('irem'), ('İREM'), ('oğuz'), ('OĞUZ'), ('ömer'), ('ÖMER'), ('sedat'), ('SEDAT'), ('şule'),`
	
	Note
	範例資料中的名字包含土耳其字母的特殊字元。如需此範例土耳其文考量事項的詳細資訊，請參閱此模式額外
		資料工程師

任務	Description	所需的技能
	資訊區段中的土耳其文語言範例。	

檢查範例資料的預設排序。	若要在 Amazon Redshift 中查看範例資料的預設排序，請執行下列查詢：
	`SELECT first_name FROM my_table ORDER BY first_name;`
	查詢會從您先前建立的資料表傳回名字清單：
	`first_name`
	`---------------`
	`Ali`
	`IRMAK`
	`OĞUZ`
	`SEDAT`
	`ali`
	`irem`
	`oğuz`
	`sedat`
	`ÖMER`
	`ömer`
	`İREM`
	`ırmak`
	`ŞULE`
	`şule`
	查詢結果的順序不正確，因為預設的二進位 UTF-8 排序並不符合土耳其文特殊字元的語言順序。
		資料工程師

任務	Description	所需的技能
建立純量 Python UDF。	若要建立純量 Python UDF，請使用下列 SQL 程式碼： `CREATE OR REPLACE FUNCTION collate_sort (value varchar) RETURNS varchar IMMUTABLE AS $$ def sort_str(val): import string dictionary = { 'I': 'ı', 'ı': 'h~', 'İ': 'i', 'Ş': 's~', 'ş': 's~', 'Ğ': 'g~', 'ğ': 'g~', 'Ü': 'u~', 'ü': 'u~', 'Ö': 'o~', 'ö': 'o~', 'Ç': 'c~', 'ç': 'c~' } for key, value in dictionary.items(): val = val.replace(key, value) return val.lower() return sort_str(value)`	資料工程師

任務	Description	所需的技能
	`$$ LANGUAGE plpythonu;`	

查詢範例資料。	若要使用 Python UDF 查詢範例資料，請執行下列 SQL 查詢：
	`SELECT first_name FROM my_table ORDER BY collate_sort(first_name);`
	查詢現在會以土耳其文語言順序傳回範例資料：
	`first_name`
	`---------------`
	`ali`
	`Ali`
	`ırmak`
	`IRMAK`
	`irem`
	`İREM`
	`oğuz`
	`OĞUZ`
	`ömer`
	`Ömer`
	`sedat`
	`SEDAT`
	`şule`
	`ŞULE`
		資料工程師

相關資源

- ORDER BY 子句 (Amazon Redshift 文件)
- 建立純量 Python UDF (Amazon Redshift 文件)

其他資訊

土耳其文語言範例

Amazon Redshift 會根據二進位 UTF-8 排序順序傳回查詢結果，而非特定於語言的排序順序。這表示如果您查詢包含土耳其文字元的 Amazon Redshift 資料表，則查詢結果不會根據土耳其文語言的語言順序排序。土耳其文包含六個特殊字元 (ç、ı、ğ、ö、ş 和 ü)，不會顯示在拉丁字母中。這些特殊字元會根據二進位 UTF-8 順序放置在排序結果集的結尾，如下表所示。

二進位 UTF-8 排序	土耳其文語言排序
a	a
b	b
c	c
d	ç (*)
e	d
f	e
g	f
h	g
i	ğ (*)
j	h
k	ı (*)
l	i
m	j
n	k
o	l
p	m

r	n
s	o
t	ö (*)
u	p
v	r
y	s
z	ş (*)
ç (*)	t
ğ (*)	u
ı (*)	ü (*)
ö (*)	v
ş (*)	y
ü (*)	z

> **Note**
>
> 星號 (*) 表示土耳其文的特殊字元。

如上表所示，特殊字元 ç 在土耳其文語言排序中介於 c 和 d 之間，但在 z 之後以二進位 UTF-8 排序顯示。此模式中的純量 Python UDF 使用下列字元取代字典，以對應的拉丁同等字元取代土耳其文特殊字元。

土耳其文特殊字元	拉丁文同等角色
ç	c~
ı	h~

ğ	g~
ö	o~
ş	s~
ü	u~

> **Note**
>
> 波狀符號 (~) 字元會附加到取代其對應土耳其文特殊字元的拉丁文字元結尾。

修改純量 Python UDF 函數

若要從此模式修改純量 Python UDF 函數，讓函數接受定位參數並支援多個交易字典，請使用下列 SQL 程式碼：

```
CREATE OR REPLACE FUNCTION collate_sort (value varchar, locale varchar)
RETURNS varchar
IMMUTABLE
AS
$$
    def sort_str(val):
       import string
       # Turkish Dictionary
       if locale == 'tr-TR':
            dictionary = {
               'I': 'ı',
               'ı': 'h~',
               'İ': 'i',
               'Ş': 's~',
               'ş': 's~',
               'Ğ': 'g~',
               'ğ': 'g~',
               'Ü': 'u~',
               'ü': 'u~',
               'Ö': 'o~',
               'ö': 'o~',
               'Ç': 'c~',
               'ç': 'c~'
```

```
        }
        # German Dictionary
        if locale == 'de-DE':
            dictionary = {
               ....
               ....
        }

        for key, value in dictionary.items():
            val = val.replace(key, value)

        return val.lower()

    return sort_str(value)

$$ LANGUAGE plpythonu;
```

下列範例程式碼示範如何查詢修改後的 Python UDF：

```
SELECT first_name FROM my_table ORDER BY collate_order(first_name, 'tr-TR');
```

訂閱來自不同 AWS 區域中 S3 儲存貯體的事件通知的 Lambda 函數

Sureshonathala，Amazon Web Services

Andrew Preston 和 Arindom Sarkar，無

總結

Amazon Simple Storage Service (Amazon S3) Event Notifications 會發佈 S3 儲存貯體中特定事件的通知（例如，物件建立的事件、物件移除事件或還原物件事件）。您可以使用 AWS Lambda 函數，根據您的應用程式需求處理這些通知。不過，Lambda 函數無法直接從在不同 AWS 區域中託管的 S3 儲存貯體訂閱通知。

此模式的方法會部署廣發性案例，透過為每個區域使用 Amazon S3 Simple Notification Service (Amazon SNS) 主題來處理來自跨區域 S3 儲存貯體的 Amazon S3 通知。這些區域 SNS 主題會將 Amazon S3 事件通知傳送至包含 Lambda 函數的中央區域中的 Amazon Simple Queue Service (Amazon SQS) 佇列。Lambda 函數會訂閱此 SQS 佇列，並根據組織的需求處理事件通知。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在多個區域中現有的 S3 儲存貯體，包括託管 Amazon SQS 佇列和 Lambda 函數的中央區域。
- AWS Command Line Interface (AWS CLI)，已安裝並設定。如需詳細資訊，請參閱 AWS CLI 文件中的安裝、更新和解除安裝 AWS CLI。
- 熟悉 Amazon SNS 中的廣發案例。如需詳細資訊，請參閱 Amazon SNS 文件中的常見 Amazon SNS 案例。 Amazon SNS

Architecture

下圖顯示此模式方法的架構。

該圖顯示以下工作流程：

1. Amazon S3 會將有關 S3 儲存貯體的事件通知（例如，建立的物件、移除的物件或還原的物件）傳送至相同區域中的 SNS 主題。

2. SNS 主題會將事件發佈至中央區域中的 SQS 佇列。
3. SQS 佇列設定為 Lambda 函數的事件來源，並緩衝 Lambda 函數的事件訊息。
4. Lambda 函數會輪詢訊息的 SQS 佇列，並根據您的應用程式需求處理 Amazon S3 事件通知。

技術堆疊

- Lambda
- Amazon SNS
- Amazon SQS
- Amazon S3

工具

- AWS CLI – AWS 命令列界面 (AWS CLI) 是一種開放原始碼工具，可透過命令列 shell 中的命令與 AWS 服務互動。透過最少的組態，您可以從命令提示中執行 AWS CLI 命令，該命令會實作相當於瀏覽器型 AWS 管理主控台所提供功能的功能。
- AWS CloudFormation – AWS CloudFormation 可協助您建立模型並設定 AWS 資源、快速一致地佈建資源，以及在整個生命週期中管理資源。您可以使用範本來描述您的資源及其相依性，並將它們一起啟動和設定為堆疊，而不是個別管理資源。您可以管理和佈建跨多個 AWS 帳戶和 AWS 區域的堆疊。
- AWS Lambda – AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。只需為使用的運算時間支付費用，一旦未執行程式碼，就會停止計費。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 會協調和管理發佈者和用戶端之間的訊息傳遞或傳送，包括 Web 伺服器和電子郵件地址。訂閱者會收到發佈到所訂閱主題的所有訊息，且某一主題的所有訂閱者均會收到相同訊息。
- Amazon SQS – Amazon Simple Queue Service (Amazon SQS) 提供安全、耐用且可用的託管佇列，可讓您整合和解耦分散式軟體系統和元件。Amazon SQS 同時支援標準佇列和 FIFO 佇列。

史詩

在中央區域中建立 SQS 佇列和 Lambda 函數

任務	Description	所需的技能
使用 Lambda 觸發條件建立 SQS 佇列。	登入 AWS 管理主控台，並使用 AWS Lambda 文件中的教學課程使用 Lambda 搭配 Amazon SQS 的指示，在您的中央區域中建立下列資源：AWS Lambda • Lambda 執行角色 • 處理 Amazon S3 事件的 Lambda 函數 • SQS 佇列 **Note** 請務必將 SQS 佇列設定為 Lambda 函數的事件來源。	AWS DevOps，雲端架構師

為每個必要區域中的 S3 儲存貯體建立 SNS 主題並設定事件通知

任務	Description	所需的技能
建立 SNS 主題以接收 Amazon S3 事件通知。	在您要接收 Amazon S3 事件通知的區域中建立 SNS 主題。如需詳細資訊，請參閱 Amazon SNS 文件中的建立 SNS 主題。 Amazon SNS	AWS DevOps，雲端架構師

任務	Description	所需的技能
	Important 請務必記錄 SNS 主題的 Amazon Resource Name (ARN)。	
訂閱 SNS 主題至中央 SQS 佇列。	將您的 SNS 主題訂閱您的中央區域託管的 SQS 佇列。如需詳細資訊，請參閱 Amazon SNS SNS 文件中的訂閱 SNS 主題。	AWS DevOps，雲端架構師

任務	Description	所需的技能
更新 SNS 主題的存取政策。	1. 開啟 Amazon SNS 主控台，選擇主題，然後選擇您先前建立的 SNS 主題。 2. 選擇編輯，然後展開存取政策 - 選用區段。 3. 將下列存取政策連接至 SNS 主題，以允許 Amazon S3 的`sns:publish` 許可，然後選擇儲存： `{ "Version": "2012-10-17", "Statement": [ { "Sid": "0", "Effect": "Allow", "Principal": { "Service": "s3.amazonaws.com" }, "Action": "sns:Publish", "Resource": "arn:aws:sns:us-west-2::s3Events-SNSTopic-us-west-2" } ] }`	AWS DevOps，雲端架構師

任務	Description	所需的技能
設定區域中每個 S3 儲存貯體的通知。	設定區域中每個 S3 儲存貯體的事件通知。如需詳細資訊，請參閱《Amazon S3 文件》中的使用 Amazon S3 主控台啟用和設定事件通知。 Amazon S3 **Note** 在目的地區段中，選擇 SNS 主題，並指定您先前建立之 SNS 主題的 ARN。	AWS DevOps，雲端架構師
針對所有必要的區域重複此史詩。	**Important** 針對您要接收 Amazon S3 事件通知的每個區域重複此史詩中的任務，包括您的中央區域。	AWS DevOps，雲端架構師

相關資源

- 設定存取政策 (Amazon SQS 文件）
- 將 SQS 佇列設定為事件來源 (AWS Lambda 文件）
- 設定 SQS 佇列以啟動 Lambda 函數 (Amazon SQS 文件）
- AWS::Lambda::Function 資源 (AWS CloudFormation 文件）

將資料轉換為 Apache Parquet 的三種 AWS Glue ETL 任務類型

Adnan Alvee、Nitth Govindasivan 和 Karthikeyan Ramachandran，Amazon Web Services

總結

在 Amazon Web Services (AWS) 雲端上，AWS Glue 是全受管擷取、轉換和載入 (ETL) 服務。AWS Glue 可讓您以符合成本效益的方式分類資料、清理資料、擴充資料，並在各種資料存放區和資料串流之間可靠地移動資料。

此模式在 AWS Glue 中提供不同的任務類型，並使用三種不同的指令碼來示範撰寫 ETL 任務。

您可以使用 AWS Glue 在 Python shell 環境中寫入 ETL 任務。您也可以在受管 Apache Spark 環境中使用 Python (PySpark) 或 Scala 建立批次和串流 ETL 任務。為了協助您開始撰寫 ETL 任務，此模式著重於使用 Python shell、PySpark 和 Scala 的批次 ETL 任務。Python shell 任務適用於需要較少運算能力的工作負載。受管 Apache Spark 環境適用於需要高運算能力的工作負載。

Apache Parquet 旨在支援高效的壓縮和編碼機制。它可以加速您的分析工作負載，因為它以單欄式方式存放資料。將資料轉換為 Parquet 可以節省長時間執行時的儲存空間、成本和時間。若要進一步了解 Parquet，請參閱部落格文章 Apache Parquet：如何使用開放原始碼單欄式資料格式成為英雄。

先決條件和限制

先決條件

- AWS Identity and Access Management (IAM) 角色 （ 如果您沒有角色，請參閱其他資訊一節。）

Architecture

目標技術堆疊

- AWS Glue
- Amazon Simple Storage Service (Amazon S3)
- Apache Parquet

自動化和擴展

- AWS Glue 工作流程支援 ETL 管道的完整自動化。

- 您可以變更資料處理單位 (DPUs或工作者類型的數量，以水平和垂直擴展。

工具

AWS 服務

- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Glue 是全受管 ETL 服務，可在各種資料存放區和資料串流之間分類、清理、擴充和移動資料。

其他工具

- Apache Parquet 是一種開放原始碼資料欄導向的資料檔案格式，專為儲存和擷取而設計。

組態

使用下列設定來設定 AWS Glue ETL 的運算能力。若要降低成本，請在執行此模式中提供的工作負載時使用最小設定。

- Python shell – 您可以使用 1 個 DPU 來利用 16 GB 的記憶體，或使用 0.0625 個 DPU 來利用 1 GB 的記憶體。此模式使用 0.0625 DPU，這是 AWS Glue 主控台中的預設值。
- Python 或 Scala for Spark – 如果您在主控台中選擇 Spark 相關任務類型，AWS Glue 預設會使用 10 個工作者和 G.1X 工作者類型。此模式使用兩個工作者，這是允許的最小數量，具有標準工作者類型，足夠且經濟實惠。

下表顯示 Apache Spark 環境的不同 AWS Glue 工作者類型。由於 Python shell 任務不使用 Apache Spark 環境來執行 Python，因此不會包含在資料表中。

	標準	G.1X	G.2X
vCPU	4	4	8
記憶體	16 GB	16 GB	32 GB
磁碟空間	50 GB	64 GB	128 GB

每個工作者的執行器	2	1	1

Code

如需此模式中使用的程式碼，包括 IAM 角色和參數組態，請參閱其他資訊一節。

史詩

上傳資料

任務	Description	所需的技能
將資料上傳至新的或現有的 S3 儲存貯體。	在帳戶中建立或使用現有的 S3 儲存貯體。從附件區段上傳 sample_data.csv 檔案，並記下 S3 儲存貯體和字首位置。	一般 AWS

建立並執行 AWS Glue 任務

任務	Description	所需的技能
建立 AWS Glue 任務。	在 AWS Glue 主控台的 ETL 區段下，新增 AWS Glue 任務。選取適當的任務類型、AWS Glue 版本，以及對應的 DPU/工作者類型和工作者數量。如需詳細資訊，請參閱組態一節。	開發人員、雲端或資料
變更輸入和輸出位置。	複製對應至 AWS Glue 任務的程式碼，並變更您在上傳資料圖示中記下的輸入和輸出位置。	開發人員、雲端或資料
設定參數。	您可以使用其他資訊區段中提供的程式碼片段來設定 ETL 任	開發人員、雲端或資料

任務	Description	所需的技能
	務的參數。AWS Glue 在內部使用四個引數名稱： • `--conf` • `--debug` • `--mode` • `--JOB_NAME` `--JOB_NAME` 參數必須在 AWS Glue 主控台上明確輸入。選擇任務、編輯任務、安全組態、指令碼程式庫和任務參數（選用）。輸入 `--JOB_NAME` 做為索引鍵，並提供值。您也可以使用 AWS Command Line Interface (AWS CLI) 或 AWS Glue API 來設定此參數。此`--JOB_NAME` 參數由 Spark 使用，在 Python shell 環境任務中不需要。 您必須在每個參數名稱`--`之前新增 ；否則，程式碼將無法運作。例如，對於程式碼片段，位置參數必須由 `--input_loc` 和 調用`--output_loc` 。	
執行 ETL 任務。	執行您的任務並檢查輸出。請注意，從原始檔案減少了多少空間。	開發人員、雲端或資料

相關資源

參考

- Apache Spark
- AWS Glue：運作方式
- AWS Glue 定價

教學課程和影片

- 什麼是 AWS Glue？

其他資訊

IAM 角色

建立 AWS Glue 任務時，您可以使用具有下列程式碼片段中所示許可的現有 IAM 角色或新角色。

若要建立新的角色，請使用下列 YAML 程式碼。

```
# (c) 2022 Amazon Web Services, Inc. or its affiliates. All Rights Reserved. This AWS
 Content is provided subject to the terms of the AWS Customer
# Agreement available at https://aws.amazon.com/agreement/ or other written agreement
 between Customer and Amazon Web Services, Inc.

AWSTemplateFormatVersion: "2010-09-09"

Description: This template will setup IAM role for AWS Glue service.

Resources:
  rGlueRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: "Allow"
            Principal:
              Service:
                - "glue.amazonaws.com"
            Action:
```

```
              - "sts:AssumeRole"
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AWSGlueServiceRole
      Policies:
        - PolicyName: !Sub "${AWS::StackName}-s3-limited-read-write-inline-policy"
          PolicyDocument:
            Version: "2012-10-17"
            Statement:
              - Effect: Allow
                Action:
                  - "s3:PutObject"
                  - "s3:GetObject"
                Resource: "arn:aws:s3:::*/*"
      Tags:
        - Key  : "Name"
          Value : !Sub "${AWS::StackName}"

Outputs:
  oGlueRoleName:
    Description: AWS Glue IAM role
    Value:
      Ref: rGlueRole
    Export:
      Name: !Join [ ":", [ !Ref "AWS::StackName", rGlueRole ] ]
```

AWS Glue Python Shell

Python 程式碼使用 Pandas 和 PyArrow 程式庫將資料轉換為 Parquet。Pandas 程式庫已可供使用。執行模式時會下載 PyArrow 程式庫，因為它是一次性執行。您可以使用 wheel 檔案將 PyArrow 轉換為程式庫，並以程式庫套件的形式提供檔案。如需封裝 wheel 檔案的詳細資訊，請參閱提供您自己的 Python 程式庫。

AWS Glue Python shell 參數

```
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["input_loc", "output_loc"])
```

AWS Glue Python shell 程式碼

```
from io import BytesIO
import pandas as pd
```

```
import boto3
import os
import io
import site
from importlib import reload
from setuptools.command import easy_install
install_path = os.environ['GLUE_INSTALLATION']
easy_install.main( ["--install-dir", install_path, "pyarrow"] )
reload(site)
import pyarrow


input_loc = "s3://bucket-name/prefix/sample_data.csv"
output_loc = "s3://bucket-name/prefix/"


input_bucket = input_loc.split('/', 1)[0]
object_key = input_loc.split('/', 1)[1]


output_loc_bucket = output_loc.split('/', 1)[0]
output_loc_prefix = output_loc.split('/', 1)[1]


s3 = boto3.client('s3')
obj = s3.get_object(Bucket=input_bucket, Key=object_key)
df = pd.read_csv(io.BytesIO(obj['Body'].read()))


parquet_buffer = BytesIO()
s3_resource = boto3.resource('s3')
df.to_parquet(parquet_buffer, index=False)
s3_resource.Object(output_loc_bucket, output_loc_prefix +  'data' +
 '.parquet').put(Body=parquet_buffer.getvalue())
```

使用 Python 的 AWS Glue Spark 任務

若要搭配 Python 使用 AWS Glue Spark 任務類型，請選擇 Spark 做為任務類型。選擇 Spark 3.1、Python 3 搭配改善的任務啟動時間 (Glue 3.0 版） 做為 AWS Glue 版本。

AWS Glue Python 參數

```
from awsglue.utils import getResolvedOptions
```

```
args = getResolvedOptions(sys.argv, ["JOB_NAME", "input_loc", "output_loc"])
```

使用 Python 程式碼的 AWS Glue Spark 任務

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.transforms import *
from awsglue.dynamicframe import DynamicFrame
from awsglue.utils import getResolvedOptions
from awsglue.job import Job


sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)

input_loc = "s3://bucket-name/prefix/sample_data.csv"
output_loc = "s3://bucket-name/prefix/"

inputDyf = glueContext.create_dynamic_frame_from_options(\
    connection_type = "s3", \
    connection_options = {
        "paths": [input_loc]}, \
    format = "csv",
    format_options={
        "withHeader": True,
        "separator": ","
    })


outputDF = glueContext.write_dynamic_frame.from_options(\
    frame = inputDyf, \
    connection_type = "s3", \
    connection_options = {"path": output_loc \
        }, format = "parquet")
```

對於大量壓縮大型檔案（例如 1，000 個檔案，每個檔案約為 3 MB)，請使用 compressionType 參數與 recurse 參數來讀取字首內可用的所有檔案，如下列程式碼所示。

```
input_loc = "bucket-name/prefix/"
```

```
output_loc = "bucket-name/prefix/"

inputDyf = glueContext.create_dynamic_frame_from_options(
                    connection_type = "s3",
                    connection_options = {"paths": [input_loc],
                                          "compressionType":"gzip","recurse" :"True",
                                          },
                    format = "csv",
                    format_options={"withHeader": True,"separator": ","}
                    )
```

對於大量壓縮的小型檔案（例如，各約 133 KB 的 1，000 個檔案），請使用 groupFiles 參數，以及 compressionType和 recurse 參數。groupFiles 參數會將小型檔案分組為多個大型檔案，而 groupSize 參數會以位元組為單位（例如 1 MB) 將分組控制為指定的大小。下列程式碼片段提供在程式碼中使用這些參數的範例。

```
input_loc = "bucket-name/prefix/"
output_loc = "bucket-name/prefix/"

inputDyf = glueContext.create_dynamic_frame_from_options(
                    connection_type = "s3",
                    connection_options = {"paths": [input_loc],
                                          "compressionType":"gzip","recurse" :"True",
                                           "groupFiles" :"inPartition",
 "groupSize" :"1048576",
                                          },
                    format = "csv",
                    format_options={"withHeader": True,"separator": ","}
                    )
```

如果工作者節點沒有任何變更，這些設定可讓 AWS Glue 任務讀取多個檔案（無論有無壓縮），並以 Parquet 格式寫入目標。

使用 Scala 的 AWS Glue Spark 任務

若要搭配 Scala 使用 AWS Glue Spark 任務類型，請選擇 Spark 做為任務類型，選擇語言做為 Scala。選擇 Spark 3.1、Scala 2 搭配改善的任務啟動時間 (Glue 3.0 版） 做為 AWS Glue 版本。為了節省儲存空間，下列具有 Scala 範例的 AWS Glue 也會使用 applyMapping功能來轉換資料類型。

AWS Glue Scala 參數

```
import com.amazonaws.services.glue.util.GlueArgParser val args =
 GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME", "inputLoc",
 "outputLoc").toArray)
```

具有 Scala 程式碼的 AWS Glue Spark 任務

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.DynamicFrame
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueScalaApp {
  def main(sysArgs: Array[String]) {

    @transient val spark: SparkContext = SparkContext.getOrCreate()
    val glueContext: GlueContext = new GlueContext(spark)

    val inputLoc = "s3://bucket-name/prefix/sample_data.csv"
    val outputLoc = "s3://bucket-name/prefix/"

    val readCSV = glueContext.getSource("csv", JsonOptions(Map("paths" ->
 Set(inputLoc)))).getDynamicFrame()

    val applyMapping = readCSV.applyMapping(mappings = Seq(("_c0", "string", "date",
 "string"), ("_c1", "string", "sales", "long"),
    ("_c2", "string", "profit", "double")), caseSensitive = false)

    val formatPartition = applyMapping.toDF().coalesce(1)

    val dynamicFrame = DynamicFrame(formatPartition, glueContext)

    val dataSink = glueContext.getSinkWithFormat(
        connectionType = "s3",
        options = JsonOptions(Map("path" -> outputLoc )),
        transformationContext = "dataSink", format =
 "parquet").writeDynamicFrame(dynamicFrame)
  }
```

```
}
```

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Amazon Athena 和 Amazon QuickSight 視覺化 Amazon Redshift 稽核日誌

Sanket Sirsikar 和 Gopal Krishna Bhatia，Amazon Web Services

總結

安全性是 Amazon Web Services (AWS) 雲端上資料庫操作不可或缺的一部分。您的組織應確保監控資料庫使用者活動和連線，以偵測潛在的安全事件和風險。此模式可協助您監控資料庫以進行安全性和疑難排解，也就是稱為資料庫稽核的程序。

此模式提供 SQL 指令碼，可自動在 Amazon QuickSight 中建立報告儀表板的 Amazon Athena 資料表和檢視，協助您稽核 Amazon Redshift 日誌。 Amazon QuickSight 這可確保負責監控資料庫活動的使用者可以方便地存取資料安全功能。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 現有的 Amazon Redshift 叢集。如需詳細資訊，請參閱 Amazon Redshift 文件中的建立 Amazon Redshift 叢集。
- 存取現有的 Athena 工作群組。如需詳細資訊，請參閱 Amazon Athena 文件中的工作群組運作方式。
- 具有必要 AWS Identity and Access Management (IAM) 許可的現有 Amazon Simple Storage Service (Amazon S3) 來源儲存貯體。如需詳細資訊，請參閱 Amazon Redshift 文件中的資料庫稽核記錄的 Amazon Redshift 稽核記錄儲存貯體許可。 https://docs.aws.amazon.com/redshift/latest/mgmt/db-auditing.html

Architecture

技術堆疊

- Athena
- Amazon Redshift
- Amazon S3

- QuickSight

工具

- Amazon Athena – Athena 是一種互動式查詢服務，可讓您使用標準 SQL 輕鬆分析 Amazon S3 中的資料。
- Amazon QuickSight – QuickSight 是可擴展、無伺服器、可內嵌、採用機器學習的商業智慧 (BI) 服務。
- Amazon Redshift – Amazon Redshift 是一種企業級的 PB 級全受管資料倉儲服務。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是網際網路的儲存體。

史詩

設定 Amazon Redshift 叢集

任務	Description	所需的技能
啟用 Amazon Redshift 叢集的稽核記錄。	1. 登入 AWS 管理主控台，開啟 Amazon Redshift 主控台，選擇 CLUSTERS，然後選擇您要啟用記錄的叢集。 2. 選擇屬性索引標籤，然後依照 Amazon Redshift 文件中的使用主控台設定稽核的指示啟用稽核。	DBA，資料工程師
在 Amazon Redshift 叢集參數群組中啟用記錄。	您可以使用 AWS 管理主控台、Amazon Redshift API 參考或 AWS Command Line Interface (AWS CLI)，同時啟用連線日誌、使用者日誌和使用者活動日誌的稽核。 若要稽核使用者活動日誌，您必須啟用`enable_us`	DBA，資料工程師

任務	Description	所需的技能
	er_activity_loggin g 資料庫參數。如果您只啟用稽核記錄功能，但不啟用相關聯的參數，資料庫稽核會記錄連線和使用者日誌的記錄資訊，但不會記錄使用者活動日誌。enable_user_activi ty_logging 參數預設為未啟用，但您可以將其從 變更為false來啟用它true。 **Important** 您需要在啟用 參數的情況下建立新的叢集user_acti vity_logging 參數群組，並將其連接至您的 Amazon Redshift 叢集。如需詳細資訊，請參閱 Amazon Redshift 文件中的修改叢集。 如需此任務的詳細資訊，請參閱 Amazon Redshift 文件中的 Amazon Redshift 參數群組和使用主控台設定稽核。	

任務	Description	所需的技能
設定 Amazon Redshift 叢集記錄的 S3 儲存貯體許可。	當您啟用記錄時，Amazon Redshift 會收集記錄資訊，並將其上傳至存放在 S3 儲存貯體中的日誌檔案。您可以使用現有的 S3 儲存貯體或建立新的儲存貯體。 **Important** 確定 Amazon Redshift 具有存取 S3 儲存貯體所需的 IAM 許可。如需詳細資訊，請參閱 Amazon Redshift 文件中的資料庫稽核記錄的 Amazon Redshift 稽核記錄儲存貯體許可。 https://docs.aws.amazon.com/redshift/latest/mgmt/db-auditing.html	DBA，資料工程師

建立 Athena 資料表和檢視

任務	Description	所需的技能
建立 Athena 資料表和檢視，從 S3 儲存貯體查詢 Amazon Redshift 稽核日誌資料。	開啟 Amazon Athena 主控台，並使用 `AuditLogging.sql` SQL 指令碼（已連接）的資料定義語言 (DDL) 查詢來建立使用者活動日誌、使用者日誌和連線日誌的資料表和檢視。	資料工程師

任務	Description	所需的技能
	如需詳細資訊和說明，請參閱 Amazon Athena 研討會中的建立資料表和執行查詢教學課程。	

在 QuickSight 儀表板中設定日誌監控

任務	Description	所需的技能
使用 Athena 作為資料來源建立 QuickSight 儀表板。	開啟 Amazon QuickSight 主控台，並依照 Amazon Athena 研討會中的使用 Athena 視覺化 QuickSight 教學課程中的指示建立 QuickSight 儀表板。	DBA，資料工程師

相關資源

- 在 Athena 中建立資料表並執行查詢
- 使用 Athena 透過 QuickSight 視覺化

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Amazon Quick Sight 視覺化所有 AWS 帳戶的 IAM 登入資料報告

Parag Nagwekar 和 Arun Chandapillai，Amazon Web Services

摘要

警告：IAM 使用者具有長期登入資料，這會造成安全風險。為了協助降低此風險，建議您只為這些使用者提供執行任務所需的許可，並在不再需要這些使用者時將其移除。

您可以使用 AWS Identity and Access Management (IAM) 登入資料報告，協助您滿足組織的安全性、稽核和合規要求。登入資料報告提供 AWS 帳戶中所有使用者的清單，並顯示其登入資料的狀態，例如密碼、存取金鑰和多重驗證 (MFA) 裝置。您可以針對 AWS AWS Organizations 帳戶使用登入資料報告。

此模式包含步驟和程式碼，可協助您使用 Amazon Quick Sight 儀表板建立和共用組織中所有 AWS 帳戶的 IAM 登入資料報告。您可以與組織中的利益相關者共用儀表板。這些報告可協助您的組織實現以下目標業務成果：

- 識別與 IAM 使用者相關的安全事件
- 追蹤 IAM 使用者的即時遷移至單一登入 (SSO) 身分驗證
- 追蹤 IAM 使用者存取的 AWS 區域
- 保持合規
- 與其他利益相關者共用資訊

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 具有成員帳戶的組織
- 具有存取 Organizations 中帳戶許可的 IAM 角色
- 安裝 https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html#getting-started-install-instructions並設定 AWS Command Line Interface (AWS CLI) 第 2 版
- Amazon Quick Enterprise Edition 的訂閱

Architecture

技術堆疊

- Amazon Athena
- Amazon EventBridge
- Amazon Quick Sight
- Amazon Simple Storage Service (Amazon S3)
- AWS Glue
- AWS Identity and Access Management (IAM)
- AWS Lambda
- AWS Organizations

目標架構

下圖顯示用於設定從多個 AWS 帳戶擷取 IAM 憑證報告資料的工作流程的架構。

1. EventBridge 每天叫用 Lambda 函數。
2. Lambda 函數會在組織的每個 AWS 帳戶中擔任 IAM 角色。然後，函數會建立 IAM 登入資料報告，並將報告資料存放在集中式 S3 儲存貯體中。您必須在 S3 儲存貯體上啟用加密和停用公有存取。
3. AWS Glue 爬蟲程式每天爬取 S3 儲存貯體，並相應地更新 Athena 資料表。
4. Quick Sight 會匯入和分析登入資料報告中的資料，並建置儀表板，供利益相關者視覺化和共用。

工具

AWS 服務

- Amazon Athena 是一種互動式查詢服務，可讓您使用標準 SQL 輕鬆分析 Amazon S3 中的資料。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如，Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他 AWS 帳戶中的事件匯流排。
- Amazon Quick Sight 是一種雲端規模的商業智慧 (BI) 服務，可協助您在單一儀表板中視覺化、分析和報告您的資料。Quick Sight 是 Amazon Quick 的核心元件，提供互動式資料視覺化、SPICE 記憶體內分析、內嵌分析和儀表板共用。

- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。

Code

此模式的程式碼可在 GitHub getiamcredsreport-allaccounts-org 儲存庫中使用。您可以使用此儲存庫中的程式碼，跨 Organizations 中的 AWS 帳戶建立 IAM 登入資料報告，並將其存放在中央位置。

史詩

設定基礎設施

任務	Description	所需的技能
設定 Amazon Quick Enterprise 版本。	1. 在您的 AWS 帳戶中啟用 Amazon Quick Enterprise 版本。如需詳細資訊，請參閱 Quick 文件中的管理 Amazon Quick 內的使用者存取權。 2. 若要授予儀表板許可，請取得 Quick Sight 使用者的 Amazon Resource Name (ARN)。	AWS 管理員、AWS DevOps、雲端管理員、雲端架構師
將 Amazon Quick Sight 與 Amazon S3 和 Athena 整合。	在部署 AWS CloudFormation 堆疊之前，您必須授權 Quick Sight 使用 Amazon S3 和 Athena。 AWS CloudFormation	AWS 管理員、AWS DevOps、雲端管理員、雲端架構師

部署基礎設施

任務	Description	所需的技能
複製 GitHub 儲存庫。	1. 執行下列命令，將 GitHub getiamcredsreport-allaccounts-org 儲存庫複製到本機電腦： `git clone https://github.com/aws-samples/getiamcredsreport-allaccounts-org`	AWS 管理員
部署 基礎設施。	1. 登入 AWS 管理主控台並開啟 CloudFormation 主控台。 2. 在導覽窗格中，選擇建立堆疊，然後選擇使用新資源（標準）。 3. 在識別資源頁面上，選擇下一步。 4. 在指定範本頁面上，針對範本來源， 選取上傳範本檔案。 5. 選擇選擇檔案，從複製的 GitHub 儲存庫中選取`Cloudformation-createcredrepo.yaml` 檔案，然後選擇下一步。 6. 在參數中，`IAMRoleName` 使用您的 IAM 角色更新 。這應該是您希望 Lambda 在組織的每個帳戶中擔任的 IAM 角色。此角色會建立登入資料報告。注意：在建立	AWS 管理員

任務	Description	所需的技能
	堆疊的這個步驟中，角色不必存在於所有帳戶中。 7. 在參數中，S3BucketName 使用 S3 儲存貯體的名稱進行更新，其中 Lambda 可以存放所有帳戶的登入資料。 8. 針對堆疊名稱，輸入您的堆疊名稱。 9. 選擇提交。 10請注意 Lambda 函數的角色名稱。	
建立 IAM 許可政策。	使用下列許可，為組織中的每個 AWS 帳戶建立 IAM 政策： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "iam:GenerateCredentialReport", "iam:GetCredentialReport" ], "Resource": "*" } ] }`	AWS DevOps、雲端管理員、雲端架構師、資料工程師

任務	Description	所需的技能
使用信任政策建立 IAM 角色。	1. 為 AWS 帳戶建立 IAM 角色，並連接您在上一個步驟中建立的許可政策。 2. 將下列信任政策連接至 IAM 角色： `{ "Version": "2012-10-17", "Statement":[ { "Effect": "Allow", "Principal":{ "AWS":[ "arn:aws:iam::<MasterAccountID>:role/<LambdaRole>" ] }, "Action": "sts:AssumeRole" } ] }` **Important** `arn:aws:iam::<MasterAccountID>:role/<LambdaRole>` 將取代為您先前記下的 Lambda 角色 ARN。	雲端管理員、雲端架構師、AWS 管理員

任務	Description	所需的技能
	Note 組織通常會使用自動化為其 AWS 帳戶建立 IAM 角色。如果可用，我們建議您使用此自動化。或者，您可以從 程式碼儲存庫使用`CreateRoleforOrg.py` 指令碼。指令碼需要現有的管理角色或任何其他 IAM 角色，這些角色具有在每個 AWS 帳戶中建立 IAM 政策和角色的許可。	
設定 Amazon Quick Sight 以視覺化資料。	1. 使用您的登入資料登入 Quick。 2. 使用 Athena （使用`iamcredreportdb` 資料庫和`"cfn_iamcredreport"` 資料表）建立資料集，然後自動重新整理資料集。 3. 在 Quick Sight 中建立分析。 4. 建立 Quick Sight 儀表板。	AWS DevOps、雲端管理員、雲端架構師、資料工程師

其他資訊

其他考量事項

考慮下列各項：

- 使用 CloudFormation 部署基礎設施後，您可以等待 Athena 在 Amazon S3 中建立並分析報告，直到 Lambda 和 AWS Glue 根據其排程執行。或者，您可以手動執行 Lambda 以取得 Amazon S3 中的報告，然後執行 AWS Glue 爬蟲程式以取得從資料建立的 Athena 資料表。
- Quick 是一種功能強大的工具，可根據您的業務需求分析和視覺化資料。您可以使用 Quick 中的參數，根據您選擇的資料欄位來控制小工具資料。此外，您可以使用快速分析從資料集建立參數（例如，帳戶、日期和使用者欄位`partition_0`，例如 `partition_1`、 和 `user`），以新增帳戶、日期和使用者的參數控制項。
- 若要建置自己的 Quick Sight 儀表板，請參閱 AWS Workshop Studio 網站上的 Quick Workshops。
- 若要查看 Quick Sight 儀表板範例，請參閱 GitHub getiamcredsreport-allaccounts-org 程式碼儲存庫。

目標業務成果

您可以使用此模式來實現下列目標業務成果：

- 識別與 IAM 使用者相關的安全事件 – 使用單一窗格來調查組織中每個 AWS 帳戶的每個使用者。您可以追蹤 IAM 使用者最近存取的個別 AWS 區域及其使用之服務的趨勢。
- 追蹤 IAM 使用者的即時遷移至 SSO 身分驗證 – 透過使用 SSO，使用者可以使用單一登入資料登入一次，並存取多個 AWS 帳戶和應用程式。如果您打算將 IAM 使用者遷移至 SSO，此模式可協助您轉換至 SSO，並追蹤所有 AWS 帳戶的所有 IAM 使用者登入資料用量（例如存取 AWS 管理主控台或存取金鑰的使用）。
- 追蹤 IAM 使用者存取的 AWS 區域 – 您可以基於各種目的控制 IAM 使用者對區域的存取，例如資料主權和成本控制。您也可以追蹤任何 IAM 使用者對 區域的使用。
- 保持合規 – 透過遵循最低權限原則，您只能授予執行特定任務所需的特定 IAM 許可。此外，您也可以追蹤對 AWS 服務、AWS 管理主控台和長期憑證用量的存取。
- 與其他利益相關者共用資訊 – 您可以與其他利益相關者共用精心策劃的儀表板，而無需授予他們存取 IAM 登入資料報告或 AWS 帳戶的權限。

更多模式

- 使用 Amazon Cognito 和 AWS Amplify UI 驗證現有的 React 應用程式使用者
- 使用 Amazon Textract 從 PDF 檔案自動擷取內容
- 在 Amazon SageMaker AI Studio Lab 中將 DeepAR 用於時間序列，以建置冷啟動預測模型
- 使用 AWS Cost Explorer 建立 Amazon EMR 叢集的詳細成本和用量報告
- 建立 Amazon RDS 和 Amazon Aurora 的詳細成本和用量報告
- 使用 AWS Cost Explorer 建立 AWS Glue 任務的詳細成本和用量報告 Cost Explorer
- 使用 Kinesis Data Streams 和 Firehose 搭配 將 DynamoDB 記錄交付至 Amazon S3 AWS CDK
- 使用 AWS CodePipeline CI/CD 管道部署 AWS Glue 任務 AWS CodePipeline
- 使用 Amazon Cognito 和 IaC 自動化將 Amazon Quick Sight 視覺化元件內嵌至 Web 應用程式 IaC
- Amazon DynamoDB 資料表的預估儲存成本
- 使用 Terraform 執行 Amazon Redshift SQL 查詢
- 在資料湖中擷取和查詢 AWS IoT SiteWise 中繼資料屬性
- 在 Quick Sight 中使用 AWS Mainframe Modernization 和 Amazon Q 產生資料洞見
- 使用 Quick Sight 中的 AWS Mainframe Modernization 和 Amazon Q 產生 Db2 z/OS 資料洞見
- 讓 SageMaker 筆記本執行個體暫時存取另一個 AWS 帳戶中的 CodeCommit 儲存庫
- 當 Amazon Data Firehose 資源未使用 AWS KMS 金鑰加密時，識別和提醒
- 將 psycopg2 程式庫匯入 AWS Lambda ，以與您的 PostgreSQL 資料庫互動
- 在 Microsoft Sentinel 中擷取和分析 AWS 安全日誌
- 使用 AWS DMS，以 SSL 模式將 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL
- 使用 Oracle GoldenGate 平面檔案轉接器，將 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫遷移至 Amazon Redshift
- 使用 DistCp 搭配適用於 Amazon S3 的 AWS PrivateLink，將資料從內部部署 Hadoop 環境遷移至 Amazon S3
- 從 Couchbase Server 遷移至 AWS 上的 Couchbase Capella
- 將內部部署 Cloudera 工作負載遷移至 AWS 上的 Cloudera 資料平台
- 使用應用程式復原控制器管理 EMR 叢集的多可用區域容錯移轉
- 使用 IaC 原則自動化 Amazon Aurora 全域資料庫的藍/綠部署
- 使用 GitHub 動作根據 AWS CloudFormation 範本佈建 AWS Service Catalog 產品
- AWS Glue 使用 pytest 架構在 中執行 Python ETL 任務的單元測試

- 設定 AWS ParallelCluster 的 Grafana 監控儀表板
- 將資料從 Amazon Redshift 叢集跨帳戶卸載至 Amazon S3
- 使用 Flask 和 AWS Elastic Beanstalk 視覺化 AI/ML 模型結果

運算

主題

- 容器與微服務
- 無伺服器
- 聯網
- 內容交付

容器與微服務

主題

- 從 Amazon EKS 容器存取 Amazon Neptune 資料庫
- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon ECS 上私下存取容器應用程式
- 使用 AWS Fargate、AWS PrivateLink 和 Network Load Balancer，在 Amazon ECS 上私下存取容器應用程式
- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon EKS 上私下存取容器應用程式
- 使用 AWS Batch 自動化 Amazon RDS for PostgreSQL 資料庫執行個體的備份
- 使用 CI/CD 管道在 Amazon EKS 中自動化節點終止處理常式的部署
- 使用 CI/CD 管道自動建置 Java 應用程式並將其部署到 Amazon EKS
- 跨 AWS 帳戶 和 複製 Amazon ECR 容器映像 AWS 區域
- 使用 Amazon EFS 在 EC2 執行個體上建立 Amazon ECS 任務定義並掛載檔案系統
- 使用容器映像部署 Lambda 函數
- 使用 AWS Fargate 在 Amazon ECS 上部署 Java 微服務
- 在 Amazon S3 中使用 Amazon EKS 和 Helm Chart 儲存庫部署 Kubernetes 資源和套件
- 使用 Terraform 在 Amazon EKS 中部署 CockroachDB 叢集
- 在 Amazon EKS 上部署範例 Java 微服務，並使用 Application Load Balancer 公開微服務
- 在 Amazon EKS 叢集上部署以 gRPC 為基礎的應用程式，並使用 Application Load Balancer 存取它
- 在做為 Docker 容器 AWS IoT Greengrass V2 執行時部署容器化應用程式
- 使用 Elastic Beanstalk 部署容器
- 使用 Lambda 函數、Amazon VPC 和無伺服器架構產生靜態傳出 IP 地址
- 遷移至 Amazon ECR 儲存庫時，自動識別重複的容器映像
- 使用 Kubernetes DaemonSet 在 Amazon EKS 工作者節點上安裝 SSM Agent
- 使用 preBootstrapCommands 在 Amazon EKS 工作者節點上安裝 SSM 代理程式和 CloudWatch 代理程式
- 啟用 Amazon EKS Auto 模式時遷移 NGINX 傳入控制器
- 將您的容器工作負載從 Azure Red Hat OpenShift (ARO) 遷移至 Red Hat OpenShift Service on AWS (ROSA)
- 使用 Amazon ECS Anywhere 在 Amazon WorkSpaces 上執行 Amazon ECS 任務 Amazon ECS Anywhere

- 在 Amazon EC2 Linux 執行個體上執行 ASP.NET Core Web API Docker 容器
- 使用 Amazon EFS on Amazon EKS 搭配 AWS Fargate，以持久性資料儲存來執行具狀態工作負載
- 使用 Amazon EKS Pod Identity 和 KEDA 在 Amazon EKS 中設定事件驅動的自動擴展
- 使用 PGO 在 Amazon EKS 上簡化 PostgreSQL 部署
- 使用 Application Load Balancer 在 Amazon ECS 中使用交互 TLS 簡化應用程式身分驗證
- 更多模式

從 Amazon EKS 容器存取 Amazon Neptune 資料庫

Ramakrishnan Palaninathan，Amazon Web Services

總結

此模式會在全受管圖形資料庫 Amazon Neptune 與容器協同運作服務 Amazon Elastic Kubernetes Service (Amazon EKS) 之間建立連線，以存取 Neptune 資料庫。Neptune 資料庫叢集受限於虛擬私有雲端 (VPC) AWS。因此，存取 Neptune 需要仔細設定 VPC 才能啟用連線。

與 PostgreSQL 的 Amazon Relational Database Service (Amazon RDS) 不同，Neptune 不依賴一般資料庫存取憑證。而是使用 AWS Identity and Access Management (IAM) 角色進行身分驗證。因此，從 Amazon EKS 連線至 Neptune 需要設定具有存取 Neptune 必要許可的 IAM 角色。

此外，Neptune 端點只能在叢集所在的 VPC 內存取。這表示您必須設定網路設定，以促進 Amazon EKS 和 Neptune 之間的通訊。根據您的特定需求和聯網偏好設定，有多種方法來設定 VPC，以啟用 Neptune 和 Amazon EKS 之間的無縫連線。每個方法都有不同的優點和考量，可讓您靈活地設計資料庫架構，以符合應用程式的需求。

先決條件和限制

先決條件

- 安裝最新版本的 kubectl （請參閱說明)。若要檢查您的版本，請執行：

```
kubectl version --short
```

- 安裝最新版本的 eksctl （請參閱說明)。若要檢查您的版本，請執行：

```
eksctl info
```

- 安裝最新版本的 AWS Command Line Interface (AWS CLI) 第 2 版 （請參閱說明)。若要檢查您的版本，請執行：

```
aws --version
```

- 建立 Neptune 資料庫叢集 （請參閱說明)。請務必透過 VPC 對等互連、 或其他方法，在叢集的 VPC 和 Amazon EKS 之間建立通訊。 AWS Transit Gateway此外，請確定叢集的狀態為「可用」，且其對安全群組的連接埠 8182 具有傳入規則。
- 在現有的 Amazon EKS 叢集上設定 IAM OpenID Connect (OIDC) 供應商 （請參閱說明)。

產品版本

- Amazon EKS 1.27
- Amazon Neptune 引擎版本 1.3.0.0 (2023-11-15)

Architecture

下圖顯示 Amazon EKS 叢集中的 Kubernetes Pod 與 Neptune 之間的連線，以提供 Neptune 資料庫的存取權。

自動化和擴展

您可以使用 Amazon EKS Horizontal Pod Autoscaler 來擴展此解決方案。

工具

服務

- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 上執行 Kubernetes， AWS 而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon Neptune 是一種圖形資料庫服務，可協助您建置和執行使用高度連線資料集的應用程式。

最佳實務

如需最佳實務，請參閱《Amazon EKS 最佳實務指南》中的 Identity and Access Management。

史詩

設定環境變數

任務	Description	所需的技能
驗證叢集內容。	使用 Helm 或其他命令列工具與 Amazon EKS 叢集互動之前，您必須定義封裝叢集詳細資訊的環境變數。這些變數用	AWS 管理員、雲端管理員

任務	Description	所需的技能
	於後續命令，以確保它們以正確的叢集和資源為目標。 首先，確認您是在正確的叢集內容中操作。這可確保任何後續命令都傳送至預期的 Kubernetes 叢集。若要驗證目前的內容，請執行下列命令。 `kubectl config current-context`	
定義 CLUSTER_NAME 變數。	定義 Amazon EKS 叢集CLUSTER_NAME 的環境變數。在下列命令中，將範例值取代 AWS 區域 為您的叢集us-west-2 的正確 。將範例值取代eks-workshop 為您現有的叢集名稱。 `export CLUSTER_NAME=$(aws eks describe-cluster --region us-west-2 --name eks-workshop --query "cluster.name" --output text)`	AWS 管理員、雲端管理員
驗證輸出。	若要驗證變數是否已正確設定，請執行下列命令。 `echo $CLUSTER_NAME` 確認此命令的輸出符合您在上一個步驟中指定的輸入。	AWS 管理員、雲端管理員

建立 IAM 角色並將其與 Kubernetes 建立關聯

任務	Description	所需的技能
建立 服務帳戶。	您可以使用服務帳戶的 IAM 角色，將 Kubernetes 服務帳戶映射至 IAM 角色，為在 Amazon EKS 上執行的應用程式啟用精細的許可管理。您可以使用 eksctl 來建立 IAM 角色，並將其與 Amazon EKS 叢集中的特定 Kubernetes 服務帳戶建立關聯。 AWS 受管政策`NeptuneFullAccess` 允許寫入和讀取您指定的 Neptune 叢集。 **Important** 您必須先有與叢集相關聯的 OIDC 端點，才能執行這些命令。 建立您要與名為 的 AWS 受管政策建立關聯的服務帳戶`NeptuneFullAccess` 。 `eksctl create iamservic eaccount --name eks- neptune-sa --namespa ce default --cluster $CLUSTER_NAME -- attach-policy-arn arn:aws:iam::aws:p olicy/NeptuneFullA ccess --approve --`	AWS 管理員、雲端管理員

任務	Description	所需的技能

	`override-existing-serviceaccounts`
	
	其中 `eks-neptune-sa` 是您要建立的服務帳戶名稱。
	
	完成後，此命令會顯示下列回應：
	
	`2024-02-07 01:12:39 [#] created serviceaccount "default/eks-neptune-sa"`	

任務	Description	所需的技能

確認帳戶已正確設定。	請確定已在叢集的預設命名空間中正確設定eks-neptune-sa 服務帳戶。
	`kubectl get sa eks-neptune-sa -o yaml`
	輸出應如下所示：
	`apiVersion: v1`
	`kind: ServiceAccount`
	`metadata:`
	`  annotations:`
	`  eks.amazonaws.com/role-arn: arn:aws:iam::123456789123:role/eksctl-eks-workshop-addon-iamserviceaccount-d-Role1-Q35yKgdQOlmM`
	`  creationTimestamp: "2024-02-07T01:12:39Z"`
	`  labels:`
	`  app.kubernetes.io/managed-by: eksctl`
	`  name: eks-neptune-sa`
	`  namespace: default`
	`  resourceVersion: "5174750"`
	`  uid: cd6ba2f7-a0f5-40e1-a6f4-4081e0042316`
		AWS 管理員、雲端管理員

任務	Description	所需的技能

檢查連線能力。	部署名為 的範例 Pod，pod-util並檢查與 Neptune 的連線。
	
	`apiVersion: v1`
	`kind: Pod`
	`metadata:`
	`  name: pod-util`
	`  namespace: default`
	`spec:`
	`  serviceAccountName: eks-neptune-sa`
	`  containers:`
	`  - name: pod-util`
	`  image: public.ecr.aws/patrickc/troubleshoot-util`
	`  command:`
	`  - sleep`
	`  - "3600"`
	`  imagePullPolicy: IfNotPresent`
	
	`kubectl apply -f pod-util.yaml`
	
	`kubectl exec --stdin --tty pod-util -- /bin/bash`
	`bash-5.1# curl -X POST -d '{"gremlin":"g.V().limit(1)"}' https://db-neptune-1.cluster-xxxxxxxxxxxx.us-west-2.neptune.amazonaws.com:8182/gremlin`
	`{"requestId":"a4964f2d-12b1-4ed3-8a14-e`
		AWS 管理員、雲端管理員

任務	Description	所需的技能

	<pre>ff511431a0e","stat
	us":{"message":"",
	"code":200,"attrib
	utes":{"@type":"g:
	Map","@value":[]}}
	,"result":{"data":
	{"@type":"g:List",
	"@value":[]},"meta
	":{"@type":"g:Map"
	,"@value":[]}}}
	bash-5.1# exit
	exit</pre>	

驗證連線活動

任務	Description	所需的技能
啟用 IAM 資料庫身分驗證。	根據預設，當您建立 Neptune 資料庫叢集時，IAM 資料庫身分驗證會停用。您可以使用 啟用或停用 IAM 資料庫身分驗證 AWS 管理主控台。 請依照 AWS 文件中的步驟，在 Neptune 中啟用 IAM 資料庫身分驗證。	AWS 管理員、雲端管理員
驗證連線。	在此步驟中，您會與已處於執行狀態的`pod-util`容器互動，以安裝 awscurl 並驗證連線。 1. 執行下列命令來尋找 Pod。 <pre>kubectl get pods</pre> 輸出應如下所示：	AWS 管理員、雲端管理員

任務	Description	所需的技能

	<pre>NAME READY STATUS
	 RESTARTS AGE
	pod-util 1/1 Running
	 0 50m</pre>
	2. 執行下列命令來安裝 awscurl。
	<pre>kubectl exec --stdin
	 --tty pod-util -- /
	bin/bash bash-5.1#
	pip3 install awscurl
	 Installing collected
	 packages: idna,
	 configparser,
	 configargparse,
	 charset-normalizer
	, certifi, requests,
	 awscurl Successfully
	 installed awscurl-0
	.32 certifi-2024.2.2
	 charset-normalizer
	-3.3.2 configarg
	parse-1.7 configpar
	ser-6.0.0 idna-3.6
	 requests-2.31.0
	 bash-5.1# awscurl
	 https://db-neptune
	-1.cluster-xxxxxxx
	xxxxx.us-west-2.ne
	ptune.amazonaws.co
	m:8182/status --
	region us-west-2 --
	service neptune-db
	 {"status":"healthy
	","startTime":"Thu
	 Feb 08 01:22:14 UTC
	 2024","dbEngineVer
	sion":"1.3.0.0.R1"
	,"role":"writer","</pre>	

任務	Description	所需的技能
	dfeQueryEngine":"viaQueryHint","gremlin":{"version":"tinkerpop-3.6.4"},"sparql":{"version":"sparql-1.1"},"openCypher":{"version":"Neptune-9.0.20190305-1.0"},"labMode":{"ObjectIndex":"disabled","ReadWriteConflictDetection":"enabled"},"features":{"SlowQueryLogs":"disabled","ResultCache":{"status":"disabled"},"IAMAuthentication":"enabled","Streams":"disabled","AuditLog":"disabled"},"settings":{"clusterQueryTimeoutInMs":"120000","SlowQueryLogsThreshold":"5000"}}	

疑難排解

問題	解決方案
無法存取 Neptune 資料庫。	檢閱連接至服務帳戶的 IAM 政策。請確定它允許您要執行之操作的必要動作（例如 `neptune:Connec,neptune:DescribeDBInstances` ）。

相關資源

- AWS 使用 Kubernetes 服務帳戶授予 Kubernetes 工作負載對 的存取權 (Amazon EKS 文件)
- 服務帳戶的 IAM 角色 (Amazon EKS 文件)
- 建立新的 Neptune 資料庫叢集 (Amazon Neptune 文件)

使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon ECS 上私下存取容器應用程式

Kirankumar Chandrashekar，Amazon Web Services

總結

此模式說明如何在 Network Load Balancer 後方的 Amazon Elastic Container Service (Amazon ECS) 上私有託管 Docker 容器應用程式，並使用 AWS PrivateLink 存取應用程式。然後，您可以使用私有網路安全地存取 Amazon Web Services (AWS) 雲端上的服務。Amazon Relational Database Service (Amazon RDS) 以高可用性 (HA) 託管在 Amazon ECS 上執行之應用程式的關聯式資料庫。如果應用程式需要持久性儲存，則會使用 Amazon Elastic File System (Amazon EFS)。

執行 Docker 應用程式的 Amazon ECS 服務與前端的 Network Load Balancer 可以與虛擬私有雲端 (VPC) 端點建立關聯，以便透過 AWS PrivateLink 存取。然後，您可以使用 VPC 端點與其他 VPCs 共用此 VPC 端點服務。

您也可以使用 AWS Fargate 而非 Amazon EC2 Auto Scaling 群組。如需詳細資訊，請參閱使用 AWS Fargate、AWS PrivateLink 和 Network Load Balancer 在 Amazon ECS 上私下存取容器應用程式。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在 Linux、macOS 或 Windows 上安裝和設定 AWS Command Line Interface (AWS CLI) 第 2 版 macOS
- 在 Linux、macOS 或 Windows 上安裝和設定 Docker
- 在 Docker 上執行的應用程式

Architecture

技術堆疊

- Amazon CloudWatch
- Amazon Elastic Compute Cloud (Amazon EC2)

- Amazon EC2 Auto Scaling
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon ECS
- Amazon RDS
- Amazon Simple Storage Service (Amazon S3)
- AWS Lambda
- AWS PrivateLink
- AWS Secrets Manager
- Application Load Balancer
- Network Load Balancer
- VPC

自動化和擴展

- 您可以使用 AWS CloudFormation 建立此模式，方法是使用基礎設施做為程式碼。

工具

- Amazon EC2 – Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。
- Amazon EC2 Auto Scaling – Amazon EC2 Auto Scaling 可協助您確保有正確數量的 Amazon EC2 執行個體可用於處理應用程式的負載。
- Amazon ECS – Amazon Elastic Container Service (Amazon ECS) 是一種高度可擴展、快速的容器管理服務，可讓您輕鬆執行、停止和管理叢集上的容器。
- Amazon ECR – Amazon Elastic Container Registry (Amazon ECR) 是安全、可擴展且可靠的受管 AWS 容器映像登錄服務。
- Amazon EFS – Amazon Elastic File System (Amazon EFS) 提供簡單、可擴展、全受管的彈性 NFS 檔案系統，可與 AWS 雲端服務和內部部署資源搭配使用。
- AWS Lambda – Lambda 是一種運算服務，可在不佈建或管理伺服器的情況下執行程式碼。
- Amazon RDS – Amazon Relational Database Service (Amazon RDS) 是一種 Web 服務，可讓您更輕鬆地在 AWS 雲端中設定、操作和擴展關聯式資料庫。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是網際網路的儲存體。此服務旨在降低開發人員進行網路規模運算的難度。

- AWS Secrets Manager – Secrets Manager 透過向 Secrets Manager 提供 API 呼叫以程式設計方式擷取秘密，協助您取代程式碼中的硬式編碼登入資料，包括密碼。
- Amazon VPC – Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。
- Elastic Load Balancing – Elastic Load Balancing 會將傳入的應用程式或網路流量分散到多個可用區域中的多個目標，例如 Amazon EC2 執行個體、容器和 IP 地址。
- Docker – Docker 可協助開發人員封裝、運送和執行任何應用程式，做為輕量、可攜式且自給自足的容器。

史詩

建立網路元件

任務	Description	所需的技能
建立 VPC。	1. 登入 AWS 管理主控台並開啟 Amazon VPC 主控台。選擇建立 VPC，然後選擇 VPC 等。 2. 輸入 VPC 的名稱，然後選擇適當的 CIDR 區塊範圍。 3. 指定兩個可用區域、兩個公有子網路、四個私有子網路。兩個私有子網路用於 Amazon ECS 任務，而兩個私有子網路用於 Amazon RDS 資料庫。 4. 為每個可用區域指定一個 NAT 閘道。 5. 選擇建立 VPC。	雲端管理員

建立負載平衡器

任務	Description	所需的技能
建立 Network Load Balancer。	1. 開啟 Amazon EC2 主控台，然後選擇包含 VPC 的 AWS 區域。 2. 在負載平衡下，選擇負載平衡器，然後選擇建立負載平衡器。 3. 選擇 Network Load Balancer，然後選擇建立。 4. 在設定負載平衡器頁面上，設定 Network Load Balancer 和接聽程式。重要：請確定您將 Network Load Balancer 的方案選擇為內部。 5. 選擇適用的安全設定，設定安全群組和目標群組。在設定路由區段中選擇執行個體或 IP 做為目標類型。請確定您未註冊目標。 6. 設定所有設定後，選擇下一步：檢閱，然後選擇建立。	雲端管理員
建立 Application Load Balancer。	1. 在 Amazon EC2 主控台上，選擇包含 VPC 的相同區域。 2. 在負載平衡下，選擇負載平衡器，然後選擇建立負載平衡器。 3. 選擇 Application Load Balancer，然後選擇建立。	雲端管理員

任務	Description	所需的技能
	4. **Important** 設定 Application Load Balancer 及其接聽程式。請務必將 Application Load Balancer 的方案選擇為內部。 5. 選擇適用的安全設定，設定安全群組和目標群組。在設定路由區段中選擇執行個體或 IP 做為目標類型。請確定您未註冊目標。 6. 設定所有設定後，選擇下一步：檢閱，然後選擇建立。	

建立 Amazon EFS 檔案系統

任務	Description	所需的技能
建立 Amazon EFS 檔案系統。	1. 開啟 Amazon EFS 主控台，然後選擇建立檔案系統。 2. 在建立檔案系統對話方塊中，輸入檔案系統的名稱，然後選擇您的 VPC。 3. 選擇建立以建立檔案系統。 4. 設定您的 Amazon EFS 檔案系統。	雲端管理員
為子網路掛載目標。	1. 返回 Amazon EFS 主控台，然後選擇檔案系統。檔案系統頁面會顯示您帳戶中的 Amazon EFS 檔案系統。	雲端管理員

任務	Description	所需的技能
	2. 選擇您建立的檔案系統，然後選擇管理以顯示可用區域。若要新增掛載目標，請選擇新增掛載目標，然後新增您建立的四個私有子網路。	
確認子網路已掛載為目標。	1. 在 Amazon EFS 主控台上，選擇檔案系統。 2. 選擇網路以顯示現有掛載目標的清單。請確定這些包含您建立的四個子網路。	雲端管理員

建立 S3 儲存貯體

任務	Description	所需的技能
建立 S3 儲存貯體。	開啟 Amazon S3 主控台，並視需要建立 S3 儲存貯體來存放應用程式的靜態資產。	雲端管理員

建立 Secrets Manager 秘密

任務	Description	所需的技能
建立 AWS KMS 金鑰來加密 Secrets Manager 秘密。	開啟 AWS Key Management Service (AWS KMS) 主控台並建立 KMS 金鑰。	雲端管理員
建立 Secrets Manager 秘密以存放 Amazon RDS 密碼。	1. 開啟 AWS Secrets Manager 主控台，然後選擇儲存新的秘密來建立新的秘密。 2. 選擇您建立的 KMS 金鑰，然後存放您的新秘密。	雲端管理員

建立 Amazon RDS 執行個體

任務	Description	所需的技能
建立資料庫子網路群組。	1. 開啟 Amazon RDS 主控台，然後選擇子網路群組。 2. 選擇建立資料庫子網路群組，然後輸入資料庫子網路群組的名稱和描述。 3. 選擇您先前建立的 VPC，然後選擇可用區域和子網路。然後選擇 Create (建立)。	雲端管理員
建立 Amazon RDS 執行個體。	在私有子網路內建立和設定 Amazon RDS 執行個體。確定已針對 HA 開啟異地同步備份。	雲端管理員
將資料載入 Amazon RDS 執行個體。	將應用程式所需的關聯式資料載入 Amazon RDS 執行個體。此程序會根據應用程式的需求，以及資料庫結構描述的定義和設計方式而有所不同。	雲端管理員，DBA

建立 Amazon ECS 元件

任務	Description	所需的技能
建立 ECS 叢集。	1. 開啟 Amazon ECS 主控台，然後選擇叢集。 2. 選擇建立叢集，並根據所需的規格設定 ECS 叢集。	雲端管理員
建立 Docker 影像。	遵循相關資源區段中的指示建立 Docker 映像。	雲端管理員

任務	Description	所需的技能
建立 Amazon ECR 儲存庫。	1. 在 Amazon ECR 主控台上，選擇儲存庫。 2. 選擇建立儲存庫，然後輸入儲存庫的唯一名稱。 3. 根據您的規格設定儲存庫，包括必要時的 AWS KMS 加密。	雲端管理員、DevOps 工程師
驗證 Amazon ECR 儲存庫的 Docker 用戶端。	若要驗證 Amazon ECR 儲存庫的 Docker 用戶端，請在 AWS CLI 中執行「`aws ecr get-login-password` 命令。	雲端管理員
將 Docker 映像推送至 Amazon ECR 儲存庫。	1. 識別您要推送的 Docker 映像，並在 AWS CLI 中執行`docker images`命令。 2. 使用 Amazon ECR 登錄檔、儲存庫和選用的映像標籤名稱組合來標記您的映像。 3. 執行`docker push`命令來推送 Docker 映像。 4. 針對所有必要的影像重複這些步驟。	雲端管理員

任務	Description	所需的技能
建立 Amazon ECS 任務定義。	在 Amazon ECS 中執行 Docker 容器所需的任務定義。 1. 返回 Amazon ECS 主控台，選擇任務定義，然後選擇建立新任務定義。 2. 在選取相容性頁面上，選取任務應使用的啟動類型，然後選擇下一步。 **Important** 如需設定任務定義的協助，請參閱相關資源區段中的「建立任務定義」。請務必提供您推送到 Amazon ECR 的 Docker 映像。	雲端管理員
建立 Amazon ECS 服務。	使用您先前建立的 ECS 叢集來建立 Amazon ECS 服務。請務必選擇 Amazon EC2 做為啟動類型，然後選擇在上一個步驟中建立的任務定義，以及 Application Load Balancer 的目標群組。	雲端管理員

建立 Amazon EC2 Auto Scaling 群組

任務	Description	所需的技能
建立啟動組態。	開啟 Amazon EC2 主控台，並建立啟動組態。確定使用者資料具有允許 EC2 執行個體加入所需 ECS 叢集的程式碼。如需所需程式碼的範例，請參閱相關資源一節。	雲端管理員
建立 Amazon EC2 Auto Scaling 群組。	返回 Amazon EC2 主控台，然後在 Auto Scaling 下，選擇 Auto Scaling 群組。設定 Amazon EC2 Auto Scaling 群組。請務必選擇先前建立的私有子網路和啟動組態。	雲端管理員

設定 AWS PrivateLink

任務	Description	所需的技能
設定 AWS PrivateLink 端點。	1. 在 Amazon VPC 主控台上，建立 AWS PrivateLink 端點。 2. 將此端點與 Network Load Balancer 建立關聯，讓託管在 Amazon ECS 上的應用程式可私下提供給客戶。 如需詳細資訊，請參閱相關資源一節。	雲端管理員

建立 VPC 端點

任務	Description	所需的技能
建立 VPC 端點。	為您先前建立的 AWS PrivateLink 端點建立 VPC 端點。VPC 端點完整網域名稱 (FQDN) 會指向 AWS PrivateLink 端點 FQDN。這會為 DNS 端點可存取的 VPC 端點服務建立彈性網路界面。	雲端管理員

建立 Lambda 函式

任務	Description	所需的技能
建立 Lambda 函數。	在 AWS Lambda 主控台上，建立 Lambda 函數，將 Application Load Balancer IP 地址更新為 Network Load Balancer 的目標。如需詳細資訊，請參閱使用 AWS Lambda 為 Application Load Balancer 啟用靜態 IP 地址部落格文章。	應用程式開發人員

相關資源

建立負載平衡器：

- 使用適用於 Amazon ECS 的 Network Load Balancer
- 建立 Network Load Balancer
- 使用適用於 Amazon ECS 的 Application Load Balancer
- 建立 Application Load Balancer

建立 Amazon EFS 檔案系統：

- 建立 Amazon EFS 檔案系統
- 在 Amazon EFS 中建立掛載目標

建立 S3 儲存貯體：

- 建立 S3 儲存貯體

建立 Secrets Manager 秘密：

- 在 AWS KMS 中建立金鑰
- 在 AWS Secrets Manager 中建立秘密

建立 Amazon RDS 執行個體：

- 建立 Amazon RDS 資料庫執行個體

建立 Amazon ECS 元件：

- 建立 Amazon ECS 叢集
- 建立 Docker 映像
- 建立 Amazon ECR 儲存庫
- 使用 Amazon ECR 儲存庫驗證 Docker
- 將映像推送至 Amazon ECR 儲存庫
- 建立 Amazon ECS 任務定義
- 建立 Amazon ECS 服務

建立 Amazon EC2 Auto Scaling 群組：

- 建立啟動組態
- 使用啟動組態建立 Auto Scaling 群組
- 使用 Amazon EC2 使用者資料引導容器執行個體

設定 AWS PrivateLink：

- VPC 端點服務 (AWS PrivateLink)

建立 VPC 端點：

- 介面 VPC 端點 (AWS PrivateLink)

建立 Lambda 函數：

- 建立 Lambda 函數

其他資源：

- 使用 Application Load Balancer 的靜態 IP 地址
- 透過 AWS PrivateLink 安全地存取服務

使用 AWS Fargate、AWS PrivateLink 和 Network Load Balancer，在 Amazon ECS 上私下存取容器應用程式

Kirankumar Chandrashekar，Amazon Web Services

總結

此模式說明如何使用 Amazon Elastic Container Service (Amazon ECS) 搭配 AWS Fargate 啟動類型，在 Network Load Balancer 後方私下託管 Amazon Web Services (AWS) 雲端上的 Docker 容器應用程式，並使用 AWS PrivateLink 存取應用程式。Amazon Relational Database Service (Amazon RDS) 以高可用性 (HA) 託管在 Amazon ECS 上執行之應用程式的關聯式資料庫。如果應用程式需要持久性儲存，您可以使用 Amazon Elastic File System (Amazon EFS)。

此模式針對執行 Docker 應用程式的 Amazon ECS 服務使用 Fargate 啟動類型，並在前端使用 Network Load Balancer。然後，它可以與虛擬私有雲端 (VPC) 端點建立關聯，以便透過 AWS PrivateLink 進行存取。然後，您可以使用 VPC 端點與其他 VPCs 共用此 VPC 端點服務。

您可以使用 Fargate 搭配 Amazon ECS 執行容器，而不必管理 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的伺服器或叢集。您也可以使用 Amazon EC2 Auto Scaling 群組，而非 Fargate。如需詳細資訊，請參閱使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon ECS 上私下存取容器應用程式。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在 Linux、macOS 或 Windows 上安裝和設定 AWS Command Line Interface (AWS CLI) 第 2 版 macOS
- 在 Linux、macOS 或 Windows 上安裝和設定 Docker
- 在 Docker 上執行的應用程式

Architecture

技術堆疊

- Amazon CloudWatch

- Amazon Elastic Container Registry (Amazon ECR)
- Amazon ECS
- Amazon EFS
- Amazon RDS
- Amazon Simple Storage Service (Amazon S3)
- AWS Fargate
- AWS PrivateLink
- AWS Secrets Manager
- Application Load Balancer
- Network Load Balancer
- VPC

自動化和擴展

- 您可以使用 AWS CloudFormation 建立此模式，方法是使用基礎設施做為程式碼。

工具

AWS 服務

- Amazon Elastic Container Registry (Amazon ECR) 是安全、可擴展且可靠的受管 AWS 容器映像登錄服務。
- Amazon Elastic Container Service (Amazon ECS) 是一種高度可擴展、快速的容器管理服務，可讓您輕鬆地執行、停止和管理叢集上的容器。
- Amazon Elastic File System (Amazon EFS) 提供簡單、可擴展、全受管的彈性 NFS 檔案系統，可與 AWS 雲端服務和內部部署資源搭配使用。
- AWS Fargate 是一項技術，您可以與 Amazon ECS 搭配使用來執行容器，而無需管理 Amazon EC2 執行個體的伺服器或叢集。
- Amazon Relational Database Service (Amazon RDS) 是一種 Web 服務，可讓您更輕鬆地在 中設定、操作和擴展關聯式資料庫 AWS 雲端。
- Amazon Simple Storage Service (Amazon S3) 是網際網路的儲存體。此服務旨在降低開發人員進行網路規模運算的難度。

- AWS Secrets Manager 可協助您將程式碼中的硬式編碼憑證 (包括密碼) 取代為 Secrets Manager 的 API 呼叫，以便透過程式設計方法來擷取機密。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。
- Elastic Load Balancing (ELB) 會將傳入的應用程式或網路流量分配到多個可用區域中的多個目標，例如 EC2 執行個體、容器和 IP 地址。

其他工具

- Docker 可協助開發人員輕鬆封裝、運送和執行任何應用程式，做為輕量、可攜式且自給自足的容器。

史詩

建立網路元件

任務	Description	所需的技能
建立 VPC。	1. 登入 AWS 管理主控台，然後開啟 Amazon VPC 主控台。選擇建立 VPC，然後選擇 VPC 等。 2. 輸入 VPC 的名稱，然後選擇適當的 CIDR 區塊範圍。 3. 指定兩個可用區域、兩個公有子網路、四個私有子網路。兩個私有子網路用於 Amazon ECS 任務，而兩個私有子網路用於 Amazon RDS 資料庫。 4. 為每個可用區域指定一個 NAT 閘道。 5. 選擇建立 VPC。	雲端管理員

建立負載平衡器

任務	Description	所需的技能
建立 Network Load Balancer。	1. 開啟 Amazon EC2 主控台，然後選擇包含 VPC 的 AWS 區域。 2. 在負載平衡下，選擇負載平衡器，然後選擇建立負載平衡器。 3. 選擇 Network Load Balancer，然後選擇建立。 4. 在設定負載平衡器頁面上，設定 Network Load Balancer 和接聽程式。重要：請務必將 Network Load Balancer 的方案選擇為內部。 5. 選擇適用的安全設定，設定安全群組和目標群組。在設定路由區段中選擇 IP 做為目標類型。請確定您未註冊目標。 6. 設定所有設定後，選擇下一步：檢閱，然後選擇建立。 如需此案例和其他案例的協助，請參閱相關資源一節。	雲端管理員
建立 Application Load Balancer。	1. 在 Amazon EC2 主控台上，選擇包含 VPC 的相同區域。 2. 在負載平衡下，選擇負載平衡器，然後選擇建立負載平衡器。	雲端管理員

任務	Description	所需的技能
	3. 選擇 Application Load Balancer，然後選擇建立。 4. **Important** 設定 Application Load Balancer 及其接聽程式。請務必將 Application Load Balancer 的方案選擇為內部。 5. 選擇適用的安全設定，設定安全群組和目標群組。在設定路由區段中選擇 IP 做為目標類型。請確定您未註冊目標。 6. 設定所有設定後，選擇下一步：檢閱，然後選擇建立。	

建立 Amazon EFS 檔案系統

任務	Description	所需的技能
建立 Amazon EFS 檔案系統。	1. 開啟 Amazon EFS 主控台，然後選擇建立檔案系統。 2. 在建立檔案系統對話方塊中，輸入檔案系統的名稱，然後選擇您的 VPC。 3. 選擇建立以建立檔案系統。 4. 設定您的 Amazon EFS 檔案系統。	雲端管理員
為子網路掛載目標。	1. 返回 Amazon EFS 主控台，然後選擇檔案系統。檔案系	雲端管理員

任務	Description	所需的技能
	統頁面會顯示您帳戶中的 Amazon EFS 檔案系統。 2. 選擇您建立的檔案系統，然後選擇管理以顯示可用區域。 3. 若要新增掛載目標，請選擇新增掛載目標，然後新增您建立的四個私有子網路。	
確認子網路已掛載為目標。	1. 在 Amazon EFS 主控台上，選擇檔案系統。 2. 選擇網路以顯示現有掛載目標的清單。請確定這些包含您建立的四個子網路。	雲端管理員

建立 S3 儲存貯體

任務	Description	所需的技能
建立 S3 儲存貯體。	開啟 Amazon S3 主控台，並視需要建立 S3 儲存貯體來存放應用程式的靜態資產。	雲端管理員

建立 Secrets Manager 秘密

任務	Description	所需的技能
建立 AWS KMS 金鑰來加密 Secrets Manager 秘密。	開啟 AWS Key Management Service (AWS KMS) 主控台並建立 KMS 金鑰。	雲端管理員
建立 Secrets Manager 秘密以存放 Amazon RDS 密碼。	1. 開啟 AWS Secrets Manager 主控台，然後選擇儲存新的秘密來建立新的秘密。	雲端管理員

任務	Description	所需的技能
	2. 選擇您建立的 KMS 金鑰，並存放您的新秘密。	

建立 Amazon RDS 執行個體

任務	Description	所需的技能
建立資料庫子網路群組。	1. 開啟 Amazon RDS 主控台，然後選擇子網路群組。 2. 選擇建立資料庫子網路群組，然後輸入資料庫子網路群組的名稱和描述。 3. 選擇您先前建立的 VPC，然後選擇可用區域和子網路。然後選擇 Create (建立)。	雲端管理員
建立 Amazon RDS 執行個體。	在私有子網路內建立和設定 Amazon RDS 執行個體。確保已開啟異地同步備份以實現高可用性 (HA)。	雲端管理員
將資料載入 Amazon RDS 執行個體。	將應用程式所需的關聯式資料載入 Amazon RDS 執行個體。此程序會根據應用程式的需求，以及資料庫結構描述的定義和設計方式而有所不同。	DBA

建立 Amazon ECS 元件

任務	Description	所需的技能
建立 ECS 叢集。	1. 開啟 Amazon ECS 主控台，然後選擇叢集。	雲端管理員

任務	Description	所需的技能
	2. 選擇建立叢集，並根據所需的規格設定 ECS 叢集。	
建立 Docker 映像。	遵循 AWS 文件中的指示建立 Docker 映像。	雲端管理員
建立 Amazon ECR 儲存庫。	1. 開啟 Amazon ECR 主控台，然後選擇儲存庫。 2. 選擇建立儲存庫，然後輸入儲存庫的唯一名稱。 3. 根據您的規格設定儲存庫，包括必要時的 AWS KMS 加密。	雲端管理員、DevOps 工程師
將 Docker 映像推送至 Amazon ECR 儲存庫。	1. 識別您要推送的 Docker 映像，並在 AWS CLI 中執行 `docker images`命令。 2. 使用 Amazon ECR 登錄檔、儲存庫和選用的映像標籤名稱組合來標記您的映像。 3. 執行 `docker push`命令來推送 Docker 映像。 4. 針對所有必要的影像重複這些步驟。	雲端管理員

任務	Description	所需的技能
建立 Amazon ECS 任務定義。	在 Amazon ECS 中執行 Docker 容器所需的任務定義。 1. 返回 Amazon ECS 主控台，選擇任務定義，然後選擇建立新任務定義。 2. 在選取相容性頁面上，選取任務應使用的啟動類型，然後選擇下一步。 **Important** 如需設定任務定義的協助，請參閱相關資源區段中的「建立任務定義」。請務必提供您推送到 Amazon ECR 的 Docker 映像。	雲端管理員
建立 ECS 服務，然後選擇 Fargate 作為啟動類型。	1. 使用您先前建立的 ECS 叢集來建立 Amazon ECS 服務。請務必選擇 Fargate 作為啟動類型。 2. 選擇在上一個步驟中建立的任務定義，然後選擇 Application Load Balancer 的目標群組。	雲端管理員

設定 AWS PrivateLink

任務	Description	所需的技能
設定 AWS PrivateLink 端點。	1. 開啟 Amazon VPC 主控台，並建立 AWS PrivateLink 端點。 2. 將此端點與 Network Load Balancer 建立關聯，讓託管在 Amazon ECS 上的應用程式可私下提供給客戶。	雲端管理員

建立 VPC 端點

任務	Description	所需的技能
建立 VPC 端點。	為您先前建立的 AWS PrivateLink 端點建立 VPC 端點。VPC 端點完整網域名稱 (FQDN) 會指向 AWS PrivateLink 端點 FQDN。這會為網域名稱服務端點可存取的 VPC 端點服務建立彈性網路界面。	雲端管理員

設定目標

任務	Description	所需的技能
新增 Application Load Balancer 做為目標。	若要將 Application Load Balancer 新增為 Network Load Balancer 的目標，請遵循 AWS 文件中的指示。	應用程式開發人員

相關資源

建立負載平衡器：

- 使用適用於 Amazon ECS 的 Network Load Balancer
- 建立 Network Load Balancer
- 使用適用於 Amazon ECS 的 Application Load Balancer
- 建立 Application Load Balancer

建立 Amazon EFS 檔案系統：

- 建立 Amazon EFS 檔案系統
- 在 Amazon EFS 中建立掛載目標

建立 Secrets Manager 秘密：

- 在 AWS KMS 中建立金鑰
- 在 AWS Secrets Manager 中建立秘密

建立 Amazon RDS 執行個體：

- 建立 Amazon RDS 資料庫執行個體

建立 Amazon ECS 元件

- 建立 Amazon ECR 儲存庫
- 使用 Amazon ECR 儲存庫驗證 Docker
- 將映像推送至 Amazon ECR 儲存庫
- 建立 Amazon ECS 任務定義
- 建立 Amazon ECS 服務

其他資源：

- 透過 AWS PrivateLink 安全地存取服務

使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon EKS 上私下存取容器應用程式

Kirankumar Chandrashekar，Amazon Web Services

總結

此模式說明如何在 Network Load Balancer 後方的 Amazon Elastic Kubernetes Service (Amazon EKS) 上私有託管 Docker 容器應用程式，並使用 AWS PrivateLink 存取應用程式。然後，您可以使用私有網路安全地存取 Amazon Web Services (AWS) 雲端上的服務。

執行 Docker 應用程式的 Amazon EKS 叢集與前端的 Network Load Balancer 可以與虛擬私有雲端 (VPC) 端點建立關聯，以便透過 AWS PrivateLink 存取。然後，您可以使用 VPC 端點與其他 VPCs 共用此 VPC 端點服務。

此模式描述的設定是在 VPCs 和 AWS 帳戶之間共用應用程式存取權的安全方式。它不需要特殊的連線或路由組態，因為消費者和提供者帳戶之間的連線位於全球 AWS 骨幹，而且不會周遊公有網際網路。

先決條件和限制

先決條件

- 在 Linux、macOS 或 Windows 上安裝和設定 Docker。
- 在 Docker 上執行的應用程式。
- 作用中的 AWS 帳戶
- 在 Linux、macOS 或 Windows 上安裝和設定 AWS Command Line Interface (AWS CLI) 第 2 版。
  macOS
- 具有標記私有子網路並設定為託管應用程式的現有 Amazon EKS 叢集。如需詳細資訊，請參閱 Amazon EKS 文件中的子網路標記。
- Kubectl，已安裝並設定為存取 Amazon EKS 叢集上的資源。如需詳細資訊，請參閱 Amazon EKS 文件中的安裝 kubectl。

Architecture

技術堆疊

- Amazon EKS

- AWS PrivateLink
- Network Load Balancer

自動化和擴展

- 您可以在 Git 型儲存庫上追蹤和管理 Kubernetes 資訊清單，並在 AWS CodePipeline 中使用持續整合和持續交付 (CI/CD) 部署。
- 您可以使用 AWS CloudFormation 建立此模式，方法是使用基礎設施做為程式碼 (IaC)。

工具

- AWS CLI – AWS 命令列界面 (AWS CLI) 是一種開放原始碼工具，可讓您使用命令列 Shell 中的命令與 AWS 服務互動。
- Elastic Load Balancing – Elastic Load Balancing 會將傳入的應用程式或網路流量分散到一或多個可用區域中的多個目標，例如 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、容器和 IP 地址。

- Amazon EKS – Amazon Elastic Kubernetes Service (Amazon EKS) 是一項受管服務，可讓您在 AWS 上執行 Kubernetes，而無需安裝、操作和維護您自己的 Kubernetes 控制平面或節點。
- Amazon VPC – Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。
- Kubectl – Kubectl 是針對 Kubernetes 叢集執行命令的命令列公用程式。

史詩

部署 Kubernetes 部署和服務資訊清單檔案

任務	Description	所需的技能

建立 Kubernetes 部署資訊清單檔案。	根據您的需求修改下列範例檔案，以建立部署資訊清單檔案。
	
	`apiVersion: apps/v1`
	`kind: Deployment`
		DevOps 工程師

任務	Description	所需的技能

	<pre>metadata:
	  name: sample-app
	spec:
	  replicas: 3
	  selector:
	    matchLabels:
	      app: nginx
	  template:
	    metadata:
	      labels:
	        app: nginx
	    spec:
	      containers:
	        - name: nginx
	          image:
	 public.ecr.aws/z9d
	2n7e1/nginx:1.19.5
	          ports:
	            - name: http
	              container
	Port: 80</pre>
	Note
	這是使用 NGINX Docker 映像部署的 NGINX 範例組態檔案。如需詳細資訊，請參閱 Docker 文件中的如何使用官方 NGINX Docker 映像。	

任務	Description	所需的技能
部署 Kubernetes 部署資訊清單檔案。	執行下列命令，將部署資訊清單檔案套用至您的 Amazon EKS 叢集： `kubectl apply -f <your_deployment_file_name>`	DevOps 工程師

任務	Description	所需的技能

建立 Kubernetes 服務資訊清單檔案。	根據您的需求修改下列範例檔案，以建立服務資訊清單檔案。
	
	`apiVersion: v1`
	`kind: Service`
	`metadata:`
	`  name: sample-service`
	`  annotations:`
	`  service.beta.kubernetes.io/aws-load-balancer-type: nlb`
	`  service.beta.kubernetes.io/aws-load-balancer-internal: "true"`
	`spec:`
	`  ports:`
	`  - port: 80`
	`  targetPort: 80`
	`  protocol: TCP`
	`  type: LoadBalancer`
	`  selector:`
	`  app: nginx`
	
	Important
	請確定您已包含下列項目`annotations`來定義內部 Network Load Balancer：
	
	`service.beta.kubernetes.io/aws-load-balancer-type: nlb`
	`service.beta.kubernetes.io/aws-l`
		DevOps 工程師

任務	Description	所需的技能
	`oad-balancer-inter nal: "true"`	
部署 Kubernetes 服務資訊清單檔案。	執行下列命令，將服務資訊清單檔案套用至您的 Amazon EKS 叢集： `kubectl apply -f <your_service_file _name>`	DevOps 工程師

建立端點

任務	Description	所需的技能
記錄 Network Load Balancer 的名稱。	執行下列命令來擷取 Network Load Balancer 的名稱： `kubectl get svc sample-service -o wide` 記錄 Network Load Balancer 的名稱，這是建立 AWS PrivateLink 端點的必要項目。	DevOps 工程師
建立 AWS PrivateLink 端點。	登入 AWS 管理主控台，開啟 Amazon VPC 主控台，然後建立 AWS PrivateLink 端點。將此端點與 Network Load Balancer 建立關聯，這可讓客戶私下使用應用程式。如需詳細資訊，請參閱 Amazon VPC 文件中的 VPC 端點服務 (AWS PrivateLink)。	雲端管理員

任務	Description	所需的技能
	Important 如果消費者帳戶需要存取應用程式，則必須將消費者帳戶的 AWS 帳戶 ID 新增至 AWS PrivateLink 端點組態的允許主體清單。如需詳細資訊，請參閱 Amazon VPC 文件中的新增和移除端點服務的許可。	
建立 VPC 端點。	在 Amazon VPC 主控台上，選擇端點服務，然後選擇建立端點服務。為 AWS PrivateLink 端點建立 VPC 端點。 VPC 端點的完整網域名稱 (FQDN) 會指向 AWS PrivateLink 端點的 FQDN。這會為 DNS 端點可存取的 VPC 端點服務建立彈性網路界面。	雲端管理員

相關資源

- 使用官方 NGINX Docker 映像
- Amazon EKS 上的網路負載平衡
- 建立 VPC 端點服務 (AWS PrivateLink)
- 新增和移除端點服務的許可

使用 AWS Batch 自動化 Amazon RDS for PostgreSQL 資料庫執行個體的備份

Kirankumar Chandrashekar，Amazon Web Services

總結

備份 PostgreSQL 資料庫是一項重要的任務，通常可以使用 pg_dump 公用程式完成，該公用程式預設使用 COPY 命令來建立 PostgreSQL 資料庫的結構描述和資料傾印。不過，如果您需要定期備份多個 PostgreSQL 資料庫，此程序可能會變得重複。如果您的 PostgreSQL 資料庫託管在雲端，您也可以利用 Amazon Relational Database Service (Amazon RDS) for PostgreSQL 提供的自動備份功能。此模式說明如何使用 pg_dump 公用程式自動化 Amazon RDS for PostgreSQL 資料庫執行個體的定期備份。

注意：指示假設您使用的是 Amazon RDS。不過，您也可以針對在 Amazon RDS 外部託管的 PostgreSQL 資料庫使用此方法。若要進行備份，AWS Lambda 函數必須能夠存取您的資料庫。

以時間為基礎的 Amazon CloudWatch Events 事件會啟動 Lambda 函數，以搜尋套用至 Amazon RDS 上 PostgreSQL 資料庫執行個體中繼資料的特定備份標籤。 PostgreSQL 如果 PostgreSQL 資料庫執行個體具有 bkp：AutomatedDBDump = Active 標籤和其他必要的備份標籤，Lambda 函數會將每個資料庫備份的個別任務提交至 AWS Batch。

AWS Batch 會處理這些任務，並將備份資料上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體。此模式使用 Dockerfile 和 entrypoint.sh 檔案來建置 Docker 容器映像，用於在 AWS Batch 任務中進行備份。備份程序完成後，AWS Batch 會將備份詳細資訊記錄到 Amazon DynamoDB 上的清查資料表。此外，CloudWatch Events 事件會在 AWS Batch 中的任務失敗時啟動 Amazon Simple Notification Service (Amazon SNS) 通知。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 現有的受管或未受管運算環境。如需詳細資訊，請參閱 AWS Batch 文件中的受管和未受管運算環境。
- 安裝和設定 AWS Command Line Interface (CLI) 第 2 版 Docker 映像。
- 現有的 Amazon RDS for PostgreSQL 資料庫執行個體。
- 現有的 S3 儲存貯體。
- 在 Linux、macOS 或 Windows 上安裝和設定 Docker。

- 熟悉 Lambda 中的編碼。

Architecture

技術堆疊

- Amazon CloudWatch Events
- Amazon DynamoDB
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon RDS
- Amazon SNS
- Amazon S3
- AWS Batch
- AWS Key Management Service (AWS KMS)
- AWS Lambda
- AWS Secrets Manager
- Docker

工具

- Amazon CloudWatch Events – CloudWatch Events 提供近乎即時的系統事件串流，說明 AWS 資源的變更。
- Amazon DynamoDB – DynamoDB 是全受管的 NoSQL 資料庫服務，可提供快速且可預測的效能和無縫的可擴展性。
- Amazon ECR – Amazon Elastic Container Registry (Amazon ECR) 是安全、可擴展且可靠的受管 AWS 容器映像登錄服務。
- Amazon RDS – Amazon Relational Database Service (Amazon RDS) 是一種 Web 服務，可讓您更輕鬆地在 AWS 雲端中設定、操作和擴展關聯式資料庫。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一種受管服務，可將訊息從發佈者交付給訂閱者。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是網際網路的儲存體。
- AWS Batch – AWS Batch 可協助您在 AWS 雲端上執行批次運算工作負載。

- AWS KMS – AWS Key Management Service (AWS KMS) 是一種受管服務，可讓您輕鬆地建立和控制用於加密資料的加密金鑰。
- AWS Lambda – Lambda 是一種運算服務，可協助您執行程式碼，而無需佈建或管理伺服器。
- AWS Secrets Manager – Secrets Manager 可協助您以 API 呼叫 Secrets Manager，以程式設計方式擷取秘密，取代程式碼中的硬式編碼登入資料，包括密碼。
- Docker – Docker 可協助開發人員輕鬆封裝、運送和執行任何應用程式，做為輕量、可攜式且自給自足的容器。

Amazon RDS 上的 PostgreSQL 資料庫執行個體必須將標籤套用至其中繼資料。Lambda 函數會搜尋標籤以識別應備份的資料庫執行個體，並通常會使用下列標籤。

Tag	Description
bkp：AutomatedDBDump = 作用中	將 Amazon RDS 資料庫執行個體識別為備份的候選項目。
bkp：AutomatedBackupSecret = <secret_name>	識別包含 Amazon RDS 登入憑證的 Secrets Manager 秘密。
bkp：AutomatedDBDumpS3Bucket = <s3_bucket_name>	識別要傳送備份的 S3 儲存貯體。
bkp：AutomatedDBDumpFrequency bkp：AutomatedDBDumpTime	識別應備份資料庫的頻率和時間。
bkp：pgdumpcommand = <pgdump_command>	識別需要備份的資料庫。

史詩

在 DynamoDB 中建立清查資料表

任務	Description	所需的技能
在 DynamoDB 中建立資料表。	登入 AWS 管理主控台，開啟 Amazon DynamoDB 主控台，	雲端管理員、資料庫管理員

任務	Description	所需的技能
	然後建立資料表。如需此案例和其他案例的協助，請參閱相關資源一節。	
確認資料表已建立。	執行 `aws dynamodb describe-table --table-name <table-name> | grep TableStatus` 命令。如果資料表存在，命令會傳回`"TableStatus": "ACTIVE",` 結果。	雲端管理員、資料庫管理員

在 AWS Batch 中為失敗的任務事件建立 SNS 主題

任務	Description	所需的技能
建立 SNS 主題。	開啟 Amazon SNS 主控台，選擇主題，然後使用名稱 建立 SNS 主題`JobFailedAlert` 。訂閱作用中的電子郵件地址至主題，並檢查您的電子郵件收件匣，以確認來自 AWS Notifications 的 SNS 訂閱電子郵件。	雲端管理員
為 AWS Batch 建立失敗的任務事件規則。	開啟 Amazon CloudWatch 主控台，選擇事件，然後選擇建立規則。選擇顯示進階選項，然後選擇編輯。對於建立模式以選取事件供目標處理，請將現有文字取代為其他資訊區段中的「任務事件失敗」程式碼。此程式碼定義了 CloudWatch Events 規則，會	雲端管理員

任務	Description	所需的技能
	在 AWS Batch 有`Failed`事件時啟動。	
新增事件規則目標。	在目標中，選擇新增目標，然後選擇 `JobFailedAlert` SNS 主題。設定其餘詳細資訊並建立 Cloudwatch Events 規則。	雲端管理員

建置 Docker 映像並將其推送至 Amazon ECR 儲存庫

任務	Description	所需的技能
建立 Amazon ECR 儲存庫。	開啟 Amazon ECR 主控台，然後選擇您要在其中建立儲存庫的 AWS 區域。選擇儲存庫，然後選擇建立儲存庫。根據您的需求設定儲存庫。	雲端管理員
撰寫 Dockerfile。	登入 Docker，並使用其他資訊區段中的「範例 Dockerfile」和「範例 entrypoint.sh 檔案」來建置 Dockerfile。	DevOps 工程師
建立 Docker 映像並將其推送至 Amazon ECR 儲存庫。	將 Dockerfile 建置至 Docker 映像檔，並將其推送至 Amazon ECR 儲存庫。如需此案例的說明，請參閱相關資源一節。	DevOps 工程師

建立 AWS Batch 元件

任務	Description	所需的技能
建立 AWS Batch 任務定義。	開啟 AWS Batch 主控台，並建立任務定義，其中包含 Amazon ECR 儲存庫的統一資源識別符 (URI) 做為 屬性 `Image`。	雲端管理員
設定 AWS Batch 任務佇列。	在 AWS Batch 主控台上，選擇任務佇列，然後選擇建立佇列。建立任務佇列來存放任務，直到 AWS Batch 在運算環境中的資源上執行任務為止。重要：請務必為 AWS Batch 編寫邏輯，將備份詳細資訊記錄到 DynamoDB 清查資料表。	雲端管理員

建立和排程 Lambda 函數

任務	Description	所需的技能
建立 Lambda 函數以搜尋標籤。	建立 Lambda 函數，在 PostgreSQL 資料庫執行個體上搜尋標籤並識別備份候選項目。確保您的 Lambda 函數可以識別`bkp:AutomatedDBDump = Active` 標籤和所有其他必要的標籤。重要：Lambda 函數也必須能夠將任務新增至 AWS Batch 任務佇列。	DevOps 工程師
建立以時間為基礎的 CloudWatch Events 事件。	開啟 Amazon CloudWatch 主控台並建立 CloudWatch Events 事件，該事件使用	雲端管理員

任務	Description	所需的技能
	Cron 表達式定期執行 Lambda 函數。重要：所有排程事件都使用 UTC 時區。	

測試備份自動化

任務	Description	所需的技能
建立 Amazon KMS 金鑰。	開啟 Amazon KMS 主控台並建立 KMS 金鑰，可用來加密存放在 AWS Secrets Manager 中的 Amazon RDS 登入資料。	雲端管理員
建立 AWS Secrets Manager 秘密。	開啟 AWS Secrets Manager 主控台，並將 Amazon RDS for PostgreSQL 資料庫登入資料儲存為秘密。	雲端管理員
將必要的標籤新增至 PostgreSQL 資料庫執行個體。	**Important** 開啟 Amazon RDS 主控台，並將標籤新增至您要自動備份的 PostgreSQL 資料庫執行個體。您可以在工具區段中使用資料表中的標籤。如果您需要來自相同 Amazon RDS 執行個體內多個 PostgreSQL 資料庫的備份，請使用 `-d test:-d test1`做為`bkp:pgdumpcommand` 標籤的值。 `test`和 `test1`是	雲端管理員

任務	Description	所需的技能
	資料庫名稱。請確定冒號 (:) 後方沒有空格。	
驗證備份自動化。	若要驗證備份自動化，您可以叫用 Lambda 函數或等待備份排程開始。備份程序完成後，請檢查 DynamoDB 清查資料表是否有 PostgreSQL 資料庫執行個體的有效備份項目。如果它們相符，則備份自動化程序會成功。	雲端管理員

相關資源

在 DynamoDB 中建立清查資料表

- 建立 Amazon DynamoDB 資料表

在 AWS Batch 中為失敗的任務事件建立 SNS 主題

- 建立 Amazon SNS 主題
- 針對 AWS Batch 中的失敗任務事件傳送 SNS 提醒

建置 Docker 映像並將其推送至 Amazon ECR 儲存庫

- 建立 Amazon ECR 儲存庫
- 撰寫 Dockerfile、建立 Docker 映像，然後將其推送至 Amazon ECR

建立 AWS Batch 元件

- 建立 AWS Batch 任務定義
- 設定您的運算環境和 AWS Batch 任務佇列
- 在 AWS Batch 中建立任務佇列

建立 Lambda 函數

- 建立 Lambda 函數和寫入程式碼
- 搭配 DynamoDB 使用 Lambda

建立 CloudWatch Events 事件

- 建立以時間為基礎的 CloudWatch Events 事件
- 在 Cloudwatch Events 中使用 Cron 表達式

測試備份自動化

- 建立 Amazon KMS 金鑰
- 建立 Secrets Manager 秘密
- 將標籤新增至 Amazon RDS 執行個體

其他資訊

失敗的任務事件：

```
{
  "detail-type": [
    "Batch Job State Change"
  ],
  "source": [
    "aws.batch"
```

```
  ],
  "detail": {
    "status": [
      "FAILED"
    ]
  }
}
```

Dockerfile 範例：

```
FROM alpine:latest
RUN apk --update add py-pip postgresql-client jq bash && \
pip install awscli && \
rm -rf /var/cache/apk/*
ADD entrypoint.sh /usr/bin/
RUN chmod +x /usr/bin/entrypoint.sh
ENTRYPOINT ["entrypoint.sh"]
```

範例 entrypoint.sh 檔案：

```
 #!/bin/bash
set -e
DATETIME=`date +"%Y-%m-%d_%H_%M"`
FILENAME=RDS_PostGres_dump_${RDS_INSTANCE_NAME}
FILE=${FILENAME}_${DATETIME}

aws configure --profile new-profile set role_arn arn:aws:iam::${TargetAccountId}:role/
${TargetAccountRoleName}
aws configure --profile new-profile set credential_source EcsContainer

echo "Central Account access provider IAM role is: "
aws sts get-caller-identity

echo "Target Customer Account access provider IAM role is: "
aws sts get-caller-identity --profile new-profile

securestring=$(aws secretsmanager get-secret-value --secret-id $SECRETID --output json
 --query 'SecretString' --region=$REGION --profile new-profile)

if [[ ${securestring} ]]; then
    echo "successfully accessed secrets manager and got the credentials"
    export PGPASSWORD=$(echo $securestring | jq --raw-output | jq -r '.DB_PASSWORD')
    PGSQL_USER=$(echo $securestring | jq --raw-output | jq -r '.DB_USERNAME')
```

```
    echo "Executing pg_dump for the PostGres endpoint ${PGSQL_HOST}"
    # pg_dump -h $PGSQL_HOST -U $PGSQL_USER -n dms_sample | gzip -9 -c  | aws s3 cp -
 --region=$REGION  --profile new-profile s3://$BUCKET/$FILE
    # in="-n public:-n private"
    IFS=':' list=($EXECUTE_COMMAND);
    for command in "${list[@]}";
      do
        echo $command;
        pg_dump -h $PGSQL_HOST -U $PGSQL_USER ${command} | gzip -9 -c  | aws s3 cp - --
region=$REGION --profile new-profile s3://${BUCKET}/${FILE}-${command}".sql.gz"
        echo $?;
        if  [[ $? -ne 0 ]]; then
            echo "Error occurred in database backup process. Exiting now....."
            exit 1
        else
            echo "Postgresql dump was successfully taken for the RDS endpoint
 ${PGSQL_HOST} and is uploaded to the following S3 location s3://${BUCKET}/${FILE}-
${command}.sql.gz"
            #write the details into the inventory table in central account
            echo "Writing to DynamoDB inventory table"
            aws dynamodb put-item --table-name ${RDS_POSTGRES_DUMP_INVENTORY_TABLE} --
region=$REGION --item '{ "accountId": { "S": "'"${TargetAccountId}"'" }, "dumpFileUrl":
 {"S": "'"s3://${BUCKET}/${FILE}-${command}.sql.gz"'" }, "DumpAvailableTime": {"S":
 "'"`date +"%Y-%m-%d::%H::%M::%S"` UTC"'"}}'
            echo $?
            if  [[ $? -ne 0 ]]; then
                echo "Error occurred while putting item to DynamoDb Inventory Table.
 Exiting now....."
                exit 1
            else
                echo "Successfully written to DynamoDb Inventory Table
 ${RDS_POSTGRES_DUMP_INVENTORY_TABLE}"
            fi
        fi
      done;
else
    echo "Something went wrong {$?}"
    exit 1
fi

exec "$@"
```

使用 CI/CD 管道在 Amazon EKS 中自動化節點終止處理常式的部署

Sandip Gangapadhyay、Sandeep Gawande、Vyoma Sachdeva、Pragtideep Singh 和 John Vargas，Amazon Web Services

總結

請注意：新客戶不再使用 AWS CodeCommit。AWS CodeCommit 的現有客戶可以繼續正常使用服務。進一步了解

在 Amazon Web Services (AWS) 雲端上，您可以使用開放原始碼專案 AWS Node Termination Handler，正常處理 Kubernetes 內的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體關閉。AWS Node Termination Handler 有助於確保 Kubernetes 控制平面適當回應可能導致 EC2 執行個體無法使用的事件。這類事件包括下列項目：

- EC2 執行個體排程維護
- Amazon EC2 Spot 執行個體中斷
- Auto Scaling 群組縮減
- 跨可用區域重新平衡 Auto Scaling 群組
- 透過 API 或 AWS 管理主控台終止 EC2 執行個體

如果未處理事件，您的應用程式程式碼可能無法正常停止。復原完整可用性也可能需要更長的時間，或者可能會意外地將工作排程到正在停機的節點。`aws-node-termination-handler` (NTH) 可以兩種不同的模式運作：執行個體中繼資料服務 (IMDS) 或佇列處理器。如需這兩種模式的詳細資訊，請參閱讀我檔案。

此模式使用 AWS CodeCommit，並透過持續整合和持續交付 (CI/CD) 管道，使用佇列處理器自動化 NTH 的部署。

> **Note**
>
> 如果您使用的是 EKS 受管節點群組，則不需要 `aws-node-termination-handler`。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 支援搭配 AWS 管理主控台使用的 Web 瀏覽器。請參閱支援的瀏覽器清單。
- 已安裝 AWS 雲端開發套件 (AWS CDK)。
- `kubectl`已安裝 Kubernetes 命令列工具https://kubernetes.io/docs/tasks/tools/。
- `eksctl`已安裝適用於 Amazon Elastic Kubernetes Service (Amazon EKS) 的 AWS Command Line Interface (AWS CLI)https://docs.aws.amazon.com/eks/latest/userguide/eksctl.html。
- 執行 1.20 版或更新版本的 EKS 叢集。
- 連接至 EKS 叢集的自我管理節點群組。若要使用自我管理節點群組建立 Amazon EKS 叢集，請執行下列命令。

```
eksctl create cluster --managed=false --region <region> --name <cluster_name>
```

  如需 的詳細資訊`eksctl`，請參閱 eksctl 文件。
- 叢集的 AWS Identity and Access Management (IAM) OpenID Connect (OIDC) 提供者。如需詳細資訊，請參閱為您的叢集建立 IAM OIDC 提供者。

限制

- 您必須使用支援 Amazon EKS 服務的 AWS 區域。

產品版本

- Kubernetes 1.20 版或更新版本
- `eksctl` 0.107.0 版或更新版本
- AWS CDK 2.27.0 版或更新版本

Architecture

目標技術堆疊

- 虛擬私有雲端 (VPC)
- EKS 叢集
- Amazon Simple Queue Service (Amazon SQS)
- IAM

- Kubernetes

目標架構

下圖顯示節點終止開始時end-to-end步驟的高階檢視。

圖表中顯示的工作流程包含下列高階步驟：

1. 自動擴展 EC2 執行個體終止事件會傳送至 SQS 佇列。
2. NTH Pod 會監控 SQS 佇列中的新訊息。
3. NTH Pod 會收到新訊息並執行下列動作：
    - 繫結節點，讓新的 Pod 不會在節點上執行。
    - 耗盡節點，以清空現有的 Pod
    - 將生命週期掛鉤訊號傳送至 Auto Scaling 群組，以便終止節點。

自動化和擴展

- 程式碼由 AWS CDK 管理和部署，並由 AWS CloudFormation 巢狀堆疊提供支援。
- Amazon EKS 控制平面跨多個可用區域執行，以確保高可用性。
- 對於自動擴展，Amazon EKS 支援 Kubernetes Cluster Autoscaler 和 Karpenter。

工具

AWS 服務

- AWS 雲端開發套件 (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。
- AWS CodeBuild 是一項全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 AWS 上執行 Kubernetes，而無需安裝或維護您自己的 Kubernetes 控制平面或節點。

- Amazon EC2 Auto Scaling 可協助您維持應用程式的可用性，並可讓您根據您定義的條件自動新增或移除 Amazon EC2 執行個體。
- Amazon Simple Queue Service (Amazon SQS) 提供安全、耐用且可用的託管佇列，可協助您整合和分離分散式軟體系統和元件。

其他工具

- kubectl 是針對 Kubernetes 叢集執行命令的 Kubernetes 命令列工具。您可以使用 kubectl 來部署應用程式、檢查和管理叢集資源，以及檢視日誌。

Code

此模式的程式碼可在 GitHub.com 的 deploy-nth-to-eks 儲存庫中取得。 GitHub.com. 程式碼儲存庫包含下列檔案和資料夾。

- `nth folder` – 用於掃描和部署節點終止處理常式之 AWS CloudFormation 範本的 Helm Chart、值檔案和指令碼。
- `config/config.json` – 應用程式的組態參數檔案。此檔案包含部署 CDK 所需的所有參數。
- `cdk` – AWS CDK 原始碼。
- `setup.sh` – 用來部署 AWS CDK 應用程式的指令碼，用來建立必要的 CI/CD 管道和其他必要的資源。
- `uninstall.sh` – 用來清除資源的指令碼。

若要使用範例程式碼，請遵循 Epics 區段中的指示。

最佳實務

如需自動化 AWS Node Termination Handler 時的最佳實務，請參閱下列各項：

- EKS 最佳實務指南
- 節點終止處理常式 - 組態

史詩

設定您的環境

任務	Description	所需的技能

複製儲存庫。	若要使用 SSH（安全殼層）複製儲存庫，請執行下列命令。
	`git clone git@github.com:aws-samples/deploy-nth-to-eks.git`
	若要使用 HTTPS 複製儲存庫，請執行下列命令。
	`git clone https://github.com/aws-samples/deploy-nth-to-eks.git`
	複製儲存庫會建立名為 的資料夾`deploy-nth-to-eks` 。
	變更為該目錄。
	`cd deploy-nth-to-eks`
		應用程式開發人員、AWS DevOps、DevOps 工程師

設定 kubeconfig 檔案。	在終端機中設定您的 AWS 登入資料，並確認您有權擔任叢集角色。您可以使用下列範例程式碼。 `aws eks update-kubeconfig --name <Cluster_Name> --`	AWS DevOps、DevOps 工程師、應用程式開發人員

任務	Description	所需的技能
	`region <region>--role-arn <Role_ARN>`	

部署 CI/CD 管道

任務	Description	所需的技能
設定參數。	在 `config/config.json` 檔案中，設定下列必要參數。 • `pipelineName` ：由 AWS CDK 建立的 CI/CD 管道名稱（例如 `deploy-nth-to-eks-pipeline`)。AWS CodePipeline 會建立具有此名稱的管道。 • `repositoryName` ：要建立的 AWS CodeCommit 儲存庫（例如 `deploy-nth-to-eks-repo`)。AWS CDK 會建立此儲存庫，並將其設定為 CI/CD 管道的來源。 **Note** 此解決方案將建立此 CodeCommit 儲存庫和分支（在下列分支參數中提供）。 • `branch`：儲存庫中的分支名稱（例如 `main`)。對此分支的遞交將啟動 CI/CD 管道。	應用程式開發人員、AWS DevOps、DevOps 工程師

任務	Description	所需的技能
	• `cfn_scan_script` ：用來掃描 AWS CloudFormation 範本 NTH () 的指令碼路徑`scan.sh`。此指令碼存在於將成為 AWS CodeCommit 儲存庫一部分的`nth`資料夾中。 • `cfn_deploy_scrip t`：用來部署 NTH () 之 AWS CloudFormation 範本的指令碼路徑`installApp.sh` 。 • `stackName` ：要部署的 CloudFormation 堆疊名稱。 • `eksClusterName` ：現有 EKS 叢集的名稱。 • `eksClusterRole` ：IAM 角色將用於存取所有 Kubernetes API 呼叫的 EKS 叢集（例如 `clusterad min`)。通常，此角色會新增至 `aws-auth` 。`ConfigMap` • `create_cluster_rol e` ：若要建立 `eksCluste rRole` IAM 角色，請輸入 yes。如果您想要在 `eksClusterRole` 參數中提供現有的叢集角色，請輸入否。 • `create_iam_oidc_pr ovider` ：若要為您的叢集建立 IAM OIDC 提供者，請輸入 yes。如果 IAM OIDC	

任務	Description	所需的技能
	提供者已存在，請輸入否。如需詳細資訊，請參閱為您的叢集建立 IAM OIDC 提供者。 • AsgGroupName ：屬於 EKS 叢集一部分的 Auto Scaling 群組名稱逗號分隔清單（例如 ASG_Group_1,ASG_Group_2 ）。 • region：叢集所在的 AWS 區域名稱（例如 us-east-2 ）。 • install_cdk ：如果 AWS CDK 目前未安裝在機器上，請輸入 yes。執行 cdk --version 命令來檢查已安裝的 AWS CDK 版本是 2.27.0 或更新版本。在這種情況下，請輸入否。 如果您輸入是，https://setup.sh 指令碼會執行 sudo npm install -g cdk@2.27.0 命令，以在機器上安裝 AWS CDK。指令碼需要 sudo 許可，因此請在出現提示時提供帳戶密碼。	

任務	Description	所需的技能
建立 CI/CD 管道以部署 NTH。	執行 setup.sh 指令碼。 `./setup.sh` 指令碼將部署 AWS CDK 應用程式，該應用程式將根據`config/config.json` 檔案中的使用者輸入參數，使用範例程式碼、管道和 CodeBuild 專案來建立 CodeCommit 儲存庫。 此指令碼會在使用 sudo 命令安裝 npm 套件時要求密碼。	應用程式開發人員、AWS DevOps、DevOps 工程師

任務	Description	所需的技能
檢閱 CI/CD 管道。	開啟 AWS 管理主控台，並檢閱在堆疊中建立的下列資源。 • 包含 nth 資料夾內容的 CodeCommit 儲存庫 • AWS CodeBuild 專案 cfn-scan，會掃描 CloudFormation 範本是否有漏洞。 • CodeBuild 專案 Nth-Deploy ，其將透過 AWS CodePipeline 管道部署 AWS CloudFormation 範本和對應的 NTH Helm Chart。AWS CodePipeline • 用於部署 NTH 的 CodePipeline 管道。 管道成功執行後，Helm Release aws-node-termination-handler 會安裝在 EKS 叢集中。此外，名為 的 Pod aws-node-termination-handler 正在叢集的kube-system 命名空間中執行。	應用程式開發人員、AWS DevOps、DevOps 工程師

測試 NTH 部署

任務	Description	所需的技能
模擬 Auto Scaling 群組縮減事件。	若要模擬自動擴展縮減事件，請執行下列動作：	

任務	Description	所需的技能
	1. 在 AWS 主控台上，開啟 EC2 主控台，然後選擇 Auto Scaling 群組。 2. 選取與 中提供的名稱相同的 Auto Scaling 群組`config/config.json` ，然後選擇編輯。 3. 將所需容量和最小容量減少 1。 4. 選擇更新。	
檢閱日誌。	在縮減事件期間，NTH Pod 會封鎖並耗盡對應的工作者節點（將在縮減事件中終止的 EC2 執行個體）。若要檢查日誌，請使用其他資訊區段中的程式碼。	應用程式開發人員、AWS DevOps、DevOps 工程師

清除

任務	Description	所需的技能
清除所有 AWS 資源。	若要清除此模式建立的資源，請執行下列命令。 `./uninstall.sh` 這將透過刪除 CloudFormation 堆疊來清除在此模式中建立的所有資源。	DevOps 工程師

疑難排解

問題	解決方案

npm 登錄檔設定不正確。	在此解決方案的安裝期間，指令碼會安裝 npm 安裝以下載所有必要的套件。如果在安裝期間，您看到一則訊息，指出「找不到模組」npm 登錄檔可能設定不正確。若要查看目前的登錄設定，請執行下列命令。
	`npm config get registry`
	若要使用 設定登錄檔`https://registry.npmjs.org/` ，請執行下列命令。
	`npm config set registry https://registry.npmjs.org`

延遲 SQS 訊息傳遞。	在疑難排解過程中，如果您想要延遲 SQS 訊息交付至 NTH Pod，您可以調整 SQS 交付延遲參數。如需詳細資訊，請參閱 Amazon SQS 延遲佇列。

相關資源

- AWS 節點終止處理常式原始碼
- EC2 研討會
- AWS CodePipeline
- Amazon Elastic Kubernetes Service (Amazon EKS)
- AWS 雲端開發套件
- AWS CloudFormation

其他資訊

1. 尋找 NTH Pod 名稱。

```
kubectl get pods -n kube-system |grep aws-node-termination-handler
aws-node-termination-handler-65445555-kbqc7   1/1     Running   0          26m
kubectl get pods -n kube-system |grep aws-node-termination-handler
aws-node-termination-handler-65445555-kbqc7   1/1     Running   0          26m
```

2. 檢查日誌。範例日誌如下所示。它顯示節點已在傳送 Auto Scaling 群組生命週期掛鉤完成訊號之前封鎖並耗盡。

```
kubectl -n kube-system logs aws-node-termination-handler-65445555-kbqc7
022/07/17 20:20:43 INF Adding new event to the event store
 event={"AutoScalingGroupName":"eksctl-my-cluster-target-nodegroup-
ng-10d99c89-NodeGroup-ZME36IGAP7O1","Description":"ASG Lifecycle Termination
 event received. Instance will be interrupted at 2022-07-17 20:20:42.702
 +0000 UTC \n","EndTime":"0001-01-01T00:00:00Z","EventID":"asg-lifecycle-
term-33383831316538382d353564362d343332362d613931352d383430666165636334333564","InProgress":fal
east-2.compute.internal","NodeProcessed":false,"Pods":null,"ProviderID":"aws:///us-
east-2c/i-0409f2a9d3085b80e","StartTime":"2022-07-17T20:20:42.702Z","State":""}
2022/07/17 20:20:44 INF Requesting instance drain event-id=asg-lifecycle-
term-33383831316538382d353564362d343332362d613931352d383430666165636334333564
 instance-id=i-0409f2a9d3085b80e kind=SQS_TERMINATE node-name=ip-192-168-75-60.us-
east-2.compute.internal provider-id=aws:///us-east-2c/i-0409f2a9d3085b80e
2022/07/17 20:20:44 INF Pods on node node_name=ip-192-168-75-60.us-
east-2.compute.internal pod_names=["aws-node-qchsw","aws-node-termination-
handler-65445555-kbqc7","kube-proxy-mz5x5"]
2022/07/17 20:20:44 INF Draining the node
2022/07/17 20:20:44 ??? WARNING: ignoring DaemonSet-managed Pods: kube-system/aws-node-
qchsw, kube-system/kube-proxy-mz5x5
2022/07/17 20:20:44 INF Node successfully cordoned and drained
 node_name=ip-192-168-75-60.us-east-2.compute.internal reason="ASG Lifecycle
 Termination event received. Instance will be interrupted at 2022-07-17 20:20:42.702
 +0000 UTC \n"
2022/07/17 20:20:44 INF Completed ASG Lifecycle Hook (NTH-K8S-TERM-HOOK) for instance
 i-0409f2a9d3085b80e
```

使用 CI/CD 管道自動建置 Java 應用程式並將其部署到 Amazon EKS

MAHESH RAGHUNANDANAN、Jomcy Pappachen 和 Amazon Web Services 的 James Radtke

總結

此模式說明如何建立持續整合和持續交付 (CI/CD) 管道，以使用建議的 DevSecOps 實務自動建置和部署 Java 應用程式至 上的 Amazon Elastic Kubernetes Service (Amazon EKS) 叢集 AWS 雲端。此模式使用使用 Spring Boot Java 架構開發並使用 Apache Maven 的問候語應用程式。

您可以使用此模式的方法來建置 Java 應用程式的程式碼、將應用程式成品封裝為 Docker 映像、安全性掃描映像，以及將映像上傳為 Amazon EKS 上的工作負載容器。如果您想要從緊密耦合的單體架構遷移到微服務架構，此模式的方法很有用。此方法也可協助您監控和管理 Java 應用程式的整個生命週期，確保更高層級的自動化，並協助避免錯誤。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- AWS Command Line Interface (AWS CLI) 第 2 版，已安裝並設定。如需詳細資訊，請參閱 AWS CLI 文件中的安裝或更新至最新版本的 AWS CLI。

  AWS CLI 第 2 版必須使用建立 Amazon EKS 叢集的相同 AWS Identity and Access Management (IAM) 角色進行設定，因為只有該角色有權將其他 IAM 角色新增至 `aws-auth ConfigMap`。如需設定的資訊和步驟 AWS CLI，請參閱 AWS CLI 文件中的設定設定。
- 具有完整存取權的 IAM 角色和許可 AWS CloudFormation。如需詳細資訊，請參閱 CloudFormation 文件中的使用 IAM 控制存取。
- 現有的 Amazon EKS 叢集，其中包含 IAM 角色名稱的詳細資訊，以及 EKS 叢集中工作者節點的 IAM 角色 Amazon Resource Name (ARN)。
- 在 Amazon EKS 叢集中安裝和設定 Kubernetes Cluster Autoscaler。如需詳細資訊，請參閱 Amazon EKS 文件中的使用 Karpenter 和 Cluster Autoscaler 擴展叢集運算。
- 存取 GitHub 儲存庫中的程式碼。

> **Important**
>
> AWS Security Hub CSPM 會啟用 做為範本的一部分，這些 CloudFormation 範本包含在此模式的程式碼中。根據預設，Security Hub CSPM 啟用後，會隨附 30 天的免費試用。試用

之後，會產生與此相關的成本 AWS 服務。如需定價的詳細資訊，請參閱 AWS Security Hub CSPM 定價。

產品版本

- Helm 3.4.2 版或更新版本
- Apache Maven 3.6.3 版或更新版本
- BridgeCrew Checkov 2.2 版或更新版本
- Aqua Security Trivy 0.37 版或更新版本

Architecture

技術堆疊

- AWS CodeBuild
- AWS CodeCommit
- Amazon CodeGuru
- AWS CodePipeline
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon EKS
- Amazon EventBridge
- AWS Security Hub CSPM
- Amazon Simple Notification Service (Amazon SNS)

目標架構

該圖顯示以下工作流程：

1. 開發人員會更新 CodeCommit 儲存庫基本分支中的 Java 應用程式程式碼，這會建立提取請求 (PR)。
2. 提交 PR 後，Amazon CodeGuru Reviewer 會自動檢閱程式碼、根據 Java 的最佳實務進行分析，並向開發人員提供建議。
3. 將 PR 合併至基本分支後，會建立 Amazon EventBridge 事件。

4. EventBridge 事件會啟動啟動的 CodePipeline 管道。
5. CodePipeline 執行 CodeSecurity Scan 階段（持續安全性）。
6. AWS CodeBuild 會啟動安全掃描程序，其中使用 Checkov 掃描 Dockerfile 和 Kubernetes 部署 Helm 檔案，並根據增量程式碼變更掃描應用程式原始碼。應用程式原始碼掃描是由 CodeGuru Reviewer Command Line Interface (CLI) 包裝函式執行。

> **Note**
>
> 自 2025 年 11 月 7 日起，您無法在 Amazon CodeGuru Reviewer 中建立新的儲存庫關聯。若要了解具有類似 CodeGuru Reviewer 功能的服務，請參閱 CodeGuru Reviewer 文件中的 Amazon CodeGuru Reviewer 可用性變更。 CodeGuru

7. 如果安全掃描階段成功，則會啟動建置階段（持續整合）。
8. 在建置階段，CodeBuild 會建置成品、將成品封裝至 Docker 映像、使用 Aqua Security Trivy 掃描映像是否有安全漏洞，並將映像存放在 Amazon ECR 中。
9. 從步驟 8 偵測到的漏洞會上傳至 Security Hub CSPM，供開發人員或工程師進一步分析。Security Hub CSPM 提供修復漏洞的概觀和建議。
10. CodePipeline 管道中循序階段的電子郵件通知會透過 Amazon SNS 傳送。
11. 持續整合階段完成後，CodePipeline 會進入部署階段（持續交付）。
12. Docker 映像會使用 Helm Chart 以容器工作負載 (Pod) 部署至 Amazon EKS。
13. 應用程式 Pod 是使用 Amazon CodeGuru Profiler 代理程式設定，會將應用程式的分析資料 (CPU、堆積用量和延遲）傳送至 CodeGuru Profiler，協助開發人員了解應用程式的行為。

工具

AWS 服務

- CloudFormation 可協助您設定 AWS 資源、快速一致地佈建資源，以及在整個 AWS 帳戶 和 區域的生命週期中管理資源。
- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。
- Amazon CodeGuru Profiler 會從即時應用程式收集執行時間效能資料，並提供可協助您微調應用程式效能的建議。

- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 上執行 Kubernetes， AWS 而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與各種來源的即時資料連線，包括 AWS Lambda 函數、使用 API 目的地的 HTTP 呼叫端點，或其他事件匯流排 AWS 帳戶。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Security Hub CSPM 提供安全狀態的完整檢視 AWS。它還可協助您根據安全產業標準和最佳實務來檢查 AWS 環境。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他服務

- Helm 是 Kubernetes 的開放原始碼套件管理員。
- Apache Maven 是軟體專案管理和理解工具。
- BridgeCrew Checkov 是一種靜態程式碼分析工具，可將基礎設施掃描為程式碼 (IaC) 檔案，以找出可能導致安全或合規問題的錯誤組態。
- Aqua Security Trivy 是全方位的掃描器，除了組態問題之外，還可找出容器映像、檔案系統和 Git 儲存庫中的漏洞。

Code

此模式的程式碼可在 GitHub aws-codepipeline-devsecops-amazoneks 儲存庫中使用。

最佳實務

- 此模式遵循 IAM 安全最佳實務，以在解決方案的所有階段套用 IAM 實體的最低權限原則。如果您想要使用其他 AWS 服務 或第三方工具擴展解決方案，建議您檢閱在 IAM 文件中套用最低權限許可的章節。
- 如果您有多個 Java 應用程式，建議您為每個應用程式建立個別的 CI/CD 管道。
- 如果您有整體應用程式，我們建議您盡可能將應用程式分成微型服務。微服務更具彈性，可讓您更輕鬆地將應用程式部署為容器，而且可讓您更清楚地了解應用程式的整體建置和部署。

史詩

設定環境

任務	Description	所需的技能
複製 GitHub 儲存庫。	若要複製儲存庫，請執行下列命令。 `git clone https://github.com/aws-samples/aws-codepipeline-devsecops-amazoneks`	應用程式開發人員、DevOps 工程師
建立 S3 儲存貯體並上傳程式碼。	1. 登入 AWS 管理主控台，開啟 Amazon S3 主控台，然後在您計劃部署此解決方案 AWS 區域 的 中建立 S3 儲存貯體。如需詳細資訊，請參閱 Amazon S3 文件中的建立儲存貯體。 2. 在 S3 儲存貯體中，建立名為 的資料夾`code`。 3. 導覽至您複製儲存庫的位置。若要使用 .zip 副檔名 (`cicdstack.zip`) 建立整個程式碼的壓縮版本，並驗	AWS DevOps、雲端管理員、DevOps 工程師

任務	Description	所需的技能

	證 .zip 檔案，請依序執行下列命令。
	
	`cd aws-codepipeline-devsecops-amazoneks`
	`python -m zipfile -c cicdstack.zip *`
	`python -m zipfile -t cicdstack.zip`
	
	Note
	如果python命令失敗並指出找不到 Python，請python3改用 。
	
	4. 將`cicdstack.zip` 檔案上傳至您先前在 S3 儲存貯體中建立的程式碼資料夾。	

任務	Description	所需的技能
建立 CloudFormation 堆疊。	1. 開啟 CloudFormation 主控台，然後選擇 Create stack (建立堆疊)。 2. 在指定範本中，選擇上傳範本檔案、上傳`cf_templates/codecommit_ecr.yaml` 檔案，然後選擇下一步。 3. 在指定堆疊詳細資訊中，輸入堆疊名稱，然後提供下列輸入參數值： • `CodeCommitRepositoryBranchName` ：程式碼所在的分支名稱 （預設值為 `main`) • `CodeCommitRepositoryName` ：要建立的 CodeCommitrepository 名稱 • `CodeCommitRepositoryS3Bucket` ：您建立程式碼資料夾的 S3 儲存貯體名稱 • `CodeCommitRepositoryS3BucketObjKey` ：`code/cicdstack.zip` • `ECRRepositoryName` ：要建立的 Amazon ECR 儲存庫名稱 4. 選擇下一步，使用設定堆疊選項的預設設定，然後選擇下一步。	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	5. 在檢閱區段中，驗證範本和堆疊詳細資訊，然後選擇建立堆疊。接著會建立堆疊，包括 CodeCommit 和 Amazon ECR 儲存庫。 6. 請注意 CodeCommit 和 Amazon ECR 儲存庫的名稱，這將是設定 Java CI/CD 管道的必要項目。	
驗證 CloudFormation 堆疊部署。	1. 在 CloudFormation 主控台的 Stacks 下，驗證您部署的 CloudFormation 堆疊狀態。堆疊的狀態應為 CREATE COMPLETE。 2. 從 主控台，驗證 CloudFormation 和 Amazon ECR 儲存庫是否已佈建並準備就緒。	AWS DevOps，DevOps 工程師
刪除 S3 儲存貯體。	清空並刪除您先前建立的 S3 儲存貯體。如需詳細資訊，請參閱 Amazon S3 文件中的刪除儲存貯體。	AWS DevOps，DevOps 工程師

設定 Helm Chart

任務	Description	所需的技能
設定 Java 應用程式的 Helm Chart。	1. 在您複製 GitHub 儲存庫的位置，導覽至資料夾 `helm_charts/aws-proserve-java-greeting` 。在此資料夾中，`values.dev.yaml` 檔案	DevOps 工程師

任務	Description	所需的技能

	包含 Kubernetes 資源組態的相關資訊，您可以將容器部署修改為 Amazon EKS。提供您的 AWS 帳戶 ID 和 Amazon ECR 儲存庫名稱 AWS 區域，以更新 Docker 儲存庫參數。
	`image:`
	`  repository:`
	`<account-id>.dkr.ecr.<region>.amazonaws.com/<app-ecr-repo-name>`
	2. Java Pod 的服務類型設定為 `LoadBalancer` 。
	`service:`
	`  type: LoadBalancer`
	`  port: 80`
	`  targetPort: 8080`
	`  path: /hello`
	`  initialDelaySeconds: 60`
	`  periodSeconds: 30`
	若要使用不同的服務（例如 NodePort），您可以變更此參數。如需詳細資訊，請參閱 Kubernetes 文件。
	3. 您可以將 `autoscaling` 參數變更為 ，以啟用 Kubernetes Horizontal Pod Autoscaler`enabled: true`。	

任務	Description	所需的技能

	<pre>autoscaling:
	  enabled: true
	  minReplicas: 1
	  maxReplicas: 100
	  targetCPUUtilizati
	onPercentage: 80
	  # targetMem
	oryUtilizationPerc
	entage: 80</pre>
	4. 您可以透過變更 `values.<ENV>.yaml` 檔案中的值來啟用 Kubernetes 工作負載的不同功能，其中 <ENV>是您的開發、生產、UTA 或 QA 環境。	

驗證 Helm Chart 是否有語法錯誤。	1. 從終端機執行下列命令，確認 Helm v3 已安裝在您的本機工作站中。
	<pre>helm --version</pre>
	如果未安裝 Helm v3，請進行安裝。
	2. 在終端機中，導覽至 Helm Charts 目錄 (`helm_charts/aws-proserve-java-greeting`)，然後執行下列命令。
	<pre>helm lint . -f
	 values.dev.yaml</pre>
	這將檢查 Helm Chart 是否有任何語法錯誤。
		DevOps 工程師

設定 Java CI/CD 管道

任務	Description	所需的技能
建立 CI/CD 管道。	1. 開啟 CloudFormation 主控台，然後選擇 Create stack (建立堆疊)。 2. 在指定範本中，選擇上傳範本檔案、上傳`cf_templates/build_deployment.yaml` 範本，然後選擇下一步。 3. 在指定堆疊詳細資訊中，指定堆疊名稱，然後為輸入參數提供下列值： • `CodeBranchName` ：程式碼所在之 CodeCommit 儲存庫的分支名稱 • `EKSClusterName` ：EKS 叢集的名稱 （非 `EKSCluster` ID) • `EKSCodeBuildAppName` ：應用程式 Helm Chart 的名稱 (`aws-proserve-java-greeting`) • `EKSWorkerNodeRoleARN` ：指派給 Amazon EKS 工作者節點之 IAM 角色的 ARN • `EKSWorkerNodeRoleName` ：指派給 Amazon EKS 工作者節點的 IAM 角色名稱	AWS DevOps

任務	Description	所需的技能
	• `EcrDockerRepository` ：存放程式碼 Docker 影像的 Amazon ECR 儲存庫名稱 • `EmailRecipient` ：應傳送組建通知的電子郵件地址 • `EnvType`：環境（例如，dev、test 或 prod) • `SourceRepoName` ：程式碼所在的 CodeCommit 儲存庫名稱 4. 選擇下一步。使用設定堆疊選項中的預設設定，然後選擇下一步。 5. 在檢閱區段中，驗證 CloudFormation 範本和堆疊詳細資訊，然後選擇下一步。 6. 選擇建立堆疊。 7. 在 CloudFormation 堆疊部署期間，您在參數中提供的電子郵件地址擁有者會收到訂閱 SNS 主題的訊息。若要訂閱 Amazon SNS，擁有者必須在訊息中選擇連結。 8. 建立堆疊後，開啟堆疊的輸出索引標籤，然後記錄 `EksCodeBuildkubeRoleARN` 輸出金鑰的 ARN 值。當您提供 CodeBuild IAM 角色在 Amazon EKS 叢集中部署工作負載的許可	

任務	Description	所需的技能
	時，稍後會需要此 IAM ARN 值。	

啟用 Security Hub CSPM 和 Aqua Security 之間的整合

任務	Description	所需的技能
開啟 Aqua Security 整合。	將 Trivy 報告的 Docker 映像漏洞調查結果上傳到 Security Hub CSPM 時需要此步驟。由於 CloudFormation 不支援 Security Hub CSPM 整合，此程序必須手動完成。 1. 開啟 AWS Security Hub CSPM 主控台，然後導覽至整合。 2. 搜尋 Aqua Security，然後選取 Aqua Security：Aqua Security。 3. 選擇接受問題清單。	AWS 管理員、DevOps 工程師

設定 CodeBuild 以執行 Helm 或 kubectl 命令

任務	Description	所需的技能
允許 CodeBuild 在 Amazon EKS 叢集中執行 Helm 或 kubectl 命令。	若要讓 CodeBuild 進行身分驗證以搭配 Amazon EKS 叢集使用 Helm 或`kubectl`命令，您必須將 IAM 角色新增至 `aws-auth ConfigMap` 。在此情況下，請新增 IAM 角色 的 ARN`EksCodeBuildkubeRoleARN` ，這是為 CodeBuild	DevOps

任務	Description	所需的技能

	服務建立的 IAM 角色，以存取 Amazon EKS 叢集並在其中部署工作負載。這是一次性的活動。
	
	Important
	下列程序必須在 CodePipeline 中的部署核准階段之前完成。
	
	1. 在 Amazon Linux 或 macOS 環境中開啟 `cf_templates/kube_aws_auth_configmap_patch.sh` shell 指令碼。
	2. 執行下列命令來驗證 Amazon EKS 叢集。
	
	`aws eks --region <aws-region> update-kubeconfig --name <eks-cluster-name>`
	
	3. 使用以下命令執行 shell 指令碼，`<rolearn-eks-codebuild-kubectl>` 將 取代`EksCodeBuildkubeRoleARN` 為您先前記錄的 ARN 值。
	
	`bash cf_templates/kube_aws_auth_configmap_patch.sh`	

任務	Description	所需的技能
	<rolearn-eks-codebuild-kubectl> aws_auth ConfigMap 已設定，並授予存取權。	

驗證 CI/CD 管道

任務	Description	所需的技能
確認 CI/CD 管道會自動啟動。	1. 如果 Checkov 偵測到 Dockerfile 或 Helm Chart 中的漏洞，管道中的 CodeSecurity 掃描階段通常會失敗。不過，此範例的目的是建立識別潛在安全漏洞的程序，而不是透過 CI/CD 管道修正，通常是 DevSecOps 程序。在 檔案中buildspec/buildspec_secscan.yaml ，checkov命令會使用 --soft-fail 旗標來避免管道故障。 - echo -e "\n Running Dockerfile Scan" - checkov -f code/app/Dockerfile --framework dockerfile --soft-fail --summary-position bottom - echo -e "\n Running Scan of Helm Chart files"	DevOps

任務	Description	所需的技能
	``` - cp -pv helm_char ts/$EKS_CODEBUILD_ APP_NAME/values.de v.yaml helm_char ts/$EKS_CODEBUILD_ APP_NAME/values.ya ml - checkov -d helm_char ts/$EKS_CODEBUILD_ APP_NAME --framewo rk helm --soft-fail --summary-position bottom - rm -rfv helm_char ts/$EKS_CODEBUILD_ APP_NAME/values.ya ml ``` 若要讓管道在回報 Dockerfil e 和 Helm Chart 的漏洞時失敗，必須從 checkov命令中移除 `--soft-fail` 選項。然後，開發人員或工程師可以修正漏洞，並將變更遞交至 CodeCommit 原始程式碼儲存庫。 2. 與 CodeSecurity Scan 類似，建置階段會使用 Aqua Security Trivy 在將應用程式推送至 Amazon ECR 之前識別 `HIGH`和 `CRITICAL` Docker 映像漏洞。 ``` - AWS_REGION=$AWS_DE FAULT_REGION AWS_ACCOUNT_ID=$AW S_ACCOUNT_ID trivy - d image --no-progress	

任務	Description	所需的技能
	<pre>--ignore-unfixed --exit-code 0 --severity HIGH,CRITICAL --format template --template "@securityhub/asff.tpl" -o securityhub/report.asff $AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/$IMAGE_REPO_NAME:$CODEBUILD_RESOLVED_SOURCE_VERSION</pre> 在此範例中，回報 Docker 映像漏洞時管道不會失敗，因為buildspec/buildspec.yml 檔案中的 trivy命令包含值為 的旗標 --exit-code 0。若要讓管道在回報 HIGH和 CRTICAL 漏洞時失敗，請將的值變更為 --exit-code 1。然後，開發人員或工程師可以修正漏洞，並將變更遞交至 CodeCommit 原始程式碼儲存庫。 3. Aqua Security Trivy 報告的 Docker 映像漏洞會上傳至 Security Hub CSPM。在 Security Hub CSPM 主控台上，導覽至問題清單。使用記錄狀態 = Active 和產品 = Aqua Security 篩選問題清單。這列出 Security Hub CSPM 中的 Docker 映	

任務	Description	所需的技能
	像漏洞。漏洞可能需要 15 分鐘到一小時才會出現在 Security Hub CSPM 中。 如需使用 CodePipeline 啟動管道的詳細資訊，請參閱 CodePipeline 文件中的在 CodePipeline 中啟動管道、手動啟動管道，以及依排程啟動管道。 CodePipeline	
核准部署。	1. 建置階段完成後，會有部署核准閘道。檢閱者或發行管理員應檢查組建，如果符合所有要求，請予以核准。對於使用持續交付進行應用程式部署的團隊，建議使用此方法。 2. 核准後，管道會啟動部署階段。 3. 部署階段成功後，此階段的 CodeBuild 日誌會提供應用程式的 URL。使用 URL 驗證應用程式的準備程度。	DevOps

任務	Description	所需的技能
驗證應用程式分析。	部署完成且應用程式 Pod 部署在 Amazon EKS 之後，在應用程式中設定的 Amazon CodeGuru Profiler 代理程式會嘗試將應用程式的分析資料 (CPU、堆積摘要、延遲和瓶頸）傳送至 CodeGuru Profiler。 對於應用程式的初始部署，CodeGuru Profiler 大約需要 15 分鐘才能視覺化分析資料。	AWS DevOps

相關資源

- AWS CodePipeline 文件
- 在 中使用 Trivy 掃描影像 AWS CodePipeline(AWS 部落格文章）
- 使用 Amazon CodeGuru Profiler 改善 Java 應用程式 (AWS 部落格文章）
- AWS 安全調查結果格式 (ASFF) 語法
- Amazon EventBridge 事件模式
- Helm 升級

其他資訊

- CodeGuru Profiler 在功能 AWS X-Ray 方面不應與服務混淆。我們建議您使用 CodeGuru Profiler 來識別可能導致瓶頸或安全問題的最昂貴程式碼行，並在它們成為潛在風險之前對其進行修正。X-Ray 服務用於應用程式效能監控。
- 在此模式中，事件規則會與預設事件匯流排相關聯。如有需要，您可以擴展模式以使用自訂事件匯流排。
- 此模式使用 CodeGuru Reviewer 作為應用程式碼的靜態應用程式安全測試 (SAST) 工具。您也可以將此管道用於其他工具，例如 SonarQube 或 Checkmarx。您可以將任何這些工具的掃描設定指示新增至 ，`buildspec/buildspec_secscan.yaml`以取代 CodeGuru 掃描指示。

> **Note**
>
> 自 2025 年 11 月 7 日起，您無法在 Amazon CodeGuru Reviewer 中建立新的儲存庫關聯。若要了解具有類似 CodeGuru Reviewer 功能的服務，請參閱 CodeGuru Reviewer 文件中的 Amazon CodeGuru Reviewer 可用性變更。 CodeGuru

跨 AWS 帳戶 和 複製 Amazon ECR 容器映像 AWS 區域

Faisal Shahdad，Amazon Web Services

總結

此模式說明如何使用無伺服器方法，將現有 Amazon Elastic Container Registry (Amazon ECR) 儲存庫中的標記影像複寫至其他 AWS 帳戶 和 AWS 區域。解決方案使用 AWS Step Functions 來管理複寫工作流程，並使用 AWS Lambda 函數來複製大型容器映像。

Amazon ECR 使用原生跨區域和跨帳戶複寫功能，跨區域和帳戶複寫容器映像。但這些功能只會從開啟複寫的那一刻開始複寫映像。沒有可在不同區域和帳戶中複寫現有映像的機制。

此模式可協助人工智慧 (AI) 團隊將容器化機器學習 (ML) 模型、架構 （ 例如 PyTorch、TensorFlow 和 Hugging Face) 和相依性分發給其他帳戶和區域。這可協助您克服服務限制並最佳化 GPU 運算資源。您也可以選擇性地從特定來源帳戶和區域複寫 Amazon ECR 儲存庫。如需詳細資訊，請參閱 Amazon ECR 中的跨區域複寫已登陸。

先決條件和限制

先決條件

- 兩個或多個作用中 AWS 帳戶 （ 一個來源帳戶和一個目的地帳戶，至少 ）
- 所有帳戶中的適當 AWS Identity and Access Management (IAM) 許可
- 用於建置 Lambda 容器映像的 Docker
- AWS Command Line Interface 為所有帳戶設定的 (AWS CLI)

限制

- 未標記的映像排除 – 解決方案只會複製具有明確標籤的容器映像。它會略過與SHA256摘要一起存在的未標記影像。
- Lambda 執行逾時限制 – AWS Lambda 限制為最長 15 分鐘的執行逾時，這可能不足以複製大型容器映像或儲存庫。
- 手動容器映像管理 – `crane-app.py` Python 程式碼需要重建和重新部署 Lambda 容器映像。
- 有限的平行處理容量 – `MaxConcurrency` 狀態設定會限制您可以同時複製的儲存庫數量。不過，您可以在來源帳戶的 AWS CloudFormation 範本中修改此設定。請注意，較高的並行值可能會導致您超過服務速率限制和帳戶層級的 Lambda 執行配額。

Architecture

目標堆疊

模式有四個主要元件：

- 來源帳戶基礎設施 – 建立協同運作元件的 CloudFormation 範本
- 目的地帳戶基礎設施 – 建立跨帳戶存取角色的 CloudFormation 範本
- Lambda 函數 – 使用 Crane 進行高效率映像複製的 Python 函數
- 容器映像 – 使用必要工具封裝 Lambda 函數的 Docker 容器

目標架構

Step Functions 工作流程

Step Functions 狀態機器會協調下列項目，如下圖所示：

- `PopulateRepositoryList` – 掃描 Amazon ECR 儲存庫並填入 Amazon DynamoDB
- `GetRepositoryList` – 從 DynamoDB 擷取唯一儲存庫清單
- `DeduplicateRepositories` – 確保沒有重複的處理
- `CopyRepositories` – 處理儲存庫的平行複製
- `NotifySuccess/NotifyFailure` – 根據執行結果的 Amazon Simple Notification Service (Amazon SNS) 通知

工具

Amazon 工具

- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您在 上執行的應用程式。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。

- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 Lambda 函數和其他 AWS 服務來建置業務關鍵應用程式。

其他工具

- Crane 是 Docker 協同運作工具。它類似於 Docker Compose，但有其他功能。
- Docker 是一組平台即服務 (PaaS) 產品，可在作業系統層級使用虛擬化在容器中交付軟體。

程式碼儲存庫

- 此模式的程式碼可在 GitHub sample-ecr-copy 儲存庫中使用。您可以從儲存庫使用 CloudFormation 範本來建立基礎資源。

最佳實務

遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。

史詩

準備您的環境

任務	Description	所需的技能

設定 AWS CLI 設定檔。	1. 設定來源帳戶設定檔：
	`aws configure --profile source-account`
	`# Enter: Access Key ID, Secret Access Key, Default region, Output format (json)`
	2. 設定目的地帳戶設定檔：
		DevOps 工程師、資料工程師、ML 工程師

任務	Description	所需的技能
	``` aws configure -- profile destination- account # Enter: Access Key ID, Secret Access Key, Default region, Output format (json) ```	
	3. 驗證組態： ``` aws sts get-caller- identity --profile source-account aws sts get-caller- identity --profile destination-account ```	

任務	Description	所需的技能

收集必要資訊。	1. 取得來源帳戶 ID：
	`export SOURCE_ACCOUNT_ID=$(aws sts get-caller-identity --profile source-account --query Account --output text)`
	`echo "Source Account ID: $SOURCE_ACCOUNT_ID"`
	2. 取得目的地帳戶 ID：
	`export DEST_ACCOUNT_ID=$(aws sts get-caller-identity --profile destination-account --query Account --output text)`
	`echo "Destination Account ID: $DEST_ACCOUNT_ID"`
	3. 設定 AWS 區域。為您的區域修改此命令：
	`export SOURCE_REGION="us-east-1"`
	`export DEST_REGION="us-east-2"`
	4. 列出來源帳戶中現有的 Amazon ECR 儲存庫：
	`aws ecr describe-repositories --profile source-account --region`
		DevOps 工程師、資料工程師、ML 工程師

任務	Description	所需的技能
	`$SOURCE_REGION --query 'repositories[].repositoryName' --output table`	
複製儲存庫。	將模式的儲存庫複製到本機工作站： `git clone https://github.com/aws-samples/sample-ecr-copy`	DevOps 工程師、資料工程師、ML 工程師

部署目的地帳戶的基礎設施

任務	Description	所需的技能
驗證範本。	驗證 CloudFormation 範本： `aws cloudformation validate-template \ --template-body file://"Destination Account cf_template.yml" \ --profile destination-account`	DevOps 工程師、ML 工程師、資料工程師
部署目的地基礎設施。	1. 部署目的地帳戶堆疊： `aws cloudformation deploy \ --template-file "Destination Account cf_template.yml" \ --stack-name ecr-copy-destination \`	資料工程師、ML 工程師、DevOps 工程師

任務	Description	所需的技能

	<pre> --parameter-overri
	des \
	    SourceAccountId=
	$SOURCE_ACCOUNT_ID \
	    SourceRol
	eName=ECRContainer
	LambdaRole \
	  --capabilities
	 CAPABILITY_NAMED_I
	AM \
	  --profile destinati
	on-account \
	  --region $DEST_REG
	ION</pre>
	2. 等待堆疊完成：
	<pre>aws cloudformation
	 wait stack-create-
	complete \
	  --stack-name ecr-
	copy-destination \
	  --profile destinati
	on-account \
	  --region $DEST_REG
	ION</pre>	

任務	Description	所需的技能
驗證部署。	1. 取得堆疊輸出： ``` aws cloudformation describe-stacks \ --stack-name ecr- copy-destination \ --profile destinati on-account \ --region $DEST_REG ION \ --query 'Stacks[0].Outputs' \ --output table ``` 2. 存放跨帳戶 IAM 角色： ``` export CROSS_ACC OUNT_ROLE_ARN=$(aw s cloudformation describe-stacks \ --stack-name ecr- copy-destination \ --profile destinati on-account \ --region $DEST_REG ION \ --query 'Stacks[0].Outputs[?OutputK ey==`CrossAccountR oleArn`].OutputVal ue' \ --output text) echo "Cross-Account Role ARN: $CROSS_AC COUNT_ROLE_ARN" ```	DevOps 工程師、ML 工程師、資料工程師

建置和部署 Lambda 容器映像

任務	Description	所需的技能

準備容器建置。	1. 確認 Docker 正在執行：
	`docker --version`
	`docker info`
	2. 確定 `crane-app.py` 和 `Dockerfile` 位於目前的目錄中：
	`ls -la crane-app.py Dockerfile`
		資料工程師、ML 工程師、DevOps 工程師

建置容器映像。	1. 建置 Lambda 容器映像：
	`docker build -t ecr-copy-lambda . --no-cache`
	2. 確認已建立映像：
	`docker images ecr-copy-lambda`
	3. （選用）在本機測試容器：
	`docker run --rm --entrypoint python ecr-copy-lambda -c "import boto3; print('Container working')"`
		資料工程師、ML 工程師、DevOps 工程師

建立儲存庫並上傳映像。	1. 在來源帳戶中建立 Amazon ECR 儲存庫：	資料工程師、ML 工程師、DevOps 工程師

任務	Description	所需的技能

	<pre>aws ecr create-repository \
	  --repository-name ecr-copy-lambda \
	  --profile source-account \
	  --region $SOURCE_REGION</pre>
	2. 取得 Amazon ECR 登入字符並驗證 Docker：
	<pre>aws ecr get-login-password \
	  --profile source-account \
	  --region $SOURCE_REGION | \
	  docker login --username AWS --password-stdin \
	  $SOURCE_ACCOUNT_ID.dkr.ecr.$SOURCE_REGION.amazonaws.com</pre>
	3. 標記 Amazon ECR 的映像：
	<pre>docker tag ecr-copy-lambda:latest \
	  $SOURCE_ACCOUNT_ID.dkr.ecr.$SOURCE_REGION.amazonaws.com/ecr-copy-lambda:latest</pre>
	4. 將映像上傳至 Amazon ECR：	

任務	Description	所需的技能

	`docker push $SOURCE_ACCOUNT_ID.dkr.ecr.$SOURCE_REGION.amazonaws.com/ecr-copy-lambda:latest`
	5. 存放映像 URI 以供日後使用：
	`export LAMBDA_IMAGE_URI="$SOURCE_ACCOUNT_ID.dkr.ecr.$SOURCE_REGION.amazonaws.com/ecr-copy-lambda:latest"`
	`echo "Lambda Image URI: $LAMBDA_IMAGE_URI"`	

任務	Description	所需的技能

驗證映像。	1. 列出儲存庫中的映像：
	`aws ecr list-images \ --repository-name ecr-copy-lambda \ --profile source-account \ --region $SOURCE_REGION`
	2. 取得映像詳細資訊：
	`aws ecr describe-images \ --repository-name ecr-copy-lambda \ --profile source-account \ --region $SOURCE_REGION`
		資料工程師、ML 工程師、DevOps 工程師

部署來源帳戶基礎設施

任務	Description	所需的技能

準備部署參數。	1. 設定通知電子郵件：
	`export NOTIFICATION_EMAIL="your-email@company.com"`
	2. 定義要複製的儲存庫（逗號分隔）：
	`export REPOSITORY_LIST="app-fronte`
		資料工程師、DevOps 工程師、ML 工程師

任務	Description	所需的技能
	nd,app-backend,dat abase-migrations" 3. 設定環境： export ENVIRONME NT="dev" echo "Deployment Parameters:" echo "Source Account: $SOURCE_ACCOUNT_ID" echo "Destination Account: $DEST_ACC OUNT_ID" echo "Source Region: $SOURCE_REGION" echo "Destination Region: $DEST_REG ION" echo "Lambda Image: $LAMBDA_IMAGE_URI" echo "Notification Email: $NOTIFICA TION_EMAIL" echo "Repositories: $REPOSITORY_LIST"	
驗證來源範本。	驗證來源 CloudFormation 範本： aws cloudformation validate-template \ --template-body file://"Source Account Cf template.yml" \ --profile source-ac count	資料工程師、ML 工程師、DevOps 工程師

任務	Description	所需的技能

部署來源基礎設施。	1. 部署來源帳戶堆疊：
	
	``
	aws cloudformation
	 deploy \
	  --template-file
	 "Source Account Cf
	 template.yml" \
	  --stack-name ecr-
	copy-source \
	  --parameter-overri
	des \
	    SourceAccountId=
	$SOURCE_ACCOUNT_ID \
	    Destinati
	onAccountId=$DEST_
	ACCOUNT_ID \
	    Destinati
	onRegion=$DEST_REG
	ION \
	    SourceRegion=
	$SOURCE_REGION \
	    Notificat
	ionEmail=$NOTIFICA
	TION_EMAIL \
	    Repositor
	yList="$REPOSITORY
	_LIST" \
	    LambdaImageUri=
	$LAMBDA_IMAGE_URI \
	    Environment=
	$ENVIRONMENT \
	  --capabilities
	 CAPABILITY_NAMED_I
	AM \
	  --profile source-ac
	count \
	  --region $SOURCE_R
	EGION
	``
	
	2. 等待堆疊完成（最多可能需要 10 分鐘）：
		資料工程師、ML 工程師、DevOps 工程師

任務	Description	所需的技能
	`aws cloudformation wait stack-create-complete \ --stack-name ecr-copy-source \ --profile source-account \ --region $SOURCE_REGION`	

任務	Description	所需的技能
驗證部署並收集輸出。	1. 取得堆疊輸出： ``` aws cloudformation describe-stacks \ --stack-name ecr- copy-source \ --profile source-ac count \ --region $SOURCE_R EGION \ --query 'Stacks[0].Outputs' \ --output table ``` 2. 存放狀態機器和 SNS 主題的 Amazon Resource Name (ARNs)： ``` export STATE_MAC HINE_ARN=$(aws cloudformation describe-stacks \ --stack-name ecr- copy-source \ --profile source-ac count \ --region $SOURCE_R EGION \ --query 'Stacks[0].Outputs[?OutputK ey==`StateMachineA rn`].OutputValue' \ --output text) export SNS_TOPIC _ARN=$(aws cloudform ation describe- stacks \ --stack-name ecr- copy-source \ ```	DevOps 工程師、ML 工程師、資料工程師

任務	Description	所需的技能

	<pre> --profile source-account \
	  --region $SOURCE_REGION \
	  --query 'Stacks[0].Outputs[?OutputKey==`SNSTopicArn`].OutputValue' \
	  --output text)
	
	echo "State Machine ARN: $STATE_MACHINE_ARN"
	echo "SNS Topic ARN: $SNS_TOPIC_ARN"</pre>	

確認您的電子郵件訂閱。	1. 請檢查您的電子郵件以確認您的 SNS 訂閱。
	2. 選擇電子郵件中的確認連結。
	3. 驗證訂閱狀態。
	<pre>aws sns list-subscriptions-by-topic \
	  --topic-arn $SNS_TOPIC_ARN \
	  --profile source-account \
	  --region $SOURCE_REGION</pre>
		資料工程師、ML 工程師、DevOps 工程師

執行和監控複製程序

任務	Description	所需的技能
執行並監控複製程序。	1. 登入 AWS 管理主控台，然後開啟 Step Functions 主控台。 2. 找到狀態機器。 3. 選擇 Start execution (開始執行)。 完成後，結果會顯示在執行輸入和輸出索引標籤上。 4. （選用）如果您想要使用繼續執行 Step Functions AWS CLI，請遵循此史詩中的其餘步驟。	DevOps 工程師、ML 工程師、資料工程師

執行步驟函數。	1. 產生唯一名稱：
	`export EXECUTION_NAME="ecr-copy-$(date +%Y%m%d-%H%M%S)"`
	2. 執行步驟函數。
	`export EXECUTION_ARN=$(aws stepfunctions start-execution \ --state-machine-arn $STATE_MACHINE_ARN \ --name $EXECUTION_NAME \ --profile source-account \ --region $SOURCE_REGION \`
		DevOps 工程師、ML 工程師、資料工程師

任務	Description	所需的技能
	--query 'executionArn' \ --output text) echo "Execution started: $EXECUTION_ARN" echo "Execution Name: $EXECUTION_NAME"	

任務	Description	所需的技能
監控進度。	1. 檢查狀態： ``` aws stepfunctions describe-execution \ --execution-arn $EXECUTION_ARN \ --profile source-ac count \ --region $SOURCE_R EGION \ --query '{Status: status,StartDate:s tartDate,StopDate: stopDate}' \ --output table ``` 2. 取得歷史記錄： ``` aws stepfunctions get-execution-hist ory \ --execution-arn $EXECUTION_ARN \ --profile source-ac count \ --region $SOURCE_R EGION \ --query 'events[? type==`TaskStateEn tered` || type==`Ta skSucceeded` || type==`TaskFailed`].{Type:type,Times tamp:timestamp,Det ails:stateEnteredE ventDetails.name}' \ --output table ```	DevOps 工程師、ML 工程師、資料工程師

任務	Description	所需的技能

檢查結果。	等待程序完成（每 30 秒更新一次）：
	
	`while true; do`
	`  STATUS=$(aws stepfunctions describe-execution \`
	`  --execution-arn $EXECUTION_ARN \`
	`  --profile source-account \`
	`  --region $SOURCE_REGION \`
	`  --query 'status' \`
	`  --output text)`
	
	`  echo "Current status: $STATUS"`
	
	`  if [[ "$STATUS" == "SUCCEEDED" || "$STATUS" == "FAILED" || "$STATUS" == "TIMED_OUT" || "$STATUS" == "ABORTED" ]]; then`
	`  break`
	`  fi`
	
	`  sleep 30`
	`done`
	
	`echo "Final execution status: $STATUS"`
		DevOps 工程師、ML 工程師、資料工程師

任務	Description	所需的技能

驗證映像。	1. 列出目的地帳戶中的儲存庫：
	
	``
	aws ecr describe-repositories \
	  --profile destination-account \
	  --region $DEST_REGION \
	  --query 'repositories[].repositoryName' \
	  --output table
	``
	
	2. 檢查儲存庫映像：
	
	``
	for repo in $(echo $REPOSITORY_LIST | tr ',' ' '); do
	  echo "\nImages in repository: $repo"
	  aws ecr list-images \
	    --repository-name $repo \
	    --profile destination-account \
	    --region $DEST_REGION \
	    --query 'imageIds[].imageTag' \
	    --output table
	 2>/dev/null || echo "Repository $repo not found or no images"
	done
	``
		DevOps 工程師、資料工程師、ML 工程師

疑難排解

問題	解決方案

步驟函數無法執行。	1. 若要從歷史記錄擷取詳細的失敗事件，請執行下列 AWS CLI 命令：
	`if [[ "$STATUS" == "FAILED" ]]; then`
	`  echo "Getting failure details..."`
	`  aws stepfunctions get-execution-history \`
	`  --execution-arn $EXECUTION_ARN \`
	`  --profile source-account \`
	`  --region $SOURCE_REGION \`
	`  --query 'events[?type==`TaskFailed`]' \`
	`  --output json`
	`fi`
	2. 若要擷取失敗 Lambda 函數的日誌，請執行下列 AWS CLI 命令：
	`# Check Lambda function logs`
	`echo "\nLambda function logs:"`
	`aws logs describe-log-groups \`
	`  --log-group-name-prefix "/aws/lambda/ecr-copy-source" \`
	`  --profile source-account \`
	`  --region $SOURCE_REGION \`
	`  --query 'logGroups[].logGroupName' \`
	`  --output table`

相關資源

- Crane 文件
- 什麼是 Amazon Elastic Container Registry？
- 什麼是 AWS Lambda？

- 什麼是 Step Functions？

其他資訊

組態參數

參數	描述	範例
SourceAccountId	來源 AWS 帳戶 ID	11111111111
DestinationAccountId	目的地 AWS 帳戶 ID	22222222222
DestinationRegion	目標 AWS 區域	us-east-2
SourceRegion	來源 AWS 區域	us-east-1
NotificationEmail	通知的電子郵件	abc@xyz.com
RepositoryList	要複製的儲存庫	repo1,repo2,repo3
LambdaImageUri	Lambda 容器映像 URI	${ACCOUNT}.dkr.ecr.${REGION}.amazonaws.com/ecr-copy-lambda:latest

使用 Amazon EFS 在 EC2 執行個體上建立 Amazon ECS 任務定義並掛載檔案系統

Durga Prasad Cheepuri，Amazon Web Services

總結

此模式提供程式碼範例和步驟，以建立在 Amazon Web Services (AWS) Cloud 中 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上執行的 Amazon Elastic Container Service (Amazon ECS) 任務定義，同時使用 Amazon Elastic File System (Amazon EFS) 在這些 EC2 執行個體上掛載檔案系統。使用 Amazon EFS 的 Amazon ECS 任務會自動掛載您在任務定義中指定的檔案系統，並將這些檔案系統提供給 AWS 區域中所有可用區域中的任務容器。

為了滿足您的持久性儲存和共用儲存需求，您可以同時使用 Amazon ECS 和 Amazon EFS。例如，您可以使用 Amazon EFS，透過在不同可用區域中執行的作用中和待命 ECS 容器對來存放應用程式的持久性使用者資料和應用程式資料，以獲得高可用性。您也可以使用 Amazon EFS 來存放共用資料，以供 ECS 容器和分散式任務工作負載平行存取。

若要將 Amazon EFS 與 Amazon ECS 搭配使用，您可以將一或多個磁碟區定義新增至任務定義。磁碟區定義包含 Amazon EFS 檔案系統 ID、存取點 ID，以及傳輸中 AWS Identity and Access Management (IAM) 授權或 Transport Layer Security (TLS) 加密的組態。您可以使用任務定義中的容器定義來指定在容器執行時掛載的任務定義磁碟區。當使用 Amazon EFS 檔案系統的任務執行時，Amazon ECS 會確保檔案系統已掛載，並可供需要存取的容器使用。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 具有虛擬私有網路 (VPN) 端點或路由器的虛擬私有雲端 (VPC)
- （建議）Amazon ECS 容器代理程式 1.38.0 或更新版本，以與 Amazon EFS 存取點和 IAM 授權功能相容（如需詳細資訊，請參閱 AWS 部落格文章 New for Amazon EFS – IAM Authorization and Access Points。）

限制

- 1.35.0 之前的 Amazon ECS 容器代理程式版本不支援使用 EC2 啟動類型的任務使用 Amazon EFS 檔案系統。

Architecture

下圖顯示使用 Amazon ECS 在 ECS 容器中的 EC2 執行個體上建立任務定義和掛載 Amazon EFS 檔案系統的應用程式範例。

該圖顯示以下工作流程：

1. 建立 Amazon EFS 檔案系統。
2. 使用容器建立任務定義。
3. 設定容器執行個體以掛載 Amazon EFS 檔案系統。任務定義參考磁碟區掛載，因此容器執行個體可以使用 Amazon EFS 檔案系統。ECS 任務可存取相同的 Amazon EFS 檔案系統，無論這些任務是在哪個容器執行個體上建立。
4. 使用任務定義的三個執行個體建立 Amazon ECS 服務。

技術堆疊

- Amazon EC2
- Amazon ECS
- Amazon EFS

工具

- Amazon EC2 – Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以使用 Amazon EC2 視需要啟動任意數量或任意數量的虛擬伺服器，也可以向外擴展或向內擴展。
- Amazon ECS – Amazon Elastic Container Service (Amazon ECS) 是一種高度可擴展的快速容器管理服務，用於執行、停止和管理叢集上的容器。您可以在 AWS Fargate 管理的無伺服器基礎設施上執行任務和服務。或者，若要進一步控制您的基礎設施，您可以在您管理的 EC2 執行個體叢集上執行任務和服務。
- Amazon EFS – Amazon Elastic File System (Amazon EFS) 提供簡單、可擴展、全受管的彈性 NFS 檔案系統，可與 AWS 雲端服務和內部部署資源搭配使用。
- AWS CLI – AWS 命令列界面 (AWS CLI) 是一種開放原始碼工具，可透過命令列 shell 中的命令與 AWS 服務互動。透過最少的組態，您可以從命令提示中執行 AWS CLI 命令，該命令會實作相當於瀏覽器型 AWS 管理主控台所提供功能的功能。

史詩

建立 Amazon EFS 檔案系統

任務	Description	所需的技能
使用 AWS 管理主控台建立 Amazon EFS 檔案系統。	1. **Note** 建立 Amazon EFS 檔案系統，然後選擇包含您容器的 VPC。：如果您使用不同的 VPC，請設定 VPC 對等互連。 2. 請注意檔案系統 ID。	AWS DevOps

使用 Amazon EFS 檔案系統或 AWS CLI 建立 Amazon ECS 任務定義

任務	Description	所需的技能
使用 Amazon EFS 檔案系統建立任務定義。	使用以下組態，使用新的 Amazon ECS 主控台或傳統 Amazon ECS 主控台建立任務定義： • 如果您使用新的主控台，請選擇應用程式環境的 Amazon EC2 執行個體。如果您使用傳統主控台，請選擇 EC2 作為啟動類型。 • 新增磁碟區。輸入磁碟區的名稱，選擇磁碟區類型的 EFS，然後選擇您先前記下的檔案系統 ID。針對根目錄，選擇您要在 Amazon	AWS DevOps

任務	Description	所需的技能
	ECS 容器主機上託管的 Amazon EFS 檔案系統路徑。	

任務	Description	所需的技能

使用 AWS CLI 建立任務定義。	1. 若要為任務定義建立具有輸入參數預留位置的 JSON 範本，請執行下列命令：
	`aws ecs register-task-definition --generate-cli-skeleton`
	2. 若要使用 JSON 範本建立任務定義，請執行下列命令：
	`aws ecs register-task-definition --cli-input-json file://<path_to_your_json_file>`
	3. **Note**
	根據 `task_definition_parameters.json` 檔案（已連接），在 JSON 範本中輸入輸入參數。：如需輸入參數的詳細資訊，請參閱任務定義參數 (Amazon ECS 文件）和 register-task-definition (AWS CLI 命令參考）。
		AWS DevOps

相關資源

- Amazon ECS 任務定義
- Amazon EFS 磁碟區

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用容器映像部署 Lambda 函數

Ram Kandaswamy，Amazon Web Services

總結

AWS Lambda 支援容器映像做為部署模型。此模式說明如何透過容器映像部署 Lambda 函數。

Lambda 是一種無伺服器、事件驅動的運算服務，您可以用來為幾乎任何類型的應用程式或後端服務執行程式碼，而無需佈建或管理伺服器。透過 Lambda 函數的容器映像支援，您可以為應用程式成品獲得高達 10 GB 的儲存空間，以及使用熟悉的容器映像開發工具的能力。

此模式中的範例使用 Python 做為基礎程式設計語言，但您可以使用其他語言，例如 Java、Node.js 或 Go。對於來源，請考慮 Git 型系統，例如 GitHub、GitLab 或 Bitbucket，或使用 Amazon Simple Storage Service (Amazon S3)。

先決條件和限制

先決條件

- 已啟用 Amazon Elastic Container Registry (Amazon ECR)
- 應用程式碼
- 具有執行時間介面用戶端和最新版本 Python 的 Docker 映像
- Git 的工作知識

限制

- 支援的影像大小上限為 10 GB。
- Lambda 型容器部署的執行時間上限為 15 分鐘。

Architecture

目標架構

1. 您可以建立 Git 儲存庫，並將應用程式程式碼遞交至儲存庫。
2. AWS CodeBuild 專案由遞交變更觸發。

3. CodeBuild 專案會建立 Docker 映像，並將建置的映像發佈至 Amazon ECR。
4. 您可以使用 Amazon ECR 中的映像建立 Lambda 函數。

自動化和擴展

此模式可以透過使用 SDK 中的 AWS CloudFormation AWS Cloud Development Kit (AWS CDK)或 API 操作來自動化。Lambda 可以根據請求數量自動擴展，您可以使用並行參數進行調整。如需詳細資訊，請參閱 Lambda 文件。

工具

AWS 服務

- AWS CloudFormation AWS CloudFormationhelps設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和 生命週期進行管理 AWS 區域。此模式使用 AWS CloudFormation Application Composer，可協助您以視覺化方式檢視和編輯 CloudFormation 範本。
- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。

其他工具

- Docker 是一組平台即服務 (PaaS) 產品，可在作業系統層級使用虛擬化在容器中交付軟體。
- GitHub、GitLab 和 Bitbucket 是一些常用的 Git 型來源控制系統，用於追蹤來源碼變更。

最佳實務

- 讓您的函數盡可能有效率且小，以避免載入不必要的檔案。
- 努力在 Docker 檔案清單中提高靜態層，並放置更頻繁地降低變更的層。這可改善快取，進而改善效能。
- 映像擁有者負責更新和修補映像。將更新節奏新增至您的操作程序。如需詳細資訊，請參閱 AWS Lambda 文件。

史詩

在 CodeBuild 中建立專案

任務	Description	所需的技能
建立 Git 儲存庫。	建立包含應用程式原始碼、Dockerfile 和 `buildspec.yaml` 檔案的 Git 儲存庫。	開發人員
建立 CodeBuild 專案。	若要使用 CodeBuild 專案建立自訂 Lambda 映像，請執行下列動作： 1. 登入 AWS 管理主控台，並在 https：//https://console.aws.amazon.com/codesuite/codebuild/ 開啟 CodeBuild 主控台。 2. 建立新專案。針對來源，選擇您建立的 Git 儲存庫。如需有關不同類型的 Git 儲存庫整合的資訊，請參閱使用連線文件。 3. 確認已啟用特權模式。若要建置 Docker 映像，這是必要的。否則，映像將無法成功建置。 4. 提供專案名稱和描述的值。	開發人員
編輯 Dockerfile。	Dockerfile 應位於您要開發應用程式的頂層目錄中。Python 程式碼應該位於 `src` 資料夾中。 當您建立映像時，請使用官方的 Lambda 支援映像。否則會	開發人員

任務	Description	所需的技能
	發生引導錯誤，使封裝程序更困難。 如需詳細資訊，請參閱其他資訊一節。	
在 Amazon ECR 中建立儲存庫。	在 Amazon ECR 中建立容器儲存庫。在下列範例命令中，建立的儲存庫名稱為 `cf-demo`： `aws ecr create-repository --cf-demo` 儲存庫將在 `buildspec.yaml` 檔案中參考。	AWS 管理員、開發人員
將映像推送至 Amazon ECR。	您可以使用 CodeBuild 來執行映像建置程序。CodeBuild 需要與 Amazon ECR 互動和使用 S3 的許可。在此程序中，會建置 Docker 映像並推送至 Amazon ECR 登錄檔。如需範本和程式碼的詳細資訊，請參閱其他資訊一節。	開發人員
確認映像位於儲存庫中。	若要驗證映像是否在儲存庫中，請在 Amazon ECR 主控台上選擇儲存庫。如果已在 Amazon ECR 設定中開啟該功能，則應使用標籤和漏洞掃描報告的結果列出映像。 如需詳細資訊，請參閱 AWS 文件。	開發人員

建立 Lambda 函數以執行映像

任務	Description	所需的技能
建立 Lambda 函數。	在 Lambda 主控台上，選擇建立函數，然後選擇容器映像。輸入 Amazon ECR 儲存庫中映像的函數名稱和 URI，然後選擇建立函數。如需詳細資訊，請參閱 AWS Lambda 文件。	應用程式開發人員
測試 Lambda 函數。	若要叫用和測試函數，請選擇測試。如需詳細資訊，請參閱 AWS Lambda 文件。	應用程式開發人員

疑難排解

問題	解決方案
組建未成功。	1. 檢查 CodeBuild 專案的特權模式是否已開啟。 2. 確保 Docker 相關命令具有必要的許可。嘗試將 sudo 新增至命令。 3. 確認與 CodeBuild 相關聯的 IAM 角色具有與 Amazon ECR、Amazon S3 和 CloudWatch 日誌互動的適當動作政策。

相關資源

- Lambda 的基礎映像
- CodeBuild 的 Docker 範例
- 傳遞臨時登入資料

其他資訊

編輯 Dockerfile

下列程式碼顯示您在 Dockerfile 中編輯的命令：

```
FROM public.ecr.aws/lambda/python:3.xx

# Copy function code
COPY app.py ${LAMBDA_TASK_ROOT}
COPY requirements.txt  ${LAMBDA_TASK_ROOT}

# install dependencies
RUN pip3 install --user -r requirements.txt

# Set the CMD to your handler (could also be done as a parameter override outside of
 the Dockerfile)
CMD [ "app.lambda_handler" ]
```

在 FROM命令中，針對 Lambda 支援的 Python 版本使用適當的值 （ 例如，3.12)。這將是公有 Amazon ECR 映像儲存庫中可用的基礎映像。

COPY app.py ${LAMBDA_TASK_ROOT} 命令會將程式碼複製到 Lambda 函數將使用的任務根目錄。此命令使用 環境變數，因此我們不必擔心實際路徑。要執行的函數會以引數形式傳遞至 CMD ["app.lambda_handler"]命令。

COPY requirements.txt 命令會擷取程式碼所需的相依性。

RUN pip install --user -r requirements.txt 命令會將相依性安裝到本機使用者目錄。

若要建置映像，請執行下列命令。

```
docker build -t <image name> .
```

在 Amazon ECR 中新增映像

在下列程式碼中，將 取代aws_account_id為帳號，us-east-1如果您使用的是不同的區域，請取代 。buildspec 檔案使用 CodeBuild 組建編號，將影像版本唯一識別為標籤值。您可以變更此項目以符合您的需求。

buildspec 自訂程式碼

```
phases:
  install:
    runtime-versions:
       python: 3.xx
  pre_build:
    commands:
      - python3 --version
      - pip3 install --upgrade pip
      - pip3 install --upgrade awscli
      - sudo docker info
  build:
    commands:
      - echo Build started on `date`
      - echo Building the Docker image...
      - ls
      - cd app
      - docker build -t cf-demo:$CODEBUILD_BUILD_NUMBER .
      - docker container ls
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker image...
      - aws ecr get-login-password --region us-east-1 | docker login --username AWS --
password-stdin aws_account_id.dkr.ecr.us-east-1.amazonaws.com
      - docker tag cf-demo:$CODEBUILD_BUILD_NUMBER aws_account_id.dkr.ecr.us-
east-1.amazonaws.com/cf-demo:$CODEBUILD_BUILD_NUMBER
      - docker push aws_account_id.dkr.ecr.us-east-1.amazonaws.com/cf-demo:
$CODEBUILD_BUILD_NUMBER
```

使用 AWS Fargate 在 Amazon ECS 上部署 Java 微服務

Vijay Thompson 和 Sandeep Bondugula，Amazon Web Services

總結

此模式提供使用 AWS Fargate 在 Amazon Elastic Container Service (Amazon ECS) 上部署容器化 Java 微服務的指引。模式不會使用 Amazon Elastic Container Registry (Amazon ECR) 進行容器管理；而是從 Docker 中樞提取 Docker 映像。

先決條件和限制

先決條件

- Docker 中樞上的現有 Java 微服務應用程式
- 公有 Docker 儲存庫
- 作用中的 AWS 帳戶
- 熟悉 AWS 服務，包括 Amazon ECS 和 Fargate
- Docker、Java 和 Spring Boot 架構
- Amazon Relational Database Service (Amazon RDS) 啟動並執行 （選用）
- 如果應用程式需要 Amazon RDS，則為虛擬私有雲端 (VPC) （選用）

Architecture

來源技術堆疊

- Java 微服務 （例如，在 Spring Boot 中實作） 並部署在 Docker

來源架構

目標技術堆疊

- 使用 Fargate 託管每個微服務的 Amazon ECS 叢集
- 託管 Amazon ECS 叢集和相關聯安全群組的 VPC 網路
- 每個微服務使用 Fargate 啟動容器的叢集/任務定義

目標架構

工具

工具

- Amazon ECS 不需要安裝和操作您自己的容器協同運作軟體、管理和擴展虛擬機器叢集，或在這些虛擬機器上排程容器。
- AWS Fargate 可協助您執行容器，而不需要管理伺服器或 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。它與 Amazon Elastic Container Service (Amazon ECS) 搭配使用。
- Docker 是一種軟體平台，可讓您快速建置、測試和部署應用程式。Docker 會將軟體封裝至稱為容器的標準化單位，其中包含軟體執行所需的一切，包括程式庫、系統工具、程式碼和執行時間。

Docker 程式碼

下列 Dockerfile 指定使用的 Java 開發套件 (JDK) 版本，其中有 Java 封存 (JAR) 檔案、公開的連接埠號碼，以及應用程式的進入點。

```
FROM openjdk:11
ADD target/Spring-docker.jar Spring-docker.jar
EXPOSE 8080
ENTRYPOINT ["java","-jar","Spring-docker.jar"]
```

史詩

建立新的任務定義

任務	Description	所需的技能
建立任務定義。	在 Amazon ECS 中執行 Docker 容器需要任務定義。在 https://console.aws.amazon.com/ecs/：// 開啟 Amazon ECS 主控台，選擇任務定義，然後建立新的任務定義。如需詳細資訊，請參閱 Amazon ECS 文件。	AWS 系統管理員、應用程式開發人員

任務	Description	所需的技能
選擇啟動類型。	選擇 Fargate 作為啟動類型。	AWS 系統管理員、應用程式開發人員
設定任務。	定義任務名稱，並使用適當數量的任務記憶體和 CPU 設定應用程式。	AWS 系統管理員、應用程式開發人員
定義容器。	指定容器名稱。針對映像，輸入 Docker 網站名稱、儲存庫名稱，以及 Docker 映像的標籤名稱 (`docker.io/sample-repo/sample-application:sample-tag-name`)。設定應用程式的記憶體限制，並為允許的連接埠設定連接埠映射 (`8080, 80`)。	AWS 系統管理員、應用程式開發人員
建立任務。	當任務和容器組態就緒時，請建立任務。如需詳細說明，請參閱相關資源區段中的連結。	AWS 系統管理員、應用程式開發人員

設定叢集

任務	Description	所需的技能
建立和設定叢集。	選擇僅聯網做為叢集類型、設定名稱，然後建立叢集，或在可用時使用現有叢集。如需詳細資訊，請參閱 Amazon ECS 文件。	AWS 系統管理員、應用程式開發人員

設定任務

任務	Description	所需的技能
建立 任務。	在叢集中，選擇執行新任務。	AWS 系統管理員、應用程式開發人員
選擇啟動類型。	選擇 Fargate 作為啟動類型。	AWS 系統管理員、應用程式開發人員
選擇任務定義、修訂和平台版本。	選擇您要執行的任務、任務定義的修訂，以及平台版本。	AWS 系統管理員、應用程式開發人員
選取 叢集。	選擇您要從中執行任務的叢集。	AWS 系統管理員、應用程式開發人員
指定任務數量。	設定應執行的任務數量。如果您使用兩個或多個任務啟動，則需要負載平衡器，才能在任務之間分配流量。	AWS 系統管理員、應用程式開發人員
指定任務群組。	（選用）指定任務群組名稱，以將一組相關任務識別為任務群組。	AWS 系統管理員、應用程式開發人員
設定叢集 VPC、子網路和安全群組。	設定叢集 VPC 和您要部署應用程式的子網路。建立或更新安全群組 (HTTP、HTTPS 和連接埠 8080)，以提供傳入和傳出連線的存取權。	AWS 系統管理員、應用程式開發人員
設定公有 IP 設定。	啟用或停用公有 IP，取決於您是否要為 Fargate 任務使用公有 IP 地址。預設的建議選項為已啟用。	AWS 系統管理員、應用程式開發人員
檢閱設定並建立任務	檢閱您的設定，然後選擇執行任務。	AWS 系統管理員、應用程式開發人員

剪下

任務	Description	所需的技能
複製應用程式 URL。	當任務狀態更新為執行中時，選取任務。在聯網區段中，複製公有 IP。	AWS 系統管理員、應用程式開發人員
測試您的應用程式。	在瀏覽器中，輸入公有 IP 以測試應用程式。	AWS 系統管理員、應用程式開發人員

相關資源

- Amazon ECS 的 Docker 基本概念 (Amazon ECS 文件）
- AWS Fargate 上的 Amazon ECS (Amazon ECS 文件）
- 建立任務定義 (Amazon ECS 文件）
- 建立叢集 (Amazon ECS 文件）
- 設定基本服務參數 (Amazon ECS 文件）
- 設定網路 (Amazon ECS 文件）
- 在 Amazon ECS 上部署 Java Microservices （部落格文章）

在 Amazon S3 中使用 Amazon EKS 和 Helm Chart 儲存庫部署 Kubernetes 資源和套件

Sagar Panigrahi，Amazon Web Services

總結

此模式可協助您有效率地管理 Kubernetes 應用程式，無論其複雜性為何。模式會將 Helm 整合到現有的持續整合和持續交付 (CI/CD) 管道中，以將應用程式部署到 Kubernetes 叢集。Helm 是 Kubernetes 套件管理員，可協助您管理 Kubernetes 應用程式。Helm Chart 有助於定義、安裝和升級複雜的 Kubernetes 應用程式。圖表可以版本化並存放在 Helm 儲存庫中，這可改善中斷期間的平均還原時間 (MTTR)。

此模式針對 Kubernetes 叢集使用 Amazon Elastic Kubernetes Service (Amazon EKS)。它使用 Amazon Simple Storage Service (Amazon S3) 做為 Helm Chart 儲存庫，以便整個組織的開發人員集中管理和存取圖表。

先決條件和限制

先決條件

- 具有虛擬私有雲端 (VPC) 的作用中 Amazon Web Services (AWS) 帳戶
- Amazon EKS 叢集
- 在 Amazon EKS 叢集中設定並準備好接受工作負載的工作者節點
- Kubectl 用於設定用戶端機器中目標叢集的 Amazon EKS kubeconfig 檔案
- 建立 S3 儲存貯體的 AWS Identity and Access Management (IAM) 存取權
- 從用戶端機器存取 Amazon S3 的 IAM （程式設計或角色）
- 原始程式碼管理和 CI/CD 管道

限制

- 目前不支援升級、刪除或管理自訂資源定義 (CRDs)。
- 如果您使用的是參考 CRD 的資源，則必須單獨安裝 CRD （在圖表之外）。

產品版本

- Helm 3.6.3 版

Architecture

目標技術堆疊

- Amazon EKS
- Amazon VPC
- Amazon S3
- 原始程式碼管理
- Helm
- kubectl

目標架構

自動化和擴展

- AWS CloudFormation 可用來自動建立基礎設施。如需詳細資訊，請參閱 Amazon EKS 文件中的使用 AWS CloudFormation 建立 Amazon EKS 資源。
- Helm 要整合到您現有的 CI/CD 自動化工具中，以自動化 Helm Chart 的封裝和版本控制（此模式的範圍外）。
- GitVersion 或 Jenkins 建置號碼可用來自動化圖表的版本控制。

工具

工具

- Amazon EKS – Amazon Elastic Kubernetes Service (Amazon EKS) 是一種受管服務，可在 AWS 上執行 Kubernetes，而不需要站立或維護您自己的 Kubernetes 控制平面。Kubernetes 是一套開放原始碼系統，用於容器化應用程式的自動化部署、擴展與管理。
- Helm – Helm 是 Kubernetes 套件管理員，可協助您在 Kubernetes 叢集上安裝和管理應用程式。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是網際網路的儲存體。您可以使用 Amazon S3 隨時從 Web 任何地方存放和擷取任意資料量。
- Kubectl – Kubectl 是一種命令列公用程式，用於對 Kubernetes 叢集執行命令。

Code

已連接範例程式碼。

史詩

設定和初始化 Helm

任務	Description	所需的技能
安裝 Helm 用戶端。	若要在您的本機系統上下載並安裝 Helm 用戶端，請使用下列命令。 `sudo curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 | bash`	DevOps 工程師
驗證 Helm 安裝。	若要驗證 Helm 是否能夠與 Amazon EKS 叢集中的 Kubernetes API 伺服器通訊，請執行 `helm version`。	DevOps 工程師

在 Amazon EKS 叢集中建立並安裝 Helm Chart

任務	Description	所需的技能
建立 NGINX 的 Helm Chart。	若要在用戶端機器`my-nginx`上建立名為 的 Helm Chart，請執行 `helm create my-nginx`。	DevOps 工程師
檢閱圖表的結構。	若要檢閱圖表的結構，請執行樹狀命令 `tree my-nginx/`。	DevOps 工程師
在圖表中停用服務帳戶建立。	在 的 `serviceAccount` 區段`values.yaml` 下，	DevOps 工程師

任務	Description	所需的技能
	將`create`金鑰設定為`false`。這是關閉的，因為不需要為此模式建立服務帳戶。	
驗證 (lint) 修改後的圖表是否有語法錯誤。	若要驗證圖表是否有任何語法錯誤，再將其安裝在目標叢集中，請執行 `helm lint my-nginx/` 。	DevOps 工程師
安裝圖表以部署 Kubernetes 資源。	若要執行 Helm Chart 安裝，請使用下列命令。 `helm install --name my-nginx-release --debug my-nginx/ --namespace helm-space` 選用旗標會在安裝期間`debug`輸出所有偵錯訊息。`namespace` 旗標指定要在其中建立此圖表資源部分的命名空間。	DevOps 工程師
檢閱 Amazon EKS 叢集中的資源。	若要檢閱在`helm-space` 命名空間中作為 Helm Chart 一部分建立的資源，請使用下列命令。 `kubectl get all -n helm-space`	DevOps 工程師

回復至舊版的 Kubernetes 應用程式

任務	Description	所需的技能
修改和升級版本。	若要修改圖表，請在 中將`replicaCount` 值`values.yaml` 變更為 2。然後執行下列命令來升級已安裝的版本。 `helm upgrade my-nginx-release my-nginx/ --namespace helm-space`	DevOps 工程師
檢閱 Helm 版本的歷史記錄。	若要列出使用 Helm 安裝的特定版本的所有修訂，請執行下列命令。 `helm history my-nginx-release`	DevOps 工程師
檢閱特定修訂的詳細資訊。	在切換或轉返到工作版本之前，以及安裝修訂之前的額外驗證層，請使用下列命令檢視傳遞到每個修訂的值。 `helm get --revision=2 my-nginx-release`	DevOps 工程師
回復至先前的版本。	若要復原至先前的修訂版，請使用下列命令。 `helm rollback my-nginx-release 1` 此範例會轉返至修訂版編號 1。	DevOps 工程師

將 S3 儲存貯體初始化為 Helm 儲存庫

任務	Description	所需的技能
為 Helm Chart 建立 S3 儲存貯體。	建立唯一的 S3 儲存貯體。在儲存貯體中，建立名為 的資料夾`charts`。此模式中的範例使用 `s3://my-helm-charts/charts` 做為目標圖表儲存庫。	雲端管理員
安裝適用於 Amazon S3 的 Helm 外掛程式。	若要在用戶端機器上安裝 helm-s3 外掛程式，請使用下列命令。 `helm plugin install https://github.com/hypnoglow/helm-s3.git --version 0.10.0` 注意：Helm V3 支援適用於外掛程式 0.9.0 版及更新版本。	DevOps 工程師
初始化 Amazon S3 Helm 儲存庫。	若要將目標資料夾初始化為 Helm 儲存庫，請使用下列命令。 `helm S3 init s3://my-helm-charts/charts` 命令會在目標中建立 `index.yaml` 檔案，以追蹤存放在該位置的所有圖表資訊。	DevOps 工程師
將 Amazon S3 儲存庫新增至 Helm。	若要在用戶端機器中新增儲存庫，請使用下列命令。	DevOps 工程師

任務	Description	所需的技能
	`helm repo add my-helm-charts s3://my-helm-charts/charts` 此命令會將別名新增至 Helm 用戶端機器中的目標儲存庫。	
檢閱儲存庫清單。	若要檢視 Helm 用戶端機器中的儲存庫清單，請執行 `helm repo list`。	DevOps 工程師

在 Amazon S3 Helm 儲存庫中封裝和存放圖表

任務	Description	所需的技能
封裝圖表。	若要封裝您建立的`my-nginx`圖表，請執行 `helm package ./my-nginx/` 。命令會將`my-nginx`圖表資料夾的所有內容封裝至封存檔案，該檔案會使用`Chart.yaml` 檔案中提及的版本編號來命名。	DevOps 工程師
將套件存放在 Amazon S3 Helm 儲存庫中。	若要將套件上傳至 Amazon S3 中的 Helm 儲存庫，請使用 `.tgz` 檔案的正確名稱執行下列命令。 `helm s3 push ./my-nginx-0.1.0.tgz my-helm-charts`	DevOps 工程師
搜尋 Helm Chart。	若要確認圖表在本機和 Amazon S3 中的 Helm 儲存	DevOps 工程師

任務	Description	所需的技能
	庫中同時出現，請執行下列命令。 `helm search repo my-nginx`	

修改、版本和封裝圖表

任務	Description	所需的技能
修改和封裝圖表。	在 中`values.yaml` ，將 `replicaCount` 值設定為 1。然後執行 封裝圖表`helm package ./my-nginx/` ，這次將 中的版本變更為 `Chart.yaml` `0.1.1`。 在 CI/CD 管道中使用 GitVersion 或 Jenkins 建置號碼等工具，最好透過自動化更新版本控制。自動化版本編號超出此模式的範圍。	DevOps 工程師
將新版本推送至 Amazon S3 中的 Helm 儲存庫。	若要將 0.1.1 版的新套件推送至 Amazon S3 中的 `my-helm-charts` Helm 儲存庫，請執行下列命令。 `helm s3 push ./my-nginx-0.1.1.tgz my-helm-charts`	DevOps 工程師

從 Amazon S3 Helm 儲存庫搜尋並安裝圖表

任務	Description	所需的技能
搜尋 my-nginx 圖表的所有版本。	若要檢視圖表的所有可用版本，請使用 `--versions` 旗標執行下列命令。 `helm search repo my-nginx --versions` 如果沒有 旗標，Helm 預設會顯示圖表的最新上傳版本。	DevOps 工程師
從 Amazon S3 Helm 儲存庫安裝圖表。	先前任務的搜尋結果會顯示`my-nginx`圖表的多個版本。若要從 Amazon S3 Helm 儲存庫安裝新版本 (0.1.1)，請使用下列命令。 `helm upgrade my-nginx-release my-helm-charts/my-nginx --version 0.1.1 --namespace helm-space`	DevOps 工程師

相關資源

- HELM 文件
- helm-s3 外掛程式 (MIT 授權)
- HELM 用戶端二進位
- Amazon EKS 文件

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Terraform 在 Amazon EKS 中部署 CockroachDB 叢集

Sandip Gangapadhyay 和 Kalyan Senthilnathan，Amazon Web Services

總結

此模式提供 HashiCorp Terraform 模組，用於使用 CockroachDB 運算子在 Amazon Elastic Kubernetes Service (Amazon EKS) 上部署多節點 CockroachDB 叢集。CockroachDB 是分散式 SQL 資料庫，可在地理分佈叢集中提供自動水平分片、高可用性和一致的效能。此模式使用 Amazon EKS 作為受管 Kubernetes 平台，並實作 cert-manager 進行 TLS 安全節點通訊。它還使用 Network Load Balancer 進行流量分佈，並使用 Pod 建立 CockroachDB StatefulSets，以自動複寫資料以實現容錯能力和效能。

目標對象

若要實作此模式，建議您熟悉下列項目：

- HashiCorp Terraform 概念和基礎設施即程式碼 (IaC) 實務
- AWS 服務，特別是 Amazon EKS
- Kubernetes 基本概念，包括 StatefulSets、運算子和服務組態
- 分散式 SQL 資料庫
- 安全概念，例如 TLS 憑證管理。
- DevOps 實務、CI/CD 工作流程和基礎設施自動化

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 在 Amazon EKS 叢集中部署資源的許可
- 標記節點的 Amazon EKS 叢集版本 1.23 或更新版本 `node=cockroachdb`
- Amazon Elastic Block Store 容器儲存界面 (CSI) 驅動程式 1.19.0 版或更新版本，安裝在 Amazon EKS 叢集中
- Terraform CLI 1.0.0 版或更新版本，已安裝
- kubectl，已安裝
- Git，已安裝

- AWS Command Line Interface (AWS CLI) 2.9.18 版或更新版本，已安裝並設定

限制

- CockroachDB Kubernetes Operator 不支援多區域部署的多個 Kubernetes 叢集。如需更多限制，請參閱跨多個 Kubernetes 叢集協調 CockroachDB (CockroachDB 文件） 和 CockroachDB Kubernetes Operator (GitHub)。
- 持久性磁碟區宣告 (PVCs的自動剔除目前預設為停用。這表示在解除委任並移除節點之後，運算子不會移除掛載至其 Pod 的持久性磁碟區。如需詳細資訊，請參閱 CockroachDB 文件中的自動 PVC 剔除。

產品版本

- CockroachDB 22.2.2 版

Architecture

目標架構

下圖顯示虛擬私有雲端 (VPC) 中三個 AWS 可用區域的高可用性 CockroachDB 部署。CockroachDB Pod 是透過 Amazon EKS 管理。架構說明使用者如何透過 Network Load Balancer 存取資料庫，將流量分配到 CockroachDB Pod。在每個可用區域中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上執行的 Pod，可提供彈性和容錯能力。

已建立資源

部署此模式中使用的 Terraform 模組會建立下列資源：

1. Network Load Balancer – 此資源可做為用戶端請求的進入點，並將流量平均分配到 CockroachDB 執行個體。
2. CockroachDB StatefulSet – StatefulSet 定義 Amazon EKS 叢集中 CockroachDB 部署的所需狀態。它會管理 CockroachDB Pod 的排序部署、擴展和更新。
3. CockroachDB Pod – 這些 Pod 是做為 Kubernetes Pod 內容器執行的 CockroachDB 執行個體。這些 Pod 會跨分散式叢集存放和管理資料。
4. CockroachDB 資料庫 – 這是由 CockroachDB 管理的分散式資料庫，跨越多個 Pod。它會複寫資料以實現高可用性、容錯能力和效能。

工具

AWS 服務

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 上執行 Kubernetes， AWS 而無需安裝或維護您自己的 Kubernetes 控制平面或節點。

其他工具

- HashiCorp Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。
- kubectl 是一種命令列界面，可協助您針對 Kubernetes 叢集執行命令。

程式碼儲存庫

此模式的程式碼可在 GitHub 中使用 Terraform 儲存庫在 Amazon EKS 中部署 CockroachDB 叢集中取得。程式碼儲存庫包含下列適用於 Terraform 的檔案和資料夾：

- `modules` 資料夾 – 此資料夾包含 CockroachDB 的 Terraform 模組
- `main` 資料夾 – 此資料夾包含呼叫 CockroachDB 子模組的根模組，以建立 CockroachDB 資料庫叢集。

最佳實務

- 請勿縮減至少於三個節點。這被視為 CockroachDB 上的反模式，並可能導致錯誤。如需詳細資訊，請參閱 CockroachDB 文件中的叢集擴展。
- 使用 Karpernter 或 Cluster Autoscaler 實作 Amazon EKS 自動擴展。這可讓 CockroachDB 叢集自動水平擴展和自動擴展新節點。如需詳細資訊，請參閱 Amazon EKS 文件中的使用 Karpenter 和 Cluster Autoscaler 擴展叢集運算。

> **Note**
>
> 由於 `podAntiAffinity` Kubernetes 排程規則，在一個 Amazon EKS 節點中只能排程一個 CockroachDB Pod。

- 如需 Amazon EKS 安全最佳實務，請參閱 Amazon EKS 文件中的安全最佳實務。
- 如需 CockroachDB 的 SQL 效能最佳實務，請參閱 CockroachDB 文件中的 SQL 效能最佳實務。
- 如需為 Terraform 狀態檔案設定 Amazon Simple Storage Service (Amazon S3) 遠端後端的詳細資訊，請參閱 Terraform 文件中的 Amazon S3。

史詩

設定您的環境

任務	Description	所需的技能
複製程式碼儲存庫。	輸入下列命令來複製儲存庫： `git clone https://github.com/aws-samples/crdb-cluster-eks-terraform.git`	DevOps 工程師，Git
更新 Terraform 變數。	1. 輸入下列命令以導覽至複製儲存庫中的主資料夾： `cd crdb-cluster-eks-terraform/main` 2. 開啟 variable.tf 檔案。 3. 設定下列變數的預設值： • `region` – 輸入目標 AWS 區域 • `eks_cluster_name` – 輸入目標 Amazon EKS 叢集的名稱 • `number_of_nodes` – 輸入要部署的節點數量 4. 儲存並關閉 variable.tf 檔案。	DevOps 工程師，Terraform

部署 資源

任務	Description	所需的技能

部署 基礎設施。	1. 輸入下列命令來初始化 Terraform 部署：
	`terraform init`
	2. 輸入下列命令來產生執行計畫：
	`terraform plan`
	3. 檢閱計劃，並驗證要建立的資源和基礎設施元件。
	4. 輸入下列命令來部署基礎設施：
	`terraform apply`
	5. 出現提示時，請輸入 yes 以確認部署。
	6. 等待部署完成。
		DevOps 工程師，Terraform

驗證部署

任務	Description	所需的技能
驗證資源建立。	1. 輸入下列命令以使用 設定 Amazon EKS 內容 AWS CLI： `aws eks update-kubeconfig –name <eks_cluster_name>`	DevOps 工程師

任務	Description	所需的技能

	2. 輸入下列命令來驗證使用 CockroachDB 的 Pod 數量：
	`kubectl get pods -n <namespace>`
	下列為範例輸出。
	`NAME READY STATUS RESTARTS AGE`
	`cockroach-operator-655fbf7847-zn9v8 1/1 Running 0 30m`
	`cockroachdb-0 1/1 Running 0 24m`
	`cockroachdb-1 1/1 Running 0 24m`
	`cockroachdb-2 1/1 Running 0 24m`
	3. 確認 Pod 數量符合您在 variable.tf 檔案中定義的值。	

任務	Description	所需的技能
（選用）向上或向下擴展。	1. 在 variable.tf 檔案中，增加或減少節點數量，然後儲存檔案。 2. 重複這些步驟，透過 Terraform 部署基礎設施。Terraform 新增或移除 Pod。 3. 重複這些步驟來驗證使用 Cockroach 資料庫的 Pod 數量。例如，如果您將節點數量從三個增加到四個，您現在應該會看到四個 Pod 正在執行。	DevOps 工程師，Terraform

清除

任務	Description	所需的技能
刪除基礎設施。	將節點擴展至 0可降低運算成本。不過，對於此模組建立的持久性 Amazon EBS 磁碟區，您仍需支付費用。若要降低儲存成本，請依照下列步驟刪除所有磁碟區： 1. 輸入下列命令來刪除基礎設施： `terraform destroy` 2. 出現提示時，輸入 yes 進行確認。	Terraform

疑難排解

問題	解決方案
驗證供應商登入資料時發生錯誤	當您執行 Terraform `apply`或 `destroy`命令時，您可能會遇到下列錯誤： `Error: configuring Terraform AWS Provider: error validating provider credentials: error calling sts:GetCallerIdentity: operation error STS: GetCallerIdentity, https response error StatusCode: 403, RequestID: 123456a9-fbc1-40ed-b8d8-513d0133ba7f, api error InvalidClientTokenId: The security token included in the request is invalid.` 此錯誤是由本機電腦組態中使用的登入資料的安全字符過期所造成。如需如何解決錯誤的指示，請參閱 AWS CLI 文件中的設定和檢視組態設定。
處於待定狀態的 CockroachDB Pod	1. 由於 `podAntiAffinity` Kubernetes 排程規則，在一個 Amazon EKS 節點中只能排程一個 CockroachDB Pod。如果 CockroachDB Pod 的數量超過可用的 Amazon EKS 節點數量，則 CockroachDB Pod 可能處於待定狀態。在這種情況下，您需要實作 Cluster Autoscaler 或 Karpenter，以便theAmazon EKS 節點自動擴展。如需詳細資訊，請參閱使用 Karpenter 擴展叢集運算和 Cluster Autoscaler。 2. 輸入下列命令，檢查 Kubernetes 工作者節點是否有`node=cockroachdb` 標籤：

問題	解決方案
	`kubectl get nodes --show-labels` 如果沒有，您需要確保所有工作者節點都已正確標記。如需詳細資訊，請參閱 Amazon EKS 文件中的編輯節點群組組態。

相關資源

- 在單一 Kubernetes 叢集中部署 CockroachDB (CockroachDB 文件)
- 跨多個 Kubernetes 叢集協調 CockroachDB (CockroachDB 文件)
- AWS 提供者 (Terraform 文件)

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

在 Amazon EKS 上部署範例 Java 微服務，並使用 Application Load Balancer 公開微服務

Vijay Thompson 和 Akkamahadevi 租用數學，Amazon Web Services

總結

此模式說明如何使用`eksctl`命令列公用程式和 Amazon Elastic Container Registry (Amazon ECR)，將範例 Java 微服務部署為 Amazon Elastic Kubernetes Service (Amazon EKS) 上的容器化應用程式。您可以使用 Application Load Balancer 來負載平衡應用程式流量。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在 macOS、Linux 或 Windows 上安裝和設定 AWS Command Line Interface (AWS CLI) 1.7 版或更新版本
- 執行中的 Docker 協助程式
- 在 macOS、Linux 或 Windows 上安裝和設定的`eksctl`命令列公用程式（如需詳細資訊，請參閱 Amazon EKS 文件中的 Amazon EKS – eksctl 入門。）
- 在 macOS、Linux 或 Windows 上安裝和設定的`kubectl`命令列公用程式（如需詳細資訊，請參閱 Amazon EKS 文件中的安裝或更新 kubectl。）

限制

- 此模式不包含 Application Load Balancer 的 SSL 憑證安裝。

Architecture

目標技術堆疊

- Amazon ECR
- Amazon EKS
- Elastic Load Balancing

目標架構

下圖顯示用於在 Amazon EKS 上容器化 Java 微服務的架構。

工具

- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 AWS 上執行 Kubernetes，而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- Elastic Load Balancing 會自動將傳入流量分散到一或多個可用區域中的多個目標，例如 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、容器和 IP 地址。
- eksctl 可協助您在 Amazon EKS 上建立叢集。
- kubectl 可讓您針對 Kubernetes 叢集執行命令。
- Docker 可協助您在稱為容器的套件中建置、測試和交付應用程式。

史詩

使用 eksctl 建立 Amazon EKS 叢集

任務	Description	所需的技能
建立 Amazon EKS 叢集。	若要建立使用兩個 t2.small Amazon EC2 執行個體做為節點的 Amazon EKS 叢集，請執行下列命令： `eksctl create cluster --name <your-cluster-name> --version <version-number> --nodes=1 --node-type=t2.small`	開發人員、系統管理員

任務	Description	所需的技能
	Note 程序可能需要 15 到 20 分鐘。建立叢集之後，適當的 Kubernetes 組態會新增至您的 kubeconfig 檔案。您可以使用 kubeconfig 檔案搭配 kubectl，在後續步驟中部署應用程式。	
驗證 Amazon EKS 叢集。	若要確認叢集已建立且您可以連線到叢集，請執行 kubectl get nodes命令。	開發人員、系統管理員

建立 Amazon ECR 儲存庫並推送 Docker 映像。

任務	Description	所需的技能
建立 Amazon ECR 儲存庫。	遵循 Amazon ECR 文件中建立私有儲存庫的指示。	開發人員、系統管理員
建立 POM XML 檔案。	根據此模式額外資訊區段中pom.xml的範例 POM 檔案程式碼來建立檔案。	開發人員、系統管理員

建立來源檔案。	根據下列範例，在src/main/java/eksExample 路徑HelloWorld.java 中建立名為 的來源檔案：
	`package eksExample;`
	`import static spark.Spark.get;`	

任務	Description	所需的技能

	<pre>public class HelloWorld {
	    public static void main(String[] args) {
	        get("/", (req, res) -> {
	            return "Hello World!";
	        });
	    }
	}</pre>
	請務必使用下列目錄結構：
	<pre>### Dockerfile
	### deployment.yaml
	### ingress.yaml
	### pom.xml
	### service.yaml
	### src
	    ### main
	        ### java
	            ### eksExample
	                ### HelloWorld.java</pre>	

建立 Dockerfile。	`Dockerfile` 根據此模式額外資訊區段中的範例 Dockerfile 程式碼建立 。	開發人員、系統管理員

任務	Description	所需的技能

建置並推送 Docker 映像。	在您希望 `Dockerfile` 建置、標記映像並將映像推送至 Amazon ECR 的目錄中，執行下列命令：
	
	`aws ecr get-login -password --region <region>| docker login --username <username> --password-stdin <account_number>.dkr.ecr.<region>.amazonaws.com`
	`docker buildx build --platform linux/amd64 -t hello-world-java:v1 .`
	`docker tag hello-world-java:v1 <account_number>.dkr.ecr.<region>.amazonaws.com/<repository_name>:v1`
	`docker push <account_number>.dkr.ecr.<region>.amazonaws.com/<repository_name>:v1`
	
	Note
	修改上述命令中的 AWS 區域、帳戶號碼和儲存庫詳細資訊。請務必記下映像 URL 以供日後使用。	

任務	Description	所需的技能
	Important 具有 M1 晶片的 macOS 系統在建置與在 AMD64 平台上執行的 Amazon EKS 相容的映像時發生問題。若要解決此問題，請使用 Docker buildx 建置適用於 Amazon EKS 的 Docker 映像。	

部署 Java 微服務

任務	Description	所需的技能
建立部署檔案。	`deployment.yaml` 根據此模式額外資訊區段中的範例部署檔案程式碼，建立名為 的 YAML 檔案。 **Note** 使用您先前複製的映像 URL 做為 Amazon ECR 儲存庫映像檔案的路徑。	開發人員、系統管理員
在 Amazon EKS 叢集上部署 Java 微服務。	若要在 Amazon EKS 叢集中建立部署，請執行 `kubectl apply -f deployment.yaml` 命令。	開發人員、系統管理員

任務	Description	所需的技能
驗證 Pod 的狀態。	1. 若要驗證 Pod 的狀態，請執行 `kubectl get pods`命令。 2. 等待狀態變更為就緒。	開發人員、系統管理員
建立服務。	1. `service.yaml` 根據此模式額外資訊區段中的範例服務檔案程式碼，建立名為 的檔案。 2. 執行 `kubectl apply -f service.yaml` 命令。	開發人員、系統管理員
安裝 AWS Load Balancer 控制器附加元件。	遵循 Amazon EKS 文件中安裝 AWS Load Balancer 控制器附加元件的指示。 **Note** 您必須安裝 附加元件，才能為 Kubernetes 服務建立 Application Load Balancer 或 Network Load Balancer。	開發人員、系統管理員
建立輸入資源。	`ingress.yaml` 根據此模式額外資訊區段中的範例輸入資源檔案程式碼，建立名為 的 YAML 檔案。	開發人員、系統管理員
建立 Application Load Balancer。	若要部署輸入資源並建立 Application Load Balancer，請執行 `kubectl apply -f ingress.yaml` 命令。	開發人員、系統管理員

測試應用程式。

任務	Description	所需的技能
測試並驗證應用程式。	1. 若要從 ADDRESS 欄位取得負載平衡器的 DNS 名稱，請執行 `kubectl get ingress.networking.k8s.io/java-microservice-ingress` 命令。 2. 在與 Amazon EKS 節點位於相同 VPC 的 EC2 執行個體上，執行 `curl -v <DNS address from previous command>`命令。	開發人員、系統管理員

相關資源

- 建立私有儲存庫 (Amazon ECR 文件）
- 推送 Docker 映像 (Amazon ECR 文件）
- 輸入控制器 (Amazon EKS 研討會）
- Docker buildx (Docker 文件）

其他資訊

POM 檔案範例

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/
maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
```

```
  <groupId>helloWorld</groupId>
  <artifactId>helloWorld</artifactId>
  <version>1.0-SNAPSHOT</version>


  <dependencies>
    <dependency>
      <groupId>com.sparkjava</groupId><artifactId>spark-core</
artifactId><version>2.0.0</version>
    </dependency>
  </dependencies>
  <build>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId><artifactId>maven-jar-plugin</
artifactId><version>2.4</version>
        <configuration><finalName>eksExample</finalName><archive><manifest>
              <addClasspath>true</addClasspath><mainClass>eksExample.HelloWorld</
mainClass><classpathPrefix>dependency-jars/</classpathPrefix>
            </manifest></archive>
        </configuration>
      </plugin>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId><artifactId>maven-compiler-plugin</
artifactId><version>3.1</version>
        <configuration><source>1.8</source><target>1.8</target></configuration>
      </plugin>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId><artifactId>maven-assembly-plugin</
artifactId>
        <executions>
          <execution>
            <goals><goal>attached</goal></goals><phase>package</phase>
            <configuration>
              <finalName>eksExample</finalName>
              <descriptorRefs><descriptorRef>jar-with-dependencies</descriptorRef></
descriptorRefs>
              <archive><manifest><mainClass>eksExample.HelloWorld</mainClass></
manifest></archive>
            </configuration>
          </execution>
        </executions>
      </plugin>
    </plugins>
```

```
  </build>
</project>
```

範例 Dockerfile

```
FROM bellsoft/liberica-openjdk-alpine-musl:17

RUN apk add maven
WORKDIR /code

# Prepare by downloading dependencies
ADD pom.xml /code/pom.xml
RUN ["mvn", "dependency:resolve"]
RUN ["mvn", "verify"]

# Adding source, compile and package into a fat jar
ADD src /code/src
RUN ["mvn", "package"]

EXPOSE 4567
CMD ["java", "-jar", "target/eksExample-jar-with-dependencies.jar"]
```

部署檔案範例

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: microservice-deployment
spec:
  replicas: 2
  selector:
    matchLabels:
      app.kubernetes.io/name: java-microservice
  template:
    metadata:
      labels:
        app.kubernetes.io/name: java-microservice
    spec:
      containers:
      - name: java-microservice-container
        image: .dkr.ecr.amazonaws.com/:
        ports:
```

```
        - containerPort: 4567
```

服務檔案範例

```
apiVersion: v1
kind: Service
metadata:
  name: "service-java-microservice"
spec:
  ports:
    - port: 80
      targetPort: 4567
      protocol: TCP
  type: NodePort
  selector:
    app.kubernetes.io/name: java-microservice
```

範例輸入資源檔案

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: "java-microservice-ingress"
  annotations:
    kubernetes.io/ingress.class: alb
    alb.ingress.kubernetes.io/load-balancer-name: apg2
    alb.ingress.kubernetes.io/target-type: ip
  labels:
    app: java-microservice
spec:
  rules:
    - http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: "service-java-microservice"
                port:
                  number: 80
```

在 Amazon EKS 叢集上部署以 gRPC 為基礎的應用程式，並使用 Application Load Balancer 存取它

Kirankumar Chandrashekar 和 Huy Nguyen，Amazon Web Services

摘要

此模式說明如何在 Amazon Elastic Kubernetes Service (Amazon EKS) 叢集上託管以 gRPC 為基礎的應用程式，並透過 Application Load Balancer 安全地存取它。

gRPC 是一種可在任何環境中執行的開放原始碼遠端程序呼叫 (RPC) 架構。您可以使用它進行微服務整合和用戶端-伺服器通訊。如需 gRPC 的詳細資訊，請參閱 AWS 部落格文章 Application Load Balancer 支援end-to-end HTTP/2 和 gRPC。

此模式說明如何託管在 Amazon EKS 上的 Kubernetes Pod 上執行的 gRPC 型應用程式。gRPC 用戶端會透過具有 SSL/TLS 加密連線的 HTTP/2 通訊協定連線至 Application Load Balancer。Application Load Balancer 會將流量轉送至在 Amazon EKS Pod 上執行的 gRPC 應用程式。您可以使用 Kubernetes Horizontal Pod Autoscaler，根據流量自動擴展 gRPC Pod 的數量。Application Load Balancer 的目標群組會對 Amazon EKS 節點執行運作狀態檢查、評估目標是否正常運作，以及僅將流量轉送至運作狀態良好的節點。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在 Linux、macOS 或 Windows 上安裝和設定 Docker。
- 在 Linux、macOS 或 Windows 上安裝和設定 AWS Command Line Interface (AWS CLI) 第 2 版。macOS
- eksctl，在 Linux、macOS 或 Windows 上安裝和設定。
- `kubectl`，已安裝並設定為存取 Amazon EKS 叢集上的資源。如需詳細資訊，請參閱 Amazon EKS 文件中的安裝或更新 kubectl。
- gRPCurl，已安裝並設定。
- 新的或現有的 Amazon EKS 叢集。如需詳細資訊，請參閱 Amazon EKS 入門。
- 您的電腦終端機已設定為存取 Amazon EKS 叢集。如需詳細資訊，請參閱《Amazon EKS 文件》中的設定您的電腦與叢集通訊。
- AWS Load Balancer 控制器，在 Amazon EKS 叢集中佈建。

- 具有有效 SSL 或 SSL/TLS 憑證的現有 DNS 主機名稱。您可以使用 AWS Certificate Manager (ACM) 或將現有憑證上傳至 ACM，來取得網域的憑證。如需這兩個選項的詳細資訊，請參閱 ACM 文件中的請求公有憑證和將憑證匯入 AWS Certificate Manager。

Architecture

下圖顯示此模式實作的架構。

下圖顯示從卸載至 Application Load Balancer 的 gRPC 用戶端接收 SSL/TLS 流量的工作流程。流量會以純文字轉送至 gRPC 伺服器，因為它來自虛擬私有雲端 (VPC)。

工具

AWS 服務

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 Shell 中的命令與 AWS 服務互動。
- Elastic Load Balancing 會將傳入的應用程式或網路流量分散到多個目標。例如，您可以將流量分散到一或多個可用區域中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、容器和 IP 地址。
- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 AWS 上執行 Kubernetes，而無需安裝或維護您自己的 Kubernetes 控制平面或節點。

工具

- eksctl 是在 Amazon EKS 上建立叢集的簡單 CLI 工具。
- kubectl 是一種命令列公用程式，用於對 Kubernetes 叢集執行命令。
- AWS Load Balancer 控制器 可協助您 管理 Kubernetes 叢集的 AWS Elastic Load Balancer。
- gRPCurl 是一種命令列工具，可協助您與 gRPC 服務互動。

程式碼儲存庫

此模式的程式碼可在 GitHub grpc-traffic-on-alb-to-eks 儲存庫中使用。

史詩

建置 gRPC 伺服器的 Docker 映像並將其推送至 Amazon ECR

任務	Description	所需的技能
建立 Amazon ECR 儲存庫。	登入 AWS 管理主控台，開啟 Amazon ECR 主控台，然後建立 Amazon ECR 儲存庫。如需詳細資訊，請參閱 Amazon ECR 文件中的建立儲存庫。請務必記錄 Amazon ECR 儲存庫的 URL。 您也可以執行下列命令，使用 AWS CLI 建立 Amazon ECR 儲存庫： `aws ecr create-repository --repository-name helloworld-grpc`	雲端管理員
建置 Docker 影像。	1. 複製 GitHub grpc-traffic-on-alb-to-eks 儲存庫。 `git clone https://github.com/aws-samples/grpc-traffic-on-alb-to-eks.git` 2. 從儲存庫的根目錄中，確定 Dockerfile 存在，然後執行下列命令來建置 Docker 映像：	DevOps 工程師

任務	Description	所需的技能
	`docker build -t <amazon_ecr_repository_url>:<Tag> .` **Important** 請務必`<amazon_ecr_repository_url>` 將取代為您先前建立的 Amazon ECR 儲存庫 URL。	

任務	Description	所需的技能

將 Docker 映像推送至 Amazon ECR。	1. 執行下列命令以登入 Amazon ECR 儲存庫：
	`aws ecr get-login -password --region us-east-1 --no-cli-auto-prompt | docker login --username AWS --password-stdin <your_aws_account_id>.dkr.ecr.us-east-1.amazonaws.com`
	2. 執行下列命令，將 Docker 映像推送至 Amazon ECR 儲存庫：
	`docker push <your_aws_account_id>.dkr.ecr.us-east-1.amazonaws.com/helloworld-grpc:1.0`
	Important
	請確定您使用 AWS 帳戶 ID`<your_aws_account_id>` 取代。
		DevOps 工程師

將 Kubernetes 資訊清單部署至 Amazon EKS 叢集

任務	Description	所需的技能
修改 Kubernetes 資訊清單檔案中的值。	1. 根據您的需求修改儲存庫 `grpc-samp`	DevOps 工程師

任務	Description	所需的技能
	le.yaml Kubernetes 資料夾中的 Kubernetes 資訊清單檔案。您必須修改輸入資源中的註釋和主機名稱。如需範例輸入資源，請參閱 其他資訊 一節。如需輸入註釋的詳細資訊，請參閱 Kubernetes 文件中的輸入註釋。 2. 在 Kubernetes 部署資源中，將部署資源的 image 變更為您推送 Docker 映像的 Amazon ECR 儲存庫的統一資源識別符 (URI)。如需範例部署資源，請參閱其他資訊 一節。	
部署 Kubernetes 資訊清單檔案。	執行下列kubectl命令，將grpc-sample.yaml 檔案部署至 Amazon EKS 叢集： `kubectl apply -f ./kubernetes/grpc-sample.yaml`	DevOps 工程師

建立 Application Load Balancer FQDN 的 DNS 記錄

任務	Description	所需的技能
記錄 Application Load Balancer 的 FQDN。	1. 執行下列kubectl命令來描述管理 Application Load Balancer 的 Kubernetes 輸入資源：	DevOps 工程師

任務	Description	所需的技能
	`kubectl get ingress -n grpcserver` 範例輸出會在其他資訊 區段中提供。在輸出中， HOSTS 欄位會顯示建立 SSL 憑證的 DNS 主機名稱。 2. 從輸出Address的 欄位記錄 Application Load Balancer 的完整網域名稱 (FQDN)。 3. 建立指向 Application Load Balancer FQDN 的 DNS 記錄。如果您的 DNS 供應商是 Amazon Route 53，您可以建立指向 Application Load Balancer FQDN 的別名記錄。如需此選項的詳細資訊，請參閱 Route 53 文件中的選擇別名和非別名記錄。	

測試解決方案

任務	Description	所需的技能
測試 gRPC 伺服器。	執行下列命令，使用 gRPCurl 來測試端點： `grpcurl grpc.example.com:443 list grpc.reflection.v1alpha.ServerReflection helloworld.helloworld`	DevOps 工程師

任務	Description	所需的技能
	Note `grpc.example.com` 將 取代為您的 DNS 名稱。	

使用 gRPC 用戶端測試 gRPC 伺服器。	在`helloworld_client_ssl.py` 範例 gRPC 用戶端中，`grpc.example.com` 將來自 的主機名稱取代為用於 gRPC 伺服器的主機名稱。
	
	下列程式碼範例顯示用戶端請求的 gRPC 伺服器回應：
	
	`python ./app/helloworld_client_ssl.py`
	`message: "Hello to gRPC server from Client"`
	
	`message: "Thanks for talking to gRPC server!! Welcome to hello world. Received message is \"Hello to gRPC server from Client\""`
	`received: true`
	
	這表示用戶端可以與伺服器通訊，而且連線成功。
		DevOps 工程師

清除

任務	Description	所需的技能
移除 DNS 記錄。	移除指向您先前建立之 Application Load Balancer FQDN 的 DNS 記錄。	雲端管理員
移除負載平衡器。	在 Amazon EC2 主控台上，選擇負載平衡器，然後移除 Kubernetes 控制器為輸入資源建立的負載平衡器。	雲端管理員
刪除 Amazon EKS 叢集。	使用 刪除 Amazon EKS 叢集`eksctl`： `eksctl delete cluster -f ./eks.yaml`	AWS DevOps

相關資源

- Amazon EKS 上的網路負載平衡
- Application Load Balancer 的目標群組

其他資訊

範例輸入資源：

```
---
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  annotations:
    alb.ingress.kubernetes.io/healthcheck-protocol: HTTP
    alb.ingress.kubernetes.io/ssl-redirect: "443"
    alb.ingress.kubernetes.io/backend-protocol-version: "GRPC"
    alb.ingress.kubernetes.io/listen-ports: '[{"HTTP": 80}, {"HTTPS":443}]'
    alb.ingress.kubernetes.io/scheme: internet-facing
```

```
    alb.ingress.kubernetes.io/target-type: ip
    alb.ingress.kubernetes.io/certificate-arn: arn:aws:acm:<AWS-
Region>:<AccountId>:certificate/<certificate_ID>
  labels:
    app: grpcserver
    environment: dev
  name: grpcserver
  namespace: grpcserver
spec:
  ingressClassName: alb
  rules:
  - host: grpc.example.com # <----- replace this as per your host name for which the
 SSL certtficate is available in ACM
    http:
      paths:
      - backend:
          service:
            name: grpcserver
            port:
              number: 9000
        path: /
        pathType: Prefix
```

部署資源範例：

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: grpcserver
  namespace: grpcserver
spec:
  selector:
    matchLabels:
      app: grpcserver
  replicas: 1
  template:
    metadata:
      labels:
        app: grpcserver
    spec:
      containers:
      - name: grpc-demo
```

```
        image: <your_aws_account_id>.dkr.ecr.us-east-1.amazonaws.com/helloworld-
grpc:1.0   #<------- Change to the URI that the Docker image is pushed to
        imagePullPolicy: Always
        ports:
        - name: grpc-api
          containerPort: 9000
        env:
        - name: POD_IP
          valueFrom:
            fieldRef:
              fieldPath: status.podIP
      restartPolicy: Always
```

輸出範例：

```
NAME             CLASS           HOSTS                          Address
 PORTS          AGE
 grpcserver      <none>      <DNS-HostName>                 <ELB-address>
 80            27d
```

在做為 Docker 容器 AWS IoT Greengrass V2 執行時部署容器化應用程式

Salih Bakir、Giuseppe Di Bella 和 Gustav Svalander，Amazon Web Services

總結

AWS IoT Greengrass Version 2作為 Docker 容器部署時， 原生不支援執行 Docker 應用程式容器。此模式示範如何根據 AWS IoT Greengrass V2 啟用 Docker-in-Docker (DinD) 功能的最新版本 建立自訂容器映像。使用 DinD，您可以在 AWS IoT Greengrass V2 環境中執行容器化應用程式。

您可以將此模式部署為獨立的解決方案，或將其與 Amazon ECS Anywhere 等容器協同運作平台整合。在任一部署模型中，您維護包括 AWS IoT SiteWise Edge 處理功能的完整 AWS IoT Greengrass V2 功能，同時啟用可擴展的容器型部署。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 如需一般 AWS IoT Greengrass Version 2 先決條件，請參閱 AWS IoT Greengrass Version 2 文件中的先決條件。
- 在 Linux、macOS 或 Windows 上安裝和設定 Docker Engine。
- Docker Compose （ 如果您使用 Docker Compose 命令列界面 (CLI) 來執行 Docker 映像 ）。
- Linux 作業系統。
- 具有支援虛擬化之主機伺服器的 Hypervisor。
- 系統需求：
  - 2 GB RAM （ 最低 ）
  - 5 GB 的可用磁碟空間 （ 最低 ）
  - 對於 AWS IoT SiteWise Edge，具有 16 GB RAM 和 50 GB 可用磁碟空間的 x86_64 四核心 CPU。如需 AWS IoT SiteWise 資料處理的詳細資訊，請參閱 AWS IoT SiteWise 文件中的資料處理套件需求。

產品版本

- AWS IoT Greengrass Version 2 2.5.3 版或更新版本
- Docker-in-Docker 1.0.0 版或更新版本

- Docker Compose 1.22 版或更新版本
- Docker 引擎 20.10.12 版或更新版本

限制

- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

目標技術堆疊

- 資料來源 – 產生資料處理資料的 IoT 裝置、感應器或工業設備
- AWS IoT Greengrass V2 – 做為具有 D-in-D 功能的 Docker 容器執行，部署在邊緣基礎設施上
- 容器化應用程式 – 以巢狀 Docker 容器的形式在 AWS IoT Greengrass V2 環境中執行的自訂應用程式
- （選用）Amazon ECS Anywhere – 管理容器部署的 AWS IoT Greengrass V2 容器協同運作
- 其他 AWS 服務 – AWS IoT Core AWS IoT SiteWise和其他 AWS 服務 用於資料處理和管理

目標架構

下圖顯示使用 Amazon ECS Anywhere 的範例目標部署架構，Amazon ECS Anywhere 是一種容器管理工具。

該圖顯示以下工作流程：

1：容器映像儲存 – Amazon ECR 會存放 AWS IoT Greengrass 容器映像和邊緣處理所需的任何自訂應用程式容器。

2 和 3：容器部署 – Amazon ECS Anywhere 會將 AWS IoT Greengrass 容器映像從 Amazon ECR 部署到節點，管理容器生命週期和部署程序。

4：元件部署 – 部署 AWS IoT Greengrass 的核心會根據其組態自動部署其相關元件。元件包括在容器化環境中的 AWS IoT SiteWise Edge 和其他必要的邊緣處理元件。

5：資料擷取 – 完全設定後， AWS IoT Greengrass 開始從節點的各種 IoT 資料來源擷取遙測和感應器資料。

6：資料處理和雲端整合 – 容器化 AWS IoT Greengrass 核心會使用其部署的元件（包括工業資料的 AWS IoT SiteWise Edge) 在本機處理資料。然後，它會將處理的資料傳送至 AWS 雲端 服務，以進一步分析和儲存。

工具

AWS 服務

- Amazon ECS Anywhere 可協助您在自己的基礎設施上部署、使用和管理 Amazon ECS 任務和服務。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- AWS IoT Greengrass 是一種開放原始碼物聯網 (IoT) 邊緣執行期和雲端服務，可協助您在裝置上建置、部署和管理 IoT 應用程式。
- AWS IoT SiteWise 可協助您大規模收集、建模、分析和視覺化工業設備的資料。

其他工具

- Docker 是一組平台即服務 (PaaS) 產品，可在作業系統層級使用虛擬化在容器中交付軟體。
- Docker Compose 是一種用於定義和執行多容器應用程式的工具。
- Docker Engine 是一種開放原始碼容器化技術，用於建置和容器化應用程式。

程式碼儲存庫

此模式的程式碼可在 GitHub AWS IoT Greengrass v2 Docker-in-Docker 儲存庫中使用。

史詩

建置 AWS IoT Greengrass V2 Docker-in-Docker 映像

任務	Description	所需的技能
複製並導覽至儲存庫。	若要複製儲存庫，請使用下列命令：	DevOps 工程師，AWS DevOps

任務	Description	所需的技能

	`git clone https://github.com/aws-samples/aws-iot-greengrass-v2-docker-in-docker.git`
	
	若要導覽至 docker目錄，請使用下列命令：
	
	`cd aws-iot-greengrass-v2-docker-in-docker/docker`	

任務	Description	所需的技能

建置 Docker 影像。	若要使用預設（最新）版本建置 Docker 映像，請執行下列命令：
	
	`docker build -t x86_64/aws-iot-greengrass:latest .`
	
	或者，若要使用特定版本建置 Docker 映像，請執行下列命令：
	
	`docker build --build-arg GREENGRASS_RELEASE_VERSION=2.12.0 -t x86_64/aws-iot-greengrass:2.12.0 .`
	
	若要驗證組建，請執行下列命令：
	
	`docker images | grep aws-iot-greengrass`
		AWS DevOps、DevOps 工程師、應用程式開發人員

任務	Description	所需的技能

（選用）推送至 Amazon ECR。	1. 若要建立儲存庫，請執行下列命令：
	
	`aws ecr create-repository --repository-name aws-iot-greengrass-dind --region us-east-1`
	
	2. 若要驗證，請執行下列命令：
	
	`aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin <ACCOUNT-ID>.dkr.ecr.us-east-1.amazonaws.com`
	
	3. 若要標記和推送，請執行下列命令：
	
	`docker tag x86_64/aws-iot-greengrass:latest <ACCOUNT-ID>.dkr.ecr.us-east-1.amazonaws.com/aws-iot-greengrass-dind:latest`
	
	`docker push <ACCOUNT_ID>.dkr.ecr.us-east-1.amazonaws.com/aws-iot-g`
		應用程式開發人員、AWS DevOps、DevOps 工程師

任務	Description	所需的技能
	`reengrass-dind:latest`	

設定 AWS 登入資料

任務	Description	所需的技能
選取身分驗證方法。	請選擇下列其中一個選項： • 選項 1（建議）：IAM 角色 – 如果在具有適當 IAM 角色的 Amazon EC2、Amazon ECS 或 Amazon EKS 上執行，請使用 。不需要額外的組態。 • 選項 2：環境變數 – 用於測試和開發。 • 選項 3：登入資料檔案 – 不建議用於生產。 • 選項 4：舊版`env.cfg`檔案。	AWS 管理員
設定身分驗證方法。	對於您選取的身分驗證方法，請使用下列組態指引： • 選項 2（環境變數）– 準備 `AWS_ACCESS_KEY_ID` 、`AWS_SECRET_ACCESS_KEY` 和選用的 `AWS_SESSION_TOKEN` 。 • 選項 3（憑證檔案）– 建立登入資料目錄和檔案： • (a) 執行下列命令：	AWS 管理員

任務	Description	所需的技能
	mkdir -p ~/greengrass-v2-credentials • (b) 在 建立下列檔案~/greengrass-v2-credentials/credentials ： [default] aws_access_key_id=YourAccessKey aws_secret_access_key=YourSecretKey • (c) 保護檔案，如下所示： chmod 600 ~/greengrass-v2-credentials/credentials • 選項 4 (env.cfg) – 在 docker目錄中建立 env.cfg 檔案，如下所示： GGC_ROOT_PATH=/greengrass/v2 AWS_REGION=us-east-1 PROVISION=true THING_NAME=MyGreengrassCore	

任務	Description	所需的技能

	`THING_GROUP_NAME=MyGreengrassCoreGroup`
	
	`TES_ROLE_NAME=GreengrassV2TokenExchangeRole`
	
	`TES_ROLE_ALIAS_NAME=GreengrassCoreTokenExchangeRoleAlias`
	
	`COMPONENT_DEFAULT_USER=ggc_user:ggc_group`	

使用 Docker Compose 執行

任務	Description	所需的技能
設定 `docker-compose.yml` 。	使用環境變數更新 `docker-compose.yml` 檔案，如下所示： • 金鑰變數：`PROVISION`, `AWS_REGION` 、`THING_NAME` 、`THING_GROUP_NAME` 、`TES_ROLE_NAME` 、 `TES_ROLE_ALIAS_NAME` • 對於透過環境變數的登入資料，請取消註解下列項目：	DevOps 工程師

任務	Description	所需的技能
	environment: - AWS_ACCESS_KEY_ID=YourKey - AWS_SECRET_ACCESS_KEY=YourSecret • 對於登入資料檔案，請取消註解下列項目： volumes: - ~/greengrass-v2-credentials:/root/.aws/:ro • 對於日誌持久性，請新增下列項目： volumes: - ./logs:/greengrass/v2/logs	

任務	Description	所需的技能

啟動並驗證容器。	若要從前景開始，請執行下列命令：
	`docker-compose up --build`
	或者，若要從背景開始，請執行下列命令：
	`docker-compose up --build -d`
	若要驗證狀態，請執行下列命令：
	`docker-compose ps`
	若要監控日誌，請執行下列命令：
	`docker-compose logs -f`
		DevOps 工程師

使用 Docker CLI 執行

任務	Description	所需的技能
使用 Docker CLI 執行容器。	• 若要在沒有佈建的情況下啟動基本執行，請執行下列命令： `docker run --init --privileged -it --name aws-iot-greengrass x86_64/aw`	DevOps 工程師

任務	Description	所需的技能
	s-iot-greengrass:l atest • 若要使用環境變數啟動執行，請執行下列命令： docker run --init --privileged - it --name aws- iot-greengrass - e PROVISION=true - e AWS_REGION=us- east-1 -e AWS_ACCES S_KEY_ID=YourKey - e AWS_SECRET_ACCESS_ KEY=YourSecret -e THING_NAME=MyGreen grassCore -e THING_GROUP_NAME=M yGreengrassCoreGro up -p 8883:8883 x86_64/aws-iot-gre engrass:latest • 若要使用登入資料檔案啟動執行，請執行下列命令： docker run --init --privileged -it --name aws-iot-g reengrass -v ~/ greengrass-v2-cr edentials:/root/.a ws:ro -e PROVISION =true -e AWS_REGIO	

任務	Description	所需的技能
	N=us-east-1 -e THING_NAME=MyGreengrassCore -p 8883:8883 x86_64/aws-iot-greengrass:latest • 若要使用 env.cfg 檔案啟動執行，請執行下列命令： docker run --init --privileged -it --name aws-iot-greengrass -v ~/greengrass-v2-credentials:/root/.aws:ro --env-file env.cfg -p 8883:8883 x86_64/aws-iot-greengrass:latest • 若要使用持久性日誌啟動執行，請執行下列命令： docker run --init --privileged -it --name aws-iot-greengrass -v \$(pwd)/logs:/greengrass/v2/logs x86_64/aws-iot-greengrass:latest	

任務	Description	所需的技能

驗證容器。	• 若要檢查容器的狀態，請執行下列命令：
	`docker ps | grep aws-iot-greengrass`
	• 若要監控日誌，請執行下列命令：
	`docker logs -f aws-iot-greengrass`
		DevOps 工程師

管理容器化應用程式

任務	Description	所需的技能

部署應用程式。	• 若要部署應用程式容器，請執行下列命令：
	`docker run -d --name my-app nginx:alpine`
	• 若要驗證部署，請執行下列命令：
	`docker ps`
	• 若要存取日誌，請執行下列命令：
	`cd /greengrass/v2/logs && cat greengrass.log`
		應用程式開發人員

任務	Description	所需的技能

存取和測試 Docker-in-Docker。	1. 若要在容器中開啟 Shell，請執行下列命令：
	`docker exec -it aws-iot-greengrass sh`
	2. 若要驗證 Docker，請執行下列命令：
	`docker --version`
	`docker ps`
	3. 若要測試 Docker-in-Docker，請執行下列命令：
	`docker run --rm hello-world`
		DevOps 工程師

（選用）與 Amazon ECS Anywhere 整合

任務	Description	所需的技能

設定 Amazon ECS 叢集。	• 若要建立 Amazon ECS 叢集，請執行下列命令：
	`aws ecs create-cluster --cluster-name greengrass-cluster --region us-east-1`
	• 若要建立 SSM 啟用，請執行下列命令：
	`aws ssm create-activation --default`
		AWS 管理員

任務	Description	所需的技能
	-instance-name greengrass-edge --iam-role GreengrassECSAnywhereRole --registration-limit 10 --region us-east-1 • 若要在邊緣裝置上安裝 SSM Agent，請執行下列命令： curl -o install.sh https://s3.amazonaws.com/aws-ssm-downloads/latest/linux_amd64/install.sh chmod +x install.sh ./install.sh activation-code activation-id us-east-1	

任務	Description	所需的技能

部署 Amazon ECS 任務。	• greengrass-task-definition.json 使用任務組態建立 ，包括 containerDefinitions privileged:true 、環境變數，以及 /var/lib/docker 和 /greengrass/v2.Register 任務的磁碟區掛載。執行以下命令：
	
	`aws ecs register-task-definition --cli-input-json file://greengrass-task-definition.json`
	
	• 若要建立服務，請執行下列命令：
	
	`aws ecs create-service --cluster greengrass-cluster --service-name greengrass-service --task-definition greengrass-dind --desired-count 1 --launch-type EXTERNAL --region us-east-1`
	
	• 若要驗證部署，請執行下列命令：
		AWS 管理員

任務	Description	所需的技能
	`aws ecs describe-services --cluster greengrass-cluster --services greengrass-service --region us-east-1`	

停止和清除

任務	Description	所需的技能

停止容器。	• 若要使用 Docker Compose 停止容器，請執行下列命令：
	`docker-compose stop`
	`docker-compose down`
	`docker-compose down -v`
	• 若要使用 Docker CLI 停止容器（移除磁碟區），請執行下列命令：
	`docker stop aws-iot-greengrass`
	`docker rm aws-iot-greengrass`
	• 若要使用強制移除來停止容器，請執行下列命令：
		DevOps 工程師

任務	Description	所需的技能
	`docker rm -f aws-iot-greengrass`	

疑難排解

問題	解決方案

容器無法從許可錯誤開始。	1. 若要確保容器使用 Docker-in-Docker 所需的 `--privileged` 旗標執行，請執行下列命令：
	
	`docker run --init --privileged -it --name aws-iot-greengrass x86_64/aws-iot-greengrass:latest`
	
	2. 在 `docker-compose.yml` 檔案中，設定下列項目：
	
	`privileged: true`
	
	⚠ Warning
	`--privileged` 會將延伸權限授予容器。

佈建失敗並出現登入資料錯誤。	若要驗證登入資料設定是否正確，請使用下列步驟： 1. 檢查 IAM 角色： `aws sts get-caller-identity` 2. 驗證環境變數：

問題	解決方案

	`docker exec -it aws-iot-greengrass sh -c 'echo \$AWS_ACCESS_KEY_ID'`
	3. 檢查登入資料檔案：
	`docker exec -it aws-iot-greengrass sh -c 'cat /root/.aws/credentials'`
	確定 IAM 許可包含 `iot:CreateThing` 、`iot:CreatePolicy` 、`iam:CreateRole` 、`iot:AttachPolicy` 和 `iam:AttachRolePolicy` 。

無法連線到容器內的 Docker 協助程式。	1. 檢查 Docker 協助程式狀態：
	`docker exec -it aws-iot-greengrass sh -c 'ps aux | grep dockerd'`
	2. 確認 Docker 通訊端存在：
	`docker exec -it aws-iot-greengrass sh -c 'ls -la /var/run/docker.sock'`
	3. 視需要手動啟動：
	`docker exec -it aws-iot-greengrass sh -c 'dockerd > /var/log/docker.log 2>&1 &'`

問題	解決方案

容器用盡磁碟空間。	1. 檢查磁碟空間：
	`df -h`
	2. 清除 Docker 資源：
	`docker system prune -a --volumes`
	3. 移除舊的 Greengrass 成品：
	`docker exec -it aws-iot-greengrass sh -c 'rm -rf /greengrass/v2/work/*  '`
	確保磁碟空間下限：基本操作為 5 GB，AWS IoT SiteWise 邊緣為 50 GB

建置問題。	1. 請確定您位於 docker目錄中：
	`cd docker && pwd`
	2. 驗證 Dockerfile 是否存在：
	`ls -la Dockerfile`
	3. 檢查 Docker 是否已安裝並執行：
	`docker --version && docker ps`

問題	解決方案

網路連線問題。	1. 測試 DNS 解析：
	`docker exec -it aws-iot-greengrass sh -c 'nslookup google.com'`
	2. 測試 AWS 連線：
	`docker exec -it aws-iot-greengrass sh -c 'curl https://iot.us-east-1.amazonaws.com'`
	確認防火牆允許傳出 HTTPS (443) 和 MQTT (8883) 流量。

Greengrass 元件無法部署。	1. 檢查 Greengrass 日誌：
	`docker exec -it aws-iot-greengrass sh -c 'cat /greengrass/v2/logs/greengrass.log'`
	2. 驗證 TES 角色組態：
	`aws iot describe-role-alias --role-alias TES_ROLE_ALIAS_NAME`
	檢查 `/greengrass/v2/logs/` 目錄中的元件特定日誌。

問題	解決方案

容器會在啟動後立即結束。	1. 檢查容器日誌是否有錯誤：
	`docker logs aws-iot-greengrass`
	2. 使用互動式 shell 執行 以進行偵錯：
	`docker run --init --privileged -it --entrypoint sh x86_64/aws-iot-greengrass:latest`
	如果 ，請確認已正確設定所有必要的環境變數 `PROVISION=true` 。啟動容器時，請務必使用 `--init`旗標。

相關資源

AWS resources

- Amazon Elastic Container Service
- 設定 AWS IoT SiteWise 模型和資產的邊緣資料處理
- 什麼是 AWS IoT Greengrass

其他資源

- Docker 文件

其他資訊

- 對於 AWS IoT SiteWise Edge 資料處理，Docker 必須在 AWS IoT Greengrass 環境中可用。
- 若要執行巢狀容器，您必須使用管理員層級登入資料執行 AWS IoT Greengrass 容器。

使用 Elastic Beanstalk 部署容器

Thomas Scott 和 Jean-Baptiste Guillois，Amazon Web Services

總結

在 Amazon Web Services (AWS) 雲端上，AWS Elastic Beanstalk 支援 Docker 做為可用的平台，讓容器可以使用建立的環境來執行。此模式示範如何使用 Elastic Beanstalk 服務部署容器。此模式的部署將使用以 Docker 平台為基礎的 Web 伺服器環境。

若要使用 Elastic Beanstalk 部署和擴展 Web 應用程式和服務，請上傳程式碼並自動處理部署。容量佈建、負載平衡、自動擴展和應用程式運作狀態監控也包含在內。當您使用 Elastic Beanstalk 時，您可以完全控制其代表您建立的 AWS 資源。使用 Elastic Beanstalk 並不收取其他費用。您只需為用於存放和執行應用程式的 AWS 資源付費。

此模式包含使用 AWS Elastic Beanstalk 命令列界面 (EB CLI) 和 AWS 管理主控台進行部署的說明。

使用案例

Elastic Beanstalk 的使用案例包括下列項目：

- 部署原型環境以示範前端應用程式。（此模式使用 Dockerfile 作為範例。）
- 部署 API 來處理指定網域的 API 請求。
- 使用 Docker-Compose 部署協同運作解決方案 (`docker-compose.yml` 不在此模式中做為實際範例）。

先決條件和限制

先決條件

- 一個 AWS 帳戶
- 本機安裝的 AWS EB CLI
- 安裝在本機電腦上的 Docker

限制

- 免費計劃上每個 IP 地址每 6 小時的 Docker 提取限制為 100 次提取。

Architecture

目標技術堆疊

- Amazon Elastic Compute Cloud (Amazon EC2) 執行個體
- 安全群組
- Application Load Balancer
- Auto Scaling 群組

目標架構

自動化和擴展

AWS Elastic Beanstalk 可以根據提出的請求數量自動擴展。為環境建立的 AWS 資源包括一個 Application Load Balancer、Auto Scaling 群組，以及一或多個 Amazon EC2 執行個體。

負載平衡器位於 Amazon EC2 執行個體前面，這是 Auto Scaling 群組的一部分。Amazon EC2 Auto Scaling 會自動啟動額外的 Amazon EC2 執行個體，以容納您的應用程式增加的負載。如果您的應用程式負載減少，Amazon EC2 Auto Scaling 會停止執行個體，但至少會讓一個執行個體持續執行。

自動擴展觸發條件

Elastic Beanstalk 環境中的 Auto Scaling 群組會使用兩個 Amazon CloudWatch 警示來啟動擴展操作。當每個執行個體的平均傳出網路流量，在五分鐘期間高於 6 MB 或低於 2 MB 時，預設的觸發條件就會擴展。如要有效地使用 Amazon EC2 Auto Scaling，請根據您的應用程式、執行個體類型和服務需求，設定適用的觸發。您可以根據多項統計資料來進行擴展，包括延遲、磁碟 I/O、CPU 使用率和請求計數。如需詳細資訊，請參閱 Auto Scaling 觸發條件。

工具

AWS 服務

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- AWS EB Command Line Interface (EB CLI) 是命令列用戶端，可用來建立、設定和管理 Elastic Beanstalk 環境。
- Elastic Load Balancing 會將傳入的應用程式或網路流量分散到多個目標。例如，您可以在一或多個可用區域中跨 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、容器和 IP 地址分配流量。

其他服務

- Docker 會將軟體封裝至稱為容器的標準化單位，其中包括程式庫、系統工具、程式碼和執行時間。

Code

此模式的程式碼可在 GitHub 叢集範例應用程式儲存庫中使用。

史詩

使用 Dockerfile 建置

任務	Description	所需的技能
複製遠端儲存庫。	• 若要複製儲存庫，請執行命令 `git clone https://github.com/aws-samples/cluster-sample-app.git .</p>`	應用程式開發人員、AWS 管理員、AWS DevOps
初始化 Elastic Beanstalk Docker 專案。	1. 在根`aws.json`目錄建立名為 的檔案。 2. 在 `aws.json` 檔案中，新增下列程式碼。 `{ "AWSEBDockerrunVersion":"1", "Image":{ "Name":"cluster-sample-app" }, "Ports":[ { "ContainerPort":80 , "HostPort":8080 }`	應用程式開發人員、AWS 管理員、AWS DevOps

任務	Description	所需的技能

	``
	        ]
	    }
	``` 3. `eb init -p docker` 在專案的根目錄執行 命令。	

在本機測試專案。	1. `eb local run` 在專案的根目錄執行 命令。 2. 導覽至 以測試應用程式`http://localhost` 。	應用程式開發人員、AWS 管理員、AWS DevOps

使用 EB CLI 部署

任務	Description	所需的技能
執行部署命令	1. `eb create docker-sample-cluster-app` 在專案的根目錄執行 命令。	應用程式開發人員、AWS 管理員、AWS DevOps
存取部署的版本。	部署命令完成後，請使用 `eb open`命令存取專案。	應用程式開發人員、AWS 管理員、AWS DevOps

使用主控台部署

任務	Description	所需的技能
使用瀏覽器部署應用程式。	1. 開啟 主控台。 2. 導覽至 Elastic Beanstalk 主控台。 3. 選擇建立應用程式。 4. 在應用程式名稱中，輸入 Cluster-Sample-App。 5. 選擇 Docker 做為平台。 6. 選擇上傳您的程式碼。	應用程式開發人員、AWS 管理員、AWS DevOps

任務	Description	所需的技能
	7. 選擇您的本機 .zip 檔案（在複製專案的根目錄中）或公有 Amazon Simple Storage Service (Amazon S3) URL。	
存取部署的版本。	部署之後，請存取部署的應用程式，然後選擇提供的 URL。	應用程式開發人員、AWS 管理員、AWS DevOps

相關資源

- Web 伺服器環境
- 在 macOS 上安裝 EB CLI
- 手動安裝 EB CLI

其他資訊

使用 Elastic Beanstalk 的優點

- 自動基礎設施佈建
- 基礎平台的自動管理
- 支援應用程式的自動修補和更新
- 應用程式自動擴展
- 自訂節點數量的能力
- 能夠視需要存取基礎設施元件
- 輕鬆部署到其他容器部署解決方案

使用 Lambda 函數、Amazon VPC 和無伺服器架構產生靜態傳出 IP 地址

Thomas Scott，Amazon Web Services

摘要

此模式說明如何使用無伺服器架構，在 Amazon Web Services (AWS) 雲端中產生靜態傳出 IP 地址。如果組織想要使用安全檔案傳輸通訊協定 (SFTP) 將檔案傳送至不同的商業實體，則可以從此方法中受益。這表示商業實體必須能夠存取允許檔案通過其防火牆的 IP 地址。

模式的方法可協助您建立使用彈性 IP 地址做為傳出 IP 地址的 AWS Lambda 函數。透過遵循此模式中的步驟，您可以建立 Lambda 函數和虛擬私有雲端 (VPC)，透過具有靜態 IP 地址的網際網路閘道路由傳出流量。若要使用靜態 IP 地址，請將 Lambda 函數連接至 VPC 及其子網路。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 建立和部署 Lambda 函數，以及建立 VPC 及其子網路的 AWS Identity and Access Management (IAM) 許可。如需詳細資訊，請參閱 AWS Lambda 文件中的執行角色和使用者許可。
- 如果您打算使用基礎設施做為程式碼 (IaC) 來實作此模式的方法，則需要整合的開發環境 (IDE)，例如 AWS Cloud9。如需詳細資訊，請參閱 AWS Cloud9 文件中的什麼是 AWS Cloud9？。 AWS Cloud9

Architecture

下圖顯示此模式的無伺服器架構。

該圖顯示以下工作流程：

1. 傳出流量會在 NAT gateway 1中離開Public subnet 1。
2. 傳出流量會在 NAT gateway 2中離開Public subnet 2。
3. Lambda 函數可以在 Private subnet 1或 中執行Private subnet 2。
4. Private subnet 1 和 將流量Private subnet 2路由到公有子網路中的 NAT 閘道。
5. NAT 閘道會從公有子網路將傳出流量傳送至網際網路閘道。
6. 傳出資料會從網際網路閘道傳輸到外部伺服器。

技術堆疊

- Lambda
- Amazon Virtual Private Cloud (Amazon VPC)

自動化和擴展

您可以在不同的可用區域中使用兩個公有和兩個私有子網路，以確保高可用性 (HA)。即使一個可用區域無法使用，模式的解決方案仍會繼續運作。

工具

- AWS Lambda – AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。只需為使用的運算時間支付費用，一旦未執行程式碼，就會停止計費。
- Amazon VPC – Amazon Virtual Private Cloud (Amazon VPC) 會佈建 AWS 雲端的邏輯隔離區段，您可以在已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路與您在資料中心中操作的傳統網路非常相似，且具備使用 AWS 可擴展基礎設施的優勢。

史詩

建立新 VPC

任務	Description	所需的技能
建立新 VPC	登入 AWS 管理主控台，開啟 Amazon VPC 主控台，然後建立名為 `Lambda VPC` 且`10.0.0.0/25` 具有 IPv4 CIDR 範圍的 VPC。 如需建立 VPC 的詳細資訊，請參閱 Amazon VPC 文件中的 Amazon VPC 入門。	AWS 管理員

建立兩個公有子網路

任務	Description	所需的技能
建立第一個公有子網路。	1. 在 Amazon VPC 主控台上，選擇子網路 ，然後選擇建立子網路。 2. 針對名稱標籤，輸入 `public-one` 。 3. 對於 VPC，請選擇 Lambda VPC。 4. 選擇可用區域並進行記錄。 5. 針對 IPv4 CIDR 區塊，輸入`10.0.0.0/28` ，然後選擇建立子網路。	AWS 管理員
建立第二個公有子網路。	1. 在 Amazon VPC 主控台上，選擇子網路 ，然後選擇建立子網路。 2. 針對名稱標籤，輸入 `public-two` 。 3. 對於 VPC，請選擇 Lambda VPC。 4. **Important** 選擇可用區域並進行記錄。：您無法使用包含`public-one` 子網路的可用區域。 5. 針對 IPv4 CIDR 區塊，輸入`10.0.0.16/28` ，然後選擇建立 子網路。	AWS 管理員

建立兩個私有子網路

任務	Description	所需的技能
建立第一個私有子網路。	1. 在 Amazon VPC 主控台上，選擇子網路 ，然後選擇建立子網路。 2. 針對名稱標籤，輸入 `private-one` 。 3. 對於 VPC，請選擇 Lambda VPC。 4. 選擇包含您先前建立之`public-one` 子網路的可用區域。 5. 針對 IPv4 CIDR 區塊，輸入`10.0.0.32/28` ，然後選擇建立子網路。	AWS 管理員
建立第二個私有子網路。	1. 在 Amazon VPC 主控台上，選擇子網路 ，然後選擇建立子網路。 2. 針對名稱標籤，輸入 `private-two` 。 3. 對於 VPC，請選擇 Lambda VPC。 4. 選擇包含您先前建立之`public-two` 子網路的相同可用區域。 5. 針對 IPv4 CIDR 區塊，輸入`10.0.0.64/28` ，然後選擇建立子網路。	AWS 管理員

為您的 NAT 閘道建立兩個彈性 IP 地址

任務	Description	所需的技能
建立第一個彈性 IP 地址。	1. 在 Amazon VPC 主控台上，選擇彈性 IPs，然後選擇配置新地址。 2. 選擇 配置 ，並記錄新建立彈性 IP 地址的配置 ID 。 **Note** 此彈性 IP 地址用於您的第一個 NAT 閘道。	AWS 管理員
建立第二個彈性 IP 地址。	1. 在 Amazon VPC 主控台上，選擇彈性 IPs，然後選擇配置新地址。 2. 選擇 配置 ，並記錄第二個彈性 IP 地址的配置 ID 。 **Note** 此彈性 IP 地址用於您的第二個 NAT 閘道。	AWS 管理員

建立網際網路閘道

任務	Description	所需的技能
建立網際網路閘道	1. 在 Amazon VPC 主控台上，選擇網際網路閘道，然後選擇建立網際網路閘道。	AWS 管理員

任務	Description	所需的技能
	2. 輸入 `Lambda internet gateway`做為名稱，然後選擇建立網際網路閘道。請務必記錄網際網路閘道 ID。	
將網際網路閘道連接至 VPC。	選取您剛建立的網際網路閘道，然後選擇 Actions, Attach to VPC (動作、連接到 VPC)。	AWS 管理員

建立兩個 NAT 閘道

任務	Description	所需的技能
建立第一個 NAT 閘道。	1. 在 Amazon VPC 主控台上，選擇 NAT 閘道 ，然後選擇 建立 NAT 閘道。 2. 輸入`nat-one` 做為 NAT 閘道名稱。 3. 選擇`public-one` 作為要在其中建立 NAT 閘道的子網路。 4. 針對連線類型，選擇公有。 5. 針對彈性 IP 配置 ID，選擇您先前建立的第一個彈性 IP 地址，並將其與 NAT 閘道建立關聯。 6. 選擇建立 NAT 閘道。	AWS 管理員
建立第二個 NAT 閘道。	1. 在 Amazon VPC 主控台上，選擇 NAT 閘道 ，然後選擇 建立 NAT 閘道。 2. 輸入`nat-two` 做為 NAT 閘道名稱。	AWS 管理員

任務	Description	所需的技能
	3. 選擇public-two 作為要在其中建立 NAT 閘道的子網路。 4. 針對連線類型，選擇公有。 5. 針對彈性 IP 配置 ID，選擇您先前建立的第二個 彈性 IP 地址，並將其與 NAT 閘道建立關聯。 6. 選擇建立 NAT 閘道。	

為您的公有和私有子網路建立路由表

任務	Description	所需的技能
建立公有-1 子網路的路由表。	1. 在 Amazon VPC 主控台上，選擇路由表，然後選擇建立路由表。 2. 輸入public-one-subnet 做為路由表名稱，然後選擇建立路由表。 3. 選擇public-one-subnet路由表，選擇編輯路由，然後選擇新增路由。 4. 0.0.0.0 在目的地方塊中指定 ，然後在目標清單中選擇網際網路閘道 ID。 5. 在子網路關聯索引標籤上，選擇編輯子網路關聯，選擇 具有 10.0.0.0/28 CIDR 範圍的public-one 子網路，然後選擇儲存關聯。	AWS 管理員

任務	Description	所需的技能
	6. 選擇 Save Changes (儲存變更)。	
建立公有-二子網路的路由表。	1. 在 Amazon VPC 主控台上，選擇路由表，然後選擇建立路由表。 2. 輸入public-two-subnet 做為路由表名稱，然後選擇建立路由表。 3. 選擇public-two-subnet 路由表，選擇編輯路由，然後選擇新增路由。 4. 0.0.0.0 在目的地方塊中指定 ，然後在目標清單中選擇網際網路閘道 ID。 5. 在子網路關聯索引標籤上，選擇編輯子網路關聯，選擇具有 10.0.0.16/28 CIDR 範圍的public-two 子網路，然後選擇儲存關聯。 6. 選擇 Save Changes (儲存變更)。	AWS 管理員

任務	Description	所需的技能
建立私有子網路的路由表。	1. 在 Amazon VPC 主控台上，選擇路由表，然後選擇建立路由表。 2. 輸入`private-one-subnet` 做為路由表名稱，然後選擇建立路由表。 3. 選擇`private-one-subnet` 路由表，選擇編輯路由，然後選擇新增路由。 4. `0.0.0.0` 在目的地方塊中指定 ，然後在目標清單中選擇`public-one` 子網路中的 NAT 閘道。 5. 在子網路關聯索引標籤上，選擇編輯子網路關聯，選擇具有 `10.0.0.32/28` CIDR 範圍的`private-one` 子網路，然後選擇儲存關聯。 6. 選擇 Save Changes (儲存變更)。	AWS 管理員

任務	Description	所需的技能
建立私有-兩個子網路的路由表。	1. 在 Amazon VPC 主控台上，選擇路由表，然後選擇建立路由表。 2. 輸入`private-two-subnet` 做為路由表名稱，然後選擇建立路由表。 3. 選擇`private-two-subnet` 路由表，選擇編輯路由，然後選擇新增路由。 4. `0.0.0.0` 在目的地方塊中指定 ，然後在目標清單中選擇`public-two` 子網路中的 NAT 閘道。 5. 在子網路關聯索引標籤上，選擇編輯子網路關聯，選擇具有 `10.0.0.64/28` CIDR 範圍的`private-two` 子網路，然後選擇儲存關聯。 6. 選擇 Save Changes (儲存變更)。	AWS 管理員

建立 Lambda 函數，將其新增至 VPC，然後測試解決方案

任務	Description	所需的技能
建立新 Lambda 函數。	1. 開啟 AWS Lambda 主控台，然後選擇建立函數。 2. 在基本資訊下，在函數名稱`Lambda test`下輸入 ，然後在執行時間下選擇您選擇的語言。	AWS 管理員

任務	Description	所需的技能
	3. 選擇建立函數。	
將 Lambda 函數新增至 VPC。	1. 在 AWS Lambda 主控台上，選擇函數，然後選擇您先前建立的函數。 2. 選擇 Configuration (組態)，然後選擇 VPC。 3. 選擇編輯 ，然後選擇Lambda VPC 和兩個私有子網路。 4. 選擇預設安全群組 進行測試，然後選擇儲存。	AWS 管理員
編寫程式碼以呼叫外部服務。	1. 在您選擇的程式設計語言中，撰寫程式碼來呼叫傳回 IP 地址的外部服務。 2. 確認傳回的 IP 地址符合您其中一個彈性 IP 地址。	AWS 管理員

相關資源

- 設定 Lambda 函式以存取 VPC 中的資源

遷移至 Amazon ECR 儲存庫時，自動識別重複的容器映像

Rishabh Yadav 和 Rishi Singla，Amazon Web Services

總結

模式提供自動化解決方案，以識別存放在不同容器儲存庫中的映像是否為重複。當您計劃將映像從其他容器儲存庫遷移到 Amazon Elastic Container Registry (Amazon ECR) 時，此檢查很有用。

如需基礎資訊，模式也會說明容器映像的元件，例如映像摘要、資訊清單和標籤。當您計劃遷移至 Amazon ECR 時，您可以比較映像摘要，以決定跨容器登錄檔同步容器映像。遷移容器映像之前，您需要檢查 Amazon ECR 儲存庫中是否存在這些映像，以防止重複。不過，比較映像摘要可能很難偵測重複項目，這可能會導致初始遷移階段發生問題。 此模式會比較存放在不同容器登錄檔中的兩個類似影像的摘要，並說明摘要為何不同，以協助您準確比較影像。

先決條件和限制

- 作用中 AWS 帳戶
- 存取 Amazon ECR 公有登錄檔
- 熟悉下列項目 AWS 服務：
  - AWS CodeCommit
  - AWS CodePipeline
  - AWS CodeBuild
  - AWS Identity and Access Management (IAM)
  - Amazon Simple Storage Service (Amazon S3)
- 設定的 CodeCommit 登入資料（請參閱說明）

Architecture

容器映像元件

下圖說明容器映像的一些元件。這些元件會在圖表後說明。

術語和定義

下列術語在開放式容器計畫 (OCI) 映像規格中定義。

- 登錄檔：映像儲存和管理的服務。

- 用戶端：與登錄檔通訊並搭配本機映像使用的工具。
- 推送：將映像上傳至登錄檔的程序。
- 提取：從登錄檔下載映像的程序。
- Blob：由登錄檔存放且可由摘要處理之內容的二進位形式。
- 索引：用於識別不同電腦平台（例如 x86-64 或 ARM 64 位元）或媒體類型的多個影像資訊清單的建構。如需詳細資訊，請參閱 OCI 影像索引規格。
- 資訊清單：JSON 文件，定義透過資訊清單端點上傳的影像或成品。資訊清單可以使用描述項來參考儲存庫中的其他 Blob。如需詳細資訊，請參閱 OCI 映像資訊清單規格。
- 檔案系統層：影像的系統程式庫和其他相依性。
- 組態：包含成品中繼資料並在資訊清單中參考的 Blob。如需詳細資訊，請參閱 OCI 映像組態規格。
- 物件或成品：儲存在 Blob 中的概念內容項目，並與具有組態的隨附資訊清單相關聯。
- 摘要：從資訊清單內容的密碼編譯雜湊建立的唯一識別符。映像摘要有助於唯一識別不可變的容器映像。當您使用映像摘要提取映像時，每次在任何作業系統或架構上都會下載相同的映像。如需詳細資訊，請參閱 OCI Image Specification。
- 標籤：人類可讀取的資訊清單識別符。相較於不可變的影像摘要，標籤是動態的。指向影像的標籤可以變更，並從一個影像移至另一個影像，但基礎影像摘要保持不變。

目標架構

下圖顯示此模式所提供解決方案的高階架構，透過比較存放在 Amazon ECR 和私有儲存庫中的映像來識別重複的容器映像。

工具

AWS 服務

- CloudFormation 可協助您設定 AWS 資源、快速一致地佈建資源，以及在整個 AWS 帳戶 和 區域的生命週期中管理資源。
- AWS CodeBuild是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。

- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。

程式碼

此模式的程式碼可在 GitHub 儲存庫中取得 自動化解決方案，以識別儲存庫之間的重複容器映像。

最佳實務

- CloudFormation 最佳實務
- AWS CodePipeline 最佳實務

史詩

從 Amazon ECR 公有和私有儲存庫提取容器映像

任務	Description	所需的技能

從 Amazon ECR 公有儲存庫中提取映像。	從終端機執行下列命令，`amazonlinux` 從 Amazon ECR 公有儲存庫提取映像。
	`$~ % docker pull public.ecr.aws/amazonlinux/amazonlinux:2018.03`
	將映像提取至本機機器後，您會看到下列提取摘要，其代表映像索引。
	`2018.03: Pulling from amazonlinux/amazonlinux`
	`4ddc0f8d367f: Pull complete`
		應用程式開發人員、AWS DevOps、AWS 管理員

任務	Description	所需的技能
	Digest: sha256:f9 72d24199508c52de7a d37a298bda35d8a1bd 7df158149b381c03f6 c6e363b5 Status: Downloade d newer image for public.ecr.aws/ama zonlinux/amazonlin ux:2018.03 public.ecr.aws/a mazonlinux/amazonl inux:2018.03	

任務	Description	所需的技能

將映像推送至 Amazon ECR 私有儲存庫。	1. 在美國東部（維吉尼亞北部）區域 () 建立名為 `test_ecr_repository` 的私有 Amazon ECR us-east-1 儲存庫。
	
	`$~ % aws ecr get-login -password --region us-east-1 | docker login --username AWS --password-stdin <account-id>.dkr.ecr.us-east-1.amazonaws.com`
	`Login Succeeded`
	
	其中 `<account-id>` 是指您的 AWS 帳戶。
	2. 標記您先前提取的本機映像。使用 值並將其`public.ecr.aws/amazonlinux/amazonlinux:2018.03` 推送至 Amazon ECR 私有儲存庫。
	
	`$~ % docker tag public.ecr.aws/amazonlinux/amazonlinux:2018.03 <account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest`
	
	`$~ % docker push <account-id>.dkr.ecr.us-east-1.amazo`
		AWS 管理員、AWS DevOps、應用程式開發人員

任務	Description	所需的技能
	naws.com/test_ecr_repository:latest 當您將映像推送至 Amazon ECR 儲存庫時，Docker 會推送基礎映像，而不是映像索引。 The push refers to repository [<account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository] d5655967c2c4: Pushed latest: digest: sha256:52db9000073d93b9bdee6a7246a68c35a741aaade05a8f4febba0bf795cdac02 size: 529	

任務	Description	所需的技能

從 Amazon ECR 私有儲存庫提取相同的映像。	1. 從終端機執行下列命令，以提取您先前推送到 Amazon ECR 私有儲存庫的映像。
	
	`$~ % docker pull <account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest`
	`latest: Pulling from test_ecr_repository`
	`Digest: sha256:52db9000073d93b9bdee6a7246a68c35a741aaade05a8f4febba0bf795cdac02`
	`Status: Image is up to date for <account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest`
	`<account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest`
	
	此映像的摘要符合您推送至 Amazon ECR 私有儲存庫的映像摘要，並代表基礎映像。此值與您從公有儲存庫提取的影像索引不相符。
	2. 若要驗證，請依摘要擷取影像索引。
	
	`curl -k -H "Authorization: Bearer $TOKEN" https://public.ecr`
		應用程式開發人員、AWS DevOps、AWS 管理員

任務	Description	所需的技能

	.aws/v2/amazonlinux/amazonlinux/manifests/sha256:f972d24199508c52de7ad37a298bda35d8a1bd7df158149b381c03f6c6e363b55
	{
	"schemaVersion": 2,
	"mediaType": "application/vnd.docker.distribution.manifest.list.v2+json",
	"manifests": [
	{
	"mediaType": "application/vnd.docker.distribution.manifest.v2+json",
	"size": 529,
	"digest": "sha256:52db9000073d93b9bdee6a7246a68c35a741aaade05a8f4febba0bf795cdac02",
	"platform": {
	"architecture": "amd64",
	"os": "linux"
	}
	}
	]
	}	

比較影像資訊清單

任務	Description	所需的技能

尋找存放在 Amazon ECR 公有儲存庫中的映像資訊清單。	從終端機執行下列命令，`public.ecr.aws/amazonlinux/amazonlinux:2018.03` 從 Amazon ECR 公有儲存庫提取映像資訊清單。
	
	`$~ % docker manifest inspect public.ecr.aws/amazonlinux/amazonlinux:2018.03`
	`{`
	`"schemaVersion": 2,`
	`"mediaType": "application/vnd.docker.distribution.manifest.list.v2+json",`
	`"manifests": [`
	`{`
	`"mediaType": "application/vnd.docker.distribution.manifest.v2+json",`
	`"size": 529,`
	`"digest": "sha256:52db9000073d93b9bdee6a7246a68c35a741aaade05a8f4febba0bf795cdac02",`
	`"platform": {`
	`"architecture": "amd64",`
	`"os": "linux"`
	`}`
	`}`
	`]`
		AWS 管理員、AWS DevOps、應用程式開發人員

任務	Description	所需的技能
	} ```	

任務	Description	所需的技能

尋找存放在 Amazon ECR 私有儲存庫中的映像資訊清單。	從終端機執行下列命令，`<account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest` 從 Amazon ECR 私有儲存庫提取映像資訊清單。
	
	`$~ % docker manifest inspect <account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest`
	
	`{ "schemaVersion": 2, "mediaType": "application/vnd.docker.distribution.manifest.v2+json", "config": { "mediaType": "application/vnd.docker.container.image.v1+json", "size": 1477, "digest": "sha256:f7cee5e1af28ad4e147589c474d399b12d9b551ef4c3e11e02d982fce5eebc68" }, "layers": [{ "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",`
		AWS DevOps、AWS 系統管理員、應用程式開發人員

任務	Description	所需的技能
	"size": 62267075, "digest": "sha256:4ddc0f8d367f424871a060e2067749f32bd36a91085e714dcb159952f2d71453" }] }	

任務	Description	所需的技能
比較 Docker 提取的摘要與 Amazon ECR 私有儲存庫中影像的資訊清單摘要。	另一個問題是為什麼 docker pull 命令提供的摘要與影像 的資訊清單摘要不同<account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest 。 用於 docker pull 的摘要代表影像資訊清單的摘要，存放在登錄檔中。此摘要會被視為雜湊鏈的根目錄，因為資訊清單包含將下載並匯入 Docker 的內容雜湊。 在此資訊清單中，Docker 中使用的映像 ID 為 config.digest 。這代表 Docker 使用的映像組態。因此，您可以說資訊清單是信封，而影像是信封的內容。資訊清單摘要一律與影像 ID 不同。不過，特定資訊清單應一律產生相同的影像 ID。由於資訊清單摘要是雜湊鏈，我們無法保證指定影像 ID 永遠相同。在大多數情況下，它會產生相同的摘要，但 Docker 無法保證這一點。資訊清單摘要中的可能差異源於 Docker 不會儲存本機使用 gzip 壓縮的 Blob。因此，匯出層可能會產生不同的摘要，但未壓縮的內容保持不變。影像 ID 會驗證未壓縮的內容是否相同；	AWS DevOps、AWS 系統管理員、應用程式開發人員

任務	Description	所需的技能
	也就是說，影像 ID 現在是內容可定址識別符 (chainID)。 若要確認此資訊，您可以比較 Amazon ECR 公有和私有儲存庫上 docker inspect 命令的輸出： 1. 針對存放在 Amazon ECR 公有儲存庫中的映像，從您的終端機執行下列命令。 ``` $~ % docker inspect public.ecr.aws/ama zonlinux/amazonlin ux:2018.03 ``` 如需 命令的輸出，請參閱其他資訊一節。 2. 針對存放在 Amazon ECR 私有儲存庫中的映像，從您的終端機執行下列命令。 ``` $~ % docker inspect <account-id>.dkr.e cr.us-east-1.amazo naws.com/test_ecr_ repository:latest ``` 如需 命令的輸出，請參閱其他資訊一節。 結果會驗證兩個影像具有相同的影像 ID 摘要和圖層摘要。	

任務	Description	所需的技能
	ID： `f7cee5e1af28ad4e14 7589c474d399b12d9b 551ef4c3e11e02d982 fce5eebc68` 圖層： `d5655967c 2c4e8d68f8ec7cf753 218938669e6c16ac13 24303c073c736a2e2a2` 此外，摘要是根據本機受管物件的位元組 （本機檔案是容器映像層的 tar) 或推送至登錄伺服器的 Blob。不過，當您將 Blob 推送至登錄檔時，會壓縮 tar，並在壓縮的 tar 檔案中計算摘要。因此，Docker 提取摘要值的差異源自於在登錄檔 (Amazon ECR 私有或公有） 層級套用的壓縮。 **Note** 此說明專屬於使用 Docker 用戶端。您將不會在其他用戶端看到此行為，例如 nerdctl 或 Finch，因為它們不會在推送和提取操作期間自動壓縮映像。	

自動識別 Amazon ECR 公有和私有儲存庫之間的重複映像

任務	Description	所需的技能
複製儲存庫。	將此模式的 Github 儲存庫複製到本機資料夾： `$git clone https://github.com/aws-samples/automated-solution-to-identify-duplicate-container-images-between-repositories`	AWS 管理員、AWS DevOps
設定 CI/CD 管道。	GitHub 儲存庫包含一個 `.yaml` 檔案，可建立 CloudFormation 堆疊以在其中設定管道 AWS CodePipeline。 1. 登入 AWS 管理主控台 並開啟 CloudFormation 主控台。 2. 使用範本`pipeline.yaml` 檔案建立堆疊，該檔案位於複製儲存庫的 `code` 資料夾中。 3. 接受或變更參數的預設值。指定下列項目的值： • Stack name (堆疊名稱) • `ArtifactStoreBucketName` – 用來存放 AWS CodePipeline 成品的現有 S3 儲存貯體 • `OutputBucket` – 現有的 S3 儲存貯體，用來存放重複映像URIs	AWS 管理員、AWS DevOps

任務	Description	所需的技能
	• SourceImageFile – 名為 的現有文字檔案input.txt ，其中包含來自公有儲存庫的映像 URIs，將針對 Amazon ECR 私有儲存庫進行檢查，以偵測重複項目 4. 檢閱和調整堆疊選項，然後選擇提交以執行範本。 管道將設定兩個階段 (CodeCommit 和 CodeBuild，如架構圖所示），以識別私有儲存庫中也存在於公有儲存庫中的映像。管道是以下列資源設定： • 用於協調部署管道的 CodePipeline。 • 儲存 bash 指令碼和輸入檔案的 CodeCommit 儲存庫。bash 指令碼用於比較公有和私有儲存庫中的容器映像 IDs，以尋找重複項目。此檢查會在單一 中指定的所有儲存庫 AWS 帳戶 中執行 AWS 區域。 • CodeBuild 專案會叫用 bash 指令碼，以識別已存在於 Amazon ECR 儲存庫中的映像。 • 允許存取的必要 IAM 角色。 • 儲存包含映像 URIs 的輸出檔案的 S3 儲存貯體。	

任務	Description	所需的技能
	• 儲存 CodePipeline 成品的另一個 S3 儲存貯體。	

任務	Description	所需的技能
填入 CodeCommit 儲存庫。	若要填入 CodeCommit 儲存庫，請執行下列步驟： 1. 開啟 CodeCommit 主控台並導覽至 AWS 區域 您建立 CloudFormation 堆疊的 。 2. 從清單中尋找您使用 CloudFormation 指令碼佈建的儲存庫，選擇複製 URL，然後複製 HTTPS URL 通訊協定以連線至儲存庫。 3. 開啟命令提示字元，並使用您在上一個步驟中複製的 HTTPS URL 執行 git 複製命令。 4. 導覽至根目錄。建立名為 的檔案，`input.txt` 並將您要在私有 Amazon ECR 儲存庫中搜尋的 Amazon ECR 公有映像登錄 URIs 填入此檔案。 5. `input.txt` 從 GitHub `script.sh` 儲存庫的本機複本複製檔案 `buildspec.yml` 、 和 自動化解決方案，以識別儲存庫與複製的 CodeCommit 儲存庫之間的重複容器映像。CodeCommit 6. 使用以下命令將檔案上傳至 CodeCommit： `git add .`	AWS 管理員、AWS DevOps

任務	Description	所需的技能
	`git commit -m "added input files"` `git push`	
清除。	若要避免產生未來費用，請依照下列步驟刪除資源： 1. 導覽至存放 CodePipeline 成品的 S3 儲存貯體，然後清空儲存貯體。 2. 導覽至存放重複映像 URIs的 S3 儲存貯體，然後清空儲存貯體。 3. 導覽至 CloudFormation 主控台，並刪除您建立來設定管道的堆疊。	AWS 管理員

疑難排解

問題	解決方案
當您嘗試從終端機或命令列推送、提取或以其他方式與 CodeCommit 儲存庫互動時，系統會提示您提供使用者名稱和密碼，而且您必須為 IAM 使用者提供 Git 登入資料。	此錯誤的最常見原因如下： • 您的本機電腦正在執行不支援登入資料管理的作業系統，或未安裝登入資料管理公用程式。 • IAM 使用者的 Git 登入資料尚未儲存至其中一個登入資料管理系統。 根據作業系統和本機環境，您可能需要安裝登入資料管理工具、設定作業系統中包含的登入資料管理工具，或自訂本機環境以使用登入資料儲存體。例如，如果您的電腦正在執行 macOS，您可以使用 Keychain Access 公用程式來存放您的登入資料。如果您的電腦執行 Windows，

問題	解決方案
	您可以使用隨著適用於 Windows 的 Git 安裝的 Credential Manager。如需詳細資訊，請參閱 CodeCommit 文件中的使用 Git 登入資料設定 HTTPS 使用者，以及 Git 文件中的登入資料儲存。
當您將映像推送到 Amazon ECR 儲存庫時，遇到 HTTP 403 或「沒有基本身分驗證登入資料」錯誤。	即使您已使用 aws ecr get-login-password 命令成功向 Docker 進行身分驗證，也可能會從 docker push 或 docker pull 命令中遇到這些錯誤訊息。 get-login-password 已知原因包括： • 您已向不同的 區域進行身分驗證。如需詳細資訊，請參閱 Amazon ECR 文件中的私有登錄檔身分驗證。 • 您已驗證將 推送到您無權使用的儲存庫。如需詳細資訊，請參閱 Amazon ECR 文件中的私有儲存庫政策。 • 您的字符已過期。使用 `GetAuthorizationToken` 操作取得字符的預設過期期間為 12 小時。

相關資源

- 識別儲存庫之間重複容器映像的自動化解決方案 (GitHub 儲存庫)
- Amazon ECR 公有圖庫
- Amazon ECR 中的私有映像 (Amazon ECR 文件)
- AWS::CodePipeline::Pipeline 資源 (CloudFormation 文件)
- OCI 影像格式規格

其他資訊

Amazon ECR 公有儲存庫中映像的 Docker 檢查輸出

```
[
```

```
    {
        "Id":
 "sha256:f7cee5e1af28ad4e147589c474d399b12d9b551ef4c3e11e02d982fce5eebc68",
        "RepoTags": [
            "<account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest",
            "public.ecr.aws/amazonlinux/amazonlinux:2018.03"
        ],
        "RepoDigests": [
            "<account-id>.dkr.ecr.us-east-1.amazonaws.com/
test_ecr_repository@sha256:52db9000073d93b9bdee6a7246a68c35a741aaade05a8f4febba0bf795cdac02",
            "public.ecr.aws/amazonlinux/
amazonlinux@sha256:f972d24199508c52de7ad37a298bda35d8a1bd7df158149b381c03f6c6e363b5"
        ],
        "Parent": "",
        "Comment": "",
        "Created": "2023-02-23T06:20:11.575053226Z",
        "Container":
 "ec7f2fc7d2b6a382384061247ef603e7d647d65f5cd4fa397a3ccbba9278367c",
        "ContainerConfig": {
            "Hostname": "ec7f2fc7d2b6",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
            ],
            "Cmd": [
                "/bin/sh",
                "-c",
                "#(nop) ",
                "CMD [\"/bin/bash\"]"
            ],
            "Image":
 "sha256:c1bced1b5a65681e1e0e52d0a6ad17aaf76606149492ca0bf519a466ecb21e51",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": {}
```

```
        },
        "DockerVersion": "20.10.17",
        "Author": "",
        "Config": {
            "Hostname": "",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
            ],
            "Cmd": [
                "/bin/bash"
            ],
            "Image":
 "sha256:c1bced1b5a65681e1e0e52d0a6ad17aaf76606149492ca0bf519a466ecb21e51",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": null
        },
        "Architecture": "amd64",
        "Os": "linux",
        "Size": 167436755,
        "VirtualSize": 167436755,
        "GraphDriver": {
            "Data": {
                "MergedDir": "/var/lib/docker/overlay2/
c2c2351a82b26cbdf7782507500e5adb5c2b3a2875bdbba79788a4b27cd6a913/merged",
                "UpperDir": "/var/lib/docker/overlay2/
c2c2351a82b26cbdf7782507500e5adb5c2b3a2875bdbba79788a4b27cd6a913/diff",
                "WorkDir": "/var/lib/docker/overlay2/
c2c2351a82b26cbdf7782507500e5adb5c2b3a2875bdbba79788a4b27cd6a913/work"
            },
            "Name": "overlay2"
        },
        "RootFS": {
            "Type": "layers",
```

```
            "Layers": [

 "sha256:d5655967c2c4e8d68f8ec7cf753218938669e6c16ac1324303c073c736a2e2a2"
            ]
        },
        "Metadata": {
            "LastTagTime": "2023-03-02T10:28:47.142155987Z"
        }
    }
]
```

Amazon ECR 私有儲存庫中映像的 Docker 檢查輸出

```
[
    {
        "Id":
 "sha256:f7cee5e1af28ad4e147589c474d399b12d9b551ef4c3e11e02d982fce5eebc68",
        "RepoTags": [
            "<account-id>.dkr.ecr.us-east-1.amazonaws.com/test_ecr_repository:latest",
            "public.ecr.aws/amazonlinux/amazonlinux:2018.03"
        ],
        "RepoDigests": [
            "<account-id>.dkr.ecr.us-east-1.amazonaws.com/
test_ecr_repository@sha256:52db9000073d93b9bdee6a7246a68c35a741aaade05a8f4febba0bf795cdac02",
            "public.ecr.aws/amazonlinux/
amazonlinux@sha256:f972d24199508c52de7ad37a298bda35d8a1bd7df158149b381c03f6c6e363b5"
        ],
        "Parent": "",
        "Comment": "",
        "Created": "2023-02-23T06:20:11.575053226Z",
        "Container":
 "ec7f2fc7d2b6a382384061247ef603e7d647d65f5cd4fa397a3ccbba9278367c",
        "ContainerConfig": {
            "Hostname": "ec7f2fc7d2b6",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
```

```
                "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
            ],
            "Cmd": [
                "/bin/sh",
                "-c",
                "#(nop) ",
                "CMD [\"/bin/bash\"]"
            ],
            "Image":
"sha256:c1bced1b5a65681e1e0e52d0a6ad17aaf76606149492ca0bf519a466ecb21e51",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": {}
        },
        "DockerVersion": "20.10.17",
        "Author": "",
        "Config": {
            "Hostname": "",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
            ],
            "Cmd": [
                "/bin/bash"
            ],
            "Image":
"sha256:c1bced1b5a65681e1e0e52d0a6ad17aaf76606149492ca0bf519a466ecb21e51",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": null
        },
        "Architecture": "amd64",
        "Os": "linux",
```

```
        "Size": 167436755,
        "VirtualSize": 167436755,
        "GraphDriver": {
            "Data": {
                "MergedDir": "/var/lib/docker/overlay2/
c2c2351a82b26cbdf7782507500e5adb5c2b3a2875bdbba79788a4b27cd6a913/merged",
                "UpperDir": "/var/lib/docker/overlay2/
c2c2351a82b26cbdf7782507500e5adb5c2b3a2875bdbba79788a4b27cd6a913/diff",
                "WorkDir": "/var/lib/docker/overlay2/
c2c2351a82b26cbdf7782507500e5adb5c2b3a2875bdbba79788a4b27cd6a913/work"
            },
            "Name": "overlay2"
        },
        "RootFS": {
            "Type": "layers",
            "Layers": [

 "sha256:d5655967c2c4e8d68f8ec7cf753218938669e6c16ac1324303c073c736a2e2a2"
            ]
        },
        "Metadata": {
            "LastTagTime": "2023-03-02T10:28:47.142155987Z"
        }
    }
]
```

使用 Kubernetes DaemonSet 在 Amazon EKS 工作者節點上安裝 SSM Agent

Mahendra Revanasiddappa，Amazon Web Services

總結

請注意，2021 年 9 月：最新的 Amazon EKS 最佳化 AMIs 會自動安裝 SSM 代理程式。如需詳細資訊，請參閱 2021 年 6 月 AMIs版本備註。

在 Amazon Elastic Kubernetes Service (Amazon EKS) 中，基於安全準則，工作者節點不會連接 Secure Shell (SSH) 金鑰對。此模式顯示如何使用 Kubernetes DaemonSet 資源類型在所有工作者節點上安裝 AWS Systems Manager Agent (SSM Agent)，而不是手動安裝或取代節點的 Amazon Machine Image (AMI)。DaemonSet 在工作者節點上使用 Cron 任務來排程 SSM Agent 的安裝。您也可以使用此模式在工作者節點上安裝其他套件。

當您對叢集中的問題進行疑難排解時，安裝隨需 SSM Agent 可讓您使用工作者節點建立 SSH 工作階段、收集日誌或查看執行個體組態，而不需要 SSH 金鑰對。

先決條件和限制

先決條件

- 具有 Amazon Elastic Compute Cloud (Amazon EC2) 工作者節點的現有 Amazon EKS 叢集。
- 容器執行個體應具備與 SSM 服務通訊所需的許可。AWS Identity and Access Management (IAM) 受管角色 AmazonSSMManagedInstanceCore 提供 SSM Agent 在 EC2 執行個體上執行的必要許可。如需詳細資訊，請參閱 AWS Systems Manager 文件。

限制

- 此模式不適用於 AWS Fargate，因為 Fargate 平台不支援 DaemonSets。
- 此模式僅適用於 Linux 型工作者節點。
- DaemonSet Pod 會以特殊權限模式執行。如果 Amazon EKS 叢集有一個 Webhook，以特殊權限模式封鎖 Pod，則不會安裝 SSM Agent。

Architecture

下圖說明此模式的架構。

工具

工具

- kubectl 是一種命令列公用程式，用於與 Amazon EKS 叢集互動。此模式使用 `kubectl` 在 Amazon EKS 叢集上部署 DaemonSet，這會在所有工作者節點上安裝 SSM Agent。
- Amazon EKS 可讓您在 AWS 上輕鬆執行 Kubernetes，而無需安裝、操作和維護您自己的 Kubernetes 控制平面或節點。Kubernetes 是一套開放原始碼系統，用於容器化應用程式的自動化部署、擴展與管理。
- AWS Systems Manager Session Manager 可讓您透過互動式、一鍵式瀏覽器型 shell 或透過 AWS Command Line Interface (AWS CLI) 來管理您的 EC2 執行個體、內部部署執行個體和虛擬機器 (VMs)。

Code

使用下列程式碼來建立 DaemonSet 組態檔案，該檔案將在 Amazon EKS 叢集上安裝 SSM Agent。請遵循 Epics 區段中的指示。

```
cat << EOF > ssm_daemonset.yaml
apiVersion: apps/v1
kind: DaemonSet
metadata:
  labels:
    k8s-app: ssm-installer
  name: ssm-installer
  namespace: kube-system
spec:
  selector:
    matchLabels:
      k8s-app: ssm-installer
  template:
    metadata:
      labels:
        k8s-app: ssm-installer
    spec:
      containers:
      - name: sleeper
        image: busybox
```

```
        command: ['sh', '-c', 'echo I keep things running! && sleep 3600']
      initContainers:
      - image: amazonlinux
        imagePullPolicy: Always
        name: ssm
        command: ["/bin/bash"]
        args: ["-c","echo '* * * * * root yum install -y https://s3.amazonaws.com/
ec2-downloads-windows/SSMAgent/latest/linux_amd64/amazon-ssm-agent.rpm & rm -rf /etc/
cron.d/ssmstart' > /etc/cron.d/ssmstart"]
        securityContext:
          allowPrivilegeEscalation: true
        volumeMounts:
        - mountPath: /etc/cron.d
          name: cronfile
        terminationMessagePath: /dev/termination-log
        terminationMessagePolicy: File
      volumes:
      - name: cronfile
        hostPath:
          path: /etc/cron.d
          type: Directory
      dnsPolicy: ClusterFirst
      restartPolicy: Always
      schedulerName: default-scheduler
      terminationGracePeriodSeconds: 30
EOF
```

史詩

設定 kubectl

任務	Description	所需的技能
安裝並設定 kubectl 以存取 EKS 叢集。	如果 `kubectl` 尚未安裝並設定為存取 Amazon EKS 叢集，請參閱 Amazon EKS 文件中的安裝 kubectl。	DevOps

部署 DaemonSet

任務	Description	所需的技能
建立 DaemonSet 組態檔案。	在此模式稍早的 Codesection 中使用程式碼來建立名為 的 DaemonSet 組態檔案`ssm_daemonset.yaml` ，該檔案將部署到 Amazon EKS 叢集。 DaemonSet 啟動的 Pod 具有主要容器和 `init`容器。主要容器具有 `sleep`命令。`init` 容器包含 `command`區段，可建立 cron 任務檔案，以在路徑 上安裝 SSM Agent`/etc/cron.d/` 。Cron 任務只會執行一次，並在任務完成後自動刪除其建立的檔案。 當初始化容器完成時，主要容器會等待 60 分鐘再結束。60 分鐘後，會啟動新的 Pod。如果缺少 SSM Agent，此 Pod 會安裝 SSM Agent，或將 SSM Agent 更新為最新版本。 如有需要，您可以修改 `sleep`命令，每天重新啟動一次 Pod 或更頻繁地執行。	DevOps
在 Amazon EKS 叢集上部署 DaemonSet。	若要在 Amazon EKS 叢集上部署您在上一個步驟中建立的 DaemonSet 組態檔案，請使用下列命令：	DevOps

任務	Description	所需的技能
	`kubectl apply -f ssm_daemonset.yaml` 此命令會建立 DaemonSet，以在工作者節點上執行 Pod，以安裝 SSM Agent。	

相關資源

- 安裝 kubectl (Amazon EKS 文件)
- 設定 Session Manager (AWS Systems Manager 文件)

使用 preBootstrapCommands 在 Amazon EKS 工作者節點上安裝 SSM 代理程式和 CloudWatch 代理程式

Akamahadevi hiremath，Amazon Web Services

總結

此模式提供程式碼範例，以及在 Amazon EKS 叢集建立期間，在 Amazon Web Services (AWS) 雲端的 Amazon Elastic Kubernetes Service (Amazon EKS) 工作者節點上安裝 AWS Systems Manager Agent (SSM Agent) 和 Amazon CloudWatch 代理程式的步驟。 Amazon CloudWatch 您可以使用`eksctl`組態檔案結構描述 (Weaveworks 文件） 中的 `preBootstrapCommands` 屬性來安裝 SSM Agent 和 CloudWatch 代理程式。然後，您可以使用 SSM Agent 連線到工作者節點，而無需使用 Amazon Elastic Compute Cloud (Amazon EC2) 金鑰對。此外，您可以使用 CloudWatch 代理程式來監控 Amazon EKS 工作者節點上的記憶體和磁碟使用率。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在 macOS、Linux 或 Windows 上安裝和設定的 eksctl 命令列公用程式
- 在 macOS、Linux 或 Windows 上安裝和設定的 kubectl 命令列公用程式

限制

- 建議您避免將長時間執行的指令碼新增至 `preBootstrapCommands`屬性，因為這會延遲節點在擴展活動期間加入 Amazon EKS 叢集。我們建議您改為建立自訂 Amazon Machine Image (AMI)。
- 此模式僅適用於 Amazon EC2 Linux 執行個體。

Architecture

技術堆疊

- Amazon CloudWatch
- Amazon Elastic Kubernetes Service (Amazon EKS)
- AWS Systems Manager 參數存放區

目標架構

下圖顯示使用者使用 SSM Agent 連線至 Amazon EKS 工作者節點的範例，該 SSM Agent 是使用 安裝。 `preBootstrapCommands`

該圖顯示以下工作流程：

1. 使用者使用 `eksctl`組態檔案搭配 `preBootstrapCommands` 屬性來建立 Amazon EKS 叢集，這會安裝 SSM 代理程式和 CloudWatch 代理程式。
2. 任何稍後因擴展活動而加入叢集的新執行個體都會使用預先安裝的 SSM Agent 和 CloudWatch 代理程式建立。
3. 使用者使用 SSM 代理程式連線至 Amazon EC2，然後使用 CloudWatch 代理程式監控記憶體和磁碟使用率。

工具

- Amazon CloudWatch 可協助您即時監控 AWS 資源的指標，以及您在 AWS 上執行的應用程式。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 AWS 上執行 Kubernetes，而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- AWS Systems Manager 參數存放區為組態資料管理和秘密管理提供安全的階層式儲存。
- AWS Systems Manager Session Manager 可協助您透過互動式、一鍵式瀏覽器型 shell 或透過 AWS Command Line Interface (AWS CLI) 來管理 EC2 執行個體、內部部署執行個體和虛擬機器。
- eksctl 是一種命令列公用程式，用於在 Amazon EKS 上建立和管理 Kubernetes 叢集。
- kubectl 是與叢集 API 伺服器通訊的命令列公用程式。

史詩

建立 Amazon EKS 叢集

任務	Description	所需的技能
存放 CloudWatch 代理程式組態檔案。	將 CloudWatch 代理程式組態檔案存放在您要建立 Amazon EKS 叢集的 AWS 區域中的 AWS AWS Systems Manager	DevOps 工程師

任務	Description	所需的技能
	參數存放區中。若要這樣做，請在 AWS Systems Manager 參數存放區中建立參數，並記下參數的名稱（例如 `AmazonCloudwatch-linux`）。 如需詳細資訊，請參閱此模式額外資訊區段中的範例 CloudWatch 代理程式組態檔案程式碼。	
建立 eksctl 組態檔案和叢集。	1. 建立包含 CloudWatch 代理程式和 SSM 代理程式安裝步驟的`eksctl`組態檔案。如需詳細資訊，請參閱此模式額外資訊區段中的範例 eksctl 組態檔案程式碼。 2. 執行 `eksctl create cluster -f cluster.yaml` 命令來建立叢集。	AWS DevOps

驗證 SSM Agent 和 CloudWatch 代理程式是否正常運作

任務	Description	所需的技能
測試 SSM 代理程式。	透過使用 AWS Systems Manager 文件中開始工作階段中涵蓋的任何方法，使用 SSH 連線至 Amazon EKS 叢集節點。	AWS DevOps
測試 CloudWatch 代理程式。	使用 CloudWatch 主控台來驗證 CloudWatch 代理程式：	AWS DevOps

任務	Description	所需的技能
	1. 登入 AWS 管理主控台並開啟 CloudWatch 主控台。 2. 在導覽窗格中，展開指標，然後選擇所有指標。 3. 在瀏覽索引標籤的搜尋方塊中，輸入 ，然後選擇 CWAgent 指標以查看記憶體和磁碟指標。	

相關資源

- 在伺服器上安裝和執行 CloudWatch 代理程式 (Amazon CloudWatch 文件）
- 建立 Systems Manager 參數（主控台）(AWS Systems Manager 文件）
- 建立 CloudWatch 代理程式組態檔案 (Amazon CloudWatch 文件）
- 啟動工作階段 (AWS CLI) (AWS Systems Manager 文件）
- 啟動工作階段 (Amazon EC2 主控台）(AWS Systems Manager 文件）

其他資訊

CloudWatch 代理程式組態檔案範例

在下列範例中，CloudWatch 代理程式設定為監控 Amazon Linux 執行個體上的磁碟和記憶體使用率：

```
{
    "agent": {
        "metrics_collection_interval": 60,
        "run_as_user": "cwagent"
    },
    "metrics": {
        "append_dimensions": {
            "AutoScalingGroupName": "${aws:AutoScalingGroupName}",
            "ImageId": "${aws:ImageId}",
            "InstanceId": "${aws:InstanceId}",
            "InstanceType": "${aws:InstanceType}"
        },
        "metrics_collected": {
```

```
            "disk": {
                "measurement": [
                    "used_percent"
                ],
                "metrics_collection_interval": 60,
                "resources": [
                    "*"
                ]
            },
            "mem": {
                "measurement": [
                    "mem_used_percent"
                ],
                "metrics_collection_interval": 60
            }
        }
    }
}
```

範例 eksctl 組態檔案

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: test
  region: us-east-2
  version: "1.24"
managedNodeGroups:
  - name: test
    minSize: 2
    maxSize: 4
    desiredCapacity: 2
    volumeSize: 20
    instanceType: t3.medium
    preBootstrapCommands:
    - sudo yum install amazon-ssm-agent -y
    - sudo systemctl enable amazon-ssm-agent
    - sudo systemctl start amazon-ssm-agent
    - sudo yum install amazon-cloudwatch-agent -y
    - sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetch-
config -m ec2 -s -c ssm:AmazonCloudwatch-linux
    iam:
      attachPolicyARNs:
```

```
        - arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
        - arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
        - arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
        - arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy
        - arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
```

其他程式碼詳細資訊

- 在 preBootstrapCommands 屬性的最後一行中， AmazonCloudwatch-linux是 AWS System Manager 參數存放區中建立的參數名稱。您必須在建立 Amazon EKS 叢集的相同 AWS 區域中的參數存放區AmazonCloudwatch-linux中包含 。您也可以指定檔案路徑，但建議您使用 Systems Manager，以便於自動化和重複使用。
- 如果您在eksctl組態檔案中使用 ，您會preBootstrapCommands在 AWS 管理主控台中看到兩個啟動範本。第一個啟動範本包含 中指定的命令preBootstrapCommands。第二個範本包含 中指定的命令preBootstrapCommands和預設 Amazon EKS 使用者資料。需要此資料才能讓節點加入叢集。節點群組的 Auto Scaling 群組使用此使用者資料來啟動新的執行個體。
- 如果您在eksctl組態檔案中使用 iam 屬性，則必須列出預設 Amazon EKS 政策，以及您連接的 AWS Identity and Access Management (IAM) 政策中所需的任何其他政策。在建立 eksctl 組態檔案和叢集步驟的程式碼片段中，新增CloudWatchAgentServerPolicyAmazonSSMMangedInstanceCore了額外的政策，以確保 CloudWatch 代理程式和 SSM 代理程式如預期般運作。AmazonEKSWorkerNodePolicy、AmazonEKS_CNI_Policy、AmazonEC2ContainerRegistryReadOnly政策是 Amazon EKS 叢集正常運作所需的必要政策。

啟用 Amazon EKS Auto 模式時遷移 NGINX 傳入控制器

Olawale Olaleye 和 Shamanth Devagari，Amazon Web Services

總結

適用於 Amazon Elastic Kubernetes Service (Amazon EKS) 的 EKS Auto Mode 可減少在 Kubernetes 叢集上執行工作負載的操作開銷。此模式 AWS 也允許 代表您設定和管理基礎設施。在現有叢集上啟用 EKS Auto Mode 時，您必須仔細規劃 NGINX 傳入控制器組態的遷移。這是因為無法直接傳輸 Network Load Balancer。

在現有 Amazon EKS 叢集中啟用 EKS Auto Mode 時，您可以使用藍/綠部署策略來遷移 NGINX 傳入控制器執行個體。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 執行 Kubernetes 1.29 版或更新版本的 Amazon EKS 叢集
- 執行最低版本的 Amazon EKS 附加元件
- 最新版本的 kubectl
- 現有的 NGINX 傳入控制器執行個體
- （選用）Amazon Route 53 中用於 DNS 型流量轉移的託管區域

Architecture

藍/綠部署是一種部署策略，您可以在其中建立兩個不同但相同的環境。藍/綠部署提供近乎零的停機時間發佈和復原功能。基本概念是在執行不同應用程式版本的兩個相同環境之間轉移流量。

下圖顯示啟用 EKS Auto Mode 時，Network Load Balancer 從兩個不同的 NGINX 傳入控制器執行個體遷移。您可以使用藍/綠部署，在兩個 Network Load Balancer 之間轉移流量。

原始命名空間是藍色命名空間。在啟用 EKS Auto 模式之前，這是原始 NGINX 傳入控制器服務和執行個體執行的位置。原始服務和執行個體會連線至具有在 Route 53 中設定之 DNS 名稱的 Network Load Balancer。Load AWS Load Balancer 控制器將此 Network Load Balancer 部署在目標虛擬私有雲端 (VPC) 中。

下圖顯示下列工作流程，以設定藍/綠部署的環境：

1. 在不同的命名空間中安裝和設定另一個 NGINX 傳入控制器執行個體，即綠色命名空間。
2. 在 Route 53 中，設定新 Network Load Balancer 的 DNS 名稱。

工具

AWS 服務

- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 上執行 Kubernetes， AWS 而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- Elastic Load Balancing 會將傳入的應用程式或網路流量分散到多個目標。例如，您可以在一或多個可用區域中跨 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、容器和 IP 地址分配流量。
- Amazon Route 53 是一種可用性高、可擴展性強的 DNS Web 服務。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。

其他工具

- Helm 是 Kubernetes 的開放原始碼套件管理員，可協助您在 Kubernetes 叢集上安裝和管理應用程式。
- kubectl 是一種命令列界面，可協助您針對 Kubernetes 叢集執行命令。
- NGINX 輸入控制器透過請求處理、身分驗證、自助式自訂資源和偵錯來連接 Kubernetes 應用程式和服務。

史詩

檢閱現有環境

任務	Description	所需的技能
確認原始 NGINX 傳入控制器執行個體可運作。	輸入下列命令來驗證 `ingress-nginx` 命名空間中的資源是否可運作。如果您已在另一個命名空間中部署	DevOps 工程師

任務	Description	所需的技能

	NGINX 傳入控制器，請更新此命令中的命名空間名稱。
	`kubectl get all -n ingress-nginx`
	在輸出中，確認 NGINX 輸入控制器 Pod 處於執行中狀態。以下是輸出範例：
	`NAME READY STATUS RESTARTS AGE pod/ingress-nginx-admission-create-xqn9d 0/1 Completed 0 88m pod/ingress-nginx-admission-patch-lhk4j 0/1 Completed 1 88m pod/ingress-nginx-controller-68f68f859-xrz74 1/1 Running 2 (10m ago) 72m NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE service/ingress-nginx-controller LoadBalancer 10.100.67.255 k8s-`	

任務	Description	所需的技能
	ingressn-ingressn-abcdefg-12345.elb.eu-west-1.amazonaws.com 80:30330/TCP,443:31462/TCP 88m service/ingress-nginx-controller-admission ClusterIP 10.100.201.176 <none> 443/TCP 88m NAME READY UP-TO-DATE AVAILABLE AGE deployment.apps/ingress-nginx-controller 1/1 1 1 88m NAME DESIRED CURRENT READY AGE replicaset.apps/ingress-nginx-controller-68f68f859 1 1 1 72m replicaset.apps/ingress-nginx-controller-d8c96cf68 0 0 0 88m NAME	

任務	Description	所需的技能
	STATUS COMPLETIONS DURATION AGE job.batch/ingress-nginx-admission-create Complete 1/1 4s 88m job.batch/ingress-nginx-admission-patch Complete 1/1 5s 88m	

部署範例 HTTPd 工作負載以使用 NGINX 傳入控制器

任務	Description	所需的技能

建立 Kubernetes 資源。	輸入下列命令來建立範例 Kubernetes 部署、服務和輸入：
	`kubectl create deployment demo --image=httpd --port=80`
	`kubectl expose deployment demo`
	`kubectl create ingress demo --class=nginx \ --rule nginxautomode.local.dev/=demo:80`
		DevOps 工程師

檢閱部署的資源。	輸入下列命令以檢視已部署資源的清單： `kubectl get all,ingress`	DevOps 工程師

任務	Description	所需的技能

	在輸出中，確認範例 HTTPd Pod 處於執行中狀態。以下是輸出範例：
	
	`NAME READY STATUS RESTARTS AGE pod/demo-7d94f8cb4f-q68wc 1/1 Running 0 59m`
	
	`NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE service/demo ClusterIP 10.100.78.155 <none> 80/TCP 59m service/kubernetes ClusterIP 10.100.0.1 <none> 443/TCP 117m`
	
	`NAME READY UP-TO-DATE AVAILABLE AGE deployment.apps/demo 1/1 1 1 59m`
	
	`NAME DESIRED CURRENT READY AGE replicaset.apps/demo-7d94f8cb4f 1 1 1 59m`
	
	`NAME CLASS HOSTS`	

任務	Description	所需的技能

	<pre>ADDRESS
	
	PORTS
	AGE
	ingress.networ
	king.k8s.io/demo
	nginx nginxauto
	mode.local.dev
	k8s-ingre
	ssn-ingressn-abcde
	fg-12345.elb.eu-we
	st-1.amazonaws.com
	80 56m</pre>	

確認服務可連線。	輸入下列命令，確認可透過 Network Load Balancer 的 DNS 名稱存取服務：
	<pre>curl -H "Host: nginxauto
	mode.local.dev"
	http://k8s-ingress
	n-ingressn-abcdefg
	-12345.elb.eu-west
	-1.amazonaws.com</pre>
	以下是預期的輸出：
	<pre><html><body> It
	works! </body></
	html></pre>
		DevOps 工程師

任務	Description	所需的技能

（選用）建立 DNS 記錄。	1. 遵循使用 Amazon Route 53 主控台 (Route 53 文件）建立記錄中的指示，為設定的網域建立 DNS 記錄。
	2. 輸入下列命令，確認可透過設定的網域名稱存取服務：
	`curl "http://nginxautomode.local.dev/?[1-5]"`
	以下是預期的輸出：
	`<html><body><h1>It works!</h1></body></html>`
	`<html><body><h1>It works!</h1></body></html>`
	`<html><body><h1>It works!</h1></body></html>`
	`<html><body><h1>It works!</h1></body></html>`
	`<html><body><h1>It works!</h1></body></html>`
		DevOps 工程師，AWS DevOps

在現有叢集上啟用 EKS Auto 模式

任務	Description	所需的技能
啟用 EKS Auto 模式。	遵循在現有叢集上啟用 EKS Auto Mode 中的指示 (Amazon EKS 文件）。	AWS DevOps

安裝新的 NGINX 傳入控制器

任務	Description	所需的技能
設定新的 NGINX 傳入控制器執行個體。	1. 下載 deploy.yaml 範本。 2. 在您偏好的編輯器中開啟 deploy.yaml 範本。 3. 在 `kind: Namespace` 區段中，輸入命名空間的唯一名稱，例如 `ingress-nginx-v2` ： `apiVersion: v1 kind: Namespace metadata: labels: app.kubernetes.io/instance: ingress-nginx app.kubernetes.io/name: ingress-nginx name: ingress-nginx-v2` 4. 針對每個區段，將`namespace` 值更新為新名稱。 5. 在 `kind: Deployment` 區段中，執行下列動作： a. 輸入 的唯一值`--controller-class` ，例如 `k8s.io/ingress-nginx-v2` 。 b. 輸入 的唯一值`--ingress-class` ，例如 `nginx-v2`。	DevOps 工程師

任務	Description	所需的技能

	<pre>apiVersion: apps/v1
	kind: Deployment
	  name: ingress-n
	ginx-controller
	  namespace: ingress-
	nginx-v2
	...
	    spec:
	      containers:
	      - args:
	        - /nginx-in
	gress-controller
	        - --publish
	-service=$(POD_NAM
	ESPACE)/ingress-ng
	inx-controller
	        - --electio
	n-id=ingress-nginx-
	leader
	        - --control
	ler-class=k8s.io/i
	ngress-nginx-v2
	        - --ingress-
	class=nginx-v2</pre>	

	6. 在 kind: IngressCl
	ass 區段中，輸入--
	ingress-class 您在上
	一節中使用的 --control
	ler-class 和 的相同
	值：
	<pre>apiVersion: networkin
	g.k8s.io/v1
	kind: IngressClass
	metadata:
	  labels:</pre>	

任務	Description	所需的技能

	<pre>app.kuber
	netes.io/component:
	controller
	app.kuber
	netes.io/instance:
	ingress-nginx
	app.kuber
	netes.io/name:
	ingress-nginx
	app.kuber
	netes.io/part-of:
	ingress-nginx
	app.kuber
	netes.io/version:
	1.12.0
	name: nginx-v2
	spec:
	controller: k8s.io/
	ingress-nginx-v2</pre>	

	7. 在下一節中，新增 `loadBalancerClass: eks.amazonaws.com/nlb` 為 NGINX 傳入控制器執行個體佈建 Network Load Balancer：
	<pre>apiVersion: v1
	kind: Service
	metadata:
	name: ingress-n
	ginx-controller
	namespace: ingress-
	nginx-v2
	spec:
	...
	selector:
	app.kuber
	netes.io/component:
	controller</pre>	

任務	Description	所需的技能

	<pre>app.kuber
	netes.io/instance:
	ingress-nginx
	app.kuber
	netes.io/name:
	ingress-nginx
	type: LoadBalancer
	loadBalancerClass:
	eks.amazonaws.com/
	nlb</pre>
	8. 儲存並關閉 deploy.yaml 範本。	

部署新的 NGINX 執行個體控制器執行個體。	輸入下列命令以套用修改的資訊清單檔案：
	<pre>kubectl apply -f
	deploy.yaml</pre>
		DevOps 工程師

任務	Description	所需的技能

確認部署成功。	輸入下列命令來驗證 `ingress-nginx-v2` 命名空間中的資源是否正常運作：
	`kubectl get all -n ingress-nginx-v2`
	在輸出中，確認 NGINX 輸入控制器 Pod 處於執行中狀態。以下是輸出範例：
	`NAME READY STATUS RESTARTS AGE pod/ingress-nginx-admission-create-7shrj 0/1 Completed 0 24s pod/ingress-nginx-admission-patch-vkxr5 0/1 Completed 1 24s pod/ingress-nginx-controller-757bfcbc6d-4fw52 1/1 Running 0 24s NAME TYPE CLUSTER-IP EXTERNAL-IP`
		DevOps 工程師

任務	Description	所需的技能
	PORT(S) AGE service/ingress- nginx-controller LoadBalancer 10.100.208.114 k8s-ingressn-ingre ssn-2e5e37fab6-848 337cd9c9d520f.elb.eu- west-1.amazonaws.com 80:31469/TCP,443:3 0658/TCP 24s service/ingress- nginx-controller- admission ClusterIP 10.100.150.114 <none> 443/TCP 24s NAME READY UP-TO-DATE AVAILABLE AGE deployment.apps/ ingress-nginx-co ntroller 1/1 1 1 24s NAME DESIRED CURRENT READY AGE replicaset.apps/ ingress-nginx-co ntroller-757bfcbc6d 1 1 1 24s	

任務	Description	所需的技能
	`NAME   STATUS COMPLETIONS  DURATION AGE job.batch/ingress- nginx-admission-create  Complete 1/1  4s 24s job.batch/ingress- nginx-admission-patch  Complete 1/1  5s 24s`	
為範例 HTTPd 工作負載建立新的輸入。	輸入下列命令，為現有的範例 HTTPd 工作負載建立新的輸入： `kubectl create ingress  demo-new --class=n ginx-v2 \  --rule nginxauto mode.local.dev/=de mo:80`	DevOps 工程師

任務	Description	所需的技能

確認新的輸入正常運作。	輸入下列命令來確認新的輸入是否正常運作：
	`curl -H "Host: nginxautomode.local.dev" k8s-ingressn-ingressn-2e5e37fab6-848337cd9c9d520f.elb.eu-west-1.amazonaws.com`
	以下是預期的輸出：
	`<html><body><h1>It works!</h1></body></html>`
		DevOps 工程師

剪下

任務	Description	所需的技能
切換到新的命名空間。	1. （選用）遵循編輯記錄 (Route 53 文件）中的指示來更新 DNS 記錄。 2. 當您確認新的 NGINX 傳入控制器執行個體如預期般運作時，請刪除原始執行個體。 3. 刪除自我管理 AWS Load Balancer控制器。如需說明，請參閱從已棄用的 ALB 傳入控制器遷移應用程式 (Amazon EKS 文件）。	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	4. 耗盡受管節點群組。如需說明，請參閱刪除和耗盡節點群組 (eksctl 文件）。	

檢閱兩個輸入。	輸入下列命令來檢閱為範例 HTTPd 工作負載建立的兩個輸入：
	`kubectl get ingress`
	以下是輸出範例：
	`NAME CLASS HOSTS ADDRESS PORTS AGE`
	`demo nginx nginxautomode.local.dev k8s-ingressn-ingressn-abcdefg-12345.elb.eu-west-1.amazonaws.com 80 95m`
	`demo-new nginx-v2 nginxautomode.local.dev k8s-ingressn-ingressn-2e5e37fab6-848337cd9c9d520f.elb.eu-west-1.amazonaws.com 80 33s`
		DevOps 工程師

相關資源

- 在現有叢集上啟用 EKS Auto Mode (Amazon EKS 文件）
- 對 Amazon EKS (re：Post 知識中心） 中 Kubernetes 服務控制器建立的負載平衡器進行故障診斷AWS
- NGINX 傳入控制器 (NGINX 文件）

將您的容器工作負載從 Azure Red Hat OpenShift (ARO) 遷移至 Red Hat OpenShift Service on AWS (ROSA)

Naveen Ramasamy、Srikanth Rangavajhala 和 Gireesh Sreekantan，Amazon Web Services

總結

此模式提供step-by-step說明。 OpenShift Red Hat OpenShift Service on AWS ROSA 是由 Red Hat 與 合作提供的受管 Kubernetes 服務 AWS。它可協助您使用 Kubernetes 平台部署、管理和擴展容器化應用程式，並受益於 Red Hat 在 Kubernetes 和 AWS 雲端 基礎設施方面的專業知識。

從 ARO、其他雲端或內部部署遷移容器工作負載到 ROSA 需要將應用程式、組態和資料從一個平台傳輸到另一個平台。此模式有助於確保順利轉換，同時最佳化 AWS 雲端 服務、安全性和成本效益。它涵蓋兩種將工作負載遷移至 ROSA 叢集的方法：CI/CD 和 Migration Toolkit for Containers (MTC)。

此模式涵蓋這兩種方法。您選擇的方法取決於遷移程序的複雜性和確定性。如果您完全控制應用程式的狀態，並且可以透過管道保證一致的設定，我們建議您使用 CI/CD 方法。不過，如果您的應用程式狀態涉及不確定性、不可預見的變更或複雜的生態系統，建議您使用 MTC 做為可靠且受控制的路徑，將應用程式及其資料遷移至新的叢集。如需這兩種方法的詳細比較，請參閱其他資訊一節。

遷移至 ROSA 的優點：

- ROSA 與 無縫整合 AWS 為原生服務。您可以透過 輕鬆存取， AWS 管理主控台 並透過單一 計費 AWS 帳戶。它提供與其他 的完整相容性 AWS 服務 ，並提供來自 AWS 和 Red Hat 的協作支援。
- ROSA 支援混合多雲端部署。它可讓應用程式在內部部署資料中心和多個雲端環境中一致地執行。
- ROSA 受益於 Red Hat 的安全重點，並提供角色型存取控制 (RBAC)、映像掃描和漏洞評估等功能，以確保安全的容器環境。
- ROSA 旨在輕鬆擴展應用程式，並提供高可用性選項。它允許應用程式視需要成長，同時維持可靠性。
- 相較於手動設定和管理方法，ROSA 可自動化並簡化 Kubernetes 叢集的部署。這可加速開發和部署程序。
- ROSA 受益於 AWS 雲端 服務，並提供與資料庫服務、儲存解決方案和安全服務等 AWS 方案的無縫整合。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 為 AWS 服務 該 ROSA 設定的許可依賴 來交付功能。如需詳細資訊，請參閱 ROSA 文件中的先決條件。
- ROSA 主控台上已啟用 ROSA。如需說明，請參閱 ROSA 文件。
- 安裝並設定 ROSA 叢集。如需詳細資訊，請參閱 ROSA 文件中的 ROSA 入門。若要了解設定 ROSA 叢集的不同方法，請參閱 AWS 規範指引指南 ROSA 實作策略。
- 從內部部署網路到 AWS 透過 AWS Direct Connect（偏好） 或 AWS Virtual Private Network (Site-to-Site VPN) 建立的網路連線。
- Amazon Elastic Compute Cloud (Amazon EC2) 執行個體或其他虛擬伺服器，用於安裝工具，例如 `aws client`、OpenShift CLI (oc) 用戶端、ROSA 用戶端和 Git 二進位檔。

CI/CD 方法的其他先決條件：

- 存取內部部署 Jenkins 伺服器，具有建立新管道、新增階段、新增 OpenShift 叢集和執行建置的許可。
- 存取維護應用程式原始碼的 Git 儲存庫，具有建立新 Git 分支和執行遞交至新分支的許可。

MTC 方法的其他先決條件：

- Amazon Simple Storage Service (Amazon S3) 儲存貯體，將用作複寫儲存庫。
- 來源 ARO 叢集的管理存取權。這是設定 MTC 連線的必要條件。

限制

- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

Architecture

ROSA 提供三種網路部署模式：公有、私有和 AWS PrivateLink.PrivateLinkenables Red Hat 網站可靠性工程 (SRE) 團隊，使用連線至現有 VPC 中叢集 PrivateLink 端點的私有子網路來管理叢集。

選擇 PrivateLinkoption 可提供最安全的組態。因此，我們建議將其用於敏感工作負載或嚴格的合規要求。如需公有和私有網路部署選項的相關資訊，請參閱 Red Hat OpenShift 文件。

> **Important**
>
> 您只能在安裝時建立 PrivateLink 叢集。安裝後，您無法將叢集變更為使用 PrivateLink。

下圖說明 ROSA 叢集的 PrivateLink 架構，該叢集使用 Direct Connect 連線到內部部署和 ARO 環境。

AWS ROSA 的 許可

對於 ROSA 的 AWS 許可，我們建議您使用 AWS Security Token Service (AWS STS) 搭配短期動態字符。此方法使用最低權限的預先定義角色和政策，授予 ROSA 在 中操作的最小許可 AWS 帳戶，並支援 ROSA 安裝、控制平面和運算功能。

CI/CD 管道重新部署

對於具有成熟 CI/CD 管道的使用者，CI/CD 管道重新部署是建議的方法。選擇此選項時，您可以使用任何 DevOps 部署策略，逐步將應用程式負載轉移到 ROSA 上的部署。

> **Note**
>
> 此模式假設您有一個常見的使用案例，其中您有現場部署 Git、JFrog Artifactory 和 Jenkins 管道。此方法要求您建立從內部部署網路到 AWS 的網路連線 Direct Connect，並在遵循 Epics 區段中的指示之前設定 ROSA 叢集。如需詳細資訊，請參閱先決條件一節。

下圖顯示此方法的工作流程。

MTC 方法

您可以使用 Migration Toolkit for Containers (MTC)，在不同 Kubernetes 環境之間遷移容器化工作負載，例如從 ARO 遷移至 ROSA。MTC 透過自動化數個關鍵任務並提供管理遷移生命週期的全方位架構，簡化遷移程序。

下圖顯示此方法的工作流程。

工具

AWS 服務

- AWS DataSync 是一種線上資料傳輸和探索服務，可協助您在 AWS 儲存服務之間來回移動檔案或物件資料。
- AWS Direct Connect 會透過標準乙太網路光纖纜線將您的內部網路連結至某個 Direct Connect 位置。透過此連線，您可以在網路路徑中略過網際網路服務提供者 AWS 服務 時，直接建立虛擬介面至公有。
- AWS PrivateLink 可協助您建立從虛擬私有雲端 (VPCs) 到 VPC 外部服務的單向私有連線。
- Red Hat OpenShift Service on AWS (ROSA) 是一種受管服務，可協助 Red Hat OpenShift 使用者建置、擴展和管理容器化應用程式 AWS。
- AWS Security Token Service (AWS STS) 可協助您為使用者請求暫時、有限權限的登入資料。

其他工具

- Migration Toolkit for Containers (MTC) 提供主控台和 API，可將容器化應用程式從 ARO 遷移至 ROSA。

最佳實務

- 針對彈性和如果您有安全合規工作負載，請設定使用 PrivateLink 的多可用區域 ROSA 叢集。如需詳細資訊，請參閱 ROSA 文件。

> **Note**
>
> 無法在安裝後設定 PrivateLink。

- 您用於複寫儲存庫的 S3 儲存貯體不應公開。使用適當的 S3 儲存貯體政策來限制存取。
- 如果您選擇 MTC 方法，請使用階段遷移選項來減少切換期間的停機時間時段。
- 在您佈建 ROSA 叢集之前和之後，檢閱您的服務配額。如有必要，請根據您的需求請求提高配額。如需詳細資訊，請參閱服務配額文件。
- 檢閱 ROSA 安全準則並實作安全最佳實務。
- 建議您在安裝後移除預設叢集管理員。如需詳細資訊，請參閱 Red Hat OpenShift 文件。
- 使用機器集區自動擴展來縮減 ROSA 叢集中未使用的工作者節點，以最佳化成本。如需詳細資訊，請參閱 ROSA 研討會。
- 使用適用於 OpenShift Container Platform 的 Red Hat Cost Management 服務，更深入了解和追蹤雲端和容器的成本。如需詳細資訊，請參閱 ROSA 研討會。

- 使用 監控和稽核 ROSA 叢集基礎設施服務和應用程式 AWS 服務。如需詳細資訊，請參閱 ROSA 研討會。

史詩

選項 1：使用 CI/CD 管道

任務	Description	所需的技能
將新的 ROSA 叢集新增至 Jenkins。	1. 在 Jenkins 主控台上，選擇管理 Jenkins、設定系統。 2. 在管理 Jenkins 頁面的 OpenShift 外掛程式區段中，選擇新增叢集。 3. 提供必要的資訊，例如叢集名稱、API 伺服器 URL 和字符資訊，以驗證 ROSA。	AWS 管理員、AWS 系統管理員、AWS DevOps
將oc用戶端新增至 Jenkins 節點。	1. 在 Jenkins 主控台上，選擇管理 Jenkins、全域工具組態。 2. 在 OpenShift 用戶端工具區段中，安裝與您的 ROSA 叢集版本相同的 OpenShift CLI (oc) 用戶端版本。	AWS 管理員、AWS 系統管理員、AWS DevOps
建立新的 Git 分支。	在適用於 的 Git 儲存庫中建立新的分支`rosa-dev`。此分支會將 ROSA 的程式碼或組態參數變更與您現有的管道分開。	AWS DevOps
標記 ROSA 的影像。	在建置階段中，使用不同的標籤來識別從 ROSA 管道建置的映像。	AWS 管理員、AWS 系統管理員、AWS DevOps

任務	Description	所需的技能
建立管道。	建立新的 Jenkins 管道，其與您現有的管道類似。對於此管道，請使用您先前建立的 `rosa-dev` Git 分支，並確保包含與您現有管道相同的 Git 結帳、程式碼掃描和建置階段。	AWS 管理員、AWS 系統管理員、AWS DevOps
新增 ROSA 部署階段。	在新的管道中，新增要部署到 ROSA 叢集的階段，並參考您新增至 Jenkins 全域組態的 ROSA 叢集。	AWS 管理員、AWS DevOps、AWS 系統管理員
啟動新的組建。	在 Jenkins 中，選取您的管道，然後選擇立即建置，或透過將變更遞交至 Git 中的`rosa-dev`分支來啟動新建置。	AWS 管理員、AWS DevOps、AWS 系統管理員
驗證部署。	使用 oc 命令或 ROSA 主控台來驗證應用程式是否已部署在您的目標 ROSA 叢集上。	AWS 管理員、AWS DevOps、AWS 系統管理員
將資料複製到目標叢集。	對於具狀態工作負載，請使用 AWS DataSync 或開放原始碼公用程式，例如 rsync，將資料從來源叢集複製到目標叢集，或者您可以使用 MTC 方法。如需詳細資訊，請參閱 AWS DataSync 文件。	AWS 管理員、AWS DevOps、AWS 系統管理員

任務	Description	所需的技能
測試您的應用程式。	1. 使用 oc 路由命令或 ROSA 主控台擷取應用程式的路由 URL。 https://console.aws.amazon.com/rosa 2. 使用路由 URL 測試您的應用程式。	AWS 管理員、AWS DevOps、AWS 系統管理員
切換。	如果您的測試成功，請使用適當的 Amazon Route 53 政策，將流量從 ARO 託管應用程式移至 ROSA 託管應用程式。當您完成此步驟時，應用程式的工作負載將完全轉換為 ROSA 叢集。	AWS 管理員、AWS 系統管理員

選項 2：使用 MTC

任務	Description	所需的技能
安裝 MTC 運算子。	在 ARO 和 ROSA 叢集上安裝 MTC 運算子： 1. 在 ARO 或 ROSA 主控台中，導覽至 Operators、OperatorHub 頁面。 2. 在依關鍵字篩選方塊中，尋找或輸入 MTC。 3. 選取 MTC 運算子以顯示其他資訊。 4. 閱讀有關 運算子的資訊後，請選擇安裝。	AWS 管理員、AWS DevOps、AWS 系統管理員
設定至複寫儲存庫的網路流量。	如果您使用的是代理伺服器，請將其設定為允許複寫儲存庫	AWS 管理員、AWS DevOps、AWS 系統管理員

任務	Description	所需的技能
	與叢集之間的網路流量。複寫儲存庫是 MTC 用來遷移資料的中繼儲存物件。在遷移期間，來源和目標叢集必須具有複寫儲存庫的網路存取權。	
將來源叢集新增至 MTC。	在 MTC Web 主控台上，新增 ARO 來源叢集。	AWS 管理員、AWS DevOps、AWS 系統管理員
新增 Amazon S3 做為複寫儲存庫。	在 MTC Web 主控台上，新增 Amazon S3 儲存貯體 （請參閱先決條件) 做為複寫儲存庫。	AWS 管理員、AWS DevOps、AWS 系統管理員
建立遷移計畫。	在 MTC Web 主控台上，建立遷移計畫，並將資料傳輸類型指定為複製。這將指示 MTC 將資料從來源 (ARO) 叢集複製到 S3 儲存貯體，以及從儲存貯體複製到目標 (ROSA) 叢集。	AWS 管理員、AWS DevOps、AWS 系統管理員

任務	Description	所需的技能
執行遷移計畫。	使用階段或切換選項執行遷移計劃： • 選擇階段以將資料複製到目標叢集，而不停止您的應用程式。您可以執行多個階段遷移，以在切換遷移之前複製大部分的資料。這可縮短切換遷移的持續時間。 • 選擇切換以停止來源叢集上的應用程式，同時將資源移至目標叢集。 若要防止停止應用程式，您可以在遷移期間清除來源叢集上的暫停交易核取方塊。	AWS 管理員、AWS DevOps、AWS 系統管理員

疑難排解

問題	解決方案
連線問題	當您將容器工作負載從 ARO 遷移至 ROSA 時，您可能會遇到連線問題，應加以解決，以確保成功遷移。為了在遷移期間解決這些連線問題（列於此表中），精細規劃、與網路和安全團隊的協調，以及徹底的測試至關重要。在每個步驟實作逐步遷移策略並驗證連線，有助於將潛在的中斷降至最低，並確保從 ARO 順利轉換至 ROSA。
網路組態差異	ARO 和 ROSA 的網路組態可能會有變化，例如虛擬網路 (VNet) 設定、子網路和網路政策。為了在服務之間進行適當的通訊，請確保網路設定在兩個平台之間保持一致。

問題	解決方案
安全群組和防火牆規則	ROSA 和 ARO 可能有不同的預設安全群組和防火牆設定。請務必調整和更新這些規則，以允許必要的流量在遷移期間維持容器和服務之間的連線。
IP 地址和 DNS 變更	當您遷移工作負載時，IP 地址和 DNS 名稱可能會變更。重新設定依賴靜態 IPs 或特定 DNS 名稱的應用程式。
外部服務存取	如果您的應用程式依賴資料庫或 APIs等外部服務，您可能需要更新其連線設定，以確保它們可以與 ROSA 的新服務通訊。
Azure Private Link 組態	如果您在 ARO 中使用 Azure Private Link 或私有端點服務，則需要在 ROSA 中設定同等功能，以確保資源之間的私有連線。
Site-to-Site VPN 或 Direct Connect 設定	如果您的內部部署網路與 ARO 之間存在現有 Site-to-Site VPN 或 Direct Connect 連線，您將需要與 ROSA 建立類似的連線，以便與本機資源進行不間斷的通訊。
輸入和負載平衡器設定	ARO 和 ROSA 之間的輸入控制器和負載平衡器組態可能不同。重新設定這些設定，以維護 服務的外部存取權。
憑證和 TLS 處理	如果您的應用程式使用 SSL 憑證或 TLS，請確定憑證在 ROSA 中有效且設定正確。
容器登錄檔存取	如果您的容器託管在外部容器登錄檔中，請為 ROSA 設定適當的身分驗證和存取許可。
監控和記錄	更新監控和記錄組態，以反映 ROSA 上的新基礎設施，以便您可以繼續有效監控容器的運作狀態和效能。

相關資源

AWS 參考

- 什麼是 Red Hat OpenShift Service on AWS？ (ROSA 文件）
- ROSA 入門 (ROSA 文件）
- Red Hat OpenShift Service on AWS 實作策略 (AWS 方案指引）
- Red Hat OpenShift Service on AWS 現在 GA (AWS 部落格文章）
- ROSA 研討會
- ROSA 常見問答集
- ROSA 研討會常見問答集
- ROSA 定價

Red Hat OpenShift 文件

- 在 上快速安裝叢集 AWS
- 在受限的網路 AWS 中於 上安裝叢集
- 在 上安裝叢集 AWS 到現有的 VPC
- AWS 使用 CloudFormation 範本在 的使用者佈建基礎設施上安裝叢集
- 使用使用者佈建的基礎設施在受限的網路 AWS 中安裝叢集
- AWS 使用自訂在 上安裝叢集
- OpenShift CLI 入門

其他資訊

在 MTC 和 CI/CD 管道重新部署選項之間進行選擇

將應用程式從一個 OpenShift 叢集遷移到另一個叢集需要仔細考慮。理想情況下，您會希望使用 CI/CD 管道重新部署應用程式並處理持久性磁碟區資料的遷移，以順利轉換。不過，實際上，叢集上執行中的應用程式容易隨著時間而發生不可預見的變更。這些變更可能會導致應用程式逐漸偏離其原始部署狀態。MTC 為命名空間的確切內容不確定，且將所有應用程式元件無縫遷移至新叢集的情況提供解決方案。

做出正確的選擇需要評估您的特定案例，並權衡每種方法的好處。透過這樣做，您可以確保成功且無縫的遷移符合您的需求和優先順序。以下是在兩個選項之間進行選擇的其他準則。

CI/CD 管道重新部署

如果您的應用程式可以使用管道放心地重新部署，建議使用 CI/CD 管道方法。這可確保遷移受到控制、可預測，並與現有的部署實務保持一致。選擇此方法時，您可以使用藍/綠部署或金絲雀部署策略，逐步將負載轉移到 ROSA 上的部署。在此案例中，此模式假設 Jenkins 正在從內部部署環境協調應用程式部署。

優點：

- 您不需要來源 ARO 叢集的管理存取權，也不需要在來源或目的地叢集上部署任何運算子。
- 此方法可協助您使用 DevOps 策略逐步切換流量。

缺點：

- 它需要更多精力來測試應用程式的功能。
- 如果您的應用程式包含持久性資料，則需要使用 AWS DataSync 或其他工具來複製資料的額外步驟。

MTC 遷移

在真實世界中，執行中的應用程式可能會發生非預期的變更，導致它們偏離初始部署。當您不確定來源叢集上應用程式的目前狀態時，請選擇 MTC 選項。例如，如果您的應用程式生態系統跨越各種元件、組態和資料儲存磁碟區，我們建議您選擇 MTC，以確保完整遷移涵蓋應用程式及其整個環境。

優點：

- MTC 提供工作負載的完整備份和還原。
- 遷移工作負載時，它會將持久性資料從來源複製到目標。
- 它不需要存取原始程式碼儲存庫。

缺點：

- 您需要管理權限，才能在來源和目的地叢集上安裝 MTC 運算子。
- DevOps 團隊需要訓練，才能使用 MTC 工具並執行遷移。

使用 Amazon ECS Anywhere 在 Amazon WorkSpaces 上執行 Amazon ECS 任務 Amazon ECS Anywhere

Akash Kumar，Amazon Web Services

總結

Amazon Elastic Container Service (Amazon ECS) Anywhere 支援在任何環境中部署 Amazon ECS 任務，包括 Amazon Web Services (AWS) 受管基礎設施和客戶受管基礎設施。您可以在使用在雲端中執行且隨時為最新狀態的全 AWS 受管控制平面時執行此操作。

企業通常會使用 Amazon WorkSpaces 來開發容器型應用程式。這需要具有 Amazon ECS 叢集的 Amazon Elastic Compute Cloud (Amazon EC2) 或 AWS Fargate，才能測試和執行 ECS 任務。現在，透過使用 Amazon ECS Anywhere，您可以將 Amazon WorkSpaces 直接新增為 ECS 叢集的外部執行個體，也可以直接執行任務。這可縮短您的開發時間，因為您可以使用 Amazon WorkSpaces 上的 ECS 叢集在本機測試容器。您也可以節省使用 EC2 或 Fargate 執行個體來測試容器應用程式的成本。

此模式示範如何使用 Amazon ECS Anywhere 在 Amazon WorkSpaces Amazon ECS Anywhere 任務。它設定 ECS 叢集，並使用 AWS Directory Service Simple AD 啟動 WorkSpaces。然後，範例 ECS 任務會在 WorkSpaces 中啟動 NGINX。

先決條件和限制

- 作用中的 AWS 帳戶
- AWS 命令列界面 (AWS CLI)
- 在您的機器上設定的 AWS 登入資料

Architecture

目標技術堆疊

- 虛擬私有雲端 (VPC)
- Amazon ECS 叢集
- Amazon WorkSpaces
- 具有 Simple AD 的 AWS Directory Service

目標架構

架構包含下列服務和資源：

- 自訂 VPC 中具有公有和私有子網路的 ECS 叢集
- VPC 中的簡易 AD 可讓使用者存取 Amazon WorkSpaces
- 使用 Simple AD 在 VPC 中佈建的 Amazon WorkSpaces
- 啟用 AWS Systems Manager 以新增 Amazon WorkSpaces 做為受管執行個體
- 使用 Amazon ECS 和 AWS Systems Manager Agent (SSM Agent)，Amazon WorkSpaces 已新增至 Systems Manager 和 ECS 叢集
- 要在 ECS 叢集的 WorkSpaces 中執行的範例 ECS 任務

工具

- AWS Directory Service Simple Active Directory (Simple AD) 是由 Samba 4 Active Directory 相容伺服器提供支援的獨立受管目錄。Simple AD 提供 AWS Managed Microsoft AD 提供的部分功能，包括能夠管理使用者並安全地連線至 Amazon EC2 執行個體。
- Amazon Elastic Container Service (Amazon ECS) 是快速、可擴展的容器管理服務，可協助您執行、停止和管理叢集上的容器。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Systems Manager 可協助您管理在 AWS 雲端中執行的應用程式和基礎設施。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。
- Amazon WorkSpaces 可協助您為使用者佈建以雲端為基礎的虛擬 Microsoft Windows 或 Amazon Linux 桌面，稱為 WorkSpaces。WorkSpaces 無需採購和部署硬體或安裝複雜軟體。

史詩

設定 ECS 叢集

任務	Description	所需的技能
建立和設定 ECS 叢集。	若要建立 ECS 叢集，請遵循 AWS 文件中的指示，包括下列步驟：	雲端架構師

任務	Description	所需的技能
	• 針對選取叢集相容性，請選擇僅限聯網，這將支援 Amazon WorkSpace 做為 ECS 叢集的外部執行個體。 • 選擇 以建立新的 VPC。	

啟動 Amazon WorkSpaces

任務	Description	所需的技能
設定 Simple AD 並啟動 Amazon WorkSpaces。	若要為新建立的 VPC 佈建 Simple AD 目錄並啟動 Amazon WorkSpaces，請遵循 AWS 文件中的指示。	雲端架構師

為混合環境設定 AWS Systems Manager

任務	Description	所需的技能
下載連接的指令碼。	在本機電腦上，下載附件區段中的 `ssm-trust-policy.json` 和 `ssm-activation.json` 檔案。	雲端架構師

新增 IAM 角色。	根據您的業務需求新增環境變數。
	`export AWS_DEFAULT_REGION=${AWS_REGION_ID}`
	`export ROLE_NAME=${ECS_TASK_ROLE}`
	`export CLUSTER_NAME=${ECS_CLUSTER_NAME}`
		雲端架構師

任務	Description	所需的技能

	`export SERVICE_NAME=${ECS_CLUSTER_SERVICE_NAME}`
	
	執行下列命令。
	
	`aws iam create-role --role-name $ROLE_NAME --assume-role-policy-document file://ssm-trust-policy.json`	

將 AmazonSSMManagedInstanceCore 政策新增至 IAM 角色。	執行下列命令。 `aws iam attach-role-policy --role-name $ROLE_NAME --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore`	雲端架構師
將 AmazonEC2ContainerServiceforEC2Role 政策新增至 IAM 角色。	執行下列命令。 `aws iam attach-role-policy --role-name $ROLE_NAME --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEC2ContainerServiceforEC2Role`	雲端架構師
驗證 IAM 角色。	若要驗證 IAM 角色，請執行下列命令。 `aws iam list-attached-role-policies --role-name $ROLE_NAME`	雲端架構師

任務	Description	所需的技能
啟用 Systems Manager。	執行下列命令。 `aws ssm create-activation --iam-role $ROLE_NAME | tee ssm-activation.json`	雲端架構師

將 WorkSpaces 新增至 ECS 叢集

任務	Description	所需的技能
連線至 WorkSpaces。	若要連線至 並設定工作區，請遵循 AWS 文件中的指示。	應用程式開發人員
下載 ecs-anywhere 安裝指令碼。	在命令提示中，執行下列 命令。 `curl -o "ecs-anywhere-install.sh" "https://amazon-ecs-agent-packages-preview.s3.us-east-1.amazonaws.com/ecs-anywhere-install.sh" && sudo chmod +x ecs-anywhere-install.sh`	應用程式開發人員
檢查 Shell 指令碼的完整性。	（選用）執行下列命令。 `curl -o "ecs-anywhere-install.sh.sha256" "https://amazon-ecs-agent-packages-preview.s3.us-east-1.amazonaws.com/ecs-anywhere-install.sh.sha256" &&`	應用程式開發人員

任務	Description	所需的技能
	`sha256sum -c ecs-anywhere-install.sh.sha256`	
在 Amazon Linux 上新增 EPEL 儲存庫。	若要新增適用於 Enterprise Linux (EPEL) 的額外套件儲存庫，請執行命令 `sudo amazon-linux-extras install epel -y`。	應用程式開發人員
安裝 Amazon ECS Anywhere。	若要執行安裝指令碼，請使用下列命令。 `sudo ./ecs-anywhere-install.sh --cluster $CLUSTER_NAME --activation-id $ACTIVATION_ID --activation-code $ACTIVATION_CODE --region $AWS_REGION`	
從 ECS 叢集檢查執行個體資訊。	若要檢查 Systems Manager 和 ECS 叢集執行個體資訊，並驗證是否已在叢集上新增 WorkSpaces，請從本機電腦執行下列命令。 `aws ssm describe-instance-information" && "aws ecs list-container-instances --cluster $CLUSTER_NAME`	應用程式開發人員

新增 WorkSpaces 的 ECS 任務

任務	Description	所需的技能
建立任務執行 IAM 角色。	external-task-definition.json 從附件區段下載 task-execution-assume-role.json 和 。 在本機電腦上，執行下列命令。 `aws iam --region $AWS_DEFAULT_REGION create-role --role-name $ECS_TASK_EXECUTION_ROLE --assume-role-policy-document file://task-execution-assume-role.json`	雲端架構師
將政策新增至執行角色。	執行下列命令。 `aws iam --region $AWS_DEFAULT_REGION attach-role-policy --role-name $ECS_TASK_EXECUTION_ROLE --policy-arn arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy`	雲端架構師
建立任務角色。	執行下列命令。 `aws iam --region $AWS_DEFAULT_REGION create-role --`	雲端架構師

任務	Description	所需的技能
	`role-name $ECS_TASK_EXECUTION_ROLE --assume-role-policy-document file://task-execution-assume-role.json`	
將任務定義註冊到叢集。	在本機電腦上，執行下列命令。 `aws ecs register-task-definition --cli-input-json file://external-task-definition.json`	雲端架構師
執行任務。	在本機電腦上，執行下列命令。 `aws ecs run-task --cluster $CLUSTER_NAME --launch-type EXTERNAL --task-definition nginx`	雲端架構師

任務	Description	所需的技能

驗證任務執行狀態。	若要擷取任務 ID，請執行下列命令。
	`export TEST_TASKID=$(aws ecs list-tasks --cluster $CLUSTER_NAME | jq -r '.taskArns[0]')`
	使用任務 ID，執行下列命令。
	`aws ecs describe-tasks --cluster $CLUSTER_NAME --tasks ${TEST_TASKID}`
		雲端架構師

驗證 WorkSpace 上的任務。	若要檢查 NGINX 是否在 WorkSpace 上執行，請執行 命令 `curl http://localhost:8080` 。	應用程式開發人員

相關資源

- ECS 叢集
- 設定混合環境
- Amazon WorkSpaces
- 簡易 AD

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

在 Amazon EC2 Linux 執行個體上執行 ASP.NET Core Web API Docker 容器

Vijai Anand Ramalingam 和 Sreelaxmi Pai，Amazon Web Services

總結

此模式適用於開始在 Amazon Web Services (AWS) 雲端上容器化其應用程式的人員。當您開始在雲端上容器化應用程式時，通常不會設定容器協同運作平台。此模式可協助您在 AWS 上快速設定基礎設施，以測試容器化應用程式，而不需要複雜的容器協同運作基礎設施。

現代化旅程的第一步是轉換應用程式。如果是舊版 .NET Framework 應用程式，您必須先將執行時間變更為 ASP.NET Core。然後執行下列動作：

- 建立 Docker 容器映像
- 使用建置的映像執行 Docker 容器
- 在任何容器協同運作平台上部署應用程式之前，請先驗證應用程式，例如 Amazon Elastic Container Service (Amazon ECS) 或 Amazon Elastic Kubernetes Service (Amazon EKS)。

此模式涵蓋 Amazon Elastic Compute Cloud (Amazon EC2) Linux 執行個體上現代應用程式開發的建置、執行和驗證層面。

先決條件和限制

先決條件

- 作用中的Amazon Web Services (AWS) 帳戶
- AnAWS Identity and Access Management (IAM) 角色具有足夠的存取權，可為此模式建立 AWS 資源
- Visual Studio Community 2022 或更新版本下載並安裝
- 將 .NET Framework 專案現代化為 ASP.NET Core
- GitHub 儲存庫

產品版本

- Visual Studio Community 2022 或更新版本

Architecture

目標架構

此模式使用 AWS CloudFormation 範本來建立高度可用的架構，如下圖所示。Amazon EC2 Linux 執行個體會在私有子網路中啟動。AWS Systems Manager Session Manager 用於存取私有 Amazon EC2 Linux 執行個體，以及測試在 Docker 容器中執行的 API。

1. 透過 Session Manager 存取 Linux 執行個體

工具

AWS 服務

- AWS 命令列界面 – AWS 命令列界面 (AWS CLI) 是一種開放原始碼工具，可透過命令列 shell 中的命令與 AWS 服務互動。透過最少的組態，您可以執行 AWS CLI 命令來實作相當於瀏覽器型 AWS 管理主控台所提供功能的功能。
- AWS 管理主控台 – AWS 管理主控台是一種 Web 應用程式，包含並參考各種用於管理 AWS 資源的服務主控台。若是首次登入，這時主控台頁面將會顯示。首頁提供每個服務主控台的存取權，並提供單一位置來存取執行 AWS 相關任務所需的資訊。
- AWS Systems Manager Session Manager – Session Manager 是全受管的 AWS Systems Manager 功能。使用 Session Manager，您可以管理 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。Session Manager 提供安全且可稽核的節點管理，無需開啟傳入連接埠、維護堡壘主機或管理 SSH 金鑰。

其他工具

- Visual Studio 2022 – Visual Studio 2022 是整合式開發環境 (IDE)。
- Docker – Docker 是一組平台即服務 (PaaS) 產品，可在作業系統層級使用虛擬化在容器中交付軟體。

Code

```
FROM mcr.microsoft.com/dotnet/aspnet:5.0 AS base
 WORKDIR /app
EXPOSE 80
```

```
EXPOSE 443

FROM mcr.microsoft.com/dotnet/sdk:5.0 AS build
WORKDIR /src
COPY ["DemoNetCoreWebAPI/DemoNetCoreWebAPI.csproj", "DemoNetCoreWebAPI/"]
RUN dotnet restore "DemoNetCoreWebAPI/DemoNetCoreWebAPI.csproj"
COPY . .
WORKDIR "/src/DemoNetCoreWebAPI"
RUN dotnet build "DemoNetCoreWebAPI.csproj" -c Release -o /app/build

FROM build AS publish
RUN dotnet publish "DemoNetCoreWebAPI.csproj" -c Release -o /app/publish

FROM base AS final
WORKDIR /app
COPY --from=publish /app/publish .
ENTRYPOINT ["dotnet", "DemoNetCoreWebAPI.dll"]
```

史詩

開發 ASP.NET Core Web API

任務	Description	所需的技能
使用 Visual Studio 建立範例 ASP.NET Core Web API。	若要建立範例 ASP.NET Core Web API，請執行下列動作： 1. 開啟 Visual Studio 2022。 2. 選擇 Create new project (建立新的專案)。 3. 選取 ASP.NET Core Web API 專案範本，然後選擇下一步。 4. 針對專案名稱，輸入 DemoNetCoreWebAPI，然後選擇下一步。 5. 選擇建立。 6. 若要在本機執行專案，請按 F5。	應用程式開發人員

任務	Description	所需的技能

	7. 確認預設 WeatherForecast API 端點使用 Swagger 傳回結果。
	8. 開啟命令提示字元，導覽至 .csproj 專案資料夾，然後執行下列命令，將新的 Web API 推送到您的 GitHub 儲存庫。
	`git add --all`
	`git commit -m "Initial Version"`
	`git push`	

任務	Description	所需的技能

建立 Dockerfile。	若要建立 Dockerfile，請執行下列其中一項操作：
	• 使用程式碼區段中的範例 Dockerfile 手動建立 Dockerfile。根據需求，選取適當的 .NET 基礎映像。如需 .NET 和 ASP.NET Core 相關映像的詳細資訊，請參閱Docker 中樞。
	• 使用 Visual Studio 和 Docker 桌面建立 Dockerfile。在解決方案瀏覽器中，在專案上按一下滑鼠右鍵，選擇 Add->Docker Support。針對目標作業系統，選取 Linux。確保新的 Dockerfile 與解決方案檔案 (.sln) 位於相同的路徑。
	
	若要將變更推送到您的 GitHub 儲存庫，請執行下列命令。
	`git add --all`
	`git commit -m "Dockerfile added"`
	`git push`
		應用程式開發人員

設定 Amazon EC2 Linux 執行個體

任務	Description	所需的技能
設定 基礎設施。	啟動 AWS CloudFormation 範本以建立基礎設施，其中包含下列項目： • 虛擬私有雲端 (VPC)，使用 AWS VPC Quick Start，其中有兩個公有子網路和兩個私有子網路橫跨兩個可用區域。 • 啟用 AWS Systems Manager 所需的 IAM 角色。 • 在其中一個私有子網路中，具有最新 SSM Agent 的 Amazon Linux 2 示範執行個體。雖然此執行個體沒有任何來自網際網路的直接連線，但可以使用 AWS Systems Manager Session Manager 安全地存取，而不需要堡壘主機。 **Note** Amazon Linux 2 即將結束支援。如需詳細資訊，請參閱 Amazon Linux 2 FAQs。 若要進一步了解如何使用 Session Manager 存取私有 Amazon EC2 執行個體，而不	應用程式開發人員、AWS 管理員、AWS DevOps

任務	Description	所需的技能
	需要堡壘主機，請參閱無堡壘世界部落格文章。	
登入 Amazon EC2 Linux 執行個體。	若要連線至私有子網路中的 Amazon EC2 Linux 執行個體，請執行下列動作： 1. 開啟 Amazon EC2 主控台。 2. 在導覽窗格中，選擇執行個體。 3. 選取 Amazon Linux 2 示範執行個體，然後選擇連線。 **Note** Amazon Linux 2 即將結束支援。如需詳細資訊，請參閱 Amazon Linux 2 FAQs。 4. 選擇 Session Manager (工作階段管理員)。 5. 選擇連線以開啟新的終端機視窗。 6. 執行下列命令。 `sudo su`	應用程式開發人員

任務	Description	所需的技能

安裝並啟動 Docker。	若要在 Amazon EC2 Linux 執行個體中安裝和啟動 Docker，請執行下列動作：
	1. 若要安裝 Docker，請執行下列命令。
	`yum install -y docker`
	2. 若要啟動 Docker 服務，請執行下列命令。
	`service docker start`
	3. 若要驗證 Docker 安裝，請執行下列命令。
	`docker info`
		應用程式開發人員、AWS 管理員、AWS DevOps

任務	Description	所需的技能

安裝 Git 並複製儲存庫。	若要在 Amazon EC2 Linux 執行個體上安裝 Git，並從 GitHub 複製儲存庫，請執行下列動作。
	1. 若要安裝 Git，請執行下列命令。
	`yum install git -y`
	2. 若要複製儲存庫，請執行下列命令。
	`git clone https://github.com/<username>/<repo-name>.git`
	3. 若要導覽至 Dockerfile，請執行下列命令。
	`cd <repo-name>/DemoNetCoreWebAPI/`
		應用程式開發人員、AWS 管理員、AWS DevOps

任務	Description	所需的技能

建置並執行 Docker 容器。	若要建置 Docker 映像並在 Amazon EC2 Linux 執行個體內執行容器，請執行下列動作：
	1. 若要建立 Docker 映像，請執行下列命令。
	`docker build -t aspnetcorewebapiimage -f Dockerfile .`
	2. 若要檢視所有 Docker 映像，請執行下列命令。
	`docker images`
	3. 若要建立和執行容器，請執行下列命令。
	`docker run -d -p 80:80 --name aspnetcorewebapicontainer aspnetcorewebapiimage`
		應用程式開發人員、AWS 管理員、AWS DevOps

測試 Web API

任務	Description	所需的技能
使用 curl 命令測試 Web API。	若要測試 Web API，請執行下列命令。 `curl -X GET "http://localhost/WeatherFo`	應用程式開發人員

任務	Description	所需的技能
	`recast" -H  "accept: text/plain"` 驗證 API 回應。 **Note** 您可以在本機執行時，從 Swagger 取得每個端點的 curl 命令。	

清除資源

任務	Description	所需的技能
刪除所有資源。	刪除堆疊以移除所有資源。這可確保您不需要為未使用的任何服務付費。	AWS 管理員、AWS DevOps

相關資源

- 使用 PuTTY 從 Windows 連線至 Linux 執行個體
- 使用 ASP.NET Core 建立 Web API
- 向無堡壘的世界邁進

使用 Amazon EFS on Amazon EKS 搭配 AWS Fargate，以持久性資料儲存來執行具狀態工作負載

Ricardo Morais、Rodrigo Bersa 和 Amazon Web Services Lucio Pereira

總結

此模式透過使用 AWS Fargate 佈建您的運算資源，為在 Amazon Elastic Kubernetes Service (Amazon EKS) 上執行的容器提供啟用 Amazon Elastic File System (Amazon EFS) 作為儲存裝置的指引。

此模式中描述的設定遵循安全最佳實務，並預設提供靜態安全性和傳輸中的安全性。若要加密 Amazon EFS 檔案系統，它會使用 AWS Key Management Service (AWS KMS) 金鑰，但您也可以指定金鑰別名，以分派建立 KMS 金鑰的程序。

您可以遵循此模式中的步驟，為proof-of-concept(PoC) 應用程式建立命名空間和 Fargate 設定檔、安裝用於整合 Kubernetes 叢集與 Amazon EFS 的 Amazon EFS 容器儲存界面 (CSI) 驅動程式、設定儲存類別，以及部署 PoC 應用程式。這些步驟會產生在多個 Kubernetes 工作負載之間共用的 Amazon EFS 檔案系統，透過 Fargate 執行。模式隨附可自動化這些步驟的指令碼。

如果您想要容器化應用程式中的資料持久性，並想要避免擴展操作期間遺失資料，則可以使用此模式。例如：

- DevOps 工具 – 常見的案例是開發持續整合和持續交付 (CI/CD) 策略。在這種情況下，您可以使用 Amazon EFS 作為共用檔案系統，在 CI/CD 工具的不同執行個體之間存放組態，或在 CI/CD 工具的不同執行個體之間存放管道階段的快取（例如 Apache Maven 儲存庫）。
- Web 伺服器 – 常見的案例是使用 Apache 做為 HTTP Web 伺服器。您可以使用 Amazon EFS 做為共用檔案系統，來存放 Web 伺服器不同執行個體之間共用的靜態檔案。在此範例案例中，修改會直接套用至檔案系統，而不是將靜態檔案製作成 Docker 映像。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 具有 Kubernetes 1.17 版或更新版本的現有 Amazon EKS 叢集（已測試至 1.27 版）
- 現有的 Amazon EFS 檔案系統，可動態繫結 Kubernetes StorageClass 和佈建檔案系統
- 叢集管理許可

- 設定為指向所需 Amazon EKS 叢集的內容

限制

- 當您搭配 Fargate 使用 Amazon EKS 時，需要考量一些限制。例如，不支援使用某些 Kubernetes 建構模組，例如 DaemonSets 和特殊權限容器。如需 Fargate 限制的詳細資訊，請參閱 Amazon EKS 文件中的 AWS Fargate 考量事項。
- 此模式提供的程式碼支援執行 Linux 或 macOS 的工作站。

產品版本

- AWS Command Line Interface (AWS CLI) 第 2 版或更新版本
- Amazon EFS CSI 驅動程式 1.0 版或更新版本（已測試至 2.4.8 版）
- eksctl 0.24.0 版或更新版本（已測試至 0.158.0 版）
- jq 1.6 版或更新版本
- kubectl 1.17 版或更新版本（已測試至 1.27 版）
- Kubernetes 1.17 版或更新版本（已測試至 1.27 版）

Architecture

目標架構包含下列基礎設施：

- 虛擬私有雲端 (VPC)
- 兩個可用區域
- 具有 NAT 閘道的公有子網路，可提供網際網路存取
- 具有 Amazon EKS 叢集和 Amazon EFS 掛載目標（也稱為掛載點) 的私有子網路
- VPC 層級的 Amazon EFS

以下是 Amazon EKS 叢集的環境基礎設施：

- 在命名空間層級容納 Kubernetes 建構的 AWS Fargate 設定檔
- 具有下列項目的 Kubernetes 命名空間：
  - 跨可用區域分佈的兩個應用程式 Pod

  - 在叢集層級繫結至持久性磁碟區 (PV) 的一個持久性磁碟區宣告 (PVC)
- 整個叢集的 PV，繫結至命名空間中的 PVC，並指向叢集外部私有子網路中的 Amazon EFS 掛載目標

工具

AWS 服務

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可用來從命令列與 AWS 服務互動。
- Amazon Elastic File System (Amazon EFS) 可協助您在 AWS 雲端中建立和設定共用檔案系統。在此模式中，它提供簡單、可擴展、全受管和共用的檔案系統，以便與 Amazon EKS 搭配使用。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 AWS 上執行 Kubernetes，而無需安裝或操作您自己的叢集。
- AWS Fargate 是 Amazon EKS 的無伺服器運算引擎。它為您的 Kubernetes 應用程式建立和管理運算資源。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。

其他工具

- Docker 是一組平台即服務 (PaaS) 產品，可在作業系統層級使用虛擬化在容器中交付軟體。
- eksctl 是一種命令列公用程式，用於在 Amazon EKS 上建立和管理 Kubernetes 叢集。
- kubectl 是一種命令列界面，可協助您針對 Kubernetes 叢集執行命令。
- jq 是用於剖析 JSON 的命令列工具。

Code

此模式的程式碼會在 GitHub 持續性組態中使用 AWS Fargate 儲存庫搭配 Amazon EKS 上的 Amazon EFS 中提供。指令碼會透過 在資料夾中依 epic `epic01` 組織`epic06`，對應至此模式中的 Epics 區段順序。

最佳實務

目標架構包含下列服務和元件，並遵循 AWS Well-Architected Framework 最佳實務：

- Amazon EFS，提供簡單、可擴展、全受管的彈性 NFS 檔案系統。這在 Pod 中執行的所有 PoC 應用程式複寫中用作共用檔案系統，這些複寫會分佈在所選 Amazon EKS 叢集的私有子網路中。
- 每個私有子網路的 Amazon EFS 掛載目標。這可提供叢集虛擬私有雲端 (VPC) 內每個可用區域的備援。
- 執行 Kubernetes 工作負載的 Amazon EKS。您必須先佈建 Amazon EKS 叢集，才能使用此模式，如先決條件一節中所述。
- AWS KMS，可為存放在 Amazon EFS 檔案系統中的內容提供靜態加密。
- Fargate，可管理容器的運算資源，讓您可以專注於業務需求，而不是基礎設施負擔。系統會為所有私有子網路建立 Fargate 設定檔。它提供叢集虛擬私有雲端 (VPC) 內每個可用區域的備援。
- Kubernetes Pod，用於驗證內容可由應用程式的不同執行個體共用、取用和寫入。

史詩

佈建 Amazon EKS 叢集 （選用）

任務	Description	所需的技能
建立 Amazon EKS 叢集。	**Note** 如果您已部署叢集，請跳到下一個史詩。在您現有的 AWS 帳戶中建立 Amazon EKS 叢集。在 GitHub 目錄中，使用其中一種模式，透過 Terraform 或 eksctl 部署 Amazon EKS 叢集。如需詳細資訊，請參閱《Amazon EKS 文件》中的建立 Amazon EKS 叢集。在 Terraform 模式中，也有範例顯示如何：將 Fargate 設定檔連結至 Amazon EKS	AWS 管理員、Terraform 或 eksctl 管理員、Kubernetes 管理員

任務	Description	所需的技能
	叢集、建立 Amazon EFS 檔案系統，以及在 Amazon EKS 叢集中部署 Amazon EFS CSI 驅動程式。	

任務	Description	所需的技能

匯出環境變數。	執行 env.sh 指令碼。這提供後續步驟中所需的資訊。
	`source ./scripts/env.sh`
	`Inform the AWS Account ID:`
	`<13-digit-account-id>`
	`Inform your AWS Region:`
	`<aws-Region-code>`
	`Inform your Amazon EKS Cluster Name:`
	`<amazon-eks-cluster-name>`
	`Inform the Amazon EFS Creation Token:`
	`<self-genereated-uuid>`
	如果尚未記下，您可以使用下列 CLI 命令取得上述要求的所有資訊。
	`# ACCOUNT ID`
	`aws sts get-caller-identity --query "Account" --output text`
	`# REGION CODE`
	`aws configure get region`
	`# CLUSTER EKS NAME`
	`aws eks list-clusters --query "clusters" --output text`
	`# GENERATE EFS TOKEN`
		AWS 系統管理員

任務	Description	所需的技能
	`uuidgen`	

建立 Kubernetes 命名空間和連結的 Fargate 設定檔

任務	Description	所需的技能

為應用程式工作負載建立 Kubernetes 命名空間和 Fargate 設定檔。	建立命名空間以接收與 Amazon EFS 互動的應用程式工作負載。執行 `create-k8s-ns-and-linked-fargate-profile.sh` 指令碼。您可以選擇使用自訂命名空間名稱或預設提供的命名空間 `poc-efs-eks-fargate` 。
	
	使用自訂應用程式命名空間名稱：
	
	`export $APP_NAMESPACE=<CUSTOM_NAME>`
	`./scripts/epic01/create-k8s-ns-and-linked-fargate-profile.sh \`
	`-c "$CLUSTER_NAME" -n "$APP_NAMESPACE"`
	
	沒有自訂應用程式命名空間名稱：
	
	`./scripts/epic01/create-k8s-ns-and-linked-fargate-profile.sh \`
	`-c "$CLUSTER_NAME"`
		具有授予許可的 Kubernetes 使用者

任務	Description	所需的技能
	其中 $CLUSTER_NAME 是 Amazon EKS 叢集的名稱。-n <NAMESPACE> 參數為選用；如果未收到通知，則會提供預設產生的命名空間名稱。	

建立 Amazon EFS 檔案系統

任務	Description	所需的技能
產生唯一的字符。	Amazon EFS 需要建立字符以確保等冪性操作（使用相同的建立字符呼叫操作沒有效果）。若要符合此要求，您必須透過可用的技術產生唯一的權杖。例如，您可以產生通用的唯一識別符 (UUID)，以用作建立字符。	AWS 系統管理員
建立 Amazon EFS 檔案系統。	建立 檔案系統，以接收應用程式工作負載讀取和寫入的資料檔案。您可以建立加密或未加密的檔案系統。（最佳實務是，此模式的程式碼會建立加密系統，以預設啟用靜態加密。）您可以使用唯一的對稱 AWS KMS 金鑰來加密檔案系統。如果未指定自訂金鑰，則會使用 AWS 受管金鑰。 在您為 Amazon EFS 產生唯一字符之後，請使用 create-efs.sh 指令碼來建立加密或未加密的 Amazon EFS 檔案系統。	AWS 系統管理員

任務	Description	所需的技能

	使用靜態加密，不使用 KMS 金鑰：
	
	`./scripts/epic02/create-efs.sh \ -c "$CLUSTER_NAME" \ -t "$EFS_CREATION_TOKEN"`
	
	其中 `$CLUSTER_NAME` 是 Amazon EKS 叢集的名稱`$EFS_CREATION_TOKEN` ，也是檔案系統的唯一建立字符。
	
	使用靜態加密搭配 KMS 金鑰：
	
	`./scripts/epic02/create-efs.sh \ -c "$CLUSTER_NAME" \ -t "$EFS_CREATION_TOKEN" \ -k "$KMS_KEY_ALIAS"`
	
	其中 `$CLUSTER_NAME` 是 Amazon EKS 叢集的名稱，`$EFS_CREATION_TOKEN` 是檔案系統的唯一建立字符，`$KMS_KEY_ALIAS` 是 KMS 金鑰的別名。
	
	沒有加密：
	
	`./scripts/epic02/create-efs.sh -d \`	

任務	Description	所需的技能
	`-c "$CLUSTER_NAME" \ -t "$EFS_CREATION_TOKEN"` 其中 `$CLUSTER_NAME` 是 Amazon EKS 叢集的名稱，`$EFS_CREATION_TOKEN` 是檔案系統的唯一建立字符，並-d停用靜態加密。	
建立安全群組。	建立安全群組以允許 Amazon EKS 叢集存取 Amazon EFS 檔案系統。	AWS 系統管理員
更新安全群組的傳入規則。	更新安全群組的傳入規則，以允許下列設定的傳入流量： • TCP 通訊協定 – 連接埠 2049 • 來源 – 包含 Kubernetes 叢集之 VPC 中私有子網路的 CIDR 區塊範圍	AWS 系統管理員
為每個私有子網路新增掛載目標。	針對 Kubernetes 叢集的每個私有子網路，為檔案系統和安全群組建立掛載目標。	AWS 系統管理員

在 Kubernetes 叢集中安裝 Amazon EFS 元件

任務	Description	所需的技能
部署 Amazon EFS CSI 驅動程式。	將 Amazon EFS CSI 驅動程式部署至叢集。驅動程式會根據應用程式建立的持久性磁碟區宣告佈建儲存體。執	具有授予許可的 Kubernetes 使用者

任務	Description	所需的技能
	行`create-k8s-efs-csi-sc.sh` 指令碼，將 Amazon EFS CSI 驅動程式和儲存類別部署至叢集。 `./scripts/epic03/create-k8s-efs-csi-sc.sh` 此指令碼使用 `kubectl`公用程式，因此請確定已設定內容並指向所需的 Amazon EKS 叢集。	
部署儲存類別。	將儲存體方案部署到 Amazon EFS 佈建器的叢集 (efs.csi.aws.com：//)。	具有授予許可的 Kubernetes 使用者

將 PoC 應用程式安裝到 Kubernetes 叢集

任務	Description	所需的技能
部署持久性磁碟區。	部署持久性磁碟區，並將其連結至建立的儲存體方案和 Amazon EFS 檔案系統的 ID。應用程式會使用持久性磁碟區來讀取和寫入內容。您可以在儲存欄位中指定持久性磁碟區的任何大小。Kubernetes 需要此欄位，但由於 Amazon EFS 是彈性檔案系統，因此不會強制執行任何檔案系統容量。 您可以部署具有或不具有加密的持久性磁碟區。（根據最佳實務，Amazon EFS CSI 驅	具有授予許可的 Kubernetes 使用者

任務	Description	所需的技能

	動程式預設會啟用加密。) 執行deploy-poc-app.sh 指令碼以部署持久性磁碟區、持久性磁碟區宣告和兩個工作負載。
	
	使用傳輸中加密：
	
	`./scripts/epic04/deploy-poc-app.sh \ -t "$EFS_CREATION_TOKEN"`
	
	其中 $EFS_CREATION_TOKEN 是檔案系統的唯一建立字符。
	
	沒有傳輸中加密：
	
	`./scripts/epic04/deploy-poc-app.sh -d \ -t "$EFS_CREATION_TOKEN"`
	
	其中 $EFS_CREATION_TOKEN 是檔案系統的唯一建立字符，並-d停用傳輸中的加密。	

任務	Description	所需的技能
部署應用程式請求的持久性磁碟區宣告。	部署應用程式請求的持久性磁碟區宣告，並將其連結至儲存體方案。使用與您先前建立的持久性磁碟區相同的存取模式。您可以在儲存欄位中指定持久性磁碟區宣告的任何大小。Kubernetes 需要此欄位，但由於 Amazon EFS 是彈性檔案系統，因此不會強制執行任何檔案系統容量。	具有授予許可的 Kubernetes 使用者
部署工作負載 1。	部署代表應用程式工作負載 1 的 Pod。此工作負載會將內容寫入檔案 `/data/out1.txt` 。	具有授予許可的 Kubernetes 使用者
部署工作負載 2。	部署代表應用程式工作負載 2 的 Pod。此工作負載會將內容寫入檔案 `/data/out2.txt` 。	具有授予許可的 Kubernetes 使用者

驗證檔案系統的持久性、耐用性和可共用性

任務	Description	所需的技能
檢查 的狀態`PersistentVolume` 。	輸入下列命令以檢查 的狀態`PersistentVolume` 。 `kubectl get pv` 如需輸出範例，請參閱其他資訊一節。	具有授予許可的 Kubernetes 使用者

任務	Description	所需的技能
檢查 的狀態PersistentVolumeClaim 。	輸入下列命令來檢查 的狀態PersistentVolumeClaim 。 `kubectl -n poc-efs-eks-fargate get pvc` 如需輸出範例，請參閱其他資訊一節。	具有授予許可的 Kubernetes 使用者

驗證工作負載 1 是否可以寫入檔案系統。	輸入下列命令來驗證工作負載 1 正在寫入 /data/out1.txt 。
	`kubectl exec -ti poc-app1 -n poc-efs-eks-fargate -- tail -f /data/out1.txt`
	結果如下：
	`...`
	`Thu Sep  3 15:25:07 UTC 2023 - PoC APP 1`
	`Thu Sep  3 15:25:12 UTC 2023 - PoC APP 1`
	`Thu Sep  3 15:25:17 UTC 2023 - PoC APP 1`
	`...`
		具有授予許可的 Kubernetes 使用者

任務	Description	所需的技能

驗證工作負載 2 是否可以寫入檔案系統。	輸入下列命令來驗證工作負載 2 正在寫入 `/data/out2.txt` 。
	`kubectl -n $APP_NAMESPACE exec -ti poc-app2 -- tail -f /data/out2.txt`
	結果如下：
	`... Thu Sep 3 15:26:48 UTC 2023 - PoC APP 2 Thu Sep 3 15:26:53 UTC 2023 - PoC APP 2 Thu Sep 3 15:26:58 UTC 2023 - PoC APP 2 ...`
		具有授予許可的 Kubernetes 使用者

任務	Description	所需的技能

驗證工作負載 1 可以讀取工作負載 2 寫入的檔案。	輸入下列命令來驗證工作負載 1 是否可以讀取由工作負載 2 寫入/data/out2.txt 的檔案。
	`kubectl exec -ti poc-app1 -n poc-efs-eks-fargate -- tail -n 3 /data/out2.txt`
	結果如下：
	`...`
	`Thu Sep  3 15:26:48 UTC 2023 - PoC APP 2`
	`Thu Sep  3 15:26:53 UTC 2023 - PoC APP 2`
	`Thu Sep  3 15:26:58 UTC 2023 - PoC APP 2`
	`...`
		具有授予許可的 Kubernetes 使用者

任務	Description	所需的技能

驗證工作負載 2 可以讀取由工作負載 1 寫入的檔案。	輸入下列命令來驗證工作負載 2 是否可以讀取由工作負載 1 寫入/data/out1.txt 的檔案。
	`kubectl -n $APP_NAMESPACE exec -ti poc-app2 -- tail -n 3 /data/out1.txt`
	結果如下：
	`... Thu Sep 3 15:29:22 UTC 2023 - PoC APP 1 Thu Sep 3 15:29:27 UTC 2023 - PoC APP 1 Thu Sep 3 15:29:32 UTC 2023 - PoC APP 1 ...`
		具有授予許可的 Kubernetes 使用者

任務	Description	所需的技能

在您移除應用程式元件後，驗證檔案是否已保留。	接著，您可以使用指令碼來移除應用程式元件（持久性磁碟區、持久性磁碟區宣告和 Pod)，並驗證檔案/data/out1.txt 和 /data/out2.txt 是否保留在檔案系統中。使用以下命令來執行 validate-efs-content.sh 指令碼。
	`./scripts/epic05/validate-efs-content.sh \ -t "$EFS_CREATION_TOKEN"`
	其中 $EFS_CREATION_TOKEN 是檔案系統的唯一建立字符。
	結果如下：
	`pod/poc-app-validation created Waiting for pod get Running state... Waiting for pod get Running state... Waiting for pod get Running state... Results from execution of 'find /data' on validation process pod: /data /data/out2.txt /data/out1.txt`
		具有授予許可的 Kubernetes 使用者、系統管理員

監控操作

任務	Description	所需的技能
監控應用程式日誌。	在第二天操作中，將應用程式日誌運送到 Amazon CloudWatch 以進行監控。	AWS 系統管理員、具有授與許可的 Kubernetes 使用者
使用 Container Insights 監控 Amazon EKS 和 Kubernetes 容器。	作為第二天操作的一部分，請使用 Amazon CloudWatch Container Insights 監控 Amazon EKS 和 Kubernetes 系統。此工具會從不同層級和維度的容器化應用程式收集、彙總和摘要指標。如需詳細資訊，請參閱相關資源一節。	AWS 系統管理員、具有授與許可的 Kubernetes 使用者
使用 CloudWatch 監控 Amazon EFS。	作為第二天操作的一部分，請使用 Amazon CloudWatch 監控檔案系統，該系統會收集來自 Amazon EFS 的原始資料並將其處理為可讀且幾近即時的指標。如需詳細資訊，請參閱相關資源一節。	AWS 系統管理員

清除資源

任務	Description	所需的技能
清除模式的所有已建立資源。	完成此模式後，請清除所有資源，以避免產生 AWS 費用。在您完成使用 PoC 應用程式後，執行`clean-up-resources.sh` 指令碼以移除所有資源。完成下列其中一個選項。	具有授予許可的 Kubernetes 使用者、系統管理員

任務	Description	所需的技能

	使用靜態加密搭配 KMS 金鑰：
	
	`./scripts/epic06/clean-up-resources.sh \ -c "$CLUSTER_NAME" \ -t "$EFS_CREATION_TOKEN" \ -k "$KMS_KEY_ALIAS"`
	
	其中 `$CLUSTER_NAME` 是 Amazon EKS 叢集的名稱，`$EFS_CREATION_TOKEN` 是檔案系統的建立字符，而 `$KMS_KEY_ALIAS` 是 KMS 金鑰的別名。
	
	不使用靜態加密：
	
	`./scripts/epic06/clean-up-resources.sh \ -c "$CLUSTER_NAME" \ -t "$EFS_CREATION_TOKEN"`
	
	其中 `$CLUSTER_NAME` 是 Amazon EKS 叢集的名稱，`$EFS_CREATION_TOKEN` 是檔案系統的建立字符。	

相關資源

參考

- AWS Fargate for Amazon EKS 現在支援 Amazon EFS （公告）
- 如何在 AWS Fargate 上使用 Amazon EKS 時擷取應用程式日誌 （部落格文章）
- 使用 Container Insights (Amazon CloudWatch 文件）
- 在 Amazon EKS 和 Kubernetes 上設定 Container Insights (Amazon CloudWatch 文件）
- Amazon EKS 和 Kubernetes Container Insights 指標 (Amazon CloudWatch 文件）
- 使用 Amazon CloudWatch 監控 Amazon EFS (Amazon EFS 文件）

GitHub 教學課程和範例

- 靜態佈建
- 傳輸中加密
- 從多個 Pod 存取檔案系統
- 在 StatefulSets 中使用 Amazon EFS
- 掛載子路徑
- 使用 Amazon EFS 存取點
- Terraform 的 Amazon EKS 藍圖

必要工具

- 安裝 AWS CLI 第 2 版
- 安裝 eksctl
- 安裝 kubectl
- 安裝 jq

其他資訊

以下是 `kubectl get pv`命令的範例輸出。

```
NAME         CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM
            STORAGECLASS   REASON   AGE
poc-app-pv   1Mi        RWX            Retain           Bound    poc-efs-eks-fargate/
poc-app-pvc   efs-sc                  3m56s
```

以下是 `kubectl -n poc-efs-eks-fargate get pvc`命令的範例輸出。

```
NAME          STATUS   VOLUME       CAPACITY   ACCESS MODES   STORAGECLASS   AGE
poc-app-pvc   Bound    poc-app-pv   1Mi        RWX            efs-sc         4m34s
```

使用 Amazon EKS Pod Identity 和 KEDA 在 Amazon EKS 中設定事件驅動的自動擴展

Dipen Desai、Abhay Diwan、Kamal Joshi 和 Mahendra Revanasiddappa，Amazon Web Services

總結

協同運作平台，例如 Amazon Elastic Kubernetes Service (Amazon EKS)，已簡化容器型應用程式的生命週期管理。這有助於組織專注於建置、保護、操作和維護容器型應用程式。隨著事件驅動型部署越來越常見，組織會根據各種事件來源更頻繁地擴展 Kubernetes 部署。此方法結合自動擴展，可提供隨需運算資源和專為應用程式邏輯量身打造的高效擴展，進而大幅節省成本。

KEDA 是以 Kubernetes 為基礎的事件驅動自動擴展器。KEDA 可協助您根據需要處理的事件數量，擴展 Kubernetes 中的任何容器。它很輕量，並與任何 Kubernetes 叢集整合。它也適用於標準 Kubernetes 元件，例如 Horizontal Pod Autoscaling (HPA)。KEDA 也提供 TriggerAuthentication，這項功能可協助您委派身分驗證。它可讓您描述與 ScaledObject 和部署容器分開的身分驗證參數。

AWS 提供支援各種 Kubernetes 部署選項的 AWS Identity and Access Management (IAM) 角色，包括 Amazon EKS、Amazon EKS Anywhere、 Red Hat OpenShift Service on AWS (ROSA) 和 Amazon Elastic Compute Cloud (Amazon EC2) 上的自我管理 Kubernetes 叢集。這些角色使用 OpenID Connect (OIDC) 身分提供者和 IAM 信任政策等 IAM 建構，在不同的環境中操作，而無需直接依賴 Amazon EKS 服務或 APIs。如需詳細資訊，請參閱 Amazon EKS 文件中的服務帳戶的 IAM 角色。

Amazon EKS Pod Identity 可簡化 Kubernetes 服務帳戶擔任 IAM 角色的程序，而無需 OIDC 供應商。它可讓您管理應用程式的登入資料。您可以將 IAM 角色與 Kubernetes 服務帳戶建立關聯，並將 Pod 設定為使用服務帳戶，而不是建立 AWS 登入資料並將其分發至容器或使用 Amazon EC2 執行個體的角色。這可協助您跨多個叢集使用 IAM 角色，並透過啟用跨 IAM 角色重複使用許可政策來簡化政策管理。

透過使用 Amazon EKS Pod Identity 實作 KEDA，企業可以實現高效的事件驅動型自動擴展和簡化的憑證管理。應用程式會根據需求擴展，以最佳化資源使用率並降低成本。

此模式可協助您將 Amazon EKS Pod 身分與 KEDA 整合。它展示了如何使用 `keda-operator` 服務帳戶和透過 委派身分驗證`TriggerAuthentication`。它還描述了如何在 KEDA 運算子的 IAM 角色與應用程式的 IAM 角色之間設定信任關係。此信任關係可讓 KEDA 監控事件佇列中的訊息，並調整目的地 Kubernetes 物件的擴展。

先決條件和限制

先決條件

- AWS Command Line Interface (AWS CLI) 2.13.17 版或更新版本，已安裝
- Python 3.11.5 版或更新版本，已安裝
- 適用於 Python (Boto3) 的 AWS SDK 1.34.135 版或更新版本，已安裝
- Helm 3.12.3 版或更新版本，已安裝
- kubectl 1.25.1 版或更新版本，已安裝
- Docker 引擎 26.1.1 版或更新版本，已安裝
- 已建立 Amazon EKS 叢集 1.24 版或更新版本
- 符合建立 Amazon EKS Pod Identity 代理程式的先決條件

限制

- 您必須在`keda-operator`角色與`keda-identity`角色之間建立信任關係。此模式的 Epics 區段提供說明。

Architecture

在此模式中，您會建立下列 AWS 資源：

- Amazon Elastic Container Registry (Amazon ECR) 儲存庫 – 在此模式中，此儲存庫名為 `keda-pod-identity-registry`。此私有儲存庫用於存放範例應用程式的 Docker 映像。
- Amazon Simple Queue Service (Amazon SQS) 佇列 – 在此模式中，此佇列名為 `event-messages-queue`。佇列可做為收集和儲存傳入訊息的訊息緩衝區。KEDA 會監控佇列指標，例如訊息計數或佇列長度，並根據這些指標自動擴展應用程式。
- 應用程式 IAM 角色 – 在此模式中，此角色名為 `keda-identity`。`keda-operator` 角色會擔任此角色。此角色允許存取 Amazon SQS 佇列。
- KEDA 運算子的 IAM 角色 – 在此模式中，此角色名為 `keda-operator`。KEDA 運算子使用此角色進行必要的 AWS API 呼叫。此角色具有擔任`keda-identity`角色的許可。由於 `keda-operator`與 `keda-identity`角色之間的信任關係，該`keda-operator`角色具有 Amazon SQS 許可。

透過 `TriggerAuthentication`和 `ScaledObject`Kubernetes 自訂資源，運算子會使用 `keda-identity`角色來與 Amazon SQS 佇列連線。根據佇列大小，KEDA 會自動擴展應用程式部署。它會為佇列中的每 5 個未讀取訊息新增 1 個 Pod。在預設組態中，如果 Amazon SQS 佇列中沒有未讀取的訊息，應用程式會縮減至 0 個 Pod。KEDA 運算子會以您指定的間隔監控佇列。

下圖顯示如何使用 Amazon EKS Pod Identity 為keda-operator角色提供 Amazon SQS 佇列的安全存取權。

該圖顯示以下工作流程：

1. 您在 Amazon EKS 叢集中安裝 Amazon EKS Pod Identity 代理程式。
2. 您可以在 Amazon EKS 叢集的 KEDA 命名空間中部署 KEDA 運算子。
3. 您可以在目標中建立 keda-operator和 keda-identity IAM 角色 AWS 帳戶。
4. 您可以在 IAM 角色之間建立信任關係。
5. 您可以在 security 命名空間中部署應用程式。
6. KEDA 運算子會在 Amazon SQS 佇列中輪詢訊息。
7. KEDA 會啟動 HPA，根據佇列大小自動擴展應用程式。

工具

AWS 服務

- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 上執行 Kubernetes， AWS 而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon Simple Queue Service (Amazon SQS) 提供安全、耐用且可用的託管佇列，可協助您整合和分離分散式軟體系統和元件。

其他工具

- KEDA 是以 Kubernetes 為基礎的事件驅動自動擴展器。

程式碼儲存庫

此模式的程式碼可在 GitHub 事件驅動的自動擴展中使用 EKS Pod Identity 和 KEDA 儲存庫。

最佳實務

建議您遵循下列最佳實務：

- Amazon EKS 最佳實務
- IAM 中的安全最佳實務
- Amazon SQS 最佳實務

史詩

建立 AWS 資源

任務	Description	所需的技能
建立 KEDA 運算子的 IAM 角色。	1. 登入 AWS 管理主控台，然後開啟 IAM 主控台。 2. 在導覽窗格中，選擇 Roles (角色)。 3. 選擇 Create Role (建立角色)。 4. 選擇 Custom trust policy (自訂信任政策) 角色類型。 5. 在自訂信任政策區段中，輸入此角色的下列自訂信任政策： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "Service": "pods.eks.amazonaws.com"`	AWS 管理員

任務	Description	所需的技能

	<pre>},
	"Action":
	[
	
	"sts:AssumeRole",
	
	"sts:TagSession"
	]
	}
	]
	}</pre>
	6. 在 Add permissions (新增許可) 頁面上，選擇 Next (下一步)。您不會將任何政策新增至此角色。
	7. 在角色名稱中，輸入 `keda-operator` 。
	8. 選擇建立角色。	

任務	Description	所需的技能
為範例應用程式建立 IAM 角色。	1. 在 IAM 主控台的導覽窗格中，選擇角色。 2. 選擇建立角色。 3. 選擇 Custom trust policy (自訂信任政策) 角色類型。 4. 在自訂信任政策區段中，輸入此角色的下列自訂信任政策。<account number> 將 取代為您的目標帳戶號碼： { "Version": "2012-10-17", "Statement": [{ "Effect": "Allow", "Principal": { "Service": "pods.eks.amazonaws.com", "AWS": "arn:aws:iam::<account number>:role/keda-operator" }, "Action": ["sts:AssumeRole", "sts:TagSession"] }] }	AWS 管理員

任務	Description	所需的技能
	5. 在新增許可頁面上，將下列 AWS 受管政策新增至角色： • `AmazonSQSReadOnlyAccess` • `AWSLambdaSQSQueueExecutionRole` 6. 選擇下一步。 7. 在角色名稱中，輸入 `keda-identity` 。 8. 選擇建立角色。	
建立 Amazon SQS 佇列。	1. 開啟 Amazon SQS 主控台。 2. 選擇建立佇列。 3. 針對類型，選擇標準。 4. 在建立佇列頁面上，針對名稱輸入 `event-messages-queue` 。 5. 選擇建立佇列。您不會變更此佇列的任何預設設定。	一般 AWS
建立 Amazon ECR 儲存庫。	1. 開啟 Amazon ECR 主控台。 2. 選擇建立儲存庫。 3. 針對儲存庫名稱，輸入 `keda-pod-identity-registry` 。 4. 選擇建立儲存庫。您不會變更此儲存庫的任何預設設定。	一般 AWS

設定 Amazon EKS 叢集

任務	Description	所需的技能
部署 Amazon EKS Pod Identity 代理程式。	針對目標 Amazon EKS 叢集，設定 Amazon EKS Pod Identity 代理程式。請遵循 Amazon EKS 文件中的設定 Amazon EKS Pod 身分代理程式中的指示。	AWS DevOps

部署 KEDA。	1. 輸入下列命令以在目標 Amazon EKS 叢集上部署 KEDA：
	`# Add Helm Repo for Keda`
	`helm repo add kedacore https://kedacore.github.io/charts`
	`# Update Helm repo`
	`helm repo update`
	`# Install Keda`
	`helm install keda kedacore/keda --namespace keda --create-namespace`
	如需詳細資訊，請參閱 KEDA 文件中的使用 Helm 部署。
	2. 成功部署後，請在輸出中驗證是否已為 KEDA 運算子建立三個部署。以下是範例輸出：
	`NAME READY`
		DevOps 工程師

任務	Description	所需的技能
	UP-TO-DATE AVAILABLE AGE keda-admission- webhooks 1/1 1 1 89s keda-operator 1/1 1 1 89s keda-operator- metrics-apiserver 1/1 1 1 89s	
將 IAM 角色指派給 Kubernetes 服務帳戶。	遵循 Amazon EKS 文件中將 IAM 角色指派給 Kubernetes 服務帳戶的指示。使用下列的值： • 針對 IAM 角色，輸入 `keda-operator` 。 • 針對 Kubernetes 命名空間，輸入 `keda`。 • 針對 Kubernetes 服務帳戶，輸入 `keda-operator` 。	AWS DevOps
建立 命名空間。	輸入下列命令以在目標 Amazon EKS 叢集中建立`security`命名空間： `kubectl create ns security`	DevOps 工程師

部署範例應用程式

任務	Description	所需的技能
複製應用程式檔案。	輸入下列命令，從 GitHub 使用 EKS Pod Identity 和 KEDA 儲存庫複製事件驅動的自動擴展： `git clone https://github.com/aws-samples/event-driven-autoscaling-using-podidentity-and-keda.git`	DevOps 工程師

建置 Docker 影像。	1. 輸入下列命令以導覽至複製的儲存庫：
	`cd event-driven-autoscaling-using-podidentity-and-keda`
	2. 輸入下列命令來建置範例應用程式的 Docker 映像：
	`docker build -t keda-pod-identity-registry .`
		DevOps 工程師

將 Docker 映像推送至 Amazon ECR。	1. 在您建置 Docker 映像的終端機中，輸入下列命令以登入 Amazon ECR。將 <AWS_REGION> 和 取代<AWS_ACCOUNT_ID> 為您的 AWS 環境的值： `aws ecr get-login-password \`	DevOps 工程師

任務	Description	所需的技能

	<pre>--region <AWS_REGION> | docker login \
	--username AWS \
	--password-stdin <AWS_ACCOUNT_ID>.dkr.ecr.<AWS_REGION>.amazonaws.com</pre>
	2. 輸入下列命令來標記映像。將 <AWS_REGION> 和 取代<AWS_ACCOUNT_ID> 為您的 AWS 環境的值：
	<pre>docker tag keda-pod-identity-registry:latest <AWS_ACCOUNT_ID>.dkr.ecr.<AWS_REGION>.amazonaws.com/keda-pod-identity-registry:latest</pre>
	3. 輸入下列命令，將映像推送至 Amazon ECR。將 <AWS_REGION> 和 取代<AWS_ACCOUNT_ID> 為您的 AWS 環境的值：
	<pre>docker push <AWS_ACCOUNT_ID>.dkr.ecr.<AWS_REGION>.amazonaws.com/keda-pod-identity-registry:latest</pre>	

任務	Description	所需的技能
	Note 您可以透過導覽至 Amazon ECR 儲存庫頁面，然後選擇檢視推送命令來尋找推送命令。	
部署範例應用程式。	1. 在複製的儲存庫中，開啟 deploy.yaml 檔案。 2. 將 <AWS_ACCOUNT_ID> 和 <AWS_REGION> 取代為您的環境的值。 3. 儲存並關閉 deploy.yaml 檔案。 4. 輸入下列命令，在目標 Amazon EKS 叢集上部署範例應用程式： `kubectl apply -f deploy.yaml` 此命令會在叢集中建立部署和服務帳戶。	DevOps 工程師

任務	Description	所需的技能

將 IAM 角色指派給應用程式服務帳戶。	執行下列其中一項操作，將 `keda-identity` IAM 角色與範例應用程式的 服務帳戶建立關聯：
	• 遵循 Amazon EKS 文件中將 IAM 角色指派給 Kubernetes 服務帳戶的指示。使用下列的值：
	  • 針對 IAM 角色，輸入 `keda-identity` 。
	  • 針對 Kubernetes 命名空間，輸入 `security`。
	  • 針對 Kubernetes 服務帳戶，輸入 `my-sqs-read-msgs` 。
	• 輸入下列 AWS CLI 命令。`<cluster-name>` 將 取代為目標 Amazon EKS 叢集的名稱，並將 取代`<role-ARN>` 為`keda-identity` 角色的 Amazon Resource Name (ARN)：
	`aws eks create-pod-identity-association \ --cluster-name <cluster-name> \ --role-arn <role-ARN> \ --namespace security \ --service-account my-sqs-read-msgs`
		DevOps 工程師

任務	Description	所需的技能
部署 ScaledObject 和 TriggerAuthentication 。	1. 在複製的儲存庫中，開啟 keda.yaml 檔案。 2. {{AWS_ACCOUNT_ID}} 將 取代為您目標的 ID AWS 帳戶。 3. {{AWS_REGION}} 將 取代為您的目標 AWS 區域。 4. （選用）在第 21–24 行中，更新ScaledObject 擴展政策的參數。如需這些參數的詳細資訊，請參閱下列內容： • pollingInterval • cooldownPeriod • idleReplicaCount • minReplicaCount 5. 儲存並關閉 keda.yaml 檔案。 6. 輸入下列命令來部署 ScaledObject 和資源TriggerAuthentication ： `kubectl -n security apply -f keda.yaml`	DevOps 工程師

測試自動擴展

任務	Description	所需的技能
傳送訊息至 Amazon SQS 佇列。	1. 輸入下列命令以導覽至複製的儲存庫：	DevOps 工程師

任務	Description	所需的技能

	`cd event-driven-autoscaling-using-podidentity-and-keda`
	2. 輸入下列命令，將測試訊息傳送至 Amazon SQS 佇列：
	`python sqs_send_msg.py`
	sqs_send_msg.py 指令碼可做為應用程式，產生用於測試自動擴展的訊息。
	Note
	如果您正在執行 Python 3，請輸入 `python3 sqs_send_msg.py` 。	

任務	Description	所需的技能

監控應用程式 Pod。	1. 在不同的終端機中，輸入下列命令來監控 Pod：
	`kubectl -n security get po`
	2. 對於 Amazon SQS 佇列中的每 5 個未讀取訊息，KEDA 會新增一個 Pod。在上一個命令的輸出中，確認正在新增新的 Pod。以下是範例輸出：
	`kubectl -n security get po`
	`NAME READY STATUS RESTARTS AGE`
	`q-read-797f4c7589-2bj76 1/1 Running 0 2s`
	`q-read-797f4c7589-4zxph 1/1 Running 0 49s`
	`q-read-797f4c7589-cg9dt 1/1 Running 0 18s`
	`q-read-797f4c7589-slc69 1/1 Running 0 33s`
	3. 完成測試後，在原始終端機中輸入 CTRL + C (Windows) 或 CMD + C
		DevOps 工程師

任務	Description	所需的技能
	(macOS)。這會停止 python sqs_send_msg.py 指令碼。	

疑難排解

問題	解決方案

KEDA 運算子無法擴展應用程式。	輸入下列命令來檢查 IAM `keda-operator` 角色的日誌：
	`kubectl logs -n keda -l app=keda-operator -c keda-operator`
	
	如果有`HTTP 403`回應代碼，則應用程式和 KEDA 擴展器沒有足夠的許可來存取 Amazon SQS 佇列。請完成下列步驟：
	1. 檢查`keda-identity` 角色的 IAM 政策和陳述式，以確認已授予佇列讀取存取權。
	2. 驗證 IAM 角色之間的信任關係。以下是範例：
	`{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "Service": "pods.eks.amazonaws.com" }, "Action": [ "sts:AssumeRole", "sts:TagSession" ]`

問題	解決方案

	<pre> }
	    ]
	}</pre>
	如果發生錯誤`Assume-Role` ，則 Amazon EKS 節點 IAM 角色無法擔任為 定義的 IAM 角色`TriggerAuthentication` 。請完成下列步驟：
	1. 輸入下列命令來刪除 `keda-operator` Pod 並建立新的 Pod：
	<pre>kubectl delete pod keda-operator-
	<alphenumeric-value> --namespace
	keda</pre>
	2. 輸入下列命令來檢查 Pod 擔任的身分：
	<pre>kubectl describe pod <keda-operator-
	pod-name> --namespace keda</pre>
	3. 當 Pod 成功重新啟動時，請確認下列兩個變數已新增至 Pod 描述：
	• `AWS_CONTAINER_CREDENTIALS_FULL_URI`
	• `AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE`

相關資源

- 設定 Amazon EKS Pod Identity Agent (Amazon EKS 文件）
- 部署 KEDA (KEDA 文件）
- ScaledObject 規格 (KEDA 文件）
- 使用 TriggerAuthentication 進行身分驗證 (KEDA 文件）

使用 PGO 在 Amazon EKS 上簡化 PostgreSQL 部署

Shalaka Dengale，Amazon Web Services

總結

此模式整合來自 Crunchy Data (PGO) 的 Postgres Operator 與 Amazon Elastic Kubernetes Service (Amazon EKS)，以簡化雲端原生環境中的 PostgreSQL 部署。PGO 提供在 Kubernetes 中管理 PostgreSQL 資料庫的自動化和可擴展性。當您將 PGO 與 Amazon EKS 結合時，它會形成強大的平台，以有效率地部署、管理和擴展 PostgreSQL 資料庫。

此整合提供下列主要優點：

- 自動化部署：簡化 PostgreSQL 叢集部署和管理。
- 自訂資源定義 (CRDs)： 使用 Kubernetes 基本概念進行 PostgreSQL 管理。
- 高可用性：支援自動容錯移轉和同步複寫。
- 自動化備份和還原： 簡化備份和還原程序。
- 水平擴展： 啟用 PostgreSQL 叢集的動態擴展。
- 版本升級：促進滾動升級，減少停機時間。
- 安全性：強制執行加密、存取控制和身分驗證機制。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 在 Linux、macOS 或 Windows 上安裝和設定 AWS Command Line Interface (AWS CLI) 第 2 版。macOS
- AWS CLI Config，從命令列連接 AWS 資源。
- 在 Linux、macOS 或 Windows 上安裝和設定 eksctl。
- `kubectl`，已安裝並設定為存取 Amazon EKS 叢集上的資源。如需詳細資訊，請參閱 Amazon EKS 文件中的設定 kubectl 和 eksctl。
- 您的電腦終端機已設定為存取 Amazon EKS 叢集。如需詳細資訊，請參閱《Amazon EKS 文件》中的設定您的電腦與叢集通訊。

產品版本

- Kubernetes 1.21–1.24 版或更新版本 （ 請參閱 PGO 文件)。
- PostgreSQL 10 版或更新版本。此模式使用 PostgreSQL 第 16 版。

限制

- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

Architecture

目標技術堆疊

- Amazon EKS
- Amazon Virtual Private Cloud (Amazon VPC)
- Amazon Elastic Compute Cloud (Amazon EC2)

目標架構

此模式會建置架構，其中包含具有三個節點的 Amazon EKS 叢集。每個節點在後端的一組 EC2 執行個體上執行。此 PostgreSQL 設定遵循主要複本架構，對於大量讀取的使用案例特別有效。架構包含下列元件：

- 主要資料庫容器 (pg-primary) 託管主 PostgreSQL 執行個體，其中所有寫入操作都會導向。
- 次要複本容器 (pg-replica) 託管從主要資料庫複寫資料並處理讀取操作的 PostgreSQL 執行個體。
- PgBouncer 是 PGO 隨附的 PostgreSQL 資料庫輕量型連線集區器。它位於用戶端和 PostgreSQL 伺服器之間，並充當資料庫連線的媒介。
- PGO 可在此 Kubernetes 環境中自動化 PostgreSQL 叢集的部署和管理。
- Patroni 是一種開放原始碼工具，可管理和自動化 PostgreSQL 的高可用性組態。它包含在 PGO 中。當您在 Kubernetes 中使用 Patroni 搭配 PGO 時，它在確保 PostgreSQL 叢集的彈性和容錯能力方面扮演重要角色。如需詳細資訊，請參閱 Patroni 文件。

工作流程包含以下步驟：

- 部署 PGO 運算子。您可以在在 Amazon EKS 上執行的 Kubernetes 叢集上部署 PGO 運算子。這可以透過使用 Kubernetes 資訊清單或 Helm Chart 來完成。此模式使用 Kubernetes 資訊清單。

- 定義 PostgreSQL 執行個體。當運算子執行時，您可以建立自訂資源 CRs)，以指定 PostgreSQL 執行個體的所需狀態。這包括儲存、複寫和高可用性設定等組態。
- 運算子管理。您可以透過 CRs 等 Kubernetes API 物件與運算子互動，以建立、更新或刪除 PostgreSQL 執行個體。
- 監控和維護。您可以監控在 Amazon EKS 上執行的 PostgreSQL 執行個體的運作狀態和效能。運算子通常會為監控目的提供指標和記錄。您可以視需要執行例行維護任務，例如升級和修補。如需詳細資訊，請參閱 Amazon EKS 文件中的監控叢集效能和檢視日誌。
- 擴展和備份：您可以使用運算子提供的功能來擴展 PostgreSQL 執行個體和管理備份。

此模式不包含監控、維護和備份操作。

自動化和擴展

- 您可以使用 CloudFormation 自動建立基礎設施。如需詳細資訊，請參閱《Amazon EKS 文件》中的使用 建立 CloudFormation Amazon EKS 資源。
- 您可以使用 GitVersion 或 Jenkins 建置號碼來自動化資料庫執行個體的部署。

工具

AWS 服務

- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 上執行 Kubernetes， AWS 而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。

其他工具

- eksctl 是在 Amazon EKS 上建立叢集的簡單命令列工具。
- kubectl 是命令列公用程式，用於對 Kubernetes 叢集執行命令。
- PGO 可自動化和擴展 Kubernetes 中 PostgreSQL 資料庫的管理。

最佳實務

遵循下列最佳實務，以確保部署順暢且有效率：

- 保護您的 EKS 叢集。實作 EKS 叢集的安全最佳實務，例如針對服務帳戶 AWS Identity and Access Management (IRSA)、網路政策和 VPC 安全群組使用 (IAM) 角色。限制對 EKS 叢集 API 伺服器的存取，並使用 TLS 加密節點和 API 伺服器之間的通訊。
- 確保在 Amazon EKS 上執行的 PGO 和 Kubernetes 之間的版本相容性。有些 PGO 功能可能需要特定的 Kubernetes 版本或引入相容性限制。如需詳細資訊，請參閱 PGO 文件中的元件與相容性。
- 規劃 PGO 部署的資源配置，包括 CPU、記憶體和儲存。考慮 PGO 及其管理的 PostgreSQL 執行個體的資源需求。監控資源用量並視需要擴展資源。
- 專為高可用性而設計。設計您的 PGO 部署以獲得高可用性，以盡可能減少停機時間並確保可靠性。跨多個可用區域部署多個 PGO 複本，以實現容錯能力。
- 為 PGO 管理的 PostgreSQL 資料庫實作備份和還原程序。使用 PGO 或第三方備份解決方案提供的功能，這些功能與 Kubernetes 和 Amazon EKS 相容。
- 為您的 PGO 部署設定監控和記錄，以追蹤效能、運作狀態和事件。使用 Prometheus 等工具來監控指標，並使用 Grafana 來視覺化。設定記錄以擷取 PGO 日誌以進行疑難排解和稽核。
- 正確設定聯網，以允許 PGO、PostgreSQL 執行個體和 Kubernetes 叢集中其他服務之間的通訊。使用 Amazon VPC 網路功能和 Kubernetes 網路外掛程式，例如 Calico 或 Amazon VPC CNI，以進行網路政策強制執行和流量隔離。
- 考慮效能、耐用性和可擴展性等因素，為您的 PostgreSQL 資料庫選擇適當的儲存選項。使用 Amazon Elastic Block Store (Amazon EBS) 磁碟區或 AWS 受管儲存服務進行持久性儲存。如需詳細資訊，請參閱 Amazon EKS 文件中的使用 Amazon EBS 存放 Kubernetes 磁碟區。
- 使用基礎設施做為程式碼 (IaC) 工具 CloudFormation ，例如 在 Amazon EKS 上自動化 PGO 的部署和組態。定義基礎設施元件，包括 EKS 叢集、聯網和 PGO 資源，做為一致性、可重複性和版本控制的程式碼。

史詩

建立 IAM 角色

任務	Description	所需的技能

建立 IAM 角色。	1. 在 中使用下列命令建立 IAM 角色 AWS CLI：
	
	`aws iam create-role \`
	`  --role-name {YourRoleName} \`
		AWS 管理員

任務	Description	所需的技能
	`--assume-role-policy-document '{ "Version": "2012-10-17", "Statement": [{ "Effect": "Allow", "Principal": { "Service": "eks.amazonaws.com" }, "Action": "sts:AssumeRole" }] }' && \ aws iam attach-role-policy \ --role-name {YourRoleName}\ --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy && \ aws iam attach-role-policy \ --role-name {YourRoleName}\ --policy-arn arn:aws:iam::aws:policy/AmazonEKSServicePolicy && \ aws iam attach-role-policy \ --role-name {YourRoleName}\`	

任務	Description	所需的技能
	--policy-arn arn:aws:iam::aws:policy/CloudWatchFullAccess 2. 在 中檢閱角色 AWS 管理主控台： a. 開啟 IAM 主控台。 b. 選擇角色，然後搜尋您建立的角色名稱。 c. 驗證是否已連接下列政策： AmazonEKSClusterPolicy AmazonEKSServicePolicy CloudWatchFullAccess	

建立 Amazon EKS 叢集

任務	Description	所需的技能
建立 Amazon EKS 叢集。	如果您已部署叢集，請略過此步驟。否則， AWS 帳戶 請使用 eksctl、Terraform 或 在您目前的 中部署 Amazon EKS 叢集 CloudFormation。此模式使用 eksctl進行叢集部署。	AWS 管理員、Terraform 或 eksctl 管理員、Kubernetes 管理員

任務	Description	所需的技能

	Note
	此模式使用 Amazon EC2 做為 Amazon EKS 的節點群組。如果您想要使用 AWS Fargate，請參閱 eksctl 文件中的managedNodeGroups 組態。
	
	1. 使用下列eksctl輸入檔案來產生叢集。
	
	sample-cluster.yaml：
	
	`apiVersion: eksctl.io/v1alpha5`
	`kind: ClusterConfig`
	`metadata:`
	`  name: postgresql`
	`  region: us-east-1`
	`  version: "1.29"`
	`accessConfig:`
	`  authenticationMode: API_AND_CONFIG_MAP`
	`availabilityZones:`
	`- us-east-1a`
	`- us-east-1b`
	`- us-east-1c`
	`nodeGroups:`
	`  - name: ng-1`
	`    instanceType: m5.16xlarge`
	`    desiredCapacity: 2`	

任務	Description	所需的技能

	<pre> - name: ng-2
	    instanceType:
	 m5.16xlarge
	    desiredCapacity:
	 2
	  - name: ng-3
	    instanceType:
	 m5.16xlarge
	    desiredCapacity:
	 2
	vpc:
	  cidr: 192.168.0
	.0/16
	  clusterEndpoints:
	    publicAccess:
	 true
	  nat:
	    gateway: HighlyAva
	ilable
	iamIdentity
	Mappings:
	  - arn: arn:aws:i
	am::<account-id>:r
	ole/<role-name> #
	 update the IAM role
	 ARN created in step 1
	    username: <user-
	name> # Enter the
	 user name per your
	 choice
	    noDuplicateARNs:
	 false</pre>
	2. 執行下列命令來建立叢集（提供 檔案的檔案路徑sample-cluster.yaml）：	

任務	Description	所需的技能
	`eksctl create cluster -f sample-cluster.yaml`	
驗證叢集的狀態。	執行下列命令以查看叢集中節點的目前狀態： `kubectl get nodes` 如果您遇到錯誤，請參閱 Amazon EKS 文件的疑難排解一節。	AWS 管理員、Terraform 或 eksctl 管理員、Kubernetes 管理員

建立 OIDC 身分提供者

任務	Description	所需的技能
啟用 IAM OIDC 提供者。	作為 Amazon EBS 容器儲存介面 (CSI) 驅動程式的先決條件，您必須為叢集擁有現有的 IAM OpenID Connect (OIDC) 提供者。 使用下列命令啟用 IAM OIDC 提供者： `eksctl utils associate-iam-oidc-provider --region={region} --cluster={YourClusterNameHere} --approve`	AWS 管理員

任務	Description	所需的技能
	如需此步驟的詳細資訊，請參閱 Amazon EKS 文件。	

為 Amazon EBS CSI 驅動程式建立 IAM 角色。	使用下列eksctl命令為 CSI 驅動程式建立 IAM 角色：
	`eksctl create iamserviceaccount \`
	`  --region {RegionName} \`
	`  --name ebs-csi-controller-sa \`
	`  --namespace kube-system \`
	`  --cluster {YourClusterNameHere} \`
	`  --attach-policy-arn arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy \`
	`  --approve \`
	`  --role-only \`
	`  --role-name AmazonEKS_EBS_CSI_DriverRole`
	如果您使用加密的 Amazon EBS 磁碟機，則必須進一步設定政策。如需說明，請參閱 Amazon EBS SCI 驅動程式文件。
		AWS 管理員

任務	Description	所需的技能
新增 Amazon EBS CSI 驅動程式。	使用下列`eksctl`命令來新增 Amazon EBS CSI 驅動程式： `eksctl create addon \ --name aws-ebs-csi-driver \ --cluster <YourClusterName> service-account-role-arn arn:aws:iam::$(aws sts get-caller-identity \ --query Account \ --output text):role/AmazonEKS_EBS_CSI_DriverRole \ --force`	AWS 管理員

安裝 PGO

任務	Description	所需的技能
複製 PGO 儲存庫。	複製 PGO 的 GitHub 儲存庫： `git clone https://github.com/CrunchyData/postgres-operator-examples.git`	AWS DevOps
提供建立服務帳戶的角色詳細資訊。	若要授予 Amazon EKS 叢集對所需 AWS 資源的存取權，請指定您先前在 `service_account.yaml` 檔案中建立之 OIDC 角色的 Amazon Resource Name (ARN)。此檔案位於 儲存庫的命名空間資料夾中。	AWS 管理員、Kubernetes 管理員

任務	Description	所需的技能

	`cd postgres-operator-examples`
	
	`--- metadata: annotations: eks.amazonaws.com/ role-arn: arn:aws:i am::<accountId>:role/ <role_name> # Update the OIDC role ARN created earlier`	

任務	Description	所需的技能

建立命名空間和 PGO 先決條件。	1. 執行以下命令來建立 命名空間：
	`kubectl apply -k kustomize/install/namespace`
	這會建立 PGO 的專用命名空間。如有必要，您可以修改`namespace.yml` 檔案，並將不同的名稱指派給命名空間。
	2. 執行下列命令，將預設組態套用至叢集：
	`kubectl apply --server-side -k kustomize/install/default`
	`kustomize/install/default` 提供 Kubernetes 角色型存取控制 (RBAC)、自訂資源定義 (CRD) 和 Kubernetes Manager 檔案的預設組態。
		Kunernetes 管理員

驗證 Pod 的建立。	確認已建立命名空間和預設組態： `kubectl get pods -n postgres-operator`	AWS 管理員、Kubernetes 管理員

任務	Description	所需的技能
驗證 PVCs。	使用下列命令來驗證持久性磁碟區宣告 PVCs)： `kubectl describe pvc -n postgres-operator`	AWS 管理員、Kubernetes 管理員

建立和部署 運算子

任務	Description	所需的技能

建立運算子。	修改位於 的檔案內容/kustomize/postgres/postgres.yaml 以符合下列項目：
	`spec:`
	`  instances:`
	`  - name: pg-1`
	`  replicas: 3`
	`  patroni:`
	`  dynamicConfiguration:`
	`  postgresql:`
	`  pg_hba:`
	`  - "host all all 0.0.0.0/0 trust"`
	`  # this line enabled logical replication with programmatic access`
	`  - "host all postgres 127.0.0.1/32 md5"`
	`  synchronous_mode: true`
	`  users:`
	`  - name: replicator`
	`  databases:`
		AWS 管理員、DBA、Kubernetes 管理員

任務	Description	所需的技能

	<pre>- testdb
	options: "REPLICATION"</pre>
	這些更新會執行下列動作：
	• 調整 PostgreSQL 組態設定，以方便存取 PostgreSQL 執行個體。
	• 包含複寫使用者、資料庫使用者和超級使用者的組態，以啟用串流複寫、資料庫存取和叢集管理。	

部署 運算子。	部署 PGO 運算子，以在 Kubernetes 環境中啟用 PostgreSQL 資料庫的簡化管理和操作： <pre>kubectl apply -k kustomize/postgres</pre>	AWS 管理員、DBA、Kubernetes 管理員

任務	Description	所需的技能

驗證部署。	1. 驗證是否已部署運算子：
	`kubectl get pods -n postgres-operator --selector=postgres-operator.crunchydata.com/instance-set \ -L postgres-operator.crunchydata.com/role`
	2. 確認已建立與運算子 Pod 相關聯的服務資源：
	`kubectl get svc -n postgres-operator`
	從命令輸出中，記下主要複本 (`primary_pod_name`) 和僅供讀取複本 (`read_pod_name`)。您將在後續步驟中使用這些項目。
		AWS 管理員、DBA、Kubernetes 管理員

驗證串流複寫

任務	Description	所需的技能
將資料寫入主要複本。	使用下列命令連線至 PostgreSQL 主要複本，並將資料寫入資料庫： `kubectl exec -it <primary_pod_name> bash -n postgres-operator`	AWS 管理員、Kubernetes 管理員

任務	Description	所需的技能

	`psql`
	
	`CREATE TABLE customers (firstname text, customer_id serial, date_created timestamp );`
	`\dt`	

確認僅供讀取複本具有相同的資料。	連線至 PostgreSQL 僅供讀取複本，並檢查串流複寫是否正常運作：
	
	`kubectl exec -it {read_pod_name} bash -n postgres-operator`
	
	`psql`
	
	`\dt`
	
	僅供讀取複本應具有您在上一個步驟中於主要複本中建立的資料表。
		AWS 管理員、Kubernetes 管理員

疑難排解

問題	解決方案
Pod 不會啟動。	• 使用下列命令來檢查 Pod 狀態： `kubectl get pods -n your-namespace` • 檢查日誌是否有任何錯誤：

問題	解決方案

	`kubectl logs your-pod-name -n your-namespace`
	• 檢查 Pod 事件是否有任何與您的 Pod 相關的異常事件：
	`kubectl describe pod your-pod-name -n your-namespace`

複本明顯落後於主要資料庫。	• 檢查複寫延遲：
	`SELECT * FROM pg_stat_replication;`
	• 請確定複本有足夠的 CPU 和記憶體資源。檢查資源限制：
	`kubectl describe pod your-replica-pod -n your-namespace`
	• 驗證儲存後端是否以最佳方式執行。慢速磁碟 I/O 可能會導致複寫延遲。

您無法查看 PostgreSQL 叢集的效能和運作狀態。	• 啟用 Amazon CloudWatch Logs 並確認日誌已傳送至 Amazon CloudWatch 進行分析。如需詳細資訊，請參閱 Amazon EKS 文件。 • 檢查 `pg_stat_activity` ： `SELECT * FROM pg_stat_activity;`

問題	解決方案

複寫無法運作。	• 檢視 中的複寫設定，以檢查主要組態postgresql.conf ：
	`wal_level = replica`
	`max_wal_senders = 10`
	`wal_keep_size = 64  # or wal_keep_segments in older versions`
	• 確認 pg_hba.conf 包含複寫許可：
	`host replication replica_user all md5`
	• 檢查複本組態。確定 recovery.conf 或同等設定 (standby.signal 和 primary_conninfo) 已在複本上正確設定。

相關資源

- Amazon Elastic Kubernetes Service (AWS 白皮書上的部署選項概觀)
- CloudFormation (AWS 白皮書上的部署選項概觀)
- 開始使用 Amazon EKS – eksctl (Amazon EKS 使用者指南)
- 設定 kubectl 和 eksctl (Amazon EKS 使用者指南)
- 為 OpenID Connect 聯合建立角色 (IAM 使用者指南)
- 設定 的設定 AWS CLI(AWS CLI 使用者指南)
- Crunchy Postgres for Kubernetes 文件
- Crunch & Learn：適用於 Kubernetes 5.0 的 Crunchy Postgres （影片）

使用 Application Load Balancer 在 Amazon ECS 中使用交互 TLS 簡化應用程式身分驗證

Olawale Olaleye 和 Shamanth Devagari，Amazon Web Services

總結

此模式可協助您使用 Application Application Load Balancer (ALB)，透過 Amazon Elastic Container Service (Amazon ECS) 中的交互 TLS，簡化應用程式身分驗證並卸載安全負擔。使用 ALB，您可以從中驗證 X.509 用戶端憑證 AWS 私有憑證授權單位。這種強大的組合有助於實現服務之間的安全通訊，減少應用程式中複雜身分驗證機制的需求。此外， 模式會使用 Amazon Elastic Container Registry (Amazon ECR) 來存放容器映像。

此模式中的範例使用來自公有圖庫的 Docker 影像，最初建立範例工作負載。之後，新的 Docker 映像會建置為儲存在 Amazon ECR 中。對於來源，請考慮 Git 型系統，例如 GitHub、GitLab 或 Bitbucket，或使用 Amazon Simple Storage Service Amazon S3 (Amazon S3)。若要建置 Docker 映像，請考慮 AWS CodeBuild 針對後續映像使用 。

先決條件和限制

先決條件

- AWS 帳戶 具有部署 AWS CloudFormation 堆疊存取權的作用中 。請確定您具有部署 CloudFormation 的 AWS Identity and Access Management (IAM) 使用者或角色許可。
- AWS Command Line Interface (AWS CLI) 已安裝。使用 或在 `~/.aws/credentials` 檔案中設定環境變數，在本機電腦 AWS CLI 或環境中設定您的 AWS 登入資料。
- 已安裝 OpenSSL。
- 已安裝 Docker。
- 熟悉工具 AWS 服務 中所述的 。
- Docker 和 NGINX 的知識。

限制

- Application Load Balancer 的相互 TLS 僅支援 X.509v3 用戶端憑證。不支援 X.509v1 用戶端憑證。
- 此模式程式碼儲存庫中提供的 CloudFormation 範本不包含佈建 CodeBuild 專案做為堆疊的一部分。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

- Docker 27.3.1 版或更新版本
- AWS CLI 2.14.5 版或更新版本

Architecture

下圖顯示此模式的架構元件。

該圖顯示以下工作流程：

1. 建立 Git 儲存庫，並將應用程式程式碼遞交至儲存庫。
2. 在 中建立私有憑證授權機構 (CA) AWS 私有 CA。
3. 建立 CodeBuild 專案。CodeBuildproject 由遞交變更觸發，並建立 Docker 映像並將建置映像發佈至 Amazon ECR。
4. 從 CA 複製憑證鏈和憑證內文，並將憑證套件上傳至 Amazon S3。
5. 使用您上傳到 Amazon S3 的 CA 套件建立信任存放區。將信任存放區與 Application Load Balancer (ALB) 上的交互 TLS 接聽程式建立關聯。
6. 使用私有 CA 為容器工作負載發行用戶端憑證。也使用 建立私有 TLS 憑證 AWS 私有 CA。
7. 將私有 TLS 憑證匯入 AWS Certificate Manager (ACM)，並搭配 ALB 使用。
8. 當 與 中的容器工作負載通訊時， 中的容器工作負載`ServiceTwo`會使用發行的用戶端憑證來驗證 ALB`ServiceOne`。
9. 中的容器工作負載會在與 中的容器工作負載通訊時，`ServiceOne`使用發行的用戶端憑證來驗證 ALB`ServiceTwo`。

自動化和擴展

透過使用 CloudFormation 或 SDK 的 API 操作來佈建 AWS 資源 AWS Cloud Development Kit (AWS CDK) ，即可完全自動化此模式。

您可以使用 AWS CodePipeline 實作持續整合和持續部署 (CI/CD) 管道，使用 CodeBuild 自動化容器映像建置程序，並將新版本部署至 Amazon ECS 叢集服務。

工具

AWS 服務

- AWS Certificate Manager (ACM) 可協助您建立、存放和續約公有和私有 SSL/TLS X.509 憑證和金鑰，以保護 AWS 網站和應用程式。
- AWS CloudFormation 可協助您設定 AWS 資源、快速一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。
- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- Amazon Elastic Container Service (Amazon ECS) 是一種高度可擴展的快速容器管理服務，可用於執行、停止和管理叢集上的容器。您可以在 管理的無伺服器基礎設施上執行任務和服務 AWS Fargate。或者，若要進一步控制您的基礎設施，您可以在您管理的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體叢集上執行任務和服務。
- Amazon ECS Exec 可讓您直接與容器互動，而不需要先與主機容器作業系統互動、開啟傳入連接埠或管理 SSH 金鑰。您可以使用 ECS Exec 在 Amazon EC2 執行個體上執行的容器上執行命令，或取得 Shell 至容器 AWS Fargate。
- Elastic Load Balancing (ELB) 會將傳入的應用程式或網路流量分配到多個目標。例如，您可以在一或多個可用區域中將流量分配到 Amazon EC2 執行個體、容器和 IP 地址。ELB 會監控其已註冊目標的運作狀態，並僅將流量路由至運作狀態良好的目標。ELB 會在傳入流量隨時間變更時擴展負載平衡器。它可以自動擴展到大多數工作負載。
- AWS Fargate 可協助您執行容器，而無需管理伺服器或 Amazon EC2 執行個體。Fargate 與 Amazon ECS 和 Amazon Elastic Kubernetes Service (Amazon EKS) 相容。您可以使用 Fargate 啟動類型或 Fargate 容量提供者來執行 Amazon ECS 任務和服務。若要這樣做，請將您的應用程式封裝在容器中、指定 CPU 和記憶體需求、定義聯網和 IAM 政策，以及啟動應用程式。每個 Fargate 任務都有自己的隔離界限，不會與其他任務共用基礎核心、CPU 資源、記憶體資源或彈性網路界面。
- AWS 私有憑證授權單位 可讓您建立私有憑證授權機構 (CA) 階層 (包括根 CA 和次級 CA)，而不需要操作內部部署 CA 的投資和維護成本。

其他工具

- Docker 是一組平台即服務 (PaaS) 產品，可在作業系統層級使用虛擬化在容器中交付軟體。
- GitHub、GitLab 和 Bitbucket 是一些常用的 Git 型來源控制系統，用於追蹤來源碼變更。
- NGINX Open Source 是開放原始碼負載平衡器、內容快取和 Web 伺服器。此模式會將其用作 Web 伺服器。
- OpenSSL 是一種開放原始碼程式庫，可提供 TLS 和 CMS 的 OpenSSL 實作所使用的服務。

程式碼儲存庫

此模式的程式碼可在 GitHub mTLS-with-Application-Load-Balancer-in-Amazon-ECS 儲存庫中使用。

最佳實務

- 使用 Amazon ECS Exec 執行命令或取得在 Fargate 上執行的容器 shell。您也可以使用 ECS Exec 來協助收集診斷資訊以進行偵錯。
- 使用安全群組和網路存取控制清單 ACLs) 來控制服務之間的傳入和傳出流量。Fargate 任務會從虛擬私有雲端 (VPC) 中設定的子網路接收 IP 地址。

史詩

建立儲存庫

任務	Description	所需的技能
下載原始程式碼。	若要下載此模式的原始碼，請分叉或複製 GitHub mTLS-with-Application-Load-Balancer-in-Amazon-ECS 儲存庫。	DevOps 工程師
建立 Git 儲存庫。	若要建立 Git 儲存庫以包含 Dockerfile 和 `buildspec.yaml` 檔案，請使用下列步驟： 1. 在虛擬環境中建立資料夾。將其命名為您的專案名稱。 2. 在本機電腦上開啟終端機，然後導覽至此資料夾。 3. 若要將 mTLS-with-Application-Load-Balancer-in-Amazon-ECS 儲存庫複製到您的專案目錄，請輸入下列命令：	DevOps 工程師

任務	Description	所需的技能
	`git clone https://github.com/aws-samples/mTLS-with-Application-Load-Balancer-in-Amazon-ECS.git`	

建立 CA 並產生憑證

任務	Description	所需的技能
在 中建立私有 CA AWS 私有 CA。	若要建立私有憑證授權機構 (CA)，請在終端機中執行下列命令。將範例變數中的值取代為您自己的值。 `export AWS_DEFAULT_REGION="us-west-2"` `export SERVICES_DOMAIN="www.example.com"` `export ROOT_CA_ARN=`aws acm-pca create-certificate-authority \` `--certificate-authority-type ROOT \` `--certificate-authority-configuration \` `"KeyAlgorithm=RSA_2048,` `SigningAlgorithm=SHA256WITHRSA,` `Subject={` `Country=US,` `State=WA,`	DevOps 工程師，AWS DevOps

任務	Description	所需的技能

	<pre>Locality=
	Seattle,
	Organizat
	ion=Build on AWS,
	Organizat
	ionalUnit=mTLS Amazon
	ECS and ALB Example,
	CommonName=
	${SERVICES_DOMAIN}}" \
	--query
	CertificateAuthori
	tyArn --output text`</pre> 如需詳細資訊，請參閱 AWS 文件中的在 中建立私有 CA AWS 私有 CA。	

任務	Description	所需的技能

建立並安裝私有 CA 憑證。	若要為您的私有根 CA 建立並安裝憑證，請在終端機中執行下列命令：
	
	1. 產生憑證簽署請求 (CSR)。
	
	`ROOT_CA_CSR=`aws acm-pca get-certificate-authority-csr \ --certificate-authority-arn ${ROOT_CA_ARN} \ --query Csr --output text``
	
	2. 發行根憑證。
	
	`AWS_CLI_VERSION=$( aws --version 2>&1 | cut -d/ -f2 | cut -d. -f1) [[ ${AWS_CLI_VERSION} -gt 1 ]] && ROOT_CA_CSR="$(echo ${ROOT_CA_CSR} | base64)" ROOT_CA_CERT_ARN= `aws acm-pca issue-certificate \ --certificate-authority-arn ${ROOT_CA_ARN} \ --template-arn arn:aws:acm-pca:::template/RootCACertificate/V1 \ --signing-algorithm SHA256WITHRSA \`
		AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	<pre> --validity Value=10,Type=YEARS \ --csr "${ROOT_C A_CSR}" \ --query Certifica teArn --output text`</pre> 3. 擷取根憑證。 <pre>ROOT_CA_CERT=`aws acm-pca get-certi ficate \ --certificate-arn ${ROOT_CA_CERT_ARN} \ --certificate- authority-arn ${ROOT_CA_ARN} \ --query Certifica te --output text` # store for later use aws acm-pca get-certi ficate \ --certificate-arn ${ROOT_CA_CERT_ARN} \ --certificate- authority-arn ${ROOT_CA_ARN} \ --query Certifica te --output text > ca-cert.pem</pre> 4. 匯入根 CA 憑證以將其安裝在 CA 上。 <pre>[[${AWS_CLI_VERSION} -gt 1]] && ROOT_CA_C ERT="$(echo</pre>	

任務	Description	所需的技能
	``` ${ROOT_CA_CERT} | base64)" aws acm-pca import-ce rtificate-authorit y-certificate \ --certificate- authority-arn $ROOT_CA_ARN \ --certificate "${ROOT_CA_CERT}" ``` 如需詳細資訊，請參閱 AWS 文件中的安裝 CA 憑證。	
請求受管憑證。	若要在 中請求私有憑證 AWS Certificate Manager 以搭配私有 ALB 使用，請使用下列命令： ``` export TLS_CERTI FICATE_ARN=`aws acm request-certificate \ --domain-name "*. ${DOMAIN_DOMAIN}" \ --certificate-auth ority-arn ${ROOT_CA _ARN} \ --query Certifica teArn --output text` ```	DevOps 工程師，AWS DevOps

任務	Description	所需的技能

使用私有 CA 發行用戶端憑證。	• 若要為兩項服務建立憑證簽署請求 (CSR)，請使用下列 AWS CLI 命令：
	
	`openssl req -out client_csr1.pem -new -newkey rsa:2048 -nodes -keyout client_private-key1.pem`
	
	`openssl req -out client_csr2.pem -new -newkey rsa:2048 -nodes -keyout client_private-key2.pem`
	
	此命令會傳回兩個服務的 CSR 和私有金鑰。
	
	• 若要發行 服務的憑證，請執行下列命令來使用您建立的私有 CA：
	
	`SERVICE_ONE_CERT_ARN=`aws acm-pca issue-certificate \ --certificate-authority-arn ${ROOT_CA_ARN} \ --csr fileb://client_csr1.pem \ --signing-algorithm "SHA256WITHRSA" \`
		DevOps 工程師，AWS DevOps

任務	Description	所需的技能
	--validity Value=5,Type="YEARS" --query CertificateArn --output text`  echo "SERVICE_ONE_CERT_ ARN: ${SERVICE _ONE_CERT_ARN}"  aws acm-pca get-certi ficate \ --certificate-auth ority-arn ${ROOT_CA _ARN} \ --certificate-arn ${SERVICE_ONE_CERT _ARN} \ | jq -r '.Certifi cate' > client_ce rt1.cert  SERVICE_TWO_CERT_ ARN=`aws acm-pca issue- certificate \ --certificate-auth ority-arn ${ROOT_CA _ARN} \ --csr fileb://c lient_csr2.pem \ --signing-algorith m "SHA256WITHRSA" \ --validity Value=5,Type="YEARS" --query CertificateArn --output text` echo "SERVICE_TWO_CERT_ ARN: ${SERVICE _TWO_CERT_ARN}" aws acm-pca get-certi ficate \	

任務	Description	所需的技能
	--certificate-auth ority-arn ${ROOT_CA _ARN} \ --certificate-arn ${SERVICE_TWO_CERT _ARN} \ | jq -r '.Certifi cate' > client_ce rt2.cert 如需詳細資訊，請參閱 AWS 文件中的發行私有終端實體憑證。	

佈建 AWS 服務

任務	Description	所需的技能
AWS 服務 使用 CloudForm ation 範本佈建 。	若要佈建虛擬私有雲端 (VPC)、Amazon ECS 叢集、Amazon ECS 服務、Application Load Balancer 和 Amazon Elastic Container Registry (Amazon ECR)，請使用 CloudFormation 範本。	DevOps 工程師
取得變數。	確認您有執行兩個服務的 Amazon ECS 叢集。若要擷取資源詳細資訊並將其儲存為變數，請使用下列命令： export LoadBalancerDNS= $(aws cloudformation describe-stacks -- stack-name ecs-mtls \ --output text \	DevOps 工程師

任務	Description	所需的技能

	`--query 'Stacks[0 ].Outputs[?OutputK ey==`LoadBalancerD NS`].OutputValue')`
	
	`export ECRReposi toryUri=$(aws cloudformation describe-stacks -- stack-name ecs-mtls \ --output text \ --query 'Stacks[0 ].Outputs[?OutputK ey==`ECRRepository Uri`].OutputValue')`
	
	`export ECRReposi toryServiceOneUri= $(aws cloudformation describe-stacks -- stack-name ecs-mtls \ --output text \ --query 'Stacks[0 ].Outputs[?OutputK ey==`ECRRepository ServiceOneUri`].Ou tputValue')`
	
	`export ECRReposi toryServiceTwoUri= $(aws cloudformation describe-stacks -- stack-name ecs-mtls \ --output text \ --query 'Stacks[0 ].Outputs[?OutputK ey==`ECRRepository ServiceTwoUri`].Ou tputValue')`
	
	`export ClusterName= $(aws cloudformation`	

任務	Description	所需的技能
	describe-stacks -- stack-name ecs-mtls \ --output text \ --query 'Stacks[0].Outputs[?OutputK ey==`ClusterName`] .OutputValue') export BucketName= $(aws cloudformation describe-stacks -- stack-name ecs-mtls \ --output text \ --query 'Stacks[0].Outputs[?OutputK ey==`BucketName`]. OutputValue') export Service1L istenerArn=$(aws cloudformation describe-stacks -- stack-name ecs-mtls \ --output text \ --query 'Stacks[0].Outputs[?OutputK ey==`Service1Liste nerArn`].OutputVal ue') export Service2L istenerArn=$(aws cloudformation describe-stacks -- stack-name ecs-mtls \ --output text \ --query 'Stacks[0].Outputs[?OutputK ey==`Service2Liste nerArn`].OutputVal ue')	

任務	Description	所需的技能
建立 CodeBuild 專案。	若要使用 CodeBuild 專案為您的 Amazon ECS 服務建立 Docker 映像，請執行下列動作： 1. 登入 AWS 管理主控台，並在 https：//https://console.aws.amazon.com/codesuite/codebuild/ 開啟 CodeBuild 主控台。 2. 建立新專案。針對來源，選擇您建立的 Git 儲存庫。如需有關不同類型的 Git 儲存庫整合的資訊，請參閱 AWS 文件中的使用連線。 3. 確認已啟用特權模式。若要建置 Docker 映像，此模式是必要的。否則，映像將無法成功建置。 4. 為每個服務使用共用的自訂`buildspec.yaml` 檔案。 5. 提供專案名稱和描述的值。 如需詳細資訊，請參閱 AWS 文件中的在 中建立建置專案 AWS CodeBuild。	AWS DevOps，DevOps 工程師

任務	Description	所需的技能

建置 Docker 映像。	您可以使用 CodeBuild 來執行映像建置程序。CodeBuild 需要與 Amazon ECR 互動和使用 Amazon S3 的許可。
	
	在此程序中，會建置 Docker 映像並推送至 Amazon ECR 登錄檔。如需範本和程式碼的詳細資訊，請參閱其他資訊。
	
	（選用）若要在本機建置以供測試之用，請使用下列命令：
	
	`# login to ECR`
	`aws ecr get-login`
	`-password | docker`
	`login --username`
	`AWS --password-stdin`
	`$ECRRepositoryUri`
	
	`# build image for`
	`service one`
	`cd /service1`
	`aws s3 cp s3://$Buc`
	`ketName/serviceone/`
	`service1/ --recursive`
	`docker build -t`
	`$ECRRepositoryServ`
	`iceOneUri .`
	`docker push $ECRRepos`
	`itoryServiceOneUri`
	
	`# build image for`
	`service two`
	`cd ../service2`
	`aws s3 cp s3://$Buc`
	`ketName/servicetwo/`
	`service2/ --recursive`
		DevOps 工程師

任務	Description	所需的技能
	`docker build -t $ECRRepositoryServiceTwoUri .` `docker push $ECRRepositoryServiceTwoUri`	

啟用交互 TLS

任務	Description	所需的技能
將 CA 憑證上傳至 Amazon S3。	若要將 CA 憑證上傳至 Amazon S3 儲存貯體，請使用下列範例命令： `aws s3 cp ca-cert.pem s3://$BucketName/acm-trust-store/`	AWS DevOps，DevOps 工程師
建立信任存放區。	若要建立信任存放區，請使用下列範例命令： `TrustStoreArn=`aws elbv2 create-trust-store --name acm-pca-trust-certs \` `--ca-certificates-bundle-s3-bucket $BucketName \` `--ca-certificates-bundle-s3-key acm-trust-store/ca-cert.pem --query 'TrustStores[].TrustStoreArn' --output text``	AWS DevOps，DevOps 工程師

任務	Description	所需的技能

上傳用戶端憑證。	若要將用戶端憑證上傳至 Amazon S3 for Docker 映像，請使用下列範例命令：
	
	`# for service one`
	`aws s3 cp client_cert1.cert s3://$BucketName/serviceone/`
	`aws s3 cp client_private-key1.pem s3://$BucketName/serviceone/`
	
	`# for service two`
	`aws s3 cp client_cert2.cert s3://$BucketName/servicetwo/`
	`aws s3 cp client_private-key2.pem s3://$BucketName/servicetwo/`
		AWS DevOps，DevOps 工程師

任務	Description	所需的技能

修改接聽程式。	若要在 ALB 上啟用交互 TLS，請使用下列命令修改 HTTPS 接聽程式：
	
	`aws elbv2 modify-listener \ --listener-arn $Service1ListenerArn \ --certificates CertificateArn=$TLS_CERTIFICATE_ARN_TWO \ --ssl-policy ELBSecurityPolicy-2016-08 \ --protocol HTTPS \ --port 8080 \ --mutual-authentication Mode=verify,TrustStoreArn=$TrustStoreArn,IgnoreClientCertificateExpiry=false`
	
	`aws elbv2 modify-listener \ --listener-arn $Service2ListenerArn \ --certificates CertificateArn=$TLS_CERTIFICATE_ARN_TWO \ --ssl-policy ELBSecurityPolicy-2016-08 \ --protocol HTTPS \ --port 8090 \ --mutual-authentication Mode=veri`
		AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	`fy,TrustStoreArn=$ TrustStoreArn,Igno reClientCertificat eExpiry=false` 如需詳細資訊，請參閱 AWS 文件中的在 Application Load Balancer 上設定交互 TLS。	

更新服務

任務	Description	所需的技能
更新 Amazon ECS 任務定義。	若要更新 Amazon ECS 任務定義，請修改新修訂中的 `image` 參數。 若要取得個別服務的值，請使用您在先前步驟中建置的新 Docker 映像 Uri 更新任務定義： `echo $ECRRepos itoryServiceOneUri` 或 `echo $ECRRepos itoryServiceTwoUri` `"containerDefiniti ons": [ { "name": "nginx", "image": "public.ecr.aws/ng inx/nginx:latest", # <----- change to new Uri "cpu": 0,`	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	如需詳細資訊，請參閱 AWS 文件中的使用 主控台更新 Amazon ECS 任務定義。	
更新 Amazon ECS 服務。	使用最新的任務定義更新服務。此任務定義是新建置 Docker 映像的藍圖，其中包含交互 TLS 身分驗證所需的用戶端憑證。 若要更新服務，請使用下列程序： 1. 開啟 Amazon ECS 主控台，網址為 https://console.aws.amazon.com/ecs/v2。 2. 在叢集頁面上，選擇叢集。 3. 在叢集詳細資訊頁面的服務區段中，選取服務旁的核取方塊，然後選擇更新。 4. 若要讓服務啟動新部署，請選取 Force new deployment (強制執行新部署)。 5. 針對任務定義，選擇任務定義系列和最新的修訂版。 6. 選擇更新。 對其他服務重複這些步驟。	AWS 管理員、AWS DevOps、DevOps 工程師

存取應用程式

任務	Description	所需的技能
複製應用程式 URL。	使用 Amazon ECS 主控台檢視任務。當任務狀態更新為執行中時，選取任務。在任務區段中，複製任務 ID。	AWS 管理員、AWS DevOps

測試您的應用程式。	若要測試您的應用程式，請使用 ECS Exec 存取任務。
	1. 對於服務一，請使用下列命令：
	`container="nginx"`
	`ECS_EXEC_TASK_ARN="<TASK ARN>"`
	`aws ecs execute-command --cluster $ClusterName \ --task $ECS_EXEC_TASK_ARN \ --container $container \ --interactive \ --command "/bin/bash"`
	2. 在服務一個任務的容器中，使用下列命令輸入內部負載平衡器url 和指向服務二的接聽程式連接埠。然後指定用戶端憑證的路徑以測試應用程式：
	`curl -kvs https://<internal-alb-url>:8090 --key /usr/local/share/ca-certifi`
		AWS 管理員、AWS DevOps

任務	Description	所需的技能
	cates/client.key --cert /usr/loca l/share/ca-certifi cates/client.crt 3. 在服務兩個任務的容器中，使用以下命令輸入內部負載平衡器url和指向服務一個的接聽程式連接埠。然後指定用戶端憑證的路徑以測試應用程式： curl -kvs https:// internal-alb-url>: 8080 --key /usr/loca l/share/ca-certifi cates/client.key --cert /usr/loca l/share/ca-certifi cates/client.crt Note curl 命令中的 -k旗標 （ 做為 的一部分-kvs) 會停用 SSL 憑證驗證。您可以在使用符合您網域名稱的 SSL 憑證時移除此旗標，以啟用適當的憑證驗證。	

相關資源

Amazon ECS 文件

- 使用主控台建立 Amazon ECS 任務定義
- 建立容器映像以用於 Amazon ECS
- Amazon ECS 叢集
- 適用於 的 Amazon ECS AWS Fargate
- Amazon ECS 聯網最佳實務
- Amazon ECS 服務定義參數

其他 AWS 資源

- 如何使用 AWS 私有 CA 在 Application Load Balancer 上設定 mTLS？ (AWS re:Post)

其他資訊

編輯 Dockerfile

下列程式碼顯示您在服務 1 的 Dockerfile 中編輯的命令：

```
FROM public.ecr.aws/nginx/nginx:latest
WORKDIR /usr/share/nginx/html
RUN echo "Returning response from Service 1: Ok" > /usr/share/nginx/html/index.html
ADD client_cert1.cert client_private-key1.pem /usr/local/share/ca-certificates/
RUN chmod -R 400 /usr/local/share/ca-certificates/
```

下列程式碼顯示您在服務 2 的 Dockerfile 中編輯的命令：

```
FROM public.ecr.aws/nginx/nginx:latest
WORKDIR /usr/share/nginx/html
RUN echo "Returning response from Service 2: Ok" > /usr/share/nginx/html/index.html
ADD client_cert2.cert client_private-key2.pem /usr/local/share/ca-certificates/
RUN chmod -R 400 /usr/local/share/ca-certificates/
```

如果您使用 CodeBuild 建置 Docker 映像， `buildspec` 檔案會使用 CodeBuild 組建編號來唯一地將映像版本識別為標籤值。您可以變更 `buildspec` 檔案以符合您的需求，如下列`buildspec` 自訂程式碼所示：

```
version: 0.2

phases:
```

```
  pre_build:
    commands:
      - echo Logging in to Amazon ECR...
      - aws ecr get-login-password --region $AWS_DEFAULT_REGION | docker login --
username AWS --password-stdin $ECR_REPOSITORY_URI
      - COMMIT_HASH=$(echo $CODEBUILD_RESOLVED_SOURCE_VERSION | cut -c 1-7)
      - IMAGE_TAG=${COMMIT_HASH:=latest}
  build:
    commands:
        # change the S3 path depending on the service
      - aws s3 cp s3://$YOUR_S3_BUCKET_NAME/serviceone/ $CodeBuild_SRC_DIR/ --
recursive
      - echo Build started on `date`
      - echo Building the Docker image...
      - docker build -t $ECR_REPOSITORY_URI:latest .
      - docker tag $ECR_REPOSITORY_URI:latest $ECR_REPOSITORY_URI:$IMAGE_TAG
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker images...
      - docker push $ECR_REPOSITORY_URI:latest
      - docker push $ECR_REPOSITORY_URI:$IMAGE_TAG
      - echo Writing image definitions file...
      # for ECS deployment reference
      - printf '[{"name":"%s","imageUri":"%s"}]' $CONTAINER_NAME $ECR_REPOSITORY_URI:
$IMAGE_TAG > imagedefinitions.json

artifacts:
  files:
    - imagedefinitions.json
```

更多模式

- 自動化刪除 AWS CloudFormation 堆疊和相關聯的資源
- 使用 AWS Service Catalog 和 自動化動態管道管理，以在 Gitflow 環境中部署 Hotfix 解決方案 AWS CodePipeline
- 使用 AWS CDK 自動為微服務建置 CI/CD 管道和 Amazon ECS 叢集
- 使用 GitHub 動作和 Terraform 建置 Docker 映像並將其推送至 Amazon ECR
- 容器化已由 Blu Age 現代化的大型主機工作負載
- 使用 Firelens 日誌路由器為 Amazon ECS 建立自訂日誌剖析器
- 使用 Terraform 在 Amazon Bedrock 上部署代理程式系統，搭配CrewAI 架構
- 使用 Terraform 部署容器化 Blu Age 應用程式的環境
- 使用 Amazon SageMaker 中的推論管道，將預先處理邏輯部署到單一端點中的 ML 模型
- 將工作負載從 Azure DevOps 管道部署到私有 Amazon EKS 叢集
- 使用 K8sGPT 和 Amazon Bedrock 整合實作採用 AI 技術的 Kubernetes 診斷和故障診斷
- 使用 AWS 程式碼服務和 AWS KMS 多區域金鑰，管理將微服務部署至多個帳戶和區域的藍/綠部署
- 使用 AWS CDK 設定 Amazon ECS Anywhere 來管理內部部署容器應用程式
- 在 Amazon ECS 上從 Oracle WebLogic 遷移至 Apache Tomcat (TomEE)
- 設定最低可行的資料空間以在組織之間共用資料
- 在 AWS 上現代化 ASP.NET Web Forms 應用程式
- 使用 AWS CloudFormation 和 AWS Config 監控 Amazon ECR 儲存庫是否有萬用字元許可
- 使用 CloudWatch Logs Insights 監控應用程式活動
- 使用 AWS CDK 和 GitLab 在 Amazon ECS Anywhere 上設定混合工作負載的 CI/CD 管道
- 使用 cert-manager 和 Let's Encrypt 為 Amazon EKS 上的應用程式設定end-to-end加密
- 使用 Flux 簡化 Amazon EKS 多租戶應用程式部署
- 使用 SageMaker AI 和 hydra 簡化從本機開發到可擴展實驗的機器學習工作流程
- 使用 AWS Lambda 在六邊形架構中建構 Python 專案
- 使用 LocalStack 和 Terraform Tests 測試 AWS 基礎設施
- 使用 AWS Fargate WaitCondition 勾點建構來協調資源相依性和任務執行
- 使用 Amazon Bedrock 代理程式，透過文字型提示在 Amazon EKS 中自動建立存取項目控制項

無伺服器

主題

- 使用 AWS Amplify 建置無伺服器 React Native 行動應用程式
- 透過單一控制平面管理多個 SaaS 產品的租用戶
- 使用與靜態 IP 地址相關聯的端點，整合 Amazon S3 預先簽章的 URL 產生和物件下載
- 在組織中建立跨帳戶 Amazon EventBridge 連線
- 使用 Kinesis Data Streams 和 Firehose 搭配 將 DynamoDB 記錄交付至 Amazon S3 AWS CDK
- 在 Amazon API Gateway 中使用自訂網域實作路徑型 API 版本控制
- 將 psycopg2 程式庫匯入 AWS Lambda ，以與您的 PostgreSQL 資料庫互動
- 將 Amazon API Gateway 與 Amazon SQS 整合，以處理非同步 REST APIs
- 使用 Amazon API Gateway 和 AWS Lambda 非同步處理事件
- 使用 Amazon API Gateway 和 Amazon DynamoDB Streams 非同步處理事件
- 使用 Amazon API Gateway、Amazon SQS 和 AWS Fargate 非同步處理事件
- 從 AWS Step Functions 同步執行 AWS Systems Manager 自動化任務 AWS Step Functions
- 在 AWS Lambda 函數中使用 Python 執行 S3 物件的平行讀取
- 將遙測資料從 AWS Lambda 傳送至 OpenSearch，以進行即時分析和視覺化
- 為以儲存格為基礎的架構設定無伺服器儲存格路由器
- 透過 VPC 端點設定 Amazon S3 儲存貯體的私有存取權
- AWS Step Functions 使用 Amazon Bedrock 對 中的狀態進行故障診斷
- 更多模式

使用 AWS Amplify 建置無伺服器 React Native 行動應用程式

Deekshitulu Pentakota，Amazon Web Services

總結

此模式示範如何使用 AWS Amplify 和下列 AWS 服務，為 React Native 行動應用程式建立無伺服器後端：

- AWS AppSync
- Amazon Cognito
- Amazon DynamoDB

在您使用 Amplify 設定和部署應用程式的後端之後，Amazon Cognito 會驗證應用程式使用者，並授權他們存取應用程式。然後AWS AppSync 會與前端應用程式和後端 DynamoDB 資料表互動，以建立和擷取資料。

> **Note**
>
> 此模式使用簡單的「ToDoList」應用程式做為範例，但您可以使用類似的程序來建立任何 React Native 行動應用程式。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Amplify 命令列界面 (Amplify CLI)，已安裝並設定
- XCode （任何版本）
- Microsoft Visual Studio （任何版本、任何程式碼編輯器、任何文字編輯器）
- 熟悉 Amplify
- 熟悉 Amazon Cognito
- 熟悉 AWS AppSync
- 熟悉 DynamoDB
- 熟悉 Node.js

- 熟悉 npm
- 熟悉 React 和 React Native
- 熟悉 JavaScript 和 ECMAScript 6 (ES6)
- 熟悉 GraphQL

Architecture

下圖顯示在 AWS 雲端中執行 React Native 行動應用程式後端的範例架構：

圖表顯示下列架構：

1. Amazon Cognito 會驗證應用程式使用者，並授權他們存取應用程式。
2. 若要建立和擷取資料，AWS AppSync 會使用 GraphQL API 與前端應用程式和後端 DynamoDB 資料表互動。

工具

AWS 服務

- AWS Amplify 是一組專門建置的工具和功能，可協助前端 Web 和行動開發人員在 AWS 上快速建置完整堆疊的應用程式。
- AWS AppSync 提供可擴展的 GraphQL 介面，可協助應用程式開發人員結合來自多個來源的資料，包括 Amazon DynamoDB、AWS Lambda 和 HTTP APIs。
- Amazon Cognito 為 Web 和行動應用程式提供身分驗證、授權和使用者管理。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。

Code

此模式中使用的範例應用程式的程式碼可在 GitHub aws-amplify-react-native-ios-todo-app 儲存庫中找到。若要使用範例檔案，請遵循此模式的 Epics 區段中的指示。

史詩

建立並執行您的 React Native 應用程式

任務	Description	所需的技能
設定 React Native 開發環境。	如需說明，請參閱 React Native 文件中的設定開發環境。	應用程式開發人員

在 iOS 模擬器中建立並執行 ToDoList React Native 行動應用程式。	1. 在新的終端機視窗中執行下列命令，在本機環境中的 中建立新的 React Native 行動應用程式專案目錄：
	`npx react-native init ToDoListAmplify`
	2. 執行下列命令，導覽至專案的根目錄：
	`cd ToDoListAmplify`
	3. 透過執行下列命令來執行應用程式：
	`npx react-native run-ios`
		應用程式開發人員

初始化應用程式的新後端環境

任務	Description	所需的技能
在 Amplify 中建立支援應用程式所需的後端服務。	1. 在本機環境中，從專案的根目錄 (ToDoListAmplify) 執行下列命令： `amplify init`	應用程式開發人員

任務	Description	所需的技能

	2. 出現提示，要求您提供有關應用程式的資訊。根據您的使用案例輸入必要資訊。然後按 Enter 鍵。
	
	對於此模式中使用的 ToDoList 應用程式設定，請套用下列範例組態。
	
	React Native Amplify 應用程式組態設定範例
	
	`? Name: ToDoListAmplify`
	
	`? Environment: dev`
	
	`? Default editor: Visual Studio Code`
	
	`? App type: javascript`
	
	`? Javascript framework : react-native`
	
	`? Source Directory Path: src`
	
	`? Distribution Directory Path: /`
	
	`? Build Command: npm run-script build`
	
	`? Start Command: npm run-script start`
	
	`? Select the authentication method you want to use: AWS profile`	

任務	Description	所需的技能
	``` ? Please choose the profile you want to use: default ``` 如需詳細資訊，請參閱 Amplify 開發中心文件中建立新的 Amplify 後端。 **Note** `amplify init` 命令會使用 AWS CloudFormation 佈建下列資源： - 已驗證和未驗證使用者的 AWS Identity and Access Management (IAM) 角色 (驗證角色和未驗證角色) - 用於部署的 Amazon Simple Storage Service (Amazon S3) 儲存貯體 （此模式的範例應用程式為 Amplify-meta.json) - Amplify 託管中的後端環境	

將 Amazon Cognito 身分驗證新增至 Amplify React Native 應用程式

任務	Description	所需的技能
建立 Amazon Cognito 身分驗證服務。	1. 在本機環境中，從專案的根目錄 (ToDoListAmplify) 執行下列命令：	應用程式開發人員

任務	Description	所需的技能

	`amplify add auth`
	2. 出現提示，要求您提供有關身分驗證服務組態設定的資訊。根據您的使用案例輸入必要資訊。然後按 Enter 鍵。
	
	對於此模式中使用的 ToDoList 應用程式設定，請套用下列範例組態。
	
	範例身分驗證服務組態設定
	
	`? Do you want to use the default authentication and security configuration? \`
	`Default configuration`
	
	`? How do you want users to be able to sign in? \`
	`Username`
	
	`? Do you want to configure advanced settings? \`
	`No, I am done`
	
	Note
	`amplify add auth` 命令會在專案根目錄中的本機資料夾 (擴增) 中建立必要的資料夾、檔案和相依性檔案	

任務	Description	所需的技能
	。對於此模式中使用的 ToDoList 應用程式設定，系統會為此目的建立 aws-exports.js。	

將 Amazon Cognito 服務部署至 AWS 雲端。	1. 從專案的根目錄中，執行下列 Amplify CLI 命令：
	`amplify push`
	2. 出現確認部署的提示。輸入是。然後按 Enter 鍵。
	
	Note
	若要查看專案中已部署的服務，請執行下列命令，前往 Amplify 主控台：
	
	`amplify console`
		應用程式開發人員

任務	Description	所需的技能

安裝 React Native 所需的 Amplify 程式庫，以及 iOS 的 CocoaPods 相依性。	1. 從專案的根目錄執行下列命令，安裝所需的 Amplify 開放原始碼用戶端程式庫：
	`npm install aws-amplify aws-amplify-react-native \ amazon-cognito-identity-js @react-native-community/netinfo \ @react-native-async-storage/async-storage`
	2. 執行下列命令，安裝 iOS 所需的 CocoaPods 相依性：
	`npx pod-install`
		應用程式開發人員

任務	Description	所需的技能

匯入並設定 Amplify 服務。	在應用程式的進入點檔案中（例如 App.js），輸入以下幾行程式碼匯入並載入 Amplify 服務的組態檔案：
	`import Amplify from 'aws-amplify'`
	`import config from './src/aws-exports'`
	`Amplify.configure(config)`
	Note
	如果您在應用程式進入點檔案中匯入 Amplify 服務後收到錯誤，請停止應用程式。然後，開啟 XCode 並從專案的 iOS 資料夾選取 ToDoListAmplify.xcworkspace，然後執行應用程式。
		應用程式開發人員

任務	Description	所需的技能

更新應用程式的進入點檔案，以使用 withAuthenticator 高階元件 (HOC)。	**Note**
	`withAuthenticator` HOC 只會使用幾行程式碼，在您的應用程式中提供登入、註冊和忘記密碼工作流程。如需詳細資訊，請參閱 Amplify 開發中心中的選項 1：使用預先建置的 UI 元件。此外，React 文件中的高階元件。
	
	1. 在應用程式的進入點檔案中（例如 App.js），輸入以下幾行程式碼來匯入 `withAuthenticator` HOC：
	`import { withAuthenticator } from 'aws-amplify-react-native'`
	2. 輸入下列程式碼以匯出 withAuthenticator HOC：
	`export default withAuthenticator(App)`
	
	withAuthenticator HOC 程式碼範例
		應用程式開發人員

任務	Description	所需的技能

	`import Amplify from 'aws-amplify'`
	`import config from './src/aws-exports'`
	`Amplify.configure(config)`
	`import { withAuthenticator } from 'aws-amplify-react-native';`
	
	`const App = () => {`
	`  return null;`
	`};`
	
	`export default withAuthenticator(App);`
	
	Note
	在 iOS 模擬器中，應用程式會顯示 Amazon Cognito 服務提供的登入畫面。	

任務	Description	所需的技能
測試身分驗證服務設定。	在 iOS 模擬器中，執行下列動作： 1. 使用真實的電子郵件地址在應用程式中建立新帳戶。然後，驗證碼會傳送至已註冊的電子郵件。 2. 使用您在驗證電子郵件中收到的程式碼來驗證帳戶設定。 3. 輸入您建立的使用者名稱和密碼。然後，選擇登入。出現歡迎畫面。 **Note** 您也可以開啟 Amazon Cognito 主控台，並檢查是否已在身分集區中建立新的使用者。	應用程式開發人員

將 AWS AppSync API 和 DynamoDB 資料庫連線至應用程式

任務	Description	所需的技能
建立 AWS AppSync API 和 DynamoDB 資料庫。	1. 將 AWS AppSync API 新增至您的應用程式，並從專案的根目錄執行下列 Amplify CLI 命令，以自動佈建 DynamoDB 資料庫： `amplify add api`	應用程式開發人員

任務	Description	所需的技能

	2. 出現提示，要求您提供有關 API 和資料庫組態設定的資訊。根據您的使用案例輸入必要資訊。然後按 Enter 鍵。Amplify CLI 會在文字編輯器中開啟 GraphQL 結構描述檔案。
	
	對於此模式中使用的 ToDoList 應用程式設定，請套用下列範例組態。
	
	範例 API 和資料庫組態設定
	
	`? Please select from one of the below mentioned services: \ GraphQL`
	
	`? Provide API name: todolistamplify`
	
	`? Choose the default authorization type for the API \ Amazon Cognito User Pool`
	
	`Do you want to use the default authentication and security configuration`
	
	`? Default configuration How do you want users to be able to sign in? \ Username`	

任務	Description	所需的技能

	``
	Do you want to configure
	 advanced settings? \
	No, I am done.
	
	? Do you want to
	 configure advanced
	 settings for the
	 GraphQL API \
	No, I am done.
	
	? Do you have an
	 annotated GraphQL
	 schema? \
	No
	
	? Choose a schema
	 template: \
	Single object with
	 fields (e.g., "Todo"
	 with ID, name,
	 description)
	
	? Do you want to edit
	 the schema now? \
	Yes
	``
	
	GraphQL 結構描述範例
	
	``
	 type Todo @model {
	   id: ID!
	   name: String!
	   description: String
	}
	``	

任務	Description	所需的技能

部署 AWS AppSync API。	1. 在專案的根目錄中，執行下列 Amplify CLI 命令：
	
	`amplify push`
	
	2. 出現提示，要求您提供有關 API 和資料庫組態設定的詳細資訊。根據您的使用案例輸入必要資訊。然後按 Enter 鍵。您的應用程式現在可以與 AWS AppSync API 互動。
	
	對於此模式中使用的 ToDoList 應用程式設定，請套用下列範例組態。
	
	AWS AppSync API 組態設定範例
	
	Note
	下列組態會在 AWS AppSync 中建立 GraphQL API，並在 Dynamo DB中建立 Todo 資料表。
	
	`? Are you sure you want to continue? Yes`
	`? Do you want to generate code for your newly created GraphQL API Yes`
		應用程式開發人員

任務	Description	所需的技能
	``` ? Choose the code generation language target javascript ? Enter the file name pattern of graphql queries, mutations and subscriptions src/ graphql/**/*.js ? Do you want to generate/update all possible GraphQL operations - \ queries, mutations and subscriptions Yes ? Enter maximum statement depth \ [increase from default if your schema is deeply nested] 2 ```	
將應用程式的前端連接至 AWS AppSync API。	若要使用此模式中提供的範例 ToDoList 應用程式，請從 aws-amplify-react-native-ios-todo-app GitHub 儲存庫中的 App.js 檔案複製程式碼。然後，將範例程式碼整合到您的本機環境。 儲存庫的 App.js 檔案中提供的範例程式碼執行下列動作： • 顯示使用標題和描述欄位建立 ToDo 項目的表單 • 顯示待辦事項項目清單 (標題和描述) • 使用 `aws-amplify` 方法張貼和擷取資料	應用程式開發人員

相關資源

- AWS Amplify
- Amazon Cognito
- AWS AppSync
- Amazon DynamoDB
- React (React 文件)

透過單一控制平面管理多個 SaaS 產品的租用戶

Ramanna Avancha、Kishan Kavala、Anusha Mandava 和 Jenifer Pascal，Amazon Web Services

總結

此模式說明如何在 AWS 雲端的單一控制平面上管理多個軟體即服務 (SaaS) 產品的租用戶生命週期。提供的參考架構可協助組織減少其個別 SaaS 產品的備援、共用功能的實作，並提供大規模的控管效率。

大型企業可以在各種業務單位擁有多個 SaaS 產品。這些產品通常需要佈建供不同訂閱層級的外部租戶使用。如果沒有常見的租用戶解決方案，IT 管理員必須花時間跨多個 SaaS APIs 管理未區分的功能，而不是專注於核心產品功能開發。

此模式中提供的常見租戶解決方案有助於集中管理組織的許多共用 SaaS 產品功能，包括下列項目：

- 安全
- 租戶佈建
- 租戶資料儲存
- 租戶通訊
- 產品管理
- 指標記錄和監控

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Amazon Cognito 或第三方身分提供者 (IdP) 的知識
- Amazon API Gateway 的知識
- AWS Lambda 的知識
- Amazon DynamoDB 的知識
- 了解 AWS Identity and Access Management (IAM)
- AWS Step Functions 的知識
- 了解 AWS CloudTrail 和 Amazon CloudWatch

- Python 程式庫和程式碼的知識
- 了解 SaaS APIs，包括不同類型的使用者（組織、租戶、管理員和應用程式使用者）、訂閱模型和租戶隔離模型
- 了解組織的多產品 SaaS 需求和多租戶訂閱

限制

- 此模式未涵蓋常見租用戶解決方案與個別 SaaS 產品之間的整合。
- 此模式只會在單一 AWS 區域中部署 Amazon Cognito 服務。

Architecture

目標技術堆疊

- Amazon API Gateway
- Amazon Cognito
- AWS CloudTrail
- Amazon CloudWatch
- Amazon DynamoDB
- IAM
- AWS Lambda
- Amazon Simple Storage Service (Amazon S3)
- Amazon Simple Notification Service (Amazon SNS)
- AWS Step 函數

目標架構

下圖顯示管理 AWS 雲端中單一控制平面上多個 SaaS 產品的租用戶生命週期的範例工作流程。

該圖顯示以下工作流程：

1. AWS 使用者透過呼叫 API Gateway 端點，啟動租戶佈建、產品佈建或管理相關動作。
2. 使用者透過從 Amazon Cognito 使用者集區或其他 IdP 擷取的存取字符進行身分驗證。
3. 個別佈建或管理任務是由與 API Gateway API 端點整合的 Lambda 函數執行。

4. 通用租戶解決方案APIs（適用於租戶、產品和使用者）會收集所有必要的輸入參數、標頭和字符。然後，管理 APIs會叫用相關聯的 Lambda 函數。
5. 管理 APIs和 Lambda 函數的 IAM 許可都由 IAM 服務驗證。
6. Lambda 函數會從 DynamoDB 和 Amazon S3 中的目錄（適用於租戶、產品和使用者）存放和擷取資料。
7. 驗證許可後，會叫用 AWS Step Functions 工作流程來執行特定任務。圖表中的範例顯示租用戶佈建工作流程。
8. 個別 AWS Step Functions 工作流程任務會在預先定義的工作流程（狀態機器）中執行。
9. 從 DynamoDB 或 Amazon S3 擷取執行與每個工作流程任務相關聯的 Lambda 函數所需的任何必要資料。可能需要使用 AWS CloudFormation 範本佈建其他 AWS 資源。
10. 如有需要，工作流程會傳送請求，為特定 SaaS 產品佈建額外的 AWS 資源到該產品的 AWS 帳戶。
11. 當請求成功或失敗時，工作流程會將狀態更新作為訊息發佈至 Amazon SNS 主題。
12. Amazon SNS 已訂閱 Step Functions 工作流程的 Amazon SNS 主題。
13. 然後，Amazon SNS 會將工作流程狀態更新傳回給 AWS 使用者。
14. 每個 AWS 服務動作的日誌，包括 API 呼叫的稽核線索，都會傳送至 CloudWatch。您可以在 CloudWatch 中為每個使用案例設定特定規則和警示。
15. 日誌會封存在 Amazon S3 儲存貯體中以供稽核之用。

自動化和擴展

此模式使用 CloudFormation 範本來協助自動化常見租用戶解決方案的部署。範本也可以協助您快速擴展或縮減相關聯的資源。

如需詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的使用 AWS CloudFormation 範本。

工具

AWS 服務

- Amazon API Gateway 可協助您建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。
- Amazon Cognito 為 Web 和行動應用程式提供身分驗證、授權和使用者管理。
- AWS CloudTrail 可協助您稽核 AWS 帳戶的控管、合規和營運風險。
- Amazon CloudWatch 可協助您即時監控 AWS 資源的指標，以及您在 AWS 上執行的應用程式。

- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務來建置業務關鍵應用程式。

最佳實務

此模式中的解決方案使用單一控制平面來管理多個租用戶的加入，以及佈建對多個 SaaS 產品的存取。控制平面可協助管理使用者管理其他四個功能特定的平面：

- 安全平面
- 工作流程平面
- 通訊平面
- 記錄和監控平面

史詩

設定安全平面

任務	Description	所需的技能
為您的多租戶 SaaS 平台建立需求。	建立下列項目的詳細需求： • 租用戶 • 使用者 • 角色 • SaaS 產品 • Subscriptions	雲端架構師、AWS 系統管理員

任務	Description	所需的技能
	• 設定檔交換	
設定 Amazon Cognito 服務。	請遵循《Amazon Cognito 開發人員指南》中的 Amazon Cognito 入門中的指示。	雲端架構師
設定所需的 IAM 政策。	為您的使用案例建立所需的 IAM 政策。然後，將政策映射至 Amazon Cognito 中的 IAM 角色。 如需詳細資訊，請參閱《Amazon Cognito 開發人員指南》中的使用政策和角色型存取控制管理存取權。	雲端管理員、雲端架構師、AWS IAM 安全性
設定所需的 API 許可。	使用 IAM 角色和政策以及 Lambda 授權方來設定 API Gateway 存取許可。 如需說明，請參閱《Amazon API Gateway 開發人員指南》中的下列章節： • 使用 IAM 許可控制對 API 的存取 • 使用 API Gateway Lambda 授權方	雲端管理員、雲端架構師

設定資料平面

任務	Description	所需的技能
建立所需的資料目錄。	1. 建立 DynamoDB 資料表來存放使用者目錄的資料。請務必包含使用者屬性和角	DBA

任務	Description	所需的技能
	色。此外，請確定您在目錄資料表上執行資料建模，以維護每個使用者和角色的必要和選用屬性。 2. 建立 DynamoDB 資料表來存放產品目錄的資料。請務必為 SaaS 產品建立特定使用案例的模型。 3. 建立 DynamoDB 資料表來存放租戶目錄的資料。請確定您為租戶、產品和多重 SaaS 訂閱和標籤的授權設定訂閱模型。 如需詳細資訊，請參閱《Amazon DynamoDB 開發人員指南》中的設定 DynamoDB。 DynamoDB	

設定控制平面

任務	Description	所需的技能
建立 Lambda 函數和 API Gateway APIs 以執行必要的控制平面任務。	建立個別的 Lambda 函數和 API Gateway APIs，以新增、刪除和管理下列項目： • 使用者 • 租用戶 • 產品 如需詳細資訊，請參閱《AWS Lambda 開發人員指南》中的	應用程式開發人員

任務	Description	所需的技能
	搭配 Amazon API Gateway 使用 AWS Lambda。 AWS Lambda	

設定工作流程平面

任務	Description	所需的技能
識別 AWS Step Functions 工作流程必須執行的任務。	識別並記錄下列項目的詳細 AWS Step Functions 工作流程需求： • 使用者 • 租用戶 • 產品 **Important** 確定主要利益相關者核准要求。	應用程式擁有者
建立所需的 AWS Step Functions 工作流程。	1. 在 AWS Step Functions 中為使用者、租戶和產品建立必要的工作流程。如需詳細資訊，請參閱 AWS Step Functions 開發人員指南。 2. 識別重試和錯誤處理機制。如需詳細資訊，請參閱 AWS 部落格上的處理錯誤、重試和新增提醒至 Step Function State Machines。 3. 使用 Lambda 函數實作工作流程步驟。如需說明，請參	應用程式開發人員、建置領導

任務	Description	所需的技能
	閱《AWS Step Functions Step Functions 開發人員指南》中的建立使用 Lambda 的 Step Functions 狀態機器。 4. 視需要整合任何外部服務與 AWS Step Functions。 5. 在 DynamoDB 資料表中維護每個工作流程的狀態，並使用 Amazon SNS 來傳達每個工作流程的狀態。	

設定通訊平面

任務	Description	所需的技能
建立 Amazon SNS 主題。	建立 Amazon SNS 主題以接收有關下列項目的通知： • 工作流程狀態 • 錯誤 • 重試 如需詳細資訊，請參閱《Amazon SNS SNS 開發人員指南》中的建立 SNS 主題。	應用程式擁有者、雲端架構師
訂閱端點至每個 Amazon SNS 主題。	若要接收發佈至 Amazon SNS 主題的訊息，您必須訂閱每個主題的端點。 如需詳細資訊，請參閱《Amazon SNS 開發人員指	應用程式開發人員、雲端架構師

任務	Description	所需的技能
	南》中的訂閱 Amazon SNS 主題。 Amazon SNS	

設定記錄和監控平面

任務	Description	所需的技能
為常用租用戶解決方案的每個元件啟用記錄。	為您建立的通用租用戶解決方案中的每個資源在元件層級啟用記錄。 如需詳細說明，請參閱下列主題： • 如何開啟 CloudWatch Logs 進行 API Gateway REST API 或 WebSocket API 故障診斷？ (AWS 知識中心) • 使用 CloudWatch Logs 記錄 (AWS Step Functions 開發人員指南) • Python 中的 AWS Lambda 函數記錄 (AWS Lambda 開發人員指南) • 在 Amazon Cognito 中記錄和監控 (Amazon Cognito 開發人員指南) • 使用 Amazon CloudWatch 進行監控 (Amazon DynamoDB 開發人員指南)	應用程式開發人員、AWS 系統管理員、雲端管理員

任務	Description	所需的技能
	Note 您可以使用 IAM 政策，將每個資源的日誌合併到集中式日誌帳戶。如需詳細資訊，請參閱集中式記錄和多帳戶安全護欄。	

佈建和部署通用租戶解決方案

任務	Description	所需的技能
建立 CloudFormation 範本。	使用 CloudFormation 範本，自動化完整通用租用戶解決方案及其所有元件的部署和維護。 如需詳細資訊，請參閱 AWS CloudFormation 使用者指南。	應用程式開發人員、DevOps 工程師、CloudFormation 開發人員

相關資源

- 使用 Amazon Cognito 使用者集區做為授權方控制對 REST API 的存取 (Amazon API Gateway 開發人員指南)
- 使用 API Gateway Lambda 授權方 (Amazon API Gateway 開發人員指南)
- Amazon Cognito 使用者集區 (Amazon Cognito 開發人員指南)
- 跨帳戶跨區域 CloudWatch 主控台 (Amazon CloudWatch 使用者指南)

使用與靜態 IP 地址相關聯的端點，整合 Amazon S3 預先簽章的 URL 產生和物件下載

Song Jin、Eunhye Jo 和 Amazon Web Services Jun Soung Lee

總結

此模式透過為物件下載建立安全的自訂預先簽章 URLs，簡化對 Amazon Simple Storage Service (Amazon S3) 的存取。解決方案提供具有唯一網域和靜態 IP 地址的單一端點。它專為需要在具有靜態 IP 地址的統一網域下整合 API 和 Amazon S3 端點的客戶量身打造。使用案例涉及遵循 IP 和網域允許清單防火牆政策的使用者，限制對特定網域和 IP 地址的 API 存取。

架構採用金鑰 AWS 服務，包括 AWS Global Accelerator Amazon API Gateway、 AWS Lambda Application Load Balancer AWS PrivateLink和 Amazon S3。此設計集中在單一網域下產生預先簽章 URLs和 Amazon S3 端點的 API，並連結至具有兩個靜態 IP 地址的 加速器。因此，使用者可以輕鬆地請求預先簽章URLs，並透過具有靜態 IP 地址的統一網域端點下載 Amazon S3 物件。

此架構特別有利於具有嚴格政策或合規要求的客戶，例如公有、醫療和金融領域的客戶。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 自訂網域名稱的公有託管區域
- 在您選擇的 AWS Certificate Manager (ACM) 中匯入 AWS 區域 的網域

限制

- Amazon S3 儲存貯體名稱必須符合端點的網域名稱。此要求旨在確保可透過單一 API 端點提供 Amazon S3 端點。
- API Gateway 中使用的自訂網域名稱應與單一 API 端點的網域名稱相符。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

下圖顯示此模式的目標架構和工作流程。

圖表說明下列概念和工作流程：

1. 使用者使用透過 提供的自訂端點 AWS Global Accelerator，使用自訂網域名稱和相關聯的 IP 地址，啟動產生預先簽章 URL 的請求。
2. Lambda 函數會產生預先簽章的 URL，指向自訂端點。它會以 301 重新導向回應，其中包含產生的預先簽章 URL。透過重新導向的預先簽章 URL，使用者會使用透過 Global Accelerator 提供的自訂端點自動下載物件。

預先簽章 URL 產生和物件下載工作流程的整體架構元件如下：

- 由 Global Accelerator 佈建靜態 IP 地址。
- 使用自訂網域名稱，將加速器的別名註冊為 Amazon Route 53 公有託管區域的 A 記錄。
- 建立儲存貯體名稱符合已註冊自訂網域名稱的 Amazon S3 儲存貯體。
- 建立 API Gateway 和 Amazon S3 服務的 VPC 端點。
- 要連線至 Global Accelerator 的內部 Application Load Balancer 組態。
- 指派已連接 ACM 憑證之 API Gateway 的自訂網域名稱。
- 部署與 Lambda 函數整合的私有 API Gateway。
- Lambda 函數配備連接的 AWS Identity and Access Management (IAM) 角色（具有 GetObject 許可）。

工具

AWS 服務

- Amazon API Gateway 可協助您建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。
- Application Load Balancer 會將傳入的應用程式流量分散到多個可用區域中的多個目標，例如 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。
- AWS Certificate Manager (ACM) 可協助您建立、存放和續約公有和私有 SSL/TLS X.509 憑證和金鑰，以保護 AWS 網站和應用程式。
- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端 基礎設施。

- AWS Global Accelerator 是一種全域服務，支援多個 中的端點 AWS 區域。您可以建立加速器，透過 AWS 全球網路將流量導向最佳端點。如此可改善網際網路應用程式的可用性和效能，提升全球觀眾的使用體驗。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- AWS PrivateLink 可協助您建立從虛擬私有雲端 (VPCs) 到 VPC 外部服務的單向私有連線。
- Amazon Route 53 是一種可用性高、可擴展性強的 DNS Web 服務。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他工具

- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。

程式碼儲存庫

您可以根據您的偏好，使用 AWS CDK 或 Terraform 部署此模式。Epics 區段包含兩種部署方法的說明。此模式的程式碼可在下列 GitHub 儲存庫中使用：

- AWS CDK – s3-presignedurl-staticips-endpoint-with-cdk
- Terraform – s3-presignedurl-staticips-endpoint-with-terraform

最佳實務

- 為了增強生產環境中的安全性，請務必實作授權機制，例如 Amazon Cognito，以限制對`PresignedUrl`世代 API 的存取。
- 遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。

史詩

準備環境

任務	Description	所需的技能
決定網域名稱。	決定統一 Amazon S3 端點的公有網域名稱。網域名稱也會用作 Amazon S3 儲存貯體名稱。	AWS 管理員、網路管理員
建立公有的託管區域。	在 Amazon Route 53 中建立公有託管區域。其網域名稱必須與 API Gateway 中使用的網域名稱相符。	AWS 管理員、網路管理員
準備 SSL 憑證。	使用 AWS Certificate Manager (ACM) 為您的 Web 應用程式網域請求或匯入 SSL 憑證。	AWS 管理員、網路管理員

使用 Terraform 部署模式

任務	Description	所需的技能
設定 Terraform 開發環境。	若要設定開發環境，請執行下列動作： 1. 安裝 Terraform 1.0 版或更新版本。 2. 在終端機視窗中執行下列命令，複製 GitHub s3-presignedurl-staticips-endpoint-with-terraform 儲存庫： `git clone https://github.com/aws-samples/s3-presignedur`	AWS 管理員、雲端管理員

任務	Description	所需的技能
	l-staticips-endpoint-with-terraform.git	

任務	Description	所需的技能

修改 .tfvars和 provider.tf 檔案。	1. 建立必要.tfvars檔案 – 在 1.vpc_alb_ga 和 2.apigw_s3_lambda 目錄中建立apg.tfvars 檔案。這些檔案將包含您的環境特定變數值。
	• 針對 1.vpc_alb_ga/apg.tfvars ，使用下列範本建立檔案：
	`hcl`
	`environment_name = "test"`
	`domain = "your-domain.com"`
	`s3_bucket_prefix = "your-bucket-prefix"`
	`aws_region = "ap-northeast-2"`
	`# Change to your preferred region`
	• 對於 2.apigw_s3_lambda/apg.tfvars ，使用下列範本建立檔案（在完成步驟 1 vpc_id後，您將需要更新）：
	`hcl`
	`vpc_id = "vpc-xxxxxxxxx"`
	`# Update this with the VPC ID output from step 1`
	`environment_name = "test"`
		AWS 管理員、雲端管理員

任務	Description	所需的技能
	domain = "your-domain.com" s3_bucket_prefix = "your-bucket-prefix" aws_region = "ap-northeast-2" # Change to your preferred region 2. 修改provider.tf 檔案 – 在 1.vpc_alb_ga 和 2.apigw_s3_lambda 資料夾中，編輯provider.tf 檔案以符合您的本機 AWS 組態： hcl provider "aws" { region = "ap-northeast-2" # Change to your preferred region profile = "apg" # Change to your AWS profile name } 注意下列事項： • 將.tfvars檔案中的預留位置值取代為您的實際值。 • 完成第一個 Terraform 部署後 (1.vpc_alb_ga)，請使用輸出2.apigw_s3_lambda/apg.tfvars 中的 VPC ID 更新 中的vpc_id值。	

任務	Description	所需的技能
	• 請確定檔案中的 AWS 設定檔名稱與`~/.aws/credentials` 檔案中現有的設定檔`provider.tf` 相符。 • 網域值應與您的 Route 53 託管區域網域相符。 • 與網域`s3_bucket_prefix` 結合的 將形成您的 Amazon S3 儲存貯體名稱（例如 `your-bucket-prefix.your-domain.com`)。 • 如果您變更 `aws_region` ，您可能還需要更新`main.tf`檔案中的硬式編碼可用區域，以符合您選擇的可用區域 AWS 區域。	

佈建網路資源。	若要佈建網路資源，請執行下列命令：
	`cd ./2.vpc_alb_ga`
	`terraform init`
	`terraform plan --var-file=apg.tfvars`
	`terraform apply --var-file=apg.tfvars`
	在`apply` 命令執行期間，出現提示時輸入 yes。
		AWS 管理員、雲端管理員

任務	Description	所需的技能

佈建 API Gateway、Amazon S3 和 Lambda。	若要佈建網路資源，請使用下列命令：
	`cd ./2.apigw_s3_lambda`
	`terraform init`
	`terraform plan --var-file=apg.tfvars`
	`terraform apply --var-file=apg.tfvars`
		AWS 管理員、雲端管理員

使用 部署模式 AWS CDK

任務	Description	所需的技能
設定 AWS CDK 開發環境。	若要設定開發環境，請執行下列動作： 1. 安裝 AWS CDK。 2. 在終端機視窗中執行下列命令，複製 GitHub s3-presignedurl-staticips-endpoint-with-cdk 儲存庫： `git clone https://github.com/aws-samples/s3-presignedurl-staticips-endpoint-with-cdk.git`	AWS 管理員、雲端管理員

在 `config/index.ts` 檔案中設定網域設定。	若要編輯常數變數的選項，請使用下列命令：
	`export const options = {`
	`certificateArn: '{arn of the acm which created before}',`
		AWS 管理員、雲端管理員

任務	Description	所需的技能

	<pre>dnsAttr: {
	zoneName:
	'{public hosted zone
	name}',
	hostedZoneId:
	'hosted zone Id',
	},
	domainNamePrefix:
	'{Prefix for the
	domain}',
	presignPath:
	'presign',
	objectsPath:
	'objects',
	};</pre>
	在命令中，將每個預留位置取代為您自己的資訊：
	• '{arn of the acm which created before}' 將 取代為您的 ACM 憑證 Amazon Resource Name (ARN)。
	• 將 取代'{public hosted zone name}'為您的網域名稱。
	• 'hosted zone Id' 將 取代為您的 Route 53 託管區域 ID。
	• '{Prefix for the domain}' 以您要使用的網域字首取代 。
	• 或者，如果特定使用案例需要，請修改 'presign' 和 'objects' 路徑。	

任務	Description	所需的技能

部署堆疊。	若要部署兩個堆疊，一個用於虛擬私有雲端 (VPC)，另一個用於應用程式，請使用下列命令：
	`$ npm install`
	`$ cdk synth`
	`$ cdk deploy --all`
		AWS 管理員、雲端管理員

測試模式

任務	Description	所需的技能
驗證端點的 IP 地址。	若要驗證此模式的網域具有靜態 IP 地址，請使用下列命令： `nslookup ${s3-bucket-prefix}.${domain}`	網路管理員
上傳您稍後可以下載的測試檔案。	將測試檔案上傳至 Amazon S3 儲存貯體中的 `'/objects'` 資料夾。	AWS 管理員、雲端管理員
叫用 API 以產生預先簽章的 URL。	若要產生預先簽章的 URL，請使用下列格式從瀏覽器或 API 用戶端 （例如 Postman) 呼叫 URL： `https://${s3-bucket-prefix}.${domain}/presign/objects/${uploaded-filename}` 將 `${s3-bucket-prefix}` 和 中的預留位置值取代為您	應用程式擁有者

任務	Description	所需的技能
	在先前步驟中設定${domain}的值。	
檢查結果。	預期的結果是您應該會收到 301（移動永久）重新導向狀態碼。此回應將包含預先簽章的 URL，應會自動啟動測試檔案的下載。	測試工程師

使用 Terraform 清除

任務	Description	所需的技能

銷毀 API Gateway、Amazon S3 和 Lambda 資源。	若要刪除資源，請使用下列命令：
	`cd ./2.apigw_s3_lambda`
	`terraform init`
	`terraform plan --destroy --var-file=apg.tfvars`
	`terraform destroy --var-file=apg.tfvars`
		AWS 管理員、雲端管理員

銷毀網路資源。	若要刪除網路資源，請使用下列命令：
	`cd ./1.vpc_alb_ga`
	`terraform init`
	`terraform plan --destroy --var-file=apg.tfvars`
	`terraform destroy --var-file=apg.tfvars`
		AWS 管理員、雲端管理員

使用 清除 AWS CDK

任務	Description	所需的技能
銷毀堆疊。	若要同時銷毀 VPC 和應用程式堆疊，請使用下列命令： `$ cdk destroy --all`	AWS 管理員、雲端管理員

清空並刪除 Amazon S3 儲存貯體。	清空並刪除物件 Amazon S3 儲存貯體，以及預設不會刪除的日誌 Amazon S3 儲存貯體。
	Amazon S3 儲存貯體名稱為 `${s3-bucket-prefix}.${domain}` 和 `${s3-bucket-prefix}.${domain}-logs` 。
	如果您偏好使用 AWS Command Line Interface (AWS CLI) 刪除儲存貯體，請使用下列命令：
	`$ aws s3 rm s3://${s3-bucket-prefix}.${domain} --recursive`
	`$ aws s3 rb s3://${s3-bucket-prefix}.${domain} --force`
	`$ aws s3 rm s3://${s3-bucket-prefix}.${domain}-logs --recursive`
	`$ aws s3 rb s3://${s3-bucket-prefix}.${domain}-logs --force`
		AWS 管理員、雲端管理員

任務	Description	所需的技能
	將 `${s3-bucket-prefix}` 和 取代`${domain}` 為您在先前步驟中設定的值。 , /p>	

相關資源

AWS 部落格

- 透過 提供的靜態 IP 地址存取 Amazon API Gateway AWS Global Accelerator
- 在 JavaScript 的模組化中產生預先簽章 AWS CDK 的 URL JavaScript
- 使用 ALB、S3 和 PrivateLink 託管內部 HTTPS 靜態網站

在組織中建立跨帳戶 Amazon EventBridge 連線

Sam Wilson 和 Robertstone，Amazon Web Services

總結

大型分散式系統使用 Amazon EventBridge 來傳達 AWS Organizations 組織中各種 Amazon Web Services (AWS) 帳戶之間的狀態變更。不過，EventBridge 通常只能以相同 中的端點或取用者為目標 AWS 帳戶。例外狀況是不同帳戶中的事件匯流排。該事件匯流排是有效的目標。若要使用來自另一個帳戶中事件匯流排的事件，必須將事件從來源帳戶的事件匯流排推送到目的地帳戶的事件匯流排。為了避免在不同應用程式中管理關鍵事件時遇到挑戰 AWS 帳戶，請使用此模式中顯示的建議方法。

此模式說明如何使用涉及 AWS 帳戶 AWS Organizations 組織中多個 的 EventBridge 實作事件驅動型架構。模式使用 AWS Cloud Development Kit (AWS CDK) Toolkit 和 AWS CloudFormation。

EventBridge 提供無伺服器事件匯流排，可協助您接收、篩選、轉換、路由和交付事件。EventBridge 是事件驅動架構的關鍵元件，支援訊息生產者與這些訊息消費者之間的區隔。在單一帳戶中，這是直接的。多帳戶結構需要一個帳戶中事件匯流排上的事件的額外考量，才能在相同組織內的其他帳戶中使用。

如需生產者和消費者帳戶特定考量的相關資訊，請參閱其他資訊一節。

先決條件和限制

先決條件

- 至少有兩個關聯的 AWS Organizations 組織 AWS 帳戶
- 兩者中的 AWS Identity and Access Management (IAM) 角色 AWS 帳戶 ，可讓您 AWS 帳戶 使用在兩者中佈建基礎設施 AWS CloudFormation
- 本機安裝的 Git
- AWS Command Line Interface 在本機安裝 (AWS CLI)
- 在AWS CDK 本機安裝 ，並在兩者中引導 AWS 帳戶

產品版本

此模式已使用下列工具和版本建置和測試：

- AWS CDK 工具組 2.126.0

- Node.js 18.19.0
- npm 10.2.3
- Python 3.12

此模式應適用於任何版本的 AWS CDK v2 或 npm。Node.js 13.0.0 到 13 AWS CDK.6.0 版與 不相容。

Architecture

目標架構

下圖顯示從一個帳戶推送事件並在另一個帳戶中使用事件的架構工作流程。

工作流程包含下列步驟：

1. 來源帳戶中的生產者 AWS Lambda 函數會在帳戶的 EventBridge 事件匯流排上放置事件。
2. 跨帳戶 EventBridge 規則會將事件路由到目的地帳戶中的 EventBridge 事件匯流排。
3. 目的地帳戶中的 EventBridge 事件匯流排具有目標 Lambda 規則，可叫用 Consumer Lambda 函數。

最佳實務是使用無效字母佇列 (DLQ) 來處理消費者 Lambda 函數的失敗調用。不過，為了清楚起見，此解決方案省略了 DLQ。若要進一步了解如何在工作流程中實作 DLQ，並改善工作流程從失敗中復原的能力，請參閱實作 AWS Lambda 錯誤處理模式部落格文章。

自動化和擴展

AWS CDK 會自動佈建所需的架構。EventBridge 可以根據 擴展到每秒數千筆記錄 AWS 區域。如需詳細資訊，請參閱 Amazon EventBridge 配額文件。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端 基礎設施。此模式使用 AWS CDK Toolkit，這是一個命令列雲端開發套件，可協助您與 AWS CDK 應用程式互動。

- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如， AWS Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他事件匯流排 AWS 帳戶。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。

其他工具

- Node.js 是一種事件驅動的 JavaScript 執行期環境，旨在建置可擴展的網路應用程式。
- npm 是在 Node.js 環境中執行的軟體登錄檔，用於共用或借用套件和管理私有套件的部署。
- Python 是一種一般用途的電腦程式設計語言。

程式碼儲存庫

此模式的程式碼可在 GitHub cross-account-eventbridge-in-organization 儲存庫中使用。

最佳實務

如需使用 EventBridge 時的最佳實務，請參閱下列資源：

- Amazon EventBridge 事件模式的最佳實務
- 在 Amazon EventBridge 中定義規則時的最佳實務

史詩

準備您的本機 AWS CDK 部署環境

任務	Description	所需的技能
設定來源帳戶和目的地帳戶的本機登入資料。	檢閱設定新的組態和登入資料，並使用對您的環境最有意義的身分驗證和登入資料方法。	應用程式開發人員

任務	Description	所需的技能
	Important 請務必 AWS CLI 為來源帳戶和目的地帳戶身分驗證設定 。 這些指示假設您已在本機設定兩個 AWS 設定檔： `sourceAccount` 和 `destinationAccount` 。	

同時引導兩者 AWS 帳戶。	若要引導帳戶，請執行下列命令：
	
	`cdk bootstrap --profile sourceAccount`
	`cdk bootstrap --profile destinationAccount`
		應用程式開發人員

複製模式程式碼。	若要複製儲存庫，請執行下列命令：
	
	`git clone git@github.com:aws-samples/aws-cdk-examples.git`
	
	然後，將目錄變更為新複製的專案資料夾：
	
	`cd aws-cdk-examples/python/cross-account-eventbridge-in-organization`
		應用程式開發人員

將 ProducerStack 部署至來源帳戶

任務	Description	所需的技能
cdk.json 使用 AWS Organizations 和 帳戶詳細資訊修改 。	在專案的根資料夾中，對 進行下列變更cdk.json： • organization_id – 部署所涉及帳戶的 Organizations ID • event_bus_name – CrossAccount 或您可以重新命名 • rules[].targets[].arn – 耗用帳戶的 AWS 帳戶 ID （目的地帳戶）	應用程式開發人員
部署 ProducerStack 資源。	從專案的根目錄執行下列命令： `cdk deploy ProducerStack --profile sourceAccount` 出現提示時，請接受透過 建立的新 IAM 角色和其他安全相關許可 AWS CloudFormation。	應用程式開發人員
確認已部署 ProducerStack 資源。	若要驗證資源，請執行下列動作： 1. 在來源帳戶的 AWS 管理主控台 上，選擇 CloudFormation。 2. 從堆疊清單中，選擇 ProducerStack。	應用程式開發人員

任務	Description	所需的技能
	3. 在堆疊資訊索引標籤上，確認堆疊狀態為 CREATE_COMPLETE 。或者，在資源索引標籤上，檢閱設定的資源。	

將 ConsumerStack 部署至目的地帳戶

任務	Description	所需的技能
部署 ConsumerStack 資源。	從專案的根目錄執行下列命令： `cdk deploy ConsumerStack --profile destinationAccount` 出現提示時，請接受透過 建立的新 IAM 角色和其他安全相關許可 CloudFormation。	應用程式開發人員
確認已部署 ConsumerStack 資源	1. 在目的地帳戶的 主控台上，選擇 CloudFormation。 2. 從堆疊清單中，選擇 ConsumerStack。 3. 在堆疊資訊索引標籤上，確認堆疊狀態為 CREATE_COMPLETE 。或者，在資源索引標籤上，檢閱設定的資源。	應用程式開發人員

產生和使用事件

任務	Description	所需的技能
叫用 Producer Lambda 函數。	1. 在來源帳戶的 主控台上，選擇 Lambda。 2. 從函數清單中，選擇 ProducerStack-ProducerLambdaXXXX (XXXX 代表 AWS CDK 自動產生的字元序列）。 3. 選擇測試標籤。 4. 在測試事件區段中，選擇測試。 Event JSON 文字區域內容可以是提供給 Lambda 函數作為承載的任何有效 JSON。在此情況下，預設提供的 JSON 已足夠。 5. 確認執行函數：成功訊息出現在測試事件區段上方的綠色橫幅中。	應用程式開發人員
確認已收到事件。	1. 在目的地帳戶的 主控台上，選擇 Lambda。 2. 從函數清單中，選擇 ConsumerStack-ConsumerLambdaXXXX (XXXX 代表 AWS CDK 自動產生的字元序列）。 3. 選擇 Monitor (監控) 索引標籤。 4. 在監控區段中，選擇檢視 CloudWatch 日誌。	應用程式開發人員

任務	Description	所需的技能
	5. 在新開啟的索引標籤中，選擇最新日誌串流的日誌串流名稱。 6. 確認出現如下所示的日誌陳述式： `[DEBUG] 2024-04-08T19:08:10.091Z 9c16844a-f9de-444d-b621-86afe64d4cc8 Event: {'version':'0', 'id':'0b9faa96-973f-8be2-ecf8-75e4f328b980', 'detail-type':'TestType', 'source':'Producer', 'account':'XXXXXXXXXXXX', 'time':'2024-04-08T19:08:09Z', 'region':'us-east-1', 'resources': [], 'detail': {'key1':'value1', 'key2':'value2', 'key3':'value3'}}`	

清除

任務	Description	所需的技能
銷毀 ConsumerStack 資源。	如果您使用此模式做為測試，請清除已部署的資源，以避免產生額外費用。 從專案的根目錄執行下列命令： `cdk destroy ConsumerStack --profile destinationAccount` 系統會提示您確認刪除堆疊。	應用程式開發人員
銷毀 ProducerStack 資源。	從專案的根目錄執行下列命令： `cdk destroy ProducerStack --profile sourceAccount` 系統會提示您確認刪除堆疊。	應用程式開發人員

疑難排解

問題	解決方案
目的地帳戶中未收到任何事件。	1. 驗證提供的 Organizations ID 是否正確。 2. 確認來源帳戶是所提供組織的一部分。 3. 確認來源帳戶中的事件匯流排規則對應至目的地帳戶中的正確資訊。
從主控台叫用 Lambda 函數會傳回下列錯誤： `User: arn:aws:sts::111122223333:assumed-role/ProducerStack-ProducerLambdaServiceRoleXXXX is not authorized to perform: lambda:InvokeFunction on resource: arn:aws:lambda:us-east-1:111122223333:function:ProducerStack-ProducerLambdaXXXX`	請聯絡您的 AWS 帳戶 管理員，以取得 `ProducerStack-ProducerLambdaXXXX`

問題	解決方案
`User: arn:aws:iam::123456789012:user/XXXXX is not authorized to perform: lambda:Invoke`	Lambda 函數的適當`lambda:Invoke` 動作許可。

相關資源

參考

- AWS Organizations 使用者指南
- Amazon EventBridge 事件模式
- Amazon EventBridge 中的規則

教學課程和影片

- 教學課程：建立和設定組織
- AWS re：Invent 2023 - 使用 Amazon EventBridge (COM301-R) 的進階事件驅動模式

其他資訊

製作者規則

在來源帳戶中，建立 EventBridge 事件匯流排以接受來自生產者的訊息（如架構一節所示）。在此事件匯流排上建立具有隨附 IAM 許可的規則。這些規則會根據下列`cdk.json`結構，以目的地帳戶中的 EventBridge 事件匯流排為目標：

```
"rules": [
  {
    "id": "CrossAccount",
    "sources": ["Producer"],
    "detail_types": ["TestType"],
    "targets": [
      {
        "id": "ConsumerEventBus",
        "arn": "arn:aws:events:us-east-2:012345678901:event-bus/CrossAccount"
      }
    ]
```

```
  }
]
```

對於每個耗用事件匯流排，必須包含事件模式和目標事件匯流排。

事件模式

事件模式會篩選此規則將套用的事件。基於此範例，事件來源和記錄會`detail_types`識別要從來源帳戶的事件匯流排傳輸哪些事件到目的地帳戶的事件匯流排。

目標事件匯流排

此規則以另一個帳戶中存在的事件匯流排為目標。需要完整 `arn`(Amazon Resource Name) 才能唯一識別目標事件匯流排，而 `id`是 使用的邏輯 ID AWS CloudFormation。建立目標規則時，目標事件匯流排實際上不需要存在。

目的地帳戶特定的考量事項

在目的地帳戶中，會建立 EventBridge 事件匯流排，以從來源帳戶的事件匯流排接收訊息。若要允許從來源帳戶發佈事件，您必須建立以資源為基礎的政策：

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowOrgToPutEvents",
    "Effect": "Allow",
    "Principal": "*",
    "Action": "events:PutEvents",
    "Resource": "arn:aws:events:us-east-2:012345678901:event-bus/CrossAccount",
    "Condition": {
      "StringEquals": {
        "aws:PrincipalOrgID": "o-XXXXXXXXX"
      }
    }
  }]
}
```

特別重要的是授予`events:PutEvents`許可，允許同一組織中的任何其他帳戶發佈事件到此事件匯流排。將 `aws:PrincipalOrgId`設定為組織 ID 會授予所需的許可。

事件模式

您可以修改包含的事件模式，以符合您的使用案例：

```
rule = events.Rule(
    self,
    self.id + 'Rule' + rule_definition['id'],
    event_bus=event_bus,
    event_pattern=events.EventPattern(
        source=rule_definition['sources'],
        detail_type=rule_definition['detail_types'],
    )
)
```

為了減少不必要的處理，事件模式應指定只有目的地帳戶要處理的事件才會傳輸到目的地帳戶的事件匯流排。

以資源為基礎的政策

此範例使用組織 ID 來控制允許哪些帳戶在目的地帳戶的事件匯流排上放置事件。考慮使用更嚴格的政策，例如指定來源帳戶。

EventBridge 配額

請記住下列配額：

- 每個事件匯流排 300 個規則是預設配額。這可以視需要擴展，但應該適合大多數的使用案例。
- 每個規則允許五個目標。我們建議應用程式架構師為每個目的地帳戶使用不同的規則，以支援對事件模式的精細控制。

使用 Kinesis Data Streams 和 Firehose 搭配 將 DynamoDB 記錄交付至 Amazon S3 AWS CDK

Shashank Shrivastava 和 Daniel Matuki da Cunha，Amazon Web Services

摘要

此模式提供範例程式碼和應用程式，以使用 Amazon Kinesis Data Streams 和 Amazon Data Firehose，將記錄從 Amazon DynamoDB 交付至 Amazon Simple Storage Service (Amazon S3)。模式的方法使用 AWS Cloud Development Kit (AWS CDK) L3 建構，並包含如何在資料交付至 Amazon Web Services (AWS) 雲端上的目標 S3 儲存貯體 AWS Lambda 之前使用 執行資料轉換的範例。

Kinesis Data Streams 會在 DynamoDB 資料表中記錄項目層級修改，並將其複寫至所需的 Kinesis 資料串流。您的應用程式可以存取 Kinesis 資料串流，並以近乎即時的速度檢視項目層級的變更。Kinesis Data Streams 也提供其他 Amazon Kinesis 服務的存取權，例如 Firehose 和 Amazon Managed Service for Apache Flink。這表示您可以建置應用程式，以提供即時儀表板、產生提醒、實作動態定價和廣告，以及執行複雜的資料分析。

您可以將此模式用於資料整合使用案例。例如，運輸車輛或工業設備可以將大量資料傳送至 DynamoDB 資料表。然後，這些資料可以轉換並存放在 Amazon S3 中託管的資料湖中。然後，您可以使用 Amazon Athena、Amazon Redshift Spectrum、Amazon Rekognition 和 等無伺服器服務來查詢和處理資料，並預測任何潛在的瑕疵 AWS Glue。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- AWS Command Line Interface (AWS CLI)，已安裝並設定。如需詳細資訊，請參閱 AWS CLI 文件中的 入門 AWS CLI。
- Node.js (18.x+) 和 npm，已安裝並設定。如需詳細資訊，請參閱 npm 文件中的下載並安裝 Node.js 和 npm。
- aws-cdk (2.x+)，已安裝並設定。如需詳細資訊，請參閱 AWS CDK 文件中的 入門 AWS CDK。
- GitHub aws-dynamodb-kinesisfirehose-s3-ingestion 儲存庫，在您的本機電腦上複製和設定。
- DynamoDB 資料表的現有範例資料。資料必須使用下列格式： `{"SourceDataId": {"S": "123"},"MessageData":{"S": "Hello World"}}`

Architecture

下圖顯示使用 Kinesis Data Streams 和 Firehose 將記錄從 DynamoDB 交付至 Amazon S3 的範例工作流程。

該圖顯示以下工作流程：

1. 使用 Amazon API Gateway 做為 DynamoDB 的代理來擷取資料。您也可以使用任何其他來源將資料擷取至 DynamoDB。
2. 項目層級變更會在 Kinesis Data Streams 中以近乎即時的方式產生，以交付至 Amazon S3。
3. Kinesis Data Streams 會將記錄傳送至 Firehose 以進行轉換和交付。
4. Lambda 函數會將記錄從 DynamoDB 記錄格式轉換為 JSON 格式，其中僅包含記錄項目屬性名稱和值。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。
- AWS CDK Toolkit 是命令列雲端開發套件，可協助您與 AWS CDK 應用程式互動。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。

程式碼儲存庫

此模式的程式碼可在 GitHub aws-dynamodb-kinesisfirehose-s3-ingestion 儲存庫中使用。

史詩

設定範例程式碼

任務	Description	所需的技能

安裝相依性。	在本機電腦上，執行下列命令，從 pattern/aws-dynamodb-kinesisstreams-s3 和 sample-application 目錄中package.json 的檔案安裝相依性：
	`cd <project_root>/pattern/aws-dynamodb-kinesisstreams-s3`
	`npm install && npm run build`
	`cd <project_root>/sample-application/`
	`npm install && npm run build`
		應用程式開發人員，一般 AWS

產生 CloudFormation 範本。	1. 執行 cd <project_root>/sample-application/ 命令。 2. 執行 cdk synth命令來產生 CloudFormation 範本。 3. AwsDynamodbKinesisfirehoseS3IngestionStack.template.js	應用程式開發人員、一般 AWS、AWS DevOps

任務	Description	所需的技能
	on 輸出會存放在 cdk.out目錄中。 4. 使用 AWS CDK 或 AWS 管理主控台 在 CloudFormation 中處理範本。	

部署 資源

任務	Description	所需的技能
檢查並部署 資源。	1. 執行 cdk diff命令以識別 AWS CDK 建構所建立的資源類型。 2. 執行 cdk deploy命令來部署資源。	應用程式開發人員、一般 AWS、AWS DevOps

將資料擷取至 DynamoDB 資料表以測試解決方案

任務	Description	所需的技能
將範例資料擷取至 DynamoDB 資料表。	在 中執行下列命令，將請求傳送至 DynamoDB 資料表 AWS CLI： `aws dynamodb put-item --table-name <your_table_name> --item '{"<table_partition_key>": {"S": "<partition_key_ID>"},"MessageData":{"S": "<data>"}}'`	應用程式開發人員

任務	Description	所需的技能
	範例： `aws dynamodb put-item --table-name SourceData_table --item '{"SourceDataId": {"S": "123"},"MessageData":{"S": "Hello World"}}'` 根據預設，如果操作成功，`put-item`不會傳回任何值做為輸出。如果操作失敗，則會傳回錯誤。資料存放在 DynamoDB 中，然後傳送至 Kinesis Data Streams 和 Firehose。 **Note** 您可以使用不同的方法來將資料新增至 DynamoDB 資料表。如需詳細資訊，請參閱 DynamoDB 文件中的將資料載入資料表。	
確認已在 S3 儲存貯體中建立新的物件。	登入 AWS 管理主控台 並監控 S3 儲存貯體，以確認已使用您傳送的資料建立新物件。 如需詳細資訊，請參閱 Amazon S3 文件中的 GetObject。	應用程式開發人員，一般 AWS

清除資源

任務	Description	所需的技能
清除資源。	執行 `cdk destroy`命令來刪除此模式使用的所有資源。	應用程式開發人員，一般 AWS

相關資源

- s3-static-site-stack.ts (GitHub 儲存庫）
- aws-apigateway-dynamodb 模組 (GitHub 儲存庫）
- aws-kinesisstreams-kinesisfirehose-s3 模組 (GitHub 儲存庫）
- 變更 DynamoDB Streams 的資料擷取 (DynamoDB 文件）
- 使用 Kinesis Data Streams 擷取對 DynamoDB 的變更 (DynamoDB 文件）

在 Amazon API Gateway 中使用自訂網域實作路徑型 API 版本控制

Corey Schnedl、Marcel Barbosa、Mario Lopez Martinez、Anbazhagan Ponnuswamy、Gaurav Samudra 和 Abhilash Vinod、Amazon Web Services

摘要

此模式示範如何使用自訂網域的 API 映射功能，為 Amazon API Gateway 實作以路徑為基礎的 API 版本控制解決方案。

Amazon API Gateway 是一項全受管服務，可用於建立、發佈、維護、監控和保護任何規模APIs。透過使用服務的自訂網域功能，您可以建立更簡單的自訂網域名稱，並提供更直覺URLs 給您的 API 使用者。您可以使用 API 映射將 API 階段連線至自訂網域名稱。建立網域名稱並設定 DNS 記錄之後，您可以使用 API 映射，透過自訂網域名稱將流量傳送至您的 API。

在 API 公開可用之後，消費者會使用它。隨著公有 API 的演進，其服務合約也會演進以反映新功能。不過，變更或移除現有功能並不明智。任何中斷變更都可能會影響消費者的應用程式，並在執行時間中斷它們。API 版本控制對於避免破壞回溯相容性和破壞合約非常重要。

您需要明確的 API 版本控制策略，以協助消費者採用這些策略。使用路徑型 URLs版本控制 APIs 是最直接且常用的方法。在此類型的版本控制中，版本會明確定義為 API URIs的一部分。下列範例 URLs 顯示消費者如何使用 URI 為其請求指定 API 版本：

```
https://api.example.com/api/v1/orders
```

```
https://api.example.com/api/v2/orders
```

```
https://api.example.com/api/vX/orders
```

此模式使用 AWS Cloud Development Kit (AWS CDK) 為您的 API 建置、部署和測試可擴展路徑型版本控制解決方案的範例實作。 AWS CDK 是一種開放原始碼軟體開發架構，可使用熟悉的程式設計語言來建模和佈建雲端應用程式資源。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 需要擁有網域才能使用此模式的範例儲存庫，以及使用 Amazon API Gateway 自訂網域功能。您可以使用 Amazon Route 53 為您的組織建立和管理網域。如需有關如何使用 Route 53 註冊或轉移網域的資訊，請參閱 Route 53 文件中的註冊新網域。

- 在設定 API 的自訂網域名稱之前，您必須備妥 SSL/TLS 憑證 AWS Certificate Manager。
- 您必須建立或更新 DNS 提供者的資源記錄，以映射至您的 API 端點。如果沒有這類映射，則綁定自訂網域名稱的 API 請求無法到達 API Gateway。

限制

- AWS 區域 在所有 中，自訂網域名稱必須是唯一的 AWS 帳戶。
- 若要設定具有多個層級的 API 映射，您必須使用區域性自訂網域名稱和 TLS 1.2 安全政策。
- 在 API 映射中，自訂網域名稱和映射APIs 必須位於相同的 中 AWS 帳戶。
- API 映射只能包含字母、數字和下列字元： `$-_.+!*'()/`
- API 映射中路徑的最大長度為 300 個字元。
- 您可以為每個域名設定具有 200 個具多個層級的 API 映射。
- 您只能將 HTTP APIs 映射至具有 TLS 1.2 安全政策的區域性自訂網域名稱。
- 您無法將 WebSocket APIs對應至與 HTTP API 或 REST API 相同的自訂網域名稱。
- 有些 AWS 服務 無法全部使用 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

- 此範例實作在 AWS CDK TypeScript 2.149.0 版中使用 。

Architecture

下圖顯示架構工作流程。

此圖展示了以下要點：

1. API 使用者向 Amazon API Gateway 傳送具有自訂網域名稱的請求。
2. API Gateway 會根據請求 URL 中指定的路徑，將使用者的請求動態路由到 API Gateway 的適當執行個體和階段。下表顯示如何將不同 URL 型路徑路由至不同 API Gateway 執行個體之特定階段的範例。

API	階段	路徑	預設端點

CalculationAPIv1	api	apiv1	已啟用
CalculationAPIv2	api	apiv2	已啟用
CalculationAPIvX	api	apivX	已啟用

3. 目的地 API Gateway 執行個體會處理請求，並將結果傳回給使用者。

自動化和擴展

我們建議您針對 API 的每個版本使用不同的 AWS CloudFormation 堆疊。透過此方法，您可以在可由自訂網域 APIs 的後端 API 之間進行完全隔離。這種方法的優點是，您可以獨立部署或刪除不同版本的 API，而不會帶來修改其他 API 的風險。這種方法透過隔離 CloudFormation 堆疊來提高彈性。此外，它還為您的 API 提供不同的後端選項 AWS Lambda AWS Fargate，例如 HTTP 端點和 的動作 AWS 服務。

您可以使用 Git 分支策略，例如 Gitflow，搭配隔離的 CloudFormation 堆疊來管理部署到不同 API 版本的原始碼。透過使用此選項，您可以維護不同版本的 API，而不需要複製新版本的原始程式碼。使用 Gitflow，您可以在執行版本時，將標籤新增至 git 儲存庫中的遞交。因此，您擁有與特定版本相關的原始程式碼的完整快照。由於需要執行更新，您可以查看特定版本的程式碼、進行更新，然後將更新的原始程式碼部署到與對應主要版本一致的 CloudFormation 堆疊。這種方法可降低破壞另一個 API 版本的風險，因為每個 API 版本都有隔離的原始程式碼，並部署到單獨的 CloudFormation 堆疊。

工具

AWS 服務

- Amazon API Gateway 可協助您建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。
- AWS Certificate Manager (ACM) 可協助您建立、存放和續約公有和私有 SSL/TLS X.509 憑證和金鑰，以保護 AWS 網站和應用程式。
- AWS Cloud Development Kit (AWS CDK) 是一種開放原始碼軟體開發架構，可讓您在程式碼中定義雲端基礎設施並透過其佈建 CloudFormation。此模式的範例實作使用 AWS CDK TypeScript 中的 。在 TypeScript AWS CDK 中使用 會使用熟悉的工具，包括 Microsoft TypeScript 編譯器 (`tsc`)、Node.js 和節點套件管理員 (npm)。如果您願意，雖然此模式中的範例使用 ，但您可以使用 Yarnnpm。構成AWS 建構程式庫的模組會透過儲存npm 庫 npmjs.org 進行分發。
- CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和 生命週期中管理資源 AWS 區域。

- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Route 53 是一種可用性高、可擴展性強的 DNS Web 服務。
- AWS WAF 是一種 Web 應用程式防火牆，可協助您監控轉送至受保護 Web 應用程式資源的 HTTP 和 HTTPS 請求。

其他工具

- Bruno 是開放原始碼且易於 Git 的 API 測試用戶端。
- cdk-nag 是一種開放原始碼公用程式，可透過使用規則套件來檢查 AWS CDK 應用程式是否有最佳實務。

程式碼儲存庫

此模式的程式碼可在 GitHub path-based-versioning-with-api-gateway儲存庫中使用。

最佳實務

- 使用強大的持續整合和持續交付 (CI/CD) 管道，自動化使用 建置之 CloudFormation 堆疊的測試和部署 AWS CDK。如需有關此建議的詳細資訊，請參閱 AWS Well-Architected DevOps 指南。
- AWS WAF 是受管防火牆，可輕鬆與 Amazon API Gateway 等服務整合。雖然 AWS WAF 不是此版本控制模式運作的必要元件，但我們建議將 做為安全最佳實務， AWS WAF 納入 API Gateway。
- 鼓勵 API 取用者定期升級至最新版本的 API，以便有效棄用和移除舊版的 API。
- 在生產設定中使用此方法之前，請為您的 API 實作防火牆和授權策略。
- AWS 帳戶 使用最低權限存取模型實作對 AWS 資源管理的存取。 https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#grant-least-privilege
- 若要針對使用 建置的應用程式強制執行最佳實務和安全性建議 AWS CDK，建議您使用 cdk-nag 公用程式。

史詩

準備您的本機環境

任務	Description	所需的技能
複製儲存庫。	若要複製範例應用程式儲存庫，請執行下列命令： `git clone https://github.com/aws-samples/path-based-versioning-with-api-gateway`	應用程式開發人員
導覽至複製的儲存庫。	若要導覽至複製的儲存庫資料夾位置，請執行下列命令： `cd api-gateway-custom-domain-versioning`	應用程式開發人員
安裝所需的依存項目。	若要安裝必要的相依性，請執行下列命令： `npm install`	應用程式開發人員

部署 CloudFormation 路由堆疊

任務	Description	所需的技能
啟動路由堆疊的部署。	若要啟動 CloudFormation 路由堆疊 的部署`CustomDomainRouterStack` ，請執行下列命令，`example.com` 將取代為您擁有的網域名稱：	應用程式開發人員

任務	Description	所需的技能
	`npx cdk deploy CustomDomainRouterStack --parameters PrerequisiteDomainName=example.com` **Note** 在下列網域 DNS 驗證任務成功執行之前，堆疊部署不會成功。	

驗證網域所有權

任務	Description	所需的技能
驗證網域的擁有權。	在您證明相關聯網域的擁有權之前，憑證將保持待定驗證狀態。 若要證明擁有權，請將 CNAME 記錄新增至與網域相關聯的託管區域。如需詳細資訊，請參閱 AWS Certificate Manager 文件中的 DNS 驗證。 新增適當的記錄可讓 `CustomDomainRouterStack` 部署成功。	應用程式開發人員、AWS 系統管理員、網路管理員
建立別名記錄以指向您的 API Gateway 自訂網域。	成功發出並驗證憑證後，請建立指向 Amazon API Gateway 自訂網域 URL 的 DNS 記錄。Amazon API Gateway	應用程式開發人員、AWS 系統管理員、網路管理員

任務	Description	所需的技能
	自訂網域 URL 由自訂網域的佈建唯一產生，並指定為 CloudFormation 輸出參數。以下是記錄的範例： 路由政策：簡易路由 記錄名稱： `demo.api-gateway-custom-domain-versioning.example.com` Alias (別名)：是 記錄類型：指向 AWS 資源的 "A" 類型的 DNS 記錄 Value (值)：`d-xxxxxxxxxx.execute-api.xx-xxxx-x.amazonaws.com` TTL （秒）：300	

部署 CloudFormation 堆疊並叫用 API

任務	Description	所需的技能
部署`ApiStackV1` 堆疊。	若要部署`ApiStackV1` 堆疊，請使用下列命令： `npm run deploy-v1` 下列 CDK 程式碼新增 API 映射：	應用程式開發人員

任務	Description	所需的技能
	`var apiMapping = new CfnApiMapping(this, "ApiMapping", { apiId: this.lambdaRestApi.restApiId, domainName: props.customDomainName.domainName, stage: "api", apiMappingKey: "api/v1", });`	
部署ApiStackV2 堆疊。	若要部署ApiStackV2 堆疊，請使用下列命令： `npm run deploy-v2`	應用程式開發人員
叫用 API。	若要使用 Bruno 叫用 API 並測試 API 端點，請參閱其他資訊中的指示。	應用程式開發人員

清除資源

任務	Description	所需的技能
清除資源。	若要銷毀與此範例應用程式相關聯的資源，請使用下列命令： `npx cdk destroy --all` **Note** 請務必清除為網域擁有權驗證程序手動新增的	應用程式開發人員

任務	Description	所需的技能
	任何 Route 53 DNS 記錄。	

疑難排解

問題	解決方案
部署`CustomDomainRouterStack`因為無法驗證憑證而逾時。	請確定您已新增適當的 DNS 驗證 CNAME 記錄，如先前任務所述。在新增 DNS 驗證記錄之後，您的新憑證可能會繼續顯示等待驗證狀態長達 30 分鐘。如需詳細資訊，請參閱 AWS Certificate Manager 文件中的 DNS 驗證。

相關資源

- 使用 Amazon CloudFront 實作標頭型 API Gateway 版本控制 – 此 AWS 運算部落格文章提供標頭型版本控制策略，以替代此模式中概述的路徑型版本控制策略。
- AWS CDK 研討會 – 此簡介研討會著重於 AWS 使用 在 上建置和部署應用程式 AWS Cloud Development Kit (AWS CDK)。此研討會支援 Go、Python 和 TypeScript。

其他資訊

使用 Bruno 測試您的 API

我們建議您使用開放原始碼 API 測試工具 Bruno，來驗證範例應用程式的路徑型路由是否正常運作。此模式提供範例集合，以協助測試您的範例 API。

若要叫用和測試您的 API，請使用下列步驟：

1. 安裝 Bruno。
2. 開啟 Bruno。
3. 在此模式的程式碼儲存庫中，選取 Bruno/Sample-API-Gateway-Custom-Domain-Versioning，然後開啟集合。

4. 若要查看使用者介面 (UI) 右上角的環境下拉式清單，請選取集合中的任何請求。
5. 在環境下拉式清單中，選取設定。
6. 將 REPLACE_ME_WITH_YOUR_DOMAIN值取代為您的自訂網域。
7. 選擇儲存，然後關閉組態區段。
8. 針對沙盒環境， 確認已選取作用中選項。
9. 使用所選請求的 -> 按鈕調用您的 API。
10請注意，與 V1 中如何處理驗證 （傳遞非數值）。 V2

若要查看範例 API 呼叫的螢幕擷取畫面，以及 V1 和 V2 驗證的比較，請參閱此模式程式碼儲存庫中的測試README.md檔案中的範例 API。

將 psycopg2 程式庫匯入 AWS Lambda ，以與您的 PostgreSQL 資料庫互動

Louis Hourcade，Amazon Web Services

總結

Psycopg 是適用於 Python 的 PostgresSQL 資料庫轉接器。開發人員使用 psycopg2程式庫撰寫與 PostgreSQL 資料庫互動的 Python 應用程式。

在 Amazon Web Services (AWS) 上，開發人員也會使用 AWS Lambda來執行應用程式或後端服務的程式碼。Lambda 是一種無伺服器、事件驅動的運算服務，無需佈建或管理伺服器即可執行程式碼。

根據預設，當您建立使用 Lambda 支援的 Python 執行期的新函數時，Lambda 執行期環境會從 提供的 Lambda 基礎映像建立 AWS。基本映像psycopg2中不包含 pandas或 等程式庫。若要使用程式庫，您需要將其封裝在自訂套件中，並將其連接到 Lambda。

有多種方式可以綁定和連接程式庫，包括下列項目：

- 從 .zip 檔案封存部署 Lambda 函數。
- 從自訂容器映像部署 Lambda 函數。
- 建立 Lambda 層，並將其連接至您的 Lambda 函數。

此模式示範前兩個選項。

使用 .zip 部署套件，將程式pandas庫新增至 Lambda 函數相對簡單。在 Linux 機器上建立資料夾、將 Lambda 指令碼與程式pandas庫和程式庫的相依性新增至資料夾、壓縮資料夾，並將其做為 Lambda 函數的來源。

雖然使用 .zip 部署套件是常見的做法，但該方法不適用於程式psycopg2庫。如果您使用 .zip 部署套件將程式psycopg2庫新增至 Lambda 函數，此模式會先顯示您遇到的錯誤。此模式接著會示範如何從 Dockerfile 部署 Lambda，並編輯 Lambda 映像，讓psycopg2程式庫運作。

如需模式部署的三個資源的相關資訊，請參閱其他資訊一節。

先決條件和限制

先決條件

- AWS 帳戶 具有足夠許可的作用中 ，可部署此模式使用 AWS 的資源
- AWS Cloud Development Kit (AWS CDK) 透過執行 全域安裝 `npm install -g aws-cdk`

- Git 用戶端
- Python
- Docker

限制

- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

產品版本

- Lambda 支援的 Python 執行時間版本
- Psycopg2 2.9.3 版
- Pandas 1.5.2 版

Architecture

解決方案概觀

為了說明在 Lambda 中使用程式psycopg2庫時可能面臨的挑戰， 模式會部署兩個 Lambda 函數：

- 一個 Lambda 函數，其中包含從 .zip 檔案建立的 Python 執行期。psycopg2 和 pandas程式庫是使用 pip 安裝在此 .zip 部署套件中。
- 一個 Lambda 函數，其中包含從 Dockerfile 建立的 Python 執行期。Dockerfile 會將 psycopg2和 pandas程式庫安裝到 Lambda 容器映像中。

第一個 Lambda 函數會在 .zip 檔案中安裝程式pandas庫及其相依性，Lambda 可以使用該程式庫。

第二個 Lambda 函數示範，透過為 Lambda 函數建置容器映像，您可以在 Lambda 中執行 pandas和 psycopg2程式庫。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。

- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。

其他工具

- Docker 是一組平台即服務 (PaaS) 產品，可在作業系統層級使用虛擬化在容器中交付軟體。
- pandas 是以 Python 為基礎的開放原始碼工具，用於資料分析和操作。
- Psycopg 是適用於 Python 語言的 PostgreSQL 資料庫轉接器，專為多執行緒應用程式而設計。此模式使用 Psycopg 2。
- Python 是一種一般用途的電腦程式設計語言。

程式碼儲存庫

此模式的程式碼可在 GitHub 的 import-psycopg2-in-lambda-to-interact-with-postgres-database 儲存庫中使用。

最佳實務

此模式提供您使用 從 Dockerfile AWS CDK 建立 Lambda 函數的工作範例。如果您在應用程式中重複使用此程式碼，請確定部署的資源符合所有安全需求。使用 Checkov 等工具，在部署基礎設施之前掃描雲端基礎設施組態以尋找組態錯誤。

史詩

複製儲存庫並設定部署

任務	Description	所需的技能

複製儲存庫。	若要在本機電腦上複製 GitHub 儲存庫，請執行下列命令：
	`git clone https://github.com/aws-samples/import-psycopg2-in-lambda-to-interact-with-postgres-database.git`
	`cd AWS-lambda-psycopg2`
		一般 AWS

任務	Description	所需的技能

設定您的部署。	使用下列資訊編輯 app.py 檔案 AWS 帳戶：
	`aws_acccount = "AWS_ACCOUNT_ID"`
	`region = "AWS_REGION"`
	`# Select the CPU architecture you are using to build the image (ARM or X86)`
	`architecture = "ARM"`
		一般 AWS

引導您的 AWS 帳戶並部署應用程式

任務	Description	所需的技能
引導您的 AWS 帳戶。	如果您尚未啟動 AWS 環境，請使用您 AWS 帳戶的 AWS 登入資料執行下列命令： `cdk bootstrap aws://<tooling-account-id>/<aws-region>`	一般 AWS
部署程式碼。	若要部署 AWS CDK 應用程式，請執行下列命令： `cdk deploy AWSLambdaPyscopg2`	一般 AWS

從 AWS 管理主控台測試 Lambda 函數

任務	Description	所需的技能
測試從 .zip 檔案建立的 Lambda 函數。	若要測試從 .zip 檔案建立的 Lambda 函數，請執行下列動作： 1. 登入 主控台，並在 https：//https://console.aws.amazon.com/lambda/ 開啟 Lambda 主控台。 2. 選取 `lambda-from-zip` Lambda 函數。 3. 建立測試事件以叫用 函數。 4. 調用時，函數應該引發包含下列訊息的錯誤： `"errorMessage": Unable to import module 'lambda_code': libpq.so.5: cannot open shared object, "stackTrace": [] "errorType": Runtime.ImportModuleError",` 5. 在 https://console.aws.amazon.com/cloudwatch/ 開啟 Amazon CloudWatch 主控台。CloudWatch 日誌顯示已成功匯入程式`pandas`庫，但`psycopg2`程式庫匯入失敗。	一般 AWS

任務	Description	所需的技能
	由於 Lambda 在預設映像中找不到所需的 PostgreSQL 程式庫，因此無法使用程式psycopg2庫。	

任務	Description	所需的技能

測試從 Dockerfile 建立的 Lambda 函數。	若要在 Lambda 函數中使用程式psycopg2庫，您必須編輯 Lambda Amazon Machine Image (AMI)。
	
	若要測試從 Dockerfile 建立的 Lambda 函數，請執行下列動作：
	
	1. 登入 主控台，然後開啟 Lambda 主控台。
	2. 選取 `lambda-from-docker` Lambda 函數。
	3. 建立測試事件以叫用 函數。
	4. 調用時，函數應該會成功執行。
	
	下列程式碼顯示 AWS CDK 範本建立的 Dockerfile：
	
	`# Start from lambda Python3.13 image`
	`FROM public.ecr.aws/lambda/python:3.13`
	
	`# Copy the lambda code, together with its requirements`
	`COPY lambda/requirements.txt ${LAMBDA_TASK_ROOT}`
	`COPY lambda/lambda_code.py ${LAMBDA_TASK_ROOT}`
	
	`# Install postgresql-devel in your image`
		一般 AWS

任務	Description	所需的技能

	<pre>RUN yum install -y gcc
	 postgresql-devel
	
	# install the requireme
	nts for the Lambda code
	RUN pip3 install -r
	 requirements.txt
	 --target "${LAMBDA
	_TASK_ROOT}"
	
	# Command can be
	 overwritten by
	 providing a different
	 command in the template
	 directly.
	CMD ["lambda_code.hand
	ler"]</pre>
	Dockerfile 會為 Python 執行時間取得 AWS 提供的 Lambda 映像，並安裝 postgresql-devel，其中包含編譯直接與 PostgreSQL 管理伺服器互動之應用程式所需的程式庫。Dockerfile 也會安裝 pandas和 psycopg2程式庫，這些程式庫會顯示在 `requirements.txt` 檔案中。	

相關資源

- AWS CDK 文件
- AWS Lambda 文件

其他資訊

在此模式中，AWS CDK 範本會提供具有三個資源的 AWS 堆疊：

- Lambda 函數的 AWS Identity and Access Management (IAM) 角色。
- 具有 Python 執行時間的 Lambda 函數。函數是從部署套件`Constructs/lambda/lambda_deploy.zip`部署。
- 具有 Python 執行時間的 Lambda 函數。函數是從 `Constructs` 資料夾下的 Dockerfile 部署

兩個 Lambda 函數的指令碼會檢查 `pandas`和 `psycopg2`程式庫是否已成功匯入：

```
import pandas
print("pandas successfully imported")

import psycopg2
print("psycopg2 successfully imported")

def handler(event, context):
    """Function that checks whether psycopg2  and pandas are successfully imported or
 not"""
    return {"Status": "psycopg2 and pandas successfully imported"}
```

`lambda_deploy.zip` 部署套件是以 `Constructs/lambda/build.sh` bash 指令碼建置。此指令碼會建立資料夾、複製 Lambda 指令碼、安裝 `pandas`和 `psycopg2`程式庫，以及產生 .zip 檔案。若要自行產生 .zip 檔案，請執行此 bash 指令碼並重新部署 AWS CDK 堆疊。

Dockerfile 一開始是使用 Python 執行時間為 Lambda AWS 提供的基礎映像。Dockerfile 會在預設映像上方安裝 `pandas`和 `psycopg2`程式庫。

將 Amazon API Gateway 與 Amazon SQS 整合，以處理非同步 REST APIs

Natalia Colantonio Favero 和 Gustavo Martim，Amazon Web Services

總結

部署 REST APIs 時，有時您需要公開用戶端應用程式可以發佈的訊息佇列。例如，您可能會遇到第三方 APIs 延遲和回應延遲的問題，或者您可能想要避免資料庫查詢的回應時間，或避免在有大量並行APIs時擴展伺服器。在這些情況下，發佈到佇列的用戶端應用程式只需要知道 API 收到資料，而不是收到資料之後會發生什麼情況。

此模式使用 Amazon API Gateway 將訊息傳送至 Amazon Simple Queue Service (Amazon SQS) 來建立 REST API 端點。 Amazon SQS 它會在兩個服務之間建立easy-to-implement的整合，以避免直接存取 SQS 佇列。

先決條件和限制

- 作用中 AWS 的帳戶

Architecture

圖表說明這些步驟：

1. 使用 Postman、其他 API 或其他技術等工具請求 POST REST API 端點。
2. API Gateway 會在佇列上張貼在請求內文上接收的訊息。
3. Amazon SQS 會收到訊息，並以成功或失敗代碼傳送答案給 API Gateway。

工具

- Amazon API Gateway 可協助您建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon Simple Queue Service (Amazon SQS) 提供安全、耐用且可用的託管佇列，可協助您整合和分離分散式軟體系統和元件。

史詩

建立 SQS 佇列

任務	Description	所需的技能
建立佇列。	若要建立接收來自 REST API 訊息的 SQS 佇列： 1. 請登入您的 AWS 帳戶。 2. 在 https://console.aws.amazon.com/sqs/ 開啟 Amazon SQS 主控台。 3. 選擇建立佇列。 4. 在建立佇列頁面上，AWS 區域 從區域下拉式清單中選擇正確的 。 5. 對於類型，請保留預設設定 (標準)。 6. 輸入佇列的名稱。 7. 保留所有其他設定的預設值。 8. 選擇建立佇列。	應用程式開發人員

提供 Amazon SQS 的存取權

任務	Description	所需的技能
建立 IAM 角色。	此 IAM 角色可讓 API Gateway 資源完整存取 Amazon SQS。 1. 前往網址 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。	應用程式開發人員、AWS 管理員

任務	Description	所需的技能
	2. 在導覽窗格中，選擇 Roles (角色)、Create role (建立新角色)。 3. 對於 Trusted entity type (信任的實體類型)，請選擇 AWS 服務。 4. 針對使用案例，從下拉式清單中選擇 API Gateway，然後選擇下一步、下一步。 5. 針對角色名稱，輸入 AWSGatewayRoleForSQS 和選用描述，然後選擇建立角色。 6. 在角色窗格中，搜尋 AWSGatewayRoleForSQS，然後選取其核取方塊。 7. 在許可政策區段中，選擇新增許可、連接政策。 8. 搜尋並選取 AmazonSQSFullAccess。 9. 選擇新增許可。 10. 在 AWSGatewayRoleForSQS 的摘要區段中，複製 Amazon Resource Number (ARN)。您將在後續步驟中使用此 ID。	

建立 REST API

任務	Description	所需的技能
建立 REST API。	這是傳送 HTTP 請求的 REST API。	應用程式開發人員

任務	Description	所需的技能
	1. 在以下網址開啟 API Gateway 主控台：https://console.aws.amazon.com/apigateway/。 2. 在 REST API 區段中，選擇建置。 3. 針對 API 名稱，輸入 API 的名稱和選用描述，保留所有其他預設設定，然後選擇建立 API。	

任務	Description	所需的技能
將 API Gateway 連接至 Amazon SQS。	此步驟可讓訊息從 HTTP 請求的內文內部流向 Amazon SQS。 1. 在 API Gateway 主控台上，選擇您建立的 API。 2. 在資源頁面上的方法區段中，選擇建立方法。 3. 針對方法類型，選擇 POST。 4. 針對整合類型，選擇 AWS 服務。 5. 針對 AWS 區域，選擇您建立 SQS 佇列的區域。 6. 針對 AWS 服務，選擇簡易佇列服務 (SQS)。 7. 針對 HTTP 方法，選擇 POST。 8. 針對動作類型，選擇使用路徑覆寫。 9. 針對路徑覆寫，輸入 <AWS 帳戶 ID>/<SQS 佇列名稱>。 10針對執行角色，貼上您先前建立之角色的 ARN。 11選擇建立方法。	應用程式開發人員

測試 REST API

任務	Description	所需的技能
測試 REST API。	執行測試以檢查缺少的組態：	應用程式開發人員

任務	Description	所需的技能

	1. 在 API Gateway 主控台上，選擇您建立的 REST API。
	2. 在資源窗格中，選擇 POST 方法。
	3. 選擇測試標籤。（如果未顯示標籤，請使用向右箭頭。）
	4. 針對請求內文，貼上下列 JSON 程式碼：
	`{ "message": "lorem ipsum" }`
	5. 選擇測試。
	您會收到類似以下的錯誤：
	`<UnknownOperationException/>`	

任務	Description	所需的技能
變更 API 整合以將請求正確轉送至 Amazon SQS。	完成組態以修正整合錯誤： 1. 在 API Gateway 主控台上，選擇您建立的 API，然後選擇 POST。 2. 方法執行區段顯示 API Gateway 和 Amazon SQS 之間的視覺化映射。從本節中，選擇整合請求，然後選擇編輯。 3. 展開 HTTP 標頭區段，然後選擇新增請求標頭參數。 • 針對名稱，指定 Content-Type。 • 對於映射來源，輸入 'application/x-www-form-urlencoded'。請務必包含單引號。 • 選取快取核取方塊。 4. 展開映射範本區段。 • 選擇 Add mapping template (新增對應範本)。 • 針對內容類型，輸入 application/json。 • 對於範本內文，請貼上此程式碼： `Action=SendMessage&MessageBody=$input.body` • 選擇儲存。	應用程式開發人員

任務	Description	所需的技能

在 Amazon SQS 中測試和驗證訊息。	執行測試以確認測試已成功完成：
	1. 在 API Gateway 主控台上，選擇您建立的 REST API。
	2. 在資源窗格中，選擇 POST 方法。
	3. 選擇測試標籤。（如果未顯示標籤，請使用向右箭頭。）
	4. 針對請求內文，貼上下列 JSON 程式碼：
	`{ "message": "lorem ipsum" }`
	5. 選擇測試。
	6. 開啟 Amazon SQS 主控台。
	7. 在導覽窗格中，選擇佇列，然後選擇您的佇列。
	8. 選擇傳送及接收訊息。
	9. 選擇訊息輪詢。
	10選擇 Message (訊息)。它應該會顯示下列項目：
	`Body { "message": "lorem ipsum" }`
		應用程式開發人員

任務	Description	所需的技能

使用特殊字元測試 API Gateway。	執行測試，其中包含訊息中無法接受的特殊字元（例如 &）：
	
	1. 在 API Gateway 主控台上，選擇您的 API。
	2. 使用下列 JSON 程式碼，重複先前步驟的測試：
	
	`{ "message": "lorem ipsum &" }`
	
	3. 選擇測試。
	
	您會收到如下的錯誤：
	
	`{ "Error": { "Code": "AccessDenied", "Message": "Access to the resource https://sqs.us-east-2.amazonaws.com/976166761794/Apg2 is denied.", "Type": "Sender" }, "RequestId": "e83c9c67-bcf6-5e9a-91e9-c737094b17ab" }`
	
	這是因為訊息內文中預設不支援特殊字元。在下一個步驟
		應用程式開發人員

任務	Description	所需的技能
	中，您將設定 API Gateway 以支援特殊字元。如需內容類型轉換的詳細資訊，請參閱 API Gateway 文件。	

任務	Description	所需的技能

變更 API 組態以支援特殊字元。	調整組態以接受訊息中的特殊字元：
	1. 在 API Gateway 主控台上，選擇您建立的 API，然後選擇 POST。
	2. 選擇整合請求，然後選擇編輯。
	3. 將內容處理變更為轉換為文字。
	4. 在映射範本區段中：
	• 針對內容類型，輸入 application/json。
	• 針對範本內文，指定：
	`Action=SendMessage&MessageBody=$util.urlEncode($input.body)`
	• 選擇儲存。
	5. 選擇測試標籤。
	6. 針對請求內文，輸入稍早的 JSON 程式碼：
	`{ " message": "lorem ipsum &" }`
	7. 選擇測試。
	8. 開啟 Amazon SQS 主控台。
	9. 選取您的佇列，然後選擇傳送和接收訊息、輪詢訊息、傳送訊息。
		應用程式開發人員

任務	Description	所需的技能
	新訊息應包含特殊字元。	

部署 REST API

任務	Description	所需的技能
部署 API。	若要部署 REST API： 1. 開啟 API Gateway 主控台。 2. 選擇您的 API。 3. 選擇部署 API。如需此步驟的詳細資訊，請參閱 API Gateway 文件。	應用程式開發人員
使用外部工具進行測試。	使用外部工具執行測試，以確認已成功接收訊息： 1. 開啟 Postman、Insomnia 或 cURL 等工具。 2. 執行您的 API。 3. 開啟 Amazon SQS 主控台。 4. 選取您的佇列。 5. 載入訊息以查看新訊息。	應用程式開發人員

清除

任務	Description	所需的技能
刪除 API。	在 API Gateway 主控台上，選擇您建立的 API，然後選擇刪除。	應用程式開發人員

任務	Description	所需的技能
刪除 IAM 角色。	在 IAM 主控台的角色窗格中，選取 AWSGatewayRoleForSQS，然後選擇刪除。	應用程式開發人員
刪除 SQS 佇列。	在 Amazon SQS 主控台的佇列窗格中，選擇您建立的 SQS 佇列，然後選擇刪除。	應用程式開發人員

相關資源

- SQS-SendMessage (API Gateway 文件）
- API Gateway 中的內容類型轉換 (API Gateway 文件）
- $util 變數 (API Gateway 文件）
- 如何整合 API Gateway REST API 與 Amazon SQS 並解決常見錯誤？(AWS re：Post 文章）

使用 Amazon API Gateway 和 AWS Lambda 非同步處理事件

Andrea Meroni、Mariem Kthiri、Nadim Majed 和 Michael Wallner，Amazon Web Services

總結

Amazon API Gateway 是一項全受管服務，開發人員可用來建立、發佈、維護、監控和保護任何規模 APIs。它會處理涉及接受和處理多達數十萬個並行 API 呼叫的任務。

API Gateway 的重要服務配額是整合逾時。逾時是後端服務必須在 REST API 傳回錯誤之前傳回回應的最長時間。對於同步工作負載，通常可接受 29 秒的硬性限制。不過，該限制對想要將 API Gateway 與非同步工作負載搭配使用的開發人員來說是一項挑戰。

此模式顯示使用 API Gateway 和 以非同步方式處理事件的範例架構 AWS Lambda。架構支援執行長達 15 分鐘的處理任務，並使用基本 REST API 做為界面。

Projen 用於設定本機開發環境 AWS 帳戶，以及搭配 AWS Cloud Development Kit (AWS CDK) Toolkit、Docker 和 Node.js 將範例架構部署至目標。Projen 會自動使用預先遞交和用於程式碼品質保證、安全掃描和單元測試的工具來設定 Python 虛擬環境。如需詳細資訊，請參閱工具一節。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 您的工作站上安裝了下列工具：
  - AWS Cloud Development Kit (AWS CDK) 工具組 2.85.0 版
  - Docker 20.10.21 版
  - Node.js 18.13.0 版
  - Projen 0.71.111 版
  - Python 3.9.16 版

限制

- 任務的最大執行時間受限於 Lambda 函數的最大執行時間 (15 分鐘）。
- 並行任務請求的數量上限受限於 Lambda 函數的預留並行。

Architecture

下圖顯示任務 API 與事件處理和錯誤處理 Lambda 函數的互動，以及存放在 Amazon EventBridge 事件封存中的事件。

典型的工作流程包含下列步驟：

1. 您可以驗證 AWS Identity and Access Management (IAM) 並取得安全登入資料。
2. 您可以將 HTTP `POST`請求傳送至`/jobs`任務 API 端點，在請求內文中指定任務參數。
3. 任務 API 是一種 API Gateway REST API，會傳回包含任務識別符的 HTTP 回應給您。
4. 任務 API 會以非同步方式叫用事件處理 Lambda 函數。
5. 事件處理函數會處理事件，然後將任務結果放入任務 Amazon DynamoDB 資料表
6. 您可以將 HTTP `GET`請求傳送至`/jobs/{jobId}`任務 API 端點，並將步驟 3 的任務識別符做為 `{jobId}`。
7. 任務 API 會查詢 `jobs` DynamoDB 資料表來擷取任務結果。
8. 任務 API 會傳回包含任務結果的 HTTP 回應。
9. 如果事件處理失敗，事件處理函數會將事件傳送至錯誤處理函數。
10. 錯誤處理函數會將任務參數放在 `jobs` DynamoDB 資料表中。
11. 您可以透過傳送 HTTP `GET`請求至任務 API 端點來擷取`/jobs/{jobId}`任務參數。
12. 如果錯誤處理失敗，錯誤處理函數會將事件傳送至 EventBridge 事件封存。

    您可以使用 EventBridge 重播封存的事件。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端 基礎設施。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。

- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如，Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他事件匯流排 AWS 帳戶。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。

其他工具

- autopep8 會根據 Python Enhancement Proposal (PEP) 8 樣式指南自動格式化 Python 程式碼。
- Bandit 會掃描 Python 程式碼以尋找常見的安全問題。
- Commitizen 是 Git 遞交檢查程式和CHANGELOG產生器。
- cfn-lint 是 AWS CloudFormation linter
- Checkov 是一種靜態程式碼分析工具，可將基礎設施檢查為程式碼 (IaC) 是否有安全性和合規設定錯誤。
- jq 是用於剖析 JSON 的命令列工具。
- Postman 是 API 平台。
- 預先遞交是 Git hooks 管理員。
- Projen 是專案產生器。
- pytest 是一種 Python 架構，用於撰寫可讀取的小型測試。

程式碼儲存庫

您可以在 GitHub 非同步事件處理與 API Gateway 和 Lambda 儲存庫中找到此架構程式碼範例。

最佳實務

- 此範例架構不包含已部署基礎設施的監控。如果您的使用案例需要監控，請評估新增 CDK 監控建構或其他監控解決方案。
- 此範例架構使用 IAM 許可來控制對任務 API 的存取。有權擔任 的任何人JobsAPIInvokeRole都可以叫用任務 API。因此，存取控制機制為二進位。如果您的使用案例需要更複雜的授權模型，請使用不同的存取控制機制進行評估。
- 當使用者傳送 HTTP POST請求到/jobs任務 API 端點時，輸入資料會在兩個不同的層級進行驗證：
  - Amazon API Gateway 負責第一個請求驗證。

- 事件處理函數會執行第二個請求。

  當使用者對/jobs/{jobId}任務 API 端點提出 HTTP GET請求時，不會執行驗證。如果您的使用案例需要額外的輸入驗證並提高安全性，請使用 AWS WAF 評估 來保護您的 API。

史詩

設定環境

任務	Description	所需的技能
複製儲存庫。	若要在本機複製儲存庫，請執行下列命令： `git clone https://github.com/aws-samples/asynchronous-event-processing-api-gateway-lambda-cdk.git`	DevOps 工程師

設定專案。	使用 Projen 將目錄變更為儲存庫根目錄，並設定 Python 虛擬環境和所有工具：
	`cd asynchronous-event-processing-api-gateway-api-gateway-lambda-cdk`
	`npx projen`
		DevOps 工程師

安裝預先遞交掛鉤。	若要安裝預先遞交掛鉤，請執行下列動作： 1. 啟用 Python 虛擬環境： `source .env/bin/activate` 2. 安裝預先遞交掛鉤：	DevOps 工程師

任務	Description	所需的技能
	`pre-commit install` `pre-commit install --hook-type commit-msg`	

部署範例架構

任務	Description	所需的技能
引導 AWS CDK。	若要在 AWS CDK 中引導 AWS 帳戶，請執行下列命令： `AWS_PROFILE=$YOUR_AWS_PROFILE npx projen bootstrap`	AWS DevOps
部署範例架構。	若要在 中部署範例架構 AWS 帳戶，請執行下列命令： `AWS_PROFILE=$YOUR_AWS_PROFILE npx projen deploy`	AWS DevOps

測試架構

任務	Description	所需的技能
安裝測試先決條件。	在工作站上安裝 AWS Command Line Interface (AWS CLI)、Postman 和 jq。 建議使用 Postman 測試此範例架構，但不強制。如果您選擇替代 API 測試工具，請確定	DevOps 工程師

任務	Description	所需的技能
	它支援 AWS Signature 第 4 版身分驗證，並參考可透過匯出 REST API 檢查的公開 API 端點。	
擔任 JobsAPIInvokeRole 。	假設JobsAPIInvokeRole 從部署命令列印為輸出的 ： `CREDENTIALS=$(AWS_PROFILE=$<YOUR_AWS_PROFILE> aws sts assume-role \ --no-cli-pager \ --role-arn $<JOBS_API_INVOKE_ROLE_ARN> \ --role-session-name JobsAPIInvoke) export AWS_ACCESS_KEY_ID=$(cat $CREDENTIALS \| jq '.Credentials''.AccessKeyId') export AWS_SECRET_ACCESS_KEY=$(cat $CREDENTIALS \| jq '.Credentials''.SecretAccessKey') export AWS_SESSION_TOKEN==$(cat $CREDENTIALS | jq '.Credentials''.SessionToken')`	AWS DevOps

任務	Description	所需的技能
設定 Postman。	1. 若要匯入包含在儲存庫中的 Postman 集合，請遵循 Postman 文件中的指示。 2. 使用下列值設定JobsAPI變數： • accessKey – assume-role 來自命令的 Credentials.AccessKeyId 屬性值 • baseUrl – 部署命令JobsApiJobsAPIEndpoint 輸出的值，不含結尾斜線 • region – AWS 區域 您部署範例架構的 值 • seconds – 範例任務的輸入參數值。它必須是正整數 • secretKey – assume-role 來自命令的 Credentials.SecretAccessKey 屬性值 • sessionToken – assume-role 來自命令的 Credentials.SessionToken 屬性值	AWS DevOps
測試範例架構。	若要測試範例架構，請將請求傳送至任務 API。如需詳細資訊，請參閱 Postman 文件。	DevOps 工程師

疑難排解

問題	解決方案
範例架構的銷毀和後續重新部署失敗，因為 Amazon CloudWatch Logs 日誌群組 `/aws/apigateway/JobsAPIAccessLogs` 已存在。	1. 如有必要，請將您的日誌資料匯出至 Amazon S3。 2. 刪除 CloudWatch Logs 日誌群組 `/aws/apigateway/JobsAPIAccessLogs` 。 3. 重新部署範例架構。

相關資源

- API Gateway 映射範本和存取記錄變數參考
- 設定後端 Lambda 函數的非同步調用

使用 Amazon API Gateway 和 Amazon DynamoDB Streams 非同步處理事件

Andrea Meroni、Mariem Kthiri、Nadim Majed、Alessandro Trisolini 和 Michael Wallner，Amazon Web Services

總結

Amazon API Gateway 是一項全受管服務，開發人員可用來建立、發佈、維護、監控和保護任何規模 APIs。它會處理涉及接受和處理多達數十萬個並行 API 呼叫的任務。

API Gateway 的一項重要服務配額是整合逾時。逾時是後端服務必須在 REST API 傳回錯誤之前傳回回應的最長時間。對於同步工作負載，通常可接受 29 秒的硬性限制。不過，該限制對想要將 API Gateway 與非同步工作負載搭配使用的開發人員來說是一項挑戰。

此模式顯示使用 API Gateway、Amazon DynamoDB Streams 和 非同步處理事件的範例架構 AWS Lambda。架構支援使用相同的輸入參數執行平行處理任務，並使用基本 REST API 做為界面。在此範例中，使用 Lambda 做為後端會將任務持續時間限制為 15 分鐘。您可以使用替代服務來處理傳入事件（例如，），以避免此限制 AWS Fargate。

Projen 用於設定本機開發環境 AWS 帳戶，以及搭配 AWS Cloud Development Kit (AWS CDK) Toolkit、Docker 和 Node.js 將範例架構部署至目標。Projen 會自動使用預先遞交和用於程式碼品質保證、安全掃描和單元測試的工具來設定 Python 虛擬環境。如需詳細資訊，請參閱工具一節。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 工作站上安裝了下列工具：
  - AWS Cloud Development Kit (AWS CDK) 工具組 2.85.0 版或更新版本
  - Docker 20.10.21 版或更新版本
  - Node.js 第 18 版或更新版本
  - Projen 0.71.111 版或更新版本
  - Python 3.9.16 版或更新版本

限制

- DynamoDB Streams 建議的讀取器數目上限為兩個，以避免限流。
- 任務的最大執行時間受限於 Lambda 函數的最大執行時間 (15 分鐘）。
- 並行任務請求的數量上限受限於 Lambda 函數的預留並行。

Architecture

架構

下圖顯示任務 API 與 DynamoDB Streams 的互動，以及事件處理和錯誤處理 Lambda 函數的互動，以及存放在 Amazon EventBridge 事件封存中的事件。

典型的工作流程包括以下步驟：

1. 您可以驗證 AWS Identity and Access Management (IAM) 並取得安全登入資料。
2. 您可以將 HTTP `POST`請求傳送至`/jobs`任務 API 端點，在請求內文中指定任務參數。
3. 任務 API 會傳回包含任務識別符的 HTTP 回應給您。
4. 任務 API 會將任務參數放在 Amazon DynamoDB `jobs_table` 資料表中。
5. `jobs_table` DynamoDB 資料表 DynamoDB 串流會叫用事件處理 Lambda 函數。
6. 事件處理 Lambda 函數會處理事件，然後將任務結果放入 `jobs_table` DynamoDB 資料表。為了協助確保結果一致，事件處理函數實作樂觀鎖定機制。
7. 您可以將 HTTP `GET`請求傳送至`/jobs/{jobId}`任務 API 端點，並將步驟 3 的任務識別符做為 `{jobId}`。
8. 任務 API 會查詢 `jobs_table` DynamoDB 資料表來擷取任務結果。
9. 任務 API 會傳回包含任務結果的 HTTP 回應。
10. 如果事件處理失敗，事件處理函數的來源映射會將事件傳送至錯誤處理 Amazon Simple Notification Service (Amazon SNS) 主題。
11. 錯誤處理 SNS 主題會以非同步方式將事件推送至錯誤處理函數。
12. 錯誤處理函數會將任務參數放在 `jobs_table` DynamoDB 資料表中。

    您可以透過傳送 HTTP `GET`請求至任務 API 端點來擷取`/jobs/{jobId}`任務參數。
13. 如果錯誤處理失敗，錯誤處理函數會將事件傳送至 Amazon EventBridge 封存。

    您可以使用 EventBridge 重播封存的事件。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如，AWS Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他 AWS 帳戶中的事件匯流排。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。

其他工具

- autopep8 會根據 Python Enhancement Proposal (PEP) 8 樣式指南自動格式化 Python 程式碼。
- Bandit 會掃描 Python 程式碼以尋找常見的安全問題。
- Commitizen 是 Git 遞交檢查程式和CHANGELOG產生器。
- cfn-lint 是 AWS CloudFormation linter
- Checkov 是一種靜態程式碼分析工具，可將基礎設施檢查為程式碼 (IaC) 是否有安全性和合規設定錯誤。
- jq 是用於剖析 JSON 的命令列工具。
- Postman 是 API 平台。
- 預先遞交是 Git hooks 管理員。
- Projen 是專案產生器。
- pytest 是一種 Python 架構，用於撰寫小型、可讀取的測試。

程式碼儲存庫

您可以在 GitHub 非同步處理搭配 API Gateway 和 DynamoDB Streams 儲存庫中找到此架構程式碼範例。

最佳實務

- 此範例架構不包含對已部署基礎設施的監控。如果您的使用案例需要監控，請評估新增 CDK 監控建構或其他監控解決方案。
- 此範例架構使用 IAM 許可來控制對任務 API 的存取。有權擔任 的任何人JobsAPIInvokeRole都可以叫用任務 API。因此，存取控制機制是二進位。如果您的使用案例需要更複雜的授權模型，請使用不同的存取控制機制進行評估。
- 當使用者傳送 HTTP POST請求到/jobs任務 API 端點時，輸入資料會在兩個不同的層級進行驗證：
  - API Gateway 負責第一個請求驗證。
  - 事件處理函數會執行第二個請求。

    當使用者對/jobs/{jobId}任務 API 端點提出 HTTP GET請求時，不會執行驗證。如果您的使用案例需要額外的輸入驗證和更高的安全性，請使用 評估 AWS WAF 來保護您的 API。

- 為了避免限流，DynamoDB Streams 文件會阻止使用者從相同串流碎片中讀取兩個以上的取用者。若要擴展消費者數量，建議使用 Amazon Kinesis Data Streams。
- 此範例中已使用樂觀鎖定，以確保 jobs_table DynamoDB 資料表中項目的一致更新。根據使用案例需求，您可能需要實作更可靠的鎖定機制，例如漸進式鎖定。

史詩

設定環境

任務	Description	所需的技能
複製儲存庫。	若要在本機複製儲存庫，請執行下列命令： `git clone https://github.com/aws-samples/asynchronous-event-processing-api-gateway-dynamodb-streams-cdk.git`	DevOps 工程師

任務	Description	所需的技能

設定專案。	將目錄變更為儲存庫根目錄，並使用 Projen 設定 Python 虛擬環境和所有工具：
	`cd asynchronous-event-processing-api-gateway-api-gateway-dynamodb-streams-cdk`
	`npx projen`
		DevOps 工程師

安裝預先遞交掛鉤。	若要安裝預先遞交掛鉤，請執行下列動作：
	1. 啟用 Python 虛擬環境：
	`source .env/bin/activate`
	2. 安裝預先遞交掛鉤：
	`pre-commit install`
	`pre-commit install --hook-type commit-msg`
		DevOps 工程師

部署範例架構

任務	Description	所需的技能
引導 AWS CDK。	若要在 AWS CDK中引導 AWS 帳戶，請執行下列命令： `AWS_PROFILE=$YOUR_AWS_PROFILE npx projen bootstrap`	AWS DevOps

任務	Description	所需的技能
部署範例架構。	若要在 中部署範例架構 AWS 帳戶，請執行下列命令： `AWS_PROFILE=$YOUR_AWS_PROFILE npx projen deploy`	AWS DevOps

測試架構

任務	Description	所需的技能
安裝測試先決條件。	在工作站上安裝 AWS Command Line Interface (AWS CLI)、Postman 和 jq。 建議使用 Postman 測試此範例架構，但並非強制性。如果您選擇替代 API 測試工具，請確定它支援 AWS Signature 第 4 版身分驗證，並參考可透過匯出 REST API 檢查的公開 API 端點。	DevOps 工程師
擔任 JobsAPIInvokeRole 。	假設JobsAPIInvokeRole 從 deploy命令列印為輸出的 ： `CREDENTIALS=$(AWS_PROFILE=$<YOUR_AWS_PROFILE> aws sts assume-role \ --no-cli-pager \ --role-arn $<JOBS_API_INVOKE_ROLE_ARN> \ --role-session-name JobsAPIInvoke)`	AWS DevOps

任務	Description	所需的技能
	export AWS_ACCES S_KEY_ID=$(cat $CREDENTIALS \| jq '.Credentials''.Ac cessKeyId') export AWS_SECRE T_ACCESS_KEY=$(cat $CREDENTIALS \| jq '.Credentials''.Se cretAccessKey') export AWS_SESSI ON_TOKEN==$(cat $CREDENTIALS | jq '.Credentials''.Se ssionToken')	

任務	Description	所需的技能

設定 Postman。	• 若要匯入包含在儲存庫中的 Postman 集合，請遵循 Postman 文件中的指示。
	• 使用下列值設定 JobsAPI 變數：
	  • accessKey – 來自 assume-role 命令的 Credentials.AccessKeyId 屬性值。
	  • baseUrl – deploy 來自命令的 JobsApiJobsAPIEndpoint 輸出值，不含結尾斜線。
	  • region – AWS 區域 您部署範例架構的 值。
	  • seconds – 範例任務的輸入參數值。它必須是正整數。
	  • secretKey – 來自 assume-role 命令的 Credentials.SecretAccessKey 屬性值。
	  • sessionToken – 來自 assume-role 命令的 Credentials.SessionToken 屬性值。
		AWS DevOps

測試範例架構。	若要測試範例架構，請將請求傳送至任務 API。如需詳細資訊，請參閱 Postman 文件。	DevOps 工程師

疑難排解

問題	解決方案
範例架構的銷毀和後續重新部署失敗，因為 Amazon CloudWatch Logs 日誌群組 `/aws/apigateway/JobsAPIAccessLogs` 已存在。	1. 如有必要，請將日誌資料匯出至 Amazon Simple Storage Service (Amazon S3)。 2. 刪除 CloudWatch Logs 日誌群組 `/aws/apigateway/JobsAPIAccessLogs` 。 3. 重新部署範例架構。

相關資源

- API Gateway 映射範本和存取記錄變數參考
- 變更 DynamoDB Streams 的資料擷取
- 具有版本編號的樂觀鎖定
- 使用 Kinesis Data Streams 擷取對 DynamoDB 的變更

使用 Amazon API Gateway、Amazon SQS 和 AWS Fargate 非同步處理事件

Andrea Meroni、Mariem Kthiri、Nadim Majed、Alessandro Trisolini 和 Michael Wallner，Amazon Web Services

總結

Amazon API Gateway 是一項全受管服務，開發人員可用來建立、發佈、維護、監控和保護任何規模 APIs。它會處理涉及接受和處理多達數十萬個並行 API 呼叫的任務。

API Gateway 的一項重要服務配額是整合逾時。逾時是後端服務必須在 REST API 傳回錯誤之前傳回回應的最長時間。對於同步工作負載，通常可接受 29 秒的硬性限制。不過，該限制對想要將 API Gateway 與非同步工作負載搭配使用的開發人員來說是一項挑戰。

此模式顯示使用 API Gateway、Amazon Simple Queue Service (Amazon SQS) 和 非同步處理事件的範例架構 AWS Fargate。架構支援在沒有持續時間限制的情況下執行處理任務，並使用基本 REST API 做為界面。

Projen 用於設定本機開發環境 AWS 帳戶，並將範例架構與 AWS Cloud Development Kit (AWS CDK)、Docker 和 Node.js 結合部署至目標。Projen 會自動使用預先遞交和用於程式碼品質保證、安全掃描和單元測試的工具來設定 Python 虛擬環境。如需詳細資訊，請參閱工具一節。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 工作站上安裝了下列工具：
  - AWS Cloud Development Kit (AWS CDK) 工具組 2.85.0 版或更新版本
  - Docker 20.10.21 版或更新版本
  - Node.js 第 18 版或更新版本
  - Projen 0.71.111 版或更新版本
  - Python 3.9.16 版或更新版本

限制

- 並行任務限制為每分鐘 500 個任務，這是 Fargate 可以佈建的任務數量上限。

Architecture

下圖顯示任務 API 與 `jobs` Amazon DynamoDB 資料表、事件處理 Fargate 服務和錯誤處理 AWS Lambda 函數的互動。事件會存放在 Amazon EventBridge 事件存檔中。

典型的工作流程包括以下步驟：

1. 您可以驗證 AWS Identity and Access Management (IAM) 並取得安全登入資料。
2. 您可以將 HTTP `POST`請求傳送至`/jobs`任務 API 端點，在請求內文中指定任務參數。
3. 任務 API 是一種 API Gateway REST API，會傳回包含任務識別符的 HTTP 回應給您。
4. 任務 API 會傳送訊息至 SQS 佇列。
5. Fargate 從 SQS 佇列提取訊息、處理事件，然後將任務結果放入 `jobs` DynamoDB 資料表。
6. 您可以將 HTTP `GET`請求傳送至`/jobs/{jobId}`任務 API 端點，並將步驟 3 的任務識別符做為 `{jobId}`。
7. 任務 API 會查詢 `jobs` DynamoDB 資料表來擷取任務結果。
8. 任務 API 會傳回包含任務結果的 HTTP 回應。
9. 如果事件處理失敗，SQS 佇列會將事件傳送至無效字母佇列 (DLQ)。
10. EventBridge 事件會啟動錯誤處理函數。
11. 錯誤處理函數會將任務參數放在 `jobs` DynamoDB 資料表中。
12. 您可以透過傳送 HTTP `GET`請求至任務 API 端點來擷取`/jobs/{jobId}`任務參數。
13. 如果錯誤處理失敗，錯誤處理函數會將事件傳送至 EventBridge 封存。

    您可以使用 EventBridge 重播封存的事件。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端 基礎設施。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- AWS Fargate 可協助您執行容器，而無需管理伺服器或 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。它與 Amazon Elastic Container Service (Amazon ECS) 搭配使用。

- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如，Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他事件匯流排 AWS 帳戶。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- Amazon Simple Queue Service (Amazon SQS) 提供安全、耐用且可用的託管佇列，可協助您整合和分離分散式軟體系統和元件。

其他工具

- autopep8 會根據 Python Enhancement Proposal (PEP) 8 樣式指南自動格式化 Python 程式碼。
- Bandit 會掃描 Python 程式碼來尋找常見的安全問題。
- Commitizen 是 Git 遞交檢查程式和CHANGELOG產生器。
- cfn-lint 是 AWS CloudFormation linter
- Checkov 是一種靜態程式碼分析工具，可將基礎設施檢查為程式碼 (IaC) 是否有安全性和合規設定錯誤。
- jq 是用於剖析 JSON 的命令列工具。
- Postman 是 API 平台。
- 預先遞交是 Git hooks 管理員。
- Projen 是專案產生器。
- pytest 是一種 Python 架構，用於撰寫小型且可讀取的測試。

程式碼儲存庫

您可以在 GitHub 非同步處理與 API Gateway 和 SQS 儲存庫中找到此架構程式碼範例。

最佳實務

- 此範例架構不包含對已部署基礎設施的監控。如果您的使用案例需要監控，請評估新增 CDK 監控建構或其他監控解決方案。
- 此範例架構使用 IAM 許可來控制對任務 API 的存取。有權擔任 的任何人JobsAPIInvokeRole都可以叫用任務 API。因此，存取控制機制是二進位。如果您的使用案例需要更複雜的授權模型，請使用不同的存取控制機制進行評估。
- 當使用者傳送 HTTP POST請求到/jobs任務 API 端點時，輸入資料會在兩個不同的層級進行驗證：

- API Gateway 負責第一個請求驗證。
- 事件處理函數會執行第二個請求。

  當使用者對/jobs/{jobId}任務 API 端點提出 HTTP GET請求時，不會執行驗證。如果您的使用案例需要額外的輸入驗證和更高的安全性，請使用 評估 AWS WAF 來保護您的 API。

史詩

設定環境

任務	Description	所需的技能
複製儲存庫。	若要在本機複製儲存庫，請執行下列命令： `git clone https://github.com/aws-samples/asynchronous-event-processing-api-gateway-sqs-cdk.git`	DevOps 工程師

設定專案。	將目錄變更為儲存庫根目錄，並使用 Projen 設定 Python 虛擬環境和所有工具：
	`cd asynchronous-event-processing-api-gateway-api-gateway-sqs-cdk`
	`npx projen`
		DevOps 工程師

安裝預先遞交掛鉤。	若要安裝預先遞交掛鉤，請執行下列動作： 1. 啟用 Python 虛擬環境： `source .env/bin/activate`	DevOps 工程師

任務	Description	所需的技能

	2. 安裝預先遞交掛鉤：
	`pre-commit install`
	`pre-commit install --hook-type commit-msg`	

部署範例架構

任務	Description	所需的技能
引導 AWS CDK。	若要在 AWS CDK中引導 AWS 帳戶，請執行下列命令： `AWS_PROFILE=$YOUR_AWS_PROFILE npx projen bootstrap`	AWS DevOps
部署範例架構。	若要在 中部署範例架構 AWS 帳戶，請執行下列命令： `AWS_PROFILE=$YOUR_AWS_PROFILE npx projen deploy`	AWS DevOps

測試架構

任務	Description	所需的技能
安裝測試先決條件。	在工作站上安裝 AWS Command Line Interface (AWS CLI)、Postman 和 jq。 建議使用 Postman 測試此範例架構，但不強制。如果您選	DevOps 工程師

任務	Description	所需的技能
	擇替代 API 測試工具，請確定它支援 AWS Signature 第 4 版身分驗證，並參考可透過匯出 REST API 檢查的公開 API 端點。	
擔任 JobsAPIInvokeRole 。	假設JobsAPIInvokeRole 從 deploy命令列印為輸出的 ： `CREDENTIALS=$(AWS_PROFILE=$<YOUR_AWS_PROFILE> aws sts assume-role \ --no-cli-pager \ --role-arn $<JOBS_API_INVOKE_ROLE_ARN> \ --role-session-name JobsAPIInvoke) export AWS_ACCESS_KEY_ID=$(cat $CREDENTIALS \| jq '.Credentials''.AccessKeyId') export AWS_SECRET_ACCESS_KEY=$(cat $CREDENTIALS \| jq '.Credentials''.SecretAccessKey') export AWS_SESSION_TOKEN==$(cat $CREDENTIALS | jq '.Credentials''.SessionToken')`	AWS DevOps

任務	Description	所需的技能

設定 Postman。	• 若要匯入包含在儲存庫中的 Postman 集合，請遵循 Postman 文件中的指示。
	• 使用下列值設定 JobsAPI 變數：
	  • accessKey – 來自 assume-role 命令的 Credentials.AccessKeyId 屬性值。
	  • baseUrl – deploy 來自命令的 JobsApiJobsAPIEndpoint 輸出值，不含結尾斜線。
	  • region – AWS 區域 您部署範例架構的 值。
	  • seconds – 範例任務的輸入參數值。它必須是正整數。
	  • secretKey – 來自 assume-role 命令的 Credentials.SecretAccessKey 屬性值。
	  • sessionToken – 來自 assume-role 命令的 Credentials.SessionToken 屬性值。
		AWS DevOps

測試範例架構。	若要測試範例架構，請將請求傳送至任務 API。如需詳細資訊，請參閱 Postman 文件。	DevOps 工程師

疑難排解

問題	解決方案
範例架構的銷毀和後續重新部署失敗，因為 Amazon CloudWatch Logs 日誌群組/aws/apigateway/JobsAPIAccessLogs 已存在。	1. 如有必要，請將日誌資料匯出至 Amazon Simple Storage Service (Amazon S3)。 2. 刪除 CloudWatch Logs 日誌群組 /aws/apigateway/JobsAPIAccessLogs 。 3. 重新部署範例架構。
範例架構的銷毀和後續重新部署失敗，因為 CloudWatch Logs 日誌群組/aws/ecs/EventProcessingServiceLogs 已存在。	1. 如有必要，請將您的日誌資料匯出至 Amazon S3。 2. 刪除 CloudWatch Logs 日誌群組 /aws/ecs/EventProcessingServiceLogs. 3. 重新部署範例架構。

相關資源

- API Gateway 映射範本和存取記錄變數參考
- 如何將 API Gateway REST API 與 Amazon SQS 整合並解決常見錯誤？

從 AWS Step Functions 同步執行 AWS Systems Manager 自動化任務 AWS Step Functions

Elie El khoury，Amazon Web Services

總結

此模式說明如何 AWS Step Functions 與 整合 AWS Systems Manager。它使用 AWS SDK 服務整合，從狀態機器工作流程使用任務字符呼叫 Systems Manager startAutomationExecution API，並暫停直到字符傳回成功或失敗呼叫。為了示範整合，此模式會實作圍繞 `AWS-RunShellScript`或 文件的自動化`AWS-RunPowerShellScript`文件（執行手冊）包裝函式，並使用 `.waitForTaskToken` 同步呼叫 `AWS-RunShellScript`或 `AWS-RunPowerShellScript`。如需 Step Functions 中 AWS SDK 服務整合的詳細資訊，請參閱 AWS Step Functions 開發人員指南。

Step Functions 是一種低程式碼的視覺化工作流程服務，可用來建置分散式應用程式、自動化 IT 和業務流程，以及使用 AWS 服務建置資料和機器學習管道。工作流程會管理故障、重試、平行化、服務整合和可觀測性，讓您可以專注於更高價值的商業邏輯。

自動化是 的功能 AWS Systems Manager，可簡化常見的維護、部署和修復任務， AWS 服務 例如 Amazon Elastic Compute Cloud (Amazon EC2)、Amazon Relational Database Service (Amazon RDS)、Amazon Redshift 和 Amazon Simple Storage Service (Amazon S3)。自動化可讓您精細控制自動化的並行。例如，您可以指定要同時鎖定多少資源，以及在自動化停止之前可以發生的錯誤數量。

如需實作詳細資訊，包括 Runbook 步驟、參數和範例，請參閱其他資訊一節。

先決條件和限制

先決條件

- 作用中 AWS 的帳戶
- AWS Identity and Access Management 存取 Step Functions 和 Systems Manager 的 (IAM) 許可
- 執行個體上安裝 Systems Manager Agent (SSM Agent) 的 EC2 執行個體
- 連接至您計劃執行 Runbook 之執行個體的 Systems Manager IAM 執行個體描述檔
- 具有下列 IAM 許可的 Step Functions 角色（遵循最低權限原則）：

```
{
            "Effect": "Allow",
            "Action": "ssm:StartAutomationExecution",
```

```
            "Resource": "*"
}
```

產品版本

- SSM 文件結構描述 0.3 版或更新版本
- SSM Agent 2.3.672.0 版或更新版本

Architecture

目標技術堆疊

- AWS Step Functions
- AWS Systems Manager 自動化

目標架構

自動化和擴展

- 此模式提供 AWS CloudFormation 範本，可用來在多個執行個體上部署 Runbook。（請參閱 GitHub Step Functions 和 Systems Manager 實作儲存庫。）

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和區域的生命週期中管理資源。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務 來建置業務關鍵型應用程式。
- AWS Systems Manager 可協助您管理在 中執行的應用程式和基礎設施 AWS 雲端。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。

Code

此模式的程式碼可在 GitHub Step Functions 和 Systems Manager 實作儲存庫中使用。

史詩

建立 Runbook

任務	Description	所需的技能
下載 CloudFormation 範本。	從 GitHub 儲存庫的 `cloudformation` 資料夾下載`ssm-automation-documents.cfn.json` 範本。	AWS DevOps
建立 Runbook。	登入 AWS 管理主控台，開啟 CloudFormation 主控台，然後部署 範本。如需部署 CloudFormation 範本的詳細資訊，請參閱 CloudFormation 文件中的在 CloudFormation 主控台上建立堆疊。 CloudFormation 範本會部署三個資源： • `SfnRunCommandByInstanceIds` – 可讓您執行`AWS-RunShellScript` `AWS-RunPowerShellScript` 或使用執行個體 IDs Runbook。 • `SfnRunCommandByTargets` – 可讓您執行 `AWS-RunPowerShellScript` `AWS-RunShellScript` 或使用目標的 Runbook。 • `SSMSyncRole` – Runbook 擔任的 IAM 角色。	AWS DevOps

建立範例狀態機器

任務	Description	所需的技能
建立測試狀態機器。	遵循 AWS Step Functions 開發人員指南中的指示來建立和執行狀態機器。針對 定義，請使用下列程式碼。請務必使用帳戶中啟用 Systems Manager 之有效執行個體的 ID 來更新`InstanceIds` 值。 `{ "Comment": "A description of my state machine", "StartAt": "StartAutomationWaitForCallBack", "States": { "StartAutomationWaitForCallBack": { "Type": "Task", "Resource": "arn:aws:states:::aws-sdk:ssm:startAutomationExecution.waitForTaskToken", "Parameters": { "DocumentName": "SfnRunCommandByInstanceIds", "Parameters": { "InstanceIds": ["i-1234567890abcdef0"], "taskToken.$": "States.Array($$.Task.Token)",`	AWS DevOps

任務	Description	所需的技能

	"workingD
	irectory": [
	"/home/ssm-
	user/"
	],
	"Commands": [
	"echo \"This
	is a test running
	automation waitForTa
	skToken\" >> automatio
	n.log",
	"sleep 100"
	],
	"executio
	nTimeout": [
	"10800"
	],
	"delivery
	Timeout": [
	"30"
	],
	"shell": [
	"Shell"
	]
	}
	},
	"End": true
	}
	}
	}
	
	此程式碼會呼叫 Runbook 來執行兩個命令，以示範對 Systems Manager Automation 的`waitForTaskToken` 呼叫。
	
	`shell` 參數值 (`Shell` 或 `PowerShell`) 會決定自動化文件是執行 `AWS-RunSh`	

任務	Description	所需的技能
	ellScript 還是 AWS-RunPowerShellScript 。 任務會將「這是執行自動化 waitForTaskToken」的測試寫入 /home/ssm-user/automation.log 檔案，然後休眠 100 秒，再以任務字符回應，並釋出工作流程中的下一個任務。 如果您想要改為呼叫 SfnRunCommandByTargets Runbook，請將上一個程式碼的 Parameters 區段取代為下列項目： `"Parameters": { "Targets": [ { "Key": "InstanceIds", "Values": [ "i-02573cafcfEXAMPLE", "i-0471e04240EXAMPLE" ] } ],`	

任務	Description	所需的技能
更新狀態機器的 IAM 角色。	上一個步驟會自動為狀態機器建立專用 IAM 角色。不過，它不會授予呼叫 Runbook 的許可。透過新增下列許可來更新角色： `{ "Effect": "Allow", "Action": "ssm:StartAutomationExecution", "Resource": "*" }`	AWS DevOps
驗證同步呼叫。	執行狀態機器以驗證 Step Functions 與 Systems Manager Automation 之間的同步呼叫。 如需範例輸出，請參閱其他資訊一節。	AWS DevOps

相關資源

- 入門 AWS Step Functions(AWS Step Functions 開發人員指南)
- 使用任務字符等待回呼 AWS Step Functions （開發人員指南，服務整合模式）
- send_task_success 和 send_task_failure API 呼叫 (Boto3 文件）
- AWS Systems Manager 自動化 (AWS Systems Manager 使用者指南)

其他資訊

實作詳細資訊

此模式提供 CloudFormation 範本，可部署兩個 Systems Manager Runbook：

- SfnRunCommandByInstanceIds會使用執行個體 IDs 執行 AWS-RunShellScript或 AWS-RunPowerShellScript命令。
- SfnRunCommandByTargets使用目標執行 AWS-RunShellScript或 AWS-RunPowerShellScript命令。

每個 Runbook 都會實作四個步驟，以在 Step Functions 中使用 .waitForTaskToken選項時實現同步呼叫。

步驟	Action	Description
1	Branch	檢查shell參數值 (Shell 或 PowerShell)，以決定是否要AWS-RunShellScript 針對 Linux 或 AWS-RunPowerShellScript Windows 執行 。
2	RunCommand_Shell 或 RunCommand_PowerShell	接受數個輸入並執行 RunShellScript 或 RunPowerShellScript 命令。如需詳細資訊，請參閱 Systems Manager 主控台上 RunCommand_Shell 或RunCommand_PowerShell 自動化文件的詳細資訊索引標籤。
3	SendTaskFailure	步驟 2 中止或取消時執行。它呼叫 Step Functions send_task_failure API，接受三個參數做為輸入：狀態機器傳遞的字符、失敗錯誤，以及失敗原因的描述。
4	SendTaskSuccess	步驟 2 成功時執行。它呼叫 Step Functions send_task

_success API，該 API 接受狀態機器傳遞的字符作為輸入。

Runbook 參數

SfnRunCommandByInstanceIds Runbook：

參數名稱	Type	選用或必要	Description
shell	String	必要	執行個體 shell，用於決定是否AWS-RunShellScript 要針對 Linux 或 Windows AWS-RunPowerShellScript 執行。
deliveryTimeout	Integer	選用	等待命令交付至執行個體上 SSM 代理程式的時間，以秒為單位。此參數的最小值為 30 (0.5 分鐘），最大值為 2592000 (720 小時）。
executionTimeout	String	選用	命令在被視為失敗之前完成的時間，以秒為單位。預設值為 3600 (1 小時）。最大值為 172800 (48 小時）。
workingDirectory	String	選用	在您的執行個體上的工作目錄路徑。
Commands	StringList	必要	要執行的 shell 指令碼或命令。

InstanceIds	StringList	必要	您要執行命令之執行個體的 IDs。
taskToken	String	必要	用於回呼回應的任務字符。

SfnRunCommandByTargetsRunbook：

Name	類型	選用或必要	Description
shell	String	必要	執行個體 shell，以決定是否AWS-RunShellScript 要針對 Linux 或 Windows AWS-RunPowerShellScript 執行。
deliveryTimeout	Integer	選用	等待命令交付至執行個體上 SSM 代理程式的時間，以秒為單位。此參數的最小值為 30 (0.5 分鐘），最大值為 2592000 (720 小時）。
executionTimeout	Integer	選用	命令在被視為失敗之前完成的時間，以秒為單位。預設值為 3600 (1 小時）。最大值為 172800 (48 小時）。
workingDirectory	String	選用	在您的執行個體上的工作目錄路徑。

Commands	StringList	必要	要執行的 shell 指令碼或命令。
Targets	MapList	必要	使用您指定的鍵值對來識別執行個體的搜尋條件陣列。例如：[{"Key":"InstanceIds","Values":["i-02573cafcfEXAMPLE","i-0471e04240EXAMPLE"]}]
taskToken	String	必要	用於回呼回應的任務字符。

範例輸出

下表提供 步驟函數的範例輸出。它顯示步驟 5 (TaskSubmitted) 和步驟 6 () 之間的總執行時間超過 100 秒TaskSucceeded。這會示範步驟函數等待sleep 100命令完成，然後再移至工作流程中的下一個任務。

ID	Type	步驟	資源	經過時間（毫秒）	時間戳記
1	ExecutionStarted		-	0	2022 年 3 月 11 日下午 02：50：34.303
2	TaskStateEntered	StartAutomationWaitForCallBack	-	40	2022 年 3 月 11 日下午 02：50：34.343

3	TaskScheduled	StartAutomationWaitForCallBack	-	40	2022 年 3 月 11 日下午 02：50：34.343
4	TaskStarted	StartAutomationWaitForCallBack	-	154	2022 年 3 月 11 日下午 02：50：34.457
5	TaskSubmitted	StartAutomationWaitForCallBack	-	657	2022 年 3 月 11 日下午 02：50：34.960
6	TaskSucceeded	StartAutomationWaitForCallBack	-	103835	2022 年 3 月 11 日下午 02：52：18.138
7	TaskStateExited	StartAutomationWaitForCallBack	-	103860	2022 年 3 月 11 日下午 02：52：18.163
8	ExecutionSucceeded		-	103897	2022 年 3 月 11 日下午 02：52：18：200

在 AWS Lambda 函數中使用 Python 執行 S3 物件的平行讀取

Eduardo Bortoluzzi，Amazon Web Services

總結

您可以使用此模式從 Amazon Simple Storage Service (Amazon S3) 儲存貯體即時擷取和摘要文件清單。模式提供範例程式碼，從 Amazon Web Services () 上的 S3 儲存貯體平行讀取物件AWS。模式示範如何使用 Python 使用 AWS Lambda 函數有效率地執行 I/O 繫結任務。

一家金融公司在互動式解決方案中使用此模式，以即時手動核准或拒絕相互關聯的金融交易。金融交易文件存放在與市場相關的 S3 儲存貯體中。運算子從 S3 儲存貯體中選取文件清單，分析解決方案計算的交易總值，並決定核准或拒絕選取的批次。

I/O 繫結任務支援多個執行緒。在此範例程式碼中，並行.futures.ThreadPoolExecutor 最多可與 30 個同時執行緒搭配使用，即使 Lambda 函數最多支援 1，024 個執行緒 （其中一個執行緒是您的主要程序）。此限制是因為太多執行緒會因為內容切換和運算資源的使用率而產生延遲問題。您也需要在 中增加集區連線上限，botocore以便所有執行緒可以同時執行 S3 物件下載。

範例程式碼在 S3 儲存貯體中使用一個 8.3 KB 物件搭配 JSON 資料。物件會多次讀取。Lambda 函數讀取物件後，JSON 資料會解碼為 Python 物件。在 2024 年 12 月，執行此範例後的結果為 2.3 秒內處理的 1，000 次讀取，以及 27 秒內處理的 10，000 次讀取，其使用 Lambda 函數設定為 2，304 MB 記憶體。 AWS Lambda 支援從 128 MB 到 10，240 MB (10 GB) 的記憶體組態，但將 Lambdamemory 增加到超過 2，304 MB 並有助於縮短執行此特定 I/O 繫結任務的時間。

AWS Lambda Power Tuning 工具用於測試不同的 Lambda 記憶體組態，並驗證任務的最佳 performance-to-cost比。如需測試結果，請參閱其他資訊一節。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- Python 開發的熟練度

限制

- Lambda 函數最多可以有 1，024 個執行程序或執行緒。
- 新 AWS 帳戶 的 Lambda 記憶體限制為 3，008 MB。相應地調整 AWS Lambda Power Tuning 工具。如需詳細資訊，請參閱疑難排解一節。

- Amazon S3 每個分割字首每秒限制 5，500 個 GET/HEAD 請求。

產品版本

- Python 3.9 或更高版本
- AWS Cloud Development Kit (AWS CDK) v2
- AWS Command Line Interface (AWS CLI) 第 2 版
- AWS Lambda Power Tuning 4.3.6 （選用）

Architecture

目標技術堆疊

- AWS Lambda
- Amazon S3
- AWS Step Functions （如果已部署 AWS Lambda 電源調校）

目標架構

下圖顯示從 S3 儲存貯體平行讀取物件的 Lambda 函數。此圖表也有 Step Functions 工作流程，可供 AWS Lambda Power Tuning 工具微調 Lambda 函數記憶體。這種微調有助於在成本和效能之間取得良好的平衡。

自動化和擴展

Lambda 函數會在需要時快速擴展。為了避免 Amazon S3 在高需求期間發生 503 緩慢下降錯誤，我們建議對擴展設定一些限制。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) v2 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端 基礎設施。已建立要部署的範例基礎設施 AWS CDK。
- AWS Command Line InterfaceAWS CLI 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。在此模式中，第 2 AWS CLI 版用於上傳範例 JSON 檔案。

- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- Amazon Simple Storage Service Amazon S3 是一種雲端型物件儲存服務，可協助您存放、保護和擷取任意數量的資料。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務來建置業務關鍵應用程式。

其他工具

- Python 是一種一般用途的電腦程式設計語言。Python 3.8 版中推出了閒置工作者執行緒的重複使用，此模式中的 Lambda 函數程式碼是針對 Python 3.9 版和更新版本建立的。

程式碼儲存庫

此模式的程式碼可在 aws-lambda-parallel-download GitHub 儲存庫中使用。

最佳實務

- 此 AWS CDK 建構倚賴 AWS 帳戶您 的使用者許可來部署基礎設施。如果您打算使用 AWS CDK 管道或跨帳戶部署，請參閱堆疊合成器。
- 此範例應用程式未在 S3 儲存貯體啟用存取日誌。最佳實務是在生產程式碼中啟用存取日誌。

史詩

準備開發環境

任務	Description	所需的技能
檢查 Python 安裝的版本。	此程式碼已專門在 Python 3.9 和 Python 3.13 上進行測試，且應適用於這些版本之間的所有版本。若要檢查您的 Python 版本，請在終端機`python3 -V`中執行 ，並視需要安裝較新的版本。	雲端架構師

任務	Description	所需的技能
	若要驗證已安裝必要的模組，請執行 `python3 -c "import pip, venv"`。沒有錯誤訊息表示模組已正確安裝，您已準備好執行此範例。	
安裝 AWS CDK。	若要在尚未安裝 AWS CDK 時安裝 ，請遵循 入門 AWS CDK中的指示。若要確認已安裝的 AWS CDK 版本是 2.0 或更新版本，請執行 `cdk --version`。	雲端架構師
引導您的環境。	若要引導您的環境，如果尚未完成，請依照引導環境的指示搭配 使用 AWS CDK。	雲端架構師

複製範例儲存庫

任務	Description	所需的技能
複製儲存庫。	若要複製最新版本的儲存庫，請執行下列命令： `git clone --depth 1 --branch v1.2.0 \ git@github.com:aws-samples/aws-lambda-parallel-download.git`	雲端架構師
將工作目錄變更為複製的儲存庫。	執行以下命令： `cd aws-lambda-parallel-download`	雲端架構師

任務	Description	所需的技能
建立 Python 虛擬環境。	若要建立 Python 虛擬環境，請執行下列命令： `python3 -m venv .venv`	雲端架構師
啟用虛擬環境。	若要啟用虛擬環境，請執行下列命令： `source .venv/bin/activate`	雲端架構師
安裝相依性。	若要安裝 Python 相依性，請執行 `pip`命令： `pip install -r requirements.txt`	雲端架構師
瀏覽程式碼。	（選用）從 S3 儲存貯體下載物件的範例程式碼位於 `resources/parallel.py` 。 基礎設施程式碼位於 `parallel_download` 資料夾中。	雲端架構師

部署和測試應用程式

任務	Description	所需的技能
部署應用程式。	執行 `cdk deploy`。 寫下 AWS CDK 輸出：	雲端架構師

任務	Description	所需的技能
	• ParallelDownloadStack.LambdaFunctionARN • ParallelDownloadStack.SampleS3BucketName • ParallelDownloadStack.StateMachineARN	
上傳範例 JSON 檔案。	儲存庫包含約 9 KB 的範例 JSON 檔案。若要將檔案上傳至已建立堆疊的 S3 儲存貯體，請執行下列命令： `aws s3 cp sample.json s3://<ParallelDownloadStack.SampleS3BucketName>` <ParallelDownloadStack.SampleS3BucketName> 將 取代為 AWS CDK 輸出中的對應值。	雲端架構師

任務	Description	所需的技能
執行應用程式。	若要執行應用程式，請執行下列動作： 1. 登入 AWS 管理主控台，導覽至 Lambda 主控台，然後找到具有來自 AWS CDK 輸出 之 ARN 的 Lambda 函數`ParallelDownloadStack.LambdaFunction` ARN 。 2. 在測試索引標籤上，將事件 JSON 變更為下列項目： `{"objectKey": "sample.json"}` 3. 選擇測試。 4. 若要查看結果，請選擇詳細資訊。詳細資訊會顯示平行下載的統計資料、執行的資訊和日誌。	雲端架構師
新增下載次數。	（選用）若要執行 1，500 取得物件呼叫，請在 `Test` 參數的事件 JSON 中使用下列 JSON： `{"repeat": 1500, "objectKey": "sample.json"}`	雲端架構師

選用：執行 AWS Lambda 電源調校

任務	Description	所需的技能
執行 AWS Lambda Power Tuning 工具。	1. 登入 主控台，然後導覽至 Step Functions。 2. 從 AWS CDK 輸出 找到具有 ARN 的狀態機器`ParallelDownloadStack.StateMachineARN` 。 3. 選擇開始執行，然後貼上下列 JSON： `{ "lambdaARN": "<ParallelDownloadStack.LambdaFunctionARN>", "num": 10, "strategy": "balanced", "payload": {"repeat": 2000, "objectKey": "sample.json"} }` 請記得將 取代`<ParallelDownloadStack.LambdaFunctionARN>` 為 AWS CDK 輸出中的 值。 在執行結束時，結果會在執行輸入和輸出索引標籤上。	雲端架構師
在圖形中檢視 AWS Lambda 電源調校結果。	在執行輸入和輸出索引標籤上，複製 `visualization`	雲端架構師

任務	Description	所需的技能
	屬性連結，並將其貼入新的瀏覽器索引標籤。	

清除

任務	Description	所需的技能
從 S3 儲存貯體移除物件。	在您銷毀已部署的資源之前，請先從 S3 儲存貯體移除所有物件： `aws s3 rm s3://<ParallelDownloadStack.SampleS3BucketName> \ --recursive` 請記得將 取代`<ParallelDownloadStack.SampleS3BucketName>` 為 AWS CDK 輸出中的 值。	雲端架構師
銷毀資源。	若要銷毀為此試驗建立的所有資源，請執行下列命令： `cdk destroy`	雲端架構師

疑難排解

問題	解決方案
`'MemorySize' value failed to satisfy constraint: Member must`	對於新帳戶，您可能無法在 Lambda 函數中設定超過 3，008 MB。若要使用 AWS Lambda Power Tuning 進行測試，請在啟動 Step

問題	解決方案
have value less than or equal to 3008	Functions 執行時，在輸入 JSON 新增下列屬性： `"powerValues": [ 512, 1024, 1536, 2048, 2560, 3008 ]`

相關資源

- Python – concurrent.futures.ThreadPoolExecutor
- Lambda 配額 – 函數組態、部署和執行
- 在 Python AWS CDK 中使用
- 使用 AWS Lambda Power Tuning 分析函數

其他資訊

Code

下列程式碼片段會執行平行 I/O 處理：

```
with ThreadPoolExecutor(max_workers=MAX_WORKERS) as executor:
  for result in executor.map(a_function, (the_arguments)):
    ...
```

當執行緒可用時， 會ThreadPoolExecutor重複使用執行緒。

測試和結果

這些測試是在 2024 年 12 月進行。

第一個測試處理了 2，500 個物件讀取，結果如下。

從 3，009 MB 開始，處理時間層級在任何記憶體增加時幾乎保持不變，但成本會隨著記憶體大小增加而增加。

另一項測試使用 256 MB 的倍數和處理 10，000 個物件讀取的值，調查了介於 1，536 MB 和 3，072 MB 之間的記憶體範圍，結果如下。

最佳performance-to-cost比率是使用 2，304 MB 記憶體 Lambda 組態。

為了進行比較，2，500 個物件讀取的循序程序需要 47 秒。使用 2，304 MB Lambda 組態的平行程序需要 7 秒，減少 85%。

將遙測資料從 AWS Lambda 傳送至 OpenSearch，以進行即時分析和視覺化

Tabby Ward、Guy Bachar 和 David Kilzer，Amazon Web Services

總結

現代應用程式越來越分散和事件驅動，這加強了即時監控和可觀測性的需求。 AWS Lambda 是一種無伺服器運算服務，在建置可擴展性和事件驅動型架構方面扮演關鍵角色。不過，如果您只依賴 Amazon CloudWatch Logs，監控和疑難排解 Lambda 函數可能具有挑戰性，這可能會導致延遲和有限的保留期。

為了解決此挑戰， AWS 推出了 Lambda 遙測 API，可讓 Lambda 函數將遙測資料直接傳送到第三方監控和可觀測性工具。此 API 支援日誌、指標和追蹤的即時串流，並提供 Lambda 函數效能和運作狀態的完整及時檢視。

此模式說明如何整合 Lambda 遙測 API 與 OpenSearch，這是一種開放原始碼的分散式搜尋和分析引擎。OpenSearch 提供強大且可擴展的平台，用於擷取、儲存和分析大量資料，這使得它成為 Lambda 遙測資料的理想選擇。具體而言，此模式示範如何使用 提供的 Lambda 擴充功能，將日誌從以 Python 撰寫的 Lambda 函數直接傳送至 OpenSearch 叢集 AWS。此解決方案靈活且可自訂，因此您可以建立自己的 Lambda 延伸模組，或變更範例原始程式碼，視需要變更輸出格式。

模式說明如何設定和設定 Lambda 遙測 API 與 OpenSearch 的整合，並包含安全性、成本最佳化和可擴展性的最佳實務。目標是協助您深入了解 Lambda 函數，並增強無伺服器應用程式的整體可觀測性。

> 注意：此模式著重於整合 Lambda 遙測 API 與受管 OpenSearch。不過，討論的原則和技術也適用於自我管理的 OpenSearch 和 Elasticsearch。

先決條件和限制

開始整合程序之前，請確定您已備妥下列先決條件：

AWS 帳戶：具備適當許可 AWS 帳戶 的作用中 ，可建立和管理下列 AWS 資源：

- AWS Lambda
- AWS Identity and Access Management (IAM)
- Amazon OpenSearch Service （ 如果您使用受管 OpenSearch 叢集 ）

OpenSearch 叢集：

- 您可以使用現有的自我管理 OpenSearch 叢集或受管服務，例如 OpenSearch Service。
- 如果您使用的是 OpenSearch Service，請依照 OpenSearch Service 文件中的 Amazon OpenSearch Service 入門中的指示來設定 OpenSearch 叢集。
- 請確定 OpenSearch 叢集可從 Lambda 函數存取，並使用存取政策、加密和身分驗證等必要的安全設定進行設定。
- 使用必要的索引映射和設定來設定 OpenSearch 叢集，以擷取 Lambda 遙測資料。如需詳細資訊，請參閱 OpenSearch Service 文件中的將串流資料載入 Amazon OpenSearch Service。

網路連線：

- 確保您的 Lambda 函數具有存取 OpenSearch 叢集所需的網路連線能力。如需如何設定虛擬私有雲端 (VPC) 設定的指引，請參閱 OpenSearch Service 文件中的在 VPC 內啟動 Amazon OpenSearch Service 網域。 OpenSearch

IAM 角色和政策：

- 建立具有 Lambda 函數必要許可的 IAM 角色，以存取 OpenSearch 叢集並存取存放在其中的登入資料 AWS Secrets Manager。
- 將適當的 IAM 政策連接到角色，例如`AWSLambdaBasicExecutionRole`政策以及與 OpenSearch 互動所需的任何其他許可。
- 確認授予 Lambda 函數的 IAM 許可允許它將資料寫入 OpenSearch 叢集。如需有關管理 IAM 許可的資訊，請參閱 Lambda 文件中的使用 執行角色定義 Lambda 函數許可。

程式設計語言知識：

- 您需要 Python （或您選擇的程式設計語言） 的基本知識，才能了解和修改 Lambda 函數和 Lambda 延伸模組的範例程式碼。

開發環境：

- 使用建置和部署 Lambda 函數和延伸所需的工具和相依性來設定本機開發環境。

AWS CLI 或 AWS 管理主控台：

- 安裝並設定 AWS Command Line Interface (AWS CLI)，或使用 AWS 管理主控台 具有適當登入資料的 與所需的 互動 AWS 服務。

監控和記錄：

- 熟悉 的監控和記錄最佳實務 AWS，包括 Amazon CloudWatch 等服務，以及 AWS CloudTrail 用於監控和稽核目的。
- 檢查您的 Lambda 函數的 CloudWatch Logs，以識別與 Lambda 遙測 API 整合相關的任何錯誤或例外狀況。如需疑難排解指引，請參閱 Lambda 遙測 API 文件。

Architecture

此模式使用 OpenSearch Service 來存放由 Lambda 函數產生的日誌和遙測資料。此方法可讓您將日誌直接快速串流至 OpenSearch 叢集，進而減少使用 CloudWatch Logs 做為中介裝置的延遲和相關成本。

> **Note**
>
> 您的 Lambda 延伸程式碼可以直接使用 OpenSearch API 或使用 OpenSearch 用戶端程式庫，將遙測推送至 OpenSearch Service。Lambda 延伸模組可以使用 OpenSearch API 支援的大量操作，將遙測事件批次在一起，並在單一請求中將其傳送至 OpenSearch Service。

下列工作流程圖說明當您使用 OpenSearch 叢集做為端點時，Lambda 函數的日誌工作流程。

架構包含下列元件：

- Lambda 函數：在執行期間產生日誌和遙測資料的無伺服器函數。
- Lambda 延伸模組：Python 型延伸模組，使用 Lambda 遙測 API 直接與 OpenSearch 叢集整合。此延伸項目會在相同的執行環境中與 Lambda 函數一起執行。
- Lambda 遙測 API：此 API 可讓 Lambda 延伸模組直接將遙測資料傳送至第三方監控和可觀測性工具，包括日誌、指標和追蹤。
- Amazon OpenSearch Service 叢集：託管於 的受管 OpenSearch 叢集 AWS。此叢集負責透過 Lambda 延伸來擷取、儲存和索引從 Lambda 函數串流的日誌資料。

工作流程包含下列步驟：

1. Lambda 函數稱為 ，並在其執行期間產生日誌和遙測資料。
2. Lambda 延伸項目會與 函數一起執行，以使用 Lambda 遙測 API 擷取日誌和遙測資料。
3. Lambda 擴充功能會與 OpenSearch Service 叢集建立安全連線，並即時串流日誌資料。
4. OpenSearch Service 叢集會擷取、編製索引和存放日誌資料，以便透過使用 Kibana 或其他相容應用程式等工具進行搜尋、分析和視覺化。

透過規避 CloudWatch Logs 並將日誌資料直接傳送到 OpenSearch 叢集，此解決方案提供數種優點：

- 即時日誌串流和分析，可更快速進行故障診斷並改善可觀測性。
- 減少與 CloudWatch Logs 相關的延遲和潛在的保留限制。
- 自訂 Lambda 延伸模組或為特定輸出格式或其他處理建立您自己的延伸模組的彈性。
- 與 OpenSearch Service 的搜尋、分析和視覺化功能整合，以進行日誌分析和監控。

Epics 區段提供step-by-step說明。 OpenSearch 如需監控和疑難排解解決方案的安全考量、成本最佳化策略和秘訣，請參閱最佳實務一節。

工具

AWS 服務

- AWS Lambda 是一種運算服務，讓您無需設定或管理伺服器即可運行程式碼。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。
- Amazon OpenSearch Service 是 提供的全受管服務 AWS ，可讓您輕鬆地在雲端中部署、操作和擴展 OpenSearch 叢集。
- Lambda 延伸模組透過執行自訂程式碼來擴展 Lambda 函數的功能。您可以使用 Lambda 擴充功能，將 Lambda 與各種監控、可觀測性、安全性和控管工具整合。
- AWS Lambda 遙測 API 可讓您使用擴充功能直接從 Lambda 擷取增強型監控和可觀測性資料，並將其傳送至您選擇的目的地。
- CloudFormation 可協助您建立和設定 AWS 資源的模型，以減少管理這些資源的時間，並有更多時間專注於您的應用程式。

程式碼儲存庫

- AWS Lambda 延伸項目包含來自 和 AWS 合作夥伴的示範 AWS 和範例專案，協助您開始建置自己的延伸項目。
- 適用於 OpenSearch 的 Lambda 遙測整合範例提供 Lambda 延伸範例，示範如何將日誌從 Lambda 函數傳送至 OpenSearch 叢集。

其他工具

- OpenSearch 是一種開放原始碼分散式搜尋和分析引擎，提供強大的平台，用於擷取、儲存和分析大量資料。
- Kibana 是一種開放原始碼資料視覺化和探勘工具，可與 OpenSearch 搭配使用。請注意，視覺化和分析的實作超出此模式的範圍。如需詳細資訊，請參閱 Kibana 文件和其他資源。

最佳實務

當您將 Lambda 遙測 API 與 OpenSearch 整合時，請考慮下列最佳實務。

安全性和存取控制

- 安全通訊：使用 HTTPS 加密 Lambda 函數與 OpenSearch 叢集之間的所有通訊。在 Lambda 延伸模組和 OpenSearch 組態中設定必要的 SSL/TLS 設定。
- IAM 許可：
  - 延伸模組會在與 Lambda 函數相同的執行環境中執行，因此會繼承相同層級的資源存取權，例如檔案系統、聯網和環境變數。
  - 將最低必要 IAM 許可授予 Lambda 函數，以存取 Lambda 遙測 API 並將資料寫入 OpenSearch 叢集。使用最低權限原則來限制許可範圍。
- OpenSearch 存取控制：在您的 OpenSearch 叢集中實作精細存取控制，以限制對敏感資料的存取。使用 OpenSearch 中的內建安全功能，例如使用者身分驗證、角色型存取控制和索引層級許可。
- 信任的延伸模組：一律只從信任的來源安裝延伸模組。使用基礎設施做為程式碼 (IaC) 工具，例如 CloudFormation ，以簡化將相同的延伸組態，包括 IAM 許可連接至多個 Lambda 函數的程序。IaC 工具也提供先前使用的擴充功能和版本的稽核記錄。
- 敏感資料處理：建置擴充功能時，請避免記錄敏感資料。在記錄或保留承載和中繼資料以進行稽核之前，請對其進行清理。

成本最佳化

- 監控和提醒：設定監控和提醒機制，以追蹤從 Lambda 函數傳送至 OpenSearch 的資料量。這將協助您識別和解決任何潛在的成本超支。
- 資料保留：仔細考慮 OpenSearch 中 Lambda 遙測資料的適當資料保留期間。較長的保留期可能會增加儲存成本，因此在可觀測性需求與成本最佳化之間取得平衡。
- 壓縮和索引：啟用資料壓縮並最佳化 OpenSearch 索引策略，以減少 Lambda 遙測資料的儲存體使用量。
- 減少對 CloudWatch 的依賴：透過直接整合 Lambda 遙測 API 與 OpenSearch，您可以減少對 CloudWatch Logs 的依賴，進而節省成本。這是因為 Lambda 遙測 API 可讓您將日誌直接傳送至 OpenSearch，而不需要在 CloudWatch 中存放和處理資料。

可擴展性和可靠性

- 非同步處理：使用非同步處理模式，例如 Amazon Simple Queue Service (Amazon SQS) 或 Amazon Kinesis，將 Lambda 函數執行與 OpenSearch 資料擷取分離。這有助於維持 Lambda 函數的回應能力，並改善系統的整體可靠性。
- OpenSearch 叢集擴展：監控 OpenSearch 叢集的效能和資源使用率，並視需要進行擴展或縮減，以處理不斷增加的 Lambda 遙測資料量。
- 容錯移轉和災難復原：為您的 OpenSearch 叢集實作強大的災難復原策略，包括定期備份，以及在發生故障時快速還原資料的能力。

可觀測性和監控

- 儀表板和視覺化：使用 Kibana 或其他儀表板工具建立自訂儀表板和視覺化，根據 OpenSearch 中的遙測資料，提供 Lambda 函數效能和運作狀態的洞見。
- 警示和通知：設定警示和通知，以主動監控 Lambda 函數中的異常、錯誤或效能問題。將這些提醒和通知與您現有的事件管理程序整合。
- 追蹤和相互關聯：確保您的 Lambda 遙測資料包含相關的追蹤資訊，例如請求 IDs 或相互關聯 IDs，以便在分散式無伺服器應用程式中啟用end-to-end可觀測性和故障診斷。

透過遵循這些最佳實務，您可以確保 Lambda 遙測 API 與 OpenSearch 的整合安全、經濟實惠且可擴展，並為無伺服器應用程式提供全方位的可觀測性。

史詩

建置和部署 Lambda 延伸層

任務	Description	所需的技能
下載原始程式碼。	從延伸模組儲存庫下載範例AWS Lambda 延伸模組。	應用程式開發人員、雲端架構師
導覽至 `python-example-telemetry-opensearch-extension` 資料夾。	您下載AWS Lambda 的延伸模組儲存庫包含多個使用案例和語言執行時間的許多範例。導覽至 python-example-telemetry-opensearch-extension 資料夾，以使用 Python OpenSearch 延伸模組，將日誌傳送至 OpenSearch。	應用程式開發人員、雲端架構師
新增執行延伸端點的許可。	執行下列命令，讓延伸端點可執行： `chmod +x python-example-telemetry-opensearch-extension/extension.py`	應用程式開發人員、雲端架構師
在本機安裝延伸模組相依性。	執行下列命令來安裝 Python 程式碼的本機相依性： `pip3 install -r python-example-telemetry-opensearch-extension/requirements.txt -t ./python-example-telemetry-opensearch-extension/`	應用程式開發人員、雲端架構師

任務	Description	所需的技能
	這些相依性將與延伸程式碼一起掛載。	

為延伸項目建立 .zip 套件，以將其部署為 layer。	副檔名 .zip 檔案應該包含一個名為 的根目錄`extensions/`，其中副檔名可執行檔名所在的位置，以及另一個名為 的根目錄`python-example-telemetry-opensearch-extension/`，其中副檔名的核心邏輯及其相依性所在的位置。
	
	為擴充功能建立 .zip 套件：
	
	`chmod +x extensions/python-example-telemetry-opensearch-extension`
	`zip -r extension.zip extensions python-example-telemetry-opensearch-extension`
		應用程式開發人員、雲端架構師

將延伸模組部署為 Lambda 層。	使用您的副檔名 .zip 檔案和下列命令發佈 layer：
	
	`aws lambda publish-layer-version \`
	`--layer-name "python-example-telemetry-opensearch-extension" \`
	`--zip-file "fileb://extension.zip"`
		應用程式開發人員、雲端架構師

將延伸模組整合到您的 函數

任務	Description	所需的技能
將圖層新增到您的函式中。	1. 登入 AWS 管理主控台 並開啟 AWS Lambda 主控台的函數頁面。 2. 選取您的 函數。 3. 在圖層下，選擇新增圖層。 4. 在選擇圖層下，選擇自訂圖層做為圖層來源，然後新增圖層。 如需將圖層新增至 Lambda 函數的詳細資訊，請參閱 Lambda 文件。	應用程式開發人員、雲端架構師
設定 函數的環境變數。	在函數頁面上，選擇組態索引標籤，並將下列環境變數新增至函數： • URL – 將傳送日誌之 OpenSearch 端點的 URI。 • AUTH_SECRET – 存放於 的 OpenSearch 登入資料的 ARN AWS Secrets Manager。這應該儲存為索引鍵/值對，並有兩個索引鍵： username和 password。 • PLATFORM_INDEX 、 FUNCTION_INDEX 和 EXTENSION_INDEX – 存放遙測資料、函數日誌和延伸日誌的索引名稱。確保它	應用程式開發人員、雲端架構師

任務	Description	所需的技能
	們遵守適當的命名條件。否則，將不會建立您的索引。 • DISPATCH_MIN_BATCH _SIZE – 您要批次處理的日誌事件數目。不過，當 函數關閉時，無論此設定為何，日誌都會分派。	

新增記錄陳述式並測試您的函數

任務	Description	所需的技能

將記錄陳述式新增至函數。	使用其中一個內建記錄機制或您選擇的記錄模組，將記錄陳述式新增至函數。
	以下是在 Python 中記錄訊息的範例：
	`print("Your Log Message Here")`
	`logger = logging.getLogger(__name__)`
	`logger.info("Test Info Log.")`
	`logger.error("Test Error Log.")`
		應用程式開發人員、雲端架構師

測試您的函數	1. 在函數頁面上，選擇測試索引標籤。 2. 為您的函數建立測試事件並執行測試。如需詳細資訊，請參閱 Lambda 文件中的	應用程式開發人員、雲端架構師

任務	Description	所需的技能
	在主控台中測試 Lambda 函數。 您應該會看到執行函數：如果一切正常運作，則表示成功。	

在 OpenSearch 中檢視您的日誌

任務	Description	所需的技能
查詢您的索引。	在 OpenSearch 中，執行下列命令來查詢索引： `SELECT * FROM index-name` 您的日誌應該會顯示在查詢結果中。	雲端架構師

疑難排解

問題	解決方案
連線問題	• 確認您的 Lambda 函數具有存取 OpenSearch 叢集所需的網路連線能力。如需設定 VPC 設定的指引，請參閱 OpenSearch Service 文件。 • 確認授予 Lambda 函數的 IAM 許可允許它將資料寫入 OpenSearch 叢集。如需管理 IAM 許可的相關資訊，請參閱 Lambda 文件。
資料擷取錯誤	• 檢查您的 Lambda 函數的 CloudWatch Logs，以識別與 Lambda 遙測 API 整合相關

問題	解決方案
	的任何錯誤或例外狀況。如需疑難排解指引，請參閱 Lambda 遙測 API 文件。 • 確認 OpenSearch 叢集已正確設定，且具有擷取 Lambda 遙測資料所需的索引映射和設定。如需詳細資訊，請參閱 OpenSearch 文件。

相關資源

- OpenSearch 的 Lambda 遙測整合範例 (GitHub 儲存庫)
- 使用 Lambda 延伸模組增強 Lambda 函數 (Lambda 文件)
- Lambda 遙測 API (Lambda 文件)
- 介紹 AWS Lambda 遙測 API (AWS 部落格文章)
- 將 AWS Lambda 遙測 API 與 Prometheus 和 OpenSearch 整合 (AWS 部落格文章)

其他資訊

變更日誌結構

根據預設，延伸模組會將日誌作為巢狀文件傳送至 OpenSearch。這可讓您執行巢狀查詢來擷取個別資料欄值。

如果預設日誌輸出不符合您的特定需求，您可以透過修改由 提供的 Lambda 延伸模組的原始程式碼來自訂它 AWS。 AWS encourages 客戶可根據其業務需求調整輸出。若要變更日誌輸出，請在副檔名的原始程式碼中找到 `telemetry_dispatcher.py` 檔案中的 `dispatch_to_opensearch`函數，並進行必要的變更。

為以儲存格為基礎的架構設定無伺服器儲存格路由器

Mian Tariq 和 Ioannis Lioupras，Amazon Web Services

總結

做為全域儲存格型應用程式系統的進入點，儲存格路由器負責將使用者有效率地指派給適當的儲存格，並將端點提供給使用者。儲存格路由器會處理 函數，例如儲存user-to-cell的映射、監控儲存格容量，以及在需要時請求新的儲存格。在潛在中斷期間維護儲存格路由器功能非常重要。

此模式中的儲存格路由器設計架構著重於彈性、可擴展性和整體效能最佳化。模式使用靜態路由，其中用戶端會在初始登入時快取端點，並直接與儲存格通訊。此解耦透過協助確保在儲存格路由器受損期間，以儲存格為基礎的應用程式不中斷功能，來增強系統彈性。

此模式使用 AWS CloudFormation 範本來部署架構。如需範本部署內容的詳細資訊，或使用 部署相同的組態 AWS 管理主控台，請參閱其他資訊一節。

> **Important**
>
> 此模式中顯示的示範、程式碼和 CloudFormation 範本僅供說明之用。提供的資料僅用於說明設計模式和協助理解。示範和程式碼尚未可供生產使用，不應用於任何即時生產活動。任何在生產環境中使用程式碼或示範的嘗試，都強烈建議您自行承擔風險。我們建議諮詢適當的專業人員，並在生產設定中實作此模式或其任何元件之前執行徹底的測試。

先決條件和限制

先決條件

- 作用中的 Amazon Web Services (AWS) 帳戶
- 最新版本的 AWS Command Line Interface (AWS CLI)
- 具有建立 CloudFormation 堆疊、 AWS Lambda 函數和相關資源所需許可的 AWS 登入資料

產品版本

- Python 3.12

Architecture

下圖顯示儲存格路由器的高階設計。

圖表會逐步完成下列工作流程：

1. 使用者聯絡 Amazon API Gateway，做為儲存格路由器 API 端點的前端。
2. Amazon Cognito 會處理身分驗證和授權。
3. AWS Step Functions 工作流程包含下列元件：
   - Orchestrator – `Orchestrator`使用 來 AWS Step Functions 建立工作流程或狀態機器。工作流程是由儲存格路由器 API 觸發。會根據資源路徑`Orchestrator`執行 Lambda 函數。
   - Dispatcher `Dispatcher`– Lambda 函數會識別並為每個註冊的新使用者指派一個靜態儲存格。函數會搜尋使用者數目最少的儲存格，將其指派給使用者，然後傳回端點。
   - Mapper – `Mapper`操作會在 CloudFormation 範本建立的 `RoutingDB` Amazon DynamoDB 資料庫中處理user-to-cell的映射。觸發時， `Mapper`函數會提供已指派的使用者其端點。
   - Scaler – `Scaler`函數會追蹤儲存格佔用和可用容量。如有需要，`Scaler`函數可以透過 Amazon Simple Queue Service (Amazon SQS) 將請求傳送至佈建和部署層，以請求新的儲存格。
   - 驗證器 – `Validator`函數會驗證儲存格端點並偵測任何潛在問題。
4. `RoutingDB` 存放儲存格資訊和屬性 (API 端點、 AWS 區域狀態、指標）。
5. 當儲存格的可用容量超過閾值時，儲存格路由器會透過 Amazon SQS 請求佈建和部署服務，以建立新的儲存格。

建立新儲存格時， `RoutingDB`會從佈建和部署層更新。不過，該程序超出此模式的範圍。如需儲存格型架構設計內部部署的概觀，以及此模式中所用儲存格路由器設計的詳細資訊，請參閱其他資訊一節。

工具

AWS 服務

- Amazon API Gateway 可協助您建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。
- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。
- Amazon Cognito 為 Web 和行動應用程式提供身分驗證、授權和使用者管理。

- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Amazon Simple Queue Service (Amazon SQS) 提供安全、耐用且可用的託管佇列，可協助您整合和分離分散式軟體系統和元件。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 Lambda 函數和其他 AWS 服務來建置業務關鍵應用程式。

其他工具

- Python 是一種一般用途的電腦程式設計語言。

程式碼儲存庫

此模式的程式碼可在 GitHub Serverless-Cell-Router 儲存庫中使用。

最佳實務

如需建置以儲存格為基礎的架構時的最佳實務，請參閱下列 AWS Well-Architected 指引：

- 使用以儲存格為基礎的架構減少影響範圍
- AWS Well-Architected Framework 可靠性支柱：REL10-BP04 使用大量架構來限制影響範圍

史詩

準備來源檔案

任務	Description	所需的技能
複製範例程式碼儲存庫。	若要將 Serverless-Cell-Router GitHub 儲存庫複製到您的電腦，請使用下列命令： `git clone https://github.com/aws-samp`	開發人員

任務	Description	所需的技能
	`les/Serverless-Cell-Router/`	
設定 AWS CLI 臨時登入資料。	AWS CLI 使用 的登入資料來設定 AWS 帳戶。本演練使用 IAM Identity Center Command AWS 行或程式設計存取選項提供的臨時憑證。 這會設定具有適當登入資料的 `AWS_SECRET_ACCESS_KEY` 、 `AWS_ACCESS_KEY_ID` 和 `AWS_SESSION_TOKEN` AWS 環境變數，以便與 搭配使用 AWS CLI。	開發人員
建立 S3 儲存貯體。	建立 S3 儲存貯體，用於存放和存取 Serverless-Cell-Router Lambda 函數，以供 CloudFormation 範本部署。若要建立 S3 儲存貯體，請使用下列命令： `aws s3api create-bucket --bucket <bucket name> --region eu-central-1 --create-bucket-configuration LocationConstraint=eu-central-1`	開發人員

任務	Description	所需的技能

建立 .zip 檔案。	為位於 Functions 目錄中的每個 Lambda 函數建立一個 .zip 檔案。這些 .zip 檔案將用於部署 Lambda 函數。在 Mac 上，使用下列zip命令：
	`zip -j mapper-scr.zip Functions/Mapper.py`
	`zip -j dispatcher-scr.zip Functions/Dispatcher.py`
	`zip -j scaler-scr.zip Functions/Scaler.py`
	`zip -j cp validator-scr.zip Functions/Validator.py`
	`zip -j dynamodbDummyData-scr.zip Functions/DynamodbDummyData.py`
		開發人員

任務	Description	所需的技能

將 .zip 檔案複製到 S3 儲存貯體。	若要將所有 Lambda 函數 .zip 檔案複製到 S3 儲存貯體，請使用下列命令：
	`aws s3 cp mapper-scr.zip s3://<bucket name>`
	`aws s3 cp dispatcher-scr.zip s3://<bucket name>`
	`aws s3 cp scaler-scr.zip s3://<bucket name>`
	`aws s3 cp validator-scr.zip s3://<bucket name>`
	`aws s3 cp dynamodbDummyData-scr.zip s3://<bucket name>`
		開發人員

建立 CloudFormation 堆疊

任務	Description	所需的技能

部署 CloudFormation 範本。	若要部署 CloudFormation 範本，請執行下列 AWS CLI 命令：
	`aws cloudformation create-stack --stack-name serverless.cell-router \`
	`--template-body file://Serverless-Cell-Router-Stack-v10.yaml \`
	`--capabilities CAPABILITY_IAM \`
		開發人員

任務	Description	所需的技能
	`--parameters ParameterKey=LambdaFunctionMapperS3KeyParameterSCR,ParameterValue=mapper-scr.zip \ ParameterKey=LambdaFunctionDispatcherS3KeyParameterSCR,ParameterValue=dispatcher-scr.zip \ ParameterKey=LambdaFunctionScalerS3KeyParameterSCR,ParameterValue=scaler-scr.zip \ ParameterKey=LambdaFunctionAddDynamoDBDummyItemsS3KeyParameterSCR,ParameterValue=dynamodbDummyData-scr.zip \ ParameterKey=LambdaFunctionsS3BucketParameterSCR,ParameterValue=<S3 bucket storing lambda zip files> \ ParameterKey=CognitoDomain,ParameterValue=<Cognito Domain Name> \ --region <enter your aws region id, e.g. "eu-central-1">`	

任務	Description	所需的技能
檢查進度。	登入 AWS 管理主控台，開啟位於 https：//https://console.aws.amazon.com/cloudformation/ 的 CloudFormation 主控台，並檢查堆疊開發的進度。當狀態為 時CREATE_COMPLETE ，堆疊已成功部署。	開發人員

評估和驗證

任務	Description	所需的技能
將儲存格指派給使用者。	若要啟動 Orchestrator ，請執行下列 curl 命令： `curl -X POST \ -H "Authorization: Bearer {User id_token}" \ https://xxxxxx.execute-api.eu-central-1.amazonaws.com/Cell_Router_Development/cells` 會Orchestrator 觸發 Dispatcher 函數的執行。Dispatcher 然後，會驗證使用者的存在。如果找到使用者，會Dispatcher 傳回相關聯的儲存格 ID 和端點 URLs。如果找不到使用者，會將儲存格Dispatcher 配置給使用者，並將儲存格 ID 傳送	開發人員

任務	Description	所需的技能
	至 Scaler函數，以評估指派的儲存格剩餘容量。 Scaler 函數的回應如下： "cellID : cell-0002 , endPoint_1 : https://xxxxx.execute-api.eu-north-1 .amazonaws.com/ , endPoint_2 : https://xxxxxxx.execute-api .eu-central-1.amazonaws.com/"	

任務	Description	所需的技能

擷取使用者儲存格。	若要使用 Orchestrator 執行 Mapper函數，請執行下列命令：
	
	`curl -X POST \ -H "Authorization: Bearer {User id_token}" \ https://xxxxxxxx x.execute-api.eu-c entral-1.amazonaws .com/Cell_Router_D evelopment/mapper`
	
	Orchestrator 會搜尋指派給使用者的儲存格，並在下列回應中傳回儲存格 ID URLs：
	
	`"cellID : cell-0002 , endPoint_1 : https:// xxxxx.execute-api.eu-north-1 .amazonaws.com/ , endPoint_2 : https:// xxxxxxx.execute-api .eu-central-1.amaz onaws.com/"`
		開發人員

清除

任務	Description	所需的技能
清除資源。	若要避免在您的帳戶中產生額外費用，請執行下列動作：	應用程式開發人員

任務	Description	所需的技能
	1. 清空您為 Lambda 函數建立的 S3 儲存貯體。 2. 刪除儲存貯體。 3. 刪除 CloudFormation 堆疊。	

相關資源

參考

- 使用可用區域實現靜態穩定性
- AWS 故障隔離界限：靜態穩定性

影片

Physalia：在 Amazon EBS 上提供更高可用性的儲存格型架構

https://www.youtube-nocookie.com/embed/6IknqRZMFic?controls=0

其他資訊

以儲存格為基礎的架構設計內部部署

雖然此模式著重於儲存格路由器，但請務必了解整個環境。環境分為三個離散圖層：

- 路由層，或包含儲存格路由器的精簡層
- 儲存格層，包含各種儲存格
- 佈建和部署層，可佈建儲存格和部署應用程式

即使影響其他 layer 的受損，每個 layer 也會維持功能。 AWS 帳戶 做為故障隔離界限。

下圖顯示高層級的圖層。儲存格層和佈建和部署層超出此模式的範圍。

如需儲存格型架構的詳細資訊，請參閱使用儲存格型架構降低影響範圍：儲存格路由。

儲存格路由器設計模式

儲存格路由器是跨儲存格的共用元件。為了減輕潛在的影響，路由層必須使用盡可能精簡且水平可擴展的設計。作為系統的進入點，路由層僅包含有效將使用者指派給適當儲存格所需的元件。此層中的元件不會參與儲存格的管理或建立。

此模式使用靜態路由，這表示用戶端會在初次登入時快取端點，然後與儲存格建立直接通訊。用戶端與儲存格路由器之間的定期互動會啟動，以確認目前狀態或擷取任何更新。此刻意解耦可在儲存格路由器停機時為現有使用者啟用不間斷的操作，並在系統內提供持續的功能和彈性。

在此模式中，儲存格路由器支援下列功能：

- 從佈建和部署層中的儲存格資料庫擷取儲存格資料，以及儲存或更新本機資料庫。
- 使用儲存格指派演算法，將儲存格指派給應用程式的每個新註冊使用者。
- 將user-to-cells映射存放在本機資料庫中。
- 在使用者指派期間檢查儲存格的容量，並將自動販賣機的事件引發至佈建和部署層以建立儲存格。
- 使用儲存格建立條件演算法來提供此功能。
- 透過提供靜態儲存格URLs 來回應新註冊的使用者請求。這些 URLs會以存留時間 (TTL) 快取在用戶端上。
- 提供新的或更新的 URL，以回應無效 URL 的現有使用者請求。

若要進一步了解 CloudFormation 範本設定的示範儲存格路由器，請檢閱下列元件和步驟：

1. 設定 Amazon Cognito 使用者集區。
2. 設定和設定儲存格路由器的 API Gateway API。
3. 建立 DynamoDB 資料表。
4. 建立和設定 SQS 佇列。
5. 實作 `Orchestrator`。
6. 實作 Lambda 函數：`Dispatcher`、`Scaler`、`Mapper`、`Validator`。
7. 評估和驗證。

預先假設是佈建和部署層已建立。其實作詳細資訊超出此成品的範圍。

由於這些元件是由 CloudFormation 範本設定，因此下列步驟會以描述性和高層級顯示。假設您具備完成設定和組態所需的 AWS 技能。

1。設定 Amazon Cognito 使用者集區

登入 AWS 管理主控台，並在 https://console.aws.amazon.com/cognito/：// 開啟 Amazon Cognito 主控台。使用應用程式整合CellRouterPool、託管 UI 和必要的許可，設定名為 的 Amazon Cognito 使用者集區。

2. 設定和設定儲存格路由器的 API Gateway API

在以下網址開啟 API Gateway 主控台：https://console.aws.amazon.com/apigateway/。使用與 Amazon Cognito 使用者集區 整合的 Amazon Cognito 授權方CellRouter，設定名為 的APICellRouterPool。實作下列元素：

- CellRouter API 資源，包括POST方法
- 與步驟 5 中實作的 Step Functions 工作流程整合
- 透過 Amazon Cognito 授權方的授權
- 整合請求和回應映射
- 配置必要的許可

3. 建立 DynamoDB 資料表

在 https://console.aws.amazon.com/dynamodb/：// 開啟 DynamoDB 主控台，並使用tbl_router下列組態建立名為 的標準 DynamoDB 資料表：

- 分割區索引鍵 marketId–
- 排序索引鍵 cellId–
- 容量模式 – 佈建
- Point-in-time(PITR) – 關閉

在索引索引標籤上，建立名為 的全域次要索引marketId-currentCapacity-index。Scaler Lambda 函數將使用 索引，對指派使用者數目最低的儲存格進行有效的搜尋。

使用下列屬性建立資料表結構：

- marketId – 歐洲
- cellId – cell-0002
- currentCapacity – 2
- endPoint_1 – <第一個區域的端點>
- endPoint_2 – <第二個區域的端點>

- IsHealthy – True
- maxCapacity – 10
- regionCode_1 – eu-north-1
- regionCode_2 – eu-central-1
- userIds – <您的電子郵件地址>

4. 建立和設定 SQS 佇列

在 https://console.aws.amazon.com/sqs/：// 開啟 Amazon SQS 主控台，並建立名為 的標準 SQS 佇列，以 Amazon SQS 金鑰加密CellProvisioning設定。

5. 實作協調器

開發 Step Functions 工作流程，做為路由器Orchestrator的 。工作流程可透過儲存格路由器 API 呼叫。工作流程會根據資源路徑執行指定的 Lambda 函數。將步驟函數與儲存格路由器 的 API Gateway API 整合CellRouter，並設定呼叫 Lambda 函數所需的許可。

下圖顯示工作流程。選擇狀態會叫用其中一個 Lambda 函數。如果 Lambda 函數成功，工作流程會結束。如果 Lambda 函數失敗，則會呼叫失敗狀態。

6. 實作 Lambda 函數

實作 Dispatcher、Scaler、 Mapper和 Validator函數。當您在示範中設定每個函數時，請定義函數的角色，並指派在 DynamoDB 資料表 上執行必要操作的必要許可tbl_router。此外，將每個函數整合到工作流程 Orchestrator。

Dispatcher 函數

Dispatcher 函數負責識別和指派每個新註冊使用者的單一靜態儲存格。當新使用者向 全域應用程式註冊時，請求會移至 Dispatcher函數。函數會使用預先定義的評估條件來處理請求，如下所示：

1. 區域 – 選取使用者所在市場中的儲存格。例如，如果使用者從歐洲存取 全域應用程式，請選取 AWS 區域 在歐洲使用 的儲存格。
2. 鄰近性或延遲 – 選取最接近使用者的儲存格 例如，如果使用者從荷蘭存取應用程式，則函數會考慮使用法蘭克福和愛爾蘭的儲存格。有關哪個儲存格最接近的決定是根據指標，例如使用者位置與儲存格區域之間的延遲。在此範例模式中，資訊會從佈建和部署層靜態饋送。
3. 運作狀態 – Dispatcher函數會根據提供的儲存格狀態檢查選取的儲存格是否正常運作 （運作狀態 = true 或 false)。

4. 容量 – 使用者分佈是以儲存格邏輯中最少的使用者數量為基礎，因此會將使用者指派給擁有最少使用者數量的儲存格。

> **Note**
>
> 這些條件僅用於解釋此範例模式。對於實際的儲存格路由器實作，您可以定義更精簡和使用案例型條件。

會`Orchestrator`叫用 Dispatcher 函數，將使用者指派給儲存格。在此示範函數中，市場值是定義為的靜態參數`europe`。

`Dispatcher` 函數會評估儲存格是否已指派給使用者。如果已指派儲存格，則`Dispatcher`函數會傳回儲存格的端點。如果未將儲存格指派給使用者，則函數會搜尋使用者數目最少的儲存格，將其指派給使用者，並傳回端點。使用全域次要索引來最佳化儲存格搜尋查詢的效率。

Mapper 函數

`Mapper` 函數會監督資料庫中user-to-cell映射的儲存和維護。單一儲存格會配置給每個已註冊的使用者。每個儲存格有兩個不同的 URLs每個 AWS 區域各一個。做為 API Gateway 上託管的 API 端點，這些 URLs 可做為全域應用程式的傳入點。

當`Mapper`函數從用戶端應用程式接收請求時，它會在 DynamoDB 資料表上執行查詢`tbl_router`，以擷取與提供的電子郵件 ID 相關聯的user-to-cell映射。如果找到指派的儲存格，`Mapper`函數會立即提供儲存格的兩個 URLs。該`Mapper`函數也會主動監控儲存格 URLs 的變更，並啟動使用者設定的通知或更新。

Scaler 函數

`Scaler` 函數會管理儲存格的剩餘容量。對於每個新的使用者註冊請求，`Scaler`函數會評估`Dispatcher`函數指派給使用者的儲存格的可用容量。如果儲存格已根據指定的評估條件達到其預定限制，則函數會透過 Amazon SQS 佇列向佈建和部署層啟動請求，請求佈建和部署新儲存格。儲存格的擴展可以根據一組評估條件執行，如下所示：

1. 使用者上限 – 每個儲存格最多可有 500 個使用者。
2. 緩衝容量 – 每個儲存格的緩衝容量為 20%，這表示每個儲存格可以隨時指派給 400 個使用者。剩餘的 20% 緩衝區容量會保留給未來的使用案例和非預期案例的處理 （例如，當儲存格建立和佈建服務無法使用時）。
3. 儲存格建立 – 只要現有儲存格達到容量的 70%，就會觸發 請求來建立額外的儲存格。

> **Note**
>
> 這些條件僅用於解釋此範例模式。對於實際的儲存格路由器實作，您可以定義更精簡和使用案例型條件。

示範Scaler程式碼會在 Dispatcher成功將儲存格指派給新註冊的使用者Orchestrator後，由 執行。從 收到儲存格 ID 時Scaler， 會根據預先定義的評估條件，Dispatcher評估指定的儲存格是否有足夠的容量可容納其他使用者。如果儲存格容量不足，Scaler函數會將訊息分派給 Amazon SQS 服務。佈建和部署層內的服務會擷取此訊息，以啟動新儲存格的佈建。

驗證器函數

此Validator函數可識別並解決與儲存格存取相關的問題。當使用者登入全域應用程式時，應用程式會從使用者設定檔設定擷取儲存格的 URLs，並將使用者請求路由到儲存格中兩個指派區域的其中之一。如果無法存取 URLs，應用程式可以將驗證 URL 請求分派至儲存格路由器。cell-router會Orchestrator叫用 Validator。會Validator啟動驗證程序。除了其他檢查之外，驗證可能包括下列項目：

- 請求中的交互參考儲存格 URLs 與存放在資料庫中URLs，以識別和處理潛在的更新
- 執行深層運作狀態檢查 （ 例如，對儲存格端點的HTTP GET請求 ）

總之， Validator函數會回應用戶端應用程式請求，並提供驗證狀態以及任何必要的修補步驟。

Validator 旨在增強使用者體驗。假設某些使用者在存取全域應用程式時遇到困難，因為事件導致儲存格暫時無法使用。Validator 函數可以提供教學性的修復步驟，而不是呈現一般錯誤。這些步驟可能包括下列動作：

- 將事件通知使用者。
- 在服務可用性之前提供大約的等待時間。
- 提供用於取得其他資訊的支援聯絡電話。

Validator 函數的示範程式碼會驗證請求中使用者提供的儲存格 URLs 是否符合tbl_router資料表中存放的記錄。Validator 函數也會檢查儲存格是否正常運作。

透過 VPC 端點設定 Amazon S3 儲存貯體的私有存取權

Martin Maritsch、Nicoras Jacob Baer、Gabriel Rodriguez Garcia、Shukhrat Khodjaev、Mohan Gowda Purushothama 和 Joaquin Rinaudo、Amazon Web Services

總結

在 Amazon Simple Storage Service (Amazon S3) 中，預先簽章URLs 可讓您與目標使用者共用任意大小的檔案。根據預設，Amazon S3 預先簽章URLs 可在過期時段內從網際網路存取，這使得它們易於使用。不過，企業環境通常需要存取 Amazon S3 預先簽章URLs，才能僅限於私有網路。

此模式提供無伺服器解決方案，可透過使用來自私有網路且沒有網際網路周遊的預先簽章 URLs，安全地與 S3 物件互動。在 架構中，使用者可透過內部網域名稱存取 Application Load Balancer。流量會透過 Amazon API Gateway 和 S3 儲存貯體的虛擬私有雲端 (VPC) 端點在內部路由。 AWS Lambda 函數會透過私有 VPC 端點產生檔案下載的預先簽章 URLs，這有助於增強敏感資料的安全性和隱私權。

先決條件和限制

先決條件

- 包含部署在 中子網路的 VPC AWS 帳戶 ，該子網路連接到公司網路 （ 例如，透過 AWS Direct Connect)。

限制

- S3 儲存貯體的名稱必須與網域相同，因此建議您檢查 Amazon S3 儲存貯體命名規則。
- 此範例架構不包含已部署基礎設施的監控功能。如果您的使用案例需要監控，請考慮新增AWS 監控服務。
- 此範例架構不包含輸入驗證。如果您的使用案例需要輸入驗證並提高安全性，請考慮使用 AWS WAF 來保護您的 API。
- 此範例架構不包含 Application Load Balancer 的存取記錄。如果您的使用案例需要存取記錄，請考慮啟用負載平衡器存取日誌。

版本

- Python 3.11 版或更新版本
- Terraform 1.6 版或更新版本

Architecture

目標技術堆疊

目標技術堆疊中會使用下列 AWS 服務：

- Amazon S3 是核心儲存服務，用於安全地上傳、下載和儲存檔案。
- Amazon API Gateway 公開與 S3 儲存貯體互動的資源和端點。此服務在產生預先簽章URLs 以下載或上傳資料時扮演重要角色。
- AWS Lambda 會產生預先簽章URLs，以便從 Amazon S3 下載檔案。API Gateway 呼叫 Lambda 函數。
- Amazon VPC 在 VPC 中部署資源以提供網路隔離。VPC 包含子網路和路由表，以控制流量流程。
- Application Load Balancer 會將傳入流量路由至 API Gateway 或 S3 儲存貯體的 VPC 端點。它允許來自公司網路的使用者在內部存取資源。
- Amazon S3 的 VPC 端點可在 VPC 和 Amazon S3 中的資源之間進行直接的私有通訊，而不會周遊公有網際網路。
- AWS Identity and Access Management (IAM) 控制對 AWS 資源的存取。設定許可以確保與 API 和其他 服務的安全互動。

目標架構

此圖展示了以下要點：

1. 企業網路的使用者可以透過內部網域名稱存取 Application Load Balancer。我們假設公司網路與 中的內部網路子網路之間存在連線 AWS 帳戶 （例如，透過 Direct Connect 連線）。
2. Application Load Balancer 會將傳入流量路由至 API Gateway，以產生預先簽章URLs，將資料下載或上傳至 Amazon S3 或 S3 儲存貯體的 VPC 端點。在這兩種情況下，請求都會在內部路由，不需要周遊網際網路。
3. API Gateway 會公開要與 S3 儲存貯體互動的資源和端點。在此範例中，我們提供端點，以從 S3 儲存貯體下載檔案，但也可以進行擴充以提供上傳功能。
4. Lambda 函數會產生預先簽章的 URL，Amazon S3以使用 Application Load Balancer 的網域名稱而非公有 Amazon S3 網域，從 Amazon S3 下載檔案。
5. 使用者會收到預先簽章的 URL，並使用 Application Load Balancer 從 Amazon S3 下載檔案。負載平衡器包含預設路由，可將不適用於 API 的流量傳送至 S3 儲存貯體的 VPC 端點。

6. VPC 端點會將具有自訂網域名稱的預先簽章 URL 路由至 S3 儲存貯體。S3 儲存貯體必須具有與網域相同的名稱。

自動化和擴展

此模式使用 Terraform 將基礎設施從程式碼儲存庫部署到 AWS 帳戶。

工具

工具

- Python 是一種一般用途的電腦程式設計語言。
- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。

程式碼儲存庫

此模式的程式碼可在 GitHub 儲存庫中取得，網址為 https：//https://github.com/aws-samples/private-s3-vpce。

最佳實務

此模式的範例架構使用 IAM 許可來控制對 API 的存取。擁有有效 IAM 登入資料的任何人都可以呼叫 API。如果您的使用案例需要更複雜的授權模型，建議您使用不同的存取控制機制。

史詩

在 中部署解決方案 AWS 帳戶

任務	Description	所需的技能
取得 AWS 登入資料。	檢閱您的 AWS 登入資料和對帳戶的存取。如需說明，請參閱 AWS CLI 文件中的組態和登入資料檔案設定。	AWS DevOps、一般 AWS

任務	Description	所需的技能
複製儲存庫。	複製此模式隨附的 GitHub 儲存庫： `git clone https://github.com/aws-samples/private-s3-vpce`	AWS DevOps、一般 AWS
設定變數。	1. 在電腦上的 GitHub 儲存庫中，開啟 `terraform` 資料夾： `cd terraform` 2. 開啟 `example.tfvars` 檔案，並根據您的需求自訂參數。	AWS DevOps、一般 AWS
部署解決方案。	1. 在 `terraform` 資料夾中，執行 Terraform 並傳入您自訂的變數： `terraform apply -var-file="example.tfvars"` 2. 確認架構圖中顯示的資源已成功部署。	AWS DevOps、一般 AWS

測試解決方案

任務	Description	所需的技能
建立測試檔案。	將檔案上傳至 Amazon S3，以建立檔案下載的測試案例。您可以使用 Amazon S3 主控台或下列 AWS CLI 命令：	AWS DevOps、一般 AWS

任務	Description	所需的技能
	`aws s3 cp /path/to/testfile s3://your-bucket-name/testfile`	
測試預先簽章的 URL 功能。	1. 使用 awscurl 將請求傳送至 Application Load Balancer，以建立測試檔案的預先簽章 URL： `awscurl https://your-domain-name/api/get_url?key=testfile` 此步驟會從登入資料建立有效的簽章，並由 API Gateway 驗證。 2. 從您從上一個步驟收到的回應中剖析連結，然後開啟預先簽章的 URL 以下載檔案。	AWS DevOps、一般 AWS
清除。	請務必在不再需要資源時將其移除： `terraform destroy`	AWS DevOps、一般 AWS

疑難排解

問題	解決方案
S3 物件金鑰名稱具有特殊字元，例如數字符號 (#) 會中斷 URL 參數並導致錯誤。	正確編碼 URL 參數，並確保 S3 物件金鑰名稱遵循 Amazon S3 準則。

相關資源

Amazon S3：

- 使用預先簽章URLs 共用物件
- 使用儲存貯體政策控制 VPC 端點的存取

Amazon API Gateway：

- 針對 APIs中的私有 API 使用 VPC 端點政策

Application Load Balancer：

- 使用 ALB、S3 和 PrivateLink 託管內部 HTTPS 靜態網站 (AWS 部落格文章)

AWS Step Functions 使用 Amazon Bedrock 對 中的狀態進行故障診斷

Aniket Kurzadkar 和 Sangam Kushwaha，Amazon Web Services

摘要

AWS Step Functions 錯誤處理功能可協助您看到工作流程中狀態期間發生的錯誤，但仍可能難以找出錯誤的根本原因並進行偵錯。此模式可解決挑戰，並顯示 Amazon Bedrock 如何協助您解決 Step Functions 中狀態期間發生的錯誤。

Step Functions 提供工作流程協同運作，讓開發人員更輕鬆地自動化程序。Step Functions 也提供錯誤處理功能，提供下列優點：

- 開發人員可以建立更具彈性的應用程式，在發生錯誤時不會完全失敗。
- 工作流程可以包含條件式邏輯，以不同方式處理不同類型的錯誤。
- 系統可以自動重試失敗的操作，可能具有指數退避。
- 您可以針對錯誤案例定義替代執行路徑，讓工作流程調整並繼續處理。

當 Step Functions 工作流程發生錯誤時，此模式會顯示錯誤訊息和內容如何傳送至基礎模型 (FM)，例如 Step Functions 支援的 Claude 3。FM 可以分析錯誤、分類錯誤，並建議潛在的修補步驟。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 對 AWS Step Functions 和 工作流程的基本了解
- Amazon Bedrock API 連線

限制

- 您可以針對各種 使用此模式的方法 AWS 服務。不過，結果可能會根據後續由 Amazon Bedrock 評估而建立 AWS Lambda 的提示而有所不同。
- 有些 AWS 服務 無法全部使用 AWS 區域。如需區域可用性，請參閱依區域的 AWS 服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

下圖顯示此模式的工作流程和架構元件。

圖表顯示 Step Functions 狀態機器中錯誤處理和通知的自動化工作流程：

1. 開發人員會啟動狀態機器的執行。
2. Step Functions 狀態機器會開始處理其狀態。有兩種可能的結果：
   - (a) 如果所有狀態都成功執行，工作流程會直接前往 Amazon SNS 傳送電子郵件成功通知。
   - (b) 如果任何狀態失敗，工作流程會移至錯誤處理 Lambda 函數。
3. 如果發生錯誤，會發生下列情況：
   - (a) 觸發 Lambda 函數（錯誤處理常式）。Lambda 函數會從 Step Functions 狀態機器傳遞的事件資料中擷取錯誤訊息。然後，Lambda 函數會根據此錯誤訊息準備提示，並將提示傳送至 Amazon Bedrock。提示請求與遇到的特定錯誤相關的解決方案和建議。
   - (b) 託管生成式 AI 模型的 Amazon Bedrock 會處理輸入提示。（此模式使用 Anthropic Claude 3 基礎模型 (FM)，這是 Amazon Bedrock 支援的許多 FMs之一。) AI 模型會分析錯誤內容。然後，模型會產生回應，其中包含錯誤發生原因的說明、解決錯誤的潛在解決方案，以及避免未來發生相同錯誤的建議。

     Amazon Bedrock 會將其 AI 產生的回應傳回 Lambda 函數。Lambda 函數會處理回應，可能將其格式化或擷取金鑰資訊。然後，Lambda 函數會將回應傳送至狀態機器輸出。
4. 在錯誤處理或成功執行後，工作流程會透過觸發 Amazon SNS 傳送電子郵件通知來結束。

工具

AWS 服務

- Amazon Bedrock 是一項全受管服務，可讓您透過統一 API 使用來自領導 AI 新創公司的高效能基礎模型 (FMs) 和 Amazon。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務 來建置業務關鍵型應用程式。

最佳實務

- 由於 Amazon Bedrock 是從訓練資料中學習的生成式 AI 模型，它也會使用該資料來訓練和產生內容。最佳實務是隱藏任何可能導致資料外洩問題的私有資訊。
- 雖然生成式 AI 可以提供寶貴的洞見，但重要的錯誤處理決策仍應涉及人為監督，尤其是在生產環境中。

史詩

為您的工作流程建立狀態機器

任務	Description	所需的技能
建立狀態機器。	若要建立適合您工作流程的狀態機器，請執行下列動作： 1. 登入 AWS 管理主控台，然後開啟 AWS Step Functions 主控台。 2. 從左側導覽窗格中，選擇狀態機器。 3. 選擇 Create state machine (建立狀態機器)。 4. 根據您的使用案例選擇範本，或根據您的需求選擇空白來建立範本。	AWS DevOps

建立 Lambda 函式

任務	Description	所需的技能
建立 Lambda 函式。	若要建立 Lambda 函數，請執行下列動作：	AWS DevOps

任務	Description	所需的技能
	1. 在 中 AWS 管理主控台，導覽至 AWS Lambda 主控台。 2. 在導覽面板上，選擇函式，然後選擇建立函式。 3. 在建立函數頁面上，從選項中選擇以建立函數。然後，在函數名稱中輸入名稱，然後從執行時間的下拉式清單中選擇適當的語言。 4. 選擇建立函數。	

任務	Description	所需的技能

在 Lambda 程式碼中設定所需的邏輯。	• 若要使用 連線至 Amazon Bedrock API適用於 Python (Boto3) 的 AWS SDK，請使用下列程式碼。
	
	此程式碼會為 Amazon Bedrock 設定用戶端、準備必要的參數，然後使用指定的提示將請求傳送至 Claude 3 模型。
	
	此模式會叫用 Claude 3 模型。如需所有支援的基礎模型，包括相關模型 IDs的詳細資訊，請參閱 Amazon Bedrock 文件中的 Amazon Bedrock 中支援的基礎模型。
	
	`client = boto3.client(`
	`service_name="bedrock-runtime", region_name="selected-region"`
	`)`
	
	`# Invoke Claude 3 with the text prompt`
	`model_id = "your-model-id" # Select your Model ID, Based on the Model Id, Change the body format`
	
	`try:`
	`response = client.invoke_model(`
		AWS DevOps

任務	Description	所需的技能

	<pre>modelId=m
	odel_id,
	body=json
	.dumps(
	{
	"anthropic_version":
	"bedrock-2023-05-31",
	"max_tokens": 1024,
	"messages": [
	{
	"role": "user",
	"content"
	: [{"type": "text",
	"text": prompt}],
	}
	],
	}
	),
	)</pre>
	• （選用）將 AWS 帳戶 IDs 取代為預留位置帳戶 IDs。基於安全考量，此方法可用於清理日誌、錯誤訊息或其他可能包含敏感帳戶資訊的輸出。
	
	下列程式碼會找到以冒號括住的 12 位數號碼（即 Amazon Resource Name (ARNs) 中的 AWS 帳戶 IDs 格式和一些其他識別	

任務	Description	所需的技能

	AWS 符），並以預留位置帳戶 ID 取代":123456789012:" 。
	
	`def replace_account_id(input_string):`
	`  # Use a regular expression to find the AWS account ID pattern`
	`  account_id_pattern = r'(:\d{12}:)'`
	
	`  # Replace the matched pattern with ":123456789012:"`
	`  modified_string = re.sub(account_id_pattern, ":123456789012:", input_string)`
	
	`  return modified_string`	

將 Step Functions 與 Lambda 整合

任務	Description	所需的技能
設定 Lambda 來處理 Step Functions 中的錯誤。	若要設定 Step Functions 來處理錯誤而不中斷工作流程，請執行下列動作：	AWS DevOps

任務	Description	所需的技能
	1. 在 Step Functions 主控台中，導覽至您先前建立的狀態機器。 2. 選擇編輯，然後選擇您要設定錯誤處理的服務，然後選擇錯誤處理。 3. 選擇新增擷取器，然後針對備用狀態，選擇 Lambda，然後選擇您先前建立的 Lambda 函數。如需詳細資訊，請參閱 Step Functions 文件中的擷取錯誤。	

疑難排解

問題	解決方案
Lambda 無法存取 Amazon Bedrock API（未授權執行）	當 Lambda 角色沒有存取 Amazon Bedrock API 的許可時，會發生此錯誤。若要解決此問題，請新增 Lambda 角色`AmazonBedrockFullAccess` 的政策。如需詳細資訊，請參閱《 AWS 受管政策參考指南》中的 AmazonBedrockFullAccess。
Lambda 逾時錯誤	有時可能需要超過 30 秒才能產生回應並將其傳回，視提示而定。若要解決此問題，請增加組態時間。如需詳細資訊，請參閱《 AWS Lambda 開發人員指南》中的設定 Lambda 函數逾時。

相關資源

- Amazon Bedrock

- Amazon Bedrock API 存取
- 建立您的第一個 Lambda 函數
- 使用 Step Functions 開發工作流程
- AWS Step Functions

更多模式

- 使用 Athena 存取、查詢和聯結 Amazon DynamoDB 資料表
- 使用 GitHub 動作自動化 AWS CDK Python 應用程式的 Amazon CodeGuru 檢閱
- 自動化 AWS 資源評估
- 使用 AWS SAM 自動化巢狀應用程式的部署
- 使用 GitHub Actions、Artifactory 和 Terraform 在多儲存庫設定中自動化 AWS Supply Chain 資料湖的部署
- 自動化跨 的 Amazon RDS 執行個體複寫 AWS 帳戶
- 使用 DynamoDB TTL 自動將項目封存至 Amazon S3
- 在 CodeCommit 中自動偵測變更並啟動單一儲存庫的不同 CodePipeline 管道 CodeCommit
- 在 Amazon OpenSearch Service 中建置多租戶無伺服器架構
- 在 AWS 雲端中建置進階大型主機檔案檢視器
- 使用 AWS 服務計算風險值 (VaR)
- 將 AWS Service Catalog 產品複製到不同的 AWS 帳戶和 AWS 區域
- 自動為 Java 和 Python 專案建立動態 CI 管道
- 使用 CQRS 和事件來源將整體分解為微服務
- 將以 React 為基礎的單一頁面應用程式部署至 Amazon S3 和 CloudFront
- 使用私有端點和 Application Load Balancer 在內部網站上部署 Amazon API Gateway API
- 使用基礎設施做為程式碼，在 AWS 雲端上部署和管理無伺服器資料湖
- AWS 使用 Terraform 和 Amazon Bedrock 在 上部署 RAG 使用案例
- 使用 Amazon Bedrock 代理程式和知識庫開發全自動聊天式助理
- 使用 RAG 和 ReAct 提示，開發進階生成式 AI 聊天式助理
- 使用 Step Functions 透過 IAM Access Analyzer 動態產生 IAM 政策
- 使用 Amazon Cognito 和 IaC 自動化將 Amazon Quick Sight 視覺化元件內嵌至 Web 應用程式 IaC
- 確保在啟動時啟用對 Amazon S3 的 Amazon EMR 記錄
- 預估 DynamoDB 資料表的隨需容量成本
- 使用 Amazon Personalize 產生個人化和重新排名的建議
- 使用 AWS Glue 任務和 Python 產生測試資料
- 從 SQL Server 遷移至 PostgreSQL 時，實作 PII 資料的 SHA1 雜湊
- 使用 AWS Step Functions 實作無伺服器 saga 模式

- 使用 AWS CDK 跨多個 AWS 區域、帳戶和 OUs 啟用 Amazon DevOps Guru，以改善營運效能
- 使用 Step Functions 和 Lambda 代理函數跨 AWS 帳戶啟動 CodeBuild 專案
- 使用 AWS Glue 將 Apache Cassandra 工作負載遷移至 Amazon Keyspaces
- 跨多個 監控共用 Amazon Machine Image 的使用 AWS 帳戶
- 使用 AWS CDK 和 GitHub Actions 工作流程最佳化多帳戶無伺服器部署
- 使用 透過驗證、轉換和分割來協調 ETL 管道 AWS Step Functions
- 使用 Amazon Athena 查詢具有 SQL 的 Amazon DynamoDB 資料表
- 使用 AWS Fargate 大規模執行事件驅動和排程工作負載
- 將自訂屬性傳送至 Amazon Cognito，並將其插入字符
- 使用 Amazon CloudFront 在 Amazon S3 儲存貯體中透過 VPC 提供靜態內容
- 使用自動化工作流程簡化 Amazon Lex 機器人開發和部署
- 使用 AWS Lambda 在六邊形架構中建構 Python 專案
- 將自然語言轉換為查詢 DSL for OpenSearch 和 Elasticsearch 查詢
- 將資料從 Amazon Redshift 叢集跨帳戶卸載至 Amazon S3
- 使用 AWS Fargate WaitCondition 勾點建構來協調資源相依性和任務執行
- 使用 Amazon Bedrock 代理程式，透過文字型提示在 Amazon EKS 中自動建立存取項目控制項

聯網

主題

- 使用 AWS Transit Gateway 自動化區域間對等互連的設定
- 使用 AWS Transit Gateway 集中網路連線
- 使用 Application Load Balancer 在 Oracle WebLogic 上設定 Oracle JD Edwards EnterpriseOne 的 HTTPS 加密
- 透過私有網路連線至 Application Migration Service 資料和控制平面
- 使用 AWS CloudFormation 自訂資源和 Amazon SNS 建立 Infoblox 物件
- AWS 使用 Terraform 在 上建立階層式多區域 IPAM 架構
- 自訂 的 Amazon CloudWatch 提醒 AWS Network Firewall
- 使用 Terraform 在 AWS Wavelength 區域中部署資源
- 將大量 DNS 記錄遷移至 Amazon Route 53 私有託管區域
- 當您從 F5 遷移到 AWS 上的 Application Load Balancer 時修改 HTTP 標頭
- 為多個 中的傳入網際網路存取建立 Network Access Analyzer 調查結果報告 AWS 帳戶
- 在多帳戶 AWS 環境中設定混合網路的 DNS 解析
- 確認 ELB 負載平衡器需要終止 TLS
- 使用 Splunk 檢視 AWS Network Firewall 日誌和指標
- 更多模式

使用 AWS Transit Gateway 自動化區域間對等互連的設定

Ram Kandaswamy，Amazon Web Services

總結

AWS Transit Gateway 透過中央中樞連接虛擬私有雲端 (VPCs) 和內部部署網路。Transit Gateway 流量一律保留在全球 Amazon Web Services (AWS) 骨幹上，不會周遊公有網際網路，這可減少威脅向量，例如常見的入侵和分散式拒絕服務 (DDoS) 攻擊。

如果您需要在兩個或多個 AWS 區域之間通訊，您可以使用區域間傳輸閘道對等互連，在不同區域中的傳輸閘道之間建立對等互連。不過，使用 Transit Gateway 手動設定區域間對等互連可能是具有多個步驟的耗時程序。此模式提供自動化程序，透過使用程式碼來執行對等互連來移除這些手動步驟。如果您必須在多區域組織設定期間重複設定多個區域和 AWS 帳戶，則可以使用此方法。

此模式使用 AWS CloudFormation 堆疊，其中包含 Amazon CloudWatch Logs 中的 AWS Step Functions 工作流程、AWS Lambda 函數、AWS Identity and Access Management (IAM) 角色和日誌群組。然後，您可以啟動 Step Functions 執行，並為傳輸閘道建立區域間對等互連。若要手動設定區域間對等互連，請參閱使用 AWS Transit Gateway 在不同 AWS 區域中的對等 VPCs。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 現有的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。
- 在請求者區域和接受者區域中建立和設定的傳輸閘道。請求者區域是產生對等請求的地方，接受者區域接受對等請求。如需詳細資訊，請參閱 Amazon VPC 文件中的建立和接受 VPC 對等互連。
- 在接受者和請求者區域中安裝和設定的 VPCs。如需建立 VPC 的步驟，請參閱 Amazon VPC 文件中的從 Amazon VPC 入門建立 VPC。
- VPCs 必須使用 `addToTransitGateway`標籤和 `true` 值。
- VPCs 的安全群組和網路存取控制清單 (ACLs)，根據您的需求設定。如需詳細資訊，請參閱 Amazon VPC 文件中的 VPC 和網路 ACL 的安全群組。 ACLs

AWS 區域和限制

- 只有特定 AWS 區域支援區域間對等互連。如需支援區域間對等互連的區域完整清單，請參閱 AWS Transit Gateway FAQs。

- 在連接的範本程式碼中，請求者區域假設為 `us-east-2`，接受者區域假設為 `us-west-2`。如果您想要設定不同的區域，您必須在所有 Python 檔案中編輯這些值。若要實作涉及兩個以上區域的更複雜設定，您可以變更 Step Function，將區域做為參數傳遞給 Lambda 函數，並為每個組合執行函數。

Architecture

圖表顯示具有下列步驟的工作流程：

1. 使用者會建立 AWS CloudFormation 堆疊。
2. AWS CloudFormation 會建立使用 Lambda 函數的 Step Functions 狀態機器。如需詳細資訊，請參閱 AWS Step Functions Step Functions 文件中的建立使用 Lambda 的 Step Functions 狀態機器。
3. Step Functions 會呼叫 Lambda 函數進行對等互連。
4. Lambda 函數會在傳輸閘道之間建立對等連線。
5. Step Functions 會呼叫 Lambda 函數來修改路由表。
6. Lambda 函數透過新增 VPCs 的無類別網域間路由 (CIDR) 區塊來修改路由表。

Step Functions 工作流程

圖表顯示下列 Step Functions 工作流程：

1. Step Functions 工作流程會呼叫傳輸閘道對等的 Lambda 函數。
2. 有一個計時器呼叫要等待一分鐘。
3. 對等狀態會擷取並傳送至條件區塊。區塊負責循環。
4. 如果不符合成功條件，工作流程會編碼為進入計時器階段。
5. 如果符合成功條件，則會呼叫 Lambda 函數來修改路由表。在此呼叫之後，Step Functions 工作流程會結束。

工具

- AWS CloudFormation – AWS CloudFormation 是一項服務，可協助您建立 AWS 資源的模型和設定。

- Amazon CloudWatch Logs – CloudWatch Logs 可協助您集中所有系統、應用程式和您使用的 AWS 服務的日誌。
- AWS Identity and Access Management (IAM) – IAM 是一種 Web 服務，可安全地控制對 AWS 服務的存取。
- AWS Lambda – Lambda 會在高可用性運算基礎設施上執行您的程式碼，並執行運算資源的所有管理。
- AWS Step Functions – Step Functions 可讓您輕鬆地協調分散式應用程式的元件，做為視覺化工作流程中的一系列步驟。

史詩

自動化對等互連

任務	Description	所需的技能
將連接的檔案上傳至 S3 儲存貯體。	登入 AWS 管理主控台，開啟 Amazon S3 主控台，然後將 `modify-transit-gateway-routes.zip` 、`peer-transit-gateway.zip` 和 `get-transit-gateway-peering-status.zip` 檔案（已連接）上傳到您的 S3 儲存貯體。	一般 AWS
建立 AWS CloudFormation 堆疊。	執行下列命令，使用 `transit-gateway-peering.json` 檔案（已連接）建立 AWS CloudFormation 堆疊： `aws cloudformation create-stack --stack-name myteststack --template-body file://sampletemplate.json`	DevOps 工程師

任務	Description	所需的技能
	AWS CloudFormation 堆疊會建立 Step Functions 工作流程、Lambda 函數、IAM 角色和 CloudWatch 日誌群組。 請確定 AWS CloudFormation 範本參考的 S3 儲存貯體包含您先前上傳的檔案。 **Note** 您也可以使用 AWS CloudFormation 主控台建立堆疊。如需詳細資訊，請參閱 AWS CloudFormation 文件中的在 AWS CloudFormation 主控台上建立堆疊。 AWS CloudFormation	
在 Step Functions 中啟動新的執行。	開啟 Step Functions 主控台並啟動新的執行。Step Functions 會呼叫 Lambda 函數，並為傳輸閘道建立對等連線。您不需要輸入 JSON 檔案。確認附件可用，且連線類型為對等。 如需詳細資訊，請參閱 AWS Step Functions 文件中的從 AWS Step Functions 入門開始新的執行。	DevOps 工程師，一般 AWS

任務	Description	所需的技能
驗證路由表中的路由。	在傳輸閘道之間建立區域間對等互連。路由表會以對等區域 VPC 的 IPv4 CIDR 區塊範圍更新。 開啟 Amazon VPC 主控台，然後在對應至傳輸閘道連接的路由表中選擇關聯索引標籤。驗證對等區域的 VPC CIDR 區塊範圍。 如需詳細步驟和說明，請參閱 Amazon VPC 文件中的關聯傳輸閘道路由表。	網路管理員

相關資源

- 在 Step Functions 中的執行
- 傳輸閘道對等互連附件
- 使用 AWS Transit Gateway 在不同 AWS 區域中的對等 VPCs
- 使用 AWS Transit Gateway - 示範 （ 影片 ） 跨 AWS 區域互連 VPCs

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS Transit Gateway 集中網路連線

Mydhili Palagummi 和 Nikhil Marrapu，Amazon Web Services

總結

此模式描述最簡單的組態，其中 AWS Transit Gateway 可用來將內部部署網路連線到 AWS 區域內多個 AWS 帳戶中的虛擬私有雲端 (VPCs)。使用此設定，您可以建立混合網路，連接區域中的多個 VPC 網路和內部部署網路。這可透過使用傳輸閘道和虛擬私有網路 (VPN) 連線至內部部署網路來完成。

先決條件和限制

先決條件

- 託管網路服務的帳戶，作為 AWS Organizations 中組織的成員帳戶進行管理
- 多個 AWS 帳戶中VPCs，沒有重疊的無類別網域間路由 (CIDR) 區塊

限制

此模式不支援隔離特定 VPCs或內部部署網路之間的流量。連接到傳輸閘道的所有網路將能夠互相連接。若要隔離流量，您需要在傳輸閘道上使用自訂路由表。此模式只會使用單一預設傳輸閘道路由表來連接 VPCs 和內部部署網路，這是最簡單的組態。

Architecture

目標技術堆疊

- AWS Transit Gateway
- AWS Site-to-Site VPN
- VPC
- AWS Resource Access Manager (AWS RAM)

目標架構

工具

AWS 服務

- AWS Resource Access Manager (AWS RAM) 可協助您在 AWS 帳戶、組織單位或整個組織中安全地共用資源 AWS Organizations 。
- AWS Transit Gateway 是中央中樞，可連接虛擬私有雲端 (VPCs) 和內部部署網路。

史詩

在網路服務帳戶中建立傳輸閘道

任務	Description	所需的技能
建立傳輸閘道。	在您要託管網路服務的 AWS 帳戶中，在目標 AWS 區域中建立傳輸閘道。如需說明，請參閱建立傳輸閘道。注意下列事項： • 選取預設路由表關聯。 • 選取預設路由表傳播。	網路管理員

將傳輸閘道連接至您的內部部署網路

任務	Description	所需的技能
設定 VPN 連線的客戶閘道裝置。	客戶閘道裝置連接到傳輸閘道與內部部署網路之間Site-to-Site連線的內部部署端。如需詳細資訊，請參閱 AWS Site-to-Site VPN 文件中的您的客戶閘道裝置。識別或啟動支援的現場部署客戶裝置，並記下其公有 IP 地址。VPN 組態稍後在此史詩中完成。	網路管理員
在網路服務帳戶中，建立傳輸閘道的 VPN 連接。	若要設定連線，請為傳輸閘道建立 VPN 連接。如需說明，請參閱傳輸閘道 VPN 連接。	網路管理員

任務	Description	所需的技能
在內部部署網路的客戶閘道裝置上設定 VPN。	下載與傳輸閘道相關聯的Site-to-Site連線組態檔案，並在客戶閘道裝置上設定 VPN 設定。如需說明，請參閱下載組態檔案。	網路管理員

將網路服務帳戶中的傳輸閘道分享給其他 AWS 帳戶或您的組織

任務	Description	所需的技能
在 AWS Organizations 管理帳戶中，開啟共用。	若要與您的組織或特定組織單位共用傳輸閘道，請在 AWS Organizations 中開啟共用。否則，您需要個別共用每個帳戶的傳輸閘道。如需說明，請參閱在 AWS Organizations 中啟用資源共用。	AWS 系統管理員
在網路服務帳戶中建立傳輸閘道資源共享。	若要允許組織中其他 AWS 帳戶中VPCs 連線至傳輸閘道，請在網路服務帳戶中使用 AWS RAM 主控台來共用傳輸閘道資源。如需說明，請參閱建立資源共享。	AWS 系統管理員

將 VPCs連接至傳輸閘道

任務	Description	所需的技能
在個別帳戶中建立 VPC 連接。	在已共用傳輸閘道的帳戶中，建立傳輸閘道 VPC 連接。如需說明，請參閱建立傳輸閘道連接至 VPC。	網路管理員

任務	Description	所需的技能
接受 VPC 連接請求。	在網路服務帳戶中，接受傳輸閘道 VPC 連接請求。如需說明，請參閱接受共用附件。	網路管理員

設定路由

任務	Description	所需的技能
在個別帳戶 VPCs中設定路由。	在每個個別帳戶 VPC 中，使用傳輸閘道做為目標，將路由新增至內部部署網路和其他 VPC 網路。如需說明，請參閱從路由表新增和移除路由。	網路管理員
在傳輸閘道路由表中設定路由。	來自 VPCs和 VPN 連接的路由應傳播，並應顯示在傳輸閘道預設路由表中。如有需要，請在傳輸閘道預設路由表中建立任何靜態路由（其中一個範例是靜態 VPN 連接的靜態路由）。如需說明，請參閱建立靜態路由。	網路管理員
新增安全群組和網路存取控制清單 (ACL) 規則。	對於 VPC 中的 EC2 執行個體和其他資源，請確保安全群組規則和網路 ACL 規則允許 VPCs 與內部部署網路之間的流量。如需說明，請參閱使用安全群組控制資源的流量，以及從 ACL 新增和刪除規則。	網路管理員

測試連線能力

任務	Description	所需的技能
測試 VPCs之間的連線。	確保網路 ACL 和安全群組允許網際網路控制訊息通訊協定 (ICMP) 流量，然後從 VPC 中的執行個體 ping 到另一個也連接到傳輸閘道的 VPC。	網路管理員
測試 VPCs與內部部署網路之間的連線。	確保網路 ACL 規則、安全群組規則和任何防火牆允許 ICMP 流量，然後 ping 內部部署網路和 VPCs 中的 EC2 執行個體之間。您必須先從內部部署網路啟動網路通訊，才能讓 VPN 連線進入 UP 狀態。	網路管理員

相關資源

- 建置可擴展且安全的多 VPC AWS 網路基礎設施 (AWS 白皮書）
- 使用共用資源 (AWS RAM 文件）
- 使用傳輸閘道 (AWS Transit Gateway 文件）

使用 Application Load Balancer 在 Oracle WebLogic 上設定 Oracle JD Edwards EnterpriseOne 的 HTTPS 加密

Thanigaivel Thirumalai，Amazon Web Services

總結

此模式說明如何為 Oracle WebLogic 工作負載上的 Oracle JD Edwards EnterpriseOne 中的 SSL 卸載設定 HTTPS 加密。此方法會加密使用者瀏覽器和負載平衡器之間的流量，以消除 EnterpriseOne 伺服器的加密負擔。

許多使用者使用 AWS Application Load Balancer 水平擴展 EnterpriseOne JAVA 虛擬機器 (JVM) 層。Application Load Balancer 負載平衡器可做為用戶端的單一聯絡點，並將傳入流量分配到多個 JVMs。或者，負載平衡器可以將流量分散到多個可用區域，並提高 EnterpriseOne 的可用性。

此模式中描述的程序會設定瀏覽器和負載平衡器之間的加密，而不是加密負載平衡器和 EnterpriseOne JVMs之間的流量。此方法稱為 SSL 卸載。將 SSL 解密程序從 EnterpriseOne Web 或應用程式伺服器卸載至 Application Load Balancer，可減輕應用程式端的負擔。在負載平衡器終止 SSL 之後，未加密的流量會路由到 AWS 上的應用程式。

Oracle JD Edwards EnterpriseOne 是企業資源規劃 (ERP) 解決方案，適用於生產、建構、分發、服務或管理產品或實體資產的組織。JD Edwards EnterpriseOne 支援各種硬體、作業系統和資料庫平台。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- AWS Identity and Access Management (IAM) 角色，具有發出 AWS 服務呼叫和管理 AWS 資源的許可
- SSL 憑證

產品版本

- 此模式已使用 Oracle WebLogic 12c 進行測試，但您也可以使用其他版本。

Architecture

執行 SSL 卸載的方法有多種。此模式使用 Application Load Balancer 和 Oracle HTTP Server (OHS)，如下圖所示。

下圖顯示 JD Edwards EnterpriseOne、Application Load Balancer 和 Java Application Server (JAS) JVM 配置。

工具

AWS 服務

- Application Load Balancer 會將傳入的應用程式流量分散到多個可用區域中的多個目標，例如 Amazon Elastic Compute Cloud (Amazon EC2 執行個體）。
- AWS Certificate Manager (ACM) 可協助您建立、存放和續約公有和私有 SSL/TLS X.509 憑證和金鑰，以保護 AWS 網站和應用程式。
- Amazon Route 53 是一種可用性高、可擴展性強的 DNS Web 服務。

最佳實務

- 如需 ACM 最佳實務，請參閱 ACM 文件。

史詩

設定 WebLogic 和 OHS

任務	Description	所需的技能
安裝和設定 Oracle 元件。	1. 遵循標準安裝程序來安裝 Fusion Middleware Infrastructure。此程式可協助您安裝和設定 WebLogic 網域。如需說明，請參閱 Oracle 文件。	JDE CNC、WebLogic 管理員

任務	Description	所需的技能
	2. 遵循標準安裝程序來安裝 OHS。如需說明，請參閱 Oracle 文件。 3. 安裝完成後，啟動組態精靈 (`config.sh` 檔案）來設定 OHS。 • 您可以更新現有的網域或建立新的網域。此模式假設您正在更新現有的網域。 • 針對可用範本，選擇 Oracle Enterprise Manager 限制 JRF 和 Oracle HTTP Server （限制 JRF)。選取這些 Java 必要檔案 (JRF) 選項可消除與外部資料庫的連線。 • 對於受管伺服器、叢集、伺服器範本、一致性叢集、機器、將伺服器指派給機器、虛擬目標和分割區，接受預設組態值，然後選擇下一步以移至下一個類別。 • 完成 OHS 執行個體的組態詳細資訊 （例如，管理員主機和連接埠、接聽地址和連接埠、伺服器名稱） （例如，)`ohs1`。	

任務	Description	所需的技能
在網域層級啟用 WebLogic 外掛程式。	負載平衡需要 WebLogic 外掛程式。若要啟用外掛程式： 1. 使用連結登入 WebLogic 管理主控台： `http://<WeblogicServer>:<Adminport>/console` 2. 選擇鎖定和編輯，然後選擇組態、Web 應用程式。 3. 選擇已啟用 WebLogic 外掛程式（核取方塊或下拉式清單選項）。 4. 選擇儲存並啟用變更。	JDE CNC、WebLogic 管理員

任務	Description	所需的技能
編輯組態檔案。	mod_wl_ohs.conf 檔案會設定從 OHS 到 WebLogic 的代理請求。 1. 編輯此檔案。其位於： $ORACLE_HOME/user_projects/domains/ 例如： /home/oracle/Oracle/Middleware/Oracle_Home/user_projects/domains/base_domain/config/fmwconfig/components/OHS/instances/ohs1 2. 新增 WebLogic 主機 (WebLogicHost) 和連接埠 (WebLogicPort) 值（此模式假設 localhost 和連接埠 8000。） 3. 新增 WLProxySSL 和 WLProxySSLPassThrough 值，如下所示： <VirtualHost *:8000> <Location /jde> WLSRequest On SetHandler weblogic-handler WebLogicHost localhost WebLogicPort 8000 WLProxySSL On	JDE CNC、WebLogic 管理員

任務	Description	所需的技能
	WLProxySSLPassThrough On </Location> </VirtualHost>	

任務	Description	所需的技能

使用 Enterprise Manager 啟動 OHS。	1. 使用連結登入 Enterprise Manager Fusion Middleware：
	
	`http://<WeblogicServer>:<Adminport>/em/`
	2. 在目標導覽的 HTTP 伺服器下，選取 OHS 執行個體（例如 ohs1)。
	3. 選擇關閉並啟動以重新啟動 OHS 執行個體。
	4. 當 OHS 設定完成時，您可以使用連接埠為 8000 的 HTTP 伺服器主機名稱而非 EnterpriseOne 伺服器主機名稱來連線至 EnterpriseOne HTML 用戶端。
	• 舊連結： `http://<Webserver>:80/jde/owhtml`
	• 新連結： `http://<HTTP server or web server>:8000/jde/owhtml`
	
	如果您使用預設 Oracle HTTP 連接埠以外的連接埠，請編輯 `httpd.conf` 檔案，在兩個位置新增該連接埠的接聽程式：
	
	`#[Listen] OHS_LISTEN_PORT`
		JDE CNC、WebLogic 管理員

任務	Description	所需的技能

	`Listen 8000`
	
	和：
	
	`# ServerName <Weblogic Server1>:8000`	

設定 Application Load Balancer

任務	Description	所需的技能
設定目標群組。	1. 建立 HTTP 伺服器連接埠 8000 的目標群組。 2. 使用相同的連接埠註冊目標群組下的目標。 3. 檢查目標的狀態，以確認其運作狀態良好。 4. 視需要設定運作狀態檢查設定。 如需詳細說明，請參閱 Elastic Load Balancing 文件。	AWS 管理員
設定負載平衡器。	1. 建立具有預設屬性和所需虛擬私有雲端 (VPC)、安全群組和子網路的 Application Load Balancer。如需說明，請參閱 Elastic Load Balancing 文件。 2. 新增 HTTPS 443 的接聽程式項目，並將其轉送至您在上一個步驟中建立的目標	AWS 管理員

任務	Description	所需的技能
	群組。（如需說明，請參閱 Elastic Load Balancing 文件。) HTTPS 接聽程式需要 SSL 憑證。您可以從 ACM 中選擇憑證或上傳憑證。 3. 對於這兩個接聽程式，請依照 Elastic Load Balancing 文件中的指示啟用黏性。	
新增 Route 53 (DNS) 記錄。	（選用）您可以為子網域新增 Amazon Route 53 DNS 記錄。此記錄會指向您的 Application Load Balancer。如需說明，請參閱 Route 53 文件。	AWS 管理員

疑難排解

問題	解決方案
HTTP 伺服器不會顯示。	如果 HTTP 伺服器未出現在 Enterprise Manager 主控台的目標導覽清單中，請依照下列步驟執行： 1. 在 WebLogic 網域管理下，選擇 OHS 執行個體。 2. 選擇建立以建立新的 OHS 執行個體。 3. 提供執行個體名稱，然後選擇確定以建立執行個體。 建立執行個體並啟用變更後，您就可以在目標導覽面板中看到 HTTP 伺服器。

相關資源

AWS 文件

- Application Load Balancer
- 使用公有託管區域
- 使用私有託管區域

Oracle 文件：

- Oracle WebLogic Server Proxy 外掛程式概觀
- 使用 Infrastructure Installer 安裝 WebLogic Server
- 安裝和設定 Oracle HTTP 伺服器

透過私有網路連線至 Application Migration Service 資料和控制平面

Dipin Jain 和 Mike Kuznetsov，Amazon Web Services

總結

此模式說明如何使用介面 VPC 端點，連線到私有安全網路上 AWS Application Migration Service 的資料平面和控制平面。

Application Migration Service 是高度自動化lift-and-shift（重新託管）解決方案，可簡化、加速和降低將應用程式遷移至 的成本 AWS。它可讓公司重新託管大量實體、虛擬或雲端伺服器，而不會發生相容性問題、效能中斷或長切換時段。Application Migration Service 可從 取得 AWS 管理主控台。這可與其他 無縫整合 AWS 服務，例如 AWS CloudTrail Amazon CloudWatch 和 AWS Identity and Access Management (IAM)。

您可以透過使用 Site-to-Site VPN 服務或 Application Migration Service 中的 VPC 對等互連 AWS Direct Connect，從來源資料中心連線到資料平面，也就是做為目的地 VPC 中資料複寫之暫存區域的子網路。您也可以使用 支援的介面 VPC 端點 AWS PrivateLink ，透過私有網路連線至 Application Migration Service 控制平面。

先決條件和限制

先決條件

- 預備區域子網路 – 在您設定 Application Migration Service 之前，請建立子網路，做為從來源伺服器複寫至 AWS （即資料平面）之資料的預備區域。首次存取 Application Migration Service 主控台時，您必須在複寫設定範本中指定此子網路。您可以在複寫設定範本中覆寫特定來源伺服器的此子網路。雖然您可以在 中使用現有的子網路 AWS 帳戶，但我們建議您為此建立新的專用子網路。
- 網路需求 – Application Migration Service 在預備區域子網路中啟動的複寫伺服器必須能夠將資料傳送至位於 的 Application Migration Service API 端點`https://mgn.<region>.amazonaws.com/`，其中 `<region>`是 AWS 區域 您要複寫之 的程式碼 （例如 `https://mgn.us-east-1.amazonaws.com`）。下載 Application Migration Service 軟體時，需要 Amazon Simple Storage Service (Amazon S3) 服務 URLs。
  - AWS 複寫代理程式安裝程式應可存取 AWS 區域 與 Application Migration Service 搭配使用之 的 Amazon Simple Storage Service (Amazon S3) 儲存貯體 URL。
  - 預備區域子網路應可存取 Amazon S3。
  - 安裝 AWS 複寫代理程式的來源伺服器必須能夠將資料傳送至預備區域子網路中的複寫伺服器，以及傳送至位於 的 Application Migration Service API 端點`https://mgn.<region>.amazonaws.com/`。

下表列出所需的連接埠。

來源	目標	站點	如需詳細資訊，請參閱
來源資料中心	Amazon S3 URLs	443 (TCP)	透過 TCP 連接埠 443 的通訊
來源資料中心	AWS 區域 Application Migration Service 的特定主控台地址	443 (TCP)	透過 TCP 連接埠 443 的來源伺服器與 Application Migration Service 之間的通訊
來源資料中心	暫存區域子網路	1500 (TCP)	透過 TCP 連接埠 1500 的來源伺服器與預備區域子網路之間的通訊
暫存區域子網路	AWS 區域 Application Migration Service 的特定主控台地址	443 (TCP)	透過 TCP 連接埠 443 的預備區域子網路與 Application Migration Service 之間的通訊
暫存區域子網路	Amazon S3 URLs	443 (TCP)	透過 TCP 連接埠 443 的通訊
暫存區域子網路	子網路的 Amazon Elastic Compute Cloud (Amazon EC2) 端點 AWS 區域	443 (TCP)	透過 TCP 連接埠 443 的通訊

限制

Application Migration Service 目前無法在所有 AWS 區域 和 作業系統中使用。

- 支援的 AWS 區域
- 支援的作業系統

Architecture

下圖說明典型遷移的網路架構。如需此架構的詳細資訊，請參閱 Application Migration Service 文件和 Application Migration Service 架構和網路架構影片。

下列詳細檢視顯示預備區域 VPC 中連接 Amazon S3 和 Application Migration Service 的介面 VPC 端點組態。

工具

- AWS Application Migration Service 簡化、加速並降低重新託管應用程式的成本 AWS。
- 介面 VPC 端點可讓您連線至由 提供支援的服務， AWS PrivateLink 而不需要網際網路閘道、NAT 裝置、VPN 連線或 AWS Direct Connect 連線。VPC 中的執行個體不需要公有 IP 地址，即可與服務中的資源通訊。VPC 與另一個服務之間的流量都會保持在 Amazon 網路的範圍內。

史詩

建立 Application Migration Service、Amazon EC2 和 Amazon S3 的端點

任務	Description	所需的技能
設定 Application Migration Service 的介面端點。	來源資料中心和預備區域 VPC 會透過您在目標預備區域 VPC 中建立的介面端點，私下連線至 Application Migration Service 控制平面。若要建立端點： 1. 開啟 Amazon Virtual Private Cloud (Amazon VPC) 主控台。 2. 在導覽窗格中，選擇 Endpoints (端點)，Create Endpoint (建立端點)。 3. 對於 Service category (服務類別)，選擇 AWS 服務。	遷移潛在客戶

任務	Description	所需的技能
	4. 針對服務名稱，輸入 `com.amazonaws.<region>.mgn` 。針對類型，選擇界面。 5. 針對 VPC，選取目標暫存區域 VPC 以建立端點。 6. 針對子網路，選取要在其中建立端點網路介面的子網路。 7. 若要開啟介面端點的私有 DNS，請在其他設定區段中，選取啟用 DNS 名稱。 8. 選取允許透過 TCP 443 從預備區域 VPC 子網路傳入的安全群組。 9. 選擇建立端點。 如需詳細資訊，請參閱《Amazon VPC 文件》中的 AWS 服務 使用介面 VPC 端點存取 。	

任務	Description	所需的技能
設定 Amazon EC2 的介面端點。	預備區域 VPC 會透過您在目標預備區域 VPC 中建立的介面端點，私下連線至 Amazon EC2 API。若要建立端點，請遵循上一個案例提供的指示。 • 針對服務名稱，輸入 `com.amazonaws.<region>.ec2` 。針對類型，選擇界面。 • 安全群組必須允許透過連接埠 443 從預備區域 VPC 子網路傳入的 HTTPS 流量。 • 在其他設定區段中，選取啟用 DNS 名稱。	遷移潛在客戶

任務	Description	所需的技能
設定 Amazon S3 的介面端點。	來源資料中心和預備區域 VPC 會透過您在目標預備區域 VPC 中建立的介面端點，私下連線至 Amazon S3 API。若要建立端點，請遵循第一個案例提供的指示。 • 針對服務名稱，輸入 `com.amazonaws.<region>.s3` 。針對類型，選擇界面。 • VPC 安全群組必須透過連接埠 443 允許來自暫存區域 VPC 子網路的傳入 HTTPS 流量。 • 在其他設定區段中，清除啟用 DNS 名稱。Amazon S3 介面端點不支援私有 DNS 名稱。 **Note** 您使用界面端點，因為閘道端點連線無法從 VPC 延伸。（如需詳細資訊，請參閱 AWS PrivateLink 文件。）	遷移潛在客戶

任務	Description	所需的技能
設定 Amazon S3 Gateway 端點。	在組態階段，複寫伺服器必須連線至 S3 AWS 儲存貯體，才能下載複寫伺服器的軟體更新。不過，Amazon S3 介面端點不支援私有 DNS 名稱，而且無法將 Amazon S3 端點 DNS 名稱提供給複寫伺服器。 若要緩解此問題，您可以在預備區域子網路所屬的 VPC 中建立 Amazon S3 閘道端點，並使用相關路由更新預備子網路的路由表。如需詳細資訊，請參閱 AWS PrivateLink 文件中的建立閘道端點。	雲端管理員
設定內部部署 DNS 以解析端點的私有 DNS 名稱。	Application Migration Service 和 Amazon EC2 的介面端點具有可在 VPC 中解析的私有 DNS 名稱。不過，您也需要設定內部部署伺服器來解析這些介面端點的私有 DNS 名稱。 設定這些伺服器的方法有多種。在此模式中，我們透過將內部部署 DNS 查詢轉送到暫存區域 VPC 中的 Amazon Route 53 Resolver 傳入端點來測試此功能。如需詳細資訊，請參閱 Route 53 文件中的解決 VPCs 與網路之間的 DNS 查詢。	遷移工程師

透過私有連結連線至 Application Migration Service 控制平面

任務	Description	所需的技能
使用 AWS 安裝複寫代理程式 AWS PrivateLink。	1. 將 AWS 複寫代理程式下載到目的地區域中的私有 S3 儲存貯體。 2. 登入要遷移的來源伺服器。 AWS 複寫代理程式安裝程式需要 Application Migration Service 和 Amazon S3 端點的網路存取權。由於您的內部部署網路未開放給 Application Migration Service 和 Amazon S3 公有端點，因此您必須使用 在先前步驟中建立的界面端點的協助下安裝 代理程式 AWS PrivateLink。 以下是 Linux 的範例： 1. 使用 命令下載 代理程式： `wget -O ./aws-replication-installer-init.py \ https://aws-application-migration-service-<aws_region>.bucket.<s3-endpoint-DNS-name>/latest/linux/aws-replication-installer-init.py`	遷移工程師

任務	Description	所需的技能
	例如，如果 Amazon S3 介面端點的 DNS 名稱是 `vpce-009c8b07adb052a11-qgf8q50y.s3.us-west-1.vpce.amazonaws.com` ，而 AWS 區域 是 `us-west-1` ，您將使用 命令： `wget -O ./aws-replication-installer-init.py \ https://aws-application-migration-service-us-west-1.bucket.vpce-009c8b07adb052a11-qgf8q50y.s3.us-west-1.vpce.amazonaws.com/latest/linux/aws-replication-installer-init.py` **Note** `bucket` 是靜態關鍵字，您必須在 Amazon S3 介面端點 DNS 名稱之前新增。如需詳細資訊，請參閱 Amazon S3 說明文件。 2. 安裝 代理程式： • 如果您在建立 Application Migration Service 的介面	

任務	Description	所需的技能

	端點時選取啟用 DNS 名稱，請執行 命令：
	
	`sudo python3 aws-replication-installer-init.py \ --region <aws_region> \ --aws-access-key-id <access-key> \ --aws-secret-access-key <secret-key> \ --no-prompt \ --s3-endpoint <s3-endpoint-DNS-name>`
	
	• 如果您在建立 Application Migration Service 的介面端點時未選取啟用 DNS 名稱，請執行 命令：
	
	`sudo python3 aws-replication-installer-init.py \ --region <aws_region> \ --aws-access-key-id <access-key> \ --aws-secret-access-key <secret-key> \ --no-prompt \ --s3-endpoint <s3-endpoint-DNS-name> \`	

任務	Description	所需的技能

	<pre>--endpoint
	<mgn-endpoint-DNS-name></pre>
	如需詳細資訊，請參閱 Application Migration Service 文件中的AWS 複寫代理程式安裝說明。
	
	在您與 Application Migration Service AWS 建立連線並安裝複寫代理程式之後，請依照 Application Migration Service 文件的指示，將來源伺服器遷移至目標 VPC 和子網路。	

相關資源

Application Migration Service 文件

- 概念
- 遷移工作流程
- 快速入門指南
- 常見問答集
- 疑難排解

其他資源

- AWS 使用 VPC 介面端點在 上的多帳戶架構中重新託管您的應用程式 (AWS 方案指引指南）
- AWS Application Migration Service – 技術簡介 (AWS 培訓和認證演練）
- AWS Application Migration Service 架構和網路架構 （影片）

其他資訊

針對 Linux 伺服器上的複寫代理程式安裝進行故障診斷 AWS

如果您在 Amazon Linux 伺服器上收到 gcc 錯誤，請設定套件儲存庫，並使用下列命令：

```
## sudo yum groupinstall "Development Tools"
```

使用 AWS CloudFormation 自訂資源和 Amazon SNS 建立 Infoblox 物件

Tim Sutton，Amazon Web Services

總結

注意： AWS Cloud9 不再提供給新客戶。的現有客戶 AWS Cloud9 可以繼續正常使用服務。進一步了解

Infoblox 網域名稱系統 (DNS)、動態主機組態通訊協定 (DHCP) 和 IP 地址管理 (Infoblox DDI) 可讓您集中並有效控制複雜的混合環境。使用 Infoblox DDI，除了使用相同的設備管理內部部署和 Amazon Web Services (AWS) 雲端上的 DNS 之外，您還可以在一個授權 IP 地址管理 (IPAM) 資料庫中探索和記錄所有網路資產。

此模式說明如何透過呼叫 Infoblox WAPI API，使用 AWS CloudFormation 自訂資源來建立 Infoblox 物件（例如 DNS 記錄或 IPAM 物件）。如需 Infoblox WAPI 的詳細資訊，請參閱 Infoblox 文件中的 WAPI 文件。

透過使用此模式的方法，除了移除建立記錄和佈建網路的手動程序之外，您還可以取得 AWS 和內部部署環境的 DNS 記錄和 IPAM 組態的統一檢視。您可以針對下列使用案例使用此模式的方法：

- 在建立 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體之後新增 A 記錄
- 在建立 Application Load Balancer 之後新增 CNAME 記錄
- 在建立虛擬私有雲端 (VPC) 之後新增網路物件
- 提供下一個網路範圍，並使用該範圍來建立子網路

您也可以擴展此模式並使用其他 Infoblox 裝置功能，例如新增不同的 DNS 記錄類型或設定 Infoblox vDiscovery。

模式使用中hub-and-spoke設計，其中中樞需要連線至 AWS 雲端或內部部署上的 Infoblox 設備，並使用 AWS Lambda 呼叫 Infoblox API。輪輻位於 AWS Organizations 中相同組織的相同或不同帳戶中，並使用 AWS CloudFormation 自訂資源呼叫 Lambda 函數。

先決條件和限制

先決條件

- 安裝在 AWS 雲端、內部部署或兩者上的現有 Infoblox 設備或網格，並使用可管理 IPAM 和 DNS 動作的管理員使用者進行設定。如需詳細資訊，請參閱 Infoblox 文件中的關於管理員帳戶。

- 您要在 Infoblox 設備上新增記錄的現有 DNS 授權區域。如需詳細資訊，請參閱 Infoblox 文件中的設定授權區域。
- AWS Organizations 中的兩個作用中 AWS 帳戶。一個帳戶是中樞帳戶，另一個帳戶是輻條帳戶。
- 中樞和輻條帳戶必須位於相同的 AWS 區域。
- 中樞帳戶的 VPC 必須連線至 Infoblox 設備；例如，使用 AWS Transit Gateway 或 VPC 對等互連。
- AWS Serverless Application Model (AWS SAM)，使用 AWS Cloud9 或 AWS CloudShell 在本機安裝和設定。
- `Infoblox-Hub.zip` 和 `ClientTest.yaml` 檔案（已連接），下載至包含 AWS SAM 的本機環境。

限制

- AWS CloudFormation 自訂資源的服務權杖必須來自建立堆疊的相同區域。我們建議您在每個區域中使用中樞帳戶，而不是在一個區域中建立 Amazon Simple Notification Service (Amazon SNS) 主題，並在另一個區域中呼叫 Lambda 函數。

產品版本

- Infoblox WAPI 2.7 版

Architecture

下圖顯示此模式的工作流程。

下圖顯示此模式解決方案的下列元件：

1. AWS CloudFormation 自訂資源可讓您在AWS CloudFormation、更新或刪除堆疊時執行的範本中撰寫自訂佈建邏輯。當您建立堆疊時，AWS CloudFormation 會將`create`請求傳送至由 EC2 執行個體上執行之應用程式監控的 SNS 主題。
2. 來自 AWS CloudFormation 自訂資源的 Amazon SNS 通知會透過特定 AWS Key Management Service (AWS KMS) 金鑰加密，且存取權僅限於 Organizations 中組織中的帳戶。SNS 主題會啟動呼叫 Infoblox WAPI API 的 Lambda 資源。
3. Amazon SNS 會叫用下列 Lambda 函數，以使用 Infoblox WAPI URL、使用者名稱和密碼 AWS Secrets Manager Amazon Resource Name (ARNs) 做為環境變數：

- `dnsapi.lambda_handler` – 從 AWS CloudFormation `DNSName`自訂資源接收 `DNSType`、 和 `DNSValue`值，並使用它們來建立 DNS A 記錄和 CNAMES。
- `ipaddr.lambda_handler` – 從 AWS CloudFormation 自訂資源接收 `VPCCIDR`、`SubnetPrefix`、 `Type`和 `Network Name`值，並使用這些資料將網路資料新增至 Infoblox IPAM 資料庫，或為自訂資源提供可用於建立新子網路的下一個可用網路。
- `describeprefixes.lambda_handler` – 使用`"com.amazonaws."+Region+".s3"`篩選條件來擷取所需的 ，以呼叫 `describe_managed_prefix_lists` AWS API`prefix ID`。

> **Important**
>
> 這些 Lambda 函數是以 Python 撰寫，彼此類似，但呼叫不同的 APIs。

4. 您可以將 Infoblox 網格部署為實體、虛擬或雲端型網路設備。 它可以使用一系列 Hypervisor 部署在內部部署或做為虛擬設備，包括 VMware ESXi、Microsoft Hyper-V、Linux KVM 和 Xen。您也可以使用 Amazon Machine Image (AMI) 在 AWS 雲端上部署 Infoblox 網格。
5. 此圖顯示 Infoblox 網格的混合解決方案，它將 DNS 和 IPAM 提供給 AWS 雲端和內部部署上的資源。

技術堆疊

- AWS CloudFormation
- IAM
- AWS KMS
- AWS Lambda
- AWS SAM
- AWS Secrets Manager
- Amazon SNS
- Amazon VPC

工具

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶和區域的生命週期中管理這些資源。

- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 AWS 帳戶合併到您建立並集中管理的組織。
- AWS Secrets Manager 可協助您以 API 呼叫 Secrets Manager，以程式設計方式擷取秘密，取代程式碼中的硬式編碼登入資料，包括密碼。
- AWS Serverless Application Model (AWS SAM) 是一種開放原始碼架構，可協助您在 AWS 雲端中建置無伺服器應用程式。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路類似於您在自己的資料中心內操作的傳統網路，具有使用可擴展的 AWS 基礎設施的優勢。

Code

您可以使用`ClientTest.yaml`範例 AWS CloudFormation 範本 （ 已連接 ） 來測試 Infoblox 中樞。您可以自訂 AWS CloudFormation 範本，以包含下表中的自訂資源。

使用 Infoblox 輻條自訂資源建立 A 記錄	傳回值：
	`infobloxref` – Infoblox 參考
	資源範例：
	`ARECORDCustomResource:`
	`Type: "Custom::InfobloxAPI"`
	`Properties:`

```
    ServiceToken: !Sub  arn:aws:sns:
${AWS::Region}:${HubAccountID}:Ru
nInfobloxDNSFunction

    DNSName: 'arecordtest.compa
ny.com'

    DNSType: 'ARecord'

    DNSValue: '10.0.0.1'
```

使用 Infoblox 語音自訂資源建立 CNAME 記錄

傳回值：

`infobloxref` – Infoblox 參考

資源範例：

```
CNAMECustomResource:

  Type: "Custom::InfobloxAPI"

  Properties:

    ServiceToken: !Sub arn:aws:sns:
${AWS::Region}:${HubAccountID}:Ru
nInfoblox

    DNSFunction

    DNSName: 'cnametest.company.com'

    DNSType: 'cname'

    DNSValue: 'aws.amazon.com'
```

使用 Infoblox 輻條自訂資源建立網路物件

傳回值：

infobloxref – Infoblox 參考

network – 網路範圍 （與 相同VPCCIDR)

資源範例：

```
VPCCustomResource:

  Type: 'Custom::InfobloxAPI'

  Properties:

    ServiceToken: !Sub  arn:aws:sns:
${AWS::Region}:${HubAccountID}:Ru
nInfobloxNextSubnetFunction

    VPCCIDR: !Ref VpcCIDR

    Type: VPC

    NetworkName: My-VPC
```

使用 Infoblox 輻條自訂資源擷取下一個可用的子網路	傳回值：
	`infobloxref` – Infoblox 參考
	`network` – 子網路的網路範圍
	資源範例：
	`Subnet1CustomResource:`
	`  Type: 'Custom::InfobloxAPI'`
	`  DependsOn: VPCCustomResource`
	`  Properties:`
	`  ServiceToken: !Sub arn:aws:sns:${AWS::Region}:${HubAccountID}:RunInfobloxNextSubnetFunction`
	`  VPCCIDR: !Ref VpcCIDR`
	`  Type: Subnet`
	`  SubnetPrefix: !Ref SubnetPrefix`
	`NetworkName: My-Subnet`

史詩

建立和設定中樞帳戶的 VPC

任務	Description	所需的技能
建立與 Infoblox 設備連線的 VPC。	登入您中樞帳戶的 AWS 管理主控台，並依照 AWS Quick Starts 中 AWS Cloud Quick Start 參考部署上的 Amazon VPC 中的步驟建立 VPC。	網路管理員、系統管理員

任務	Description	所需的技能
	Important VPC 必須具有與 Infoblox 設備的 HTTPS 連線，建議您為此連線使用私有子網路。	
（選用）建立私有子網路的 VPC 端點。	VPC 端點為您的私有子網路提供公有服務的連線。需要下列端點： • Amazon Simple Storage Service (Amazon S3) 的閘道端點，允許 Lambda 與 AWS CloudFormation 通訊 • Secrets Manager 的界面端點，可啟用與 Secrets Manager 的連線 • AWS KMS 的界面端點，允許 SNS 主題和 Secrets Manager 秘密的加密 如需為私有子網路建立端點的詳細資訊，請參閱 Amazon VPC 文件中的 VPC 端點。	網路管理員、系統管理員

部署 Infoblox 中樞

任務	Description	所需的技能
建置 AWS SAM 範本。	1. 在包含 AWS SAM 的環境中執行 `unzip Infoblox-Hub.zip` 命令。 2. 執行 `cd Hub/`命令，將您的目錄變更為 Hub目錄。 3. 執行 `sam build`命令來處理 AWS SAM 範本檔案、應用程式程式碼，以及任何特定語言的檔案和相依性。`sam build` 命令也會以下列案例預期的格式和位置複製建置成品。	開發人員、系統管理員

部署 AWS SAM 範本。	`sam deploy` 命令會取得必要的參數並將其儲存至 `samconfig.toml` 檔案、將 AWS CloudFormation 範本和 Lambda 函數存放在 S3 儲存貯體，然後將 AWS CloudFormation 範本部署到您的中樞帳戶。
	
	下列範例程式碼示範如何部署 AWS SAM 範本：
	
	`$ sam deploy --guided`
	
	`Configuring SAM deploy`
	`==========`
	`Looking for config file [samconfig.toml] : Found`
		開發人員、系統管理員

任務	Description	所需的技能
	Reading default arguments : Success Setting default arguments for 'sam deploy' ========== ========== Stack Name [Infoblox-Hub]: AWS Region [eu-west-1]: Parameter InfobloxUsername: Parameter InfobloxPassword: Parameter InfobloxIPAddress [xxx.xxx.xx.xxx]: Parameter AWSOrganisationID [o-xxxxxxxxx]: Parameter VPCID [vpc-xxxxxxxxx]: Parameter VPCCIDR [xxx.xxx.xxx.xxx/16]: Parameter VPCSubnetID1 [subnet-xxx]: Parameter VPCSubnetID2 [subnet-xxx]: Parameter VPCSubnetID3 [subnet-xxx]: Parameter VPCSubnetID4 []: #Shows you resources changes to be deployed and require a 'Y' to initiate deploy	

任務	Description	所需的技能

	<pre>Confirm changes before deploy [Y/n]: y
	#SAM needs permission to be able to create roles to connect to the resources in your template
	Allow SAM CLI IAM role creation [Y/n]: n
	Capabilities [['CAPABILITY_NAMED_IAM']]:
	Save arguments to configuration file [Y/n]: y
	SAM configuration file [samconfig.toml]:
	SAM configuration environment [default]:</pre>
	Important
	您每次都必須使用 `--guided`選項，因為 Infoblox 登入憑證不會存放在 `samconfig.toml` 檔案中。	

相關資源

- 使用 Postman 開始使用 WAPIs (Infoblox 部落格)
- 使用 BYOL 模型佈建 AWS 的 vNIOS (Infoblox 文件)
- quickstart-aws-vpc (GitHub 儲存庫)
- describe_managed_prefix_lists (適用於 Python 的 AWS 開發套件文件)

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

AWS 使用 Terraform 在 上建立階層式多區域 IPAM 架構

Donny Schreiber，Amazon Web Services

總結

IP 地址管理 (IPAM) 是網路管理的關鍵元件，隨著組織擴展其雲端基礎設施而變得越來越複雜。如果沒有適當的 IPAM，組織會面臨 IP 地址衝突、地址空間浪費，以及可能導致中斷和應用程式停機的複雜故障診斷的風險。此模式示範如何使用 HashiCorp Terraform 為 AWS 企業環境實作全面的 IPAM 解決方案。它可協助組織建立階層式多區域 IPAM 架構，以促進組織中所有 AWS 帳戶 的集中式 IP 地址管理AWS。

此模式可協助您使用複雜的四層集區階層實作 Amazon VPC IP Address Manager：頂層集區、區域集區、業務單位集區和環境特定集區。此結構支援適當的 IP 地址控管，同時啟用將 IP 管理委派給組織內適當的團隊。解決方案使用 AWS Resource Access Manager (AWS RAM) 在整個組織中無縫共用 IP Address Manager 集區。 AWS RAM 集中和標準化 IPAM 規格，團隊可以在所有受管帳戶中建置這些規格。

此模式可協助您達成下列目標：

- 自動化跨 AWS 區域、業務單位和環境的 IP 地址配置。
- 透過程式設計驗證強制執行組織網路政策。
- 隨著業務需求的演進，有效率地擴展網路基礎設施。
- 透過集中管理 IP 地址空間來降低營運開銷。
- 使用自助式 CIDR 範圍配置加速雲端原生工作負載部署。
- 透過以政策為基礎的控制和驗證來防止解決衝突。

先決條件和限制

先決條件

- 一或多個 AWS 帳戶，以組織身分管理 AWS Organizations。
- 做為 IP Address Manager 委派管理員的網路中樞或網路管理帳戶。
- AWS Command Line Interface (AWS CLI)，已安裝並設定。
- 已安裝 Terraform 1.5.0 版或更新版本。
- AWS Terraform 的提供者，已設定。

- 管理 IP Address Manager、 AWS RAM和虛擬私有雲端 (VPCs許可，已在 AWS Identity and Access Management (IAM) 中設定。

限制

- IP Address Manager 受限於服務配額。集區的預設服務配額為每個範圍 50 個。為 6 個區域、2 個業務單位和 4 個環境執行此部署會建立 67 個集區。因此，可能需要增加配額。
- 在配置資源之後修改或刪除 IP Address Manager 集區可能會導致相依性問題。您必須先釋出配置，才能刪除集區。
- 在 IP Address Manager 中，資源監控可能會在反映資源變更時發生些微延遲。此延遲可能約為 20 分鐘。
- IP Address Manager 無法自動強制執行不同範圍的 IP 地址唯一性。
- 自訂標籤必須遵循AWS 標記最佳實務。例如，每個金鑰都必須是唯一的，且不能以 開頭aws:。
- 將 IP Address Manager 與組織外部的帳戶整合時，有一些考量和限制。

Architecture

目標架構

IP Address Manager 組態和集區階層

下圖顯示目標架構的邏輯建構。範圍是 IP Address Manager 中的最高層級容器。每個範圍代表單一網路的 IP 地址空間。集區是範圍內連續 IP 地址範圍 （ 或 CIDR 範圍 ） 的集合。集區可協助您根據您的路由和安全需求組織 IP 地址。此圖表顯示四個層級的集區：頂層集區、區域集區、業務單位集區和環境集區。

此解決方案會建立清晰的 IP Address Manager 集區階層：

1. 最上層集區包含整個組織 IP 地址空間，例如 10.176.0.0/12。
2. 區域集區適用於區域特定的配置，例如 10.176.0.0/15 us-east-1。
3. 業務單位集區是每個集區中的特定網域配置 AWS 區域。例如， us-east-1區域中的財務業務單位可能會有 10.176.0.0/16。
4. 環境集區是不同環境的特定用途配置。例如， us-east-1區域中的財務業務單位可能具有生產環境10.176.0.0/18的 。

此部署拓撲會在地理上分配 IP Address Manager 資源，同時維持集中式控制。以下是其功能：

- IP Address Manager 部署在單一主要節點中 AWS 區域。
- 其他區域會註冊為操作區域，其中 IP Address Manager 可以管理資源。
- 每個操作區域都會從最上層集區接收專用地址集區。
- 所有操作區域中的資源都會透過主要區域中的 IP Address Manager 集中管理。
- 每個區域集區都有與其區域繫結的區域設定屬性，協助您正確配置資源。

進階 CIDR 範圍驗證

此解決方案旨在防止部署無效的組態。當您透過 Terraform 部署集區時，會在 Terraform 計劃階段驗證下列項目：

- 驗證所有環境 CIDR 範圍都包含在其父業務單位 CIDR 範圍內
- 確認所有業務單位 CIDR 範圍都包含在其父區域 CIDR 範圍內
- 驗證所有區域 CIDR 範圍是否包含在最上層 CIDR 範圍內
- 檢查相同階層層級內是否有重疊的 CIDR 範圍
- 驗證環境與其個別業務單位的適當映射

CIDR 範圍配置

下圖顯示開發人員或管理員如何建立新的 VPCs 並從集區層級配置 IP 地址的範例。

該圖顯示以下工作流程：

1. 透過 AWS 管理主控台、 或透過基礎設施即程式碼 (IaC) AWS CLI，開發人員或管理員會請求AY3環境集區中下一個可用的 CIDR 範圍。
2. IP Address Manager 會將該集區中的下一個可用 CIDR 範圍配置給 AY3-4 VPC。無法再使用此 CIDR 範圍。

自動化和擴展

此解決方案專為可擴展性而設計，如下所示：

- 區域擴展 – 透過使用其他區域集區項目擴展 Terraform 組態來新增區域。

- 業務單位成長 – 透過將新業務單位新增至 BU 組態映射來支援新業務單位。
- 環境彈性 – 根據組織需求設定不同的環境類型，例如開發或生產。
- 多帳戶支援 – 透過 將集區分享到組織中的所有帳戶 AWS RAM。
- 自動化 VPC 佈建 – 與 VPC 佈建工作流程整合，以自動化 CIDR 範圍配置。

階層結構也允許不同規模的委派和控制，如下所示：

- 網路管理員可能會管理最上層和區域集區。
- 業務單位 IT 團隊可能已委派控制各自的集區。
- 應用程式團隊可能會使用其指定環境集區的 IP 地址。

> **Note**
>
> 您也可以將此解決方案與 AWS Control Tower Account Factory for Terraform (AFT) 整合。如需詳細資訊，請參閱此模式額外資訊區段中的與 AFT 整合。

工具

AWS 服務

- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您在 上執行的應用程式。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。
- AWS Resource Access Manager (AWS RAM) 可協助您安全地跨 共用資源 AWS 帳戶 ，以減少營運開銷並提供可見性和可稽核性。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。IP Address Manager 是 Amazon VPC 的一項功能。它可協助您規劃、追蹤和監控 AWS 工作負載的 IP 地址。

其他工具

- HashiCorp Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。

程式碼儲存庫

此模式的程式碼可在 GitHub 上儲存庫的階層式 IPAM 實作範例 AWS 中取得。儲存庫結構包括：

- 根模組 – 部署協同運作和輸入變數。
- IPAM 模組 – 此模式中所述架構的核心實作。
- 標籤模組 – 所有資源的標準化標記。

最佳實務

請考慮下列網路規劃的最佳實務：

- 先規劃 – 在部署之前徹底規劃 IP 地址空間。如需詳細資訊，請參閱規劃 IP 地址佈建。
- 避免重疊的 CIDR 範圍 – 確保每個層級的 CIDR 範圍不會重疊。
- 預留緩衝空間 – 一律配置比立即所需更大的 CIDR 範圍，以適應成長。
- 文件 IP 地址配置 – 維護 IP 地址配置策略的文件。

請考慮下列部署最佳實務：

- 從非生產開始 – 首先在非生產環境中部署。
- 使用 Terraform 狀態管理 – 實作遠端狀態儲存和鎖定。如需詳細資訊，請參閱 Terraform 文件中的狀態儲存和鎖定。
- 實作版本控制 – 版本控制所有 Terraform 程式碼。
- 實作 CI/CD 整合 – 使用持續整合和持續交付 (CI/CD) 管道進行可重複的部署。

請考慮下列操作最佳實務：

- 啟用自動匯入 – 設定 IP Address Manager 集區以自動探索和匯入現有資源。依照編輯 IPAM 集區中的指示開啟自動匯入。
- 監控 IP 地址使用率 – 設定 IP 地址使用率閾值的警示。如需詳細資訊，請參閱使用 Amazon CloudWatch 監控 IPAM。

- 定期稽核 – 定期稽核 IP 地址使用情況和合規性。如需詳細資訊，請參閱追蹤 IPAM 中的 IP 地址用量。
- 清除未使用的配置 – 解除委任資源時釋出 IP 地址配置。如需詳細資訊，請參閱從集區取消佈建 CIDRs。

請考慮下列安全最佳實務：

- 實作最低權限 – 使用具有最低必要許可的 IAM 角色。如需詳細資訊，請參閱 IAM 中的安全最佳實務和 IPAM 中的身分和存取管理。
- 使用服務控制政策 – 實作服務控制政策 SCPs)，以在組織中強制執行 IP Address Manager 用量。如需詳細資訊，請參閱使用 SCPs 強制使用 IPAM 建立 VPC。
- 控制資源共用 – 仔細管理其中 IP Address Manager 資源共用的範圍 AWS RAM。如需詳細資訊，請參閱使用 共用 IPAM 集區 AWS RAM。
- 強制標記 – 對與 IP Address Manager 相關的所有資源實作強制標記。如需詳細資訊，請參閱其他資訊區段中的標記策略。

史詩

設定 IP Address Manager 的委派管理員帳戶

任務	Description	所需的技能
啟用 AWS Organizations 功能。	確定 AWS Organizations 已啟用所有功能。如需說明，請參閱 AWS Organizations 文件中的為使用 的組織啟用所有功能 AWS Organizations。	AWS 管理員
在 中啟用資源共用 AWS RAM。	使用 AWS CLI，輸入下列命令為您的組織啟用 AWS RAM 資源共用： `aws ram enable-sharing-with-aws-organization`	AWS 管理員

任務	Description	所需的技能
	如需詳細資訊，請參閱 AWS RAM 文件中的在 中啟用資源共用 AWS Organizations。	
指定 IP Address Manager 的管理員。	從組織的管理帳戶中，使用 AWS CLI輸入下列命令，其中 123456789012 是將管理 IP Address Manager 的帳戶 ID： `aws ec2 enable-ipam-organization-admin-account \ --delegated-admin-account-id 123456789012` **Note** 一般而言，網路或網路中樞帳戶會用作 IP Address Manager 的委派管理員。 如需詳細資訊，請參閱 IP Address Manager 文件中的將 IPAM 與 AWS Organization 中的帳戶整合。	AWS 管理員

部署基礎設施

任務	Description	所需的技能
定義網路架構。	定義並記錄您的網路架構，包括區域、業務單位和環境的	網路工程師

任務	Description	所需的技能
	CIDR 範圍。如需詳細資訊，請參閱 IP Address Manager 文件中的規劃 IP 地址佈建。	
複製儲存庫。	1. 輸入下列命令，將儲存庫複製到本機工作站： `git clone https://github.com/aws-samples/sample-amazon-vpc-ipam-terraform.git` 2. 輸入下列命令以導覽至 `terraform` 目錄。	DevOps 工程師
設定變數。	1. 開啟 terraform.tfvars 檔案。 2. 為您的 IP Address Manager 部署設定變數。如需已完成檔案的範例，請參閱程式碼儲存庫中的 terraform.tfvars.example。 3. 儲存並關閉 terraform.tfvars 檔案。	Terraform 網路工程師

任務	Description	所需的技能

部署 IP Address Manager 資源。	1. 輸入下列命令來初始化 Terraform：
	`terraform init`
	2. 輸入下列命令來產生執行計畫：
	`terraform plan`
	3. 檢閱計劃，並驗證要建立的資源和基礎設施元件。
	4. 輸入下列命令來部署基礎設施：
	`terraform apply`
	5. 出現提示時，請輸入 yes 以確認部署。
	6. 等待部署完成。
		Terraform

驗證部署。	1. 登入您的 AWS 帳戶，然後開啟 IP Address Manager 主控台。 2. 在導覽窗格中選擇 Pools (集區)。 3. 在集區清單中，確認已建立您設定的集區。 4. 開啟 AWS RAM 主控台。 5. 確認 IP Address Manager 資源已共用。遵循檢視您在文件中建立的資源共用中的指示。 AWS RAM	General AWS，網路工程師

建立 VPCs並設定監控

任務	Description	所需的技能
建立 VPC。	請遵循 Amazon VPC 文件中建立 VPC 中的步驟。當您達到為 VPC 選擇 CIDR 範圍的步驟時，請從您的區域、業務單位和環境集區配置下一個可用的範圍。	一般 AWS、網路管理員、網路工程師
驗證 CIDR 範圍配置。	1. 開啟 IP Address Manager 主控台。 2. 在導覽窗格中，選擇 Dashboard (儀表板)。 3. 在儀表板中，確認 CIDR 範圍已配置到您建立的 VPC。	一般 AWS、網路管理員、網路工程師
監控 IP Address Manager。	設定與 IP Address Manager 資源配置相關的監控和警示。如需詳細資訊和指示，請參閱 IP Address Manager 文件中的使用 Amazon CloudWatch 監控 IPAM 和依資源監控 CIDR 用量。	一般 AWS
強制使用 IP Address Manager。	在 中建立服務控制政策 (SCP) AWS Organizations ，要求組織中的成員在建立 VPC 時使用 IP Address Manager。如需說明，請參閱 IP Address Manager 文件中的強制使用 IPAM 搭配 SCPs 建立 VPC。	General AWS、AWS 管理員

疑難排解

問題	解決方案
找不到 IP Address Manager 資源時 Terraform 失敗	請確定 IP Address Manager 管理員帳戶已正確委派，且您的 AWS 提供者已向該帳戶進行身分驗證。
CIDR 範圍配置失敗	檢查請求的 CIDR 範圍是否在 IP Address Manager 集區的可用範圍內，且未與現有的配置重疊。
AWS RAM 共用問題	確認您的 AWS Organization 已啟用資源共用。確認 AWS RAM 共享中使用了正確的主體，即組織 Amazon Resource Name (ARN)。
集區階層驗證錯誤	請確定子集區 CIDR 範圍已正確包含在其父集區 CIDR 範圍內，且不會與同級集區重疊。
超過 IP Address Manager 配額限制	請求提高 IP Address Manager 集區的配額。如需詳細資訊，請參閱「Service Quotas 使用者指南」中的請求提高配額。

相關資源

AWS 服務 文件

- Amazon VPC IP Address Manager 文件
- AWS Resource Access Manager 文件
- AWS Organizations 文件

AWS 部落格文章

- 使用 Amazon VPCs IP Address Manager 管理跨 VPC 和區域的 IP 集區
- 使用 Amazon VPC IP Address Manager 進行大規模的網路地址管理和稽核

影片和教學課程

- AWS re：Invent 2022：Amazon VPC 設計和 IPAM (NET310) 的最佳實務
- AWS re：Invent 2022：進階 VPC 設計和新功能 (NET401)

其他資訊

與 AFT 整合

您可以將此解決方案與 AWS Control Tower Account Factory for Terraform (AFT) 整合，以確保新佈建的帳戶自動接收適當的網路組態。透過在網路中樞帳戶中部署此 IPAM 解決方案，透過 AFT 建立的新帳戶可以在建立 VPCs 時參考共用 IP Address Manager 集區。

下列程式碼範例示範使用 AWS Systems Manager 參數存放區在帳戶自訂中的 AFT 整合：

```
# Get the IP Address Manager pool ID from Parameter Store
data "aws_ssm_parameter" "dev_ipam_pool_id" {
  name = "/org/network/ipam/finance/dev/pool-id"
}

# Create a VPC using the IP Address Manager pool
resource "aws_vpc" "this" {
  ipv4_ipam_pool_id   = data.aws_ssm_parameter.dev_ipam_pool_id.value
  ipv4_netmask_length = 24

  tags = {
    Name = "aft-account-vpc"
  }
}
```

標記策略

解決方案實作全方位的標記策略，以促進資源管理。下列程式碼範例示範如何使用：

```
# Example tag configuration
module "tags" {
  source = "./modules/tags"

  # Required tags
  product_name  = "enterprise-network"
  feature_name  = "ipam"
  org_id        = "finance"
  business_unit = "network-operations"
```

```
  owner          = "network-team"
  environment    = "prod"
  repo           = "https://github.com/myorg/ipam-terraform"
  branch         = "main"
  cost_center    = "123456"
  dr_tier        = "tier1"

  # Optional tags
  optional_tags = {
    "project"    = "network-modernization"
    "stack_role" = "infrastructure"
  }
}
```

這些標籤會自動套用至所有 IP Address Manager 資源。這有助於一致的控管、成本分配和資源管理。

自訂 的 Amazon CloudWatch 提醒 AWS Network Firewall

Jason Owens，Amazon Web Services

總結

模式可協助您自訂由 產生的 Amazon CloudWatch 警示 AWS Network Firewall。您可以使用預先定義的規則，或建立自訂規則來判斷提醒的訊息、中繼資料和嚴重性。然後，您可以對這些提醒採取行動，或自動回應其他 Amazon 服務，例如 Amazon EventBridge。

在此模式中，您會產生 Suricata 相容防火牆規則。Suricata 是一種開放原始碼威脅偵測引擎。您首先建立簡單的規則，然後測試它們以確認產生並記錄 CloudWatch 提醒。成功測試規則後，您可以修改規則以定義自訂訊息、中繼資料和嚴重性，然後再次測試以確認更新。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- AWS Command Line Interface (AWS CLI) 在 Linux、macOS 或 Windows 工作站上安裝和設定。如需詳細資訊，請參閱安裝或更新最新版本的 AWS CLI。
- AWS Network Firewall 已安裝並設定為使用 CloudWatch Logs。如需詳細資訊，請參閱從 記錄網路流量 AWS Network Firewall。
- 受 Network Firewall 保護之虛擬私有雲端 (VPC) 私有子網路中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。

產品版本

- 對於 第 1 版 AWS CLI，請使用 1.18.180 或更新版本。對於 第 2 版 AWS CLI，請使用 2.1.2 或更新版本。
- 來自 Suricata 5.0.2 版的 classification.config 檔案。如需此組態檔案的副本，請參閱其他資訊一節。

Architecture

架構圖顯示下列工作流程：

1. 私有子網路中的 Amazon EC2 執行個體會使用 curl 或 Wget 提出請求。

2. Network Firewall 會處理流量並產生提醒。
3. Network Firewall 會將記錄的警示傳送至 CloudWatch Logs。

工具

AWS 服務

- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您在 上執行的應用程式。
- Amazon CloudWatch Logs 可協助您集中所有系統、應用程式的日誌， AWS 服務 以便您可以監控日誌並將其安全地存檔。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- AWS Network Firewall 是 AWS 雲端中的虛擬私有雲端 (VPC) 的有狀態、受管網路防火牆以及入侵偵測和預防服務。

其他工具

- curl 是開放原始碼命令列工具和程式庫。
- GNU Wget 是免費命令列工具。

史詩

建立防火牆規則和規則群組

任務	Description	所需的技能
建立規則。	1. 在文字編輯器中，建立您要新增至防火牆的規則清單。每個規則必須位於單獨的一行。`classtype` 參數中的值來自預設 Suricata 分類組態檔案。如需完整的組態檔案內容，請參閱其他資訊一節。以下是兩個規則範例。	AWS 系統管理員、網路管理員

任務	Description	所需的技能

	<pre>alert http any
	 any -> any any
	 (content:"badstuff
	"; classtype:misc-
	activity; sid:3;
	 rev:1;)
	alert http any any
	 -> any any (content:
	"morebadstuff";
	 classtype:bad-unkn
	own; sid:4; rev:1;)</pre> 2. 將規則儲存在名為 的檔案中`custom.rules` 。	

任務	Description	所需的技能

建立規則群組。	在 中 AWS CLI，輸入下列命令。這會建立規則群組。
	
	`# aws network-firewall create-rule-group \ --rule-group-name custom --type STATEFUL \ --capacity 10 --rules file://custom.rules \ --tags Key=environment,Value=development`
	
	以下為範例輸出。請記下RuleGroupArn 您在後續步驟中需要的 。
	
	`{ "UpdateToken": "4f998d72-973c-490a-bed2-fc3460547e23", "RuleGroupResponse": { "RuleGroupArn": "arn:aws:network-firewall:us-east-2:1234567890:stateful-rulegroup/custom", "RuleGroupName": "custom", "RuleGroupId": "238a8259-9eaf-48bb-90af-5e690cf8c48b", "Type": "STATEFUL", "Capacity": 10, "RuleGroupStatus": "ACTIVE",`
		AWS 系統管理員

任務	Description	所需的技能

	<pre>"Tags": [
	{
	"Key": "environment",
	"Value": "development"
	}
	]
	}</pre>	

更新防火牆政策

任務	Description	所需的技能

取得防火牆政策的 ARN。	在 中 AWS CLI，輸入下列命令。這會傳回防火牆政策的 Amazon Resource Name (ARN)。記錄 ARN 以供稍後在此模式中使用。
	<pre># aws network-firewall describe-firewall \
	--firewall-name aws-network-firewall-anfw \
	--query 'Firewall.FirewallPolicyArn'</pre>
	以下是此命令傳回的範例 ARN。
	<pre>"arn:aws:network-firewall:us-east-2:1234567890:firewall-policy/firewall-policy-anfw"</pre>
		AWS 系統管理員

任務	Description	所需的技能
更新防火牆政策。	在文字編輯器中，複製貼上下列程式碼。<RuleGroupArn> 將 取代為您在上一個 epic 中記錄的值。儲存檔案為 firewall-policy-anfw.json 。 `{ "StatelessDefaultActions": [ "aws:forward_to_sfe" ], "StatelessFragmentDefaultActions": [ "aws:forward_to_sfe" ], "StatefulRuleGroupReferences": [ { "ResourceArn": "<RuleGroupArn>" } ] }` 在 中輸入下列命令 AWS CLI。此命令需要更新字符才能新增新規則。字符用於確認自您上次擷取以來，政策尚未變更。 `UPDATETOKEN=(`aws network-firewall describe-firewall-policy \ --firewall-policy-name	AWS 系統管理員

任務	Description	所需的技能
	firewall-policy-anfw \ --output text --query UpdateToken`) aws network-firewall update-firewall-policy \ --update-token $UPDATETOKEN \ --firewall-policy-name firewall-policy-anfw \ --firewall-policy file://firewall-policy-anfw.json	

任務	Description	所需的技能

確認政策更新。	（選用）如果您想要確認已新增規則並檢視政策格式，請在中輸入下列命令 AWS CLI。
	
	`# aws network-firewall describe-firewall-policy \ --firewall-policy-name firewall-policy-anfw \ --query FirewallPolicy`
	
	以下為範例輸出。
	
	`{ "StatelessDefaultActions": [ "aws:forward_to_sfe" ], "StatelessFragmentDefaultActions": [ "aws:forward_to_sfe" ], "StatefulRuleGroupReferences": [ { "ResourceArn": "arn:aws:network-firewall:us-east-2:1234567890:stateful-rulegroup/custom" } ] }`
		AWS 系統管理員

測試提醒功能

任務	Description	所需的技能

產生測試提醒。	1. 登入防火牆子網路內的測試工作站。
	2. 輸入應該產生提醒的命令。例如，您可以使用 `wget`或 `curl`。
	`wget -U "badstuff" http://www.amazon.com -o /dev/null`
	`curl -A "morebadstuff" http://www.amazon.com -o /dev/null`
		AWS 系統管理員

驗證是否已記錄提醒。	1. 開啟 CloudWatch 主控台。 2. 導覽至正確的日誌群組和串流。如需詳細資訊，請參閱檢視傳送至 CloudWatch Logs 的日誌資料 (CloudWatch Logs 文件）。 3. 確認記錄的事件與下列範例類似。這些範例僅顯示提醒的相關部分。 範例 1 `"alert": { "action": "allowed", "signature_id": 3, "rev": 1,`	AWS 系統管理員

任務	Description	所需的技能

	<pre> "signatur
	e": "",
	            "category
	": "Misc activity",
	            "severity
	": 3
	        }</pre>
	範例 2
	<pre> "alert": {
	            "action":
	 "allowed",
	            "signatur
	e_id": 4,
	            "rev": 1,
	            "signatur
	e": "",
	            "category
	": "Potentially Bad
	 Traffic",
	            "severity
	": 2
	        }</pre>	

更新防火牆規則和規則群組

任務	Description	所需的技能

更新防火牆規則。	1. 在文字編輯器中，開啟 `custom.rules` 檔案。
	2. 將第一個規則變更為類似以下內容。此規則必須在 檔案的單一行中輸入。
	<pre>alert http any any ->
	 any any (msg:"Watch</pre>
		AWS 系統管理員

任務	Description	所需的技能
	out - Bad Stuff!!"; content:"badstuff" ; classtype:misc-activity; priority: 2; sid:3; rev:2; metadata:custom-field-2 Danger!, custom-field More Info;) 這會對規則進行下列變更： • 新增訊息 (Suricata 網站）字串，提供簽章或提醒的相關文字資訊。在產生的提醒中，這會映射到簽章。 • 將預設優先順序 (Suricata 網站） misc-activity 從 3 調整為 2。如需各種 的預設值classtypes ，請參閱其他資訊一節。 • 將自訂中繼資料 (Suricata 網站）新增至提醒。這是新增到簽章的其他資訊。建議您使用鍵值對。 • 將 rev (Suricata 網站）從 1 變更為 2。這表示簽章的版本。	

任務	Description	所需的技能

更新規則群組。	在 中 AWS CLI，執行下列命令。使用防火牆政策的 ARN。這些命令會取得更新字符，並使用規則變更來更新規則群組。
	`# UPDATETOKEN=(`aws network-firewall \ describe-rule-group \ --rule-group-arn arn:aws:network-firewall:us-east-2:123457890:stateful-rulegroup/custom \ --output text --query UpdateToken`)`
	`# aws network-firewall update-rule-group \ --rule-group-arn arn:aws:network-firewall:us-east-2:1234567890:stateful-rulegroup/custom \ --rules file://custom.rules \ --update-token $UPDATETOKEN`
	以下為範例輸出。
	`{ "UpdateToken": "7536939f-6a1d-414c-96d1-bb28110996ed", "RuleGroupResponse": {`
		AWS 系統管理員

任務	Description	所需的技能
	"RuleGroupArn": "arn:aws:network-firewall:us-east-2:1234567890:stateful-rulegroup/custom", "RuleGroupName": "custom", "RuleGroupId": "238a8259-9eaf-48bb-90af-5e690cf8c48b", "Type": "STATEFUL", "Capacity": 10, "RuleGroupStatus": "ACTIVE", "Tags": [{ "Key": "environment", "Value": "development" }] } }	

測試更新的提醒功能

任務	Description	所需的技能
產生測試提醒。	1. 登入防火牆子網路內的測試工作站。 2. 輸入應該產生提醒的命令。例如，您可以使用 `curl`.	AWS 系統管理員

任務	Description	所需的技能
	`curl -A "badstuff" http://www.amazon.com -o /dev/null`	
驗證已變更的提醒。	1. 開啟 CloudWatch 主控台。 2. 導覽至正確的日誌群組和串流。 3. 確認記錄的事件與下列範例類似。此範例僅顯示提醒的相關部分。 `"alert": { "action": "allowed", "signature_id": 3, "rev": 2, "signature": "Watch out - Bad Stuff!!", "category": "Misc activity", "severity": 2, "metadata": { "custom-field": [ "More Info" ], "custom-field-2": [ "Danger!" ] } }`	AWS 系統管理員

相關資源

參考

- 從 傳送提醒 AWS Network Firewall 到 Slack 頻道 (AWS 方案指引)
- AWS 使用 Suricata 在 上擴展威脅預防 (AWS 部落格文章)
- 的部署模型 AWS Network Firewall (AWS 部落格文章)
- Suricata 中繼金鑰 (Suricata 文件)

教學課程和影片

- AWS Network Firewall 研討會

其他資訊

以下是來自 Suricata 5.0.2 的分類組態檔案。建立防火牆規則時會使用這些分類。

```
# config classification:shortname,short description,priority

config classification: not-suspicious,Not Suspicious Traffic,3
config classification: unknown,Unknown Traffic,3
config classification: bad-unknown,Potentially Bad Traffic, 2
config classification: attempted-recon,Attempted Information Leak,2
config classification: successful-recon-limited,Information Leak,2
config classification: successful-recon-largescale,Large Scale Information Leak,2
config classification: attempted-dos,Attempted Denial of Service,2
config classification: successful-dos,Denial of Service,2
config classification: attempted-user,Attempted User Privilege Gain,1
config classification: unsuccessful-user,Unsuccessful User Privilege Gain,1
config classification: successful-user,Successful User Privilege Gain,1
config classification: attempted-admin,Attempted Administrator Privilege Gain,1
config classification: successful-admin,Successful Administrator Privilege Gain,1

# NEW CLASSIFICATIONS
config classification: rpc-portmap-decode,Decode of an RPC Query,2
config classification: shellcode-detect,Executable code was detected,1
config classification: string-detect,A suspicious string was detected,3
config classification: suspicious-filename-detect,A suspicious filename was detected,2
config classification: suspicious-login,An attempted login using a suspicious username
 was detected,2
```

```
config classification: system-call-detect,A system call was detected,2
config classification: tcp-connection,A TCP connection was detected,4
config classification: trojan-activity,A Network Trojan was detected, 1
config classification: unusual-client-port-connection,A client was using an unusual
 port,2
config classification: network-scan,Detection of a Network Scan,3
config classification: denial-of-service,Detection of a Denial of Service Attack,2
config classification: non-standard-protocol,Detection of a non-standard protocol or
 event,2
config classification: protocol-command-decode,Generic Protocol Command Decode,3
config classification: web-application-activity,access to a potentially vulnerable web
 application,2
config classification: web-application-attack,Web Application Attack,1
config classification: misc-activity,Misc activity,3
config classification: misc-attack,Misc Attack,2
config classification: icmp-event,Generic ICMP event,3
config classification: inappropriate-content,Inappropriate Content was Detected,1
config classification: policy-violation,Potential Corporate Privacy Violation,1
config classification: default-login-attempt,Attempt to login by a default username and
 password,2

# Update
config classification: targeted-activity,Targeted Malicious Activity was Detected,1
config classification: exploit-kit,Exploit Kit Activity Detected,1
config classification: external-ip-check,Device Retrieving External IP Address
 Detected,2
config classification: domain-c2,Domain Observed Used for C2 Detected,1
config classification: pup-activity,Possibly Unwanted Program Detected,2
config classification: credential-theft,Successful Credential Theft Detected,1
config classification: social-engineering,Possible Social Engineering Attempted,2
config classification: coin-mining,Crypto Currency Mining Activity Detected,2
config classification: command-and-control,Malware Command and Control Activity
 Detected,1
```

使用 Terraform 在 AWS Wavelength 區域中部署資源

Zahoor Chaudhrey 和 Luca Iannario，Amazon Web Services

總結

AWS Wavelength 可協助您建置針對多存取邊緣運算 (MEC) 應用程式最佳化的基礎設施。Wavelength 區域是 AWS 基礎設施部署，可將 AWS 運算和儲存服務嵌入通訊服務供應商 (CSP) 5G 網路。來自 5G 裝置的應用程式流量會到達在 Wavelength 區域中執行的應用程式伺服器，而不會離開電信網路。以下內容有助於透過 Wavelength 進行網路連線：

- 虛擬私有雲端 (VPCs) – 中的 VPCs AWS 帳戶 可以延伸到多個可用區域，包括 Wavelength 區域。Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和相關服務會顯示為區域 VPC 的一部分。VPCs是在 Amazon Virtual Private Cloud (Amazon VPC) 中建立和管理。
- 電信業者閘道 – 電信業者閘道可透過 CSP 的網路，從 Wavelength 區域中的子網路連線至 CSP 網路、網際網路或 AWS 區域 。電信業者閘道有兩個用途。它允許來自特定位置 CSP 網路的傳入流量，並允許傳出流量到電信網路和網際網路。

此模式及其相關聯的 Terraform 程式碼可協助您在 Wavelength 區域中啟動資源，例如 Amazon EC2 執行個體、Amazon Elastic Block Store (Amazon EBS) 磁碟區、VPCs、子網路和電信業者閘道。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 整合式開發環境 (IDE)
- 選擇加入目標 Wavelength 區域
- AWS Command Line Interface (AWS CLI)，已安裝並設定
- Terraform 1.8.4 版或更新版本，已安裝 (Terraform 文件)
- Terraform AWS Provider 5.32.1 版或更新版本，已設定 (Terraform 文件)
- Git，已安裝 (GitHub)
- 建立 Amazon VPC、Wavelength 和 Amazon EC2 資源的許可

限制

並非所有 都 AWS 區域 支援 Wavelength 區域。如需詳細資訊，請參閱 Wavelength 文件中的可用 Wavelength 區域。

Architecture

下圖顯示如何在 Wavelength 區域中建立子網路 AWS 和資源。在 Wavelength 區域中包含子網路的 VPCs 可以連接到電信業者閘道。電信業者閘道可讓您連線至下列資源：

- 電信業者網路上的 4G/LTE 和 5G 裝置。
- 已修正特定 Wavelength Zone 合作夥伴的無線存取。如需詳細資訊，請參閱多重存取 AWS Wavelength。
- 將流量傳出至公有網際網路資源。

工具

AWS 服務

- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。
- AWS Wavelength 將 AWS 雲端 基礎設施擴展到電信供應商的 5G 網路。這可協助您建置應用程式，為行動裝置和最終使用者提供極低延遲。

其他工具

- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。

程式碼儲存庫

此模式的程式碼可在 GitHub 使用 Terraform 建立 AWS Wavelength 基礎設施儲存庫中取得。Terraform 程式碼會部署下列基礎設施和資源：

- VPC
- Wavelength 區域
- Wavelength 區域中的 pubic 子網路
- Wavelength 區域中的電信業者閘道

- Wavelength 區域中的 Amazon EC2 執行個體

最佳實務

- 部署之前，請確認您使用的是最新版本的 Terraform 和 AWS CLI。
- 使用持續整合和持續交付 (CI/CD) 管道來部署 IaC。如需詳細資訊，請參閱 AWS 部落格上 AWS CI/CD 管道中管理 Terraform 狀態檔案的最佳實務。

史詩

佈建 基礎設施

任務	Description	所需的技能
複製儲存庫。	輸入下列命令，將使用 Terraform 儲存庫建立 AWS Wavelength 基礎設施複製到您的環境。 `git clone git@github.com:aws-samples/terraform-wavelength-infrastructure.git`	DevOps 工程師
更新變數。	1. 導覽至複製的儲存庫。 `cd terraform-wavelength-infrastructure` 2. 使用任何文字編輯器，在根目錄中建立名為 terraform.tfvars 的檔案。 3. 建立下列變數並輸入其值： • `region = <enter Region name>`	DevOps 工程師，Terraform

任務	Description	所需的技能
	• vpc_cidr = <enter CIDR block used by VPC> • wavelength_subnet_cidr = <enter CIDR block for the subnet in the Wavelength Zone> • availabilityzone_wavelength = <enter Wavelength Zone name> 4. 儲存 terraform.tfvars 檔案。	
初始化組態。	輸入下列命令來初始化工作目錄。 `terraform init`	DevOps 工程師，Terraform
預覽 Terraform 計劃。	輸入下列命令，將目標狀態與您 AWS 環境的目前狀態進行比較。此命令會產生將設定的資源預覽。 `terraform plan`	DevOps 工程師，Terraform

任務	Description	所需的技能
驗證和部署。	1. 檢閱 Terraform 計劃中的組態變更，並確認您要實作這些變更。 2. 輸入下列命令以套用計劃並建立基礎設施。 `terraform apply` 3. 輸入 `yes` 以繼續。Terraform 會建立組態檔案中宣告的架構。如需架構的詳細資訊，請參閱此模式的目標架構一節。	DevOps 工程師，Terraform

驗證和清除

任務	Description	所需的技能
驗證基礎設施部署。	1. 如果您在 的公有子網路中還沒有 Amazon EC2 執行個體 AWS 區域，請建立一個。如需說明，請參閱啟動 Linux 執行個體或啟動 Windows 執行個體。您將使用此執行個體來測試從 AWS 區域 到 Wavelength 區域的連線。 2. 測試從 中的執行個體 AWS 區域 到 Wavelength 區域中執行個體的連線。如需說明，請參閱 Wavelength 文件中的測試連線。	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
（選用）清除基礎設施。	如果您需要刪除 Terraform 佈建的所有資源，請執行下列動作： 1. 輸入以下命令。 `terraform destroy` 2. 輸入 `yes` 以確認。	DevOps 工程師，Terraform

疑難排解

問題	解決方案
連線至 中的 Amazon EC2 執行個體 AWS 區域。	請參閱連線 Linux 執行個體的故障診斷或連線 Windows 執行個體的故障診斷。
連線至 Wavelength 區域中的 Amazon EC2 執行個體。	請參閱疑難排解在 Wavelength 區域中啟動之 EC2 執行個體的 SSH 或 RDP 連線。
Wavelength 區域中的容量。	請參閱 Wavelength 區域的配額和考量。
從電信業者網路到 的行動或電信業者連線 AWS 區域。	1. 驗證電信業者閘道是否正常運作。請執行下列操作： a. 開啟 Amazon VPC 主控台。 b. 在導覽窗格中，選擇 Your VPCs (您的 VPC)。 c. 選取包含 Wavelength 區域的 VPC。 d. 在詳細資訊窗格中，針對電信業者閘道，確認已連接 值。 2. 驗證連接到 Wavelength 區域中執行個體的任何彈性 IP 地址是否可運作。請執行下列操作： a. 開啟 Amazon EC2 主控台。

問題	解決方案
	b. 在導覽窗格中，選擇執行個體。 c. 在 Wavelength 區域中選取執行個體。 d. 選擇網路標籤。 e. 確認彈性網路界面已連接彈性 IP 地址。 3. 請聯絡電信業者網路支援團隊。

相關資源

- 什麼是 AWS Wavelength？
- AWS Wavelength 運作方式
- 中的彈性 AWS Wavelength

將大量 DNS 記錄遷移至 Amazon Route 53 私有託管區域

Ram Kandaswamy，Amazon Web Services

總結

網路工程師和雲端管理員需要有效率且簡單的方法，將網域名稱系統 (DNS) 記錄新增至 Amazon Route 53 中的私有託管區域。使用手動方法將項目從 Microsoft Excel 工作表複製到 Route 53 主控台中的適當位置很繁瑣且容易出錯。此模式描述自動化方法，可減少新增多個記錄所需的時間和精力。它也為多個託管區域建立提供一組可重複的步驟。

此模式使用 Amazon Simple Storage Service (Amazon S3) 來存放記錄。為了有效率地使用資料，模式使用 JSON 格式，因為它簡單且能夠支援 Python 字典 (`dict` 資料類型）。

> **Note**
>
> 如果您可以從系統產生區域檔案，請考慮改用 Route 53 匯入功能。

先決條件和限制

先決條件

- 包含私有託管區域記錄的 Excel 工作表
- 熟悉不同類型的 DNS 記錄，例如 A 記錄、Name Authority Pointer (NAPTR) 記錄和 SRV 記錄（請參閱支援的 DNS 記錄類型）
- 熟悉 Python 語言及其程式庫

限制

- 模式不會為所有使用案例案例提供廣泛的涵蓋範圍。例如，Change_resource_record_sets 呼叫不會使用 API 的所有可用屬性。
- 在 Excel 工作表中，假設每一列中的值是唯一的。每個完整網域名稱 (FQDN) 的多個值預期會出現在相同的資料列中。如果不正確，您應該修改此模式中提供的程式碼，以執行必要的串連。
- 模式使用適用於 Python 的 AWS 開發套件 (Boto3) 直接呼叫 Route 53 服務。您可以增強程式碼以使用 `create_stack`和 `update_stack`命令的 AWS CloudFormation 包裝函式，並使用 JSON 值填入範本資源。

Architecture

技術堆疊

- 用於路由流量的 Route 53 私有託管區域
- 用於儲存輸出 JSON 檔案的 Amazon S3

工作流程包含這些步驟，如上圖所示，並在 Epics 章節中討論：

1. 將具有記錄集資訊的 Excel 工作表上傳至 S3 儲存貯體。
2. 建立並執行 Python 指令碼，將 Excel 資料轉換為 JSON 格式。
3. 從 S3 儲存貯體讀取記錄並清除資料。
4. 在私有託管區域中建立記錄集。

工具

- Route 53 –Amazon Route 53 是高度可用且可擴展的 DNS Web 服務，可處理網域註冊、DNS 路由和運作狀態檢查。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種物件儲存服務。您可以使用 Amazon S3 隨時從 Web 任何地方存放和擷取任意資料量。

史詩

準備資料以進行自動化

任務	Description	所需的技能
為您的記錄建立 Excel 檔案。	使用您從目前系統匯出的記錄來建立 Excel 工作表，其中包含記錄的必要資料欄，例如完整網域名稱 (FQDN)、記錄類型、存留時間 (TTL) 和值。對於 NAPTR 和 SRV 記錄，值是多個屬性的組合，因此請使用	資料工程師、Excel 技能

任務	Description	所需的技能
	Excel 的 `concat`方法來合併這些屬性。 FqdnN Recor Value TTL e somet A 1.1.1.1 900 .exam org	

驗證工作環境。	在您的 IDE 中，建立 Python 檔案，將 Excel 輸入工作表轉換為 JSON 格式。（您也可以使用 Amazon SageMaker 筆記本來使用 Python 程式碼，而不是 IDE。）
	
	確認您使用的 Python 版本是 3.7 版或更新版本。
	
	`python3 --version`
	
	安裝 pandas 套件。
	
	`pip3 install pandas --user`
		一般 AWS

任務	Description	所需的技能

將 Excel 工作表資料轉換為 JSON。	建立 Python 檔案，其中包含要從 Excel 轉換為 JSON 的下列程式碼。
	
	`import pandas as pd`
	`data=pd.read_excel('./Book1.xls')`
	`data.to_json(path_or_buf='my.json', orient='records')`
	
	其中 Book1是 Excel 工作表的名稱，而 my.json是輸出 JSON 檔案的名稱。
		資料工程師、Python 技能

將 JSON 檔案上傳至 S3 儲存貯體。	上傳 my.json 至 S3 儲存貯體。如需詳細資訊，請參閱 Amazon S3 文件中的建立儲存貯體。	應用程式開發人員

插入記錄

任務	Description	所需的技能

建立私有託管區域。	使用 create_hosted_zone API 和下列 Python 範例程式碼來建立私有託管區域。將參數 hostedZoneName 、vpcRegion 和 取代vpcId為您自己的值。
	
	`import boto3`
	`import random`
	`hostedZoneName ="xxx"`
	`vpcRegion = "us-east-1"`
		雲端架構師、網路管理員、Python 技能

任務	Description	所需的技能

	vpcId="vpc-xxxx"
	route53_client =
	boto3.client('rout
	e53')
	response = route53_c
	lient.create_hoste
	d_zone(
	Name= hostedZon
	eName,
	VPC={
	'VPCRegion:
	vpcRegion,
	'VPCId':
	vpcId
	},
	CallerRef
	erence=str(random.
	random()*100000),
	HostedZon
	eConfig={
	'Comment'
	: "private hosted zone
	created by automatio
	n",
	'PrivateZ
	one': True
	}
	)
	print(response)
	
	您也可以使用基礎設施即程式碼 (IaC) 工具，例如 AWS CloudFormation，將這些步驟取代為使用適當資源和屬性建立堆疊的範本。	

任務	Description	所需的技能
從 Amazon S3 擷取做為字典的詳細資訊。	使用下列程式碼從 S3 儲存貯體讀取，並以 Python 字典的形式取得 JSON 值。 `fileobj = s3_client .get_object( Bucket=bucket_name, Key='my.json' ) filedata = fileobj['Body'].read() contents = filedata.decode('utf-8') json_content=json.loads(contents) print(json_content )` 其中 `json_content` 包含 Python 字典。	應用程式開發人員、Python 技能

任務	Description	所需的技能

清除空格和 Unicode 字元的資料值。	為了確保資料正確性的安全措施，請使用下列程式碼對 中的值執行條紋操作json_content 。此程式碼會移除每個字串前面和結尾的空格字元。它也會使用 replace方法來移除硬（不中斷）空格 (\xa0字元）。
	
	`for item in json_content:`
	`    fqn_name = unicodedata.normalize("NFKD",item["FqdnName"].replace("u'", "'").replace('\xa0', '').strip())`
	`    rec_type = item["RecordType"].replace('\xa0', '').strip()`
	`    res_rec = {`
	
	`'Value': item["Value"].replace('\xa0', '').strip()`
	`}`
		應用程式開發人員、Python 技能

任務	Description	所需的技能
插入記錄。	使用以下程式碼做為上一個for迴圈的一部分。 `change_response = route53_client.change_resource_record_sets( HostedZoneId="xxxxxxxx", ChangeBatch={ 'Comment': 'Created by automation', 'Changes': [ { 'Action': 'UPSERT', 'ResourceRecordSet': { 'Name': fqn_name, 'Type': rec_type, 'TTL': item["TTL"], 'ResourceRecords': res_rec } } ] } )` 其中 xxxxxxx是此史詩中第一個步驟的託管區域 ID。	應用程式開發人員、Python 技能

相關資源

參考

- 透過匯入區域檔案建立記錄 (Amazon Route 53 文件)
- create_hosted_zone 方法 (Boto3 文件)
- change_resource_record_sets 方法 (Boto3 文件)

教學課程和影片

- Python 教學課程 (Python 文件)
- 使用 Amazon Route 53 的 DNS 設計 (YouTube 影片、AWS Online Tech Talks)

當您從 F5 遷移到 AWS 上的 Application Load Balancer 時修改 HTTP 標頭

Sachin Trivedi，Amazon Web Services

總結

當您將使用 F5 Load Balancer 的應用程式遷移至 Amazon Web Services (AWS)，並想要在 AWS 上使用 Application Load Balancer 時，遷移 F5 規則以進行標頭修改是常見的問題。Application Load Balancer 不支援標頭修改，但您可以使用 Amazon CloudFront 做為內容交付網路 (CDN) 和 Lambda@Edge 來修改標頭。

此模式說明必要的整合，並透過使用 AWS CloudFront 和 Lambda@Edge 提供標頭修改的範例程式碼。

先決條件和限制

先決條件

- 內部部署應用程式，使用 F5 負載平衡器搭配使用 取代 HTTP 標頭值的組態`if, else`。如需此組態的詳細資訊，請參閱 F5 產品文件中的 HTTP：：header。

限制

- 此模式適用於 F5 負載平衡器標頭自訂。對於其他第三方負載平衡器，請檢查負載平衡器文件以取得支援資訊。
- 您用於 Lambda@Edge 的 Lambda 函數必須位於美國東部 （維吉尼亞北部） 區域。

Architecture

下圖顯示 AWS 上的架構，包括 CDN 和其他 AWS 元件之間的整合流程。

工具

AWS 服務

- Application Load Balancer - Application Load Balancer 是一種 AWS 全受管負載平衡服務，可在開放系統互連 (OSI) 模型的第七層運作。它平衡多個目標的流量，並支援基於 HTTP 標頭和方法、查詢字串以及主機型或路徑型路由的進階路由請求。

- Amazon CloudFront – Amazon CloudFront 是一種 Web 服務，可加速將靜態和動態 Web 內容，例如 .html、.css、.js 和映像檔案分發給使用者。CloudFront 透過稱為節點的全球資料中心網路提供內容，以降低延遲並改善效能。
- Lambda@Edge – Lambda@Edge 是 AWS Lambda 的延伸，可讓您執行函數來自訂 CloudFront 提供的內容。您可以在美國東部（維吉尼亞北部）區域中編寫函數，然後將函數與 CloudFront 分佈建立關聯，以在全球各地自動複寫程式碼，而無需佈建或管理伺服器。這可減少延遲並改善使用者體驗。

Code

下列範例程式碼提供修改 CloudFront 回應標頭的藍圖。遵循 Epics 區段中的指示來部署程式碼。

```
exports.handler = async (event, context) => {
    const response = event.Records[0].cf.response;
    const headers = response.headers;


    const headerNameSrc = 'content-security-policy';
    const headerNameValue = '*.xyz.com';


    if (headers[headerNameSrc.toLowerCase()]) {
        headers[headerNameSrc.toLowerCase()] = [{
            key: headerNameSrc,
            value: headerNameValue,
        }];
        console.log(`Response header "${headerNameSrc}" was set to ` +
                    `"${headers[headerNameSrc.toLowerCase()][0].value}"`);
    }
    else {
            headers[headerNameSrc.toLowerCase()] = [{
            key: headerNameSrc,
            value: headerNameValue,
            }];
    }
    return response;
};
```

史詩

建立 CDN 分佈

任務	Description	所需的技能
建立 CloudFront Web 分佈。	在此步驟中，您會建立 CloudFront 分佈，以告知 CloudFront 您要從何處交付內容，以及如何追蹤和管理內容交付的詳細資訊。 若要使用主控台建立分佈，請登入 AWS 管理主控台，開啟 CloudFront 主控台，然後遵循 CloudFront 文件中的步驟。	雲端管理員

建立和部署 Lambda@Edge 函數

任務	Description	所需的技能
建立和部署 Lambda@Edge 函數。	您可以使用修改 CloudFront 回應標頭的藍圖來建立 Lambda@Edge 函數。（其他 bluePrints 適用於不同的使用案例；如需詳細資訊，請參閱 CloudFront 文件中的 Lambda@Edge 範例函數。） 若要建立 Lambda@Edge 函數： 1. 登入 AWS 管理主控台，並開啟位於 https://console.aws.amazon.com/lambda/ 的 AWS Lambda 主控台。	AWS 管理員

任務	Description	所需的技能
	2. 請確定您位於美國東部（維吉尼亞北部）區域。CloudFront 藍圖僅適用於此區域。 3. 選擇 Create function (建立函數)。 4. 選擇使用藍圖，然後在藍圖搜尋欄位中輸入 cloudfront。 5. 選擇 cloudfront-modify-response-header 藍圖，然後選擇設定。 6. 在基本資訊頁面上，輸入下列資訊： a. 輸入函數名稱。 b. 針對 Execution role (執行角色)，選擇 Create a new role from AWS policy templates (從 AWS 政策範本建立新角色)。 c. 關聯所需的 AWS Identity and Access Management (IAM) 角色名稱。 7. 選擇 Create function (建立函數)。 8. 在頁面的設計工具區段中，選擇您的函數名稱。 9. 在函數程式碼區段中，將範本程式碼取代為先前在此模式中提供的範例程式碼，在程式碼區段中。	

任務	Description	所需的技能
	10.在範本程式碼中，將 取代`xyz.com`為您的網域名稱。 11選擇儲存。	
部署 Lambda@Edge 函數。	遵循教學課程的步驟 4：在 Amazon CloudFront 文件中建立簡單的 Lambda@Edge 函數，以設定 CloudFront 觸發並部署函數。	AWS 管理員

相關資源

CloudFront 文件

- 自訂原始伺服器的請求和回應行為
- 使用分佈
- Lambda@Edge 範例函數
- 使用 Lambda@Edge 在邊緣自訂
- 教學課程：建立簡單的 Lambda@Edge 函數

為多個 中的傳入網際網路存取建立 Network Access Analyzer 調查結果報告 AWS 帳戶

Mike Virgilio，Amazon Web Services

總結

意外的傳入網際網路存取 AWS 資源可能會對組織的資料周邊造成風險。Network Access Analyzer 是一種 Amazon Virtual Private Cloud (Amazon VPC) 功能，可協助您識別對 Amazon Web Services () 上資源的意外網路存取AWS。您可以使用 Network Access Analyzer 來指定網路存取需求，並識別不符合指定需求的潛在網路路徑。您可以使用 Network Access Analyzer 執行下列動作：

1. 識別可透過網際網路閘道存取網際網路 AWS 的資源。
2. 驗證您的虛擬私有雲端 (VPCs) 是否已適當分割，例如隔離生產和開發環境，以及分隔交易工作負載。

Network Access Analyzer 會分析end-to-end網路連線能力條件，而不只是單一元件。為了判斷資源是否可存取網際網路，Network Access Analyzer 會評估網際網路閘道、VPC 路由表、網路存取控制清單 (ACLs)、彈性網路介面上的公有 IP 地址，以及安全群組。如果任何這些元件阻止網際網路存取，Network Access Analyzer 不會產生問題清單。例如，如果 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體有一個開放的安全群組，允許來自 的流量，`0/0`但執行個體位於無法從任何網際網路閘道路由的私有子網路中，則 Network Access Analyzer 不會產生問題清單。這可提供高逼真度的結果，讓您可以識別可從網際網路真正存取的資源。

當您執行 Network Access Analyzer 時，您可以使用 Network Access Scopes 來指定您的網路存取需求。此解決方案可識別網際網路閘道和彈性網路界面之間的網路路徑。在此模式中，您會在 AWS 帳戶組織中的集中式 中部署解決方案，並由 管理， AWS Organizations並分析 AWS 區域組織中任何帳戶中的所有 帳戶。

此解決方案的設計考量如下：

- AWS CloudFormation 範本可減少在此模式中部署 AWS 資源所需的工作量。
- 您可以在部署時調整 CloudFormation 範本和 naa-script.sh 指令碼中的參數，為您的環境自訂參數。
- Bash 指令碼會自動平行佈建和分析多個帳戶的網路存取範圍。
- Python 指令碼會處理問題清單、擷取資料，然後合併結果。您可以選擇檢閱 CSV 格式或 中的 Network Access Analyzer 調查結果合併報告 AWS Security Hub CSPM。CSV 報告的範例可在此模式的其他資訊???區段中取得。

- 您可以修復問題清單，也可以將問題清單新增至 naa-exclusions.csv 檔案，將其排除在未來的分析之外。

先決條件和限制

先決條件

- AWS 帳戶 用於託管安全服務和工具的 ，以組織的成員帳戶的形式進行管理 AWS Organizations。在此模式中，此帳戶稱為安全帳戶。
- 在安全帳戶中，您必須擁有具有傳出網際網路存取權的私有子網路。如需說明，請參閱 Amazon VPC 文件中的建立子網路。您可以使用 NAT 閘道或界面 VPC 端點來建立網際網路存取。
- 存取 AWS Organizations 管理帳戶或具有 CloudFormation 委派管理員許可的帳戶。如需說明，請參閱 CloudFormation 文件中的註冊委派管理員。
- 啟用 AWS Organizations 和 CloudFormation 之間的受信任存取。如需說明，請參閱 CloudFormation 文件中的使用 啟用受信任存取 AWS Organizations。
- 如果您要將調查結果上傳至 Security Hub CSPM，則必須在帳戶和佈建 Amazon EC2 執行個體 AWS 區域 的帳戶中啟用 Security Hub CSPM。如需詳細資訊，請參閱設定 AWS Security Hub CSPM。

限制

- 由於 Network Access Analyzer 功能的限制，目前不會分析跨帳戶網路路徑。
- 目標 AWS 帳戶 必須以組織身分管理 AWS Organizations。如果您未使用 AWS Organizations，您可以更新您環境的 naa-execrole.yaml CloudFormation 範本和 naa-script.sh 指令碼。反之，您可以提供您要執行指令碼 AWS 帳戶 IDs 和區域的清單。
- CloudFormation 範本旨在將 Amazon EC2 執行個體部署在具有傳出網際網路存取的私有子網路中。AWS Systems Manager 代理程式 (SSM 代理程式） 需要傳出存取權才能到達 Systems Manager 服務端點，而您需要傳出存取權才能複製程式碼儲存庫並安裝相依性。如果您想要使用公有子網路，您必須修改 naa-resources.yaml 範本，將彈性 IP 地址與 Amazon EC2 執行個體建立關聯。

Architecture

目標架構

選項 1：存取 Amazon S3 儲存貯體中的調查結果

圖表顯示下列程序：

1. 如果您手動執行解決方案，使用者會使用 Session Manager 驗證 Amazon EC2 執行個體，然後執行 naa-script.sh 指令碼。此 shell 指令碼會執行步驟 2–7。

   如果您自動執行解決方案，naa-script.sh 指令碼會根據您在 cron 表達式中定義的排程自動啟動。此 shell 指令碼會執行步驟 2–7。如需詳細資訊，請參閱本節結尾的自動化和擴展。
2. Amazon EC2 執行個體會從 Amazon S3 儲存貯體下載最新的 naa-exception.csv 檔案。此檔案稍後會在 Python 指令碼處理排除的程序中使用。
3. Amazon EC2 執行個體會擔任 `NAAEC2Role` AWS Identity and Access Management (IAM) 角色，授予存取 Amazon S3 儲存貯體的許可，並擔任組織中其他帳戶中的 `NAAExecRole` IAM 角色。
4. Amazon EC2 執行個體會在組織的管理帳戶中擔任 `NAAExecRole` IAM 角色，並在組織中產生帳戶清單。
5. Amazon EC2 執行個體會擔任組織成員帳戶中的 `NAAExecRole` IAM 角色（在架構圖中稱為工作負載帳戶），並在每個帳戶中執行安全評估。調查結果會以 JSON 檔案形式存放在 Amazon EC2 執行個體上。
6. Amazon EC2 執行個體使用 Python 指令碼來處理 JSON 檔案、擷取資料欄位，以及建立 CSV 報告。
7. Amazon EC2 執行個體會將 CSV 檔案上傳至 Amazon S3 儲存貯體。
8. Amazon EventBridge 規則會偵測檔案上傳，並使用 Amazon SNS 主題來傳送電子郵件，通知使用者報告已完成。
9. 使用者從 Amazon S3 儲存貯體下載 CSV 檔案。使用者將結果匯入 Excel 範本並檢閱結果。

選項 2：存取 中的問題清單 AWS Security Hub CSPM

圖表顯示下列程序：

1. 如果您手動執行解決方案，使用者會使用 Session Manager 驗證 Amazon EC2 執行個體，然後執行 naa-script.sh 指令碼。此 shell 指令碼會執行步驟 2–7。

   如果您自動執行解決方案，naa-script.sh 指令碼會根據您在 cron 表達式中定義的排程自動啟動。此 shell 指令碼會執行步驟 2–7。如需詳細資訊，請參閱本節結尾的自動化和擴展。
2. Amazon EC2 執行個體會從 Amazon S3 儲存貯體下載最新的 naa-exception.csv 檔案。此檔案稍後會在 Python 指令碼處理排除的程序中使用。

3. Amazon EC2 執行個體會擔任 NAAEC2Role IAM 角色，授予存取 Amazon S3 儲存貯體和在組織中其他帳戶中擔任 NAAExecRole IAM 角色的許可。
4. Amazon EC2 執行個體會在組織的管理帳戶中擔任 NAAExecRole IAM 角色，並在組織中產生帳戶清單。
5. Amazon EC2 執行個體會擔任組織成員帳戶中的 NAAExecRole IAM 角色（在架構圖中稱為工作負載帳戶），並在每個帳戶中執行安全評估。調查結果會以 JSON 檔案形式存放在 Amazon EC2 執行個體上。
6. Amazon EC2 執行個體使用 Python 指令碼來處理 JSON 檔案，並擷取資料欄位以匯入 Security Hub CSPM。
7. Amazon EC2 執行個體會將 Network Access Analyzer 調查結果匯入 Security Hub CSPM。
8. Amazon EventBridge 規則會偵測匯入，並使用 Amazon SNS 主題來傳送電子郵件，通知使用者程序已完成。
9. 使用者會在 Security Hub CSPM 中檢視問題清單。

自動化和擴展

您可以排程此解決方案，以自訂排程自動執行 naa-script.sh 指令碼。若要設定自訂排程，請在 naa-resources.yaml CloudFormation 範本中修改 `CronScheduleExpression` 參數。例如， 的預設值會在每週日午夜`0 0 * * 0`執行解決方案。的值`0 0 * 1-12 0`會在每月第一個星期日的午夜執行解決方案。如需使用 Cron 表達式的詳細資訊，請參閱 Systems Manager 文件中的 Cron 和 Rate 表達式。

如果您想要在部署`NAA-Resources`堆疊之後調整排程，您可以在 中手動編輯 Cron 排程`/etc/cron.d/naa-schedule`。

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如， AWS Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他事件匯流排 AWS 帳戶。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。

- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。
- AWS Security Hub CSPM 提供 中安全狀態的完整檢視 AWS。它還可協助您根據安全產業標準和最佳實務來檢查 AWS 環境。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Systems Manager 可協助您管理在 中執行的應用程式和基礎設施 AWS 雲端。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。此模式使用 Systems Manager 的功能 Session Manager。

程式碼儲存庫

此模式的程式碼可在 GitHub Network Access Analyzer 多帳戶分析儲存庫中使用。程式碼儲存庫包含下列檔案：

- naa-script.sh – 此 bash 指令碼用於 AWS 帳戶平行啟動多個 Network Access Analyzer 分析。如 naa-resources.yaml CloudFormation 範本中所定義，此指令碼會自動部署到 Amazon EC2 執行個體上的 `/usr/local/naa` 資料夾。
- naa-resources.yaml – 您可以使用此 CloudFormation 範本在組織的安全帳戶中建立堆疊。此範本會部署此帳戶所有必要的資源，以支援解決方案。此堆疊必須部署在 naa-execrole.yaml 範本之前。

> **Note**
>
> 如果刪除並重新部署此堆疊，您必須重建`NAAExecRole`堆疊集，才能在 IAM 角色之間重建跨帳戶相依性。

- naa-execrole.yaml – 您可以使用此 CloudFormation 範本建立堆疊集，在組織中的所有帳戶中部署 `NAAExecRole` IAM 角色，包括 管理帳戶。
- naa-processfindings.py – naa-script.sh 指令碼會自動呼叫此 Python 指令碼來處理 Network Access Analyzer JSON 輸出、排除 naa-exclusions.csv 檔案中任何已知良好的資源，然後產生合併結果的 CSV 檔案，或將結果匯入 Security Hub CSPM。

史詩

準備部署

任務	Description	所需的技能
複製程式碼儲存庫。	1. 在命令列界面中，將工作目錄變更為您要存放範例檔案的位置。 2. 輸入以下命令： `git clone https://github.com/aws-samples/network-access-analyzer-multi-account-analysis.git`	AWS DevOps
檢閱範本。	1. 在複製的儲存庫中，開啟 naa-resources.yaml 和 naa-execrole.yaml 檔案。 2. 檢閱這些範本建立的資源，並視需要調整您環境的範本。如需詳細資訊，請參閱 CloudFormation 文件中的使用 範本。 3. 儲存並關閉 naa-resources.yaml 和 naa-execrole.yaml 檔案。	AWS DevOps

建立 CloudFormation 堆疊

任務	Description	所需的技能
佈建安全帳戶中的資源。	使用 naa-resources.yaml 範本，您可以建立 CloudForm	AWS DevOps

任務	Description	所需的技能
	ation 堆疊，以部署安全帳戶中所有必要的資源。如需說明，請參閱 CloudFormation 文件中的建立堆疊。部署此範本時，請注意下列事項： 1. 在指定範本頁面上，選擇範本已就緒，然後上傳 naa-resources.yaml 檔案。 2. 在指定堆疊詳細資訊頁面上的堆疊名稱方塊中，輸入 `NAA-Resources` 。 3. 在參數區段中，輸入下列內容： • `VPCId` – 選取帳戶中的 VPC。 • `SubnetId` — 選取具有網際網路存取權的私有子網路。 **Note** 如果您選取公有子網路，則可能不會指派公有 IP 地址給 Amazon EC2 執行個體，因為 CloudFormation 範本預設不會佈建和連接彈性 IP 地址。 • `InstanceType` – 保留預設執行個體類型。	

任務	Description	所需的技能
	• `InstanceImageId` – 保留預設值。 • `KeyPairName` – 如果您使用 SSH 進行存取，請指定現有金鑰對的名稱。 • `PermittedSSHInbound` – 如果您使用 SSH 進行存取，請指定允許的 CIDR 區塊。如果您不使用 SSH，請保留預設值 `127.0.0.1` 。 • `BucketName` – 預設值為 `naa-<accountID>-<region>` 。您可以視需要修改此項目。如果您指定自訂值，帳戶 ID 和區域會自動附加到指定的值。 • `EmailAddress` – 在分析完成時指定 Amazon SNS 通知的電子郵件地址。 **Note** Amazon SNS 訂閱組態必須在分析完成之前確認，否則不會傳送通知。 • `NAAEC2Role` – 保留預設值，除非您的命名慣例需要此 IAM 角色的不同名稱。	

任務	Description	所需的技能
	• NAAExecRole – 保留預設值，除非部署 naa-execrole.yaml 時將使用其他名稱 • Parallelism – 指定要執行的平行評估數目。 • Regions – 指定 AWS 區域 您要分析的 。 • ScopeNameValue – 指定要指派給範圍的標籤。此標籤用於判斷網路存取範圍。 • ExclusionFile – 指定排除檔案名稱。此檔案中的項目將從問題清單排除。 • FindingsToCSV – 指定是否應將問題清單輸出至 CSV。接受的值為 true和 false。 • FindingsToSecurityHub – 指定是否應將問題清單匯入 Security Hub CSPM。接受的值為 true和 false。 • EmailNotificationsForSecurityHub – 指定將問題清單匯入 Security Hub CSPM 是否應該產生電子郵件通知。接受的值為 true和 false。	

任務	Description	所需的技能
	• `ScheduledAnalysis` – 如果您希望解決方案按照排程自動執行，請輸入 `true`，然後在 `CronScheduleExpression` 參數中自訂排程。如果您不想自動執行解決方案，請輸入 `false`。 • `CronScheduleExpression` – 如果您自動執行解決方案，請輸入 cron 運算式來定義排程。如需詳細資訊，請參閱此模式的架構區段中的自動化和擴展。 1. 在檢閱頁面上，選取下列資源需要功能：【AWS::IAM::Role】，然後選擇建立堆疊。 2. 成功建立堆疊後，在 CloudFormation 主控台的輸出索引標籤上，複製 `NAAEC2Role` Amazon Resource Name (ARN)。您稍後在部署 naa-execrole.yaml 檔案時使用此 ARN。	

任務	Description	所需的技能
在成員帳戶中佈建 IAM 角色。	在 AWS Organizations 管理帳戶或具有 CloudFormation 委派管理員許可的帳戶中，使用 naa-execrole.yaml 範本建立 CloudFormation 堆疊集。堆疊集會在組織中的所有成員帳戶中部署 `NAAExecRole` IAM 角色。如需說明，請參閱 CloudFormation 文件中的建立具有服務受管許可的堆疊集。部署此範本時，請注意下列事項： 1. 在準備範本下，選擇範本已就緒，然後上傳 naa-execrole.yaml 檔案。 2. 在指定 StackSet 詳細資訊頁面上，將堆疊集命名為 `NAA-ExecRole` 。 3. 在參數區段中，輸入下列內容： • `AuthorizedARN` – 輸入您在建立`NAA-Resources` 堆疊時複製的 `NAAEC2Role` ARN。 • `NAARoleName` – 保留預設值，`NAAExecRole` 除非在部署 naa-resources.yaml 檔案時使用另一個名稱。 4. 在 Permissions (許可) 下，選擇 Service-managed permissions (服務管理許可)。	AWS DevOps

任務	Description	所需的技能
	5. 在設定部署選項頁面的部署目標下，選擇部署到組織並接受所有預設值。 **Note** 如果您想要同時將堆疊部署到所有成員帳戶，請將並行帳戶上限和容錯能力設定為高值，例如 100。 6. 在部署區域下，選擇部署 Network Access Analyzer 的 Amazon EC2 執行個體的區域。由於 IAM 資源是全域而非區域性，因此這會在所有作用中區域中部署 IAM 角色。 7. 在檢閱頁面上，選取我確認 AWS CloudFormation 可能會使用自訂名稱建立 IAM 資源，然後選擇建立 StackSet。 8. 監控堆疊執行個體索引標籤（適用於個別帳戶狀態）和操作索引標籤（適用於整體狀態），以判斷部署何時完成。	

任務	Description	所需的技能
在管理帳戶中佈建 IAM 角色。	使用 naa-execrole.yaml 範本，您可以建立 CloudFormation 堆疊，在組織的管理帳戶中部署 `NAAExecRole` IAM 角色。您先前建立的堆疊集不會在管理帳戶中部署 IAM 角色。如需說明，請參閱 CloudFormation 文件中的建立堆疊。部署此範本時，請注意下列事項： 1. 在指定範本頁面上，選擇範本已就緒，然後上傳 naa-execrole.yaml 檔案。 2. 在指定堆疊詳細資訊頁面上的堆疊名稱方塊中，輸入 `NAA-ExecRole` 。 3. 在參數區段中，輸入下列內容： • `AuthorizedARN` – 輸入您在建立`NAA-Resources` 堆疊時複製的 `NAAEC2Role` ARN。 • `NAARoleName` – 保留預設值，`NAAExecRole` 除非在部署 naa-resources.yaml 檔案時使用另一個名稱。 4. 在檢閱頁面上，選取下列資源需要功能：【AWS::IAM::Role】，然後選擇建立堆疊。	AWS DevOps

執行分析

任務	Description	所需的技能

自訂 shell 指令碼。	1. 登入組織中的安全帳戶。
	2. 使用 Session Manager，連線至您先前佈建之 Network Access Analyzer 的 Amazon EC2 執行個體。如需說明，請參閱使用 Session Manager 連線至 Linux 執行個體。如果您無法連線，請參閱此模式的故障診斷一節。
	3. 輸入下列命令以開啟要編輯的 naa-script.sh 檔案：
	`sudo -i`
	`cd /usr/local/naa`
	`vi naa-script.sh`
	4. 視需要檢閱和修改此指令碼中的可調整參數和變數。如需自訂選項的詳細資訊，請參閱指令碼開頭的註解。
	例如，您可以修改指令碼以指定 ID 或要掃描 AWS 區域 的 AWS 帳戶 IDs，或者參考包含這些參數的外部檔案，而不是從管理帳戶取得組織中所有成員帳戶的清單。
	5. 儲存並關閉 naa-script.sh 檔案。
		AWS DevOps

任務	Description	所需的技能

分析目標帳戶。	1. 輸入下列命令：這會執行 naa-script.sh 指令碼：
	`sudo -i`
	`cd /usr/local/naa`
	`screen`
	`./naa-script.sh`
	注意下列事項：
	• `screen` 命令允許指令碼在連線逾時或您失去主控台存取權時繼續執行。
	• 掃描開始後，您可以按下 Ctrl+A D 強制分離畫面。畫面分離，您可以在分析進行時關閉執行個體連線。
	• 若要繼續分離的工作階段，請連線至執行個體，輸入 `sudo -i`，然後輸入 `screen -r`。
	2. 監控輸出是否有任何錯誤，以確保指令碼正常運作。如需範例輸出，請參閱此模式的其他資訊區段。
	3. 等候分析完成。如果您已設定電子郵件通知，當結果上傳到 Amazon S3 儲存貯體或匯入 Security Hub CSPM 時，您會收到一封電子郵件。
		AWS DevOps

任務	Description	所需的技能
選項 1 – 從 Amazon S3 儲存貯體擷取結果。	1. 從儲存naa-<accountID>-<region> 貯體下載 CSV 檔案。如需說明，請參閱 Amazon S3 文件中的下載物件。 Amazon S3 2. 從 Amazon S3 儲存貯體刪除 CSV 檔案。這是成本最佳化的最佳實務。如需說明，請參閱 Amazon S3 文件中的刪除物件。	AWS DevOps
選項 2 – 在 Security Hub CSPM 中檢閱結果。	1. 開啟 Security Hub CSPM 主控台。 2. 從導覽窗格中選擇問題清單。 3. 檢閱 Network Access Analyzer 調查結果。如需說明，請參閱 Security Hub CSPM 文件中的檢視問題清單和詳細資訊。 **Note** 您可以透過新增標題以篩選條件開頭並輸入 來搜尋問題清單Network Access Analyzer。	AWS DevOps

修復和排除問題清單

任務	Description	所需的技能
修復問題清單。	修復您要解決的任何問題清單。如需如何在 AWS 身分、資源和網路周圍建立周邊的詳細資訊和最佳實務，請參閱在上建立資料周邊 AWS (AWS 白皮書）。	AWS DevOps
排除具有已知良好網路路徑的資源。	如果 Network Access Analyzer 產生應可從網際網路存取的資源調查結果，則您可以將這些資源新增至排除清單。下次 Network Access Analyzer 執行時，不會產生該資源的問題清單。 1. 導覽至 `/usr/local/naa`，然後開啟 naa-script.sh 指令碼。請記下 `S3_EXCLUSION_FILE` 變數的值。 2. 如果`S3_EXCLUSION_FILE` 變數的值為 `true`，請從儲存`naa-<accountID>-<region>` 貯體下載 naa-exclusions.csv 檔案。如需說明，請參閱 Amazon S3 文件中的下載物件。Amazon S3 如果`S3_EXCLUSION_FILE` 變數的值為 `false`，請導覽至 ，`/usr/`	AWS DevOps

任務	Description	所需的技能

	`local/naa` 然後開啟 naa-exclusions.csv 檔案。
	
	Note
	如果`S3_EXCLUSION_FILE` 變數的值為 `false`，則指令碼會使用排除檔案的本機版本。如果您稍後將值變更為 `true`，則指令碼會使用 Amazon S3 儲存貯體中的 檔案覆寫本機版本。
	
	3. 在 naa-exclusions.csv 檔案中，輸入您要排除的資源。每行輸入一個資源，並使用下列格式。
	
	`<resource_id>,<secgroup_id>,<sgrule_cidr>,<sgrule_portrange>,<sgrule_protocol>`
	
	以下是範例資源。
	
	`eni-1111aaaaa2222bbbb,sg-3333ccccc4444dddd,0.0.0.0/0,80 to 80,tcp`
	
	4. 儲存並關閉 naa-exclusions.csv 檔案。	

任務	Description	所需的技能
	5. 如果您從 Amazon S3 儲存貯體下載 naa-exclusions.csv 檔案，請上傳新版本。如需說明，請參閱 Amazon S3 文件中的上傳物件。 Amazon S3	

（選用）更新 naa-script.sh 指令碼

任務	Description	所需的技能

更新 naa-script.sh 指令碼。	如果您想要將 naa-script.sh 指令碼更新至儲存庫中的最新版本，請執行下列動作：
	1. 使用 Session Manager 連線至 Amazon EC2 執行個體。如需說明，請參閱使用 Session Manager 連線至 Linux 執行個體。
	2. 輸入以下命令：
	`sudo -i`
	3. 導覽至 naa-script.sh 指令碼目錄：
	`cd /usr/local/naa`
	4. 輸入下列命令來存放本機指令碼，讓您可以將自訂變更合併到最新版本：
	`git stash`
		AWS DevOps

任務	Description	所需的技能

	5. 輸入下列命令以取得指令碼的最新版本：
	`git pull`
	6. 輸入下列命令，將自訂指令碼與最新版本的指令碼合併：
	`git stash pop`	

(選用) 清除

任務	Description	所需的技能
刪除所有部署的資源。	您可以在帳戶中保留部署的資源。 如果您想要取消佈建所有資源，請執行下列動作： 1. 刪除管理帳戶中佈建的`NAA-ExecRole` 堆疊。如需說明，請參閱 CloudFormation 文件中的刪除堆疊。 2. 刪除組織管理帳戶或委派管理員帳戶中佈建的`NAA-ExecRole` 堆疊集。如需說明，請參閱 CloudFormation 文件中的刪除堆疊集。 3. 刪除 `naa-<accountID>-<region>` Amazon S3 儲存貯體中的所有物件。如需說明，請參閱 Amazon S3 文件中的刪除物件。	AWS DevOps

任務	Description	所需的技能
	4. 刪除在安全帳戶中佈建的NAA-Resources 堆疊。如需說明，請參閱 CloudFormation 文件中的刪除堆疊。	

疑難排解

問題	解決方案
無法使用 Session Manager 連線至 Amazon EC2 執行個體。	SSM Agent 必須能夠與 Systems Manager 端點通訊。請執行下列操作： 1. 驗證部署 Amazon EC2 執行個體的子網路具有網際網路存取權。 2. 重新啟動 Amazon EC2 執行個體。
部署堆疊集時，CloudFormation 主控台會提示您使用 Enable trusted access with AWS Organizations to use service-managed permissions 。	這表示 AWS Organizations 和 CloudFormation 之間尚未啟用受信任的存取。部署服務受管堆疊集需要信任的存取權。選擇按鈕以啟用受信任的存取。如需詳細資訊，請參閱 CloudFormation 文件中的啟用受信任存取。

相關資源

- 新增 – Amazon VPC Network Access Analyzer (AWS 部落格文章）
- AWS re：Inforce 2022 - 驗證 AWS (NIS202)（影片）上的有效網路存取控制
- 示範 - 使用 Network Access Analyzer 進行全組織的網際網路傳入資料路徑分析 （影片）

其他資訊

範例主控台輸出

以下範例顯示產生目標帳戶清單和分析目標帳戶的輸出。

```
[root@ip-10-10-43-82 naa]# ./naa-script.sh
download: s3://naa-<account ID>-us-east-1/naa-exclusions.csv to ./naa-exclusions.csv

AWS Management Account: <Management account ID>

AWS Accounts being processed...
<Account ID 1> <Account ID 2> <Account ID 3>

Assessing AWS Account: <Account ID 1>, using Role: NAAExecRole
Assessing AWS Account: <Account ID 2>, using Role: NAAExecRole
Assessing AWS Account: <Account ID 3>, using Role: NAAExecRole
Processing account: <Account ID 1> / Region: us-east-1
Account: <Account ID 1> / Region: us-east-1 - Detecting Network Analyzer scope...
Processing account: <Account ID 2> / Region: us-east-1
Account: <Account ID 2> / Region: us-east-1 - Detecting Network Analyzer scope...
Processing account: <Account ID 3> / Region: us-east-1
Account: <Account ID 3> / Region: us-east-1 - Detecting Network Analyzer scope...
Account: <Account ID 1> / Region: us-east-1 - Network Access Analyzer scope detected.
Account: <Account ID 1> / Region: us-east-1 - Continuing analyses with Scope ID.
 Accounts with many resources may take up to one hour
Account: <Account ID 2> / Region: us-east-1 - Network Access Analyzer scope detected.
Account: <Account ID 2> / Region: us-east-1 - Continuing analyses with Scope ID.
 Accounts with many resources may take up to one hour
Account: <Account ID 3> / Region: us-east-1 - Network Access Analyzer scope detected.
Account: <Account ID 3> / Region: us-east-1 - Continuing analyses with Scope ID.
 Accounts with many resources may take up to one hour
```

CSV 報告範例

下圖是 CSV 輸出的範例。

在多帳戶 AWS 環境中設定混合網路的 DNS 解析

Anvesh Koganti，Amazon Web Services

總結

此模式提供全方位解決方案，可在包含多個 Amazon Web Services (AWS) 帳戶的混合網路環境中設定 DNS 解析。它可透過 Amazon Route 53 Resolver 端點在內部部署網路和 AWS 環境之間啟用雙向 DNS 解析。模式提供兩種解決方案，可在多帳戶集中式架構中啟用 DNS 解析：

- 基本設定不使用 Route 53 Profiles。它有助於最佳化低複雜度的中小型部署的成本。
- 增強型設定使用 Route 53 Profiles 來簡化操作。它最適合大型或更複雜的 DNS 部署。

> **Note**
>
> 在實作之前，請檢閱限制一節，了解服務限制和配額。當您做出決策時，請考慮管理開銷、成本、營運複雜性和團隊專業知識等因素。

先決條件和限制

先決條件

- 跨共用服務和工作負載帳戶部署 Amazon Virtual Private Cloud (Amazon VPC) 的 AWS 多帳戶環境（最好遵循帳戶結構的AWS 最佳實務，透過 AWS Control Tower 設定）。
- 現場部署網路與 AWS 環境之間的現有混合連線 (AWS Direct Connect 或 AWS Site-to-Site VPN)。
- Amazon VPC 對等 AWS Transit Gateway互連，或適用於 VPCs AWS 雲端 WAN。（應用程式流量需要此連線。 DNS 解析不需要它。 DNS 解析的運作與 VPCs.)
- 在內部部署環境中執行的 DNS 伺服器。

限制

- Route 53 Resolver 端點、規則和設定檔是區域性建構， AWS 區域 可能需要針對全球組織複寫多個 。
- 如需 Route 53 Resolver、私有託管區域和設定檔的服務配額完整清單，請參閱 Route 53 文件中的配額。

Architecture

目標技術堆疊

- Route 53 傳出和傳入端點
- 條件式轉送的 Route 53 Resolver 規則
- AWS Resource Access Manager (AWS RAM)
- Route 53 私有託管區域

目標架構

傳出和傳入端點

下圖顯示從 AWS 到內部部署的 DNS 解析流程。這是網域託管在內部部署的傳出解析度的連線設定。以下是設定此設定所涉及程序的高階概觀。如需詳細資訊，請參閱 Epics 區段。

1. 在共用服務 VPC 中部署傳出 Route 53 Resolver 端點。
2. 在共用服務帳戶中為內部部署託管的網域建立 Route 53 Resolver 規則 （ 轉送規則 ）。
3. 在需要解析內部部署託管網域的其他 帳戶中，共用規則並將其與 VPCs 建立關聯。這可以根據您的使用案例以不同的方式完成，如本節稍後所述。

設定連線後，傳出解析中涉及的步驟如下：

1. Amazon Elastic Compute Cloud (Amazon EC2) 執行個體會將 的 DNS 解析請求`db.onprem.example.com`傳送至 VPC+2 地址的 VPC Route 53 Resolver。
2. Route 53 Resolver 會檢查 Resolver 規則，並使用傳出端點將請求轉送至內部部署 DNS 伺服器 IPs。
3. 傳出端點會將請求轉送至內部部署 DNS IPs。流量會經過共用服務 VPC 與內部部署資料中心之間已建立的混合網路連線。
4. 內部部署 DNS 伺服器會回應傳出端點，然後將回應轉送回 VPC 的 Route 53 Resolver。Resolver 會將回應傳回 EC2 執行個體。

下圖顯示從內部部署環境到 的 DNS 解析流程 AWS。這是網域託管所在之傳入解析度的連線設定 AWS。以下是設定此設定所涉及程序的高階概觀。如需詳細資訊，請參閱 Epics 區段。

1. 在共用服務 VPC 中部署傳入解析程式端點。
2. 在共用服務帳戶中建立私有託管區域（集中式方法）。
3. 將私有託管區域與共用服務 VPC 建立關聯。共用這些區域並將其與跨帳戶 VPCs建立關聯，以進行 VPC-to-VPC DNS 解析。這可以根據您的使用案例以不同的方式完成，如本節稍後所述。

設定連線後，傳入解析中涉及的步驟如下：

1. 內部部署資源會將 的 DNS 解析請求`ec2.prod.aws.example.com`傳送至內部部署 DNS 伺服器。
2. 內部部署 DNS 伺服器會透過混合網路連線，將請求轉送至共用服務 VPC 中的傳入解析程式端點。
3. 在 VPC Route 53 Resolver 的協助下，傳入解析程式端點會在相關聯的私有託管區域中查詢請求，並取得適當的 IP 地址。
4. 這些 IP 地址會傳回現場部署 DNS 伺服器，將回應傳回現場部署資源。

此組態可讓內部部署資源透過傳入端點將查詢路由到適當的 AWS 私有託管區域，來解析私有網域名稱。在此架構中，私有託管區域集中在共用服務 VPC 中，允許單一團隊進行中央 DNS 管理。這些區域可以與許多 VPCs相關聯，以解決 VPC-to-VPC解析使用案例。或者，您可能想要將 DNS 網域擁有權和管理委派給每個網域 AWS 帳戶。在這種情況下，每個帳戶都會管理自己的私有託管區域，並將每個區域與中央共用服務 VPC 建立關聯，以便與內部部署環境統一解析。此分散式方法超出此模式的範圍。如需詳細資訊，請參閱 Amazon VPCs 混合雲端 DNS 選項白皮書中的跨多個帳戶和 VPC 擴展 DNS 管理。

當您使用解析程式端點建立基本 DNS 解析流程時，您需要判斷如何管理解析程式規則和跨 私有託管區域的共用和關聯 AWS 帳戶。您可以透過兩種方式處理此問題：透過使用 AWS RAM 共用解析程式規則和直接私有託管區域關聯的自我管理共用，如基本設定章節所述，或透過 Route 53 設定檔，如增強型設定章節所述。選擇取決於組織的 DNS 管理偏好設定和操作需求。下列架構圖說明擴展的環境，其中包含跨不同帳戶的多個 VPCs，代表典型的企業部署。

基本設定

在基本設定中，多帳戶 AWS 環境中混合 DNS 解析的實作會使用 AWS RAM 來共用解析程式轉送規則和私有託管區域關聯，以管理內部部署 AWS 和資源之間的 DNS 查詢。此方法使用共用服務 VPC 中連線至內部部署網路的集中式 Route 53 Resolver 端點，以有效率地處理傳入和傳出 DNS 解析。

- 對於傳出解決方案，解析程式轉送規則會在共用服務帳戶中建立，然後使用 AWS 帳戶 與其他 共用 AWS RAM。此共用僅限於相同區域內的帳戶。然後，目標帳戶可以將這些規則與其 VPCs建立關聯，並讓這些 VPCs中的資源解析內部部署網域名稱。
- 對於傳入解析度，私有託管區域會在共用服務帳戶中建立，並與共用服務 VPC 相關聯。這些區域接著可以使用 Route 53 API、 AWS SDKs 或 AWS Command Line Interface () 與其他VPCs 建立關聯 AWS CLI。然後，相關聯 VPCs 中的資源可以解析私有託管區域中定義的 DNS 記錄，這會在整個 AWS 環境中建立統一的 DNS 檢視。

下圖顯示此基本設定中的 DNS 解析流程。

當您在有限的規模上使用 DNS 基礎設施時，此設定可正常運作。不過，隨著環境的成長，管理 可能會變得具有挑戰性。管理私有託管區域和解析程式規則的共用和關聯方式的操作開銷會隨著規模VPCs而個別大幅增加。此外，每個私有託管區域的 300 個 VPC 關聯限制等服務配額，可能會成為大規模部署的限制因素。增強型設定可解決這些挑戰。

增強型設定

Route 53 Profiles 提供簡化的解決方案，可跨多個混合網路管理 DNS 解析 AWS 帳戶。除了個別管理私有託管區域和解析程式規則之外，您可以將 DNS 組態分組為單一容器，以便在區域中VPCs 和帳戶之間輕鬆共用和套用。此設定會在共用服務 VPC 中維護集中式 Resolver 端點架構，同時大幅簡化 DNS 組態的管理。

下圖顯示增強型設定中的 DNS 解析流程。

Route 53 Profiles 可讓您將私有託管區域關聯、解析程式轉送規則和 DNS 防火牆規則封裝為單一可共用的單位。您可以在共用服務帳戶中建立設定檔，並使用 與成員帳戶共用 AWS RAM。當設定檔共用並套用至目標 VPCs時，服務會自動處理所有必要的關聯和組態。這可大幅降低 DNS 管理的操作負荷，並為不斷成長的環境提供絕佳的可擴展性。

自動化和擴展

使用基礎設施做為程式碼 (IaC) 工具，例如 CloudFormation 或 Terraform，以自動佈建和管理 Route 53 Resolver 端點、規則、私有託管區域和設定檔。將 DNS 組態與持續整合和持續交付 (CI/CD) 管道整合，以實現一致性、可重複性和快速更新。

工具

AWS 服務

- AWS Resource Access Manager (AWS RAM) 可協助您安全地跨 共用資源 AWS 帳戶 ，以減少營運開銷並提供可見性和可稽核性。
- Amazon Route 53 Resolver 會以遞迴方式回應來自 AWS 資源的 DNS 查詢，且預設可在所有 VPCs 中使用。您可以建立解析程式端點和條件式轉送規則，以解析內部部署資料中心和 VPCs 之間的 DNS 命名空間。
- Amazon Route 53 私有託管區域是容器，其中包含您希望 Route 53 如何回應網域及其子網域的 DNS 查詢的相關資訊。
- Amazon Route 53 Profiles 可讓您以 AWS 帳戶 簡化的方式跨多個 VPCs 套用和管理 DNS 相關的 Route 53 組態。

最佳實務

本節提供最佳化 Route 53 Resolver 的一些最佳實務。這些代表 Route 53 最佳實務的子集。如需完整清單，請參閱 Amazon Route 53 的最佳實務。

使用解析程式端點避免迴圈組態

- 仔細規劃 VPC 關聯，設計您的 DNS 架構以防止遞迴路由。當 VPC 託管傳入端點時，請避免將其與可建立循環參考的解析程式規則建立關聯。
- 當您跨帳戶共用 DNS 資源時， AWS RAM 請策略性地使用 ，以維持乾淨的路由路徑。

如需詳細資訊，請參閱 Route 53 文件中的使用解析程式端點避免迴圈組態。

擴展解析程式端點

- 對於需要每秒大量查詢 (QPS) 的環境，請注意端點中的每個 ENI 限制為 10，000 個 QPS。您可以將更多 ENIs新增至端點，以擴展 DNS QPS。
- Amazon CloudWatch 提供 `InboundQueryVolume`和 `OutboundQueryVolume`指標 （請參閱 CloudWatch 文件)。建議您設定監控規則，以便在閾值超過特定值 （例如 10，000 QPS 的 80%) 時提醒您。
- 設定 Resolver 端點的狀態安全群組規則，以防止連線追蹤限制在大量流量期間造成 DNS 查詢限流。若要進一步了解連線追蹤如何在安全群組中運作，請參閱 Amazon EC2 文件中的 Amazon EC2 安全群組連線追蹤。 Amazon EC2

如需詳細資訊，請參閱 Route 53 文件中的解析程式端點擴展。

為解析程式端點提供高可用性

- 在至少兩個可用區域中建立 IP 地址的傳入端點以進行備援。
- 佈建其他網路介面，以確保在維護或流量激增期間可用性。

如需詳細資訊，請參閱 Route 53 文件中的解析程式端點高可用性。

史詩

部署 Route 53 Resolver 端點

任務	Description	所需的技能
部署傳入端點。	Route 53 Resolver 使用傳入端點從內部部署 DNS 解析程式接收 DNS 查詢。如需說明，請參閱 Route 53 文件中的轉送傳入 DNS 查詢到您的 VPCs。記下傳入端點 IP 地址。	AWS 管理員、雲端管理員
部署傳出端點。	Route 53 Resolver 使用傳出端點將 DNS 查詢傳送至內部部署 DNS 解析程式。如需說明，請參閱 Route 53 文件中的轉送傳出 DNS 查詢到您的網路。記下輸出端點 ID。	AWS 管理員、雲端管理員

設定和共用 Route 53 私有託管區域

任務	Description	所需的技能
為託管的網域建立私有託管區域 AWS。	此區域會保留 AWS託管網域（例如 `prod.aws.example.com`）中資源的 DNS 記錄，這些記錄應從內部部署環境解析。如需說明，請參閱 Route 53 文件中的建立私有託管區域。	AWS 管理員、雲端管理員

任務	Description	所需的技能
	建立私有託管區域時，您必須將 VPC 與相同帳戶擁有的託管區域建立關聯。為此選取共用服務 VPC。	
基本設定：將私有託管區域與其他帳戶中VPCs 建立關聯。	如果您使用的是基本設定（請參閱架構一節）： 若要啟用成員帳戶 VPCs來解析此私有託管區域中的 DNS 記錄，您必須將 VPCs與託管區域建立關聯。您必須授權關聯，然後以程式設計方式建立關聯。如需說明，請參閱 Route 53 文件中的關聯 Amazon VPC 和您使用不同建立的私有託管區域 AWS 帳戶。	AWS 管理員、雲端管理員

任務	Description	所需的技能
增強型設定：設定和共用 Route 53 設定檔。	如果您使用的是增強型設定（請參閱架構一節）： 1. 建立 Route 53 設定檔，並將相關的私有託管區域與其建立關聯。如需說明，請參閱 Route 53 文件中的建立 Route 53 設定檔。 2. 使用 與成員帳戶 AWS RAM 共用設定檔，然後將共用設定檔與目標 VPCs建立關聯。如需說明，請參閱 Route 53 文件中的共享 Route 53 設定檔和將 Route 53 設定檔與 VPCs 建立關聯。 **Note** 視組織的結構和 DNS 需求而定，您可能需要為不同的帳戶或工作負載建立和管理多個設定檔。	AWS 管理員、雲端管理員

設定和共用 Route 53 Resolver 轉送規則

任務	Description	所需的技能
為內部部署託管的網域建立轉送規則。	此規則會指示 Route 53 Resolver 將內部部署網域（例如 `onprem.example.com`）的任何 DNS 查詢轉送至內部部	AWS 管理員、雲端管理員

任務	Description	所需的技能
	署 DNS 解析程式。若要建立此規則，您需要內部部署 DNS 解析程式的 IP 地址和傳出端點 ID。如需說明，請參閱 Route 53 文件中的建立轉送規則。	
基本設定：與其他帳戶中VPCs共用和建立轉送規則的關聯。	如果您使用的是基本設定： 若要讓轉送規則生效，您必須與其他帳戶中VPCs 共用並建立規則的關聯。Route 53 Resolver 接著會在解析網域時考慮規則。如需說明，請參閱 Route 53 文件中的與其他 共用解析程式規則 AWS 帳戶 和使用共用規則，以及將轉送規則與 VPC 建立關聯。	AWS 管理員、雲端管理員

任務	Description	所需的技能
增強型設定：設定和共用 Route 53 設定檔。	如果您使用的是增強型設定： 1. 如果您已在先前的步驟中建立 Route 53 設定檔，則可以使用相同的設定檔。如果沒有，請建立 Route 53 設定檔，並將相關的解析程式轉送規則與其建立關聯。如需說明，請參閱 Route 53 文件中的建立 Route 53 設定檔。 2. 使用 與成員帳戶 AWS RAM 共用設定檔，然後將共用設定檔與目標 VPCs建立關聯。如需說明，請參閱 Route 53 文件中的共用 Route 53 設定檔和將 Route 53 設定檔與 VPCs 建立關聯。 **Note** 視組織的結構和 DNS 需求而定，您可能需要為不同的帳戶或工作負載建立和管理多個設定檔。	AWS 管理員、雲端管理員

設定內部部署 DNS 解析程式以進行 AWS 整合

任務	Description	所需的技能
在內部部署 DNS 解析程式中設定條件式轉送。	若要 AWS 從內部部署環境傳送至 以進行解析的 DNS 查詢，您必須在內部部署 DNS 解析程式中設定條件式轉送，以指向傳入端點 IP 地址。這會指示 DNS 解析程式將所有 AWS託管網域的 DNS 查詢（例如，針對 `prod.aws.example.com` ）轉送至傳入端點 IP 地址，以供 Route 53 Resolver 解析。	網路管理員

在混合環境中驗證end-to-end解析

任務	Description	所需的技能
測試從 AWS 到內部部署環境的 DNS 解析。	從 VPC 中具有與其相關聯之轉送規則的執行個體，執行內部部署託管網域的 DNS 查詢（例如，針對 `db.onprem.example.com` ）。	網路管理員
從現場部署環境測試 DNS 解析 AWS。	從內部部署伺服器，執行託管網域 AWS的 DNS 解析（例如，針對 `ec2.prod.aws.example.com` ）。	網路管理員

相關資源

- Amazon VPC 的混合雲端 DNS 選項 (AWS 白皮書)
- 使用私有託管區域 (Route 53 文件)

- Route 53 Resolver 入門(Route 53 文件）
- 使用 Route 53 Resolver（部落格文章）簡化多帳戶環境中的 DNS 管理AWS
- 使用具有多個 VPCs和 （部落格文章）的 Amazon Route 53 設定檔統一 DNS 管理 AWS 帳戶AWS
- 將多帳戶 DNS 環境遷移至 Amazon Route 53 Profiles (AWS 部落格文章）
- 將 Amazon Route 53 Profiles 用於可擴展的多帳戶 AWS 環境 (AWS 部落格文章）

確認 ELB 負載平衡器需要終止 TLS

Priyanka Chaudhary，Amazon Web Services

總結

在 Amazon Web Services (AWS) 雲端上，Elastic Load Balancing (ELB) 會自動將傳入的應用程式流量分散到多個目標，例如 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、容器、IP 地址和 AWS Lambda 函數。負載平衡器使用接聽程式來定義負載平衡器用來接受來自使用者的流量的連接埠和通訊協定。Application Load Balancer 在應用程式層進行路由決策，並使用 HTTP/HTTPS 通訊協定。Classic Load Balancer 會在傳輸層、使用 TCP 或 Secure Sockets Layer (SSL) 通訊協定，或使用 HTTP/HTTPS 在應用程式層進行路由決策。

此模式提供安全性控制，可檢查 Application Load Balancer 和 Classic Load Balancer 的多種事件類型。叫用函數時，AWS Lambda 會檢查事件，並確保負載平衡器合規。

函數會在下列 API 呼叫上啟動 Amazon CloudWatch Events 事件：CreateLoadBalancer、CreateLoadBalancerListeners、DeleteLoadBalancerListeners、CreateLoadBalar 和 ModifyListener。當事件偵測到其中一個 APIs時，它會呼叫執行 Python 指令碼的 AWS Lambda。Python 指令碼會評估 以查看接聽程式是否包含 SSL 憑證，以及套用的政策是否使用 Transport Layer Security (TLS)。如果 SSL 政策確定為 TLS 以外的任何項目，則函數會傳送 Amazon Simple Notification Service (Amazon SNS) 通知給使用者，其中包含相關資訊。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶

限制

- 除非更新負載平衡器接聽程式，否則此安全控制不會檢查現有的負載平衡器。
- 此安全控制是區域性的。您必須將其部署到您要監控的每個 AWS 區域。

Architecture

目標架構

自動化和擴展

- 如果您使用的是 AWS Organizations，則可以使用 AWS Cloudformation StackSets，將此範本部署到您要監控的多個帳戶中。

工具

AWS 服務

- AWS CloudFormation – AWS CloudFormation 可協助您建立模型和設定 AWS 資源、快速一致地佈建資源，以及在整個生命週期中管理資源。您可以使用範本來描述資源及其相依性，並將它們一起啟動和設定為堆疊，而不是個別管理資源。
- Amazon CloudWatch Events – Amazon CloudWatch Events 提供近乎即時的系統事件串流，說明 AWS 資源的變更。
- AWS Lambda – AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種高度可擴展的物件儲存服務，可用於各種儲存解決方案，包括網站、行動應用程式、備份和資料湖。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 會協調和管理發佈者和用戶端之間的訊息傳遞或傳送，包括 Web 伺服器和電子郵件地址。訂閱者會收到發佈到所訂閱主題的所有訊息，且某一主題的所有訂閱者均會收到相同訊息。

Code

此模式包含下列附件：

- `ELBRequirestlstermination.zip` – 用於安全控制的 Lambda 程式碼。
- `ELBRequirestlstermination.yml` – 設定事件和 Lambda 函數的 CloudFormation 範本。

史詩

設定 S3 儲存貯體

任務	Description	所需的技能
定義 S3 儲存貯體。	在 Amazon S3 主控台上，選擇或建立 S3 儲存貯體以託管 Lambda 程式碼 .zip 檔案。此 S3 儲存貯體必須與您要評估的	雲端架構師

任務	Description	所需的技能
	負載平衡器位於相同的 AWS 區域。S3 儲存貯體名稱全域唯一，且命名空間由所有 AWS 帳戶共用。S3 儲存貯體名稱不能包含正斜線。	
上傳 Lambda 程式碼。	將附件區段中提供的 Lambda 程式碼 (`ELBRequirestlstermination.zip` 檔案）上傳至 S3 儲存貯體。	雲端架構師

部署 CloudFormation 範本

任務	Description	所需的技能
啟動 AWS CloudFormation 範本。	在與 S3 儲存貯體相同的 AWS 區域中開啟 AWS CloudFormation 主控台，並部署連接的範本 `ELBRequirestlstermination.yml` 。如需部署 AWS CloudFormation 範本的詳細資訊，請參閱 CloudFormation 文件中的在 AWS CloudFormation 主控台上建立堆疊。 CloudFormation	雲端架構師
完成範本中的參數。	當您啟動範本時，系統會提示您輸入下列資訊： • S3 儲存貯體：指定您在第一個特徵中建立或選取的儲存貯體。這是您上傳連接的 Lambda 程式碼 (`ELBRequirestlsterm`	雲端架構師

任務	Description	所需的技能
	ination.zip 檔案）的位置。 • S3 金鑰：指定 S3 儲存貯體中 Lambda .zip 檔案的位置（例如 ELBRequirestlstermination.zip 或 controls/ELBRequirestlstermination.zip）。請勿包含正斜線。 • 通知電子郵件：提供您要接收 Amazon SNS 通知的作用中電子郵件地址。 • Lambda 記錄層級：指定 Lambda 函數的記錄層級和頻率。使用資訊記錄有關進度的詳細資訊訊息、仍允許部署繼續的錯誤事件錯誤，以及潛在有害情況的警告。	

確認訂閱

任務	Description	所需的技能
確認訂閱。	當 CloudFormation 範本成功部署時，它會傳送訂閱電子郵件到您提供的電子郵件地址。您必須確認此電子郵件訂閱，才能開始接收違規通知。	雲端架構師

相關資源

- 在 AWS CloudFormation 主控台上建立堆疊 (AWS CloudFormation 文件）

- 什麼是 AWS Lambda？ (AWS Lambda 文件）
- 什麼是 Classic Load Balancer？ (ELB 文件）
- 什麼是 Application Load Balancer？ (ELB 文件）

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Splunk 檢視 AWS Network Firewall 日誌和指標

Ivo Pinto，Amazon Web Services

摘要

許多組織使用 Splunk Enterprise 做為來自不同來源的日誌和指標的集中彙整和視覺化工具。 此模式可協助您設定 Splunk 使用 Splunk Add-On for AWS 從 Amazon CloudWatch Logs 擷取 AWS Network Firewall 日誌和指標。 Amazon CloudWatch

若要達成此目的，您可以建立唯讀 AWS Identity and Access Management (IAM) 角色。Splunk Add-On for AWS 使用此角色存取 CloudWatch。您可以設定 Splunk Add-On for AWS 以從 CloudWatch 擷取指標和日誌。最後，您可以從擷取的日誌資料和指標，在 Splunk 中建立視覺化效果。

先決條件和限制

先決條件

- Splunk 帳戶
- Splunk Enterprise 執行個體，8.2.2 版或更新版本
- 作用中的 AWS 帳戶
- Network Firewall，設定並設定 將日誌傳送至 CloudWatch Logs

限制

- Splunk Enterprise 必須部署為 AWS 雲端中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體叢集。
- AWS 中國區域不支援使用自動探索的 Amazon EC2 IAM 角色收集資料。

Architecture

此圖展示了以下要點：

1. Network Firewall 會將日誌發佈至 CloudWatch Logs。
2. Splunk Enterprise 從 CloudWatch 擷取指標和日誌。

若要在此架構中填入範例指標和日誌，工作負載會產生通過 Network Firewall 端點的流量，以前往網際網路。這是透過使用路由表來實現的。 雖然此模式使用單一 Amazon EC2 執行個體做為工作負載，但只要 Network Firewall 設定為將日誌傳送至 CloudWatch Logs，此模式就可以套用至任何架構。

此架構也會在另一個虛擬私有雲端 (VPC) 中使用 Splunk Enterprise 執行個體。不過，Splunk 執行個體可以位於另一個位置，例如與工作負載相同的 VPC，只要它可以到達 CloudWatch APIs。

工具

AWS 服務

- Amazon CloudWatch Logs 可協助您集中所有系統、應用程式和 AWS 服務的日誌，以便您可以監控日誌並將其安全地存檔。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- AWS Network Firewall 是 AWS 雲端中 VPCs 具狀態、受管的網路防火牆和入侵偵測和預防服務。

其他工具

- Splunk 可協助您監控、視覺化和分析日誌資料。

史詩

建立 IAM 角色

任務	Description	所需的技能

建立 IAM 政策。	遵循使用 JSON 編輯器建立政策中的指示，建立授予 CloudWatch Logs 資料和 CloudWatch 指標唯讀存取權的 IAM 政策。將以下 政策貼到 JSON 編輯器。
	
	`{`
	`  "Statement": [`
	`  {`
	`  "Action": [`
		AWS 管理員

任務	Description	所需的技能
	"cloudwatch:List*", "cloudwatch:Get*", "network-firewall:List*", "logs:Describe*", "logs:Get*", "logs:List*", "logs:StartQuery", "logs:StopQuery", "logs:TestMetricFilter", "logs:FilterLogEvents", "network-firewall:Describe*"], "Effect": "Allow", "Resource": "*" }], "Version": "2012-10-17" }	

任務	Description	所需的技能
建立新的 IAM 角色。	遵循建立角色以委派許可給 AWS 服務中的指示，以建立 Splunk Add-On for AWS 用來存取 CloudWatch 的 IAM 角色。針對許可政策，選擇您先前建立的政策。	AWS 管理員
將 IAM 角色指派給 Splunk 叢集中的 EC2 執行個體。	1. 前往 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。 2. 在導覽窗格中，選擇執行個體。 3. 在 Splunk 叢集中選取 EC2 執行個體。 4. 選擇動作、安全性，然後選擇修改 IAM 角色。 5. 選取您先前建立的 IAM 角色，然後選擇儲存。	AWS 管理員

安裝適用於 AWS 的 Splunk 附加元件

任務	Description	所需的技能
安裝 附加元件。	1. 在 Splunk 儀表板中，導覽至 Splunk 應用程式。 2. 搜尋 Amazon Web Services 的 Splunk 附加元件。 3. 選擇 Install (安裝)。 4. 提供您的 Splunk 登入資料。	Splunk 管理員

任務	Description	所需的技能
設定 AWS 登入資料。	1. 在 Splunk 儀表板中，導覽至 Splunk Add-on for AWS。 2. 選擇 Configuration (組態)。 3. 在自動探索的 IAM 角色欄中，選取您先前建立的 IAM 角色。 如需詳細資訊，請參閱 Splunk 文件中的在 Splunk 平台執行個體中尋找 IAM 角色。	Splunk 管理員

設定 Splunk 對 CloudWatch 的存取

任務	Description	所需的技能
設定從 CloudWatch Logs 擷取 Network Firewall 日誌。	1. 在 Splunk 儀表板中，導覽至 Splunk Add-on for AWS。 2. 選擇輸入。 3. 選擇建立新輸入。 4. 在清單中，選擇自訂資料類型，然後選擇 CloudWatch Logs。 5. 為您的網路防火牆日誌提供名稱、AWS 帳戶、AWS 區域和日誌群組 。 6. 選擇儲存。 根據預設，Splunk 每 10 分鐘擷取一次日誌資料。這是進階設定下的可設定參數。如需	Splunk 管理員

任務	Description	所需的技能
	詳細資訊，請參閱 Splunk 文件中的使用 Splunk Web 設定 CloudWatch Logs 輸入。	

任務	Description	所需的技能
設定從 CloudWatch 擷取 Network Firewall 指標。	1. 在 Splunk 儀表板中，導覽至 Splunk Add-on for AWS。 2. 選擇輸入。 3. 選擇建立新輸入。 4. 在清單中，選擇 CloudWatch。 5. 為您的 Network Firewall 指標提供名稱、AWS 帳戶和 AWS 區域。 6. 在指標組態旁邊，選擇進階模式中的編輯。 7. （選用）刪除所有預先設定的命名空間。 8. 選擇新增命名空間，然後命名為 AWS/NetworkFirewall。 9. 在維度值中，新增下列項目。 `[{"AvailabilityZone":[".*"],"Engine":[".*"],"FirewallName":[".*"]}]` 10針對指標，選擇全部。 11針對指標統計資料，選擇總和。 12選擇確定。 13選擇儲存。 根據預設，Splunk 每 5 分鐘擷取一次指標資料。這是進	Splunk 管理員

任務	Description	所需的技能
	階設定下的可設定參數。如需詳細資訊，請參閱 Splunk 文件中的使用 Splunk Web 設定 CloudWatch 輸入。	

使用查詢建立 Splunk 視覺化

任務	Description	所需的技能
檢視最高來源 IP 地址。	1. 在 Splunk 儀表板中，導覽至 搜尋和報告。 2. 在此處輸入搜尋方塊中，輸入下列內容。 `sourcetype="aws:cloudwatchlogs" | top event.src_ip` 此查詢會以遞減順序顯示流量最多的來源 IP 地址資料表。 3. 如需圖形表示，請選擇視覺化。	Splunk 管理員
檢視封包統計資料。	1. 在 Splunk 儀表板中，導覽至 搜尋和報告。 2. 在此處輸入搜尋方塊中，輸入下列內容。 `sourcetype="aws:cloudwatch"| timechart sum(Sum) by metric_name`	Splunk 管理員

任務	Description	所需的技能
	此查詢會顯示ReceivedPackets 每分鐘指標 DroppedPackets 、 PassedPackets 和 的資料表。 3. 如需圖形表示，請選擇視覺化。	
檢視最常用的來源連接埠。	1. 在 Splunk 儀表板中，導覽至 搜尋和報告。 2. 在此處輸入搜尋方塊中，輸入下列內容。 `sourcetype="aws:cloudwatchlogs" | top event.dest_port` 此查詢會以遞減順序顯示流量最多的來源連接埠資料表。 3. 如需圖形表示，請選擇視覺化。	Splunk 管理員

相關資源

AWS 文件

- 建立角色以委派許可給 AWS 服務 (IAM 文件)
- 建立 IAM 政策 (IAM 文件)
- 在 AWS Network Firewall 中記錄和監控 (Network Firewall 文件)
- AWS Network Firewall 的路由表組態 (Network Firewall 文件)

AWS 部落格文章

- AWS Network Firewall 部署模型

AWS Marketplace

- Splunk Enterprise Amazon Machine Image (AMI)

更多模式

- 使用 Session Manager 和 Amazon EC2 Instance Connect 存取堡壘主機
- 使用 AWS Fargate、AWS PrivateLink 和 Network Load Balancer，在 Amazon ECS 上私下存取容器應用程式
- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon ECS 上私下存取容器應用程式
- 使用 AWS Managed Microsoft AD 和內部部署 Microsoft Active Directory 集中 DNS 解析
- 使用、 AWS Amplify Angular 和 Module Federation 建立微型前端的入口網站
- 使用私有端點和 Application Load Balancer 在內部網站上部署 Amazon API Gateway API
- 使用 部署公有子網路的偵測屬性型存取控制 AWS Config
- 部署公有子網路的預防性屬性型存取控制
- 在 Amazon RDS 中啟用 PostgreSQL 資料庫執行個體的加密連線
- 使用 AWS Transit Gateway Connect 將 VRFs 擴展至 AWS AWS Transit Gateway
- 在 上將 F5 BIG-IP 工作負載遷移至 F5 BIG-IP VE AWS 雲端
- 啟用 Amazon EKS Auto 模式時遷移 NGINX 傳入控制器
- 在非工作負載子網路的多帳戶 VPC 設計中保留可路由 IP 空間
- 使用服務控制政策，防止帳戶層級的網際網路存取
- 從 AWS Network Firewall 傳送提醒到 Slack 頻道
- 使用 Amazon CloudFront 在 Amazon S3 儲存貯體中透過 VPC 提供靜態內容
- 使用 AWS Elastic Disaster Recovery 為 Oracle JD Edwards EnterpriseOne 設定災難復原
- 使用 BMC Discovery 查詢來擷取遷移資料以進行遷移規劃
- 使用 Network Firewall 從傳出流量的伺服器名稱指示擷取 DNS 網域名稱

內容交付

主題

- 使用 AWS Firewall Manager 和 Amazon Data Firehose 將 AWS WAF 日誌傳送至 Splunk
- 使用 Amazon CloudFront 在 Amazon S3 儲存貯體中透過 VPC 提供靜態內容
- 更多模式

使用 AWS Firewall Manager 和 Amazon Data Firehose 將 AWS WAF 日誌傳送至 Splunk

Michael Friedenthal、Aman Kaur Gandhi 和 JJ Johnson，Amazon Web Services

總結

從歷史上看，有兩種方式可將資料移至 Splunk：推送或提取架構。提取架構透過重試提供交付資料保證，但需要 Splunk 中輪詢資料的專用資源。由於輪詢，提取架構通常不是即時的。中的推送架構通常具有較低的延遲、更具可擴展性，並降低操作複雜性和成本。不過，它不保證交付，通常需要客服人員。

Splunk 與 Amazon Data Firehose 整合，可透過 HTTP 事件收集器 (HEC) 將即時串流資料交付至 Splunk。此整合提供推送和提取架構的優點，可保證透過重試交付資料、近乎即時、低延遲和低複雜性。HEC 會透過 HTTP 或 HTTPS 將資料直接快速有效地傳送至 Splunk。HECs以字符為基礎，無需在應用程式或支援檔案中硬式編碼登入資料。

在 AWS Firewall Manager 政策中，您可以為所有帳戶中的所有 AWS WAF Web ACL 流量設定記錄，然後使用 Firehose 交付串流將該記錄資料傳送至 Splunk 以進行監控、視覺化和分析。此解決方案提供下列優點：

- 所有帳戶中 AWS WAF Web ACL 流量的集中管理和記錄
- Splunk 與單一 整合 AWS 帳戶
- 可擴展性
- 近乎即時的日誌資料交付
- 透過使用無伺服器解決方案進行成本最佳化，因此您不需要支付未使用的資源。

先決條件和限制

先決條件

- 屬於 組織一部分 AWS 帳戶 的作用中 AWS Organizations。
- 您必須擁有下列許可，才能使用 Firehose 啟用記錄：
  - `iam:CreateServiceLinkedRole`
  - `firehose:ListDeliveryStreams`
  - `wafv2:PutLoggingConfiguration`
- AWS WAF 必須設定 及其 Web ACLs。如需說明，請參閱 入門 AWS WAF。

- AWS Firewall Manager 必須設定 。如需說明，請參閱AWS Firewall Manager 先決條件。
- AWS WAF 必須設定 的 Firewall Manager 安全政策。如需說明，請參閱 AWS Firewall ManagerAWS WAF 政策入門。
- Splunk 必須使用 Firehose 可存取的公有 HTTP 端點進行設定。

限制

- AWS 帳戶 必須在 的單一組織中進行管理 AWS Organizations。
- Web ACL 必須位於與交付串流相同的區域。如果您要擷取 Amazon CloudFront 的日誌，請在美國東部（維吉尼亞北部）區域 建立 Firehose 交付串流。 `us-east-1`
- Splunk 附加元件 for Firehose 適用於付費 Splunk Cloud 部署、分散式 Splunk Enterprise 部署和單一執行個體 Splunk Enterprise 部署。免費試用 Splunk Cloud 部署不支援此附加元件。

Architecture

目標技術堆疊

- Firewall Manager
- Firehose
- Amazon Simple Storage Service (Amazon S3)
- AWS WAF
- Splunk

目標架構

下圖顯示如何使用 Firewall Manager 集中記錄 AWS WAF 所有資料，並透過 Firehose 將其傳送至 Splunk。

1. AWS WAF Web ACLs 會將防火牆日誌資料傳送至 Firewall Manager。
2. Firewall Manager 會將日誌資料傳送至 Firehose。
3. Firehose 交付串流會將日誌資料轉送至 Splunk 和 S3 儲存貯體。S3 儲存貯體會在 Firehose 交付串流發生錯誤時做為備份。

自動化和擴展

此解決方案旨在擴展和容納組織內的所有 AWS WAF Web ALCs。您可以設定所有 Web ACLs使用相同的 Firehose 執行個體。不過，如果您想要設定和使用多個 Firehose 執行個體，您可以這麼做。

工具

AWS 服務

- AWS Firewall Manager 是一種安全管理服務，可協助您集中設定和管理帳戶和應用程式中的防火牆規則 AWS Organizations。
- Amazon Data Firehose 可協助您將即時串流資料交付至其他 AWS 服務自訂 HTTP 端點，以及受支援的第三方服務供應商所擁有的 HTTP 端點，例如 Splunk。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS WAF 是一種 Web 應用程式防火牆，可協助您監控轉送至受保護 Web 應用程式資源的 HTTP 和 HTTPS 請求。

其他工具

- Splunk 可協助您監控、視覺化和分析日誌資料。

史詩

設定 Splunk

任務	Description	所需的技能
安裝 Splunk 應用程式 AWS。	1. 登入您的 Splunk 繁重轉寄站。預設 URL 為 `http://<IP address>:8000` 。 2. 在左側導覽中，選擇應用程式旁的齒輪按鈕。 3. 選擇瀏覽更多應用程式。 4. 搜尋 AWS。 5. 對於適用於 的 Splunk 應用程式 AWS，選擇安裝。	安全管理員、Splunk 管理員

任務	Description	所需的技能
	6. 輸入您的 Splunk.com 登入憑證，接受條款和條件，然後選擇登入和安裝。 7. 選擇完成。	
安裝 的附加元件 AWS WAF。	重複上述指示以安裝適用於 Splunk 的 AWS Web Application Firewall 附加元件。	安全管理員、Splunk 管理員

任務	Description	所需的技能
安裝和設定 Splunk 附加元件 for Firehose。	1. 安裝和設定 Splunk 附加元件 for Firehose。作為安裝和組態的一部分，如果 Splunk 平台需要，您可以設定 HTTP 事件收集器，並準備基礎設施，將日誌資料傳送到您的索引器。請參閱與 Splunk 部署對應的說明： • Splunk Cloud 部署 (Splunk 文件) • 分散式 Splunk Enterprise 部署 (Splunk 文件) • 單一執行個體 Splunk Enterprise 部署 (Splunk 文件) **Important** 安裝並設定 Splunk 附加元件後，請停止此程序。請勿繼續執行設定 Firehose 將資料傳送至 Splunk 平台的指示。 2. 請記下 HTTP 事件收集器字符和 HTTP 端點。稍後當您設定交付串流時，需要此值。	安全管理員、Splunk 管理員

建立 Firehose 交付串流

任務	Description	所需的技能
授予 Firehose 存取 Splunk 目的地的權限。	設定允許 Firehose 存取 Splunk 目的地並將日誌資料備份到 S3 儲存貯體的存取政策。如需詳細資訊，請參閱授予 Firehose 存取 Splunk 目的地。	安全管理員
建立 Firehose 交付串流。	在您管理 Web ACLs的相同帳戶中 AWS WAF，在 Firehose 中建立交付串流。您在建立交付串流時必須擁有 IAM 角色。Firehose 假設 IAM 角色並取得指定 S3 儲存貯體的存取權。如需說明，請參閱建立交付串流。注意下列事項： • 交付串流名稱必須以 開頭`aws-waf-logs-` 。 • 針對來源，選擇直接 PUT。 • 針對 S3 備份模式，選擇備份所有事件，然後選擇現有的儲存貯體或建立新的儲存貯體。 • 對於目的地，請遵循 Firehose 文件中為您的目的地選擇 Splunk 中的指示。如需 Splunk 端點和端點類型值的相關資訊，請參閱 Splunk 文件中的設定 Amazon Data Firehose。	安全管理員

任務	Description	所需的技能
	針對您在 HTTP 事件收集器中設定的每個字符重複此程序。	
測試交付串流。	測試交付串流，以驗證其已正確設定。如需說明，請參閱 Firehose 文件中的測試使用 Splunk 做為目的地。	安全管理員

設定 Firewall Manager 記錄資料

任務	Description	所需的技能
設定 Firewall Manager 政策。	Firewall Manager 政策必須設定為啟用記錄，並將日誌轉送至正確的 Firehose 交付串流。如需詳細資訊和說明，請參閱設定 AWS WAF 政策的記錄。	安全管理員

相關資源

AWS resources

- 記錄 Web ACL 流量 (AWS WAF 文件）
- 設定 AWS WAF 政策的記錄 (AWS WAF 文件）
- 教學課程：使用 Amazon Data Firehose 將 VPC 流程日誌傳送至 Splunk (Firehose 文件）
- 如何使用 Amazon Data Firehose 將 VPC 流程日誌推送至 Splunk？ (AWS 知識中心）
- 使用 Amazon Data Firehose 將資料擷取至 Splunk (AWS 部落格文章）

Splunk 文件

- 適用於 Amazon Data Firehose 的 Splunk 附加元件

使用 Amazon CloudFront 在 Amazon S3 儲存貯體中透過 VPC 提供靜態內容

Angel Emmanuel Hernandez Cebrian，Amazon Web Services

總結

當您提供託管在 Amazon Web Services (AWS) 上的靜態內容時，建議方法是使用 Amazon Simple Storage Service (S3) 儲存貯體做為原始伺服器，並使用 Amazon CloudFront 來分發內容。此解決方案有兩個主要優點：在節點快取靜態內容的便利性，以及為 CloudFront 分佈定義 Web 存取控制清單 (Web ACLs) 的功能，可協助您以最少的組態和管理開銷保護對內容的請求。

不過，標準建議的方法有常見的架構限制。在某些環境中，您希望在虛擬私有雲端 (VPC) 中部署的虛擬防火牆設備檢查所有內容，包括靜態內容。標準方法不會透過 VPC 路由流量以進行檢查。此模式提供替代的架構解決方案。您仍然會使用 CloudFront 分佈來提供 S3 儲存貯體中的靜態內容，但流量會使用 Application Load Balancer 透過 VPC 路由。然後AWS Lambda 函數會從 S3 儲存貯體擷取並傳回內容。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- S3 儲存貯體中託管的靜態網站內容。

限制

- 此模式中的資源必須位於單一 AWS 區域，但可以在不同的 AWS 帳戶中佈建。
- 限制適用於 Lambda 函數可分別接收和傳送的最大請求和回應大小。如需詳細資訊，請參閱 Lambda 函數中的限制做為目標 (Elastic Load Balancing 文件）。 https://docs.aws.amazon.com/elasticloadbalancing/latest/application/lambda-functions.html
- 使用此方法時，請務必在效能、可擴展性、安全性和成本效益之間取得良好的平衡。儘管 Lambda 具有高可擴展性，但如果並行 Lambda 調用的數量超過最大配額，則會調節某些請求。如需詳細資訊，請參閱 Lambda 配額 (Lambda 文件）。使用 Lambda 時，您也需要考慮定價。若要將 Lambda 調用降至最低，請確定您已正確定義 CloudFront 分佈的快取。如需詳細資訊，請參閱最佳化快取和可用性 (CloudFront 文件）。

Architecture

目標技術堆疊

- CloudFront
- Amazon Virtual Private Cloud (Amazon VPC)
- Application Load Balancer
- Lambda
- Amazon S3

目標架構

當您需要使用 CloudFront 透過 VPC 從 S3 儲存貯體提供靜態內容時，下圖會顯示建議的架構。

1. 用戶端請求 CloudFront 分佈的 URL，以取得 S3 儲存貯體中的特定網站檔案。
2. CloudFront 會將請求傳送至 AWS WAF。AWS WAF 會使用套用至 CloudFront 分佈的 Web ACLs 來篩選請求。如果請求被判定為有效，流程會繼續。如果請求被判定為無效，用戶端會收到 403 錯誤。
3. CloudFront 會檢查其內部快取。如果有符合傳入請求的有效金鑰，則會將關聯的值傳回給用戶端做為回應。如果沒有，流程會繼續。
4. CloudFront 會將請求轉送至指定 Application Load Balancer 的 URL。
5. Application Load Balancer 具有以 Lambda 函數為基礎的與目標群組相關聯的接聽程式。Application Load Balancer 會叫用 Lambda 函數。
6. Lambda 函數會連線至 S3 儲存貯體、對其執行`GetObject`操作，並將內容傳回為回應。

自動化和擴展

若要使用此方法自動部署靜態內容，請建立 CI/CD 管道來更新託管網站的 Amazon S3 儲存貯體。

Lambda 函數會自動擴展，以在服務的配額和限制內處理並行請求。如需詳細資訊，請參閱 Lambda 函數擴展和 Lambda 配額 (Lambda 文件）。對於其他 AWS 服務和功能，例如 CloudFront 和 Application Load Balancer，AWS 會自動擴展這些服務和功能。

工具

- Amazon CloudFront 透過全球資料中心網路提供 Web 內容，從而降低延遲並改善效能，從而加快 Web 內容的發佈速度。
- Elastic Load Balancing (ELB) 會將傳入的應用程式或網路流量分配到多個目標。在此模式中，您可以使用透過 Elastic Load Balancing 佈建的 Application Load Balancer，將流量導向 Lambda 函數。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路類似於您在自己的資料中心內操作的傳統網路，具有使用可擴展的 AWS 基礎設施的優勢。

史詩

使用 CloudFront 透過 VPC 從 Amazon S3 提供靜態內容

任務	Description	所需的技能
建立 VPC。	建立 VPC 以託管在此模式中部署的資源，例如 Application Load Balancer 和 Lambda 函數。 如需說明，請參閱建立 VPC (Amazon VPC 文件）。	雲端架構師
建立 AWS WAF Web ACL。	建立 AWS WAF Web ACL。稍後在此模式中，您會將此 Web ACL 套用至 CloudFront 分佈。如需說明，請參閱建立 Web ACL (AWS WAF 文件）。	雲端架構師
建立 Lambda 函數。	建立 Lambda 函數，以網站形式提供 S3 儲存貯體中託管的靜態內容。使用此模式額外資訊區段中提供的程式碼。自訂	一般 AWS

任務	Description	所需的技能
	程式碼以識別您的目標 S3 儲存貯體。	
上傳 Lambda 函數。	輸入下列命令，將 Lambda 函數程式碼上傳至 Lambda 中的 .zip 檔案封存。 `aws lambda update-function-code \ --function-name \ --zip-file fileb://lambda-alb-s3-website.zip`	一般 AWS
建立 Application Load Balancer。	建立指向 Lambda 函數的面向網際網路的 Application Load Balancer。如需說明，請參閱建立 Lambda 函數的目標群組 (Elastic Load Balancing 文件）。如需高可用性組態，請建立 Application Load Balancer，並將其連接至不同可用區域中的私有子網路。	雲端架構師

任務	Description	所需的技能
建立 CloudFront 分佈。	建立指向您建立之 Application Load Balancer 的 CloudFront 分佈。 1. 登入 AWS 管理主控台，並在 https：//https://console.aws.amazon.com/cloudfront/v3/home 開啟 CloudFront 主控台。 2. 選擇 Create Distribution (建立分佈)。 3. 在 Create Distribution Wizard (建立分佈精靈) 的第一頁上，在 Web (Web) 區段中選擇 Get Started (開始使用)。 4. 為您的分佈指定設定。如需詳細資訊，請參閱在建立或更新分佈時您指定的值。注意下列事項： a. 將 Application Load Balancer 設定為原始伺服器。 b. 在分佈設定中，選擇您要透過 AWS WAF 套用的現有 Web ACLs。如需詳細資訊，請參閱 AWS WAF Web ACL。 5. 儲存您的變更。 6. CloudFront 建立分佈後，分佈的狀態欄值會從 InProgress 變更為已部署。如果您選擇啟用分佈，將會	雲端架構師

任務	Description	所需的技能
	在狀態切換為 Deployed (已部署) 之後準備好處理請求。	

相關資源

AWS 文件

- 最佳化快取和可用性 (CloudFront 文件)
- Lambda 函數做為目標 (Elastic Load Balancing 文件)
- Lambda 配額 (Lambda 文件)

AWS 服務網站

- Application Load Balancer
- Lambda
- CloudFront
- Amazon S3
- AWS WAF
- Amazon VPC

其他資訊

Code

下列範例 Lambda 函數是以 Node.js 撰寫。此 Lambda 函數充當 Web 伺服器，對包含網站資源的 S3 儲存貯體執行`GetObject`操作。

```
/**

 * This is an AWS Lambda function created for demonstration purposes.

 * It retrieves static assets from a defined Amazon S3 bucket.
```

```
 * To make the content available through a URL, use an Application Load Balancer with a
 Lambda integration.
 *
 * Set the S3_BUCKET environment variable in the Lambda function definition.
 */

var AWS = require('aws-sdk');

exports.handler = function(event, context, callback) {

    var bucket = process.env.S3_BUCKET;
    var key = event.path.replace('/', '');

    if (key == '') {
        key = 'index.html';
    }

    // Fetch from S3
    var s3 = new AWS.S3();
    return s3.getObject({Bucket: bucket, Key: key},
       function(err, data) {

            if (err) {
                return err;
            }

            var isBase64Encoded = false;
            var encoding = 'utf8';

            if (data.ContentType.indexOf('image/') > -1) {
                isBase64Encoded = true;
                encoding = 'base64'
            }

            var resp = {
                statusCode: 200,
                headers: {
                    'Content-Type': data.ContentType,
                },
                body: new Buffer(data.Body).toString(encoding),
                isBase64Encoded: isBase64Encoded
            };

            callback(null, resp);
```

```
        }
    );
};
```

更多模式

- 檢查 Amazon CloudFront 分佈是否有存取記錄、HTTPS 和 TLS 版本
- 在 Amazon EKS 叢集上部署以 gRPC 為基礎的應用程式，並使用 Application Load Balancer 存取它
- 部署公有子網路的預防性屬性型存取控制
- 使用 Terraform 在 AWS Wavelength 區域中部署資源
- 使用 Terraform 部署 AWS WAF 解決方案的安全自動化
- 為以儲存格為基礎的架構設定無伺服器儲存格路由器
- 使用 Amazon Q Developer 作為編碼助理，以提高您的生產力
- 使用 Splunk 檢視 AWS Network Firewall 日誌和指標

資料庫和儲存體

主題

- 資料庫
- 儲存和備份

資料庫

主題

- 使用連結的伺服器從 Amazon EC2 上的 Microsoft SQL Server 存取內部部署 Microsoft SQL Server 資料表
- 使用僅供讀取複本將 HA 新增至 Amazon RDS Custom 上的 Oracle PeopleSoft
- 分析從 Oracle 到 PostgreSQL 的部分資料庫遷移的物件相依性
- 評估將 SQL Server 資料庫遷移至 AWS 上的 MongoDB Atlas 的查詢效能
- 使用 IaC 原則自動化 Amazon Aurora 全域資料庫的藍/綠部署
- 自動化跨 的 Amazon RDS 執行個體複寫 AWS 帳戶
- 使用 AWS Lambda 和 任務排程器，在 Amazon EC2 上執行的 SQL Server Express 版本中自動化資料庫任務
- 使用 DR Orchestrator Framework 自動化跨區域容錯移轉和容錯回復
- 使用 Systems Manager 和 EventBridge 自動備份 SAP HANA 資料庫
- 使用 Python 應用程式自動產生 Amazon DynamoDB 的 PynamoDB 模型和 CRUD 函數 DynamoDB
- 使用 Cloud Custodian 封鎖對 Amazon RDS 的公開存取
- 使用 設定 Amazon RDS for Microsoft SQL Server 的 Windows 身分驗證 AWS Managed Microsoft AD
- 設定對 Amazon DynamoDB 的跨帳戶存取權
- 在 AWS 上 SQL Server 的 Always On 可用性群組中設定唯讀路由
- 在 pgAdmin 中使用 SSH 通道進行連線
- 將 JSON Oracle 查詢轉換為 PostgreSQL 資料庫 SQL
- 使用 跨帳戶複製 Amazon DynamoDB 資料表 AWS Backup
- 使用自訂實作跨帳戶複製 Amazon DynamoDB 資料表
- 建立 Amazon RDS 和 Amazon Aurora 的詳細成本和用量報告
- 使用 Terraform 在 Amazon EC2 和 Amazon FSx 上部署 SQL Server 容錯移轉叢集執行個體
- 在 Amazon Aurora PostgreSQL 和 Amazon RDS for PostgreSQL 中模擬 Oracle PL/SQL 關聯陣列
- 在 Amazon RDS 中啟用 PostgreSQL 資料庫執行個體的加密連線
- 加密現有的 Amazon RDS for PostgreSQL 資料庫執行個體
- 在啟動時強制執行 Amazon RDS 資料庫的自動標記
- 預估 DynamoDB 資料表的隨需容量成本

- Amazon DynamoDB 資料表的預估儲存成本
- 使用 AWR 報告估計 Oracle 資料庫的 Amazon RDS 引擎大小
- 使用 AWS DMS 將 Amazon RDS for SQL Server 資料表匯出至 S3 儲存貯體
- 在 Aurora PostgreSQL 中處理動態 SQL 陳述式中的匿名區塊
- 協助強制執行 DynamoDB 標記
- 使用 AWS DMS 和 Amazon Aurora 實作跨區域災難復原
- 將具有超過 100 個引數的 Oracle 函數和程序遷移至 PostgreSQL
- 將 Redis 工作負載遷移至 AWS 上的 Redis Enterprise Cloud
- 使用具有相同主機名稱的 SAP HSR 將 SAP HANA 遷移至 AWS
- 使用 遷移 Microsoft SQL Server Always On 可用性群組 AWS Application Migration Service
- 使用分散式可用性群組將 SQL Server 遷移至 AWS
- 在 上將關聯式資料庫遷移至 MongoDB Atlas AWS
- 在 上將自我託管的 MongoDB 環境遷移至 MongoDB Atlas AWS
- 使用 AWS DMS 將 Oracle 資料庫遷移至 Amazon DynamoDB
- 使用 SharePlex 和 AWS DMS 從 Oracle 8i 或 9i 遷移至 Amazon RDS for Oracle
- 將內部部署 MySQL 資料庫遷移至 Amazon EC2
- 在沒有加密的情況下監控 Amazon Aurora 是否有執行個體
- 使用 Amazon CloudWatch 監控 Oracle GoldenGate 日誌
- 在 Amazon RDS for Oracle 上將 Oracle Database Enterprise Edition 轉換為 Standard Edition 2
- 使用 Precisely Connect 將大型主機資料庫複寫至 AWS
- 使用 Lambda 和 Secrets Manager 來排程 Amazon RDS for PostgreSQL 和 Aurora PostgreSQL 的任務
- 使用內部部署 SMTP 伺服器和 Database Mail 傳送 Amazon RDS for SQL Server 資料庫執行個體的通知
- 在 AWS 上設定 SAP on IBM Db2 的災難復原
- 使用 Terraform 設定資料庫遷移的 CI/CD 管道
- 使用 FSx for Windows File Server 在 Amazon EC2 上設定 Microsoft SQL Server 容錯移轉叢集
- 使用作用中待命資料庫為 Amazon RDS Custom 上的 Oracle 電子商務套件設定 HA/DR 架構
- 在 上將資料從 IBM Db2、SAP、Sybase 和其他資料庫串流至 MongoDB Atlas AWS
- Amazon RDS Custom for Oracle 上 Oracle PeopleSoft 應用程式的轉換角色
- 將資料從 Amazon Redshift 叢集跨帳戶卸載至 Amazon S3

- 依工作負載的資料庫遷移模式
- 更多模式

使用連結的伺服器從 Amazon EC2 上的 Microsoft SQL Server 存取內部部署 Microsoft SQL Server 資料表

Tirumala Dasari 和 Eduardo Valentim，Amazon Web Services

總結

此模式說明如何使用連結的伺服器，從在 Amazon Elastic Compute Cloud (Amazon EC2) Windows 或 Linux 執行個體上執行或託管的 Microsoft SQL Server 資料庫存取在 Microsoft Windows 上執行的內部部署 Microsoft SQL Server 資料庫資料表。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Amazon EC2 搭配在 Amazon Linux AMI (Amazon Machine Image) 上執行的 Microsoft SQL Server
- 內部部署 Microsoft SQL Server (Windows) 伺服器與 Windows 或 Linux EC2 執行個體之間的 AWS Direct Connect

產品版本

- SQL Server 2016 或更新版本

Architecture

來源技術堆疊

- 在 Windows 上執行的內部部署 Microsoft SQL Server 資料庫
- Amazon EC2 搭配在 Windows AMI 或 Linux AMI 上執行的 Microsoft SQL Server

目標技術堆疊

- 在 Amazon Linux AMI 上執行的 Amazon EC2 搭配 Microsoft SQL Server
- Amazon EC2 搭配在 Windows AMI 上執行的 Microsoft SQL Server

來源和目標資料庫架構

工具

- Microsoft SQL Server Management Studio (SSMS) 是用於管理 SQL Server 基礎設施的整合環境。它提供使用者介面和一組工具，其中包含與 SQL Server 互動的豐富指令碼編輯器。

史詩

在 Windows SQL Server 中將身分驗證模式變更為 Windows for SQL Server

任務	Description	所需的技能
透過 SSMS 連線至 Windows SQL Server。		DBA
從 Windows SQL Server 執行個體的內容（按一下滑鼠右鍵）選單，將 SQL Server 中的身分驗證模式變更為 Windows。		DBA

重新啟動 Windows MSSQL 服務

任務	Description	所需的技能
重新啟動 SQL 服務。	1. 在 SSMS Object Explorer 中，選擇 SQL Server 執行個體。 2. 開啟內容（按一下滑鼠右鍵）選單。 3. 選擇重新啟動。	DBA

建立新的登入，然後選擇要在 Windows SQL Server 中存取的資料庫

任務	Description	所需的技能
在安全索引標籤中，開啟登入的內容（按一下滑鼠右鍵）選單，然後選取新的登入。		DBA
在一般索引標籤中，選擇 SQL Server 身分驗證、輸入使用者名稱、輸入密碼，然後確認密碼，並清除在下次登入時變更密碼的選項。		DBA
在伺服器角色索引標籤中，選擇公有。		DBA
在使用者映射索引標籤中，選擇您要存取的資料庫和結構描述，然後反白顯示資料庫以選取資料庫角色。	選取公有和 db_datareader 以從資料庫資料表存取資料。	DBA
選擇確定以建立使用者。		DBA

將 Windows SQL Server IP 新增至 Linux SQL Server 主機檔案

任務	Description	所需的技能
透過終端機視窗連線至 Linux SQL Server 方塊。		DBA
開啟 /etc/hosts 檔案，並使用 SQL Server 新增 Windows 機器的 IP 地址。		DBA
儲存主機檔案。		DBA

在 Linux SQL Server 上建立連結的伺服器

任務	Description	所需的技能
使用預存程序 master.sys.sp_addlinkedserver 和 master.dbo.sp_addlinkedsrvlogin 建立連結的伺服器。	如需使用這些預存程序的詳細資訊，請參閱其他資訊一節。	DBA、開發人員

驗證在 SSMS 中建立的連結伺服器和資料庫

任務	Description	所需的技能
在 SSMS 的 Linux SQL Server 中，前往連結的伺服器並重新整理。		DBA
在左側窗格中展開建立的連結伺服器和目錄。	您將看到選取的 SQL Server 資料庫，其中包含資料表和檢視。	DBA

確認您可以存取 Windows SQL Server 資料庫資料表

任務	Description	所需的技能
在 SSMS 查詢視窗中，執行查詢：「從 【sqllin】.dms_sample_win.dbo.mlb_data 選取前 3 *」。	請注意， FROM 子句使用四部分語法： computer.database.schema.table （例如 SELECT 名稱 "SQL2 database" FROM 【sqllin】.master.sys.databases)。在我們的範例中，我們在主機檔案中建立了 SQL2 的別名，因此您不需要在方括號之間輸入實際的 NetBIOS 名稱。如果您確實使用實際的 NetBIOS 名稱，請注意，AW	DBA、開發人員

任務	Description	所需的技能
	S 預設為 NetBIOS 名稱，例如 Win-xxxx，SQL Server 需要有破折號名稱的方括號。	

相關資源

- Linux 上 SQL Server 的版本備註

其他資訊

使用預存程序建立連結的伺服器

SSMS 不支援為 Linux SQL Server 建立連結伺服器，因此您必須使用這些預存程序來建立它們：

```
EXEC master.sys.sp_addlinkedserver @server= N'SQLLIN' , @srvproduct= N'SQL Server'
EXEC master.dbo.sp_addlinkedsrvlogin
 @rmtsrvname=N'SQLLIN',@useself=N'False',@locallogin=NULL,@rmtuser=N'username',@rmtpassword='Te
```

注意 1：在預存程序 中輸入您先前在 Windows SQL Server 中建立的登入憑證`master.dbo.sp_addlinkedsrvlogin`。

注意 2：`@server`名稱`SQLLIN`和主機檔案項目名稱`172.12.12.4 SQLLIN`應該相同。

您可以使用此程序為下列案例建立連結的伺服器：

- 透過連結伺服器從 Linux SQL Server 到 Windows SQL Server （如此模式所述）
- 透過連結伺服器將 Windows SQL Server 轉換為 Linux SQL Server
- 透過連結伺服器將 Linux SQL Server 連線到另一個 Linux SQL Server

使用僅供讀取複本將 HA 新增至 Amazon RDS Custom 上的 Oracle PeopleSoft

sampath kathirvel，Amazon Web Services

總結

若要在 Amazon Web Services (AWS) 上執行 Oracle PeopleSoft 企業資源規劃 (ERP) 解決方案，您可以使用 Amazon Relational Database Service (Amazon RDS) 或 Amazon RDS Custom for Oracle，以支援需要存取基礎作業系統和資料庫環境的舊版、自訂和封裝應用程式。如需規劃遷移時要考量的關鍵因素，請參閱 AWS 方案指引中的 Oracle 資料庫遷移策略。

截至本文撰寫為止，RDS Custom for Oracle 不支援異地同步備份選項，此選項可供 Amazon RDS for Oracle 做為使用儲存體複寫的 HA 解決方案使用。相反地，此模式會使用建立和維護主要資料庫實體副本的待命資料庫來實現 HA。此模式著重於使用 Oracle Data Guard 設定僅供讀取複本，在具有 HA 的 Amazon RDS Custom 上執行 PeopleSoft 應用程式資料庫的步驟。

此模式也會將僅供讀取複本變更為唯讀模式。讓僅供讀取複本處於唯讀模式可提供額外的優點：

- 從主要資料庫卸載唯讀工作負載
- 使用 Oracle Active Data Guard 功能從待命資料庫擷取運作狀態良好的區塊，以啟用損毀區塊的自動修復
- 使用 Far Sync 功能讓遠端待命資料庫保持同步，而不會產生與長途重做日誌傳輸相關的效能額外負荷。

在唯讀模式下使用複本需要 Oracle Active Data Guard 選項，因為這是 Oracle Database Enterprise Edition 的單獨授權功能，因此需要支付額外費用。

先決條件和限制

先決條件

- Amazon RDS Custom 上的現有 PeopleSoft 應用程式。如果您沒有應用程式，請參閱將 Oracle PeopleSoft 遷移至 Amazon RDS Custom 模式。
- 單一 PeopleSoft 應用程式層。不過，您可以調整此模式以使用多個應用程式層。
- Amazon RDS Custom 已設定至少 8 GB 的交換空間。
- Oracle Active Data Guard 資料庫授權，可將僅供讀取複本轉換為唯讀模式，並使用它將報告任務卸載至待命。如需詳細資訊，請參閱 Oracle Technology 商業價目表。

限制

- RDS Custom for Oracle 的一般限制和不支援的組態
- 與 Amazon RDS Custom for Oracle 僅供讀取複本相關聯的限制

產品版本

- 如需 Amazon RDS Custom 支援的 Oracle 資料庫版本，請參閱 RDS Custom for Oracle。
- 如需 Amazon RDS Custom 支援的 Oracle 資料庫執行個體類別，請參閱 RDS Custom for Oracle 的資料庫執行個體類別支援。

Architecture

目標技術堆疊

- Amazon RDS Custom for Oracle
- AWS Secrets Manager
- Oracle Active Data Guard
- Oracle PeopleSoft 應用程式

目標架構

下圖顯示 Amazon RDS Custom 資料庫執行個體和 Amazon RDS Custom 僅供讀取複本。僅供讀取複本使用 Oracle Active Data Guard 複寫到另一個可用區域。您也可以使用僅供讀取複本卸載主要資料庫上的讀取流量，並用於報告目的。

如需在 AWS 上使用 Oracle PeopleSoft 的代表性架構，請參閱在 AWS 上設定高可用性的 PeopleSoft 架構。

工具

AWS 服務

- Amazon RDS Custom for Oracle 是一種受管資料庫服務，適用於需要存取基礎作業系統和資料庫環境的舊版、自訂和封裝應用程式。
- AWS Secrets Manager 可協助您以 API 呼叫 Secrets Manager，以程式設計方式擷取秘密，取代程式碼中的硬式編碼登入資料，包括密碼。在此模式中，您會從秘密名稱`RDS_DATAGUARD`為 的

Secrets Manager 擷取資料庫使用者密碼`do-not-delete-rds-custom-+<<RDS Resource ID>>+-dg`。

其他工具

- Oracle Data Guard 可協助您建立、維護、管理和監控待命資料庫。

最佳實務

若要朝零資料遺失 (RPO=0) 目標運作，請使用 `MaxAvailability` Data Guard 保護模式，並使用重做傳輸`SYNC+NOAFFIRM`設定以獲得更好的效能。如需選取資料庫保護模式的詳細資訊，請參閱其他資訊一節。

史詩

建立僅供讀取複本

任務	Description	所需的技能
建立僅供讀取複本。	若要建立 Amazon RDS Custom 資料庫執行個體的僅供讀取複本，請遵循 Amazon RDS 文件中的指示，並使用您建立的 Amazon RDS Custom 資料庫執行個體（請參閱先決條件一節）做為來源資料庫。 根據預設，Amazon RDS Custom 僅供讀取複本會建立為實體待命，且處於掛載狀態。這是為了確保符合 Oracle Active Data Guard 授權。 此模式包含用於設定多租戶容器資料庫 (CDB) 或非 CDB 執行個體的程式碼。	DBA

將 Oracle Data Guard 保護模式變更為 MaxAvailability

任務	Description	所需的技能
存取主要資料庫上的 Data Guard 代理程式組態。	在此範例中，Amazon RDS Custom 僅供讀取複本RDS_CUSTOM_ORCL_D 適用於非 CDB 執行個體和 CDB RDS_CUSTOM_RDSCDB_B 執行個體。非 CDB 的資料庫為 orcl_a（主要）和 orcl_d（待命）。CDB 的資料庫名稱為 rdscdb_a（主要）和 rdscdb_b（待命）。 您可以直接或透過主要資料庫連線至 RDS Custom 僅供讀取複本。您可以在 $ORACLE_HOME/network/admin 目錄中的 tnsnames.ora 檔案中找到資料庫的淨服務名稱。RDS Custom for Oracle 會自動為您的主要資料庫和僅供讀取複本填入這些項目。 RDS_DATAGUARD 使用者的密碼存放在 AWS Secrets Manager 中，秘密名稱為 do-not-delete-rds-custom-+<<RDS Resource ID>>+-dg。如需如何使用從 Secrets Manager 擷取的 SSH（安全殼層）金鑰連線至 RDS Custom 執行個體的詳細資訊，請參閱使用 SSH 連線至 RDS Custom 資料庫執行個體。	DBA

任務	Description	所需的技能

	若要透過 Data Guard 命令列 (dgmgrl) 存取 Oracle Data Guard 代理程式組態，請使用下列程式碼。
	
	非 CDB
	
	`$ dgmgrl RDS_DATAG UARD@RDS_CUSTOM_OR CL_D`
	`DGMGRL for Linux:  Release 19.0.0.0.0 -  Production on Fri Sep  30 22:44:49 2022`
	`Version 19.10.0.0.0`
	`Copyright (c) 1982,  2019, Oracle and/or its  affiliates. All rights  reserved.`
	`Welcome to DGMGRL, type  "help" for informati on.`
	`Password:`
	`Connected to "ORCL_D"`
	`Connected as SYSDG.`
	`DGMGRL>`
	`DGMGRL> show database  orcl_d`
	`Database - orcl_d`
	`Role: PHYSICAL STANDBY`
	`Intended State: APPLY- ON`
	`Transport Lag: 0  seconds (computed 0  seconds ago)`
	`Apply Lag: 0 seconds  (computed 0 seconds  ago)`
	`Average Apply Rate:  11.00 KByte/s`
	`Instance(s):`	

任務	Description	所需的技能
	ORCL SUCCESS DGMGRL> CDB -bash-4.2$ dgmgrl C##RDS_DATAGUARD@R DS_CUSTOM_RDSCDB_B DGMGRL for Linux: Release 19.0.0.0.0 - Production on Wed Jan 11 20:24:11 2023 Version 19.16.0.0.0 Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved. Welcome to DGMGRL, type "help" for informati on. Password: Connected to "RDSCDB_B " Connected as SYSDG. DGMGRL> DGMGRL> show database rdscdb_b Database - rdscdb_b Role: PHYSICAL STANDBY Intended State: APPLY-ON Transport Lag: 0 seconds (computed 1 second ago) Apply Lag: 0 seconds (computed 1 second ago) Average Apply Rate: 2.00 KByte/s	

任務	Description	所需的技能
	Real Time Query: OFF Instance(s): RDSCDB Database Status: SUCCESS DGMGRL>	

任務	Description	所需的技能

從主節點連線至 DGMGRL 以變更日誌傳輸設定。	將日誌傳輸模式變更為 FastSync，對應於重做傳輸設定 SYNC+NOAFFIRM 。為了確保您在角色切換後擁有有效的設定，請同時變更主要資料庫和待命資料庫。
	
	非 CDB
	
	`DGMGRL>`
	`DGMGRL> edit database orcl_d set property logxptmode=fastsync;`
	`Property "logxptmode" updated`
	`DGMGRL> show database orcl_d LogXptMode;`
	`LogXptMode = 'fastsync'`
	`DGMGRL> edit database orcl_a set property logxptmode=fastsync;`
	`Property "logxptmode" updated`
	`DGMGRL> show database orcl_a logxptmode;`
	`LogXptMode = 'fastsync'`
	`DGMGRL>`
	
	CDB
	
	`DGMGRL> edit database rdscdb_b set property logxptmode=fastsyn c;DGMGRL> edit database rdscdb_b set property logxptmode=fastsync;`
		DBA

任務	Description	所需的技能
	Property "logxptmode" updated DGMGRL> show database rdscdb_b LogXptMode; LogXptMode = 'fastsync' DGMGRL> edit database rdscdb_a set property logxptmode=fastsync; Property "logxptmode" updated DGMGRL> show database rdscdb_a logxptmode; LogXptMode = 'fastsync' DGMGRL>	

任務	Description	所需的技能

將保護模式變更為 MaxAvailability。	DGMGRL 從主節點MaxAvailability 連線至 ，將保護模式變更為 。
	
	非 CDB
	
	`DGMGRL> edit configuration set protection mode as maxavailability;`
	`Succeeded.`
	`DGMGRL> show configuration;`
	`Configuration - rds_dg`
	`Protection Mode: MaxAvailability`
	`Members:`
	`orcl_a - Primary database`
	`orcl_d - Physical standby database`
	`Fast-Start Failover: Disabled`
	`Configuration Status:`
	`SUCCESS (status updated 38 seconds ago)`
	`DGMGRL>`
	
	CDB
	
	`DGMGRL> show configuration`
	`Configuration - rds_dg`
	`Protection Mode: MaxAvailability`
	`Members:`
	`rdscdb_a - Primary database`
	`rdscdb_b - Physical standby database`
		DBA

任務	Description	所需的技能
	Fast-Start Failover: Disabled Configuration Status: SUCCESS (status updated 57 seconds ago) DGMGRL>	

將複本狀態從掛載變更為唯讀，並啟用重做

任務	Description	所需的技能
停止重做會套用至待命資料庫。	依預設，僅供讀取複本會在 MOUNT 模式中建立。若要以唯讀模式開啟它，您必須先DGMGRL從主要節點或待命節點連線至 來關閉重做套用。 非 CDB DGMGRL> show database orcl_dDGMGRL> show database orcl_d Database - orcl_d Role: PHYSICAL STANDBY Intended State: APPLY- ON Transport Lag: 0 seconds (computed 1 second ago) Apply Lag: 0 seconds (computed 1 second ago) Average Apply Rate: 11.00 KByte/s Real Time Query: OFF Instance(s): ORCL Database Status: SUCCESS	DBA

任務	Description	所需的技能

```
DGMRL> edit database
 orcl_d set state=app
ly-off;
Succeeded.
DGMRL> show database
 orcl_d
Database - orcl_d
Role: PHYSICAL STANDBY
Intended State: APPLY-
OFF
Transport Lag: 0
 seconds (computed 1
 second ago)
Apply Lag: 42 seconds
 (computed 1 second ago)
Average Apply Rate:
 (unknown)
Real Time Query: OFF
Instance(s):
ORCL
Database Status:
SUCCESS
DGMRL>
```

CDB

```
DGMRL> show configura
tionDGMRL> show
 configuration
Configuration - rds_dg
  Protection Mode:
 MaxAvailability
  Members:
  rdscdb_a - Primary
 database
    rdscdb_b - Physical
 standby database
Fast-Start Failover:
 Disabled
Configuration Status:
```

任務	Description	所需的技能
	SUCCESS (status updated 57 seconds ago) DGMGRL> show database rdscdb_b; Database - rdscdb_b Role: PHYSICAL STANDBY Intended State: APPLY-ON Transport Lag: 0 seconds (computed 1 second ago) Apply Lag: 0 seconds (computed 1 second ago) Average Apply Rate: 2.00 KByte/s Real Time Query: OFF Instance(s): RDSCDB Database Status: SUCCESS DGMGRL> edit database rdscdb_b set state=apply-off; Succeeded. DGMGRL> show database rdscdb_b; Database - rdscdb_b Role: PHYSICAL STANDBY Intended State: APPLY-OFF Transport Lag: 0 seconds (computed 1 second ago) Apply Lag: 0 seconds (computed 1 second ago) Average Apply Rate: (unknown)	

任務	Description	所需的技能
	Real Time Query: OFF Instance(s): RDSCDB Database Status: SUCCESS	

任務	Description	所需的技能

以唯讀模式開啟僅供讀取複本執行個體。	使用 TNS 項目連接至待命資料庫，並從主要節點或待命節點連接至待命資料庫，以唯讀模式開啟該資料庫。
	
	非 CDB
	
	`$ sqlplus RDS_DATAGUARD@RDS_CUSTOM_ORCL_D as sysdg`
	`-bash-4.2$ sqlplus RDS_DATAGUARD@RDS_CUSTOM_ORCL_D as sysdg`
	`SQL*Plus: Release 19.0.0.0.0 - Production on Fri Sep 30 23:00:14 2022`
	`Version 19.10.0.0.0`
	`Copyright (c) 1982, 2020, Oracle. All rights reserved.`
	`Enter password:`
	`Last Successful login time: Fri Sep 30 2022 22:48:27 +00:00`
	`Connected to:`
	`Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production`
	`Version 19.10.0.0.0`
	`SQL> select open_mode from v$database;`
	`OPEN_MODE`
	`--------------------`
	`MOUNTED`
	`SQL> alter database open read only;`
	`Database altered.`
		DBA

任務	Description	所需的技能
	SQL> select open_mode from v$database; OPEN_MODE -------------------- READ ONLY SQL> CDB -bash-4.2$ sqlplus C##RDS_DATAGUARD@R DS_CUSTOM_RDSCDB_B as sysdg SQL*Plus: Release 19.0.0.0.0 - Productio n on Wed Jan 11 21:14:07 2023 Version 19.16.0.0.0 Copyright (c) 1982, 2022, Oracle. All rights reserved. Enter password: Last Successful login time: Wed Jan 11 2023 21:12:05 +00:00 Connected to: Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production Version 19.16.0.0.0 SQL> select name,open _mode from v$database; NAME OPEN_MODE --------- --------- ----------- RDSCDB MOUNTED SQL> alter database open read only; Database altered.	

任務	Description	所需的技能
	SQL> select name,open _mode from v$database; NAME OPEN_MODE --------- --------- ----------- RDSCDB READ ONLY SQL>	

任務	Description	所需的技能

在僅供讀取複本執行個體上啟用重做。	使用主要節點或待命節點中的 DGMGRL，在僅供讀取複本執行個體上啟用重做。
	
	非 CDB
	
	`$ dgmgrl RDS_DATAG UARD@RDS_CUSTOM_OR CL_D`
	`DGMGRL for Linux: Release 19.0.0.0.0 - Production on Fri Sep 30 23:02:16 2022`
	`Version 19.10.0.0.0`
	`Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved.`
	`Welcome to DGMGRL, type "help" for informati on.`
	`Password:`
	`Connected to "ORCL_D"`
	`Connected as SYSDG.`
	`DGMGRL>`
	`edit database orcl_d set state=apply-on;`
	`DGMGRL> edit database orcl_d set state=app ly-on;`
	`Succeeded.`
	`DGMGRL> show database orcl_d`
	`Database - orcl_d`
	`Role: PHYSICAL STANDBY`
	`Intended State: APPLY-ON`
	`Transport Lag: 0 seconds (computed 0 seconds ago)`
		DBA

任務	Description	所需的技能

```
Apply Lag: 0 seconds
 (computed 0 seconds
 ago)
Average Apply Rate:
 496.00 KByte/s
Real Time Query: ON
Instance(s):
ORCL
Database Status:
SUCCESS
DGMGRL>
```

CDB

```
-bash-4.2$ dgmgrl
 C##RDS_DATAGUARD@R
DS_CUSTOM_RDSCDB_B
-bash-4.2$ dgmgrl
 C##RDS_DATAGUARD@R
DS_CUSTOM_RDSCDB_B
DGMGRL for Linux:
 Release 19.0.0.0.0 -
 Production on Wed Jan
 11 21:21:11 2023
Version 19.16.0.0.0
Copyright (c) 1982,
 2019, Oracle and/or
 its affiliates.  All
 rights reserved.
Welcome to DGMGRL, type
 "help" for informati
on.
Password:
Connected to "RDSCDB_B
"
Connected as SYSDG.
DGMGRL> edit database
 rdscdb_b set state=app
ly-on;
Succeeded.
```

任務	Description	所需的技能
	DGMGRL> show database rdscdb_b Database - rdscdb_b Role: PHYSICAL STANDBY Intended State: APPLY-ON Transport Lag: 0 seconds (computed 0 seconds ago) Apply Lag: 0 seconds (computed 0 seconds ago) Average Apply Rate: 35.00 KByte/s Real Time Query: ON Instance(s): RDSCDB Database Status: SUCCESS DGMGRL> show database rdscdb_b Database - rdscdb_b Role: PHYSICAL STANDBY Intended State: APPLY-ON Transport Lag: 0 seconds (computed 1 second ago) Apply Lag: 0 seconds (computed 1 second ago) Average Apply Rate: 16.00 KByte/s Real Time Query: ON Instance(s): RDSCDB Database Status: SUCCESS DGMGRL>	

相關資源

- 將 Amazon RDS 設定為 Oracle PeopleSoft 資料庫 (AWS 白皮書)
- Oracle Data Guard Broker 指南 (Oracle 參考文件)
- Data Guard 概念和管理 (Oracle 參考文件)

其他資訊

選取您的資料庫保護模式

Oracle Data Guard 提供三種保護模式，根據您的可用性、保護和效能需求來設定 Data Guard 環境。下表摘要說明這三種模式。

保護模式	重做傳輸設定	Description
最大效能	ASYNC	對於主要資料庫上發生的交易，重做資料會以非同步方式傳輸並寫入待命資料庫重做日誌。因此，效能影響極小。 MaxPerformance 由於非同步日誌運送， 無法提供 RPO=0。
最大保護	SYNC+AFFIRM	對於主要資料庫上的交易，在確認交易之前，重做資料會同步傳輸並寫入待命資料庫重做磁碟上的日誌。如果待命資料庫無法使用，主要資料庫會自行關閉，以確保交易受到保護。
最大可用性	SYNC+AFFIRM	這類似於 MaxProtection 模式，除非未收到來自待命資料庫的確認。在這種情況下，它會像處於 MaxPerformance 模式一樣運作，以保留主要資料庫的可用性，直到

	能夠將其重做串流再次寫入同步待命資料庫為止。
SYNC+NOAFFIRM	對於主要資料庫上的交易，重做會同步傳輸至待命資料庫，而主要只會等待確認已在待命上收到重做，而不是已寫入待命磁碟。此模式也稱為 FastSync，可在發生多個同時故障的特殊情況下，提供可能暴露於資料遺失的效能優勢。

RDS Custom for Oracle 中的僅供讀取複本會以最大效能保護模式建立，這也是 Oracle Data Guard 的預設保護模式。最大效能模式可為主要資料庫提供最低的效能影響，這可協助您符合以秒為單位測量的復原點目標 (RPO) 需求。

若要實現零資料遺失 (RPO=0) 目標，您可以自訂 Oracle Data Guard 保護模式為 MaxAvailability，並使用 SYNC+NOAFFIRM設定進行重做傳輸以獲得更好的效能。由於只有在對應的重做向量成功傳輸到待命資料庫之後，才會確認主要資料庫上的遞交，因此主要執行個體和複本之間的網路延遲對於對遞交敏感的工作負載至關重要。我們建議您為工作負載執行負載測試，以評估在僅供讀取複本自訂為在 MaxAvailability 模式下執行時的效能影響。

與在不同可用區域中部署僅供讀取複本相比，在與主要資料庫相同的可用區域中部署僅供讀取複本可提供較低的網路延遲。不過，在相同可用區域中部署主要和僅供讀取複本可能不符合您的 HA 需求，因為在極少數的可用區域無法使用的情況下，主要執行個體和僅供讀取複本執行個體都會受到影響。

分析從 Oracle 到 PostgreSQL 的部分資料庫遷移的物件相依性

anuradha chintha，Amazon Web Services

總結

此模式說明將部分 Oracle 資料庫遷移至 Amazon Relational Database Service (Amazon RDS) 或 Amazon Aurora PostgreSQL 時，系統識別和管理系統相依性的重要性。在部分遷移中，只會遷移來自原始資料庫的資料庫物件和資料子集，而來源資料庫會繼續操作和提供依賴非遷移元件的應用程式

處理具有上游和下游相依性之緊密耦合應用程式的大規模資料庫時，您必須識別和分析遷移的範圍。若要開始部分遷移，請識別範圍物件，包括資料表、觸發條件、檢視、預存程序、函數和套件。範圍識別程序遵循全方位的方法：

- 第一層範圍物件是透過應用程式程式碼和關鍵模組特定任務中的直接參考來識別。
- 第二層物件是透過全面的相依性分析衍生的。

當您了解系統的不同部分如何互動時，您可以更好地規劃移動資料庫元件的正確順序，並降低遷移失敗的風險。下表列出不同類型的相依性分析。

分析類型	重點領域	用途
物件相依性	• 表格 • 檢視 • 預存程序 • 函數 • 觸發	識別資料庫物件與其階層結構之間的關係
區段相依性	• 外部金鑰關係 • 主要金鑰鏈 • 跨結構描述參考	映射資料關係並維護參考完整性
安全相依性	• 使用者許可 • 角色階層 • 物件權限	確保適當的存取控制遷移和安全性維護
存取模式	• 讀取操作	決定資料庫互動模式

- 寫入操作

若要維持來源和目標系統之間的一致性，請在轉換期間建立資料同步機制。您還必須修改應用程式程式碼和函數，以處理來源 Oracle 和目標 PostgreSQL 資料庫之間的資料分佈。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- Oracle 資料庫（來源）
- Amazon RDS 或 Amazon Aurora PostgreSQL（目標）

產品版本

- Oracle 19c 或更新版本
- PostgreSQL 16 或更新版本

Architecture

來源技術堆疊

- Oracle 19c 或更新版本

目標技術堆疊

- Amazon RDS 或 Amazon Aurora PostgreSQL

目標架構

下圖顯示從內部部署 Oracle 資料庫到 Amazon RDS for Oracle 的遷移程序，其中包含下列項目：

- 識別資料庫相依性
- 使用 AWS Schema Conversion Tool (AWS SCT) 遷移資料庫程式碼和物件
- 使用 AWS Database Migration Service (AWS DMS) 遷移資料
- 使用 透過變更資料擷取 (CDC) 複寫持續變更 AWS DMS

如需詳細資訊，請參閱 AWS 文件中的整合 AWS Database Migration Service 與 AWS Schema Conversion Tool 。

工具

AWS 服務

- 適用於 Oracle 的 Amazon Relational Database Service (Amazon RDS) 可協助您在 中設定、操作和擴展 Oracle 關聯式資料庫 AWS 雲端。
- Amazon Aurora 是一種全受管關聯式資料庫引擎，專為雲端而建置，並與 MySQL 和 PostgreSQL 相容。
- AWS Schema Conversion Tool (AWS SCT) 透過自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，支援異質資料庫遷移。
- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移到 AWS 雲端 或在雲端和內部部署設定的組合之間遷移。

其他服務

- Oracle SQL Developer 是一種整合的開發環境，可簡化傳統和雲端部署中 Oracle 資料庫的開發和管理。對於此模式，您可以使用 SQL*Plus。

最佳實務

如需佈建和遷移 Oracle 資料庫的最佳實務，請參閱遷移至 Amazon RDS for Oracle 的最佳實務。

史詩

識別物件相依性

任務	Description	所需的技能
建立物件資料表。	識別對應用程式功能至關重要的物件，並建立名為 的資料表`DEPENDENT_ANALYSIS_BASELINE` 。將每個物件	資料工程師，DBA

任務	Description	所需的技能
	的記錄新增至資料表。如需範例，請參閱其他資訊一節。	
建立資料庫程序。	建立名為 的預存程序`sp_object_dependency_analysis` ，使用`DBA_DEPENDENCIES` 資料表中的資料以雙向（向前和向後）分析物件相依性。如需範例，請參閱其他資訊一節。	資料工程師，DBA
執行程序。	在每個連續層級執行指令碼，直到找不到新的物件相依性為止。所有相依性和關卡都存放在 `DEPENDENT_ANALYSIS_BASELINE` 資料表中。	DBA，資料工程師

建立區段層級相依性的程序

任務	Description	所需的技能
建立相依性資料表。	建立名為 的區段層級相依性資料表`REFERENTIAL_ANALYSIS_BASELINE` 。發現所有物件層級相依性時，請透過查詢資料表`DEPENDENT_ANALYSIS_BASELINE` 來檢查 的父`DBA_CONSTRAINT` 資料表。 排除其他資料表參考基準資料表的相依性。回填會處理這些關係。以下是範例指令碼：	資料工程師，DBA

任務	Description	所需的技能
	`CREATE TABLE REFERENTIAL_ANALYSIS_BASELINE (CHILD_OWNER VARCHAR2(50 BYTE), CHILD_NAME VARCHAR2(100 BYTE), PARENT_OWNER VARCHAR2(50 BYTE), PARENT_NAME VARCHAR2(50 BYTE), REFERENCE_PATH VARCHAR2(1000 BYTE));`	
建立資料庫程序。	建立名為 的程序SP_OBJECT_REFERENTIAL_ANALYSIS ，並產生所有已識別物件的參考分析。如需範例，請參閱其他資訊一節。	資料工程師，DBA
執行程序。	執行 程序以取得參考相依性。在 中產生參考分析物件詳細資訊REFERENTIAL_ANALYSIS_BASELINE 。	資料工程師，DBA

識別讀取和寫入的物件

任務	Description	所需的技能
建立讀取和寫入物件的資料表。	使用下列指令碼建立名為 的讀取物件資料表TABLE_READ_OBJECT_DETAILS 和名為 的寫入物件資料表TABLE_WRITE_OBJECT_DETAILS ：	資料工程師，DBA

任務	Description	所需的技能

	`CREATE TABLE TABLE_READ_OBJECT_DETAILS (OWNER VARCHAR2(50 BYTE), TAB_NAME VARCHAR2(50 BYTE), READER_OWNER VARCHAR2(50 BYTE), READER_NAME VARCHAR2(50 BYTE), READER_TYPE VARCHAR2(50 BYTE));`
	
	`CREATE TABLE TABLE_WRITE_OBJECT_DETAILS (TABLE_NAME VARCHAR2(100 BYTE), WRITEOBJ_OWNER VARCHAR2(100 BYTE), WRITEOBJ_NAME VARCHAR2(100 BYTE), WRITEOBJ_TYPE VARCHAR2(100 BYTE), LINE VARCHAR2(100 BYTE), TEXT VARCHAR2(4000 BYTE), OWNER VARCHAR2(50 BYTE));`	

建立分析程序。	分別建立SP_WRITER_OBJECTS_ANALYSIS 用於分析讀取物件和寫入物件的程序 SP_READER_OBJECTS_ANALYSIS 和 。這些程序使用模式比對來尋找相關物件。如需範例，請參閱其他資訊一節。	資料工程師，DBA

任務	Description	所需的技能
執行程序。	執行這些程序來識別相依物件。	DBA，資料工程師

檢閱資料庫權限

任務	Description	所需的技能
建立用於檢閱權限的資料表。	建立資料表以分析名為 的權限OBJECT_PRIVS_ANALYSIS 。若要以遞迴方式擷取DEPENDENT_ANALYSIS_BASELINE 資料表中的物件權限，請使用下列指令碼： `CREATE TABLE OBJECT_PRIVS_ANALYSIS (OWNER VARCHAR2(50 BYTE), OBJECT_NAME VARCHAR2(50 BYTE), USER_NAME VARCHAR2(50 BYTE), PRIVS VARCHAR2(50 BYTE));`	資料工程師，DBA
建立檢閱權限的程序。	建立名為 的程序SP_OBJECT_PRIVS_ANALYSIS 。產生已識別物件的權限分析。如需範例，請參閱其他資訊一節。	DBA，資料工程師
執行 程序。	執行 程序，在 OBJECT_PRIVS_ANALYSIS 資料表中擷取它們。	DBA，資料工程師

疑難排解

問題	解決方案
無法存取字典資料表	確保建立分析物件的使用者可以存取 DBA 資料表。

相關資源

AWS 文件

- Amazon RDS 和 Aurora 文件
- Oracle 資料庫 19c 至 Amazon Aurora PostgreSQL 遷移手冊
- 什麼是 AWS Database Migration Service？
- 什麼是 AWS Schema Conversion Tool？

其他文件

- Oracle 資料庫物件

其他資訊

的指令碼 **DEPENDENT_ANALYSIS_BASELINE**

```
CREATE TABLE DEPENDENT_ANALYSIS_BASELINE
(OWNER VARCHAR2(128 BYTE) NOT NULL ENABLE,
OBJECT_NAME VARCHAR2(128 BYTE) NOT NULL ENABLE,
OBJECT_TYPE VARCHAR2(20 BYTE),
DEPEDNCY_LEVEL NUMBER,
PROJECT_NEED VARCHAR2(20 BYTE),
CATAGORY VARCHAR2(4000 BYTE),
COMMENTS VARCHAR2(4000 BYTE),
CATAGORY1 CLOB,
COMMENTS1 CLOB,
CUSTOMER_COMMENTS VARCHAR2(1000 BYTE),
BACKFILL_TO_GUS VARCHAR2(1000 BYTE),
BACKFILL_NEAR_REAL_TIME_OR_BATCH VARCHAR2(1000 BYTE),
```

```
PK_EXISTS VARCHAR2(3 BYTE),
UI_EXISTS VARCHAR2(3 BYTE),
LOB_EXISTS VARCHAR2(3 BYTE),
MASTER_LINK VARCHAR2(100 BYTE),
CONSTRAINT PK_DEPENDENT_ANALYSIS_BASELINE PRIMARY KEY (OWNER,OBJECT_NAME,OBJECT_TYPE));
```

的程序 SP_WRITER_OBJECTS_ANALYSIS

```
CREATE OR REPLACE PROCEDURE SP_WRITER_OBJECTS_ANALYSIS IS
BEGIN
  EXECUTE IMMEDIATE 'TRUNCATE TABLE TABLE_WRITE_OBJECT_DETAILS';
  FOR I IN (SELECT OWNER, OBJECT_NAME FROM DEPENDENT_ANALYSIS_BASELINE WHERE
 OBJECT_TYPE = 'TABLE')
  LOOP
    INSERT INTO TABLE_WRITE_OBJECT_DETAILS(OWNER, TABLE_NAME, WRITEOBJ_OWNER,
 WRITEOBJ_NAME, WRITEOBJ_TYPE, LINE, TEXT)
    SELECT DISTINCT I.OWNER, I.OBJECT_NAME, OWNER WRITEOBJ_OWNER, NAME, TYPE, LINE,
 TRIM(TEXT)
    FROM DBA_SOURCE
    WHERE UPPER(TEXT) LIKE '%' || I.OBJECT_NAME || '%'
      AND (UPPER(TEXT) LIKE '%INSERT%' || I.OBJECT_NAME || '%'
        OR UPPER(TEXT) LIKE '%UPDATE%' || I.OBJECT_NAME || '%'
        OR UPPER(TEXT) LIKE '%DELETE%' || I.OBJECT_NAME || '%'
        OR UPPER(TEXT) LIKE '%UPSERT%' || I.OBJECT_NAME || '%'
        OR UPPER(TEXT) LIKE '%MERGE%' || I.OBJECT_NAME || '%')
      AND UPPER(TEXT) NOT LIKE '%PROCEDURE%'
      AND UPPER(TEXT) NOT LIKE 'PROCEDURE%'
      AND UPPER(TEXT) NOT LIKE '%FUNCTION%'
      AND UPPER(TEXT) NOT LIKE 'FUNCTION%'
      AND UPPER(TEXT) NOT LIKE '%TRIGGER%'
      AND UPPER(TEXT) NOT LIKE 'TRIGGER%'
      AND UPPER(TRIM(TEXT)) NOT LIKE '%AFTER UPDATE%'
      AND UPPER(TRIM(TEXT)) NOT LIKE 'BEFORE UPDATE%'
      AND UPPER(TRIM(TEXT)) NOT LIKE 'BEFORE INSERT%'
      AND UPPER(TRIM(TEXT)) NOT LIKE 'AFTER INSERT%'
      AND UPPER(TRIM(TEXT)) NOT LIKE 'BEFORE DELETE%'
      AND UPPER(TRIM(TEXT)) NOT LIKE 'AFTER DELETE%'
      AND UPPER(TRIM(TEXT)) NOT LIKE '%GGLOGADM.GG_LOG_ERROR%'
      AND (TRIM(TEXT) NOT LIKE '/*%' AND TRIM(TEXT) NOT LIKE '--%' )
      AND (OWNER, NAME, TYPE) IN (SELECT OWNER, NAME, TYPE FROM DBA_DEPENDENCIES WHERE
 REFERENCED_NAME = I.OBJECT_NAME);
  END LOOP;
END;
```

的指令碼 SP_READER_OBJECTS_ANALYSIS

```
CREATE OR REPLACE PROCEDURE SP_READER_OBJECTS_ANALYSIS IS
BEGIN
  EXECUTE IMMEDIATE 'TRUNCATE TABLE TABLE_READ_OBJECT_DETAILS';
  FOR I IN (SELECT OWNER, OBJECT_NAME FROM DEPENDENT_ANALYSIS_BASELINE WHERE
 OBJECT_TYPE = 'TABLE')
  LOOP
    INSERT INTO TABLE_READ_OBJECT_DETAILS
    SELECT DISTINCT i.owner, i.object_name, owner, name, type
    FROM dba_dependencies
    WHERE referenced_name = I.OBJECT_NAME
    AND referenced_type = 'TABLE'
    AND type NOT IN ('SYNONYM', 'MATERIALIZED VIEW', 'VIEW')
    AND (owner, name, type) NOT IN (
      SELECT DISTINCT owner, trigger_name, 'TRIGGER'
      FROM dba_triggers
      WHERE table_name = I.OBJECT_NAME
      AND table_owner = i.owner
      UNION ALL
      SELECT DISTINCT owner, name, type
      FROM dba_source
      WHERE upper(text) LIKE '%' || I.OBJECT_NAME || '%'
      AND (upper(text) LIKE '%INSERT %' || I.OBJECT_NAME || '%'
        OR upper(text) LIKE '%UPDATE% ' || I.OBJECT_NAME || '%'
        OR upper(text) LIKE '%DELETE %' || I.OBJECT_NAME || '%'
        OR upper(text) LIKE '%UPSERT %' || I.OBJECT_NAME || '%'
        OR upper(text) LIKE '%MERGE %' || I.OBJECT_NAME || '%')
      AND upper(text) NOT LIKE '%PROCEDURE %'
      AND upper(text) NOT LIKE 'PROCEDURE %'
      AND upper(text) NOT LIKE '%FUNCTION %'
      AND upper(text) NOT LIKE 'FUNCTION %'
      AND upper(text) NOT LIKE '%TRIGGER %'
      AND upper(text) NOT LIKE 'TRIGGER %'
      AND upper(trim(text)) NOT LIKE 'BEFORE INSERT %'
      AND upper(trim(text)) NOT LIKE 'BEFORE UPDATE %'
      AND upper(trim(text)) NOT LIKE 'BEFORE DELETE %'
      AND upper(trim(text)) NOT LIKE 'AFTER INSERT %'
      AND upper(trim(text)) NOT LIKE 'AFTER UPDATE %'
      AND upper(trim(text)) NOT LIKE 'AFTER DELETE %'
      AND (trim(text) NOT LIKE '/*%' AND trim(text) NOT LIKE '--%'));
  END LOOP;
END;
```

的指令碼 SP_OBJECT_REFERENTIAL_ANALYSIS

```
CREATE OR REPLACE PROCEDURE SP_OBJECT_REFERENTIAL_ANALYSIS IS
BEGIN
  EXECUTE IMMEDIATE 'TRUNCATE TABLE REFERENTIAL_ANALYSIS_BASELINE';
  INSERT INTO REFERENTIAL_ANALYSIS_BASELINE
  WITH rel AS (
    SELECT DISTINCT c.owner, c.table_name, c.r_owner r_owner,
      (SELECT table_name FROM dba_constraints
       WHERE constraint_name = c.r_constraint_name
       AND owner = c.r_owner) r_table_name
    FROM dba_constraints c
    WHERE constraint_type = 'R'
    AND c.owner NOT IN (SELECT username FROM dba_users WHERE oracle_maintained = 'Y')
    AND c.r_owner NOT IN (SELECT username FROM dba_users WHERE oracle_maintained =
 'Y')),
  tab_list AS (
    SELECT OWNER, object_name
    FROM DEPENDENT_ANALYSIS_BASELINE
    WHERE UPPER(OBJECT_TYPE) = 'TABLE')
  SELECT DISTINCT owner child_owner, table_name child, r_owner parent_owner,
    r_table_name parent, SYS_CONNECT_BY_PATH(r_table_name, ' -> ') || ' -> ' ||
 table_name PATH
  FROM rel
  START WITH (r_owner, r_table_name) IN (SELECT * FROM tab_list)
  CONNECT BY NOCYCLE (r_owner, r_table_name) = ((PRIOR owner, PRIOR table_name))
  UNION
  SELECT DISTINCT owner child_owner, table_name child, r_owner parent_owner,
    r_table_name parent, SYS_CONNECT_BY_PATH(table_name, ' -> ') || ' -> ' ||
 r_table_name PATH
  FROM rel
  START WITH (owner, table_name) IN (SELECT * FROM tab_list)
  CONNECT BY NOCYCLE (owner, table_name) = ((PRIOR r_owner, PRIOR r_table_name));
END;
```

的指令碼 SP_OBJECT_PRIVS_ANALYSIS

```
CREATE OR REPLACE PROCEDURE SP_OBJECT_PRIVS_ANALYSIS IS
  V_SQL VARCHAR2(4000);
  V_CNT NUMBER;
BEGIN
  V_SQL := 'TRUNCATE TABLE OBJECT_PRIVS_ANALYSIS';
  EXECUTE IMMEDIATE V_SQL;
```

```
 FOR I IN (SELECT OWNER, OBJECT_NAME FROM DEPENDENT_ANALYSIS_BASELINE WHERE
OBJECT_TYPE = 'TABLE')
 LOOP
   INSERT INTO OBJECT_PRIVS_ANALYSIS(OWNER, OBJECT_NAME, USER_NAME, PRIVS)
   WITH obj_to_role AS (
     SELECT DISTINCT GRANTEE role_name,
       DECODE(privilege, 'SELECT', 'READ', 'REFERENCE', 'READ', 'INSERT', 'WRITE',
              'UPDATE', 'WRITE', 'DELETE', 'WRITE', privilege) privs
     FROM DBA_TAB_PRIVS t, DBA_ROLES r
     WHERE OWNER = I.OWNER
     AND TYPE = 'TABLE'
     AND TABLE_NAME = I.OBJECT_NAME
     AND t.GRANTEE = r.ROLE
     AND r.ROLE IN (SELECT ROLE FROM DBA_ROLES WHERE ORACLE_MAINTAINED = 'N')
   )
   SELECT I.OWNER, I.OBJECT_NAME, grantee, privs
   FROM (
     -- Recursively Role to User mapping with privilege
     SELECT DISTINCT grantee, privs
     FROM (SELECT rp.granted_role, rp.grantee, privs,
       (SELECT DECODE(COUNT(*), 0, 'ROLE', 'USER')
        FROM (SELECT 'User' FROM DBA_users WHERE username = rp.GRANTEE)) grantee_type
       FROM DBA_role_privs rp, obj_to_role r
       WHERE rp.granted_role = r.role_name
       AND grantee IN ((SELECT USERNAME FROM DBA_USERS WHERE ORACLE_MAINTAINED = 'N')
                       UNION (SELECT ROLE FROM DBA_ROLES WHERE ORACLE_MAINTAINED =
'N'))
       AND granted_role IN (SELECT ROLE FROM DBA_ROLES WHERE ORACLE_MAINTAINED = 'N')
       START WITH granted_role IN (SELECT DISTINCT role_name FROM obj_to_role)
       CONNECT BY granted_role = PRIOR grantee)
     WHERE grantee_type = 'USER'
   )
   UNION
   (
     -- Direct Object grants to User
     SELECT I.OWNER, I.OBJECT_NAME, GRANTEE,
       DECODE(privilege, 'SELECT', 'READ', 'REFERENCE', 'READ', 'INSERT', 'WRITE',
              'UPDATE', 'WRITE', 'DELETE', 'WRITE', privilege) privs
     FROM DBA_TAB_PRIVS, DBA_USERS
     WHERE GRANTEE = USERNAME
     AND OWNER = I.OWNER
     AND TYPE = 'TABLE'
     AND TABLE_NAME = I.OBJECT_NAME
   )
```

```
    ORDER BY 2 DESC;
  END LOOP;
END;
```

的程序 SP_OBJECT_DEPENDENCY_ANALYSIS

```
CREATE OR REPLACE PROCEDURE SP_OBJECT_DEPENDENCY_ANALYSIS (v_level NUMBER) IS
  TYPE typ IS RECORD (
    schema VARCHAR2(100),
    obj_type VARCHAR2(100),
    obj_name VARCHAR2(100),
    path VARCHAR2(5000)
  );
  TYPE array IS TABLE OF typ;
  l_data array;
  c SYS_REFCURSOR;
  l_errors NUMBER;
  l_errno NUMBER;
  l_msg VARCHAR2(4000);
  l_idx NUMBER;
  l_level NUMBER;
BEGIN
  l_level := v_level + 1;
  OPEN c FOR
    WITH obj_list AS (
      SELECT owner schema_name, object_type, object_name
      FROM DEPENDENT_ANALYSIS_BASELINE
      WHERE depedncy_level = v_level
    ),
    fw_dep_objects AS (
      SELECT level lvl, owner, name, type, referenced_owner, referenced_name,
        referenced_type, SYS_CONNECT_BY_PATH(name, ' -> ') || ' -> ' || referenced_name
 PATH
      FROM dba_dependencies
      START WITH (owner, CASE WHEN type = 'PACKAGE BODY' THEN 'PACKAGE' ELSE type END,
 name)
        IN (SELECT schema_name, object_type, object_name FROM obj_list)
      CONNECT BY NOCYCLE (owner, type, name) =
        ((PRIOR referenced_owner, PRIOR referenced_type, PRIOR referenced_name))
    ),
    bw_dep_objects AS (
      SELECT level lvl, owner, name, type, referenced_owner, referenced_name,
```

```
       referenced_type, SYS_CONNECT_BY_PATH(name, ' <- ') || ' <- ' || referenced_name
PATH
     FROM dba_dependencies
     START WITH (referenced_owner, CASE WHEN referenced_type = 'PACKAGE BODY' THEN
'PACKAGE'
       ELSE referenced_type END, referenced_name) IN (SELECT schema_name, object_type,
object_name FROM obj_list)
     CONNECT BY NOCYCLE (referenced_owner, referenced_type, referenced_name) =
       ((PRIOR owner, PRIOR type, PRIOR name))
   )
   SELECT * FROM (
     (SELECT DISTINCT referenced_owner schema, referenced_type obj_type,
       referenced_name obj_name, path FROM fw_dep_objects)
     UNION
     (SELECT DISTINCT owner schema, type obj_type, name obj_name, path
      FROM bw_dep_objects)
   )
   WHERE schema IN (SELECT username FROM all_users WHERE oracle_maintained = 'N')
   ORDER BY obj_type;

 LOOP
   FETCH c BULK COLLECT INTO l_data LIMIT 100;
   BEGIN
     FORALL i IN 1..l_data.count SAVE EXCEPTIONS
       INSERT INTO DEPENDENT_ANALYSIS_BASELINE (
         owner, object_name, object_type, catagory, depedncy_level, project_need,
comments
       )
       VALUES (
         l_data(i).schema,
         l_data(i).obj_name,
         CASE WHEN l_data(i).obj_type = 'PACKAGE BODY' THEN 'PACKAGE' ELSE
l_data(i).obj_type END,
         'level ' || l_level || ' dependency',
         l_level,
         '',
         'from dependency proc' || l_data(i).path
       );
   EXCEPTION
     WHEN OTHERS THEN
       l_errors := sql%bulk_exceptions.count;
       FOR i IN 1..l_errors LOOP
         l_errno := sql%bulk_exceptions(i).error_code;
         l_msg := SQLERRM(-l_errno);
```

```
          l_idx := sql%bulk_exceptions(i).error_index;
          UPDATE DEPENDENT_ANALYSIS_BASELINE
          SET catagory1 = catagory1 || ', found in level' || l_level || ' dependent of
 ' || l_data(l_idx).path,
              comments1 = comments1 || ', from dependency proc exception ' ||
 l_data(i).path
          WHERE owner = l_data(l_idx).schema
          AND object_name = l_data(l_idx).obj_name
          AND object_type = l_data(l_idx).obj_type;
        END LOOP;
    END;
    EXIT WHEN c%NOTFOUND;
  END LOOP;
  CLOSE c;
END;
```

評估將 SQL Server 資料庫遷移至 AWS 上的 MongoDB Atlas 的查詢效能

Battulga Purevragchaa，Amazon Web Services

Krishnakumar Sathyanarayana，PeerIslands 美國公司

Babu Srinivasan，MongoDB

摘要

此模式提供使用接近真實世界的資料載入 MongoDB，以及評估 MongoDB 查詢效能的指引，盡可能接近生產案例。評估提供輸入，協助您規劃從關聯式資料庫遷移至 MongoDB。此模式使用 PeerIslands Test Data Generator 和 Performance Analyzer 來測試查詢效能。

此模式對於 Microsoft SQL Server 遷移至 MongoDB 特別有用，因為執行結構描述轉換並將資料從目前的 SQL Server 執行個體載入至 MongoDB 可能非常複雜。相反地，您可以將接近真實世界的資料載入 MongoDB、了解 MongoDB 效能，並在開始實際遷移之前微調結構描述設計。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 熟悉 MongoDB Atlas
- 目標 MongoDB 結構描述
- 典型查詢模式

限制

- 資料載入時間和效能會受到 MongoDB 叢集執行個體大小的限制。我們建議您選擇建議用於生產用途的執行個體，以了解實際效能。
- PeerIslands Test Data Generator and Performance Analyzer 目前僅支援線上資料載入和查詢。尚未支援離線批次處理（例如，使用 Spark 連接器將資料載入 MongoDB)。
- PeerIslands Test Data Generator and Performance Analyzer 支援集合中的欄位關係。它不支援跨集合的關係。

產品版本

- 此模式同時支援 MongoDB Atlas 和 MongoDB Enterprise Advanced。

Architecture

目標技術堆疊

- MongoDB Atlas 或 MongoDB Enterprise Advanced

架構

PeerIslands Test Data Generator and Performance Analyzer 是使用 Java 和 Angular 建置的，並將產生的資料存放在 Amazon Elastic Block Store (Amazon EBS)。此工具包含兩個工作流程：測試資料產生和效能測試。

- 在測試資料產生中，您會建立範本，這是必須產生之資料模型的 JSON 表示法。建立範本後，您可以在目標集合中產生資料，如負載產生組態所定義。
- 在效能測試中，您會建立設定檔。設定檔是一種多階段測試案例，您可以在其中設定建立、讀取、更新和刪除 (CRUD) 操作、彙總管道、每個操作的權重，以及每個階段的持續時間。建立設定檔之後，您可以根據組態在目標資料庫上執行效能測試。

PeerIslands Test Data Generator and Performance Analyzer 將其資料存放在 Amazon EBS 上，因此您可以使用任何 MongoDB 支援的連線機制，包括對等互連、允許清單和私有端點，將 Amazon EBS 連線至 MongoDB。根據預設，工具不包含操作元件；不過，如果需要，可以使用 Amazon Managed Service for Prometheus、Amazon Managed Grafana、Amazon CloudWatch 和 AWS Secrets Manager 進行設定。

工具

- PeerIslands Test Data Generator and Performance Analyzer 包含兩個元件。測試資料產生器元件可協助您根據 MongoDB 結構描述產生高度客戶特定的真實世界資料。該工具完全由 UI 驅動，具有豐富的資料程式庫，可用於在 MongoDB 上快速產生數十億筆記錄。此工具也提供在 MongoDB 結構描述中的欄位之間實作關係的功能。Performance Analyzer 元件可協助您產生高度客戶特定的查詢和彙總，並在 MongoDB 上執行逼真的效能測試。您可以使用 Performance Analyzer，針對特定使用案例使用豐富的負載描述檔和參數化查詢來測試 MongoDB 效能。

最佳實務

請參閱下列資源：

- MongoDB 結構描述設計最佳實務 (MongoDB 開發人員網站）
- 在 AWS 上部署 MongoDB Atlas 的最佳實務 (MongoDB 網站）
- 使用 AWS PrivateLink 將應用程式安全地連接到 MongoDB Atlas 資料平面 (AWS 部落格文章）
- MongoDB 效能的最佳實務指南 (MongoDB 網站）

史詩

了解您的來源資料

任務	Description	所需的技能
了解目前 SQL Server 來源的資料庫使用量。	了解您目前的 SQL Server 使用量。這可以透過對資料庫的`INFORMATION` 結構描述執行查詢來實現。決定資料表的數量和每個資料表的大小。分析與每個資料表相關聯的索引。如需 SQL 分析的詳細資訊，請參閱 PeerIslands 網站上的部落格文章 SQL2Mongo：Data Migration Journey。	DBA
了解來源結構描述。	決定資料表結構描述和資料的業務表示方式（例如，郵遞區號、名稱和貨幣）。使用您現有的實體關係 (ER) 圖表，或從現有的資料庫產生 ER 圖表。如需詳細資訊，請參閱 PeerIslands 網站上的部落格文章 SQL2Mongo：Data Migration Journey。	DBA
了解查詢模式。	記錄您使用的前 10 個 SQL 查詢。您可以使用資料庫中可用的 performance_schema.events_statements_summary_by_digest 資料表	DBA

任務	Description	所需的技能
	來了解熱門查詢。如需詳細資訊，請參閱 PeerIslands 網站上的部落格文章 SQL2Mongo：Data Migration Journey。	
了解 SLA 承諾。	記錄資料庫操作的目標服務層級協議 (SLAs)。典型的措施包括查詢延遲和每秒查詢數。措施及其目標通常可在非功能需求 (NFR) 文件中取得。	DBA

定義 MongoDB 結構描述

任務	Description	所需的技能
定義目標結構描述。	定義目標 MongoDB 結構描述的各種選項。如需詳細資訊，請參閱 MongoDB Atlas 文件中的結構描述。根據資料表關係考慮最佳實務和設計模式。	MongoDB 工程師
定義目標查詢模式。	定義 MongoDB 查詢和彙總管道。這些查詢等同於您為 SQL Server 工作負載擷取的熱門查詢。若要了解如何建構 MongoDB 彙總管道，請參閱 MongoDB 文件。	MongoDB 工程師
定義 MongoDB 執行個體類型。	決定您計劃用於測試的執行個體大小。如需指引，請參閱 MongoDB 文件。	MongoDB 工程師

準備目標資料庫

任務	Description	所需的技能
設定 MongoDB Atlas 叢集。	若要在 AWS 上設定 MongoDB 叢集，請遵循 MongoDB 文件中的指示。	MongoDB 工程師
在目標資料庫中建立使用者。	遵循 MongoDB 文件中的指示，設定 MongoDB Atlas 叢集以進行存取和網路安全。	MongoDB 工程師
在 AWS 中建立適當的角色，並設定 Atlas 的角色型存取控制。	如有必要，請遵循 MongoDB 文件中的指示來設定其他使用者。透過 AWS 角色設定身分驗證和授權。	MongoDB 工程師
設定 MongoDB Atlas 存取的 Compass。	設定 MongoDB Compass GUI 公用程式，以便於導覽和存取。	MongoDB 工程師

使用測試資料產生器設定基本負載

任務	Description	所需的技能
安裝測試資料產生器。	在您的環境中安裝 PeerIsland Test Data Generator。	MongoDB 工程師
設定測試資料產生器以產生適當的資料。	使用資料程式庫為 MongoDB 結構描述中的每個欄位產生特定資料，以建立範本。如需詳細資訊，請參閱 MongoDB Data Generator & Perf。Analyzer 影片。	MongoDB 工程師
水平擴展測試資料產生器以產生所需的負載。	使用您建立的範本，透過設定所需的平行處理，開始針對目	MongoDB 工程師

任務	Description	所需的技能
	標集合產生負載。決定時間範圍並擴展以產生必要的資料。	
驗證 MongoDB Atlas 中的負載。	檢查載入 MongoDB Atlas 的資料。	MongoDB 工程師
在 MongoDB 上產生所需的索引。	根據查詢模式，視需要定義索引。如需最佳實務，請參閱 MongoDB 文件。	MongoDB 工程師

執行效能測試

任務	Description	所需的技能
在 Performance Analyzer 中設定負載描述檔。	透過設定特定查詢及其對應的權重、測試執行持續時間和階段，在 Performance Analyzer 中建立效能測試設定檔。如需詳細資訊，請參閱 MongoDB Data Generator & Perf。Analyzer 影片。	MongoDB 工程師
執行效能測試。	使用您建立的效能測試描述檔，透過設定所需的平行處理開始針對目標集合進行測試。水平擴展效能測試工具，以針對 MongoDB Atlas 執行查詢。	MongoDB 工程師
記錄測試結果。	記錄查詢的 P95, P99 延遲。	MongoDB 工程師
調校您的結構描述和查詢模式。	修改索引和查詢模式，以解決任何效能問題。	MongoDB 工程師

關閉專案

任務	Description	所需的技能
關閉臨時 AWS 資源。	刪除您用於測試資料產生器和效能分析器的所有臨時資源。	AWS 管理員
更新效能測試結果。	了解 MongoDB 查詢效能並將其與您的 SLAs 進行比較。如有必要，請微調 MongoDB 結構描述並重新執行程序。	MongoDB 工程師
結束專案。	關閉專案並提供意見回饋。	MongoDB 工程師

相關資源

- GitHub 儲存庫：S3toAtlas
- 結構描述：MongoDB 結構描述設計
- 彙總管道 ：MongoDB 彙總管道
- MongoDB Atlas 大小：大小層選擇
- 影片：MongoDB Data Generator & Perf。分析器
- 參考：MongoDB 文件
- 教學課程： MongoDB 開發人員指南、MMongoDB Jumpstart
- AWS Marketplace： AWS Marketplace 上的 MongoDB Atlas
- AWS 合作夥伴解決方案： AWS 參考部署上的 MongoDB Atlas

其他資源：

- SQL 分析
- MongoDB 開發人員社群論壇
- MongoDB 效能調校問題
- Atlas 和 Redshift 的操作分析
- 使用 MongoDB Atlas 和 AWS Elastic Beanstalk 進行應用程式現代化

使用 IaC 原則自動化 Amazon Aurora 全域資料庫的藍/綠部署

Ishwar Chauthaiwale、ANKIT JAIN 和 Ramu Jagini，Amazon Web Services

總結

對於在 Amazon Aurora 全域資料庫上執行關鍵工作負載的組織而言，管理資料庫更新、遷移或擴展工作可能具有挑戰性。確保在零停機時間的情況下無縫執行這些操作，對於維護服務可用性和避免使用者中斷至關重要。

藍/綠部署策略可讓您同時執行兩個相同的環境，為這項挑戰提供解決方案：藍（目前環境）和綠（新環境）。藍/綠策略可讓您實作變更、執行測試，並在風險最低且停機時間最低的環境中切換流量。

此模式使用基礎設施即程式碼 (IaC) 原則，協助您自動化 Aurora 全域資料庫的藍/綠部署程序。它使用 AWS CloudFormation、 AWS Lambda和 Amazon Route 53 來簡化藍/綠部署。為了改善可靠性，它使用全域交易識別符 (GTIDs進行複寫。與二進位日誌 (binlog) 複寫相比， GTID 式複寫可在環境之間提供更好的資料一致性和容錯移轉功能。

> **Note**
>
> 此模式假設您使用的是 Aurora MySQL 相容版本全域資料庫叢集。如果您改為使用 Aurora PostgreSQL 相容，請使用 MySQL 命令的 PostgreSQL 對等項目。

透過遵循此模式中的步驟，您可以：

- 佈建綠色 Aurora 全域資料庫：使用 CloudFormation 範本，您可以建立反映現有藍色環境的綠色環境。
- 設定 GTID 式複寫：您可以設定 GTID 複寫，讓藍色和綠色環境保持同步。
- 無縫切換流量：您可以使用 Route 53 和 Lambda，在完全同步後自動將流量從藍色環境切換到綠色環境。
- 完成部署：您驗證綠色環境作為主要資料庫可完全運作，然後停止複寫並清除任何臨時資源。

此模式中的方法提供下列優點：

- 在關鍵資料庫更新或遷移期間減少停機時間：自動化可確保在環境之間順利轉換，並將服務中斷降至最低。

- 啟用快速復原：如果流量切換到綠色環境後發生問題，您可以快速恢復到藍色環境並維持服務連續性。
- 增強測試和驗證：綠色環境可以完整測試，而不會影響即時藍色環境，從而降低生產過程中發生錯誤的可能性。
- 確保資料一致性：以 GTID 為基礎的複寫可讓您的藍色和綠色環境保持同步，防止資料在遷移期間遺失或不一致。
- 維持業務連續性：自動化藍/綠部署有助於避免長時間中斷和財務損失，方法是在更新或遷移期間保持您的服務可用。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 來源 Aurora MySQL 相容全域資料庫叢集（藍色環境）。全域資料庫提供多區域組態，以實現高可用性和災難復原。如需設定全域資料庫叢集的說明，請參閱 Aurora 文件。
- 在來源叢集上啟用 GTID 型複寫。

限制

- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

產品版本

- Aurora MySQL 相容 8.0 或更新版本

Architecture

此圖展示了以下要點：

- 全域資料庫設定：Aurora 全域資料庫叢集策略性地部署在兩個資料庫叢集中 AWS 區域。此組態可啟用地理分佈和區域備援，以增強災難復原功能。
- 主要至次要區域複寫：邏輯複寫機制可確保從主要區域到次要區域的無縫資料同步。此複寫會在地理距離之間維持資料一致性，並將延遲降至最低。

- 叢集之間的 GTID 式複寫：GTID 式複寫可維持藍色主要叢集與綠色主要叢集之間的交易一致性和排序資料流程，並確保可靠的資料同步。
- 藍色主要至次要複寫：邏輯複寫會在藍色主要叢集及其次要叢集之間建立強大的資料管道。此複寫可實現持續資料同步和高可用性。
- Route 53 DNS 組態：Route 53 託管區域記錄會管理所有藍色和綠色叢集資料庫端點的 DNS 解析。此組態提供無縫的端點映射，並在容錯移轉案例期間啟用有效的流量路由。

工具

AWS 服務

- Amazon Aurora 是一種全受管關聯式資料庫引擎，專為雲端而建置，並與 MySQL 和 PostgreSQL 相容。
- CloudFormation 可協助您建立和設定 AWS 資源的模型，以減少管理這些資源的時間，並有更多時間專注於執行 的應用程式 AWS。您可以建立範本來描述您想要的所有 AWS 資源，CloudFormation 會為您佈建和設定這些資源。
- AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。
- Amazon Route 53 是一種可用性高、可擴展性強的 DNS Web 服務。

最佳實務

建議您徹底檢閱 AWS 文件，以深入了解 Route 53 中的藍/綠部署策略、GTID 式複寫和加權路由政策。此知識對於有效實作和管理資料庫遷移、確保資料一致性，以及最佳化流量路由至關重要。透過全面了解這些 AWS 功能和最佳實務，您將更能處理未來的更新、盡可能減少停機時間，並維護彈性且安全的資料庫環境。

如需針對此模式使用 AWS 服務 的指導方針，請參閱下列 AWS 文件：

- Amazon Aurora MySQL 的最佳實務
- CloudFormation 最佳實務
- 使用 AWS Lambda 函數的最佳實務
- Amazon Route 53 的最佳實務

史詩

建立綠色環境

任務	Description	所需的技能
從藍色叢集建立快照備份。	在藍/綠部署中，綠色環境代表您目前（藍色）資料庫環境的新相同版本。您可以在切換生產流量之前，使用綠色環境安全地測試更新、驗證變更並確保穩定性。它可做為實作資料庫變更的預備基礎，並將對即時環境的干擾降至最低。 若要建立綠色環境，請先在 Aurora MySQL 相容全域資料庫中建立主要（藍色）叢集的快照。此快照是建立綠色環境的基礎。 若要建立快照： 1. 登入 AWS 管理主控台 並開啟 Amazon Relational Database Service (Amazon RDS) 主控台。 2. 選取您的主要（藍色）叢集。 3. 選擇動作、拍攝快照。 4. 提供快照的名稱，例如 `blue-green-demo` ，並啟動備份程序。 或者，您可以使用 AWS Command Line Interface (AWS CLI) 來建立快照：	DBA

任務	Description	所需的技能
	`aws rds create-db-cluster-snapshot --db-cluster-snapshot-identifier blue-green-demo --db-cluster-identifier ex-global-cluster --region eu-west-1` 在繼續下一個步驟之前，請確定快照已成功完成。	
為您的全域資料庫及其資源產生 CloudFormation 範本。	CloudFormation IaC 產生器可協助您從現有 AWS 資源產生 CloudFormation 範本。使用此功能為現有的 Aurora MySQL 相容全域資料庫及其相關聯的資源建立 CloudFormation 範本。此範本會設定子網路群組、安全群組、參數群組和其他設定。 1. 遵循 CloudFormation 文件中的指示，導覽至工具並將其連接至您的 AWS 環境。 2. 選取您的 Aurora 全域資料庫和相關聯的資源以產生範本。	DBA

任務	Description	所需的技能
修改綠色環境的 CloudFormation 範本。	自訂 CloudFormation 範本以反映綠色環境的設定。這包括更新資源名稱和識別符，以確保綠色環境獨立於藍色叢集運作。 1. 更新 `DBClusterIdentifier` 和 `DBInstanceIdentifier` 屬性以代表綠色環境。 2. 修改其他資源名稱（例如子網路群組和安全群組），以避免與現有的藍色環境發生衝突。 3. 透過設定正確的參數在範本中啟用 GTID 型複寫，如 Aurora 文件所述。 4. 變更 `SnapshotIdentifier` 屬性以指定 AWS 區域、您的帳戶 ID，以及上一個步驟的快照名稱： `SnapshotIdentifier: arn:aws:rds:<region>:<account-id>:snapshot:<snapshot-name>` **Note** 如果您使用 `SnapshotIdentifier` 屬性	DBA

任務	Description	所需的技能
	還原資料庫叢集，請避免指定屬性，例如 `GlobalClusterIdentifier`、`MasterUsername` 或 `MasterUserPassword`。	

任務	Description	所需的技能
部署 CloudFormation 堆疊以建立綠色環境的資源。	在此步驟中，您會部署自訂 CloudFormation 範本，以建立綠色環境的資源。 若要部署 CloudFormation 堆疊： 1. 開啟 AWS CloudFormation 主控台。 2. 在右上角，選擇使用新資源建立堆疊（標準）。 3. 上傳修改後的 CloudFormation 範本或指定範本 URL。選擇下一步。 4. 輸入堆疊名稱，例如 `GreenClusterStack`，並提供任何必要的參數（例如 `GreenClusterIdentifier`）。選擇下一步。 5. 視需要設定其他堆疊選項，並勾選核取方塊以確認 CloudFormation 可能會建立 AWS Identity and Access Management (IAM) 資源。選擇下一步。 6. 檢閱堆疊詳細資訊。 7. 選擇提交。 CloudFormation 會啟動建立綠色環境資源的程序。此程序可能需要幾分鐘的時間才能完成。	DBA

任務	Description	所需的技能
驗證 CloudFormation 堆疊和資源。	當 CloudFormation 堆疊部署完成時，您需要確認綠色環境已成功建立： 1. 在 CloudFormation 堆疊的輸出區段中，檢查資料庫叢集和資料庫執行個體的端點，以確認設定是否正確。 2. 開啟 Amazon RDS 主控台，並確認新的 Aurora 資料庫叢集（綠色環境）可用。 3. 確定子網路和安全群組等所有相關資源都已建立並連結至綠色環境。 驗證後，您的綠色環境已準備好進一步設定，包括從藍色環境複寫。	DBA

設定 GTID 型複寫

任務	Description	所需的技能
驗證藍色叢集上的 GTID 設定。	GTIDs 提供高度可靠的方法來複寫藍色和綠色環境之間的資料。GTID 型複寫透過為藍色環境中的每個交易指派唯一識別符，提供彈性、簡化的方法。相較於傳統的 binlog 複寫，此方法可確保環境之間的資料同步順暢、一致且易於管理。	DBA

任務	Description	所需的技能
	在設定複寫之前，您需要確保在藍色叢集上正確啟用 GTID 型複寫。此步驟保證藍色環境中的每個交易都是唯一的追蹤，並且可以在綠色環境中複寫。 若要確認 GTID 已啟用： 1. 在 Amazon RDS 主控台上，檢閱指派給藍色叢集的參數群組。 2. 確認已設定下列參數： • `gtid-mode = ON` • `enforce_gtid_consistency = ON` 這些設定可針對藍色環境中所有未來的交易啟用 GTID 追蹤。確認這些設定後，您可以開始設定複寫。	

任務	Description	所需的技能
建立複寫使用者。	若要將資料從藍色環境複寫到綠色環境，您需要在藍色叢集上建立專用的複寫使用者。此使用者將負責管理複寫程序。 若要設定複寫使用者： 1. 使用 MySQL 用戶端連線至藍色叢集。 2. 執行下列命令來建立複寫使用者： `CREATE USER 'repl_user'@'%' IDENTIFIED BY 'repl_password'; GRANT REPLICATION SLAVE ON . TO 'repl_user'@'%'; FLUSH PRIVILEGES;` 此使用者現在具有在兩個環境之間複寫資料的必要許可。	DBA

任務	Description	所需的技能
在綠色叢集上設定 GTID 型複寫。	下一個步驟是為 GTID 型複寫設定綠色叢集。此設定可確保綠色環境會持續鏡像在藍色環境中發生的所有交易。 若要設定綠色叢集： 1. 使用 MySQL 用戶端連線至綠色叢集。 2. 執行下列命令來設定複寫： `CHANGE MASTER TO MASTER_HOST='blue-cluster-endpoint', MASTER_USER='repl_user', MASTER_PASSWORD='repl_password', MASTER_AUTO_POSITION=1;` 其中： • `blue-cluster-endpoint` 將取代為藍色叢集的端點。 • `MASTER_AUTO_POSITION=1` 設定會指示 MySQL 使用 GTID 型複寫。它會自動定位綠色叢集以複寫藍色叢集的交易，而不必手動追蹤日誌和位置。	DBA

任務	Description	所需的技能
在綠色叢集上開始複寫。	您現在可以開始複寫程序。在綠色叢集上，執行 命令： `START SLAVE;` 這可讓綠色環境開始同步資料，以及從藍色環境接收和套用交易。	DBA
驗證複寫程序。	若要驗證綠色環境是否準確複寫藍色叢集的資料： 1. 在綠色叢集上執行下列命令，以檢查複寫狀態： `SHOW SLAVE STATUS\G;` 2. 檢閱輸出以驗證下列項目： • `Slave_IO_Running = Yes` • `Slave_SQL_Running = Yes` • `Retrieved_Gtid_Set` 和 `Executed_Gtid_Set` 值是up-to-date，並與藍色叢集同步。 • `Last_Error` 欄位中沒有複寫錯誤。 如果所有指標都正確，GTID 型複寫功能會順暢運作，而且綠色環境會與藍色環境完全同步。	DBA

從藍色叢集切換流量至綠色叢集

任務	Description	所需的技能
設定 Route 53 加權路由政策。	驗證藍色和綠色環境之間的資料一致性後，您可以將流量從藍色叢集切換到綠色叢集。此轉換應該是順暢的，而且應該將停機時間降至最低，並確保應用程式資料庫的完整性。若要解決這些需求，您可以使用 Route 53 進行 DNS 路由，並使用 Lambda 自動化流量切換。此外，定義明確的轉返計劃可確保您可以在發生任何問題時還原至藍色叢集。 第一步是在 Route 53 中設定加權路由。加權路由可讓您控制藍色和綠色叢集之間的流量分佈，並逐步將流量從一個環境轉移到另一個環境。 若要設定加權路由： 1. 開啟 Route 53 主控台並選擇您的託管區域。 2. 為資料庫建立兩個 DNS 記錄 (CNAMEs)：一個記錄用於藍色叢集，另一個記錄用於綠色叢集。 3. 指派初始權重： • 設定綠色叢集的低初始權重（例如 5%），以傳送一小部分的流量進行測試。	AWS DevOps

任務	Description	所需的技能
	• 為藍色叢集設定較高的權重（例如 95%），以便保留大部分流量。 此組態可讓您執行漸進式轉換，以降低風險，並在完全切換之前支援即時測試。 如需加權路由政策的詳細資訊，請參閱 Route 53 文件。	

任務	Description	所需的技能

部署 Lambda 函數以監控複寫延遲。	為了確保綠色環境與藍色環境完全同步，請部署 Lambda 函數來監控叢集之間的複寫延遲。此函數可以檢查複寫狀態，特別是 Seconds_Behind_Master 指標，以判斷綠色叢集是否已準備好處理所有流量。
	
	以下是您可以使用的範例 Lambda 函數：
	
	`import boto3`
	
	`def check_replication_lag(event, context):`
	`    client = boto3.client('rds')`
	`    response = client.describe_db_instances(DBInstanceIdentifier='green-cluster-instance')`
	`    replication_status = response['DBInstances'][0]['ReadReplicaDBInstanceIdentifiers']`
	`    if replication_status:`
	`        lag = replication_status[0]['ReplicationLag']`
	`        return lag`
	`    return -1`
		AWS DevOps

任務	Description	所需的技能
	此函數會檢查複寫延遲並傳回值。如果延遲為零，則綠色叢集會與藍色叢集完全同步。	

任務	Description	所需的技能

使用 Lambda 自動化 DNS 權重調整。	當複寫延遲達到零時，就可以將所有流量切換到綠色叢集。您可以使用另一個 Lambda 函數來自動化此轉換，該函數會調整 Route 53 中的 DNS 權重，以將 100% 的流量導向綠色叢集。
	
	以下是自動化流量切換的 Lambda 函數範例：
	
	`import boto3`
	
	`def switch_traffic(eve nt, context):`
	`route53 = boto3.cli ent('route53')`
	`lag = check_rep lication_lag(event, context)`
	`if lag == 0:`
	`response = route53.change_res ource_record_sets(`
	`HostedZon eId='YOUR_HOSTED_Z ONE_ID',`
	`ChangeBat ch={`
	`'Changes': [`
	`{`
	`'Action': 'UPSERT',`
	`'ResourceRecordSet': {`
	`'Name': 'db.examp le.com',
		AWS DevOps

任務	Description	所需的技能

	<pre>'Type': 'CNAME',
	
	'SetIdentifier': 'GreenCluster',
	
	'Weight': 100,
	
	'TTL': 60,
	
	'ResourceRecords': [{'Value': 'green-cluster-endpoint'}]
	
	}
	},
	{
	
	'Action': 'UPSERT',
	
	'ResourceRecordSet': {
	
	'Name': 'db.example.com',
	
	'Type': 'CNAME',
	
	'SetIdentifier': 'BlueCluster',
	
	'Weight': 0,
	
	'TTL': 60,
	
	'ResourceRecords': [{'Value': 'blue-cluster-endpoint'}]
	
	}
	}
	]</pre>	

任務	Description	所需的技能

	<pre> }
	        )
	        return response</pre>
	此函數會檢查複寫延遲，並在延遲為零時更新 Route 53 DNS 權重，以將流量完全切換到綠色叢集。
	
	Note
	在切換過程中，如果藍色叢集遇到大量寫入流量，請考慮在切換期間暫時暫停寫入操作。這可確保複寫趕上進度，並防止藍色和綠色叢集之間的資料不一致。	

任務	Description	所需的技能
驗證流量開關。	Lambda 函數調整 DNS 權重後，您應該驗證所有流量是否導向綠色叢集，以及切換是否成功。 若要驗證： 1. 監控 Route 53 DNS 記錄，以確認流量正導向綠色叢集。如需詳細資訊，請參閱 Route 53 文件。 2. 確認使用者是從綠色環境中提供服務，以檢查應用程式效能。 3. 驗證資料庫連線，以確認綠色叢集正在處理所有資料庫請求。 4. 監控 Amazon CloudWatch 指標是否有任何延遲、複寫延遲或效能降低的跡象。如需詳細資訊，請參閱 Aurora 文件。 如果一切如預期執行，則流量切換會完成。	AWS DevOps

任務	Description	所需的技能
如果您遇到任何問題，請復原變更。	如果流量切換之後發生任何問題，則擁有復原計劃至關重要。如有必要，以下說明如何快速還原至藍色叢集： 1. 還原 Route 53 中的 DNS 權重：使用相同的 Lambda 函數或手動調整 Route 53 DNS 權重，將 100% 的流量引導回藍色叢集。 2. 監控應用程式效能：立即監控應用程式日誌、CloudWatch 指標和資料庫效能，以確認切換回藍色環境已解決問題。 3. 識別並解決問題：在嘗試另一個流量切換之前，調查並解決綠色叢集的任何問題。 透過實作此轉返計劃，您可以確保在發生任何非預期問題時對使用者造成的干擾降到最低。	AWS DevOps

驗證並停止 GTID 型複寫

任務	Description	所需的技能
在綠色叢集上停止 GTID 型複寫。	將流量從藍色環境切換到綠色環境之後，您應該驗證轉換的成功，並確保綠色叢集如預期般運作。此外，藍色和綠色叢集之間的 GTID 型複寫必須停止，因為綠色環境現在可做為	DBA

任務	Description	所需的技能

	主要資料庫。完成這些任務可確保您的環境安全、簡化並針對持續操作進行最佳化。
	
	若要停止複寫：
	
	1. 使用 MySQL 用戶端連線到綠色叢集。
	2. 執行下列 SQL 命令來停止綠色叢集上的複寫程序：
	
	`STOP SLAVE;`
	
	3. （選用）如果需要，您可以重設複寫組態以清除任何剩餘的複寫設定：
	
	`RESET SLAVE ALL;`
	
	當您停止複寫時，綠色叢集會變得完全獨立，並做為工作負載的主要資料庫環境運作。	

任務	Description	所需的技能
清除資源。	清除從藍色遷移到綠色叢集期間建立的任何暫時或未使用的資源，可確保您的環境保持最佳化、安全且符合成本效益。清除包括調整安全設定、進行最終備份，以及停用不必要的資源。 若要清除資源： 1. 更新安全群組：設定與藍色和綠色叢集相關聯的安全群組，以反映新的主要環境（綠色）。如果不再需要藍色環境，請限制存取，並確認綠色叢集的安全設定遵循最佳實務。 2. 進行綠色叢集的最終備份：遷移完成後，拍攝綠色叢集的最終快照以做為備份。如有需要，您可以使用此快照在未來還原環境。 `aws rds create-db-snapshot --db-instance-identifier green-cluster-instance --db-snapshot-identifier green-cluster-final-snapshot` 3. 檢閱和移除臨時資源：檢閱遷移期間建立的任何臨時資源，例如臨時安全群組、快照或其他組態。刪除不再需要的資源，以防止不必要的	AWS DevOps

任務	Description	所需的技能
	成本。例如，如果不再需要藍色叢集，請將其刪除： `aws rds delete-db-cluster --db-cluster-identifier blue-cluster-identifier --skip-final-snapshot` 清理資源有助於維護安全且簡化的環境、降低成本，並確保僅保留必要的基礎設施。	

相關資源

CloudFormation:

- CloudFormation 使用者指南
- CloudFormation 最佳實務
- 使用 IaC 產生器從現有資源產生範本
- 將整個應用程式匯入 CloudFormation(AWS 部落格文章）

Amazon Aurora：

- Amazon Aurora 使用者指南
- 管理 Amazon Aurora 資料庫叢集

藍/綠部署策略：

- Amazon Aurora 藍/綠部署概觀

以 GTID 為基礎的複寫：

- 使用 GTID 式複寫 (Amazon RDS 文件）

AWS Lambda:

- AWS Lambda 開發人員指南
- 使用 AWS Lambda 函數的最佳實務

Amazon Route 53：

- 《Amazon Route 53 開發人員指南》
- 加權路由

MySQL 用戶端工具：

- PyMYSQL

自動化跨 的 Amazon RDS 執行個體複寫 AWS 帳戶

Parag Nagwekar 和 Arun Chandapillai，Amazon Web Services

總結

此模式說明如何 AWS 帳戶 使用 和 自動化複寫、追蹤和復原不同 Amazon Relational Database Service (Amazon RDS) 資料庫執行個體的程序 AWS Step Functions AWS Lambda。您可以使用此自動化來執行 RDS 資料庫執行個體的大規模複寫，而不會受到任何效能影響或營運額外負荷，無論您的組織大小為何。您也可以使用此模式來協助您的組織遵守強制性資料控管策略或合規要求，以要求在不同 和 之間複寫 AWS 帳戶 和備援您的資料 AWS 區域。跨帳戶大規模複寫 Amazon RDS 資料是一種低效率且容易出錯的手動程序，成本高昂且耗時，但此模式中的自動化可協助您安全、有效且有效率地實現跨帳戶複寫。

先決條件和限制

先決條件

- 兩個 AWS 帳戶
- 在來源中啟動和執行的 RDS 資料庫執行個體 AWS 帳戶
- 目的地中 RDS 資料庫執行個體的子網路群組 AWS 帳戶
- 在來源中建立 AWS 帳戶 並與目的地帳戶共用的 AWS Key Management Service (AWS KMS) 金鑰（如需政策詳細資訊的詳細資訊，請參閱此模式的其他資訊???區段。）
- 目的地 AWS KMS key 中的 AWS 帳戶 ，用於加密目的地帳戶中的資料庫

限制

- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

產品版本

- Python 3.9 （使用 AWS Lambda)
- PostgreSQL 11.3、13.x 和 14.x

Architecture

技術堆疊

- Amazon Relational Database Service (Amazon RDS)
- Amazon Simple Notification Service (Amazon SNS)
- AWS Key Management Service (AWS KMS)
- AWS Lambda
- AWS Secrets Manager
- AWS Step Functions

目標架構

下圖顯示使用 Step Functions 將 RDS 資料庫執行個體的排程隨需複寫從來源帳戶 （帳戶 A) 協調到目的地帳戶 （帳戶 B) 的架構。

在來源帳戶 （圖表中的帳戶 A) 中，Step Functions 狀態機器會執行下列動作：

1. 從帳戶 A 中的 RDS 資料庫執行個體建立快照。
2. 使用 AWS KMS key 來自帳戶 A 的 複製和加密快照。為了確保傳輸中的加密，無論資料庫執行個體是否已加密，快照都會加密。
3. 透過讓帳戶 B 存取快照，與帳戶 B 共用資料庫快照。
4. 將通知推送至 SNS 主題，然後 SNS 主題會叫用帳戶 B 中的 Lambda 函數。

在目的地帳戶 （圖表中的帳戶 B) 中，Lambda 函數會執行 Step Functions 狀態機器來協調下列項目：

1. 將共用快照從帳戶 A 複製到帳戶 B，同時使用 AWS KMS key 帳戶 A 的 先解密資料，然後使用帳戶 B AWS KMS key 中的 加密資料。
2. 從 Secrets Manager 讀取秘密，以擷取目前資料庫執行個體的名稱。
3. 使用 AWS KMS key Amazon RDS 的新名稱和預設值，從快照還原資料庫執行個體。
4. 讀取新資料庫的端點，並使用新資料庫端點更新 Secrets Manager 中的秘密，然後標記先前的資料庫執行個體，以便稍後刪除。
5. 保留資料庫的最新 N 個執行個體，並刪除所有其他執行個體。

工具

AWS 服務

- Amazon Relational Database Service (Amazon RDS) 可協助您在 中設定、操作和擴展關聯式資料庫 AWS 雲端。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- 適用於 Python (Boto3) 的 AWS SDK 是一種軟體開發套件，可協助您整合 Python 應用程式、程式庫或指令碼 AWS 服務。
- AWS Secrets Manager 可協助您將程式碼中的硬式編碼憑證 (包括密碼) 取代為 Secrets Manager 的 API 呼叫，以便透過程式設計方法來擷取機密。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 Lambda 函數和其他 AWS 服務來建置業務關鍵應用程式。

程式碼儲存庫

此模式的程式碼可在 GitHub Crossaccount RDS 複寫儲存庫中使用。

史詩

AWS 帳戶 只需按一下，即可自動化跨 的 RDS 資料庫執行個體複寫

任務	Description	所需的技能
在來源帳戶中部署 CloudFormation 堆疊。	1. 登入 AWS 管理主控台 來源帳戶的 （帳戶 A)，然後開啟 CloudFormation 主控台。 2. 在導覽窗格中，選擇 Stacks (堆疊)。	雲端管理員、雲端架構師

任務	Description	所需的技能
	3. 選擇建立堆疊，然後選擇使用現有資源（匯入資源）。 4. 在識別資源頁面上，選擇下一步。 5. 在指定範本頁面上，選取上傳範本。 6. 選擇選擇檔案，從 GitHub Crossaccount RDS 複寫儲存庫中選取`Cloudformation-SourceAccountRDS.yaml` 檔案，然後選擇下一步。 7. 針對堆疊名稱，輸入堆疊的名稱。 8. 在參數區段中，指定堆疊範本中定義的參數： • 針對 DestinationAccountNumber，輸入目的地 RDS 資料庫執行個體的帳號。 • 針對 KeyName，輸入您的 AWS KMS key。 • 針對 ScheduleExpression，輸入 Cron 表達式（預設值為每日上午 12：00)。 • 針對 SourceDBIdentifier，輸入來源資料庫的名稱。 • 對於 SourceDBSnapshotName，輸入快照的名稱或接受預設值。 9. 選擇下一步。	

任務	Description	所需的技能
	10.在設定堆疊選項頁面上，保留預設值，然後選擇下一步。 11.檢閱您的堆疊組態，然後選擇提交。 12.選擇堆疊的資源索引標籤，然後記下 SNS 主題的 Amazon Resource Name (ARN)。	

任務	Description	所需的技能
在目的地帳戶中部署 CloudFormation 堆疊。	1. 登入目的地帳戶（帳戶 B) AWS 管理主控台 的 ，然後開啟 CloudFormation 主控台。 2. 在導覽窗格中，選擇 Stacks (堆疊)。 3. 選擇建立堆疊，然後選擇使用現有資源（匯入資源）。 4. 在識別資源頁面上，選擇下一步。 5. 在指定範本頁面上，選取上傳範本。 6. 選擇檔案，從 GitHub Crossaccount RDS 複寫儲存庫中選取`Cloudformation-DestinationAccountRDS.yaml` 檔案，然後選擇下一步。 7. 針對堆疊名稱，輸入堆疊的名稱。 8. 在參數區段中，指定堆疊範本中定義的參數： • 在 DatabaseName 中，輸入資料庫的名稱。 • 在引擎中，輸入與來源資料庫相符的資料庫引擎類型。 • 針對 DBInstanceClass，輸入偏好的資料庫執行個體類型或接受預設值。 • 針對子網路群組，輸入現有的 VPC 子網路群組。	雲端架構師、DevOps 工程師、雲端管理員

任務	Description	所需的技能
	如需建立子網路群組的指示，請參閱 Amazon RDS 文件中的步驟 2：建立資料庫子網路群組。 • 針對 SecretName，輸入路徑和秘密名稱，或接受預設值。 • 針對 SGID，輸入目的地叢集的安全群組 ID。 • 在 KMSKey 中，輸入目的地帳戶中 KMS 金鑰的 ARN。 • 對於 NoOfOlderInstances，輸入您要為復原保留的 RDS 資料庫執行個體的舊複本數量。 9. 選擇下一步。 10.在設定堆疊選項頁面上，保留預設值，然後選擇下一步。 11.檢閱您的堆疊組態，然後選擇提交。 12.選擇堆疊的資源索引標籤，然後記下 的實體 ID 和 ARN`InvokeStepFunction` 。	
驗證在目的地帳戶中建立 RDS 資料庫執行個體。	1. 登入 AWS 管理主控台 並開啟 Amazon RDS 主控台。 2. 在導覽窗格中，選擇資料庫，然後驗證新的 RDS 資料庫執行個體是否出現在新的叢集下。	雲端管理員、雲端架構師、DevOps 工程師

任務	Description	所需的技能

訂閱 Lambda 函數至 SNS 主題。	您必須執行下列 AWS Command Line Interface (AWS CLI) 命令，將目的地帳戶 （帳戶 B) 中的 Lambda 函數訂閱至來源帳戶 （帳戶 A) 中的 SNS 主題。
	
	在帳戶 A 中，執行下列命令：
	
	`aws sns add-permission \ --label lambda-access --aws-account-id <DestinationAccount> \ --topic-arn <Arn of SNSTopic > \ --action-name Subscribe ListSubscriptionsB yTopic`
	
	在帳戶 B 中，執行下列命令：
	
	`aws lambda add-permi ssion \ --function-name <Name of InvokeStepFunction > \ --source-arn <Arn of SNSTopic > \ --statement-id function-with-sns \ --action lambda:In vokeFunction \ --principal sns.amazo naws.com`
	
	在帳戶 B 中，執行下列命令：
		雲端管理員、雲端架構師、DBA

任務	Description	所需的技能
	`aws sns subscribe \ --protocol "lambda" \ --topic-arn <Arn of SNSTopic> \ --notification-endpoint <Arn of InvokeStepFunction>`	

任務	Description	所需的技能
從來源帳戶同步 RDS 資料庫執行個體與目的地帳戶。	透過啟動來源帳戶中的 Step Functions 狀態機器來啟動隨需資料庫複寫。 1. 開啟 Step Functions 主控台。 2. 在導覽窗格中，選擇狀態機器。 3. 選擇您的狀態機器。 4. 在執行索引標籤上，選取您的函數，然後選擇開始執行以啟動工作流程。 **Note** 排程器已就緒，可協助您按排程自動執行複寫，但排程器預設為關閉。您可以在目的地帳戶中 CloudFormation 堆疊的 資源索引標籤中找到排程器的 Amazon CloudWatch 規則名稱。如需如何修改 CloudWatch Events 規則的說明，請參閱 CloudWatch 文件中的刪除或停用 CloudWatch Events 規則。 CloudWatch	雲端架構師、DevOps 工程師、雲端管理員

任務	Description	所需的技能
視需要將資料庫復原至任何先前的複本。	1. 開啟 Secrets Manager 主控台。 2. 從秘密清單中，選擇您先前使用 CloudFormation 範本建立的秘密。您的應用程式會使用秘密來存取目的地叢集中的資料庫。 3. 若要從詳細資訊頁面更新秘密值，請在秘密值區段中，選擇擷取秘密值，然後選擇編輯。 4. 輸入資料庫端點的詳細資訊。	雲端管理員、DBA、DevOps 工程師

相關資源

- 跨區域僅供讀取複本 (Amazon RDS 文件)
- 藍/綠部署 (Amazon RDS 文件)

其他資訊

您可以使用下列範例政策跨 共用您的 AWS KMS key AWS 帳戶。

```
{
    "Version": "2012-10-17",
    "Id": "cross-account-rds-kms-key",
    "Statement": [
        {
            "Sid": "Enable user permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::<SourceAccount>:root"
            },
            "Action": "kms:*",
            "Resource": "*"
```

```
        },
        {
            "Sid": "Allow administration of the key",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::<DestinationAccount>:root"
            },
            "Action": [
                "kms:Create*",
                "kms:Describe*",
                "kms:Enable*",
                "kms:List*",
                "kms:Put*",
                "kms:Update*",
                "kms:Revoke*",
                "kms:Disable*",
                "kms:Get*",
                "kms:Delete*",
                "kms:ScheduleKeyDeletion",
                "kms:CancelKeyDeletion"
            ],
            "Resource": "*"
        },
        {
            "Sid": "Allow use of the key",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::<DestinationAccount>:root",
                    "arn:aws:iam::<SourceAccount>:root"
                ]
            },
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:DescribeKey",
                "kms:CreateGrant"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

使用 AWS Lambda 和 任務排程器，在 Amazon EC2 上執行的 SQL Server Express 版本中自動化資料庫任務

Subhani Shaik，Amazon Web Services

總結

此模式示範如何在 SQL Server Express 版本中排程和管理資料庫任務，這是 SQL Server 的免費版本。不過，SQL Server Express 版本缺少通常處理自動化資料庫操作的 SQL Server Agent 服務。此模式說明如何使用任務排程器和 Lambda 做為在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上執行的 SQL Server Express 版本中自動化資料庫任務的替代方案。

任務排程器是內建的 Windows 系統公用程式，可促進例行任務的自動執行。它提供一種機制來排程和管理自動化操作，無需在週期性程序中手動介入。 AWS Lambda 是一種無伺服器運算服務，可自動執行程式碼以回應事件，而無需您管理基礎基礎設施。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 使用 Amazon Virtual Private Cloud (Amazon VPC) 建立的虛擬私有雲端 (VPC)
- 具有 Windows Server 的 Amazon EC2 執行個體
- 使用 Windows Server 連接至 Amazon EC2 執行個體的 Amazon Elastic Block Store (Amazon EBS) 磁碟區
- SQL Server Express Edition 二進位檔

限制

- 如需 SQL Server Express Edition 功能限制的相關資訊，請參閱 Microsoft 網站。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

- SQL Server 2016 或更新版本搭配 SQL Server Express 版本

Architecture

下圖顯示已安裝 SQL Server Express 版本的 Amazon EC2 執行個體。執行個體可透過遠端桌面通訊協定 (RDP) 用戶端或從 存取 AWS Systems Manager Session Manager。 AWS Key Management Service (AWS KMS) 會處理 Amazon EBS 磁碟區的資料加密，以確保data-at-rest的安全性。基礎設施也包含 AWS Identity and Access Management (IAM)，可提供存取控制和管理執行 Lambda 函數的許可。Amazon Simple Storage Service (Amazon S3) 存放 Lambda 函數。

工具

AWS 服務

- Amazon Elastic Block Store (Amazon EBS) 提供區塊層級儲存體磁碟區，可搭配使用 Amazon EC2 執行個體。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Systems Manager Session Manager 是全受管 AWS Systems Manager 工具。使用 Session Manager，您可以管理 Amazon EC2 執行個體、邊緣裝置、內部部署伺服器和虛擬機器 VMs)。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。

其他工具

- Microsoft SQL Server Management Studio (SSMS) 是一種用於管理 SQL Server 的工具，包括存取、設定和管理 SQL Server 元件。
- Python 是一種一般用途的電腦程式設計語言。您可以使用它來建置應用程式、自動化任務，以及在上開發服務AWS 雲端。

- 任務排程器是一種 Microsoft 工具，可用來自動排程電腦上的例行任務。

最佳實務

- Amazon EC2 的最佳實務
- 在 Amazon EC2 上部署 Microsoft SQL Server 的最佳實務
- 使用 AWS Lambda 函數的最佳實務
- IAM 中的安全最佳實務

史詩

建立 Amazon EC2 執行個體並安裝 SQL Server Express 版本

任務	Description	所需的技能
部署 Amazon EC2 執行個體。	若要建立 Amazon EC2 執行個體，請開啟位於 https://console.aws.amazon.com/ec2/：// 的 Amazon EC2 主控台，然後從適用於 Windows Server 的執行個體清單中選取 Amazon Machine Image (AMI)。 如需詳細資訊，請參閱 AWS 文件中的啟動 Amazon EC2 執行個體。	DBA、AWS DevOps
安裝 SQL Server Express 版本。	若要安裝 SQL Server Express 版本，請完成下列步驟： 1. 若要連線至 Amazon EC2 執行個體，請選擇一個選項： • 選項 A – 使用遠端桌面通訊協定 (RDP)。如需說明，請參閱 AWS 文件中	DBA、AWS DevOps

任務	Description	所需的技能
	的使用 RDP 用戶端連線至 Windows 執行個體。 • 選項 B – 使用 Amazon EC2 主控台和 AWS Systems Manager Session Manager。如需說明，請參閱 AWS 文件中的使用 Session Manager 連線至 Amazon EC2 執行個體。 2. 若要下載所需的 SQL Server Express 版本，請前往 Microsoft 網站上的 SQL Server Downloads。 3. 若要安裝 SQL Server Express 版本，請遵循 Microsoft 網站上的規劃 SQL Server 安裝中的指示。	

建立自動化資料庫維護任務

任務	Description	所需的技能
識別例行任務。	識別您要自動化的例行任務。例如，下列任務符合自動化的資格： • 資料庫備份（完整、差異和交易日誌） • 索引維護和重組 • 統計資料更新 • 應用程式特定的操作 • 資料清除或封存	DBA

任務	Description	所需的技能
準備 SQL 指令碼。	若要準備 SQL 指令碼，請執行下列動作： 1. 為每個維護任務建立 SQL 查詢。以下是執行特定資料庫備份的 T-SQL 查詢範例：`Backup Database <Database_Name> To Disk='C:\Backups\Database_Name.bak'` 2. 將指令碼檔案儲存為 `<File Name>.sql` 。然後，將指令碼儲存至 Amazon EC2 執行個體或網路檔案共享上伺服器本機磁碟機的可存取位置。	DBA
設定存取許可。	若要設定存取許可，請執行下列動作： 1. 設定適當的檔案系統許可。如需說明，請參閱 Microsoft 網站上的設定資料庫引擎存取的檔案系統許可。 2. 檢查 SQL Server 服務帳戶是否具有必要的存取權。如需說明，請參閱 Microsoft 網站上的設定 Windows 服務帳戶和許可。 3. 驗證遠端共用的網路連線。如需詳細資訊，請參閱 AWS 文件中的使用檔案共用存取資料。	DBA

使用任務排程器自動化任務

任務	Description	所需的技能

建立批次檔案。	• 若要建立批次檔案，請使用文字編輯器輸入下列命令。將參數 username和取代password為您自己的值。然後將檔案儲存為 <Name>.bat 。
	`sqlcmd -S servername -U username -P password -i <T-SQL query path.sql>`
	• 若要建立 SQL 任務的批次檔案，請使用文字編輯器並輸入下列命令。將參數 ServerName 、username、DatabaseName 和 取代password為您自己的值。然後將檔案儲存為 <Name>.bat 。
	`@echo off sqlcmd -S [ServerName] -d [DatabaseName] -U username -P password -i "PathToSQLScript\Script.sql" -o "PathToOutput\Output.txt"`
		AWS DevOps、DBA

在任務排程器中建立任務。	若要在任務排程器中建立任務，請使用下列步驟：	DBA

任務	Description	所需的技能
	1. 若要開啟任務排程器，請在 Windows 搜尋中輸入 taskchd.msc。 2. 選擇動作功能表，然後選取建立基本任務。 3. 針對名稱，提供任務的名稱，然後選擇下一步。 4. 針對觸發，選取您希望任務開始的時間選項，然後選擇下一步。 5. 提供任務的開始和遞迴資訊，然後選擇下一步。 6. 在動作區段中，選取啟動程式，然後選擇下一步。 7. 針對程式/指令碼，指定您在上一個任務中建立的批次檔案路徑，然後選擇下一步。 8. 選擇完成。 若要手動執行任務，請在新建立的任務上按一下滑鼠右鍵，然後選取執行。	
檢視任務狀態。	若要在任務排程器中檢視任務的狀態，請使用下列步驟： 1. 在任務排程器中，前往任務排程器程式庫，其中會顯示所有任務。 2. 若要查看您先前建立的任務狀態，請選取任務，然後前往歷史記錄索引標籤。	DBA、AWS DevOps

使用 自動化任務 AWS Lambda

任務	Description	所需的技能
實作解決方案。	若要實作此模式的解決方案，請使用下列步驟： 1. 建立 Lambda 函式。如需說明，請參閱 AWS 文件中的建立您的第一個 Lambda 函數。 2. 排程 Lambda 函數。如需說明，請參閱 AWS 文件中的依排程叫用 Lambda 函數。 3. 執行 T-SQL 查詢。如需詳細資訊，請參閱 AWS 文件中的教學課程：使用 Lambda 函數存取 Amazon RDS 資料庫。本教學課程說明如何從 Lambda 函數連接 Amazon RDS 資料庫以執行 SQL 查詢	AWS DevOps，DevOps 工程師

疑難排解

問題	解決方案
Lambda 問題	如需使用 時可能遇到的錯誤和問題的協助 AWS Lambda，請參閱 AWS 文件中的 Lambda 中的故障診斷問題。

相關資源

- Amazon EC2 執行個體類型

- AWS Lambda 文件
- AWS Lambda 定價
- 開發人員的任務排程器 (Microsoft 網站)

使用 DR Orchestrator Framework 自動化跨區域容錯移轉和容錯回復

Jitendra Kumar、Pavithra Balasubramanian 和 Oliver Francis，Amazon Web Services

總結

此模式說明如何使用 DR Orchestrator Framework 協調和自動化手動、容易出錯的步驟，以跨 Amazon Web Services (AWS) 區域執行災難復原。模式涵蓋下列資料庫：

- MySQL 的 Amazon Relational Database Service (Amazon RDS)、PostgreSQL 的 Amazon RDS 或 Amazon RDS for MariaDB
- Amazon Aurora MySQL 相容版本或 Amazon Aurora PostgreSQL 相容版本 （使用集中式檔案）
- Amazon ElastiCache (Redis OSS)

若要示範 DR Orchestrator Framework 的功能，您可以建立兩個資料庫執行個體或叢集。主要 位於 AWS 區域 `us-east-1`，次要 位於 `us-west-2`。若要建立這些資源，您可以使用 aws-cross-region-dr-databases GitHub 儲存庫的 `App-Stack` 資料夾中的 AWS CloudFormation 範本。

先決條件和限制

一般先決條件

- 部署在主要和次要 DR Orchestrator Framework AWS 區域
- 兩個 Amazon Simple Storage Service 儲存貯體
- 具有兩個子網路和一個 AWS 安全群組的虛擬私有雲端 (VPC)

引擎特定的先決條件

- Amazon Aurora – 至少必須有兩個可用的 Aurora 全域資料庫 AWS 區域。您可以使用 `us-east-1`做為主要區域，並使用 `us-west-2`做為次要區域。
- Amazon ElastiCache (Redis OSS) – ElastiCache 全域資料存放區必須提供兩種。 AWS 區域您可以`use us-east-1`做為主要區域，並使用 `us-west-2`做為次要區域。

Amazon RDS 限制

- 在進行容錯移轉或容錯回復之前，DR Orchestrator Framework 不會檢查複寫延遲。必須手動檢查複寫延遲。

- 此解決方案已使用具有一個僅供讀取複本的主要資料庫執行個體進行測試。如果您想要使用多個僅供讀取複本，請在生產環境中實作解決方案之前，先徹底測試解決方案。

Aurora 限制

- 功能可用性和支援會因每個資料庫引擎的特定版本和不同版本而有所不同 AWS 區域。如需跨區域複寫功能和區域可用性的詳細資訊，請參閱跨區域僅供讀取複本。
- Aurora 全域資料庫具有支援 Aurora 資料庫執行個體類別的特定組態需求，以及最大數量 AWS 區域。如需詳細資訊，請參閱 Amazon Aurora 全域資料庫的組態需求。
- 此解決方案已使用具有一個僅供讀取複本的主要資料庫執行個體進行測試。如果您想要使用多個僅供讀取複本，請在生產環境中實作解決方案之前，先徹底測試解決方案。

ElastiCache 限制

- 如需全域資料存放區和 ElastiCache 組態需求的區域可用性資訊，請參閱 ElastiCache 文件中的先決條件和限制。

Amazon RDS product 版本

Amazon RDS 支援下列引擎版本：

- MySQL – Amazon RDS 支援執行下列 MySQL 版本的資料庫執行個體：MySQL 8.0 和 MySQL 5.7
- PostgreSQL – 如需有關 Amazon RDS for PostgreSQL 支援版本的資訊，請參閱可用的 PostgreSQL 資料庫版本。
- MariaDB – Amazon RDS 支援執行下列 MariaDB 版本的資料庫執行個體：
  - MariaDB 10.11
  - MariaDB 10.6
  - MariaDB 10.5

Aurora 產品版本

- Amazon Aurora 全域資料庫轉換需要 Aurora MySQL 相容於 MySQL 5.7 相容性，2.09.1 版及更新版本

  如需詳細資訊，請參閱 Amazon Aurora 全域資料庫的限制。

ElastiCache (Redis OSS) 產品版本

Amazon ElastiCache (Redis OSS) 支援下列 Redis 版本：

- Redis 7.1 (增強版)
- Redis 7.0 (增強版)
- Redis 6.2 (增強版)
- Redis 6.0 (增強版)
- Redis 5.0.6 (增強版)

如需詳細資訊，請參閱支援的 ElastiCache (Redis OSS) 版本。

Architecture

Amazon RDS 架構

Amazon RDS 架構包含下列資源：

- 在主要區域 (us-east-1) 中建立的主要 Amazon RDS 資料庫執行個體，具有用戶端的讀取/寫入存取權
- 在次要區域 (us-west-2) 中建立的 Amazon RDS 僅供讀取複本，具有用戶端的唯讀存取權
- 部署在主要和次要區域的 DR Orchestrator Framework

上圖顯示以下項目：

1. 主要執行個體與次要執行個體之間的非同步複寫
2. 主要區域中用戶端的讀取/寫入存取權
3. 次要區域中用戶端的唯讀存取權

Aurora 架構

Amazon Aurora 架構包含下列資源：

- 在具有作用中寫入器端點的主要區域 (us-east-1) 中建立的主要 Aurora 資料庫叢集
- 在次要區域 (us-west-2) 中建立且具有非作用中寫入器端點的 Aurora 資料庫叢集

- 部署在主要和次要區域的 DR Orchestrator Framework

上圖顯示以下項目：

1. 主要叢集與次要叢集之間的非同步複寫
2. 具有主動寫入器端點的主要資料庫叢集
3. 具有非作用中寫入器端點的次要資料庫叢集

ElastiCache (Redis OSS) 架構

Amazon ElastiCache (Redis OSS) 架構包含下列資源：

- 使用兩個叢集建立的 ElastiCache (Redis OSS) 全域資料存放區：
  1. 主要區域中的主要叢集 (us-east-1)
  2. 次要區域中的次要叢集 (us-west-2)
- 兩個叢集之間具有 TLS 1.2 加密的 Amazon 跨區域連結
- 部署在主要和次要區域的 DR Orchestrator Framework

自動化和擴展

DR Orchestrator Framework 可擴展，並支援平行容錯移轉或容錯回復多個 AWS 資料庫。

您可以使用下列承載程式碼容錯移轉您帳戶中的多個 AWS 資料庫。在此範例中，三個 AWS 資料庫（兩個全域資料庫，例如 Aurora MySQL 相容或 Aurora PostgreSQL 相容，以及一個 Amazon RDS for MySQL 執行個體）容錯移轉至 DR 區域：

```
{
  "StatePayload": [
    {
      "layer": 1,
      "resources": [
        {
          "resourceType": "PlannedFailoverAurora",
          "resourceName": "Switchover (planned failover) of Amazon Aurora global
 databases (MySQL)",
```

```
          "parameters": {
            "GlobalClusterIdentifier": "!Import dr-globaldb-cluster-mysql-global-
identifier",
            "DBClusterIdentifier": "!Import dr-globaldb-cluster-mysql-cluster-
identifier"
          }
        },
        {
          "resourceType": "PlannedFailoverAurora",
          "resourceName": "Switchover (planned failover) of Amazon Aurora global
 databases (PostgreSQL)",
          "parameters": {
            "GlobalClusterIdentifier": "!Import dr-globaldb-cluster-postgres-global-
identifier",
            "DBClusterIdentifier": "!Import dr-globaldb-cluster-postgres-cluster-
identifier"
          }
        },
        {
          "resourceType": "PromoteRDSReadReplica",
          "resourceName": "Promote RDS for MySQL Read Replica",
          "parameters": {
            "RDSInstanceIdentifier": "!Import rds-mysql-instance-identifier",
            "TargetClusterIdentifier": "!Import rds-mysql-instance-global-arn"
          }
        }
      ]
    }
  ]
}
```

工具

AWS 服務

- Amazon Aurora 是一種全受管關聯式資料庫引擎，專為雲端而建置，並與 MySQL 和 PostgreSQL 相容。
- Amazon ElastiCache 可協助您在 中設定、管理和擴展分散式記憶體內快取環境 AWS 雲端。此模式使用 Amazon ElastiCache (Redis OSS)。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。在此模式中，Lambda 函數由 AWS Step Functions 用來執行步驟。

- Amazon Relational Database Service (Amazon RDS) 可協助您在 中設定、操作和擴展關聯式資料庫 AWS 雲端。此模式支援 Amazon RDS for MySQL、Amazon RDS for PostgreSQL 和 Amazon RDS for MariaDB。
- 適用於 Python (Boto3) 的 AWS SDK 可協助您整合 Python 應用程式、程式庫或指令碼 AWS 服務。在此模式中，Boto3 APIs用於與資料庫執行個體或全域資料庫通訊。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務 來建置業務關鍵應用程式。在此模式中，Step Functions 狀態機器用於協調和執行資料庫執行個體或全域資料庫的跨區域容錯移轉和容錯回復。

程式碼儲存庫

此模式的程式碼可在 GitHub 上的 aws-cross-region-dr-databases 儲存庫中使用。

史詩

安裝 DR Orchestrator Framework

任務	Description	所需的技能
複製 GitHub 儲存庫。	若要複製儲存庫，請執行下列命令： `git clone https://github.com/aws-samples/aws-cross-region-dr-databases.git`	AWS DevOps、AWS 管理員

在 .zip 檔案封存中封裝 Lambda 函數程式碼。	建立 Lambda 函數的封存檔案，以包含 DR Orchestrator Framework 相依性：
	`cd <YOUR-LOCAL-GIT-FOLDER>/DR-Orchestration-artifacts`
	
	`bash scripts/deploy-orchestrator-sh.sh`
		AWS 管理員

任務	Description	所需的技能
建立 S3 儲存貯體。	需要 S3 儲存貯體來存放 DR Orchestrator Framework 以及您的最新組態。建立兩個 S3 儲存貯體，一個位於主要區域 (us-east-1)，另一個位於次要區域 (us-west-2)： • dr-orchestrator-xxxxxx-us-east-1 • dr-orchestrator-xxxxxx-us-west-2 xxxxxx 以隨機值取代 ，讓儲存貯體名稱是唯一的。	AWS 管理員
建立子網路和安全群組。	在主要區域 (us-east-1) 和次要區域 (us-west-2) 中，為您的 VPC 中的 Lambda 函數部署建立兩個子網路和一個安全群組： • subnet-XXXXXXX • subnet-YYYYYYY • sg-XXXXXXXXXXX	AWS 管理員

任務	Description	所需的技能
更新 DR Orchestrator 參數檔案。	在 <YOUR-LOCAL-GIT-FOLDER>/DR-Orchestration-artifacts/cloudformation 資料夾中，更新下列 DR Orchestrator 參數檔案： • Orchestrator-Deployer-parameters-us-east-1.json • Orchestrator-Deployer-parameters-us-west-2.json 使用下列參數值，y將 x和 取代為您的資源名稱： [{ "ParameterKey": "TemplateStoreS3BucketName", "ParameterValue": "dr-orchestrator-xxxxxx-us-east-1" }, { "ParameterKey": "TemplateVPCId", "ParameterValue": "vpc-xxxxxx" }, { "ParameterKey": "TemplateLambdaSubnetID1",	AWS 管理員

任務	Description	所需的技能

	"Paramete
	rValue": "subnet-x
	xxxxx"
	},
	{
	"ParameterKey":
	"TemplateLambdaSub
	netID2",
	"Paramete
	rValue": "subnet-y
	yyyyy"
	},
	{
	"ParameterKey":
	"TemplateLambdaSec
	urityGroupID",
	"Paramete
	rValue": "sg-xxxxx
	xxxxx"
	}
	]	

任務	Description	所需的技能
將 DR Orchestrator Framework 程式碼上傳至 S3 儲存貯體。	S3 儲存貯體中的程式碼會比本機目錄中的程式碼更安全。將`DR-Orchestration-artifacts` 目錄，包括所有檔案和子資料夾，上傳至 S3 儲存貯體。 若要上傳程式碼，請執行下列動作： 1. 登入 AWS 管理主控台。 2. 導覽至 Amazon S3 主控台。 3. 選取`dr-orchestrator-xxxxxx-us-east-1` bucket。 4. 選擇上傳，然後選擇新增資料夾。 5. 選取 `DR-Orchestration-artifacts` 資料夾。 6. 選擇上傳。 7. 選取儲存`dr-orchestrator-xxxxxx-us-west-2` 貯體。 8. 重複步驟 4–7。	AWS 管理員

任務	Description	所需的技能

在主要區域中部署 DR Orchestrator Framework。	若要在主要區域 (us-east-1) 部署 DR Orchestrator Framework，請執行下列命令：
	
	`cd <YOUR-LOCAL-GIT-FOLDER>/DR-Orchestration-artifacts/cloudformation`
	
	`aws cloudformation deploy \`
	`--region us-east-1 \`
	`--stack-name dr-orchestrator \`
	`--template-file Orchestrator-Deployer.yaml \`
	`--parameter-overrides file://Orchestrator-Deployer-parameters-us-east-1.json \`
	`--capabilities CAPABILITY_AUTO_EXPAND CAPABILITY_NAMED_IAM CAPABILITY_IAM \`
	`--disable-rollback`
		AWS 管理員

任務	Description	所需的技能

在次要區域中部署 DR Orchestrator Framework。	在次要區域 (`us-west-2`) 中，執行下列命令：
	`cd <YOUR-LOCAL-GIT-FOLDER>/DR-Orchestration-artifacts/cloudformation`
	
	`aws cloudformation deploy \`
	`--region us-west-2 \`
	`--stack-name dr-orchestrator \`
	`--template-file Orchestrator-Deployer.yaml \`
	`--parameter-overrides file://Orchestrator-Deployer-parameters-us-west-2.json \`
	`--capabilities CAPABILITY_AUTO_EXPAND CAPABILITY_NAMED_IAM CAPABILITY_IAM \`
	`--disable-rollback`
		AWS 管理員

驗證部署。	如果 CloudFormation 命令成功執行，則會傳回下列輸出： `Successfully created/updated stack - dr-orchestrator` 或者，您可以導覽至 CloudFormation 主控台並驗證`dr-orchestrator` 堆疊的狀態。	AWS 管理員

建立資料庫執行個體或叢集

任務	Description	所需的技能
建立資料庫子網路和安全群組。	在您的 VPC 中，為主要 (us-east-1) 和次要 (us-west-2) 區域中的資料庫執行個體或全域資料庫建立兩個子網路和一個安全群組： • subnet-XXXXXX • subnet-XXXXXX • sg-XXXXXXXXXX	AWS 管理員
更新主要資料庫執行個體或叢集的參數檔案。	在 <YOUR LOCAL GIT FOLDER>/App-Stack 資料夾中，更新主要區域的參數檔案。 Amazon RDS 在 RDS-MySQL-parameter-us-east-1.json 檔案中，DBSecurityGroup 使用您建立的資源名稱更新 SubnetIds 和 ： `{ "Parameters": { "SubnetIds": "subnet-xxxxxx,subnet-xxxxxx", "DBSecurityGroup": "sg-xxxxxxxxxx", "MySqlGlobalIdentifier":"rds-mysql-instance", "InitialDatabaseName": "mysqldb",`	AWS 管理員

任務	Description	所需的技能
	<pre>"DBPortNumber": "3789", "PrimaryRegion": "us-east-1", "SecondaryRegion": "us-west-2", "KMSKeyAliasName": "rds/rds-mysql-instance-KmsKeyId" } }</pre> Amazon Aurora 在 Aurora-MySQL-parameter-us-east-1.json 檔案中，DBSecurityGroup 使用您建立的資源名稱更新 SubnetIds 和 ： <pre>{ "Parameters": { "SubnetIds": "subnet1-xxxxxx,subnet2-xxxxxx", "DBSecurityGroup": "sg-xxxxxxxxxx", "GlobalClusterIdentifier":"dr-globaldb-cluster-mysql", "DBClusterName":"dbcluster-01", "SourceDBClusterName":"dbcluster-02", "DBPortNumber": "3787", "DBInstanceClass": "db.r5.large", "InitialDatabaseName": "sampledb",</pre>	

任務	Description	所需的技能
	<pre>"PrimaryRegion": "us-east-1", "SecondaryRegion": "us-west-2", "KMSKeyAliasName": "rds/dr-globaldb-cluster-mysql-KmsKeyId" } }</pre> Amazon ElastiCache (Redis OSS) 在 `ElastiCache-parameter-us-east-1.json` 檔案中，`DBSecurityGroup` 使用您建立的資源名稱更新 `SubnetIds` 和 。 <pre>{ "Parameters": { "CacheNodeType": "cache.m5.large", "DBSecurityGroup": "sg-xxxxxxxxxx", "SubnetIds": "subnet-xxxxxx,subnet-xxxxxx", "EngineVersion": "5.0.6", "GlobalReplicationGroupIdSuffix": "demo-redis-global-datastore", "NumReplicas": "1", "NumShards": "1", "ReplicationGroupId": "demo-redis-cluster",</pre>	

任務	Description	所需的技能
	<pre>"DBPortNumber": "3788", "TransitEncryption": "true", "KMSKeyAliasName": "elasticache/demo-redis-global-datastore-KmsKeyId", "PrimaryRegion": "us-east-1", "SecondaryRegion": "us-west-2" } }</pre>	

任務	Description	所需的技能

在主要區域中部署資料庫執行個體或叢集。	若要在主要區域 (us-east-1) 中部署執行個體或叢集，請根據您的資料庫引擎執行下列命令。
	
	Amazon RDS
	
	`cd <YOUR-LOCAL-GIT-FOLDER>/App-Stack`
	
	`aws cloudformation deploy \`
	`--region us-east-1 \`
	`--stack-name rds-mysql-app-stack \`
	`--template-file RDS-MySQL-Primary.yaml \`
	`--parameter-overrides file://RDS-MySQL-parameter-us-east-1.json \`
	`--capabilities CAPABILITY_AUTO_EXPAND CAPABILITY_NAMED_IAM CAPABILITY_IAM \`
	`--disable-rollback`
	
	Amazon Aurora
	
	`cd <YOUR-LOCAL-GIT-FOLDER>/App-Stack`
	
	`aws cloudformation deploy \`
	`--region us-east-1 \`
	`--stack-name aurora-mysql-app-stack \`
	`--template-file Aurora-MySQL-Primary.yaml \`
		AWS 管理員

任務	Description	所需的技能
	--parameter-overrides file://Aurora-MySQL-parameter-us-east-1.json \ --capabilities CAPABILITY_AUTO_EXPAND CAPABILITY_NAMED_IAM CAPABILITY_IAM \ --disable-rollback Amazon ElastiCache (Redis OSS) cd <YOUR-LOCAL-GIT-FOLDER>/App-Stack aws cloudformation deploy \ --region us-east-1 --stack-name elasticache-ds-app-stack \ --template-file ElastiCache-Primary.yaml \ --parameter-overrides file://ElastiCache-parameter-us-east-1.json \ --capabilities CAPABILITY_AUTO_EXPAND CAPABILITY_NAMED_IAM CAPABILITY_IAM \ --disable-rollback 確認 CloudFormation 資源已成功部署。	

任務	Description	所需的技能
更新次要資料庫執行個體或叢集的參數檔案。	在 <YOUR LOCAL GIT FOLDER>/App-Stack 資料夾中，更新次要區域的 參數檔案。 Amazon RDS 在 RDS-MySQL-parameter-us-west-2.json 檔案中，DBSecurityGroup 使用您建立的資源名稱更新 SubnetIDs 和 。PrimaryRegionKMSKeyArn 使用從 CloudFormation 主要資料庫執行個體堆疊的輸出區段MySQLKmsKeyId 取得的 值來更新 ： `{ "Parameters": { "SubnetIds": "subnet-aaaaaaaaa, subnet-bbbbbbbbb", "DBSecurityGroup": "sg-ccccccccc", "MySqlGlobalIdentifier":"rds-mysql-instance", "InitialDatabaseName": "mysqldb", "DBPortNumber": "3789", "PrimaryRegion": "us-east-1", "SecondaryRegion": "us-west-2", "KMSKeyAliasName": "rds/rds-mysql-instance-KmsKeyId",`	AWS 管理員

任務	Description	所需的技能

	"PrimaryRegionKMSKeyArn":"arn:aws:kms:us-east-1:xxxxxxxxx:key/mrk-xxxxxxxxxxxxxxxxx"
	}
	}
	Amazon Aurora
	在 Aurora-MySQL-parameter-us-west-2.json 檔案中，DBSecurityGroup 使用您建立的資源名稱更新 SubnetIDs 和 。PrimaryRegionKMSKeyArn 使用AuroraKmsKeyId 取自主要資料庫執行個體 CloudFormation 堆疊輸出區段的 值來更新 ：
	{
	"Parameters": {
	"SubnetIds": "subnet1-aaaaaaaaa,subnet2-bbbbbbbbb",
	"DBSecurityGroup": "sg-ccccccccc",
	"GlobalClusterIdentifier":"dr-globaldb-cluster-mysql",
	"DBClusterName":"dbcluster-01",
	"SourceDBClusterName":"dbcluster-02",
	"DBPortNumber": "3787",	

任務	Description	所需的技能
	<pre>"DBInstanceClass": "db.r5.large", "InitialDatabaseName": "sampledb", "PrimaryRegion": "us-east-1", "SecondaryRegion": "us-west-2", "KMSKeyAliasName": "rds/dr-globaldb-cluster-mysql-KmsKeyId" } }</pre> Amazon ElastiCache (Redis OSS) 在 `ElastiCache-parameter-us-west-2.json` 檔案中，DBSecurityGroup 使用您建立的資源名稱更新 SubnetIDs 和。PrimaryRegionKMSKeyArn 使用ElastiCacheKmsKeyId 取自主要資料庫執行個體 CloudFormation 堆疊輸出區段的 值更新 ： <pre>{ "Parameters": { "CacheNodeType": "cache.m5.large", "DBSecurityGroup": "sg-ccccccccc", "SubnetIds": "subnet-aaaaaaaaa, subnet-bbbbbbbbb",</pre>	

任務	Description	所需的技能
	"EngineVersion": "5.0.6", "GlobalReplicationGroupIdSuffix": "demo-redis-global-datastore", "NumReplicas": "1", "NumShards": "1", "ReplicationGroupId": "demo-redis-cluster", "DBPortNumber": "3788", "TransitEncryption": "true", "KMSKeyAliasName": "elasticache/demo-redis-global-datastore-KmsKeyId", "PrimaryRegion": "us-east-1", "SecondaryRegion": "us-west-2" } }	

任務	Description	所需的技能

在次要區域中部署資料庫執行個體或叢集。	根據您的資料庫引擎執行下列命令。
	
	Amazon RDS
	
	`cd <YOUR-LOCAL-GIT-FOLDER>/App-Stack`
	
	`aws cloudformation deploy \`
	`--region us-west-2 \`
	`--stack-name rds-mysql-app-stack \`
	`--template-file RDS-MySQL-DR.yaml \`
	`--parameter-overrides file://RDS-MySQL-parameter-us-west-2.json \`
	`--capabilities CAPABILITY_AUTO_EXPAND CAPABILITY_NAMED_IAM CAPABILITY_IAM \`
	`--disable-rollback`
	
	Amazon Aurora
	
	`cd <YOUR-LOCAL-GIT-FOLDER>/App-Stack`
	
	`aws cloudformation deploy \`
	`--region us-west-2 \`
	`--stack-name aurora-mysql-app-stack \`
	`--template-file Aurora-MySQL-DR.yaml \`
	`--parameter-overrides file://Aurora-MySQ`
		AWS 管理員

任務	Description	所需的技能
	L-parameter-us-wes t-2.json \ --capabilities CAPABILITY_AUTO_EX PAND CAPABILIT Y_NAMED_IAM CAPABILIT Y_IAM \ --disable-rollback Amazon ElastiCache (Redis OSS) cd <YOUR-LOCAL-GIT-FO LDER>/App-Stack aws cloudformation deploy \ --region us-west-2 \ --stack-name elasticac he-ds-app-stack \ --template-file ElastiCache-DR.yaml \ --parameter-overrides file://ElastiCache -parameter-us-west -2.json \ --capabilities CAPABILITY_AUTO_EX PAND CAPABILIT Y_NAMED_IAM CAPABILIT Y_IAM \ --disable-rollback 確認 CloudFormation 資源已成功部署。	

相關資源

- 資料庫在 上的災難復原策略 AWS (AWS 方案指引策略)

- 自動化 上關聯式資料庫的 DR 解決方案 AWS (AWS 方案指引指南)
- 使用 Amazon Aurora 全球資料庫
- AWS 區域 使用全域資料存放區跨 複寫
- 自動化 上關聯式資料庫的 DR 解決方案 AWS (AWS 方案指引指南)

使用 Systems Manager 和 EventBridge 自動備份 SAP HANA 資料庫

Ambarish Satarkar 和 Gaurav Rath，Amazon Web Services

總結

此模式說明如何使用 AWS Systems Manager、Amazon EventBridge、Amazon Simple Storage Service (Amazon S3) 和 AWS Backint Agent for SAP HANA 來自動化 SAP HANA 資料庫備份。

此模式使用 `BACKUP DATA`命令提供 Shell 指令碼型方法，並不需要在多個系統中維護每個作業系統 (OS) 執行個體的指令碼和任務組態。

注意：截至 2023 年 4 月，AWS Backup 宣布支援 Amazon Elastic Compute Cloud (Amazon EC2) 上的 SAP HANA 資料庫。如需詳細資訊，請參閱 Amazon EC2 執行個體上的 SAP HANA 資料庫備份。

根據您的組織需求，您可以使用 AWS Backup 服務自動備份 SAP HANA 資料庫，也可以使用此模式。

先決條件和限制

先決條件

- 在為 Systems Manager 設定的受管 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上，具有執行中狀態支援版本的現有 SAP HANA 執行個體
- Systems Manager Agent (SSM Agent) 2.3.274.0 或更新版本已安裝
- 未啟用公有存取的 S3 儲存貯體
- 名為 的 `hdbuserstore`Ankey`SYSTEM`
- Automation Runbook 要按排程執行的 AWS Identity and Access Management (IAM) 角色
- `AmazonSSMManagedInstanceCore` 和 `ssm:StartAutomationExecution`政策會連接至 Systems Manager Automation 服務角色。

限制

- AWS Backint Agent for SAP HANA 不支援重複資料刪除。
- AWS Backint Agent for SAP HANA 不支援資料壓縮。

產品版本

下列作業系統支援 AWS Backint Agent：

- SUSE Linux Enterprise Server
- SUSE Linux Enterprise Server for SAP
- 適用於 SAP 的 Red Hat Enterprise Linux

AWS Backint Agent 支援下列資料庫：

- SAP HANA 1.0 SP12（單一節點和多個節點）
- SAP HANA 2.0 和更新版本（單一節點和多個節點）

Architecture

目標技術堆疊

- AWS 後端代理程式
- Amazon S3
- AWS Systems Manager
- Amazon EventBridge
- SAP HANA

目標架構

下圖顯示安裝 AWS Backint Agent、S3 儲存貯體以及 Systems Manager 和 EventBridge 的安裝指令碼，這些指令碼使用 Command 文件來排程定期備份。

自動化和擴展

- 您可以使用 Systems Manager Automation Runbook 安裝多個 AWS Backint 代理程式。
- Systems Manager Runbook 的每個執行都可以根據目標選擇，擴展至 n 個 SAP HANA 執行個體。
- EventBridge 可以自動化 SAP HANA 備份。

工具

- AWS Backint Agent for SAP HANA 是一種獨立應用程式，可與您現有的工作流程整合，將 SAP HANA 資料庫備份到您在組態檔案中指定的 S3 儲存貯體。AWS Backint Agent 支援 SAP HANA 資料庫的完整、增量和差異備份。它在 SAP HANA 資料庫伺服器上執行，其中備份和目錄會從 SAP HANA 資料庫傳輸到 AWS Backint Agent。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可用來將應用程式與來自各種來源的資料連線。EventBridge 會將即時資料從您的應用程式、軟體即服務 (SaaS) 應用程式和 AWS 服務串流傳送至目標，例如 AWS Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他帳戶中的事件匯流排。
- Amazon Simple Storage Service (Amazon S3) 是一種物件儲存服務。您可以使用 Amazon S3 隨時從 Web 任何地方存放和擷取任意資料量。
- AWS Systems Manager 可協助您在 AWS 上檢視和控制您的基礎設施。使用 Systems Manager 主控台，您可以檢視來自多個 AWS 服務的操作資料，並自動化 AWS 資源的操作任務。

Code

此模式的程式碼可在 aws-backint-automated-backup GitHub 儲存庫中使用。

史詩

建立 hdbuserstore 金鑰 SYSTEM

任務	Description	所需的技能
建立 hdbuserstore 金鑰。	1. 導覽至 `/usr/sap/<SID>/HDB<InstNo>/exe`。 2. 執行下列命令，以XX 做為 SAP HANA 資料庫執行個體編號。 `hdbuserstore -i set SYSTEM <hostname>:3XX13@SYSTEMDB SYSTEM`	AWS 管理員、SAP HANA 管理員

任務	Description	所需的技能
	例如，對於執行個體編號為 的 SAP HANA 主機saphanadb 00，請執行下列命令。 `hdbuserstore -i set SYSTEM saphanadb:30013@SYSTEMDB SYSTEM`	

安裝 AWS Backint Agent

任務	Description	所需的技能
安裝 AWS Backint Agent。	請遵循 AWS Backint Agent 文件中的安裝和設定適用於 SAP HANA 的 AWS Backint Agent 中的指示。	AWS 管理員、SAP HANA 管理員

建立 Systems Manager 命令文件

任務	Description	所需的技能
建立 Systems Manager 命令文件。	1. 登入 AWS 管理主控台並開啟 AWS Systems Manager 主控台。 2. 選擇文件，然後選擇由我擁有。 3. 確認您位於與 SAP HANA 資料庫相同的 AWS 區域。 4. 選擇建立文件、命令或工作階段以建立文件。	AWS 管理員、SAP HANA 管理員

任務	Description	所需的技能
	5. 使用唯一且描述性的名稱，不含空格 （例如 SAP HANA-Backup)。 6. 確定文件類型設定為 命令文件。 7. 在內容標頭下，有一些範例程式碼。請確定您選擇 JSON 程式碼類型，並將程式碼取代為 GitHub 儲存庫HDB_Backup_SSM_Document.json 檔案中的程式碼。 8. 選擇 Create document (建立文件)。 9. 在我擁有的區段中檢查您的文件。	

定期排程備份

任務	Description	所需的技能
使用 Amazon EventBridge 排程定期備份。	1. 開啟 Amazon EventBridge 主控台，選擇規則，然後選擇建立規則。 2. 在定義規則詳細資訊畫面上，輸入規則的唯一名稱和描述，並使用預設事件匯流排。 3. 在規則類型下，選擇排程，然後選擇下一步。 4. 在定義排程畫面上，根據所需的頻率選擇適當的排程模式和 Cron 或 Rate 表達式。	AWS 管理員、SAP HANA 管理員

任務	Description	所需的技能
	5. 在選取目標畫面上，針對目標類型選擇 AWS 服務。在選取目標下，選擇 Systems Manager Run Command。 6. 選擇您先前建立的文件。 7. 在目標金鑰和目標值下，提供執行個體 ID。您可以使用標籤名稱和標籤值來新增多個執行個體。 8. 在設定自動化參數下，選擇增量或差異備份的常數。如果您想要完整備份，請選擇無參數。 9. 選擇是否建立新角色或使用現有角色。如果您使用現有角色，請確定其具有叫用目標所需的政策。 10.保留預設的其他設定，然後選擇下一步。 11.設定標籤畫面是選用的。選擇下一步。 12.在檢閱和建立畫面上，檢閱規則設定，然後選擇建立。應該成功建立規則。	
	您可以從 S3 儲存貯體路徑驗證備份成功。 `s3:/<your_bucket_name>/<target folder>/<SID>/usr/sap/<SID>`	

任務	Description	所需的技能
	`/SYS/global/hdb/backint/DB_<SID>/` 您也可以從 SAP HANA 備份目錄驗證備份。	

相關資源

- 適用於 SAP HANA 的 AWS 後端代理程式
- 安裝和設定適用於 SAP HANA 的 AWS Backint Agent

使用 Python 應用程式自動產生 Amazon DynamoDB 的 PynamoDB 模型和 CRUD 函數 DynamoDB

Vijit Vashishtha、Dheeraj Alimchandani 和 Dhananjay Karanjkar，Amazon Web Services

總結

通常需要實體和建立、讀取、更新和刪除 (CRUD) 操作函數，才能有效率地執行 Amazon DynamoDB 資料庫操作。PynamoDB 是以 Python 為基礎的介面，支援 Python 3。它還提供功能，例如支援 Amazon DynamoDB 交易、自動屬性值序列化和還原序列化，以及與常見 Python 架構的相容性，例如 Flask 和 Django。此模式透過提供簡化 DynamoDB 模型和 CRUD 操作函數自動建立的PynamoDB。雖然它為資料庫資料表產生必要的 CRUD 函數，但也可以從 Amazon PynamoDB 資料表反向工程 PynamoDB 模型和 CRUD 函數。 DynamoDB 此模式旨在使用 Python 型應用程式簡化資料庫操作。

以下是此解決方案的主要功能：

- PynamoDB 模型的 JSON 結構描述 – 透過匯入 JSON 結構描述檔案，在 Python 中自動產生 PynamoDB 模型。
- CRUD 函數產生 – 自動產生函數，以在 DynamoDB 資料表上執行 CRUD 操作。
- 從 DynamoDB 反向工程 – 使用 PynamoDB 物件關聯映射 (ORM) 對現有 Amazon DynamoDB 資料表的 PynamoDB 模型和 CRUD 函數進行反向工程。 DynamoDB

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- Python 3.8 版或更新版本，已下載並安裝
- Jinja2 3.1.2 版或更新版本，已下載並安裝
- 您要為其產生 ORM 的 Amazon DynamoDB 資料表
- AWS Command Line Interface (AWS CLI)，已安裝並設定
- PynamoDB 5.4.1 版或更新版本，已安裝

Architecture

目標技術堆疊

- JSON 指令碼
- Python 應用程式
- PynamoDB 模型
- Amazon DynamoDB 資料庫執行個體

目標架構

1. 您可以建立輸入 JSON 結構描述檔案。此 JSON 結構描述檔案代表您要從 和 CRUD 函數建立 PynamoDB 模型之個別 DynamoDB 資料表的屬性。 PynamoDB 它包含以下三個重要金鑰：
   - `name` –目標 DynamoDB 資料表的名稱。
   - `region` – 資料表託管 AWS 區域 所在的
   - `attributes` – 屬於目標資料表一部分的屬性，例如分割區索引鍵 （ 也稱為雜湊屬性)、排序索引鍵、本機次要索引、全域次要索引，以及任何非索引鍵屬性。此工具預期輸入結構描述只會在應用程式直接從目標資料表擷取金鑰屬性時提供非金鑰屬性。如需如何在 JSON 結構描述檔案中指定屬性的範例，請參閱此模式的其他資訊一節。
2. 執行 Python 應用程式，並提供 JSON 結構描述檔案做為輸入。
3. Python 應用程式會讀取 JSON 結構描述檔案。
4. Python 應用程式會連線至 DynamoDB 資料表，以衍生結構描述和資料類型。應用程式會執行 describe_table 操作，並擷取資料表的索引鍵和索引屬性。
5. Python 應用程式結合了 JSON 結構描述檔案和 DynamoDB 資料表中的屬性。它使用 Jinja 範本引擎來產生 PynamoDB 模型和對應的 CRUD 函數。
6. 您可以存取 PynamoDB 模型，在 DynamoDB 資料表上執行 CRUD 操作。

工具

AWS 服務

- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。

其他工具

- Jinja 是一種可擴展的範本引擎，可將範本編譯為最佳化的 Python 程式碼。此模式使用 Jinja 透過在範本中嵌入預留位置和邏輯來產生動態內容。

- PynamoDB 是 Amazon DynamoDB 的 Python 型界面。
- Python 是一種一般用途的電腦程式設計語言。

程式碼儲存庫

此模式的程式碼可在 GitHub 自動產生 PynamoDB 模型和 CRUD 函數儲存庫中使用。儲存庫分為兩個主要部分：控制器套件和範本。

控制器套件

控制器 Python 套件包含主要應用程式邏輯，可協助產生 PynamoDB 模型和 CRUD 函數。其中包含下列各項：

- `input_json_validator.py` – 此 Python 指令碼會驗證輸入 JSON 結構描述檔案，並建立 Python 物件，其中包含目標 DynamoDB 資料表的清單，以及每個資料表的必要屬性。
- `dynamo_connection.py` – 此指令碼會建立與 DynamoDB 資料表的連線，並使用 `describe_table`操作擷取建立 PynamoDB 模型所需的屬性。
- `generate_model.py` – 此指令碼包含 Python 類別`GenerateModel`，可根據輸入 JSON 結構描述檔案和 `describe_table`操作來建立 PynamoDB 模型。
- `generate_crud.py` – 對於 JSON 結構描述檔案中定義的 DynamoDB 資料表，此指令碼會使用 `GenerateCrud`操作來建立 Python 類別。

範本

此 Python 目錄包含下列 Jinja 範本：

- `model.jinja` – 此 Jinja 範本包含用於產生 PynamoDB 模型指令碼的範本表達式。
- `crud.jinja` – 此 Jinja 範本包含用於產生 CRUD 函數指令碼的範本表達式。

史詩

設定環境

任務	Description	所需的技能
複製儲存庫。	輸入下列命令以複製自動產生 PynamoDB 模型和 CRUD 函數儲存庫。 `git clone https://github.com/aws-samples/amazon-reverse-engineer-dynamodb.git`	應用程式開發人員

設定 Python 環境。	1. 導覽至複製儲存庫中的最上層目錄。
	`cd amazon-reverse-engineer-dynamodb`
	2. 輸入下列命令來安裝所需的程式庫和套件。
	`pip install -r requirements.txt`
		應用程式開發人員

產生 PynamoDB 模型和 CRUD 函數

任務	Description	所需的技能
修改 JSON 結構描述檔案。	1. 導覽至複製儲存庫中的最上層目錄。 `cd amazon-reverse-engineer-dynamodb`	應用程式開發人員

任務	Description	所需的技能
	2. 在偏好的編輯器中開啟 `test.json` 檔案。您可以使用此檔案做為建立自己的 JSON 結構描述檔案的參考，也可以更新此檔案中的值以符合您的環境。 3. 修改目標 DynamoDB 資料表的名稱 AWS 區域和屬性值。 **Note** 如果您定義不存在於 JSON 結構描述檔案中的資料表，此解決方案不會為該資料表產生模型或 CRUD 函數。 4. 儲存並關閉 `test.json` 檔案。建議您使用新名稱儲存此檔案。	
執行 Python 應用程式。	輸入下列命令來產生 PynamoDB 模型和 CRUD 函數，其中 `<input_schema.json>` 是 JSON 結構描述檔案的名稱。 `python main.py --file <input_schema.json>`	應用程式開發人員

驗證 PynamoDB 模型和 CRUD 函數

任務	Description	所需的技能
驗證產生的 PynamoDB 模型。	1. 在複製儲存庫的最上層目錄中，輸入下列命令以導覽至`models`儲存庫。 `cd models` 2. 根據預設，此解決方案會命名 PynamoDB 模型檔案 `demo_model.py` 。驗證此檔案是否存在。	應用程式開發人員
驗證產生的 CRUD 函數。	1. 在複製儲存庫的最上層目錄中，輸入下列命令以導覽至`crud`儲存庫。 `cd crud` 2. 根據預設，此解決方案會命名指令碼 `demo_crud.py` 。驗證此檔案是否存在。 3. 使用 `demo_crud.py` 檔案中的 Python 類別，對目標 DynamoDB 資料表執行 CRUD 操作。確認操作已成功完成。	應用程式開發人員

相關資源

- Amazon DynamoDB 的核心元件 (DynamoDB 文件)
- 使用次要索引改善資料存取 (DynamoDB 文件)

其他資訊

JSON 結構描述檔案的範例屬性

```
[
{
"name": "test_table",
"region": "ap-south-1",
"attributes": [
{
"name": "id",
"type": "UnicodeAttribute"
},
{
"name": "name",
"type": "UnicodeAttribute"
},
{
"name": "age",
"type": "NumberAttribute"
}
]
}
]
```

使用 Cloud Custodian 封鎖對 Amazon RDS 的公開存取

abhay kumar 和 Dwarika Patra，Amazon Web Services

總結

許多組織會在多個雲端廠商上執行工作負載和服務。在這些混合雲端環境中，除了個別雲端提供者提供的安全性之外，雲端基礎設施還需要嚴格的雲端控管。Amazon Relational Database Service (Amazon RDS) 等雲端資料庫是一項重要的服務，必須監控是否有任何存取和許可漏洞。雖然您可以透過設定安全群組來限制對 Amazon RDS 資料庫的存取，但您可以新增第二層保護來禁止公開存取等動作。封鎖公開存取可協助您符合一般資料保護法規 (GDPR)、健康保險流通與責任法案 (HIPAA)、國家標準與技術研究所 (NIST)，以及支付卡產業資料安全標準 (PCI DSS)。

Cloud Custodian 是開放原始碼規則引擎，可用來強制執行 Amazon RDS 等 Amazon Web Services (AWS) 資源的存取限制。使用 Cloud Custodian，您可以設定規則，根據定義的安全和合規標準來驗證環境。您可以使用 Cloud Custodian 來管理雲端環境，方法是協助確保符合安全政策、標籤政策，以及未使用的資源和成本管理的垃圾回收。使用 Cloud Custodian，您可以使用單一界面在混合雲端環境中實作控管。例如，您可以使用 Cloud Custodian 介面與 AWS 和 Microsoft Azure 互動，減少使用 AWS Config AWS 安全群組和 Azure 政策等機制的工作量。

此模式提供在 上使用 Cloud Custodian AWS 以在 Amazon RDS 執行個體上強制執行公有可存取性限制的指示。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 金鑰對
- AWS Lambda 已安裝

Architecture

下圖顯示 Cloud Custodian 將政策部署到 AWS Lambda、 AWS CloudTrail 啟動`CreateDBInstance`事件，以及將 Amazon RDS 上的 Lambda 函數設定為 `PubliclyAccessible` false。

工具

AWS 服務

- AWS CloudTrail 可協助您稽核 的控管、合規和營運風險 AWS 帳戶。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 中設定、操作和擴展關聯式資料庫 AWS 雲端。

其他工具

- Cloud Custodian 會將許多組織用來管理公有雲端帳戶的工具和指令碼統一為單一開放原始碼工具。它使用無狀態規則引擎進行政策定義和強制執行，具有指標、結構化輸出和雲端基礎設施的詳細報告。它與無伺服器執行期緊密整合，以低營運開銷提供即時修補和回應。

史詩

設定 AWS CLI

任務	Description	所需的技能
安裝 AWS CLI。	若要安裝 AWS CLI，請遵循 AWS 文件中的指示。	AWS 管理員

設定 AWS 登入資料。	設定 AWS CLI 用來與 互動的設定 AWS，包括您要使用的 AWS 區域 和輸出格式。
	`$>aws configure`
	`AWS Access Key ID [None]: <your_access_key_id>`
		AWS 管理員

任務	Description	所需的技能
	`AWS Secret Access Key [None]: <your_secret_access_key> Default region name [None]: Default output format [None]:` 如需詳細資訊，請參閱 AWS 文件。	
建立 IAM 角色。	若要使用 Lambda 執行角色建立 IAM 角色，請執行下列命令。 `aws iam create-role --role-name lambda-ex --assume-role-policy-document '{"Version": "2012-10-17", "Statement": [{ "Effect": "Allow", "Principal": {"Service": "lambda.amazonaws.com"}, "Action": "sts:AssumeRole"}]}`	AWS DevOps

設定雲端託管人

任務	Description	所需的技能
安裝 Cloud Custodian。	若要為您的作業系統和環境安裝 Cloud Custodian，請遵循 Cloud Custodian 文件中的指示。	DevOps 工程師

任務	Description	所需的技能
檢查雲端託管結構描述。	若要查看您可以對其執行政策的 Amazon RDS 資源完整清單，請使用下列命令。 `custodian schema aws.rds`	DevOps 工程師
建立 Cloud Custodian 政策。	使用 YAML 擴充功能，在其他資訊區段中儲存 Cloud Custodian 政策檔案下的程式碼。	DevOps 工程師
定義 Cloud Custodian 動作以變更可公開存取的旗標。	1. 找到託管人代碼（例如 `/Users/abcd/custodian/lib/python3.9/site-packages/c7n/resources/rds.py`）。 2. 在 中尋找 `RDSSetPublicAvailability` 類別`rds.py`，並使用其他資訊區段中 c7n 資源 rds.py 檔案下的程式碼來修改此類別。	DevOps 工程師
執行試轉。	（選用）若要檢查政策識別哪些資源，而不對資源執行任何動作，請使用下列命令。 `custodian run -dryrun <policy_name>.yaml -s <output_directory>`	DevOps 工程師

部署政策

任務	Description	所需的技能
使用 Lambda 部署政策。	若要建立將執行政策的 Lambda 函數，請使用下列命令。 `custodian run -s policy.yaml` 此政策接著將由事件 AWS CloudTrail `CreateDBInstance` 啟動。 因此，針對符合條件的執行個體，AWS Lambda 會將可公開存取的旗標設定為 `false`。	DevOps 工程師

相關資源

- AWS Lambda website
- Amazon RDS 網站
- 雲端託管人文件

其他資訊

雲端託管人政策 YAML 檔案

```
policies:
  - name: "block-public-access"
    resource: rds
    description: |
      This Enforcement blocks public access for RDS instances.
    mode:
      type: cloudtrail
      events:
```

```
        - event: CreateDBInstance # Create RDS instance cloudtrail event
          source: rds.amazonaws.com
          ids: requestParameters.dBInstanceIdentifier
      role: arn:aws:iam::1234567890:role/Custodian-compliance-role
    filters:
      - type: event
        key: 'detail.requestParameters.publiclyAccessible'
        value: true
    actions:
      - type: set-public-access
        state: false
```

c7n 資源 rds.py 檔案

```
@actions.register('set-public-access')
 class RDSSetPublicAvailability(BaseAction):

     schema = type_schema(
         "set-public-access",
         state={'type': 'boolean'})
     permissions = ('rds:ModifyDBInstance',)

     def set_accessibility(self, r):
         client = local_session(self.manager.session_factory).client('rds')
         waiter = client.get_waiter('db_instance_available')
         waiter.wait(DBInstanceIdentifier=r['DBInstanceIdentifier'])
         client.modify_db_instance(
             DBInstanceIdentifier=r['DBInstanceIdentifier'],
             PubliclyAccessible=self.data.get('state', False))


     def process(self, rds):
         with self.executor_factory(max_workers=2) as w:
             futures = {w.submit(self.set_accessibility, r): r for r in rds}
             for f in as_completed(futures):
                 if f.exception():
                     self.log.error(
                         "Exception setting public access on %s  \n %s",
                         futures[f]['DBInstanceIdentifier'], f.exception())
         return rds
```

Security Hub CSPM 整合

雲端託管人可與 整合AWS Security Hub CSPM，以傳送安全調查結果並嘗試修復動作。如需詳細資訊，請參閱與 宣布雲端託管整合 AWS Security Hub CSPM。

使用 設定 Amazon RDS for Microsoft SQL Server 的 Windows 身分驗證 AWS Managed Microsoft AD

Ramesh Babu Donti，Amazon Web Services

總結

此模式示範如何使用 AWS Directory Service for Microsoft Active Directory () 為 SQL Server 執行個體的 Amazon Relational Database Service (Amazon RDS) 設定 Windows 身分驗證AWS Managed Microsoft AD。Windows 身分驗證允許使用者使用其網域登入資料來連線至 RDS 執行個體，而不是資料庫特定的使用者名稱和密碼。

您可以在建立新的 RDS SQL Server 資料庫時啟用 Windows 身分驗證，或將其新增至現有的資料庫執行個體。資料庫執行個體與 整合 AWS Managed Microsoft AD ，為存取 SQL Server 資料庫的網域使用者提供集中式身分驗證和授權。

此組態透過利用現有的 Active Directory 基礎設施來增強安全性，無需為網域使用者管理單獨的資料庫登入資料。

先決條件和限制

先決條件

- AWS 帳戶 具有適當許可的作用中
- 具有下列項目的虛擬私有雲端 (VPC)：
  - 設定的網際網路閘道和路由表
  - 公有子網路中的 NAT 閘道 （如果執行個體需要網際網路存取）
- AWS Identity and Access Management (IAM) 角色：
  - 具有下列 AWS 受管政策的網域角色：
    - `AmazonSSMManagedInstanceCore` 以啟用 AWS Systems Manager
    - `AmazonSSMDirectoryServiceAccess` 提供將執行個體加入目錄的許可
  - RDS 增強型監控角色 （如果已啟用增強型監控）
- 安全群組：
  - 目錄服務安全群組，以允許 Active Directory 通訊連接埠
  - Amazon Elastic Compute Cloud (Amazon EC2) 安全群組，允許 RDP 3389和網域通訊
  - 允許1433來自授權來源之 SQL Server 連接埠的 RDS 安全群組

- 網路連接：
  - 子網路之間的適當 DNS 解析和網路連線

限制

- 如需 AWS Managed Microsoft AD RDS for SQL Server AWS 區域 支援的相關資訊，請參閱 區域和版本可用性。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

Architecture

來源技術堆疊

- 內部部署 Active Directory 或 AWS Managed Microsoft AD

目標技術堆疊

- Amazon EC2
- Amazon RDS for Microsoft SQL Server
- AWS Managed Microsoft AD

目標架構

架構包含下列項目：

- 將 Amazon EC2 執行個體加入 AWS Managed Microsoft AD 網域的 IAM 角色。
- 用於資料庫管理和測試的 Amazon EC2 Windows 執行個體。
- 具有私有子網路的 Amazon VPC，可跨可用區域託管 Amazon RDS 執行個體和內部資源。
- 網路存取控制的安全群組：
  - Amazon RDS 安全群組，用於控制1433從授權來源對 SQL Server 連接埠的傳入存取。
  - 透過連接埠3389和網域通訊連接埠管理 RDP 存取的 Amazon EC2 安全群組。
  - 透過連接埠 53、389、 88和 進行 Active Directory 通訊的 Directory Services 安全群組445。
- AWS Managed Microsoft AD 為 Windows 資源提供集中式身分驗證和授權服務。

- 啟用 Windows 身分驗證之私有子網路中的 Amazon RDS for SQL Server 資料庫執行個體。

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 中設定、操作和擴展關聯式資料庫 AWS 雲端。
- AWS Directory Service 提供多種搭配其他 使用 Microsoft Active Directory (AD) 的方式， AWS 服務 例如 Amazon Elastic Compute Cloud (Amazon EC2)、適用於 SQL Server 的 Amazon Relational Database Service (Amazon RDS) 和適用於 Windows File Server 的 Amazon FSx。
- AWS Directory Service for Microsoft Active Directory 可讓您的目錄感知工作負載 AWS 和資源在 中使用 Microsoft Active Directory AWS 雲端。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。

其他服務

- Microsoft SQL Server Management Studio (SSMS) 是一種用於管理 SQL Server 的工具，包括存取、設定和管理 SQL Server 元件。

最佳實務

- 如需一般最佳實務，請參閱 Amazon RDS 的最佳實務。

史詩

設定 AWS Managed Microsoft AD

任務	Description	所需的技能
設定目錄類型。	1. 從 AWS 管理主控台導覽至 AWS Directory Service。 2. 選擇設定目錄。	DBA，DevOps 工程師

任務	Description	所需的技能
	3. AWS Managed Microsoft AD 針對目錄類型選取 。 4. 選擇建立新的 AWS 受管 AD 網域，然後選擇下一步。	
設定目錄資訊。	在目錄資訊區段中，輸入必要資訊，並保留選用值： 1. 在版本下，選取符合您需求的版本。 2. 在目錄 DNS 名稱下，輸入完整網域名稱 (FQDN)。 3. 在管理員密碼下，設定管理員帳戶的密碼，然後選擇下一步。	DBA，DevOps 工程師
設定 VPC 和子網路。	1. 在聯網下，選取目標 VPC（您至少必須分別設定兩個子網路 AWS 可用區域)。 2. 在網路類型下，僅選取 IPv4。 3. 在子網路下，選取個別的兩個私有子網路 AWS 可用區域，然後選擇下一步。	DBA，DevOps 工程師
檢閱並建立 目錄。	1. 檢閱組態值，然後選擇建立目錄。 2. 等待目錄狀態變更為作用中。	DBA，DevOps 工程師

建立和設定 Amazon EC2 執行個體

任務	Description	所需的技能
設定適用於 Windows 的 AMI。	1. 從 AWS 管理主控台導覽至 EC2。 2. 選擇啟動執行個體。 3. 在名稱和標籤下，輸入名稱和任何適用的標籤。 4. 為 Windows Server 選擇符合您需求的 Amazon Machine Image (AMI)。 5. 在執行個體類型下，選取適當大小的類型。 6. 在金鑰對（登入）下，選取現有的金鑰對或建立新的金鑰對。	DBA，DevOps 工程師
設定網路設定。	1. 在網路設定下，選取用於 的相同 VPC AWS Directory Service。 2. 選擇私有子網路。 3. 在防火牆（安全群組）下，建立允許 RDP 連接埠3389和網域通訊的群組。	DBA，DevOps 工程師
設定儲存體。	視需要設定 Amazon EBS 磁碟區。	DBA，DevOps 工程師
設定進階詳細資訊並啟動執行個體。	1. 展開進階詳細資訊區段。 2. 對於網域聯結目錄，選取先前建立的 AWS Managed Microsoft AD。 3. 針對 IAM 執行個體描述檔，選取具有 政策和 AmazonSSMManagedIn	DBA，DevOps 工程師

任務	Description	所需的技能
	stanceCore 的角色AmazonSSMDirectoryServiceAccess 。 4. 檢閱所有組態值，然後選擇啟動執行個體。	

使用 Windows 身分驗證建立和設定 RDS for SQL Server

任務	Description	所需的技能
建立資料庫並設定引擎選項。	1. 導覽至 Aurora 和 RDS 主控台，然後選擇建立資料庫。 2. 在引擎選項下，選擇 Microsoft SQL Server。 3. 請為資料庫管理類型選擇 Amazon RDS。 4. 針對版本，選擇符合您需求的 SQL Server。 5. 針對引擎版本，選擇最新的支援版本。	DBA，DevOps 工程師
選擇一個範本。	選擇符合您需求的範例範本。	DBA，DevOps 工程師
設定資料庫設定。	1. 在資料庫執行個體識別符下的設定區段中，輸入唯一名稱。 2. 在主要使用者名稱下，設定管理員登入資料。 3. 在登入資料管理下，選擇在 中受管 AWS Secrets Manager或自我管理。	DBA，DevOps 工程師

任務	Description	所需的技能
設定執行個體。	在執行個體組態區段的資料庫執行個體類別下，選取符合您需求的執行個體大小。	DBA，DevOps 工程師
設定儲存體。	1. 在儲存區段的儲存類型下，選擇符合您需求的類型。我們建議使用 gp3、io1 或 io2。 2. 設定配置儲存、佈建 IOPS 和儲存輸送量所需的初始值。 3. （選用）展開其他儲存組態區段，然後選取啟用儲存體自動調整規模。	DBA，DevOps 工程師

任務	Description	所需的技能
設定連線。	1. 在連線區段中，選擇您是否要設定與資料庫運算資源的連線。 2. 針對 VPC，選擇與 相同的 VPC AWS Directory Service。 3. 針對資料庫子網路群組，選擇跨越多個可用區域的群組。 4. 針對公有存取，選擇否。 5. 針對 VPC 安全群組 （防火牆），選擇現有的群組或建立新的群組，以允許透過 SQL Server 連接埠 存取1433。 6. 選取您偏好的可用區域。 7. 展開其他組態區段，然後選擇您是否要使用自訂資料庫連接埠。	DBA，DevOps 工程師
設定 Windows 身分驗證。	1. 在 Microsoft SQL Server Windows 身分驗證區段中，選取啟用 Microsoft SQL Server Windows 身分驗證核取方塊。 2. 針對 Windows 身分驗證類型，選擇 AWS Managed Microsoft AD。 3. 針對目錄，選擇瀏覽目錄，然後選取 AWS Managed Microsoft AD。	DBA，DevOps 工程師

任務	Description	所需的技能
設定監控。	1. 在監控區段中，選擇標準或進階資料庫洞見。 2. 在效能洞察下，選取啟用效能洞察核取方塊。 3. 選取保留期間和 AWS KMS 金鑰。 4. 在其他監控設定下，選取增強型監控核取方塊。 5. （選用）在日誌匯出下，選取錯誤日誌核取方塊。 注意：當您想要查看不同的程序或執行緒如何使用 CPU 時，指標很有用。如果啟用錯誤日誌，您也可以將錯誤日誌匯出至 Amazon CloudWatch。	DBA，DevOps 工程師

任務	Description	所需的技能
設定其他設定。	1. 展開其他組態區段。 2. 針對資料庫參數群組和資料庫選項群組，選擇預設或自訂值。 3. 設定您偏好的時區。 4. 針對定序，設定值。預設值為 `SQL_Latin1_General_CP1_CI_AS` 。 5. 在備份下： • 選取啟用自動備份核取方塊。這會建立資料庫快照。 • 針對備份保留期間，選擇所需的天數。 • 針對備份視窗，選擇值。 • （選用）針對備份複寫，選擇在另一個 中啟用複寫 AWS 區域。 • 選取啟用加密核取方塊以使用 加密執行個體 AWS KMS。 6. 在維護時段下，選取選擇時段核取方塊，並設定偏好的時間。 7. 選取啟用刪除保護核取方塊。	DBA，DevOps 工程師
檢閱成本並建立資料庫。	檢閱預估每月成本區段，然後選擇建立資料庫。	DBA，DevOps 工程師

設定資料庫存取和測試連線

任務	Description	所需的技能
連線至 Windows 機器。	連接至 Windows 機器並啟動 SQL Server Management Studio。 1. 使用 RDP 使用 AWS Managed Microsoft AD 登入資料連線至您的 Windows 機器 2. 在開始功能表中輸入 SSMS 來啟動 SSMS，然後選取 SQL Server Management Studio。	DBA，DevOps 工程師
設定 SSMS 連線。	使用 Windows 身分驗證設定資料庫連線。 1. 出現連線至伺服器對話方塊時（或導覽至物件總管、連線、資料庫引擎)，請將伺服器類型設定為資料庫引擎。 2. 輸入 RDS SQL Server 端點（例如 `your-rds-instance.region.rds.amazonaws.com` ） 3. 選擇 Windows 身分驗證。	DBA，DevOps 工程師
設定安全設定。	設定 SSMS 第 20 版或更新版本的必要安全參數。 1. 在連線屬性索引標籤中，將加密設為強制性。 2. 選取信任伺服器憑證核取方塊。	DBA，DevOps 工程師

任務	Description	所需的技能
	3. 將憑證欄位中的主機名稱保留空白。 4. （選用）設定資料庫名稱並視需要調整連線逾時。	

建立 Windows 登入。	1. 設定和測試網域使用者的 Windows 身分驗證。
	2. 若要建立初始連線，請選擇連線。
	3. 在查詢視窗中，執行下列動作：
	`CREATE LOGIN [<domainName>\<user_name>] FROM WINDOWS;`
	`GO`
		DBA，DevOps 工程師

測試 Windows 身分驗證。	1. 登出 Amazon EC2 執行個體。 2. 使用您的網域登入資料重新登入 EC2 執行個體。 3. 啟動 SSMS。 4. 使用 Windows 身分驗證進行連線。 5. 驗證連線是否成功。	DBA，DevOps 工程師

相關資源

- 建立您的 AWS Managed Microsoft AD
- 將 Amazon EC2 Windows 執行個體加入您的 AWS Managed Microsoft AD
- Amazon RDS 中 Kerberos 身分驗證支援的區域和資料庫引擎
- 什麼是 SQL Server Management Studio (SSMS)？

設定對 Amazon DynamoDB 的跨帳戶存取權

Shashi Dalmia、Imhoertha Ojior 和 Esteban Serna Parra，Amazon Web Services

總結

此模式說明使用資源型政策設定 Amazon DynamoDB 跨帳戶存取的步驟。對於使用 DynamoDB 的工作負載，使用工作負載隔離策略將安全威脅降至最低並符合合規要求變得越來越常見。實作工作負載隔離策略通常需要跨帳戶和跨區域存取 DynamoDB 資源，方法是使用 AWS Identity and Access Management (IAM) 身分型政策。這包括設定 IAM 許可，以及在 之間建立信任關係 AWS 帳戶。

DynamoDB 的資源型政策可大幅簡化跨帳戶工作負載的安全狀態。此模式提供步驟和範例程式碼，示範如何在一個 中設定 AWS Lambda 函數 AWS 帳戶 ，以將資料寫入不同帳戶中的 DynamoDB 資料庫資料表。

先決條件和限制

先決條件

- 兩個作用中 AWS 帳戶。此模式將這些帳戶稱為帳戶 A 和帳戶 B。
- AWS Command Line Interface (AWS CLI) 已安裝並設定為存取帳戶 A，以建立 DynamoDB 資料表。此模式中的其他步驟提供使用 IAM、DynamoDB 和 Lambda 主控台的指示。如果您打算 AWS CLI 改用 ，請將其設定為存取這兩個帳戶。

限制

- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

Architecture

下圖顯示單一帳戶架構。 AWS Lambda、Amazon Elastic Compute Cloud (Amazon EC2) 和 DynamoDB 都位於同一個帳戶中。在此案例中，Lambda 函數和 Amazon EC2 執行個體可以存取 DynamoDB。若要授予 DynamoDB 資料表的存取權，您可以在 IAM 中建立身分型政策，也可以在 DynamoDB 中建立資源型政策。

下圖顯示多帳戶架構。如果其中一個資源 AWS 帳戶 需要存取不同帳戶中的 DynamoDB 資料表，您需要在 DynamoDB 中設定資源型政策，以授予所需的存取權。例如，在下圖中，帳戶 A 中 DynamoDB 資料表的存取權是透過使用資源型政策授予帳戶 B 中 Lambda 函數的。

此模式描述 Lambda 和 DynamoDB 之間的跨帳戶存取。 AWS 服務 如果兩個帳戶都設定了適當的許可，則可以對其他帳戶使用類似的步驟。例如，如果您想要提供 Lambda 函數存取帳戶 A 中的 Amazon Simple Storage Service (Amazon S3) 儲存貯體，您可以在 Amazon S3 中建立資源型政策，並將許可新增至帳戶 B 中的 Lambda 執行角色。

工具

AWS 服務

- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。

Code

此模式包含額外資訊區段中的範例程式碼，示範如何在帳戶 B 中設定 Lambda 函數，以寫入帳戶 A 中的 DynamoDB 資料表。此程式碼僅供說明和測試之用。如果您要在生產環境中實作此模式，請使用程式碼做為參考，並針對您自己的環境進行自訂。

最佳實務

- 遵循 DynamoDB 文件中資源型政策的最佳實務。
- 遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。

史詩

在帳戶 B 中為 Lambda 函數建立 IAM 政策和角色

任務	Description	所需的技能
在帳戶 B 中建立政策。	此 IAM 政策允許帳戶 A 中 DynamoDB 資料表的 PutItem 動作。 1. 在 中登入帳戶 B AWS 管理主控台。 2. 開啟 IAM 主控台。 3. 在導覽窗格中，選擇政策，然後選擇建立政策。 4. 在指定許可頁面上，針對政策編輯器選取 JSON。 5. 輸入下列政策。 `{ "Version": "2012-10-17", "Statement": [ { "Sid": "Statement1", "Effect": "Allow", "Action": "dynamodb:PutItem", "Resource": "arn:aws:dynamodb:<Region>:<Account-A-ID>:table/Table-Account-A" } ] }`	一般 AWS

任務	Description	所需的技能
	6. 將 <Region>和 取代<Account-A-ID> 為您的值，然後選擇下一步。 7. 針對政策名稱，輸入政策的唯一名稱，例如 DynamoDB-PutItem-Policy 。 8. （選用）新增政策描述。 9. 選擇建立政策。	

任務	Description	所需的技能
在帳戶 B 中建立角色。	帳戶 B 中的 Lambda 函數使用此 IAM 角色來存取帳戶 A 中的 DynamoDB 資料表。 1. 開啟 IAM 主控台。 2. 在導覽窗格中，選擇角色，然後選擇建立角色。 3. 在 Select trusted entity (選取受信任實體) 中，請選擇 AWS 服務。 4. 在使用案例區段中，選擇 Lambda。 5. 選擇下一步：許可。 6. 在篩選政策方塊中，輸入 DynamoDB。 7. 在 DynamoDB 政策清單中，選擇 `DynamoDB-PutItem-Policy` 。 8. 清除篩選政策方塊，然後輸入 Lambda。 9. 在 Lambda 政策清單中，選擇 AWSLambdaExecute。 10選擇下一步：名稱、檢閱和建立。 11針對 Role name (角色名稱)，為您的角色輸入唯一名稱 (例如 `DynamoDB-PutItemAccess`)。 12.(選用) 新增角色描述。 13(選用) 藉由連接標籤作為鍵值對，將中繼資料新增至角色。	一般 AWS

任務	Description	所需的技能
	14選擇建立角色。 如需建立角色的詳細資訊，請參閱 IAM 文件。	
請記下 角色 ARN。	1. 開啟 IAM 主控台。 2. 在導覽窗格中，選擇 Roles (角色)。 3. 在搜尋方塊中，輸入 `DynamoDB-PutItemAccess` ，然後選擇角色。 4. 在角色的摘要頁面上，複製 Amazon Resource Name (ARN)。您可以在設定 Lambda 函數時使用 ARN。	一般 AWS

在帳戶 A 中建立 DynamoDB 資料表

任務	Description	所需的技能

建立 DynamoDB 資料表。	使用下列 AWS CLI 命令來建立 DynamoDB 資料表。
	
	`aws dynamodb create-table \`
	`  --table-name Table-Account-A \`
	`  --attribute-definitions \`
	`  AttributeName=category,AttributeType=S \`
	`  AttributeName=item,AttributeType=S \`
		一般 AWS

任務	Description	所需的技能

	<pre>--key-schema \
	  Attribute Name=category,KeyType=HASH \
	  Attribute Name=item,KeyType=RANGE \
	--provisioned-throughput \
	  ReadCapacityUnits=5,WriteCapacityUnits=5 \
	--resource-policy \
	  '{
	  "Version": "2012-10-17",
	  "Statement": [
	  {
	  "Sid": "Statement1",
	  "Effect": "Allow",
	  "Principal": {
	  "AWS": "arn:aws:iam::<Account-B-ID>:role/<Role-Name>"
	  },
	  "Action": "dynamodb:PutItem",
	  "Resource": "arn:aws:dynamodb:<Region>:<Account-A-ID>:table/Table-Account-A"
	  }
	  ]</pre>	

任務	Description	所需的技能
	}'	

取代此程式碼範例中的以下內容：

- <Account-B-ID> 是帳戶 B 的 ID。
- <Role-Name> 是您建立的 IAM 角色名稱，例如 DynamoDB-PutItemAccess 。
- <Region> 是您建立 DynamoDB 資料表 AWS 區域 的 。
- <Account-A-ID> 是帳戶 A 的 ID。

> **Note**
>
> 您可以使用 --resource-policy旗標，在 create-table 陳述式中指定資源型政策組態。此政策是指帳戶 A 中 DynamoDB 資料表的 ARN。

如需建立資料表的詳細資訊，請參閱 DynamoDB 文件。

在帳戶 B 中建立 Lambda 函數

任務	Description	所需的技能
建立 Lambda 函數以將資料寫入 DynamoDB。	1. 在 中登入帳戶 B AWS 管理主控台。 2. 開啟 Lambda 主控台。 3. 在導覽面板上選擇函數，然後選擇建立函數。 4. 對於名稱，輸入 `lambda_write_function` 。 5. 針對執行期，選擇 Python 3.8 或更新版本。 6. 在變更預設執行角色下，選擇使用現有角色。 7. 針對現有角色，選擇您建立的 IAM 角色，例如 `DynamoDB-PutItemAccess` 。 8. 選擇 Create function (建立函數)。 9. 在程式碼索引標籤中，貼上此模式額外資訊區段中提供的範例程式碼。取代此程式碼範例中的以下內容： • `<Account-A-ID>` 是帳戶 A 的 ID。 • `<Region>` 是您建立 DynamoDB 資料表的 AWS 區域 。 10選擇部署。 11選擇測試。這會提示您設定測試事件。使用您偏好的	一般 AWS

任務	Description	所需的技能
	名稱建立新的事件，例如 `MyTestEventForWrite` ，然後儲存組態。 12.再次選擇 Test (測試)。這會使用您提供的事件名稱執行 Lambda 函數。 13.檢查函數的輸出。它應該指出函數存取帳戶 A 中的 DynamoDB 資料表，並且能夠將資料寫入其中。 如需建立 Lambda 函數的詳細資訊，請參閱 Lambda 文件。	

清除

任務	Description	所需的技能
刪除 資源。	若要避免產生與此模式中建立的資源相關的成本，請執行下列動作來刪除這些資源： 1. 在帳戶 B 中，刪除您建立以連線至 DynamoDB 的 Lambda 函數。如需說明，請參閱 Lambda 文件。 2. 在帳戶 A 中，刪除您建立的 DynamoDB 資料表。如需說明，請參閱 DynamoDB 文件。 3. 針對安全最佳實務，請刪除不再需要的 IAM 政策 (`DynamoDB-PutItem-P`	一般 AWS

任務	Description	所需的技能
	olicy)。如需詳細資訊，請參閱 IAM 文件。 4. 針對安全最佳實務，請刪除不再需要的 IAM 角色 (DynamoDB-PutItemAccess)。如需詳細資訊，請參閱 IAM 文件。	

疑難排解

問題	解決方案
建立 Lambda 函數時，您會收到ResourceNotFoundException 錯誤。	確認您已正確輸入帳戶 A 的 AWS 區域 和 ID。這些是 DynamoDB 資料表 ARN 的一部分。

相關資源

- DynamoDB 入門 (DynamoDB 文件)
- Lambda 入門 (Lambda 文件)
- 使用 DynamoDB 的資源型政策 (DynamoDB 文件)
- 建立 IAM 政策 (IAM 文件)
- 跨帳戶政策評估邏輯 (IAM 文件)
- IAM JSON 政策元素參考 (IAM 文件)

其他資訊

範例程式碼

```
import boto3
from datetime import datetime

dynamodb_client = boto3.client('dynamodb')
```

```
def lambda_handler(event, context):
     now = datetime.now().isoformat()
     data = dynamodb_client.put_item(TableName='arn:aws:dynamodb:<Region>:<Account-
A-ID>:table/Table-Account-A', Item={"category": {"S": "Fruit"},"item": {"S":
 "Apple"},"time": {"S": now}})
     return data
```

> **Note**
>
> 執行個體化 DynamoDB 用戶端時，會提供 DynamoDB 資料表的 ARN，而不是資料表名稱。這是必要的，以便 Lambda 函數在執行時連接到正確的 DynamoDB 資料表。

在 AWS 上 SQL Server 的 Always On 可用性群組中設定唯讀路由

Subhani Shaik，Amazon Web Services

總結

此模式涵蓋如何在 SQL Server Always On 中使用待命次要複本，方法是將唯讀工作負載從主要複本卸載至次要複本。

資料庫鏡像具有one-to-one映射。您無法直接讀取次要資料庫，因此您必須建立快照。Always On 可用性群組功能已在 Microsoft SQL Server 2012 中推出。在較新版本中，已引進主要功能，包括唯讀路由。在 Always On 可用性群組中，您可以將複本模式變更為唯讀，以直接從次要複本讀取資料。

Always On 可用性群組解決方案支援高可用性 (HA)、災難復原 (DR)，以及資料庫鏡像的替代方案。Always On 可用性群組可在資料庫層級運作，並將一組使用者資料庫的可用性最大化。

SQL Server 使用唯讀路由機制，將傳入的唯讀連線重新導向至次要僅供讀取複本。若要達成此目的，您應該在連線字串中新增下列參數和值：

- `ApplicationIntent=ReadOnly`
- `Initial Catalog=<database name>`

先決條件和限制

先決條件

- 具有虛擬私有雲端 (VPC)、兩個可用區域、私有子網路和安全群組的作用中 AWS 帳戶
- 兩部 Amazon Elastic Compute Cloud (Amazon EC2) 機器，具有在執行個體層級設定的 SQL Server 2019 Enterprise Edition Amazon Machine Image with Windows Server 容錯移轉叢集 (WSFC)，以及在主要節點 () 和次要節點 (`WSFCNODE1`) 之間的 SQL Server 層級設定的 Always On 可用性群組`WSFCNODE2`，這是名為 AWS Directory Service for Microsoft Active Directory 目錄的一部分 `tagechtalk.com`
- 在次要複本`read-only`中設定為接受的一或多個節點
- 名為 Always On 可用性群組`SQLAG1`的接聽程式
- 在兩個節點上使用相同服務帳戶執行的 SQL Server Database Engine
- SQL Server Management Studio (SSMS)
- 名為 的測試資料庫 `test`

產品版本

- SQL Server 2014 及更新版本

Architecture

目標技術堆疊

- Amazon EC2
- AWS 受管 Microsoft AD
- Amazon FSx

目標架構

下圖顯示 Always On 可用性群組 (AG) 接聽程式如何將連線中包含 `ApplicationIntent` 參數的查詢重新導向至適當的次要節點。

1. 請求會傳送至 Always On 可用性群組接聽程式。
2. 如果連線字串沒有 `ApplicationIntent` 參數，請求會傳送至主要執行個體。
3. 如果連線字串包含 `ApplicationIntent=ReadOnly`，則請求會傳送至具有唯讀路由組態 的次要執行個體，這是具有 Always On 可用性群組的 WSFC。

工具

AWS 服務

- AWS Directory Service for Microsoft Active Directory 可讓您的目錄感知工作負載和 AWS 資源在 AWS 雲端中使用 Microsoft Active Directory。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon FSx 提供支援業界標準連線通訊協定的檔案系統，並跨 AWS 區域提供高可用性和複寫功能。

其他服務

- SQL Server Management Studio (SSMS) 是一種用於連接、管理和管理 SQL Server 執行個體的工具。
- sqlcmd 是命令列公用程式。

最佳實務

如需 Always On 可用性群組的詳細資訊，請參閱 SQL Server 文件。

史詩

設定唯讀路由

任務	Description	所需的技能
將複本更新為唯讀。	若要同時將主要複本和次要複本更新為唯讀，請從 SSMS 連線至主要複本，然後從其他資訊區段執行步驟 1 程式碼。	DBA
建立路由 URL。	若要建立兩個複本的路由 URL，請從其他資訊區段執行步驟 2 程式碼。在此程式碼中，`tagechtalk.com` 是 AWS Managed Microsoft AD 目錄的名稱。	DBA
建立路由清單。	若要建立兩個複本的路由清單，請從其他資訊區段執行步驟 3 程式碼。	DBA
驗證路由清單。	從 SQL Server Management Studio 連線至主要執行個體，並從其他資訊區段執行步驟 4 程式碼，以驗證路由清單。	DBA

測試唯讀路由

任務	Description	所需的技能
使用 ApplicationIntent 參數進行連線。	1. 從 SSMS，使用 連線至 Always On 可用性群組接聽程式名稱`ApplicationIntent=ReadOnly; Initial Catalog=test` 。 2. 使用次要複本建立連線。若要測試，請執行下列命令以顯示連線的伺服器名稱。 `SELECT SERVERPROPERTY('ComputerNamePhysicalNetBios')` 輸出會顯示目前的次要複本名稱 (`WSFCNODE2`)。	DBA
執行容錯移轉。	1. 從 SSMS 連線至 Always On 可用性群組接聽程式名稱。 2. 確認主要和次要資料庫處於同步狀態，而不會遺失資料。 3. 執行容錯移轉，讓目前的主要複本成為次要複本，而次要複本則成為主要複本。 4. 從 SSMS，使用 連線至 Always On 可用性群組接聽程式名稱`ApplicationIntent=ReadOnly; Initial Catalog=test` 。	DBA

任務	Description	所需的技能
	5. 使用次要複本建立連線。若要測試此項目，請執行下列命令來顯示連線的伺服器名稱。 `SELECT SERVERPROPERTY('ComputerNamePhysicalNetBios')` 它會顯示目前的次要複本名稱 (WSFCNODE1)。	

使用 sqlcmd 命令列公用程式進行連線

任務	Description	所需的技能
使用 sqlcmd 連線。	若要從 sqlcmd 連線，請從命令提示字元的其他資訊區段執行步驟 5 程式碼。連線後，請執行下列命令以顯示連線的伺服器名稱。 `SELECT SERVERPROPERTY('ComputerNamePhysicalNetBios') .` 輸出會顯示目前的次要複本名稱 (WSFCNODE1)。	DBA

疑難排解

問題	解決方案
建立接聽程式失敗，並顯示「WSFC 叢集無法讓網路名稱資源上線」訊息。	如需詳細資訊，請參閱 Microsoft 部落格文章 Create Listener Fails with Message 'WSFC 叢集無法讓網路名稱資源上線'。
潛在問題，包括其他接聽程式問題或網路存取問題。	請參閱 Microsoft 文件中的 Always On 可用性群組組態 (SQL Server) 故障診斷。

相關資源

- 設定 Always On 可用性群組的唯讀路由
- 故障診斷 Always On 可用性群組組態 (SQL Server)

其他資訊

步驟 1. 將複本更新為唯讀

```
ALTER AVAILABILITY GROUP [SQLAG1] MODIFY REPLICA ON N'WSFCNODE1' WITH (SECONDARY_ROLE
 (ALLOW_CONNECTIONS = READ_ONLY))
GO
ALTER AVAILABILITY GROUP [SQLAG1] MODIFY REPLICA ON N'WSFCNODE2' WITH (SECONDARY_ROLE
 (ALLOW_CONNECTIONS = READ_ONLY))
GO
```

步驟 2. 建立路由 URL

```
ALTER AVAILABILITY GROUP [SQLAG1] MODIFY REPLICA ON N'WSFCNODE1' WITH (SECONDARY_ROLE
 (READ_ONLY_ROUTING_URL = N'TCP://WSFCNode1.tagechtalk.com:1433'))
GO
ALTER AVAILABILITY GROUP [SQLAG1] MODIFY REPLICA ON N'WSFCNODE2' WITH (SECONDARY_ROLE
 (READ_ONLY_ROUTING_URL = N'TCP://WSFCNode2.tagechtalk.com:1433'))
GO
```

步驟 3. 建立路由清單

```
ALTER AVAILABILITY GROUP [SQLAG1] MODIFY REPLICA ON N'WSFCNODE1' WITH
 (PRIMARY_ROLE(READ_ONLY_ROUTING_LIST=('WSFCNODE2','WSFCNODE1')));
GO
ALTER AVAILABILITY GROUP [SQLAG1] MODIFY REPLICA ON N'WSFCNODE2' WITH (PRIMARY_ROLE
 (READ_ONLY_ROUTING_LIST=('WSFCNODE1','WSFCNODE2')));
GO
```

步驟 4. 驗證路由清單

```
SELECT AGSrc.replica_server_name AS PrimaryReplica, AGRepl.replica_server_name AS
 ReadOnlyReplica, AGRepl.read_only_routing_url AS RoutingURL , AGRL.routing_priority
 AS RoutingPriority FROM sys.availability_read_only_routing_lists AGRL INNER JOIN
 sys.availability_replicas AGSrc ON AGRL.replica_id = AGSrc.replica_id INNER JOIN
 sys.availability_replicas AGRepl ON AGRL.read_only_replica_id = AGRepl.replica_id
 INNER JOIN sys.availability_groups AV ON AV.group_id = AGSrc.group_id ORDER BY
 PrimaryReplica
```

步驟 5. SQL 命令公用程式

```
sqlcmd -S SQLAG1,1433 -E -d test -K ReadOnly
```

在 pgAdmin 中使用 SSH 通道進行連線

Jeevan Shetty 和 Bhanu Ganesh Gudivada，Amazon Web Services

總結

基於安全考量，將資料庫放置在私有子網路中一律是不錯的做法。透過 Amazon Web Services (AWS) 雲端上的公有子網路中的 Amazon Elastic Compute Cloud (Amazon EC2) 堡壘主機，可以對資料庫執行查詢。這需要在 Amazon EC2 主機上安裝開發人員或資料庫管理員常用的軟體，例如 pgAdmin 或 DBeaver。

在 Linux 伺服器上執行 pgAdmin 並透過 Web 瀏覽器存取，需要安裝其他相依性、許可設定和組態。

做為替代解決方案，開發人員或資料庫管理員可以使用 pgAdmin 從本機系統啟用 SSH 通道，以連線至 PostgreSQL 資料庫。在此方法中，pgAdmin 會在連線至資料庫之前，使用公有子網路中的 Amazon EC2 主機做為中介主機。架構區段中的圖表顯示設定。

> **Note**
>
> 確定連接至 PostgreSQL 資料庫的安全群組允許從 Amazon EC2 主機連線連接埠 5432。

先決條件和限制

先決條件

- 現有的 AWS 帳戶
- 具有公有子網路和私有子網路的虛擬私有雲端 (VPC)
- 連接安全群組的 EC2 執行個體
- 連接安全群組的 Amazon Aurora PostgreSQL 相容版本資料庫
- 用於設定通道的安全殼層 (SSH) 金鑰對

產品版本

- pgAdmin 6.2+ 版
- Amazon Aurora PostgreSQL 相容版本 12.7+

Architecture

目標技術堆疊

- Amazon EC2
- Amazon Aurora PostgreSQL 相容

目標架構

下圖顯示搭配 SSH 通道使用 pgAdmin 來透過網際網路閘道連線至 EC2 執行個體，而 EC2 執行個體會連線至資料庫。

工具

AWS 服務

- Amazon Aurora PostgreSQL 相容版本是完全受管的 ACID 相容關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。

其他服務

- pgAdmin 是 PostgreSQL 的開放原始碼管理工具。它提供圖形界面，可協助您建立、維護和使用資料庫物件。

史詩

建立連線

任務	Description	所需的技能
建立伺服器。	在 pgAdmin 中，選擇建立，然後選擇伺服器。如需設定 pgAdmin 以註冊伺服器、設定連線，以及使用伺服器對話方塊透過 SSH 通道連線的其他說	DBA

任務	Description	所需的技能
	明，請參閱相關資源一節中的連結。	
提供伺服器的名稱。	在一般索引標籤上，輸入名稱。	DBA
輸入資料庫詳細資訊。	在連線索引標籤上，輸入下列項目的值： • 主機名稱/地址 • 連接埠 • 維護資料庫 • 使用者名稱 • 密碼	DBA

任務	Description	所需的技能
輸入 Amazon EC2 伺服器詳細資訊。	在 SSH 通道索引標籤上，提供公有子網路中 Amazon EC2 執行個體的詳細資訊。 • 將使用 SSH 通道設定為是，以指定 pgAdmin 在連線至指定的伺服器時應使用 SSH 通道。 • 在通道主機欄位中，指定 SSH 主機的名稱或 IP 地址（例如 10.x.x.x)。 • 在通道連接埠欄位中，指定 SSH 主機的連接埠（例如 22)。 • 在使用者名稱欄位中，指定具有 SSH 主機登入權限的使用者名稱（例如，ec2-user)。 • 將身分驗證類型指定為身分檔案，以便 pgAdmin 在連線時使用私有金鑰檔案。 • 在身分檔案欄位中包含隱私權增強郵件 (PEM) 檔案的位置。.pem 檔案是 Amazon EC2 金鑰對。	DBA
儲存並連線。	選擇儲存以完成設定，並使用 SSH 通道連線至 Aurora PostgreSQL 相容資料庫。	DBA

相關資源

- 伺服器對話方塊

- 連線至伺服器

將 JSON Oracle 查詢轉換為 PostgreSQL 資料庫 SQL

Pinesh Singal 和 Lokesh Gurram，Amazon Web Services

總結

此從內部部署移至 Amazon Web Services (AWS) Cloud 的遷移程序會使用 AWS Schema Conversion Tool (AWS SCT)，將程式碼從 Oracle 資料庫轉換為 PostgreSQL 資料庫。AWS SCT 會自動轉換大部分的程式碼。不過，JSON 相關的 Oracle 查詢不會自動轉換。

從 Oracle 12.2 版本開始，Oracle Database 支援各種 JSON 函數，可協助將 JSON 型資料轉換為 ROW 型資料。不過，AWS SCT 不會自動將 JSON 型資料轉換為 PostgreSQL 支援的語言。

此遷移模式主要著重於使用 `JSON_OBJECT`、 `JSON_ARRAYAGG`和 等函數，將 JSON 相關的 Oracle 查詢`JSON_TABLE`從 Oracle 資料庫手動轉換為 PostgreSQL 資料庫。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署 Oracle 資料庫執行個體 （啟動和執行中）
- 適用於 PostgreSQL 的 Amazon Relational Database Service (Amazon RDS) 或 Amazon Aurora PostgreSQL 相容版本資料庫執行個體 （啟動和執行中）

限制

- JSON 相關查詢需要固定的 `KEY`和 `VALUE` 格式。不使用該格式會傳回錯誤的結果。
- 如果 JSON 結構中的任何變更在結果區段中新增新的 `KEY`和 `VALUE`對，則必須在 SQL 查詢中變更對應的程序或函數。
- 舊版 Oracle 和 PostgreSQL 支援某些 JSON 相關函數，但功能較少。

產品版本

- Oracle 資料庫 12.2 版及更新版本
- Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 相容 9.5 版及更新版本
- AWS SCT 最新版本 （使用 1.0.664 版測試）

Architecture

來源技術堆疊

- 版本為 19c 的 Oracle 資料庫執行個體

目標技術堆疊

- 版本為 13 的 Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 相容資料庫執行個體

目標架構

1. 使用 AWS SCT 搭配 JSON 函數程式碼，將原始程式碼從 Oracle 轉換為 PostgreSQL。
2. 轉換會產生 PostgreSQL 支援的遷移 .sql 檔案。
3. 手動將非轉換的 Oracle JSON 函數程式碼轉換為 PostgreSQL JSON 函數程式碼。
4. 在目標 Aurora PostgreSQL 相容資料庫執行個體上執行 .sql 檔案。

工具

AWS 服務

- Amazon Aurora 是一種全受管關聯式資料庫引擎，專為雲端而建置，並與 MySQL 和 PostgreSQL 相容。
- 適用於 PostgreSQL 的 Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展 PostgreSQL 關聯式資料庫。
- AWS Schema Conversion Tool (AWS SCT) 會自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，以支援異質資料庫遷移。

其他服務

- Oracle SQL Developer 是一種整合的開發環境，可簡化傳統和雲端部署中 Oracle 資料庫的開發和管理。
- pgAdmin 或 DBeaver。pgAdmin 是 PostgreSQL 的開放原始碼管理工具。它提供圖形界面，可協助您建立、維護和使用資料庫物件。DBeaver 是一種通用資料庫工具。

最佳實務

使用 JSON_TABLE函數時，Oracle 查詢的預設CAST類型為 。最佳實務也是在 PostgreSQL CAST中使用，使用兩倍大於字元 (>>)。

如需詳細資訊，請參閱其他資訊區段中的 Postgres_SQL_Read_JSON。

史詩

在 Oracle 和 PostgreSQL 資料庫中產生 JSON 資料

任務	Description	所需的技能
將 JSON 資料存放在 Oracle 資料庫中。	在 Oracle 資料庫中建立資料表，並將 JSON 資料存放在 CLOB欄中。 使用額外資訊區段中的 Oracle_Table_Creation_Insert_Script。	遷移工程師
將 JSON 資料存放在 PostgreSQL 資料庫中。	在 PostgreSQL 資料庫中建立資料表，並將 JSON 資料存放在 TEXT欄中。使用其他資訊區段中的 Postgres_Table_Creation_Insert_Script。	遷移工程師

將 JSON 轉換為 ROW 格式

任務	Description	所需的技能
轉換 Oracle 資料庫上的 JSON 資料。	撰寫 Oracle SQL 查詢，以 ROW 格式讀取 JSON 資料。如需詳細資訊和語法範例，請參閱其他資訊區段中的 Oracle_SQL_Read_JSON。	遷移工程師
轉換 PostgreSQL 資料庫上的 JSON 資料。	撰寫 PostgreSQL 查詢，以 ROW 格式讀取 JSON 資料。如需詳細資訊和語法範	遷移工程師

任務	Description	所需的技能
	例，請參閱其他資訊區段中的 Postgres_SQL_Read_JSON。	

使用 SQL 查詢手動轉換 JSON 資料，並以 JSON 格式報告輸出

任務	Description	所需的技能
在 Oracle SQL 查詢上執行彙總和驗證。	若要手動轉換 JSON 資料，請對 Oracle SQL 查詢執行聯結、彙總和驗證，並以 JSON 格式報告輸出。在其他資訊區段中的 Oracle_SQL_JSON_Aggregation_Join 下使用程式碼。 1. JOIN – JSON 格式的資料會以輸入參數的形式傳遞至查詢。此靜態資料與 Oracle 資料庫資料表 中的 JSON 資料之間會建立內部 JOINaws_test_table 。 2. 使用驗證彙總 – JSON 資料具有 KEY和 VALUE 參數，其值如 accountNumber 、parentAccountNumber businessUnitId 和 positionId ，用於 COUNT SUM和 彙總。 3. JSON 格式 – 在聯結和彙總之後，會使用 JSON_OBJECT 和 以 JSON 格式報告資料JSON_ARRAYAGG 。	遷移工程師

任務	Description	所需的技能
在 Postgres SQL 查詢上執行彙總和驗證。	若要手動轉換 JSON 資料，請對 PostgreSQL 查詢執行聯結、彙總和驗證，並以 JSON 格式報告輸出。在其他資訊區段的 Postgres_SQL_JSON_Aggregation_Join 下使用程式碼。 1. JOIN – JSON 格式的資料 (tab1) 會以輸入參數的形式傳遞至WITH子句查詢。JOIN 會在此靜態資料與 JSON 資料之間建立，該資料位於 tab 資料表中。JOIN 也會使用WITH子句製作，該子句在 aws_test_pg_table 資料表中具有 JSON 資料。 2. 彙總 – JSON 資料具有KEY和 VALUE 參數，其值例如 accountNumber 、businessUnitId 、 parentAccountNumber 和 positionId ，用於COUNT SUM和 彙總。 3. JSON 格式 – 在聯結和彙總之後，會使用 JSON_BUILD_OBJECT 和 以 JSON 格式報告資料JSON_AGG。	遷移工程師

將 Oracle 程序轉換為包含 JSON 查詢的 PostgreSQL 函數

任務	Description	所需的技能
將 Oracle 程序中的 JSON 查詢轉換為資料列。	如需 Oracle 程序範例，請使用上一個 Oracle 查詢，以及其他資訊區段中的代碼 underOracle_procedure_with_JSON_Query。	遷移工程師
將具有 JSON 查詢的 PostgreSQL 函數轉換為資料列型資料。	如需 PostgreSQL 函數範例，請使用先前的 PostgreSQL 查詢，以及其他資訊區段中 Postgres_function_with_JSON_Query 下的程式碼。	遷移工程師

相關資源

- Oracle JSON 函數
- PostgreSQL JSON 函數
- Oracle JSON 函數範例
- PostgreSQL JSON 函數範例
- AWS Schema Conversion Tool

其他資訊

若要將 JSON 程式碼從 Oracle 資料庫轉換為 PostgreSQL 資料庫，請依序使用下列指令碼。

1。Oracle_Table_Creation_Insert_Script

```
create table aws_test_table(id number,created_on date default sysdate,modified_on
 date,json_doc clob);

REM INSERTING into EXPORT_TABLE
SET DEFINE OFF;
Insert into aws_test_table (ID,CREATED_ON,MODIFIED_ON,json_doc)
```

```
values (1,to_date('02-AUG-2022 12:30:14','DD-MON-YYYY HH24:MI:SS'),to_date('02-AUG-2022
 12:30:14','DD-MON-YYYY HH24:MI:SS'),TO_CLOB(q'[{
  "metadata" : {
    "upperLastNameFirstName" : "ABC XYZ",
    "upperEmailAddress" : "abc@gmail.com",
    "profileType" : "P"
  },
  "data" : {
    "onlineContactId" : "032323323",
    "displayName" : "Abc, Xyz",
    "firstName" : "Xyz",
    "lastName" : "Abc",
    "emailAddress" : "abc@gmail.com",
    "productRegistrationStatus" : "Not registered",
    "positionId" : "0100",
    "arrayPattern" : " -'",
    "a]')
|| TO_CLOB(q'[ccount" : {
      "companyId" : "SMGE",
      "businessUnitId" : 7,
      "accountNumber" : 42000,
      "parentAccountNumber" : 32000,
      "firstName" : "john",
      "lastName" : "doe",
      "street1" : "retOdertcaShr ",
      "city" : "new york",
      "postalcode" : "XY ABC",
      "country" : "United States"
    },
    "products" : [
      {
        "appUserGuid" : "i0acc4450000001823fbad478e2eab8a0",
        "id" : "0000000046",
]')
|| TO_CLOB(q'[        "name" : "ProView",
        "domain" : "EREADER",
        "registrationStatus" : false,
        "status" : "11"
      }
    ]
  }
}]'));
```

```
Insert into aws_test_table (ID,CREATED_ON,MODIFIED_ON,json_doc) values (2,to_date('02-
AUG-2022 12:30:14','DD-MON-YYYY HH24:MI:SS'),to_date('02-AUG-2022 12:30:14','DD-MON-
YYYY HH24:MI:SS'),TO_CLOB(q'[{
  "metadata" : {
    "upperLastNameFirstName" : "PQR XYZ",
    "upperEmailAddress" : "pqr@gmail.com",
    "profileType" : "P"
  },
  "data" : {
    "onlineContactId" : "54534343",
    "displayName" : "Xyz, pqr",
    "firstName" : "pqr",
    "lastName" : "Xyz",
    "emailAddress" : "pqr@gmail.com",
    "productRegistrationStatus" : "Not registered",
    "positionId" : "0090",
    "arrayPattern" : " -'",
    "account" : {
      "companyId" : "CARS",
      "busin]')
|| TO_CLOB(q'[essUnitId" : 6,
      "accountNumber" : 42001,
      "parentAccountNumber" : 32001,
      "firstName" : "terry",
      "lastName" : "whitlock",
      "street1" : "UO  123",
      "city" : "TOTORON",
      "region" : "NO",
      "postalcode" : "LKM 111",
      "country" : "Canada"
    },
    "products" : [
      {
        "appUserGuid" : "ia744d7790000016899f8cf3f417d6df6",
        "id" : "0000000014",
        "name" : "ProView eLooseleaf",
      ]')
|| TO_CLOB(q'[  "domain" : "EREADER",
        "registrationStatus" : false,
        "status" : "11"
      }
    ]
  }
}]'));
```

```
commit;
```

2. Postgres_Table_Creation_Insert_Script

```
create table aws_test_pg_table(id int,created_on date ,modified_on date,json_doc text);
insert into aws_test_pg_table(id,created_on,modified_on,json_doc)
values(1,now(),now(),'{
  "metadata" : {
    "upperLastNameFirstName" : "ABC XYZ",
    "upperEmailAddress" : "abc@gmail.com",
    "profileType" : "P"
  },
  "data" : {
    "onlineContactId" : "032323323",
    "displayName" : "Abc, Xyz",
    "firstName" : "Xyz",
    "lastName" : "Abc",
    "emailAddress" : "abc@gmail.com",
    "productRegistrationStatus" : "Not registered",
    "positionId" : "0100",
    "arrayPattern" : " -",
    "account" : {
      "companyId" : "SMGE",
      "businessUnitId" : 7,
      "accountNumber" : 42000,
      "parentAccountNumber" : 32000,
      "firstName" : "john",
      "lastName" : "doe",
      "street1" : "retOdertcaShr ",
      "city" : "new york",
      "postalcode" : "XY ABC",
      "country" : "United States"
    },
    "products" : [
      {
        "appUserGuid" : "i0acc4450000001823fbad478e2eab8a0",
        "id" : "0000000046",
        "name" : "ProView",
        "domain" : "EREADER",
        "registrationStatus" : false,
        "status" : "11"
      }
```

```
    ]
  }
}');


insert into aws_test_pg_table(id,created_on,modified_on,json_doc)
values(2,now(),now(),'{
  "metadata" : {
    "upperLastNameFirstName" : "PQR XYZ",
    "upperEmailAddress" : "pqr@gmail.com",
    "profileType" : "P"
  },
  "data" : {
    "onlineContactId" : "54534343",
    "displayName" : "Xyz, pqr",
    "firstName" : "pqr",
    "lastName" : "Xyz",
    "emailAddress" : "a*b**@h**.k**",
    "productRegistrationStatus" : "Not registered",
    "positionId" : "0090",
    "arrayPattern" : " -",
    "account" : {
      "companyId" : "CARS",
      "businessUnitId" : 6,
      "accountNumber" : 42001,
      "parentAccountNumber" : 32001,
      "firstName" : "terry",
      "lastName" : "whitlock",
      "street1" : "UO  123",
      "city" : "TOTORON",
      "region" : "NO",
      "postalcode" : "LKM 111",
      "country" : "Canada"
    },
    "products" : [
      {
        "appUserGuid" : "ia744d7790000016899f8cf3f417d6df6",
        "id" : "0000000014",
        "name" : "ProView eLooseleaf",
        "domain" : "EREADER",
        "registrationStatus" : false,
        "status" : "11"
      }
    ]
```

```
  }
}');
```

3. Oracle_SQL_Read_JSON

下列程式碼區塊顯示如何將 Oracle JSON 資料轉換為資料列格式。

查詢和語法範例

```
SELECT   JSON_OBJECT(
 'accountCounts' VALUE JSON_ARRAYAGG(
            JSON_OBJECT(
                'businessUnitId' VALUE business_unit_id,
                        'parentAccountNumber' VALUE parent_account_number,
                        'accountNumber' VALUE account_number,
                        'totalOnlineContactsCount' VALUE online_contacts_count,
                        'countByPosition' VALUE
                    JSON_OBJECT(
                        'taxProfessionalCount' VALUE tax_count,
                        'attorneyCount' VALUE attorney_count,
                        'nonAttorneyCount' VALUE non_attorney_count,
                        'clerkCount' VALUE clerk_count
                               ) ) ) ) FROM
   (SELECT   tab_data.business_unit_id,
            tab_data.parent_account_number,
            tab_data.account_number,
            SUM(1) online_contacts_count,
            SUM(CASE WHEN tab_data.position_id = '0095' THEN  1 ELSE 0 END) tax_count,
            SUM(CASE    WHEN tab_data.position_id = '0100' THEN 1 ELSE 0 END)
 attorney_count,
            SUM(CASE    WHEN tab_data.position_id = '0090' THEN 1 ELSE 0 END)
 non_attorney_count,
            SUM(CASE    WHEN tab_data.position_id = '0050' THEN 1 ELSE 0 END)
 clerk_count
       FROM aws_test_table scco,JSON_TABLE ( json_doc, '$' ERROR ON ERROR
 COLUMNS (
         parent_account_number NUMBER PATH
          '$.data.account.parentAccountNumber',
           account_number NUMBER PATH '$.data.account.accountNumber',
           business_unit_id NUMBER PATH '$.data.account.businessUnitId',
           position_id VARCHAR2 ( 4 ) PATH '$.data.positionId'    )
           ) AS tab_data
           INNER JOIN JSON_TABLE ( '{
       "accounts": [{
```

```
          "accountNumber": 42000,
          "parentAccountNumber": 32000,
          "businessUnitId": 7
        }, {
          "accountNumber": 42001,
          "parentAccountNumber": 32001,
          "businessUnitId": 6
        }]
      }', '$.accounts[*]' ERROR ON ERROR
      COLUMNS (
      parent_account_number PATH '$.parentAccountNumber',
      account_number PATH '$.accountNumber',
      business_unit_id PATH '$.businessUnitId')
      ) static_data
      ON ( static_data.parent_account_number = tab_data.parent_account_number
           AND static_data.account_number = tab_data.account_number
           AND static_data.business_unit_id = tab_data.business_unit_id )
        GROUP BY
            tab_data.business_unit_id,
            tab_data.parent_account_number,
            tab_data.account_number );
```

JSON 文件會將資料儲存為集合。每個集合都可以有 KEY和 VALUE對。每個 VALUE都可以有巢狀 KEY和 VALUE對。下表提供VALUE從 JSON 文件讀取特定 的相關資訊。

KEY	用來取得 VALUE 的 HIERARCHY 或 PATH	VALUE
`profileType`	`metadata` -> `profileType`	"P"
`positionId`	`data` -> `positionId`	"0100"
`accountNumber`	`data` -> 帳戶 -> `accountNumber`	42000

在上表中， KEY`profileType`是 `metadata` VALUE的 KEY。KEY `positionId` 是 VALUE的 `data` KEY。KEY `accountNumber` 是 VALUE的 `account` KEY，而 `account`KEY是 VALUE的 `data` KEY。

範例 JSON 文件

```
{
```

```
  "metadata" : {
    "upperLastNameFirstName" : "ABC XYZ",
    "upperEmailAddress" : "abc@gmail.com",
"profileType" : "P"
  },
  "data" : {
    "onlineContactId" : "032323323",
    "displayName" : "Abc, Xyz",
    "firstName" : "Xyz",
    "lastName" : "Abc",
    "emailAddress" : "abc@gmail.com",
    "productRegistrationStatus" : "Not registered",
"positionId" : "0100",
    "arrayPattern" : " -",
    "account" : {
      "companyId" : "SMGE",
      "businessUnitId" : 7,
"accountNumber" : 42000,
      "parentAccountNumber" : 32000,
      "firstName" : "john",
      "lastName" : "doe",
      "street1" : "retOdertcaShr ",
      "city" : "new york",
      "postalcode" : "XY ABC",
      "country" : "United States"
    },
    "products" : [
      {
        "appUserGuid" : "i0acc4450000001823fbad478e2eab8a0",
        "id" : "0000000046",
        "name" : "ProView",
        "domain" : "EREADER",
        "registrationStatus" : false,
        "status" : "11"
      }
    ]
  }
}
```

用於從 JSON 文件取得所選欄位的 SQL 查詢

```
select parent_account_number,account_number,business_unit_id,position_id from
 aws_test_table aws,JSON_TABLE ( json_doc, '$' ERROR ON ERROR
```

```
COLUMNS (
parent_account_number NUMBER PATH '$.data.account.parentAccountNumber',
account_number NUMBER PATH '$.data.account.accountNumber',
business_unit_id NUMBER PATH '$.data.account.businessUnitId',
position_id VARCHAR2 ( 4 ) PATH '$.data.positionId'
)) as sc
```

在先前的查詢中， JSON_TABLE 是 Oracle 中的內建函數，可將 JSON 資料轉換為資料列格式。JSON_TABLE 函數預期 JSON 格式的參數。

中的每個項目COLUMNS都有預先定義的 PATH，並且會以資料列格式KEY傳回VALUE適合給定的項目。

上一個查詢的結果

PARENT_AC COUNT_NUMBER	ACCOUNT_NUMBER	BUSINESS_UNIT_ID	POSITION_ID
32000	42000	7	0100
32001	42001	6	0090

4. Postgres_SQL_Read_JSON

查詢和語法範例

```
select *
from (
select (json_doc::json->'data'->'account'->>'parentAccountNumber')::INTEGER as
 parentAccountNumber,
(json_doc::json->'data'->'account'->>'accountNumber')::INTEGER as accountNumber,
(json_doc::json->'data'->'account'->>'businessUnitId')::INTEGER as businessUnitId,
(json_doc::json->'data'->>'positionId')::VARCHAR as positionId
from aws_test_pg_table) d ;
```

在 Oracle 中， PATH 用於識別特定 KEY和 VALUE。不過，PostgreSQL 會使用HIERARCHY模型VALUE從 JSON 讀取 KEY 和 。以下範例Oracle_SQL_Read_JSON使用 所述的相同 JSON 資料。

不允許類型為 CAST 的 SQL 查詢

（如果您強制輸入 CAST，查詢會失敗並出現語法錯誤。）

```
select *
from (
select (json_doc::json->'data'->'account'->'parentAccountNumber') as
 parentAccountNumber,
(json_doc::json->'data'->'account'->'accountNumber')as accountNumber,
(json_doc::json->'data'->'account'->'businessUnitId') as businessUnitId,
(json_doc::json->'data'->'positionId')as positionId
from aws_test_pg_table) d ;
```

使用單一大於運算子 (>) 將傳回為該 VALUE定義的 KEY。例如，KEY： positionId和 VALUE："0100"。

當您使用單一大於運算子 () 時，CAST不允許輸入 >。

允許類型為 CAST 的 SQL 查詢

```
select *
from (
select (json_doc::json->'data'->'account'->>'parentAccountNumber')::INTEGER as
 parentAccountNumber,
(json_doc::json->'data'->'account'->>'accountNumber')::INTEGER as accountNumber,
(json_doc::json->'data'->'account'->>'businessUnitId')::INTEGER as businessUnitId,
(json_doc::json->'data'->>'positionId')::varchar as positionId
from aws_test_pg_table) d ;
```

若要使用類型 CAST，您必須使用雙大於運算子。如果您使用單一大於運算子，查詢會傳回VALUE已定義的 （例如 KEY： positionId和 VALUE："0100")。使用雙大於運算子 (>>) 將傳回為該值定義的實際值 KEY（例如 KEY： positionId和 VALUE：0100，不含雙引號）。

在上述情況下， parentAccountNumber 是 CAST的類型INT， accountNumber 是 CAST的類型INT， businessUnitId 是 CAST的類型INT， positionId是 的類型， 是 CAST的類型VARCHAR。

下表顯示查詢結果，說明單一大於運算子 (>) 和雙大於運算子 () 的角色>>。

在第一個資料表中，查詢使用單一大於運算子 (>)。每個資料欄都是 JSON 類型，無法轉換為其他資料類型。

parentAccountNumber	accountNumber	businessUnitId	positionId

2003565430	2003564830	7	"0100"
2005284042	2005284042	6	"0090"
2000272719	2000272719	1	"0100"

在第二個資料表中，查詢使用大於運算子的兩倍 (>>)。每個資料欄都支援CAST以資料欄值為基礎的類型。例如，在此內容INTEGER中。

parentAccountNumber	accountNumber	businessUnitId	positionId
2003565430	2003564830	7	0100
2005284042	2005284042	6	0090
2000272719	2000272719	1	0100

5. Oracle_SQL_JSON_Aggregation_Join

查詢範例

```
SELECT
    JSON_OBJECT(
        'accountCounts' VALUE JSON_ARRAYAGG(
            JSON_OBJECT(
                'businessUnitId' VALUE business_unit_id,
                        'parentAccountNumber' VALUE parent_account_number,
                        'accountNumber' VALUE account_number,
                        'totalOnlineContactsCount' VALUE online_contacts_count,
                        'countByPosition' VALUE
                    JSON_OBJECT(
                        'taxProfessionalCount' VALUE tax_count,
                        'attorneyCount' VALUE attorney_count,
                        'nonAttorneyCount' VALUE non_attorney_count,
                        'clerkCount' VALUE clerk_count
                               ) ) ) )
FROM
    (SELECT
            tab_data.business_unit_id,
```

```
            tab_data.parent_account_number,
            tab_data.account_number,
            SUM(1) online_contacts_count,
            SUM(CASE WHEN tab_data.position_id = '0095' THEN  1 ELSE 0 END) tax_count,
            SUM(CASE    WHEN tab_data.position_id = '0100' THEN 1 ELSE 0 END)
attorney_count,
            SUM(CASE    WHEN tab_data.position_id = '0090' THEN 1 ELSE 0 END)
non_attorney_count,
            SUM(CASE    WHEN tab_data.position_id = '0050' THEN 1 ELSE 0 END)
clerk_count
       FROM aws_test_table scco,JSON_TABLE ( json_doc, '$' ERROR ON ERROR
COLUMNS (
         parent_account_number NUMBER PATH
          '$.data.account.parentAccountNumber',
           account_number NUMBER PATH '$.data.account.accountNumber',
           business_unit_id NUMBER PATH '$.data.account.businessUnitId',
           position_id VARCHAR2 ( 4 ) PATH '$.data.positionId'    )
           ) AS tab_data
           INNER JOIN JSON_TABLE ( '{
       "accounts": [{
         "accountNumber": 42000,
         "parentAccountNumber": 32000,
         "businessUnitId": 7
       }, {
         "accountNumber": 42001,
         "parentAccountNumber": 32001,
         "businessUnitId": 6
       }]
     }', '$.accounts[*]' ERROR ON ERROR
     COLUMNS (
     parent_account_number PATH '$.parentAccountNumber',
     account_number PATH '$.accountNumber',
     business_unit_id PATH '$.businessUnitId')
     ) static_data
     ON ( static_data.parent_account_number = tab_data.parent_account_number
          AND static_data.account_number = tab_data.account_number
          AND static_data.business_unit_id = tab_data.business_unit_id )
       GROUP BY
           tab_data.business_unit_id,
           tab_data.parent_account_number,
           tab_data.account_number
   );
```

若要將資料列層級資料轉換為 JSON 格式，Oracle 具有內建函數，例如 JSON_OBJECT、JSON_OBJECTAGG、 JSON_ARRAY和 JSON_ARRAYAGG。

- JSON_OBJECT 接受兩個參數： KEY和 VALUE。參數KEY本質上應為硬式編碼或靜態。VALUE 參數衍生自資料表輸出。
- JSON_ARRAYAGG 接受 JSON_OBJECT 做為參數。這有助於將一組JSON_OBJECT元素分組為清單。例如，如果您的 JSON_OBJECT 元素有多個記錄（資料集中的多個 KEY和 VALUE對），會JSON_ARRAYAGG附加資料集並建立清單。根據資料結構語言， LIST是元素群組。在此內容中，LIST是一組JSON_OBJECT元素。

下列範例顯示一個 JSON_OBJECT 元素。

```
{
   "taxProfessionalCount": 0,
   "attorneyCount": 0,
   "nonAttorneyCount": 1,
   "clerkCount": 0
}
```

下一個範例顯示兩個JSON_OBJECT元素，以方括號 () LIST表示[ ]。

```
[
    {
        "taxProfessionalCount": 0,
        "attorneyCount": 0,
        "nonAttorneyCount": 1,
        "clerkCount": 0
      }
,
    {
        "taxProfessionalCount": 2,
        "attorneyCount": 1,
        "nonAttorneyCount": 3,
        "clerkCount":4
      }
]
```

SQL 查詢範例

```
SELECT
```

```
    JSON_OBJECT(
        'accountCounts' VALUE JSON_ARRAYAGG(
            JSON_OBJECT(
                'businessUnitId' VALUE business_unit_id,
                        'parentAccountNumber' VALUE parent_account_number,
                        'accountNumber' VALUE account_number,
                        'totalOnlineContactsCount' VALUE online_contacts_count,
                        'countByPosition' VALUE
                    JSON_OBJECT(
                        'taxProfessionalCount' VALUE tax_count,
                        'attorneyCount' VALUE attorney_count,
                        'nonAttorneyCount' VALUE non_attorney_count,
                        'clerkCount' VALUE clerk_count
                               )
                        )
                                        )
              )
FROM
    (SELECT
            tab_data.business_unit_id,
            tab_data.parent_account_number,
            tab_data.account_number,
            SUM(1) online_contacts_count,
            SUM(CASE WHEN tab_data.position_id = '0095' THEN  1 ELSE   0 END
            )      tax_count,
            SUM(CASE    WHEN tab_data.position_id = '0100' THEN        1    ELSE
 0 END
            )      attorney_count,

            SUM(CASE    WHEN tab_data.position_id = '0090' THEN        1    ELSE
 0 END
            )      non_attorney_count,

            SUM(CASE    WHEN tab_data.position_id = '0050' THEN        1    ELSE
 0 END
            )      clerk_count

        FROM
            aws_test_table scco,  JSON_TABLE ( json_doc, '$' ERROR ON ERROR
            COLUMNS (
            parent_account_number NUMBER PATH '$.data.account.parentAccountNumber',
            account_number NUMBER PATH '$.data.account.accountNumber',
            business_unit_id NUMBER PATH '$.data.account.businessUnitId',
            position_id VARCHAR2 ( 4 ) PATH '$.data.positionId'    )
```

```
            ) AS tab_data
            INNER JOIN JSON_TABLE ( '{
        "accounts": [{
          "accountNumber": 42000,
          "parentAccountNumber": 32000,
          "businessUnitId": 7
        }, {
          "accountNumber": 42001,
          "parentAccountNumber": 32001,
          "businessUnitId": 6
        }]
      }', '$.accounts[*]' ERROR ON ERROR
      COLUMNS (
      parent_account_number PATH '$.parentAccountNumber',
      account_number PATH '$.accountNumber',
      business_unit_id PATH '$.businessUnitId')
      ) static_data ON ( static_data.parent_account_number =
tab_data.parent_account_number
                         AND static_data.account_number = tab_data.account_number

                         AND static_data.business_unit_id =
tab_data.business_unit_id )
        GROUP BY
            tab_data.business_unit_id,
            tab_data.parent_account_number,
            tab_data.account_number
    );
```

上一個 SQL 查詢的範例輸出

```
{
  "accountCounts": [
    {
      "businessUnitId": 6,
      "parentAccountNumber": 32001,
      "accountNumber": 42001,
      "totalOnlineContactsCount": 1,
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 0,
        "nonAttorneyCount": 1,
        "clerkCount": 0
      }
```

```
    },
    {
      "businessUnitId": 7,
      "parentAccountNumber": 32000,
      "accountNumber": 42000,
      "totalOnlineContactsCount": 1,
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 1,
        "nonAttorneyCount": 0,
        "clerkCount": 0
      }
    }
  ]
}
```

6. Postgres_SQL_JSON_Aggregation_Join

PostgreSQL 內建函數，JSON_BUILD_OBJECT並將 ROW 層級資料JSON_AGG轉換為 JSON 格式。 PostgreSQL JSON_BUILD_OBJECT和 JSON_AGG 相當於 Oracle JSON_OBJECT和 JSON_ARRAYAGG。

查詢範例

```
select
JSON_BUILD_OBJECT ('accountCounts',
    JSON_AGG(
        JSON_BUILD_OBJECT ('businessUnitId',businessUnitId
        ,'parentAccountNumber',parentAccountNumber
        ,'accountNumber',accountNumber
        ,'totalOnlineContactsCount',online_contacts_count,
        'countByPosition',
            JSON_BUILD_OBJECT (
            'taxProfessionalCount',tax_professional_count
            ,'attorneyCount',attorney_count
            ,'nonAttorneyCount',non_attorney_count
            ,'clerkCount',clerk_count
            )
        )
    )
)
from (
with tab as (select * from (
```

```
select (json_doc::json->'data'->'account'->>'parentAccountNumber')::INTEGER as
 parentAccountNumber,
(json_doc::json->'data'->'account'->>'accountNumber')::INTEGER as accountNumber,
(json_doc::json->'data'->'account'->>'businessUnitId')::INTEGER as businessUnitId,
(json_doc::json->'data'->>'positionId')::varchar as positionId
from aws_test_pg_table) a ) ,
tab1 as ( select
(json_array_elements(b.jc -> 'accounts') ->> 'accountNumber')::integer accountNumber,
(json_array_elements(b.jc -> 'accounts') ->> 'businessUnitId')::integer
 businessUnitId,
(json_array_elements(b.jc -> 'accounts') ->> 'parentAccountNumber')::integer
 parentAccountNumber
from (
select '{
        "accounts": [{
          "accountNumber": 42001,
          "parentAccountNumber": 32001,
          "businessUnitId": 6
        }, {
          "accountNumber": 42000,
          "parentAccountNumber": 32000,
          "businessUnitId": 7
        }]
      }'::json as jc) b)
select
tab.businessUnitId::text,
tab.parentAccountNumber::text,
tab.accountNumber::text,
SUM(1) online_contacts_count,
SUM(CASE WHEN tab.positionId::text = '0095' THEN 1 ELSE 0  END)
 tax_professional_count,
SUM(CASE WHEN tab.positionId::text = '0100' THEN 1 ELSE 0  END)      attorney_count,
SUM(CASE  WHEN tab.positionId::text = '0090' THEN      1  ELSE      0 END)
 non_attorney_count,
SUM(CASE  WHEN tab.positionId::text = '0050' THEN      1  ELSE      0 END)
 clerk_count
from tab1,tab
where tab.parentAccountNumber::INTEGER=tab1.parentAccountNumber::INTEGER
and tab.accountNumber::INTEGER=tab1.accountNumber::INTEGER
and tab.businessUnitId::INTEGER=tab1.businessUnitId::INTEGER
GROUP BY      tab.businessUnitId::text,
            tab.parentAccountNumber::text,
            tab.accountNumber::text) a;
```

上述查詢的範例輸出

Oracle 和 PostgreSQL 的輸出完全相同。

```
{
  "accountCounts": [
    {
      "businessUnitId": 6,
      "parentAccountNumber": 32001,
      "accountNumber": 42001,
      "totalOnlineContactsCount": 1,
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 0,
        "nonAttorneyCount": 1,
        "clerkCount": 0
      }
    },
    {
      "businessUnitId": 7,
      "parentAccountNumber": 32000,
      "accountNumber": 42000,
      "totalOnlineContactsCount": 1,
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 1,
        "nonAttorneyCount": 0,
        "clerkCount": 0
      }
    }
  ]
}
```

7.Oracle_procedure_with_JSON_Query

此程式碼會將 Oracle 程序轉換為具有 JSON SQL 查詢的 PostgreSQL 函數。它顯示查詢如何將 JSON 轉換為資料列和反向。

```
CREATE OR REPLACE PROCEDURE p_json_test(p_in_accounts_json IN varchar2,
 p_out_accunts_json  OUT varchar2)
IS
BEGIN
/*
```

```
p_in_accounts_json paramter should have following format:
       {
        "accounts": [{
          "accountNumber": 42000,
          "parentAccountNumber": 32000,
          "businessUnitId": 7
        }, {
          "accountNumber": 42001,
          "parentAccountNumber": 32001,
          "businessUnitId": 6
        }]
      }
*/
SELECT
    JSON_OBJECT(
        'accountCounts' VALUE JSON_ARRAYAGG(
            JSON_OBJECT(
                'businessUnitId' VALUE business_unit_id,
                        'parentAccountNumber' VALUE parent_account_number,
                        'accountNumber' VALUE account_number,
                        'totalOnlineContactsCount' VALUE online_contacts_count,
                        'countByPosition' VALUE
                    JSON_OBJECT(
                        'taxProfessionalCount' VALUE tax_count,
                        'attorneyCount' VALUE attorney_count,
                        'nonAttorneyCount' VALUE non_attorney_count,
                        'clerkCount' VALUE clerk_count
                               ) ) ) )
into p_out_accunts_json
FROM
    (SELECT
            tab_data.business_unit_id,
            tab_data.parent_account_number,
            tab_data.account_number,
            SUM(1) online_contacts_count,
            SUM(CASE WHEN tab_data.position_id = '0095' THEN  1 ELSE 0 END) tax_count,
            SUM(CASE    WHEN tab_data.position_id = '0100' THEN 1 ELSE 0 END)
 attorney_count,
            SUM(CASE    WHEN tab_data.position_id = '0090' THEN 1 ELSE 0 END)
 non_attorney_count,
            SUM(CASE    WHEN tab_data.position_id = '0050' THEN 1 ELSE 0 END)
 clerk_count
       FROM aws_test_table scco,JSON_TABLE ( json_doc, '$' ERROR ON ERROR
            COLUMNS (
```

```
            parent_account_number NUMBER PATH '$.data.account.parentAccountNumber',
            account_number NUMBER PATH '$.data.account.accountNumber',
            business_unit_id NUMBER PATH '$.data.account.businessUnitId',
            position_id VARCHAR2 ( 4 ) PATH '$.data.positionId'    )
            ) AS tab_data
            INNER JOIN JSON_TABLE ( p_in_accounts_json, '$.accounts[*]' ERROR ON ERROR

      COLUMNS (
      parent_account_number PATH '$.parentAccountNumber',
      account_number PATH '$.accountNumber',
      business_unit_id PATH '$.businessUnitId')
      ) static_data
      ON ( static_data.parent_account_number = tab_data.parent_account_number
           AND static_data.account_number = tab_data.account_number
           AND static_data.business_unit_id = tab_data.business_unit_id )
        GROUP BY
            tab_data.business_unit_id,
            tab_data.parent_account_number,
            tab_data.account_number
    );
EXCEPTION
WHEN OTHERS THEN
   raise_application_error(-20001,'Error while running the JSON query');
END;
/
```

執行程序

下列程式碼區塊說明如何使用程序的範例 JSON 輸入來執行先前建立的 Oracle 程序。它也會提供您此程序的結果或輸出。

```
set serveroutput on;
declare
v_out varchar2(30000);
v_in varchar2(30000):= '{
        "accounts": [{
          "accountNumber": 42000,
          "parentAccountNumber": 32000,
          "businessUnitId": 7
        }, {
          "accountNumber": 42001,
          "parentAccountNumber": 32001,
          "businessUnitId": 6
```

```
        }]
      }';
begin
  p_json_test(v_in,v_out);
  dbms_output.put_line(v_out);
end;
/
```

程序輸出

```
{
  "accountCounts": [
    {
      "businessUnitId": 6,
      "parentAccountNumber": 32001,
      "accountNumber": 42001,
      "totalOnlineContactsCount": 1,
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 0,
        "nonAttorneyCount": 1,
        "clerkCount": 0
      }
    },
    {
      "businessUnitId": 7,
      "parentAccountNumber": 32000,
      "accountNumber": 42000,
      "totalOnlineContactsCount": 1,
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 1,
        "nonAttorneyCount": 0,
        "clerkCount": 0
      }
    }
  ]
}
```

8.Postgres_function_with_JSON_Query

範例函數

```
CREATE OR REPLACE  FUNCTION f_pg_json_test(p_in_accounts_json  text)
RETURNS text
LANGUAGE plpgsql
AS
$$
DECLARE
 v_out_accunts_json   text;
BEGIN
SELECT
JSON_BUILD_OBJECT ('accountCounts',
    JSON_AGG(
        JSON_BUILD_OBJECT ('businessUnitId',businessUnitId
        ,'parentAccountNumber',parentAccountNumber
        ,'accountNumber',accountNumber
        ,'totalOnlineContactsCount',online_contacts_count,
        'countByPosition',
            JSON_BUILD_OBJECT (
            'taxProfessionalCount',tax_professional_count
            ,'attorneyCount',attorney_count
            ,'nonAttorneyCount',non_attorney_count
            ,'clerkCount',clerk_count
            ))))
INTO v_out_accunts_json
FROM (
WITH tab AS (SELECT * FROM (
SELECT (json_doc::json->'data'->'account'->>'parentAccountNumber')::INTEGER AS
 parentAccountNumber,
(json_doc::json->'data'->'account'->>'accountNumber')::INTEGER AS accountNumber,
(json_doc::json->'data'->'account'->>'businessUnitId')::INTEGER AS businessUnitId,
(json_doc::json->'data'->>'positionId')::varchar AS positionId
FROM aws_test_pg_table) a ) ,
tab1 AS ( SELECT
(json_array_elements(b.jc -> 'accounts') ->> 'accountNumber')::integer accountNumber,
(json_array_elements(b.jc -> 'accounts') ->> 'businessUnitId')::integer businessUnitId,
(json_array_elements(b.jc -> 'accounts') ->> 'parentAccountNumber')::integer
 parentAccountNumber
FROM (
SELECT p_in_accounts_json::json AS jc) b)
SELECT
tab.businessUnitId::text,
tab.parentAccountNumber::text,
tab.accountNumber::text,
SUM(1) online_contacts_count,
```

```
SUM(CASE WHEN tab.positionId::text = '0095' THEN 1 ELSE 0  END)
 tax_professional_count,
SUM(CASE WHEN tab.positionId::text = '0100' THEN 1 ELSE 0  END)      attorney_count,
SUM(CASE  WHEN tab.positionId::text = '0090' THEN      1  ELSE      0 END)
 non_attorney_count,
SUM(CASE  WHEN tab.positionId::text = '0050' THEN      1  ELSE      0 END)
 clerk_count
FROM tab1,tab
WHERE tab.parentAccountNumber::INTEGER=tab1.parentAccountNumber::INTEGER
AND tab.accountNumber::INTEGER=tab1.accountNumber::INTEGER
AND tab.businessUnitId::INTEGER=tab1.businessUnitId::INTEGER
GROUP BY      tab.businessUnitId::text,
            tab.parentAccountNumber::text,
            tab.accountNumber::text) a;
RETURN v_out_accunts_json;
END;
$$;
```

執行 函數

```
select    f_pg_json_test('{
        "accounts": [{
          "accountNumber": 42001,
          "parentAccountNumber": 32001,
          "businessUnitId": 6
        }, {
          "accountNumber": 42000,
          "parentAccountNumber": 32000,
          "businessUnitId": 7
        }]
      }')   ;
```

函數輸出

下列輸出類似於 Oracle 程序輸出。差別在於此輸出為文字格式。

```
{
  "accountCounts": [
    {
      "businessUnitId": "6",
      "parentAccountNumber": "32001",
      "accountNumber": "42001",
      "totalOnlineContactsCount": 1,
```

```
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 0,
        "nonAttorneyCount": 1,
        "clerkCount": 0
      }
    },
    {
      "businessUnitId": "7",
      "parentAccountNumber": "32000",
      "accountNumber": "42000",
      "totalOnlineContactsCount": 1,
      "countByPosition": {
        "taxProfessionalCount": 0,
        "attorneyCount": 1,
        "nonAttorneyCount": 0,
        "clerkCount": 0
      }
    }
  ]
}
```

使用 跨帳戶複製 Amazon DynamoDB 資料表 AWS Backup

Ramkumar Ramanujam，Amazon Web Services

總結

在 上使用 Amazon DynamoDB 時 AWS，常見的使用案例是將開發、測試或預備環境中的 DynamoDB 資料表與生產環境中的資料表資料進行複製或同步。根據標準實務，每個環境都使用不同的 AWS 帳戶。

AWS Backup 支援跨區域和跨帳戶備份和還原 DynamoDB、Amazon Simple Storage Service (Amazon S3) 和其他 的資料 AWS 服務。此模式提供使用 AWS Backup 跨帳戶備份和還原來複製 DynamoDB 資料表的步驟 AWS 帳戶。

先決條件和限制

先決條件

- 在 中 AWS 帳戶 屬於相同組織的兩個作用中 AWS Organizations
- 在兩個帳戶中建立 DynamoDB 資料表的許可
- AWS Identity and Access Management 建立和使用保存庫的 (IAM) AWS Backup 許可

限制

- 來源和目標 AWS 帳戶 應該是 中相同組織的一部分 AWS Organizations。

Architecture

目標技術堆疊

- AWS Backup
- Amazon DynamoDB

目標架構

1. 在來源帳戶中的備份文件庫中建立 DynamoDB 資料表 AWS Backup 備份。

2. 將備份複製到目標帳戶中的備份保存庫。
3. 使用目標帳戶中備份文件庫的備份來還原目標帳戶中的 DynamoDB 資料表。

自動化和擴展

您可以使用 AWS Backup 來排程以特定間隔執行備份。

工具

- AWS Backup 是一種全受管服務，可集中和自動化跨雲端、雲端和內部部署 AWS 服務的資料保護。使用此服務，您可以在 AWS 單一位置設定資源的備份政策並監控活動。它可讓您自動化和合併先前由 服務執行的備份任務，並不需要建立自訂指令碼和手動程序。
- Amazon DynamoDB 是全受管的 NoSQL 資料庫服務，可提供快速且可預測的效能和無縫的可擴展性。

史詩

開啟來源和目標帳戶中 AWS Backup 的功能

任務	Description	所需的技能
開啟 DynamoDB 和跨帳戶備份的進階功能。	在來源和目標中 AWS 帳戶，執行下列動作： 1. 在 上 AWS 管理主控台，開啟 AWS Backup 主控台。 2. 選擇設定。 3. 在 Amazon DynamoDB 備份的進階功能下，確認已啟用進階功能，或選擇啟用。 4. 在跨帳戶管理下，針對跨帳戶備份，選擇啟用。	AWS DevOps，遷移工程師

在來源和目標帳戶中建立備份保存庫

任務	Description	所需的技能
建立備份保存庫。	在來源和目標中 AWS 帳戶，執行下列動作： 1. 在AWS Backup 主控台上，選擇備份保存庫。 2. 選擇 Create backup vault (建立備份文件庫)。 3. 複製備份保存庫的 Amazon Resource Name (ARN) 並將其儲存。 當您在來源和目標帳戶之間複製 DynamoDB 資料表備份時，將需要來源和目標備份文件庫的 ARNs。	AWS DevOps，遷移工程師

使用備份保存庫執行備份和還原

任務	Description	所需的技能
在來源帳戶中，建立 DynamoDB 資料表備份。	若要為來源帳戶中的 DynamoDB 資料表建立備份，請執行下列動作： 1. 在 AWS Backup 儀表板頁面上，選擇建立隨需備份。 2. 在設定區段中，針對資源類型，選取 DynamoDB，然後選取資料表名稱。 3. 在備份保存庫下拉式清單中，選取您在來源帳戶中建立的備份保存庫。	AWS DevOps、DBA、遷移工程師

任務	Description	所需的技能
	4. 選取您想要的保留期間。 5. 選擇 Create on-demand backup (建立隨需備份)。 建立新的備份任務。 若要監控備份任務的狀態，請在 AWS Backup 任務頁面上，選擇備份任務索引標籤。所有作用中、進行中和已完成的備份任務都會列在此索引標籤上。	

任務	Description	所需的技能
將備份從來源帳戶複製到目標帳戶。	備份任務完成後，將 DynamoDB 資料表備份從來源帳戶中的備份保存庫複製到目標帳戶中的備份保存庫。 若要複製備份保存庫，請在來源帳戶中執行下列動作： 1. 在 AWS Backup 主控台上，選擇備份保存庫。 2. 在備份項目下，選擇 DynamoDB 資料表備份。 3. 選擇 Actions (動作)、Copy (複製)。 4. 輸入目標帳戶的 AWS 區域 。 5. 對於外部保存庫 ARN，輸入您在目標帳戶中建立的備份保存庫 ARN。 6. 若要將備份從來源帳戶複製到目標帳戶，請在目標帳戶備份文件庫中啟用來自不同帳戶的存取。	AWS DevOps，遷移工程師，DBA
還原目標帳戶中的備份。	在目標中 AWS 帳戶，執行下列動作： 1. 在 AWS Backup 主控台上，選擇備份保存庫。 2. 在備份下，選取您從來源帳戶複製的備份。 3. 選擇動作、還原。 4. 輸入您要還原的目標 DynamoDB 資料表名稱。	AWS DevOps、DBA、遷移工程師

相關資源

- AWS Backup 搭配 DynamoDB 使用
- 跨 建立備份複本 AWS 帳戶
- AWS Backup 定價

使用自訂實作跨帳戶複製 Amazon DynamoDB 資料表

Ramkumar Ramanujam，Amazon Web Services

總結

在 Amazon Web Services (AWS) 上使用 Amazon DynamoDB 時，常見的使用案例是將開發、測試或預備環境中的 DynamoDB 資料表與生產環境中的資料表資料進行複製或同步。根據標準實務，每個環境都使用不同的 AWS 帳戶。

DynamoDB 現在支援使用 AWS Backup 的跨帳戶備份。如需使用 AWS Backup 時相關儲存成本的資訊，請參閱 AWS Backup 定價。當您使用 AWS Backup 跨帳戶複製時，來源和目標帳戶必須是 AWS Organizations 組織的一部分。還有其他使用 AWS Glue 等 AWS 服務進行跨帳戶備份和還原的解決方案。不過，使用這些解決方案會增加應用程式足跡，因為要部署和維護的 AWS 服務更多。

您也可以使用 Amazon DynamoDB Streams 擷取來源帳戶中的資料表變更。然後，您可以啟動 AWS Lambda 函數，並在目標帳戶中的目標資料表中進行對應的變更。但是，該解決方案適用於來源和目標資料表必須保持同步的使用案例。它可能不適用於頻繁更新資料的開發、測試和預備環境。

此模式提供實作自訂解決方案的步驟，將 Amazon DynamoDB 資料表從一個帳戶複製到另一個帳戶。此模式可以使用常見的程式設計語言實作，例如 C#、Java 和 Python。建議使用 AWS 開發套件支援的語言。

先決條件和限制

先決條件

- 兩個作用中的 AWS 帳戶
- 兩個帳戶中的 DynamoDB 資料表
- 了解 AWS Identity and Access Management (IAM) 角色和政策
- 了解如何使用任何常見的程式設計語言存取 Amazon DynamoDB 資料表，例如 C#、Java 或 Python

限制

此模式適用於大約 2 GB 或更小的 DynamoDB 資料表。透過額外的邏輯來處理連線或工作階段中斷、限流、故障和重試，它可用於較大的資料表。

DynamoDB 掃描操作會從來源資料表讀取項目，在單一呼叫中最多只能擷取 1 MB 的資料。對於大於 2 GB 的較大資料表，此限制可能會增加執行完整資料表複本的總時間。

Architecture

下圖顯示來源和目標 AWS 帳戶之間的自訂實作。IAM 政策和安全字符會與自訂實作搭配使用。從來源帳戶中的 Amazon DynamoDB 讀取資料，並寫入目標帳戶中的 DynamoDB。

自動化和擴展

此模式適用於大小較小的 DynamoDB 資料表，約為 2 GB。

若要將此模式套用至較大的資料表，請解決下列問題：

- 在資料表複製操作期間，會使用不同的安全字符來維護兩個作用中工作階段。如果資料表複製操作花費的時間超過字符過期時間，您必須設定邏輯來重新整理安全字符。
- 如果未佈建足夠的讀取容量單位 RCUs) 和寫入容量單位 WCUs)，則來源或目標資料表上的讀取或寫入可能會受到調節。請務必擷取並處理這些例外狀況。
- 處理任何其他失敗或例外狀況，並設定重試機制，以便在複製操作失敗時重試或繼續。

工具

工具

- Amazon DynamoDB – Amazon DynamoDB 是全受管的 NoSQL 資料庫服務，可提供快速且可預測的效能和無縫的可擴展性。
- 所需的其他工具會根據您為實作選擇的程式設計語言而有所不同。例如，如果您使用 C#，您將需要 Microsoft Visual Studio 和下列 NuGet 套件：
  - `AWSSDK`
  - `AWSSDK.DynamoDBv2`

Code

下列 Python 程式碼片段會使用 Boto3 程式庫刪除並重新建立 DynamoDB 資料表。

請勿使用 `AWS_SECRET_ACCESS_KEY` IAM 使用者的 `AWS_ACCESS_KEY_ID`和 ，因為這些是長期登入資料，因此應該避免以程式設計方式存取 AWS 服務。如需暫時登入資料的詳細資訊，請參閱最佳實務一節。

下列程式碼片段TEMPORARY_SESSION_TOKEN中使用的AWS_ACCESS_KEY_IDAWS_SECRET_ACCESS_KEY、 和 是從 AWS Security Token Service (AWS STS) 擷取的暫時登入資料。

```
import boto3
import sys
import json

#args = input-parameters = GLOBAL_SEC_INDEXES_JSON_COLLECTION,
 ATTRIBUTES_JSON_COLLECTION, TARGET_DYNAMODB_NAME, TARGET_REGION, ...

#Input param: GLOBAL_SEC_INDEXES_JSON_COLLECTION
#[{"IndexName":"Test-index","KeySchema":[{"AttributeName":"AppId","KeyType":"HASH"},
{"AttributeName":"AppType","KeyType":"RANGE"}],"Projection":
{"ProjectionType":"INCLUDE","NonKeyAttributes":["PK","SK","OwnerName","AppVersion"]}}]

#Input param: ATTRIBUTES_JSON_COLLECTION
#[{"AttributeName":"PK","AttributeType":"S"},
{"AttributeName":"SK","AttributeType":"S"},
{"AttributeName":"AppId","AttributeType":"S"},
{"AttributeName":"AppType","AttributeType":"N"}]

region = args['TARGET_REGION']
target_ddb_name = args['TARGET_DYNAMODB_NAME']

global_secondary_indexes = json.loads(args['GLOBAL_SEC_INDEXES_JSON_COLLECTION'])
attribute_definitions = json.loads(args['ATTRIBUTES_JSON_COLLECTION'])

# Drop and create target DynamoDB table
dynamodb_client = boto3.Session(
        aws_access_key_id=args['AWS_ACCESS_KEY_ID'],
        aws_secret_access_key=args['AWS_SECRET_ACCESS_KEY'],
        aws_session_token=args['TEMPORARY_SESSION_TOKEN'],
    ).client('dynamodb')

# Delete table
print('Deleting table: ' + target_ddb_name + ' ...')

try:
    dynamodb_client.delete_table(TableName=target_ddb_name)

    #Wait for table deletion to complete
    waiter = dynamodb_client.get_waiter('table_not_exists')
```

```
    waiter.wait(TableName=target_ddb_name)
    print('Table deleted.')
except dynamodb_client.exceptions.ResourceNotFoundException:
    print('Table already deleted / does not exist.')
    pass

print('Creating table: ' + target_ddb_name + ' ...')

table = dynamodb_client.create_table(
    TableName=target_ddb_name,
    KeySchema=[
        {
            'AttributeName': 'PK',
            'KeyType': 'HASH'  # Partition key
        },
        {
            'AttributeName': 'SK',
            'KeyType': 'RANGE'  # Sort key
        }
    ],
    AttributeDefinitions=attribute_definitions,
    GlobalSecondaryIndexes=global_secondary_indexes,
    BillingMode='PAY_PER_REQUEST'
)

waiter = dynamodb_client.get_waiter('table_exists')
waiter.wait(TableName=target_ddb_name)

print('Table created.')
```

最佳實務

臨時憑證

作為安全最佳實務，以程式設計方式存取 AWS 服務時，請避免使用 AWS_SECRET_ACCESS_KEY IAM 使用者的 AWS_ACCESS_KEY_ID和 ，因為這些是長期憑證。一律嘗試使用臨時登入資料，以程式設計方式存取 AWS 服務。

例如，開發人員會在開發期間硬式編碼應用程式中 AWS_SECRET_ACCESS_KEY IAM 使用者的 AWS_ACCESS_KEY_ID和 ，但無法在將變更推送至程式碼儲存庫之前移除硬式編碼值。這些公開的登入資料可供非預期或惡意使用者使用，這可能會產生嚴重影響 （特別是當公開的登入資料具有管理員權限時）。這些公開的登入資料應該使用 IAM 主控台或 AWS Command Line Interface (AWS CLI) 立即停用或刪除。

若要取得以程式設計方式存取 AWS 服務的臨時登入資料，請使用 AWS STS。暫時登入資料僅在指定的時間內有效（從 15 分鐘到 36 小時）。臨時登入資料的允許持續時間上限會根據角色設定和角色鏈結等因素而有所不同。如需 AWS STS 的詳細資訊，請參閱 文件。

史詩

設定 DynamoDB 資料表

任務	Description	所需的技能
建立 DynamoDB 資料表。	在來源和目標 AWS 帳戶中建立具有索引的 DynamoDB 資料表。 將容量佈建設定為隨需模式，這可讓 DynamoDB 根據工作負載動態擴展讀取/寫入容量。 或者，您可以使用佈建容量搭配 4000 RCUs 和 4000 WCUs。	應用程式開發人員、DBA、遷移工程師
填入來源資料表。	使用測試資料填入來源帳戶中的 DynamoDB 資料表。擁有至少 50 MB 或更多的測試資料可協助您查看資料表複製期間消耗的峰值和平均 RCUs。然後，您可以視需要變更容量佈建。	應用程式開發人員、DBA、遷移工程師

設定登入資料以存取 DynamoDB 資料表

任務	Description	所需的技能
建立 IAM 角色以存取來源和目標 DynamoDB 資料表。	在來源帳戶中建立具有存取（讀取）來源帳戶中 DynamoDB 資料表許可的 IAM 角色。	應用程式開發人員、AWS DevOps

任務	Description	所需的技能
	新增來源帳戶做為此角色的信任實體。 在目標帳戶中建立具有存取（建立、讀取、更新、刪除）目標帳戶中 DynamoDB 資料表許可的 IAM 角色。 新增目標帳戶做為此角色的信任實體。	

將資料表資料從一個帳戶複製到另一個帳戶

任務	Description	所需的技能
取得 IAM 角色的臨時登入資料。	取得來源帳戶中建立之 IAM 角色的臨時登入資料。 取得在目標帳戶中建立之 IAM 角色的臨時登入資料。 取得 IAM 角色臨時登入資料的一種方法是從 AWS CLI 使用 AWS STS。 `aws sts assume-role --role-arn arn:aws:iam::<account-id>:role/<role-name> --role-session-name <session-name> --profile <profile-name>` 使用適當的 AWS 設定檔（對應至來源或目標帳戶）。	應用程式開發人員、遷移工程師

任務	Description	所需的技能
	如需取得臨時登入資料之不同方式的詳細資訊，請參閱下列內容： • AWS Security Token Service API 參考 • 取得 CLI 存取的 IAM 角色登入資料	
初始化來源和目標 DynamoDB 存取的 DynamoDB 用戶端。	針對來源和目標 DynamoDB 資料表初始化 AWS 開發套件提供的 DynamoDB 用戶端。 • 對於來源 DynamoDB 用戶端，請使用從來源帳戶擷取的暫時憑證。 • 對於目標 DynamoDB 用戶端，請使用從目標帳戶擷取的暫時登入資料。 如需使用 IAM 臨時登入資料提出請求的詳細資訊，請參閱 AWS 文件。	應用程式開發人員

任務	Description	所需的技能
捨棄並重新建立目標資料表。	使用目標帳戶 DynamoDB 用戶端，在目標帳戶中刪除並重新建立目標 DynamoDB 資料表（以及索引）。 從 DynamoDB 資料表刪除所有記錄是一項昂貴的操作，因為它會使用佈建WCUs。刪除和重新建立資料表可避免這些額外費用。 您可以在建立索引後將索引新增至資料表，但這需要 2-5 分鐘的時間。透過將索引集合傳遞至`createTable` 呼叫，在資料表建立期間建立索引更有效率。	應用程式開發人員

任務	Description	所需的技能
執行資料表複製。	重複下列步驟，直到複製所有資料： • 使用來源 DynamoDB 用戶端，對來源帳戶中的資料表執行掃描。每個 DynamoDB 掃描只會從資料表擷取 1 MB 的資料，因此您必須重複此操作，直到讀取所有項目或記錄為止。 • 對於每組掃描的項目，使用適用於 DynamoDB 的 AWS 開發套件中的 `BatchWriteItem` 呼叫，使用目標 DynamoDB 用戶端將項目寫入目標帳戶中的資料表。這可減少對 DynamoDB 提出的`PutItem`請求數量。 • `BatchWriteItem` 具有 25 個寫入或放置的限制，或高達 16 MB。呼叫 之前，您必須新增邏輯來累積計數為 25 的掃描項目`BatchWriteItem` 。 會`BatchWriteItem` 傳回無法成功複製的項目清單。使用此清單，新增重試邏輯，以僅對未成功的項目執行另一個`BatchWriteItem` 呼叫。 如需詳細資訊，請參閱附件區段中的 C# 中的參考實作（適用於捨棄、建立和填入資料	應用程式開發人員

任務	Description	所需的技能
	表）。也會連接範例資料表組態 JavaScript 物件標記法 (JSON) 檔案。	

相關資源

- Amazon DynamoDB 文件
- 在 AWS 帳戶中建立 IAM 使用者
- AWS 開發套件
- 搭配 AWS 資源使用臨時登入資料

其他資訊

此模式使用 C# 實作，以複製具有 200，000 個項目的 DynamoDB 資料表（平均項目大小為 5 KB，資料表大小為 250 MB)。目標 DynamoDB 資料表的設定佈建容量為 4000 RCUs 和 4000 WCUs。

完整的資料表複製操作（從來源帳戶到目標帳戶），包括捨棄和重新建立資料表，需要 5 分鐘。使用的總容量單位：30，000 RCUs 和大約 400，000 WCUs。

如需 DynamoDB 容量模式的詳細資訊，請參閱 AWS 文件中的讀取/寫入容量模式。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

建立 Amazon RDS 和 Amazon Aurora 的詳細成本和用量報告

Lakshmanan Lakshmanan 和 Sudarshan Narasimhan，Amazon Web Services

總結

此模式說明如何透過設定使用者定義的成本分配標籤來追蹤 Amazon Relational Database Service (Amazon RDS) 或 Amazon Aurora 叢集的使用成本。 https://docs.aws.amazon.com/awsaccountbilling/latest/aboutv2/custom-tags.html您可以使用這些標籤，在 AWS Cost Explorer 中為多個維度的叢集建立詳細的成本和用量報告。例如，您可以在團隊、專案或成本中心層級追蹤用量成本，然後在 Amazon Athena 中分析資料。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 一或多個 Amazon RDS 或 Amazon Aurora 執行個體

限制

如需標記限制，請參閱 AWS Billing 使用者指南。

Architecture

目標技術堆疊

- Amazon RDS 或 Amazon Aurora
- AWS 成本和用量報告
- AWS Cost Explorer
- Amazon Athena

工作流程和架構

標記和分析工作流程包含下列步驟：

1. 資料工程師、資料庫管理員或 AWS 管理員會為 Amazon RDS 或 Aurora 叢集建立使用者定義的成本分配標籤。

2. AWS 管理員會啟用標籤。
3. 這些標籤會向 AWS Cost Explorer 報告中繼資料。
4. 資料工程師、資料庫管理員或 AWS 管理員會建立每月成本分配報告。
5. 資料工程師、資料庫管理員或 AWS 管理員會使用 Amazon Athena 分析每月成本分配報告。

下圖顯示如何套用標籤來追蹤 Amazon RDS 或 Aurora 執行個體的使用成本。

下列架構圖顯示成本分配報告如何與 Amazon Athena 整合以進行分析。

每月成本分配報告會存放在您指定的 Amazon S3 儲存貯體中。當您使用 AWS CloudFormation 範本設定 Athena 時，如 Epics 一節所述，範本會為 Lambda 函數佈建數個額外的資源，包括 AWS Glue 爬蟲程式、AWS Glue 資料庫、Amazon Simple Notification System (Amazon SNS) 事件、AWS Lambda 函數，以及 AWS Identity and Access Management (IAM) 角色。當新的成本資料檔案送達 S3 儲存貯體時，事件通知會用來將這些檔案轉送至 Lambda 函數進行處理。Lambda 函數會啟動 AWS Glue 爬蟲程式任務，以在 AWS Glue Data Catalog 中建立或更新資料表。然後，此資料表用於查詢 Athena 中的資料。

工具

- Amazon Athena 是一種互動式查詢服務，可讓您使用標準 SQL 輕鬆分析 Amazon S3 中的資料。
- Amazon Aurora 是一種全受管關聯式資料庫引擎，專為雲端而建置，並與 MySQL 和 PostgreSQL 相容。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。
- AWS CloudFormation 是一種基礎設施即程式碼 (IaC) 服務，可讓您輕鬆地建立、佈建和管理 AWS 和第三方資源。
- AWS Cost Explorer 可協助您檢視和分析 AWS 成本和用量。

史詩

為您的 Amazon RDS 或 Aurora 叢集建立和啟用標籤

任務	Description	所需的技能
為您的 Amazon RDS 或 Aurora 叢集建立使用者定義的成本分配標籤。	若要將標籤新增至新的或現有的 Amazon RDS 或 Aurora 叢集，請遵循 Amazon Aurora 使用者指南中的新增、列出和移除標籤中的指示。 **Note** 如需有關如何設定 Amazon Aurora 叢集的資訊，請參閱《Amazon Aurora 使用者指南》中的 MySQL 和 PostgreSQL 的說明。	AWS 管理員、資料工程師、DBA
啟用使用者定義的成本分配標籤。	遵循 AWS Billing 使用者指南中啟用使用者定義的成本分配標籤中的指示。	AWS 管理員

建立成本和用量報告

任務	Description	所需的技能
為您的叢集建立和設定成本和用量報告。	1. 登入 AWS 管理主控台並開啟 AWS Billing 主控台。 2. 在左側導覽窗格中，選擇成本與用量報告。 3. 選擇 Create report (建立報告)。	應用程式擁有者、AWS 管理員、DBA、一般 AWS、資料工程師

任務	Description	所需的技能
	4. 提供報告名稱，保留其他選項的預設設定，然後選擇下一步。 5. 選擇設定並提供現有 S3 儲存貯體的詳細資訊。您也可以從此畫面選擇建立新的 S3 儲存貯體。選擇下一步。 6. 驗證要套用至儲存貯體的預設政策，選取確認核取方塊，然後選擇儲存。 7. 對於報告路徑字首，指定您要在報告名稱前面加上的字首。 8. 對於時間精細程度，根據您希望為報告收集資料的頻率，選擇每小時、每日或每月。 9. 對於報告版本控制，選擇您希望報告的新版本單獨建立，還是覆寫每個版本的現有報告。 10針對啟用報告資料整合，選擇 Amazon Athena。確認壓縮類型設定為 Parquet。 11選擇下一步。 12檢閱報告設定，然後選擇檢閱並完成。 資料將在 24 小時內提供。	

分析成本和用量報告資料

任務	Description	所需的技能
分析成本和用量報告資料。	1. 設定並使用 Athena 來分析報告資料。如需說明，請參閱《AWS 成本和用量報告使用者指南》中的使用 Amazon Athena 查詢成本和用量報告。 我們建議您使用 Athena 提供的 AWS CloudFormation 範本。 2. 執行 Athena 查詢。例如，您可以使用下列 SQL 查詢來檢查資料重新整理的狀態。 `select status from cost_and_usage_data_status` 如需詳細資訊，請參閱《AWS 成本和用量報告使用者指南》中的執行 Amazon Athena 查詢。 **Note** 當您執行 SQL 查詢時，請確定已從下拉式清單中選取正確的資料庫。	應用程式擁有者、AWS 管理員、DBA、一般 AWS、資料工程師

相關資源

參考

- 使用 AWS CloudFormation 範本設定 Athena （建議）
- 手動設定 Athena
- 執行 Amazon Athena 查詢
- 將報告資料載入其他資源

教學課程和影片

- 使用 Amazon Athena 分析成本和用量報告 (YouTube 影片）

使用 Terraform 在 Amazon EC2 和 Amazon FSx 上部署 SQL Server 容錯移轉叢集執行個體

Mark Hudson 和 Matt Burgess，Amazon Web Services

總結

此模式使用 Terraform 在 Amazon Elastic Compute Cloud (Amazon EC2) 上的 Windows Server 容錯移轉叢集 (WSFC) 節點之間部署 SQL Server 容錯移轉叢集執行個體 (FCIs)。此外， 模式會使用 Amazon FSx 共用儲存來儲存資料和日誌檔案。

當 SQL Server 資料庫遷移至 時 AWS，第一個選擇是 Amazon RDS for SQL Server。不過，有時 Amazon RDS for SQL Server 不適合，而且 SQL Server 必須部署在高度可用的架構中的 Amazon EC2 上。在此解決方案中，SQL Server FCIs會跨 WSFC 節點安裝。

此模式隨附的 Terraform 模組最多可佈建兩個 Amazon EC2 SQL Server 執行個體。Amazon FSx for Windows File Server 檔案系統可做為仲裁見證，並存放共用的資料和日誌檔案。無論設定的執行個體數量為何，SQL Server 執行個體節點一律會建立並加入 FCI 叢集，以確保環境同位。（通常，一個執行個體設定為開發，兩個執行個體用於生產環境。）對於使用兩個節點以獲得高可用性的組態，會佈建內部 Network Load Balancer。Network Load Balancer 使用 FCI 叢集上設定的運作狀態探查來識別哪個節點是主要節點。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 在個別可用區域中具有兩個子網路的 Amazon Virtual Private Cloud (Amazon VPC)。
- Amazon VPC DHCP 選項集。設定網域名稱以解析您的 Active Directory 網域名稱，以及網域和 NetBIOS 名稱伺服器以指向您的 Active Directory 網域控制站。如需詳細資訊，請參閱其他資訊中的 VPC 組態。
- AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD).
- 自訂 Amazon Machine Image (AMI)。如需詳細資訊，請參閱其他資訊中的 AMI 組態。
- 包含 SQL Server ISO 映像的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。只有在搭配提供的 `component.yaml` 檔案使用 EC2 Image Builder 來建置自訂 AMI 時，才需要此先決條件。
- AWS Key Management Service (AWS KMS) 加密金鑰。
- 根據預設，SQL Server 是使用開發人員版本產品金鑰進行安裝。生產系統應該使用由相關變數傳遞給模組的有效產品金鑰。

限制

- 此解決方案需要 AWS Managed Microsoft AD。不過，如果您願意，您可以改用自我管理 Active Directory 實作。若要這麼做，請修改包含的 Amazon FSx Terraform 模組以移除 `active_directory_id` 屬性。然後，新增自我管理 Active Directory 所需的四個屬性，如 Terraform 文件所示。
- SQL Server 設定為使用混合模式身分驗證。如果您願意，您可以使用僅限 Windows 的身分驗證。若要這樣做，請在提供的使用者資料指令碼中，移除提供給`setup.exe`命令的 `/SECURITYMODE`和 `/SAPWD` 參數。您可以移除`sql_accounts.tf`檔案，也可以修改`instances.tf`檔案以移除`sql_sa_password`項目。
- 刪除部署的叢集時，您必須在 Active Directory 中移除對應的虛擬電腦物件和個別電腦物件。若要移除物件，請使用 Active Directory 管理工具。
- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

此解決方案已使用下列版本進行測試：

- Windows Server 2019
- SQL Server 2019
- Terraform 0.13.0 版

Architecture

來源技術堆疊

- SQL Server

目標技術堆疊

- 使用 Amazon EC2 的 WSFC 節點上的 SQL Server FCI
- Amazon FSx for Windows File Server
- Amazon S3 儲存貯體
- AWS Secrets Manager

- AWS Managed Microsoft AD
- AWS KMS
- AWS Identity and Access Management (IAM)

目標架構

下圖顯示此解決方案的架構。

上圖顯示以下項目：

- 提供 EC2 執行個體存取 AWS KMS 和 Secrets Manager 的 IAM 角色
- 兩個 SQL Server 節點部署在跨兩個可用區域的私有子網路中的 Amazon EC2 執行個體上
- Network Load Balancer，用於促進與作用中 SQL Server 執行個體的連線（設定單一節點叢集時未部署）
- Amazon FSx for Windows File Server 檔案系統部署在兩個私有子網路中，供 SQL Server 節點共用儲存
- 用於存放 Active Directory 和 SQL Server 登入資料和組態的 Secrets Manager
- 用於存放 SQL Server 安裝映像的 Amazon S3 儲存貯體
- AWS Managed Microsoft AD 適用於 Windows 身分驗證
- AWS KMS 用於建立加密金鑰

自動化和擴展

您可以使用 GitHub 儲存庫中的 Terraform 模組來自動化目標架構的部署。您必須修改 `terraform.tfvars` 檔案，以包含您環境特有的變數值。Amazon S3 儲存貯體、 AWS Managed Microsoft AD 元件、 AWS KMS 加密金鑰和某些秘密是此部署的先決條件，不包含在 Terraform 程式碼中。

工具

AWS 服務

- AWS Directory Service for Microsoft Active Directory 可讓您的目錄感知工作負載 AWS 和資源在中使用 Microsoft Active Directory AWS 雲端。在此模式中， AWS Managed Microsoft AD 用於 Windows Server 和 SQL Server 身分驗證，以及 DNS。

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。在此模式中，SQL Server 容錯移轉叢集執行個體會安裝在 Amazon EC2 執行個體上。
- EC2 Image Builder 可協助您自動化自訂伺服器映像的建立、管理和部署。
- Amazon FSx for Windows File Server 在 Windows Server 上提供全受管共用儲存。在此模式中，FSx for Windows File Server 為 SQL Server 資料和日誌檔案以及規定人數見證提供共用儲存。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。在此模式中，它會用來加密 Secrets Manager 秘密、Amazon Elastic Block Store (Amazon EBS) 磁碟區上的 SQL Server 儲存體，以及 FSx for Windows File Server 檔案系統。
- AWS Secrets Manager 可協助您將程式碼中的硬式編碼憑證 (包括密碼) 取代為 Secrets Manager 的 API 呼叫，以便透過程式設計方法來擷取機密。在此模式中，用於安裝和執行 SQL Server 的 Active Directory 登入資料、sa使用者登入資料和資料庫連線資訊會儲存在 Secrets Manager 中。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。此模式使用 Amazon S3 儲存貯體來存放 SQL Server 安裝映像。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。

其他工具

- Microsoft SQL Server FCIs安裝在 Windows Server 叢集節點之間。此外，它們可以跨多個子網路安裝。在此模式中，SQL Server FCI 執行個體會跨 WSFC 節點安裝。
- Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。在此模式中，Terraform 用於建立資源和設定 SQL Server FCI 執行個體。
- Windows Server 容錯移轉叢集提供基礎設施功能，可支援 SQL Server 等託管伺服器應用程式的高可用性。在此模式中，FCI 節點會利用 WSFC 功能，透過執行個體層級的備援提供本機高可用性。

程式碼儲存庫

此模式的程式碼可在 GitHub cluster-amazon-elastic-compute-cloud-amazon-fsx-microsoft-sql-server 儲存庫中使用。下列資源可在 儲存庫中使用：

- 提供解決方案概觀和其他安裝和使用資訊`README.md`的檔案
- 一組基本的 Terraform 組態檔案和 Amazon FSx 特定模組，用於佈建此模式的元件

- 做為 Amazon EC2 使用者資料指令碼執行的執行個體設定指令碼
- Image Builder 可用來建立自訂 AMI 的 `component.yaml` 檔案

最佳實務

安全性和修補

- AMI 先決條件安裝和組態是部署 SQL Server FCI 叢集的最低需求。可能需要其他軟體和組態，才能符合組織的標準和安全要求。
- 部署之後，請持續修補 Windows。您可以直接修補執行中的執行個體，或使用最新的 Windows 修補程式建立新的 AMI，並使用新的 AMI 取代執行個體 （一次一個）。 每月 AWS 發行新的 Windows AMIs，其中包含最新的作業系統修補程式、驅動程式和啟動代理程式。我們建議您在啟動新執行個體或建置自訂映像時，檢查是否有最新的 AMI。
- Amazon EC2 執行個體設定為允許所有傳出流量。在生產環境中部署時，應設定安全群組中的傳出規則，將此流量限制在所需的目的地。
- FSx for Windows File Server 檔案系統可以自動記錄檔案共用和檔案和資料夾存取的稽核日誌，如果這是您環境中的需求，請將它們運送到您所需的目的地。
- 定期自動輪換 Secrets Manager 秘密。對於 Amazon EC2 執行個體金鑰對，請考慮自動輪換解決方案，如如何使用 AWS Secrets Manager 安全地存放和輪換 SSH 金鑰對所述。對於 Active Directory 登入資料和 SQL Server `sa`登入資料秘密，請根據您的密碼管理政策設定自動輪換。

Active Directory 管理

- 在 FCI 叢集中，Windows 會在 Active Directory 中產生電腦名稱物件 (CNO)。CNO 會回應 DNS 請求，並將流量轉送至作用中的 SQL 節點。我們不建議使用此 Active Directory 提供的 DNS。TTL 太高，無法提供合理的容錯移轉時間，通常需要最多 5 分鐘才能反映新的主要 IP 地址。相反地，對於高可用性的安裝，內部 Network Load Balancer 設定為在 30 秒內容錯移轉。
- 建立叢集需要 Active Directory 網域管理員。此需求是因為在 Active Directory 中建立叢集物件和修改許可所需的許可較高。不過，SQL Server 服務不需要以網域管理員身分執行。因此，我們建議您為此目的建立第二個 Active Directory 使用者。不過，如果服務將以網域管理員使用者身分執行，您可以消除此使用者。在這種情況下，網域管理員使用者必須新增至在此模式中建立的 Active Directory 管理員群組。

史詩

設定叢集登入資料

任務	Description	所需的技能
建立 Active Directory 群組。	在 中 AWS Managed Microsoft AD，建立下列群組： • 叢集管理員群組 – 此群組將新增至每個叢集節點上的本機管理員群組。 • 叢集遠端桌面群組 – 此群組將新增至每個叢集節點上的本機遠端桌面使用者群組。 如需詳細資訊，請參閱 AWS 文件中的建立 AWS Managed Microsoft AD 群組。	AD 管理員
建立 Active Directory 使用者。	在 中 AWS Managed Microsoft AD，建立下列使用者 • 網域管理員使用者 – 使用此帳戶建立叢集。 • 網域使用者 – SQL Server 服務將使用此帳戶執行。將此使用者新增至您在上一個任務中建立的叢集管理員群組。 如需詳細資訊，請參閱 AWS 文件中的建立 AWS Managed Microsoft AD 使用者。	AD 管理員
將 Active Directory 登入資料新增至秘密。	使用 Secrets Manager 建立四個秘密來存放下列資訊：	AWS 管理員

任務	Description	所需的技能
	• 網域管理員使用者的使用者名稱 • 網域管理員使用者的密碼 • 網域使用者的使用者名稱 • 網域使用者的密碼 如需詳細資訊，請參閱 AWS 文件中的建立 AWS Secrets Manager 秘密。	

準備環境

任務	Description	所需的技能
建立 Windows AMI。	建立包含必要軟體和組態的自訂 Windows AMI。如需詳細資訊，請參閱其他資訊。	AWS 管理員、AWS DevOps
安裝 Terraform。	若要安裝 Terraform，請遵循 Terraform 網站上的說明。	AWS DevOps
複製儲存庫。	複製此模式的儲存庫。如需詳細資訊，請參閱 GitHub 網站上的複製儲存庫。	AWS DevOps

安裝叢集

任務	Description	所需的技能
修改 Terraform 變數。	更新提供的`terraform.tfvars` 檔案，將所有變數設定為適合您環境的值。	AWS DevOps

任務	Description	所需的技能
	例如，更新 `domain_group_administrators` 和 `domain_group_rdp_users` 變數以使用您的 Active Directory 網域名稱和先前建立的 Active Directory 群組名稱。	
初始化 Terraform。	若要查看提議的部署，請導覽至儲存庫的根目錄。使用 Terraform 命令列界面 (CLI) 執行 `terraform init`，然後執行 `terraform plan`。	AWS DevOps
部署 資源。	若要部署 SQL 叢集和相關聯的資源，請使用 Terraform CLI 來執行 `terraform apply`。	AWS DevOps、AWS 管理員

任務	Description	所需的技能
驗證部署。	若要驗證部署，請使用下列步驟： 1. 使用遠端桌面連線至其中一個已部署的 Windows Amazon EC2 執行個體。 2. 如需詳細資訊，請參閱 AWS 文件中的使用 RDP 連線至 Windows 執行個體。 3. 開啟 Windows 容錯移轉叢集管理員。驗證叢集是否已建立，以及 SQL Server 角色是否已建立且正在執行。 4. 若要使用 SQL Server 測試連線和身分驗證，請使用資料庫工具，例如 SQL Server Management Studio 來連線至 SQL Server 端點。（端點值存放在 Secrets Manager 中。）如需詳細資訊，請參閱 AWS 文件中的連線至 Amazon EC2 上的 Microsoft SQL Server。	DBA、AWS 系統管理員

疑難排解

問題	解決方案
Terraform 佈建已完成，但 Windows 容錯移轉叢集管理員未顯示叢集已建立或叢集處於不可操作狀態。	叢集的整個資源和組態安裝可能需要 45-60 分鐘。Terraform 完成後，使用者資料指令碼必須執行到完成，這需要多次重新啟動。若要監控進度，您可以使用`C:\`磁碟機中的 `Checkpoints` 目錄和 中的 SQL Server 安裝

問題	解決方案
	日誌`C:\Program Data\Microsoft SQL Server\150\Log` 。完成後，`C:\ProgramData\Amazon\EC2-Windows\Launch\Log\UserdataExecution.log` 檔案中會提供安裝完成訊息。
佈建正常運作的叢集之後，使用 Terraform 刪除並重新建立叢集不會成功。Terraform 完成，但叢集未正確設定。	佈建程序的一部分涉及向 Active Directory 和 Active Directory DNS 註冊機器和虛擬物件。當 Amazon EC2 叢集節點和叢集節點存在電腦名稱時，FCI 無法正確初始化，且佈建會失敗。 若要修正此問題，請執行下列步驟： 1. 刪除 Active Directory 使用者和電腦中的 Amazon EC2 節點、叢集虛擬電腦和叢集 ID。 2. 在 Active Directory DNS 中刪除叢集虛擬電腦的 DNS 項目。 3. 執行下列命令來刪除叢集 ID 隨機字串 Terraform 資源 `terragrunt destroy -target=random_string.cluster_id` 。此動作將刪除現有的 Amazon EC2 執行個體。 4. 執行 `terraform apply`，並預期以下三個新資源：兩個 FCI Amazon EC2 執行個體和 1 個隨機字串叢集 ID。

相關資源

AWS 文件

- 使用映像建置器建立自訂映像
- 建立 KMS 金鑰
- 建立一般用途儲存貯體

- 建立金鑰政策
- 建立您的 AWS Managed Microsoft AD

其他資訊

Terraform 模組資訊

此模組使用混合 AMI 組態和使用者資料組態，以取得佈建時間和穩定性的良好組合。在佈建期間，Windows 需要多次重新啟動和等待。已實作檢查點方法，以防止持久性使用者資料重新啟動期間無限迴圈。使用者資料設定為持久性。因此，使用者資料組態指令碼具有且必須繼續開發為等冪性。冪等性可簡化更新程序，允許在更新週期期間交換執行個體，無需手動設定即可重新加入或重新建立 FCI 叢集。

SQL Server 連線字串和容錯移轉叢集

模組將發佈秘密，其中包含應用於此資料庫連線字串的端點地址。秘密名稱遵循格式 `{environment_name}/sqlserver/{cluster_name}/endpoint`。對於只使用一個節點的安裝，您可以預期這是 Amazon EC2 執行個體 SQL Server 介面的 IP 地址。對於高可用性安裝（兩個執行個體），您可以預期這是內部 Network Load Balancer 的 DNS 名稱。

此模組不支援容錯移轉叢集虛擬 IPs。虛擬 IP 必須保留在相同的子網路中才能運作。在 中 AWS，單一子網路無法跨越多個可用區域。因此，使用虛擬 IPs 會移除將此模組視為高度可用的能力。

每個 Amazon EC2 執行個體都會獲得三個私有 IP 地址。它們的 usai 如下所示：

- 網路流量的主要 IP – 輸出流量的來源 IP。
- FCI 通訊 – 用來維護容錯移轉叢集的狀態和同步。
- SQL Server (TCP 連接埠 1433) – 接聽程式和 也會接聽活動訊號流量，以判斷哪個執行個體是主要執行個體。

VPC 組態

先決條件會列出設定為使用 Active Directory 進行 DNS 解析的 DHCP 選項集。不過，此先決條件並非硬性要求。硬性要求是 EC2 執行個體必須能夠解析您的 Active Directory 網域名稱。您可以透過其他方式達成此要求，例如使用 Amazon Route 53 Resolver 端點。如需詳細資訊，請參閱整合 Directory Service 的 DNS 解析與 Amazon Route 53 解析程式 (AWS 部落格文章）。

AMI 組態

此模式中使用的 AMI 必須包含下列必要軟體和組態：

1. 下載 SQL Server 2019 安裝檔案並將其展開至 C:\SQL_Install_media。
2. 安裝下列 Windows 功能：
   - Install-WindowsFeature Failover-Clustering
   - Install-WindowsFeature RSAT-AD-PowerShell
   - Install-WindowsFeature RSAT-AD-Tools
   - Install-WindowsFeature RSAT-Clustering-Mgmt
   - Install-WindowsFeature RSAT-Clustering-PowerShell
   - Install-WindowsFeature RSAT-Clustering-CmdInterface
3. 停用 Windows 防火牆，如下所示：
   - Get-NetFirewallProfile | Set-NetFirewallProfile -Enabled False
4. 啟用 CredSSP 身分驗證方法 (<domain>以組織的 Windows 網域名稱取代），如下所示：
   - Enable-WSManCredSSP -Role "Server" -Force
   - Enable-WSManCredSSP -Role "Client" -DelegateComputer *.<domain>.com -Force
5. 設定下列登錄機碼：
   - 允許 NTLM 身分驗證憑證：
     - HKLM:\Software\Policies\Microsoft\Windows\CredentialsDelegation
       - 名稱：AllowFreshCredentialsWhenNTLMOnly
       - 值：1
       - 類型：REG_DWORD
   - 允許本機網域電腦從 PowerShell 使用 NTLM：
     - 路徑： HKLM:\Software\Policies\Microsoft\Windows\CredentialsDelegation\AllowFreshCredentialsWhenNTLMOnly
       - 名稱：1
       - 值：wsman/*.<domain>.com
       - 類型：REG_SZ
6. 設定 PowerShell Gallery，如下所示：
   - [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
   - Install-PackageProvider -Name NuGet -MinimumVersion 2.8.5.201 -Force
   - Set-PSRepository -Name PSGallery -InstallationPolicy Trusted

7. 安裝下列 Windows PowerShell 模組*：
   - `Install-Module -Name ComputerManagementDsc`
   - `Install-Module -Name FailOverClusterDsc`
   - `Install-Module -Name PSDscResources`
   - `Install-Module -Name xSmbShare`
   - `Install-Module -Name xActiveDirectory`
   - `Install-Module -Name SqlServer`

若要使用映像建置器建立 AMI，請遵循映像建置器文件中使用 EC2 Image Builderconsole 精靈建立映像管道中的指示。若要使用先前的先決條件建立配方的元件，請使用下列步驟：

1. 從 GitHub 儲存庫的 `ami` 資料夾下載 component.yaml 檔案。
2. 將內容複製到新的映像建置器元件。
3. 使用您的資訊更新下列預留位置：
   - `<domain>` – 您的 Active Directory 網域名稱
   - `<bucket_name>` – 包含 SQL Server 映像的 Amazon S3 儲存貯體名稱

在 Amazon Aurora PostgreSQL 和 Amazon RDS for PostgreSQL 中模擬 Oracle PL/SQL 關聯陣列

Rajkumar Raghuwanshi、Bhanu Ganesh Gudivada 和 Sachin Khanna，Amazon Web Services

總結

此模式說明如何模擬 Amazon Aurora PostgreSQL 和 Amazon RDS for PostgreSQL 環境中具有空索引位置的 Oracle PL/SQL 關聯陣列。它還描述了 Oracle PL/SQL 關聯陣列和 PostgreSQL 陣列在遷移期間如何處理空索引位置的一些差異。

我們提供 PostgreSQL 替代方案，除了在遷移 Oracle 資料庫時使用 `aws_oracle_ext`函數來處理空的索引位置。此模式使用額外的資料欄來存放索引位置，並維護 Oracle 處理稀疏陣列，同時整合原生 PostgreSQL 功能。

Oracle

在 Oracle 中，集合可以初始化為空白，並使用`EXTEND`集合方法填入，這會將`NULL`元素附加到陣列。使用由 編製索引的 PL/SQL 關聯陣列時`PLS_INTEGER`，`EXTEND`方法會依序新增`NULL`元素，但元素也可以在非連續索引位置初始化。任何未明確初始化的索引位置都會保持空白。

此彈性允許稀疏陣列結構，其中元素可以填入任意位置。使用`FOR LOOP`具有 `FIRST`和 `LAST` 邊界的反覆查看集合時，只會處理初始化的元素 （ 無論 `NULL`還是具有定義的值 ），同時略過空位置。

PostgreSQL (Amazon Aurora 和 Amazon RDS)

PostgreSQL 處理空值的方式與`NULL`值不同。它會將空值儲存為使用一個位元組儲存體的不同實體。當陣列具有空值時，PostgreSQL 會指派循序索引位置，就像非空值一樣。但是，循序索引需要額外處理，因為系統必須逐一查看所有索引位置，包括空的位置。這會使稀疏資料集的傳統陣列建立效率降低。

AWS Schema Conversion Tool

AWS Schema Conversion Tool (AWS SCT) 通常會使用 `aws_oracle_ext`函數處理 Oracle-to-PostgreSQL 的遷移。在此模式中，我們提出使用原生 PostgreSQL 功能的替代方法，該功能結合了 PostgreSQL 陣列類型和用於儲存索引位置的額外資料欄。然後，系統只需使用索引欄即可反覆查看陣列。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 具有管理員許可的 AWS Identity and Access Management (IAM) 使用者。
- 與 Amazon RDS 或 Aurora PostgreSQL 相容的執行個體。
- 對關聯式資料庫的基本了解。

限制

- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

產品版本

此模式已使用下列版本進行測試：

- Amazon Aurora PostgreSQL 13.3
- Amazon RDS for PostgreSQL 13.3
- AWS SCT 1.0.674
- Oracle 12c EE 12.2

Architecture

來源技術堆疊

- 內部部署 Oracle 資料庫

目標技術堆疊

- Amazon Aurora PostgreSQL
- Amazon RDS for PostgreSQL

目標架構

上圖顯示以下項目：

- 來源 Amazon RDS for Oracle 資料庫執行個體

- 使用 AWS SCT 將 Oracle 函數轉換為 PostgreSQL 對等項目的 Amazon EC2 執行個體
- 與 Amazon Aurora PostgreSQL 相容的目標資料庫

工具

AWS 服務

- Amazon Aurora 是一種全受管關聯式資料庫引擎，專為雲端而建置，並與 MySQL 和 PostgreSQL 相容。
- Amazon Aurora PostgreSQL 相容版本是完全受管的 ACID 相容關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 中設定、操作和擴展關聯式資料庫 AWS 雲端。
- Amazon Relational Database Service (Amazon RDS) for Oracle 可協助您在 中設定、操作和擴展 Oracle 關聯式資料庫 AWS 雲端。
- 適用於 PostgreSQL 的 Amazon Relational Database Service (Amazon RDS) 可協助您在 中設定、操作和擴展 PostgreSQL 關聯式資料庫 AWS 雲端。
- AWS Schema Conversion Tool (AWS SCT) 透過自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，支援異質資料庫遷移。

其他工具

- Oracle SQL Developer 是一種整合的開發環境，可簡化傳統和雲端部署中 Oracle 資料庫的開發和管理。
- pgAdmin 是 PostgreSQL 的開放原始碼管理工具。它提供圖形界面，可協助您建立、維護和使用資料庫物件。在此模式中，pgAdmin 會連線至 RDS for PostgreSQL 資料庫執行個體並查詢資料。或者，您可以使用 psql 命令列用戶端。

最佳實務

- 測試資料集邊界和邊緣案例。
- 考慮針對out-of-bounds索引條件實作錯誤處理。

- 最佳化查詢以避免掃描稀疏資料集。

史詩

Oracle 關聯陣列行為 （ 來源 ）

任務	Description	所需的技能
在 Oracle 中建立來源 PL/SQL 區塊。	在 Oracle 中建立使用下列關聯陣列的來源 PL/SQL 區塊： `DECLARE TYPE country_codes IS TABLE OF VARCHAR2(100) INDEX BY pls_integer; cc country_codes; cc_idx NUMBER := NULL; BEGIN cc(7) := 'India'; cc(3) := 'UK'; cc(5) := 'USA'; cc(0) := 'China'; cc(-2) := 'Invalid'; dbms_output.put_line('cc_length:' || cc.COUNT); IF (cc.COUNT > 0) THEN cc_idx := cc.FIRST; FOR i IN 1..cc.COUNT LOOP dbms_output.put_line('cc_idx:' || cc_idx || ' country:' || cc(cc_idx)); cc_idx := cc.next(cc_idx);`	DBA

任務	Description	所需的技能

	END LOOP;
	END IF;
	END;	

執行 PL/SQL 區塊。	在 Oracle 中執行來源 PL/SQL 區塊。如果關聯陣列的索引值之間存在間隙，則不會在這些間隙中存放任何資料。這可讓 Oracle 迴圈僅逐一查看索引位置。	DBA
檢閱輸出。	以非連續間隔將五個元素插入陣列 (cc)。陣列計數會顯示在下列輸出中： cc_length:5 cc_idx:-2 country:I nvalid cc_idx:0 country:China cc_idx:3 country:UK cc_idx:5 country:USA cc_idx:7 country:India	DBA

PostgreSQL 關聯陣列行為 （目標）

任務	Description	所需的技能
在 PostgreSQL 中建立目標 PL/pgSQL 區塊。 PostgreSQL	在 PostgreSQL 中建立使用以下關聯陣列的目標 PL/pgSQL 區塊： PostgreSQL DO $$ DECLARE cc character varying(100)[]; cc_idx integer := NULL;	DBA

任務	Description	所需的技能

	BEGIN
	    cc[7] := 'India';
	    cc[3] := 'UK';
	    cc[5] := 'USA';
	    cc[0] := 'China';
	    cc[-2] := 'Invalid'
	;
	    RAISE NOTICE
	'cc_length: %',
	ARRAY_LENGTH(cc, 1);
	    IF (ARRAY_LE
	NGTH(cc, 1) > 0) THEN
	        FOR i IN
	ARRAY_LOWER(cc,
	1)..ARRAY_UPPER(cc,
	1)
	        LOOP
	            RAISE NOTICE
	'cc_idx:% country:%',
	i, cc[i];
	        END LOOP;
	    END IF;
	END;
	$$;	

執行 PL/pgSQL 區塊。	在 PostgreSQL 中執行目標 PL/pgSQL 區塊。 PostgreSQL 如果關聯陣列的索引值之間存在間隙，則不會在這些間隙中存放任何資料。這可讓 Oracle 迴圈僅逐一查看索引位置。	DBA

任務	Description	所需的技能

檢閱輸出。	陣列長度大於 5，因為 NULL 存放在索引位置之間的間隙中。如下列輸出所示，迴圈會完成 10 次反覆運算，以擷取陣列中的 5 個值。
	
	`cc_length:10`
	`cc_idx:-2 country:I`
	`nvalid`
	`cc_idx:-1 country:<`
	`NULL>`
	`cc_idx:0 country:China`
	`cc_idx:1 country:<`
	`NULL>`
	`cc_idx:2 country:<`
	`NULL>`
	`cc_idx:3 country:UK`
	`cc_idx:4 country:<`
	`NULL>`
	`cc_idx:5 country:USA`
	`cc_idx:6 country:<`
	`NULL>`
	`cc_idx:7 country:India`
		DBA

模擬 Oracle 關聯陣列行為

任務	Description	所需的技能
使用陣列和使用者定義的類型建立目標 PL/pgSQL 區塊。	為了最佳化效能並符合 Oracle 的功能，我們可以建立使用者定義的類型，同時存放索引位置及其對應的資料。這種方法透過維持索引和值之間的直接關聯來減少不必要的反覆運算。	DBA

任務	Description	所需的技能

	DO $$
	DECLARE
	    cc country_codes[];
	    cc_append country_c
	odes := NULL;
	    i record;
	BEGIN
	    cc_append.idx = 7;
	    cc_append.val =
	 'India';
	    cc := array_app
	end(cc, cc_append);
	    cc_append.idx = 3;
	    cc_append.val =
	 'UK';
	    cc := array_app
	end(cc, cc_append);
	    cc_append.idx = 5;
	    cc_append.val =
	 'USA';
	    cc := array_app
	end(cc, cc_append);
	    cc_append.idx = 0;
	    cc_append.val =
	 'China';
	    cc := array_app
	end(cc, cc_append);
	    cc_append.idx = -
	 2;
	    cc_append.val =
	 'Invalid';
	    cc := array_app
	end(cc, cc_append);
	    RAISE NOTICE
	 'cc_length: %',
	 ARRAY_LENGTH(cc, 1);
	    IF (ARRAY_LE
	NGTH(cc, 1) > 0) THEN
	        FOR i IN (
	            SELECT
	                *	

任務	Description	所需的技能

	<pre>FROM
	
	unnest(cc)
	ORDER BY
	idx)
	LOOP
	
	RAISE NOTICE 'cc_idx:
	% country:%', i.idx,
	i.val;
	END
	LOOP;
	END IF;
	END;
	$$;</pre>	

執行 PL/pgSQL 區塊。	執行目標 PL/pgSQL 區塊。如果關聯陣列的索引值之間存在間隙，則不會在這些間隙中存放任何資料。這可讓 Oracle 迴圈僅逐一查看索引位置。	DBA

檢閱輸出。	如下列輸出所示，使用者定義的類型只會存放填入的資料元素，這表示陣列長度符合值的數量。因此，LOOP反覆運算已最佳化，僅處理現有資料，無需追蹤空位置。
	
	<pre>cc_length:5
	cc_idx:-2 country:I
	nvalid
	cc_idx:0 country:China
	cc_idx:3 country:UK
	cc_idx:5 country:USA
	cc_idx:7 country:India</pre>
		DBA

相關資源

AWS 文件

- AWS 資料庫部落格
- Oracle 到 Aurora PostgreSQL 遷移手冊

其他文件

- Oracle 關聯陣列
- PostgreSQL 陣列函數和運算子
- PostgreSQL 使用者定義類型

在 Amazon RDS 中啟用 PostgreSQL 資料庫執行個體的加密連線

Rohit Kapoor，Amazon Web Services

總結

Amazon Relational Database Service (Amazon RDS) 支援 PostgreSQL 資料庫執行個體的 SSL 加密。使用 SSL，您可以加密應用程式與 Amazon RDS for PostgreSQL 資料庫執行個體之間的 PostgreSQL 連線。根據預設，Amazon RDS for PostgreSQL 會使用 SSL/TLS，並預期所有用戶端都會使用 SSL/TLS 加密進行連線。Amazon RDS for PostgreSQL 支援 TLS 1.1 和 1.2 版。

此模式說明如何啟用 Amazon RDS for PostgreSQL 資料庫執行個體的加密連線。您可以使用相同的程序來啟用 Amazon Aurora PostgreSQL 相容版本的加密連線。

先決條件和限制

- 作用中的 AWS 帳戶
- Amazon RDS for PostgreSQL 資料庫執行個體
- SSL 套件

Architecture

工具

- pgAdmin 是 PostgreSQL 的開放原始碼管理和開發平台。您可以在 Linux、Unix、macOS 和 Windows 上使用 pgAdmin 來管理 PostgreSQL 10 和更新版本中的資料庫物件。
- PostgreSQL 編輯器提供更易於使用的界面，協助您建立、開發和執行查詢，以及根據您的需求編輯程式碼。

最佳實務

- 監控不安全的資料庫連線。
- 稽核資料庫存取權。
- 確定備份和快照已在靜態加密。
- 監控資料庫存取。

- 避免不受限制的存取群組。
- 使用 Amazon GuardDuty 增強您的通知。
- 定期監控政策遵循。

史詩

下載信任的憑證並將其匯入您的信任存放區

任務	Description	所需的技能
將信任的憑證載入您的電腦。	若要將憑證新增至您電腦的信任根憑證授權機構存放區，請遵循下列步驟。（這些指示使用 Window Server 做為範例。） 1. 在 Windows Server 中，選擇開始、執行，然後輸入mmc。 2. 在 主控台中，選擇檔案、新增/移除 Snap-in。 3. 可用不足嵌入，選擇憑證，然後選擇新增。 4. 在此Snap-in下，一律會管理憑證，選擇電腦帳戶，下一步。 5. ChooseLocal 電腦，完成。 6. 如果您不再需要將 Snap-in 新增至主控台，請選擇確定。 7. 在主控台樹狀目錄中，按兩下憑證。 8. 用滑鼠右鍵按一下信任的根憑證授權機構。 9. 選擇所有任務、匯入以匯入下載的憑證。	DevOps 工程師、遷移工程師、DBA

任務	Description	所需的技能
	10請遵循憑證匯入精靈中的步驟。	

強制 SSL 連線

任務	Description	所需的技能
建立參數群組並設定 rds.force_ssl 參數。	如果 PostgreSQL 資料庫執行個體具有自訂參數群組，請編輯參數群組並變更為 `rds.force_ssl` 1。 如果資料庫執行個體使用未`rds.force_ssl` 啟用的預設參數群組，請建立新的參數群組。您可以使用 Amazon RDS API 或手動修改新的參數群組，如下列說明所示。 若要建立新的參數群組： 1. 登入 AWS 管理主控台，並為託管資料庫執行個體的 AWS 區域開啟 Amazon RDS 主控台。 2. 在導覽窗格中，選擇 Parameter groups (參數群組)。 3. 選擇建立參數群組，然後設定下列值： • 針對參數群組系列，選擇 postgres14。	DevOps 工程師、遷移工程師、DBA

任務	Description	所需的技能
	• 針對群組名稱，輸入 pgsql-<database_instance>-ssl。 • 針對描述，輸入您要新增之參數群組的自由格式描述。 • 選擇建立。 4. 選擇您建立的參數群組。 5. 從 Parameter group actions (參數群組動作)，選擇 Edit (編輯)。 6. 尋找 rds.force_ssl 並將其設定變更為 1。 **Note** 變更此參數之前，請先執行用戶端測試。 7. 選擇儲存變更。 若要將參數群組與 PostgreSQL 資料庫執行個體建立關聯： 1. 在 Amazon RDS 主控台的導覽窗格中，選擇資料庫，然後選擇 PostgreSQL 資料庫執行個體。 2. 選擇 Modify (修改)。 3. 在其他組態下，選擇新的參數群組，然後選擇繼續。 4. 在排程修改下，選擇立即套用。	

任務	Description	所需的技能
	5. 選擇 Modify DB instance (修改資料庫執行個體)。 如需詳細資訊，請參閱 Amazon RDS 文件。	
強制 SSL 連線。	連線至 Amazon RDS for PostgreSQL 資料庫執行個體。不使用 SSL 的連線嘗試會遭到拒絕，並顯示錯誤訊息。如需詳細資訊，請參閱 Amazon RDS 文件。	DevOps 工程師、遷移工程師、DBA

安裝 SSL 延伸模組

任務	Description	所需的技能

安裝 SSL 延伸模組。	1. 啟動 psql 或 pgAdmin 連線做為 DBA。
	2. 呼叫 ssl_is_used() 函數來判斷是否使用 SSL。
	
	`select ssl_is_used();`
	
	t 如果連線使用 SSL，函數會傳回 ；否則會傳回 f。
	3. 安裝 SSL 延伸模組。
	
	`create extension sslinfo;`
	`show ssl;`
	`select ssl_cipher();`
		DevOps 工程師、遷移工程師、DBA

任務	Description	所需的技能
	如需詳細資訊，請參閱 Amazon RDS 文件。	

為 SSL 設定 PostgreSQL 用戶端

任務	Description	所需的技能

設定 SSL 的用戶端。	透過使用 SSL，您可以啟動 PostgreSQL 伺服器，並支援使用 TLS 通訊協定的加密連線。伺服器會接聽相同 TCP 連接埠上的標準和 SSL 連線，並與任何連線用戶端協商是否使用 SSL。根據預設，這是用戶端選項。
	
	如果您使用的是 psql 用戶端：
	
	1. 確定 Amazon RDS 憑證已載入您的本機電腦。
	2. 新增下列項目以啟動 SSL 用戶端連線：
	
	`psql postgres -h SOMEHOST.amazonaws.com -p 8192 -U someuser sslmode=verify-full sslrootcert=rds-ssl-ca-cert.pem`
	`select ssl_cipher();`
	
	對於其他 PostgreSQL 用戶端：
		DevOps 工程師、遷移工程師、DBA

任務	Description	所需的技能
	• 修改個別應用程式公有金鑰參數。這可以做為選項、做為連線字串的一部分，或做為 GUI 工具中連線頁面上的屬性。 檢閱這些用戶端的下列頁面： • pgAdmin 文件 • JDBC 文件	

疑難排解

問題	解決方案
無法下載 SSL 憑證。	請檢查您的網站連線，然後重試下載憑證到您的本機電腦。

相關資源

- Amazon RDS for PostgreSQL 文件
- 搭配 PostgreSQL 資料庫執行個體使用 SSL (Amazon RDS 文件）
- 使用 SSL 保護 TCP/IP 連線 (PostgreSQL 文件）
- 使用 SSL (JDBC 文件）

加密現有的 Amazon RDS for PostgreSQL 資料庫執行個體

Piyush Goyal、Shobana Raghu 和 Yaser Raja，Amazon Web Services

總結

此模式說明如何在最短的停機時間下，加密 AWS 雲端中 PostgreSQL 資料庫執行個體的現有 Amazon Relational Database Service (Amazon RDS)。此程序也適用於 Amazon RDS for MySQL 資料庫執行個體。

您可以在建立 Amazon RDS 資料庫執行個體時啟用加密，但無法在建立之後啟用加密。不過，您可以透過建立資料庫執行個體的快照，然後建立該快照的加密複本，將加密新增至未加密的資料庫執行個體。然後，您可以從加密快照還原資料庫執行個體，以取得原始資料庫執行個體的加密副本。如果您的專案允許在此活動期間停機（至少用於寫入交易），這正是您需要做的。當資料庫執行個體的新加密複本可用時，您可以將應用程式指向新資料庫。不過，如果您的專案不允許此活動的重大停機時間，您需要替代方法來協助將停機時間降至最低。此模式使用 AWS Database Migration Service (AWS DMS) 來遷移和持續複寫資料，以便在最短的停機時間內完成切換到新的加密資料庫。

Amazon RDS 加密資料庫執行個體使用業界標準的 AES-256 加密演算法，在託管 Amazon RDS 資料庫執行個體的伺服器上加密您的資料。資料加密後，Amazon RDS 會以透明的方式處理資料的存取和解密身分驗證，並將對效能的影響降至最低。您不需要修改資料庫用戶端應用程式即可使用加密。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 未加密的 Amazon RDS for PostgreSQL 資料庫執行個體
- 使用（建立、修改或停止）AWS DMS 任務的經驗（請參閱 AWS DMS 文件中的使用 AWS DMS 任務）
- 熟悉用於加密資料庫的 AWS Key Management Service (AWS KMS)（請參閱 AWS KMS 文件）

限制

- 您只能在建立 Amazon RDS 資料庫執行個體時啟用加密，而不是在建立資料庫執行個體之後。
- 未使用快照還原未記錄資料表中的資料。如需詳細資訊，請參閱使用 PostgreSQL 的最佳實務。
- 未加密資料庫執行個體不可以有加密僅供讀取複本，加密資料庫執行個體也不可以有未加密僅供讀取複本。

- 您無法將未加密的備份或快照還原至已加密的資料庫執行個體。
- AWS DMS 不會自動傳輸序列，因此需要其他步驟才能處理此問題。

如需詳細資訊，請參閱 Amazon RDS 文件中的 Amazon RDS 加密資料庫執行個體限制。

Architecture

來源架構

- 未加密的 RDS 資料庫執行個體

目標架構

- 加密的 RDS 資料庫執行個體
  - 透過還原來源 RDS 資料庫執行個體的資料庫快照複本，即可建立目的地 RDS 資料庫執行個體。
  - 還原快照時，AWS KMS 金鑰用於加密。
  - AWS DMS 複寫任務用於遷移資料。

工具

用來啟用加密的工具：

- 用於加密的 AWS KMS 金鑰 – 當您建立加密的資料庫執行個體時，您可以選擇客戶受管金鑰或 Amazon RDS 的 AWS 受管金鑰來加密資料庫執行個體。如果您未指定客戶受管金鑰的金鑰識別符，Amazon RDS 會為您的新資料庫執行個體使用 AWS 受管金鑰。Amazon RDS 會為您的 AWS 帳戶建立 Amazon RDS 的 AWS 受管金鑰。您的 AWS 帳戶對於每個 AWS 區域都有不同的 Amazon RDS AWS 受管金鑰。如需使用 KMS 金鑰進行 Amazon RDS 加密的詳細資訊，請參閱加密 Amazon RDS 資源。

用於持續複寫的工具：

- AWS DMS – 您可以使用 AWS Database Migration Service (AWS DMS) 將變更從來源資料庫複寫到目標資料庫。請務必讓來源和目標資料庫保持同步，以將停機時間降至最低。如需有關設定 AWS DMS 和建立任務的資訊，請參閱 AWS DMS 文件。

史詩

建立來源資料庫執行個體的快照並進行加密

任務	Description	所需的技能
檢查來源 PostgreSQL 資料庫執行個體的詳細資訊。	在 Amazon RDS 主控台上，選擇來源 PostgreSQL 資料庫執行個體。在組態索引標籤上，確定執行個體未啟用加密。如需畫面圖例，請參閱其他資訊一節。	DBA
建立資料庫快照。	建立您要加密之執行個體的資料庫快照。建立快照所需的時間取決於資料庫的大小。如需說明，請參閱 Amazon RDS 文件中的建立資料庫快照。	DBA
加密快照。	在 Amazon RDS 主控台導覽窗格中，選擇快照，然後選取您建立的資料庫快照。針對 Actions (動作) 選擇 Copy Snapshot (複製快照)。在對應的欄位中，提供目的地 AWS 區域和資料庫快照複本的名稱。選取啟用加密核取方塊。在 Master Key (主金鑰) 中，指定用來加密資料庫快照副本的 KMS 金鑰識別符。選擇 Copy Snapshot (複製快照)。如需詳細資訊，請參閱 Amazon RDS 文件中的複製快照。	DBA

準備目標資料庫執行個體

任務	Description	所需的技能
還原資料庫快照。	在 Amazon RDS 主控台上，選擇快照。選擇您建立的加密快照。針對 Actions (動作)，選擇 Restore Snapshot (還原快照)。針對資料庫執行個體識別符，提供新資料庫執行個體的唯一名稱。檢閱執行個體詳細資訊，然後選擇還原資料庫執行個體。系統會從您的快照建立新的加密資料庫執行個體。如需詳細資訊，請參閱 Amazon RDS 文件中的從資料庫快照還原。	DBA
使用 AWS DMS 遷移資料。	在 AWS DMS 主控台上，建立 AWS DMS 任務。針對遷移類型，選擇遷移現有資料並複寫進行中的變更。在任務設定中，針對目標資料表準備模式，選擇截斷。如需詳細資訊，請參閱 AWS DMS 文件中的建立任務。	DBA
啟用資料驗證。	在任務設定中，選擇啟用驗證。這可讓您比較來源資料與目標資料，以確認資料已準確遷移。	DBA
停用目標資料庫執行個體的限制。	停用目標資料庫執行個體上的任何觸發條件和外部金鑰限制，然後啟動 AWS DMS 任務。如需停用觸發條件和外部	DBA

任務	Description	所需的技能
	金鑰限制的詳細資訊，請參閱 AWS DMS 文件。	
驗證資料。	完全載入完成後，請驗證目標資料庫執行個體上的資料，以查看是否符合來源資料。如需詳細資訊，請參閱 AWS DMS 文件中的 AWS DMS 資料驗證。	DBA

切換到目標資料庫執行個體

任務	Description	所需的技能
在來源資料庫執行個體上停止寫入操作。	停止來源資料庫執行個體上的寫入操作，以便開始應用程式停機時間。確認 AWS DMS 已完成管道中資料的複寫。在目標資料庫執行個體上啟用觸發和外部索引鍵。	DBA
更新資料庫序列	如果來源資料庫包含任何序號，請驗證並更新目標資料庫中的序列。	DBA
設定應用程式端點。	設定您的應用程式連線以使用新的 Amazon RDS 資料庫執行個體端點。資料庫執行個體現在已加密。	DBA，應用程式擁有者

相關資源

- 建立 AWS DMS 任務
- 使用 Amazon CloudWatch 監控複寫任務

- 監控 AWS DMS 任務
- 更新 Amazon RDS 加密金鑰

其他資訊

檢查來源 PostgreSQL 資料庫執行個體的加密：

此模式的其他備註：

- 將 `rds.logical_replication` 參數設定為 1，以在 PostgreSQL 上啟用複寫。

重要注意事項：複寫槽會保留預先寫入日誌 (WAL) 檔案，直到檔案在外部使用，例如 `pg_recvlogical`； 擷取、轉換和載入 (ETL) 任務；或 AWS DMS。當您將`rds.logical_replication`參數值設定為 1 時，AWS DMS 會設定 `wal_level`、`max_wal_senders`、 `max_replication_slots`和 `max_connections` 參數。如果邏輯複寫槽存在，但複寫槽保留的 WAL 檔案沒有取用者，您可能會看到交易日誌磁碟用量增加，以及可用儲存空間持續減少。如需詳細資訊和解決此問題的步驟，請參閱 AWS Support 知識中心的文章如何識別導致 Amazon RDS for PostgreSQL 上的「沒有剩餘空間」或「DiskFull」錯誤的原因。

- 您在建立資料庫快照後對來源資料庫執行個體所做的任何結構描述變更都不會出現在目標資料庫執行個體上。
- 建立加密的資料庫執行個體之後，您無法變更該資料庫執行個體所使用的 KMS 金鑰。在建立加密的資料庫執行個體之前，請務必判斷您的 KMS 金鑰需求。
- 您必須先在目標資料庫執行個體上停用觸發和外部金鑰，才能執行 AWS DMS 任務。您可以在任務完成時重新啟用這些項目。

在啟動時強制執行 Amazon RDS 資料庫的自動標記

Susanne Kangnoh 和 Archit Mathur，Amazon Web Services

總結

Amazon Relational Database Service (Amazon RDS) 是一種 Web 服務，可讓您更輕鬆地在 Amazon Web Services (AWS) 雲端中設定、操作和擴展關聯式資料庫。其能為產業標準的關聯式資料庫提供具成本效益、可調整大小的容量，並管理常見的資料庫管理任務。

您可以使用標記，以不同的方式分類您的 AWS 資源。當您帳戶中有許多資源，而且您想要根據標籤快速識別特定資源時，關聯式資料庫標記很有用。您可以使用 Amazon RDS 標籤將自訂中繼資料新增至 RDS 資料庫執行個體。標籤由使用者定義的索引鍵和值組成。我們建議您建立一組一致的標籤，以符合組織的需求。

此模式提供 AWS CloudFormation 範本，協助您監控和標記 RDS 資料庫執行個體。範本會建立監控 AWS CloudTrail CreateDBInstance 事件的 Amazon CloudWatch Events 事件。 CloudTrail CreateDBInstance (CloudTrail 會將 Amazon RDS 的 API 呼叫擷取為事件。) 當偵測到此事件時，它會呼叫 AWS Lambda 函數，自動套用您定義的標籤索引鍵和值。範本也會使用 Amazon Simple Notification Service (Amazon SNS) 傳送執行個體已標記的通知。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 用於上傳 Lambda 程式碼的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。
- 您想要接收標記通知的電子郵件地址。

限制

- 解決方案支援 CloudTrail CreateDBInstance 事件。它不會為任何其他事件建立通知。

Architecture

工作流程架構

自動化和擴展

- 您可以針對不同的 AWS 區域和帳戶多次使用 AWS CloudFormation 範本。您只需要在每個區域或帳戶中執行範本一次。

工具

AWS 服務

- AWS CloudTrail – AWS CloudTrail 是一種 AWS 服務，可協助您進行 AWS 帳戶的控管、合規以及營運和風險稽核。使用者、角色或 AWS 服務所執行的動作會在 CloudTrail 中記錄為事件。
- Amazon CloudWatch Events – Amazon CloudWatch Events 提供近乎即時的系統事件串流，說明 AWS 資源的變更。CloudWatch Events 會在操作變更發生時得知並在必要時採取修正動作，方法是傳送訊息以回應環境、啟用 函數、進行變更，以及擷取狀態資訊。
- AWS Lambda – AWS Lambda 是一種運算服務，支援執行程式碼，而不需要佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。只需為使用的運算時間支付費用，一旦未執行程式碼，就會停止計費。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種高度可擴展的物件儲存服務，可用於各種儲存解決方案，包括網站、行動應用程式、備份和資料湖。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一種 Web 服務，可讓應用程式、最終使用者和裝置立即從雲端傳送和接收通知。

Code

此模式包含兩個檔案的附件：

- `index.zip` 是一種壓縮檔案，其中包含此模式的 Lambda 程式碼。
- `rds.yaml` 是部署 Lambda 程式碼的 CloudFormation 範本。

如需如何使用這些檔案的資訊，請參閱 Epics 一節。

史詩

部署 Lambda 程式碼

任務	Description	所需的技能
將程式碼上傳至 S3 儲存貯體。	建立新的 S3 儲存貯體或使用現有的 S3 儲存貯體上傳	雲端架構師

任務	Description	所需的技能
	連接`index.zip` 的檔案 (Lambda 程式碼）。此儲存貯體必須與您要監控的資源 (RDS 資料庫執行個體） 位於相同的 AWS 區域。	
部署 CloudFormation 範本。	在與 S3 儲存貯體相同的 AWS 區域中開啟 Cloudformation 主控台，並部署附件中提供`rds.yaml`的檔案。在下一個 Epic 中，提供範本參數的值。	雲端架構師

完成 CloudFormation 範本中的參數

任務	Description	所需的技能
提供 S3 儲存貯體名稱。	輸入您在第一個 epic 中建立或選取的 S3 儲存貯體名稱。此 S3 儲存貯體包含 Lambda 程式碼的 .zip 檔案，且必須與 CloudFormation 範本和您要監控的 RDS 資料庫執行個體位於相同的 AWS 區域。	雲端架構師
提供 S3 金鑰。	在您的 S3 儲存貯體中提供 Lambda 程式碼 .zip 檔案的位置，而不帶正斜線 （例如 `index.zip` 或 `controls/index.zip`)。	雲端架構師
提供電子郵件地址。	提供您要接收違規通知的作用中電子郵件地址。	雲端架構師

任務	Description	所需的技能
指定記錄層級。	指定記錄層級和詳細程度。會`Info`指定應用程式進度的詳細資訊性訊息，並應僅用於偵錯。 會`Error`指定仍然可以允許應用程式繼續執行的錯誤事件。 會`Warning`指定潛在的有害情況。	雲端架構師
輸入 RDS 資料庫執行個體的標籤索引鍵和值。	輸入您要自動套用至 RDS 執行個體的必要標籤索引鍵和值。如需詳細資訊，請參閱 AWS 文件中的標記 Amazon RDS 資源。	雲端架構師

確認訂閱

任務	Description	所需的技能
確認電子郵件訂閱。	當 CloudFormation 範本成功部署時，它會傳送訂閱電子郵件訊息到您提供的電子郵件地址。若要在執行個體加上標籤時接收通知，您必須確認此電子郵件訂閱。	雲端架構師

相關資源

- 建立儲存貯體 (Amazon S3 文件）
- 標記 Amazon RDS 資源 (Amazon Aurora 文件）
- 上傳物件 (Amazon S3 文件）
- 使用 AWS CloudTrail 建立在 AWS API 呼叫上觸發的 CloudWatch Events 規則 CloudTrail (Amazon CloudWatch 文件）

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

預估 DynamoDB 資料表的隨需容量成本

Moinul Al-Mamun，Amazon Web Services

總結

Amazon DynamoDB 是 NoSQL 交易資料庫，即使在 PB 規模下也能提供單一位數毫秒延遲。此 Amazon Web Services (AWS) 無伺服器產品因其一致的效能和可擴展性而廣受歡迎。 您不需要佈建基礎基礎設施。您的單一資料表最多可增長至 PB。

使用隨需容量模式時，您需要為應用程式在資料表上執行的資料讀取和寫入支付每個請求的費用。AWS 費用是以每月累積的讀取請求單位 (RRUs) 和寫入請求單位 WRUs) 為基礎。DynamoDB 會在一個月內持續監控資料表的大小，以判斷您的儲存費用。它支援使用point-in-time-recovery(PITR) 進行連續備份。DynamoDB 會在一個月內持續監控已啟用 PITR 的資料表大小，以判斷您的備份費用。

若要預估專案的 DynamoDB 成本，請務必計算產品生命週期的不同階段會耗用多少 RRU、WRU 和儲存體。對於粗略成本估算，您可以使用 AWS 定價計算器，但必須為資料表提供大約數量 RRUs、WRUs和儲存需求。這些在專案開始時可能很難預估。AWS 定價計算器不會考慮資料成長率或項目大小，也不會分別考慮基底資料表和全域次要索引 (GSIs讀取和寫入次數。若要使用 AWS 定價計算器，您必須估計所有這些層面，以假設 WRU、RRU 和儲存體大小的 ballpark 數字，以取得您的成本估算。

此模式提供一種機制和可重複使用的 Microsoft Excel 範本，可針對隨需容量模式預估基本的 DynamoDB 成本因素，例如寫入、讀取、儲存、備份和復原成本。它比 AWS 定價計算器更精細，並獨立考慮基礎資料表和 GSIs要求。它也會考慮每月項目資料成長率，並預測三年的成本。

先決條件和限制

先決條件

- DynamoDB 和 DynamoDB 資料模型設計的基本知識
- DynamoDB 定價、WRU、RRU、儲存以及備份和復原的基本知識 （ 如需詳細資訊，請參閱隨需容量定價)
- 了解 DynamoDB 中的資料、資料模型和項目大小
- DynamoDB GSIs的知識

限制

- 範本提供您近似的計算，但不適用於所有組態。若要取得更準確的預估值，您必須測量基礎資料表和 GSIs 中每個項目的個別項目大小。
- 為了獲得更準確的預估，您必須考慮平均月份中每個項目的預期寫入次數（插入、更新和刪除）和讀取次數。
- 此模式支援根據固定的資料成長假設，估計未來幾年的寫入、讀取、儲存和備份和復原成本。

工具

AWS 服務

- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。

其他工具

- AWS 定價計算器是一種 Web 型規劃工具，可用來建立 AWS 使用案例的預估值。

最佳實務

為了協助降低成本，請考慮下列 DynamoDB 設計最佳實務。

- 分割區索引鍵設計 – 使用高基數分割區索引鍵來均勻分配負載。
- 相鄰清單設計模式 – 使用此設計模式來管理一對多和多對多關係。
- 稀鬆索引 – 對 GSI 使用稀鬆索引。在您建立 GSI 時，指定一個分割區索引鍵和 (選用) 一個排序索引鍵。只有在基本資料表中包含對應 GSI 分割區索引鍵的項目才會出現在稀疏索引中。這有助於保持 GSI 更小。
- 索引過載 – 使用相同的 GSI 對各種類型的項目編製索引。
- GSI 寫入碎片 – 明智地進行碎片以跨分割區分佈資料，以實現高效、更快的查詢。
- 大型項目 – 僅將中繼資料儲存在表內，將 Blob 儲存在 Amazon S3 中，並將參考保留在 DynamoDB 中。將大型項目分解為多個項目，並使用排序索引鍵有效率地編製索引。

如需更多設計最佳實務，請參閱《Amazon DynamoDB 開發人員指南》。

史詩

從 DynamoDB 資料模型擷取項目資訊

任務	Description	所需的技能
取得項目大小。	1. 檢查您要在資料表中存放多少不同類型的項目。 2. 若要以 KB 為單位計算每個項目的大小，請新增每個屬性的金鑰和值大小。 3. 計算基底資料表和每個 GSI 的項目大小。	資料工程師

預估寫入成本。	若要在隨需容量模式中預估寫入成本，您必須先測量一個月將耗用多少 WRUs。因此，您需要考慮下列因素：
	• 每月每個項目的建立、更新和刪除操作數量。
	• 可用的 GSIs 數目。獨立考慮每個索引。
	  • 索引項目的平均大小
	  • 索引上的同步時間數
	• 每月會在資料表中新增多少個新物件（例如元件或產品）？ 每月新增的實物數量可能不同，但您可以根據業務案例假設平均成長率。
	
	如需詳細資訊，請參閱其他資訊一節。
		資料工程師

預估讀取成本。	若要在隨需模式中預估讀取成本，您必須先測量一個月將耗	資料工程師、應用程式開發人員

任務	Description	所需的技能
	用多少 RRUs。因此，您需要考慮下列因素： • 可用的 GSIs 數目。獨立考慮每個索引。 • 索引項目的平均大小 • 每月每個產品的平均讀取次數。 • DynamoDB 資料表中的可用物件總數（元件或產品）。	

任務	Description	所需的技能
估算儲存體大小和成本。	首先，根據資料表中的項目大小估計每月平均儲存需求。然後將儲存體大小乘以 AWS 區域的每 GB 儲存體價格，以計算儲存體成本。 如果您已輸入用於估算寫入成本的資料，則不需要再次輸入它來計算儲存體大小。否則，若要估計儲存體大小，您需要考慮下列因素： • 根據您的資料表設計，模組（產品）中的資料項目數量。 • 平均項目大小，以 KB 為單位。 • 可用的 GSIs 數目。獨立考慮每個索引。 • 索引項目的平均大小 • 每月會在資料表中新增多少新產品？ 每月新產品的數量可能不同，但您可以根據業務案例來假設平均成長率。此範例每月平均使用 1，000 萬種新產品。	資料工程師

在 Excel 範本中輸入項目和物件資訊

任務	Description	所需的技能
從附件區段下載 Excel 範本，並針對您的使用案例資料表進行調整。	1. 下載 Excel 範本。	資料工程師

任務	Description	所需的技能
	2. 根據您的資料表設計調整業務模組和 GSIs。	
在 Excel 範本中輸入資訊。	1. 更新工作表中的項目資訊。僅更新橘色儲存格中的資料。 2. 調整物件號碼：每月可新增多少項目至資料表？ 3. 更新您 AWS 區域的 WRU 和 RRU 每百萬價格。 4. 更新您 AWS 區域的每月每 GB 儲存和備份價格。 5. 更新您 AWS 區域的每 GB 復原價格。 在範本中，有三個項目或實體：資訊、中繼資料和關係。有兩個 GSIs。針對您的使用案例，如果您需要更多項目，請建立新的資料列。如果您需要更多 GSIs，請複製現有的 GSI 區塊，然後貼上以視需要建立任意數量的 GSI 區塊。然後調整 SUM 和總計資料欄計算。	資料工程師

相關資源

參考

- Amazon DynamoDB 隨需容量定價
- DynamoDB 的 AWS 定價計算器
- 使用 DynamoDB 進行設計和架構的最佳實務
- DynamoDB 入門

指南和模式

- 使用 Amazon DynamoDB 建立資料模型
- Amazon DynamoDB 資料表的預估儲存成本

其他資訊

寫入成本計算範例

DynamoDB 資料模型設計顯示產品的三個項目，平均項目大小為 4 KB。當您將新產品新增至 DynamoDB 基礎資料表時，它會耗用項目數量 *（項目大小/1 KB 寫入單位）= 3 * (4/1) = 12 WRU。在此範例中，針對寫入 1 KB，產品會耗用 1 個 WRU。

讀取成本計算範例

若要取得 RRU 估算，請考慮每個項目一個月讀取的平均次數。例如，資訊項目將平均讀取一個月 10 次，中繼資料項目將讀取兩次，而關係項目將讀取五次。在範例範本中，所有元件的總 RRU = 每月建立的新元件數量 * 每個元件每月 RRU = 1，000 萬 * 17 RRU = 每月 1.700 萬 RRU。

每個月都會新增新事物（元件或產品將會新增，且產品總數將會隨著時間成長。因此，RRU 需求也會隨著時間增加。

- 第一個月的 RRU 使用量將為 1.7 億。
- 第二個月，RRU 使用量將為 2 * 1.7 億 = 3.4 億。
- 第三個月的 RRU 使用量將為 3 * 1.7 億 = 5.1 億。

下圖顯示每月 RRU 用量和成本預測。

請注意，圖表中的價格僅供說明之用。若要為您的使用案例建立準確的預測，請查看 AWS 定價頁面，並在 Excel 工作表中使用這些價格。

儲存、備份和復原成本計算範例

DynamoDB 儲存、備份和還原全部會彼此連線。備份會直接與儲存體連線，而復原則會直接與備份大小連線。隨著資料表大小的增加，對應的儲存、備份和還原成本將按比例增加。

儲存體大小和成本

儲存成本會根據您的資料成長率隨時間增加。例如，假設基礎資料表和 GSIs 中元件或產品的平均大小為 11 KB，每月會將 1，000 萬個新產品新增至您的資料庫資料表。在這種情況下，您的 DynamoDB 資料表大小將增長 (11 KB * 1000 萬) /1024/1024 = 每月 105 GB。在第一個月，您的資料表儲存大小將為 105 GB，第二個月將為 105 + 105 = 210 GBs，以此類推。

- 第一個月，您的 AWS 區域的儲存成本為每 GB 105 GB * 儲存價格。
- 第二個月的儲存成本將是您所在區域的每 GB 210 GB * 儲存價格。
- 第三個月的儲存成本將是您所在區域的每 GB 315 GB * 儲存價格。

如需未來三年的儲存大小和成本，請參閱儲存大小和預測一節。

備份成本

備份成本會根據您的資料成長率隨時間增加。當您使用point-in-time-recovery(PITR) 開啟連續備份時，連續備份費用是以每月平均儲存 GB 為基礎。在日曆月中，平均備份大小將與您的資料表儲存大小相同，但實際大小可能略有不同。隨著每個月都會新增新產品，總儲存大小和備份大小會隨著時間增加。例如，第一個月的平均備份大小 105 GB 可能會增加到第二個月的 210 GB。

- 對於您的 AWS 區域，第一個月的備份成本為每 GB 105 GB/月 * 連續備份價格。
- 第二個月的備份成本將是 210 GB-月 * 您區域每 GB 的連續備份價格。
- 第三個月的備份成本為 315 GB/月 * 您所在區域的每 GB 連續備份價格。
- 和 ，以此類推

備份成本包含在儲存體大小和成本預測區段的圖表中。

復原成本

當您在啟用 PITR 的情況下進行連續備份時，復原操作費用會根據還原的大小而定。每次還原時，您都會根據 GB 的還原資料付費。如果您的資料表大小很大，而且您在一個月內執行多次還原，則其成本很高。

為了預估還原成本，此範例假設您每個月在月底執行一次 PITR 復原。此範例使用每月平均備份大小作為該月的還原資料大小。對於第一個月，平均備份大小為 105 GB，對於月底的復原，還原資料大小為 105 GB。第二個月會是 210 GBs以此類推。

復原成本會根據您的資料成長率隨時間增加。

- 對於您的 AWS 區域，第一個月的復原成本將為每 GB 105 GB * 還原價格。

- 第二個月的復原成本將是 210 GB * 還原價格，每個 GB 的 區域。
- 第三個月的復原成本將為 315 GB * 還原價格，每個 GB 的 區域。

如需詳細資訊，請參閱 Excel 範本中的儲存、備份和復原索引標籤，以及下一節中的圖表。

儲存體大小和成本預測

在範本中，實際計費儲存體大小的計算方式是將標準資料表類別的每月免費方案減去 25 GB。在工作表中，您會看到預測圖表分為每月值。

下列範例圖表預測未來 36 個日曆月的每月儲存大小，包括 GB、計費儲存成本、隨需備份成本和復原成本。所有成本都是 USD。從圖表中，顯然儲存、備份和復原成本會隨著儲存大小的增加而按比例增加。

請注意，圖表中使用的價格僅供說明之用。若要為您的使用案例建立準確的價格，請查看 AWS 定價頁面，並在 Excel 範本中使用這些價格。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

Amazon DynamoDB 資料表的預估儲存成本

Moinul Al-Mamun，Amazon Web Services

總結

Amazon DynamoDB 是 NoSQL 交易資料庫，即使在 PB 規模下也能提供單一位數毫秒延遲。的這項熱門無伺服器產品 AWS 提供一致的效能和可擴展性。您不需要佈建儲存體，而且您的單一資料表最多可增長至 PB。

DynamoDB 會在一個月內持續監控資料表的大小，以判斷您的儲存費用。 AWS 然後， 會針對以 GB 為單位的平均儲存大小向您收費。隨著時間的推移，資料表的成長越多，儲存成本的成長就越多。若要計算儲存成本，您可以使用 AWS 定價計算器，但您需要提供資料表的大致大小，包括全域次要索引 (GSIs)，這在專案開始時很難估計。此外， AWS 定價計算器不會考慮資料成長率。

此模式提供一種機制和可重複使用的 Microsoft Excel 範本，用於計算 DynamoDB 儲存體大小和成本。它會獨立考慮基礎資料表和 GSIs的儲存需求。它透過考慮個別項目的大小和隨時間推移的資料成長率來計算儲存體大小。

若要取得預估值，請將兩個資訊插入範本：

- 基底資料表和 GSIs 的個別項目大小，以 KB 為單位
- 一個月平均可以新增多少個新物件或產品至資料表 （ 例如 1，000 萬個 ）

範本會產生未來三年的儲存體和成本預測圖表，如下列範例所示。

先決條件和限制

先決條件

- DynamoDB 的基本知識，包括 DynamoDB 儲存和定價
- 了解 DynamoDB 中的資料、資料模型和項目大小
- DynamoDB 全域次要索引 (GSIs的知識

限制

- 範本提供您近似的計算，但不適用於所有組態。若要取得更準確的預估值，您必須測量基礎資料表和 GSIs 中每個項目的個別項目大小。

- 此模式僅支援根據固定資料成長假設來估計未來幾年的儲存大小和成本。

工具

AWS 服務

- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。

其他工具

- AWS 定價計算器是一種 Web 型規劃工具，可用來建立 AWS 使用案例的預估值。

史詩

從 DynamoDB 資料模型擷取項目資訊

任務	Description	所需的技能
取得項目大小。	1. 決定您要在資料表中存放多少不同的項目類型。 2. 若要以 KB 為單位計算每個項目的大小，請新增每個屬性的 `Key`和 `Value`大小。 3. 計算基底資料表和每個 GSI 的項目大小。	資料工程師
取得一個月內新增的物件數量。	估計一個月平均會將多少個元件或物件新增至 DynamoDB 資料表。	資料工程師

在 Excel 範本中輸入項目和物件資訊

任務	Description	所需的技能
下載並調整 Excel 試算表。	1. 從附加的文件下載 Excel 範本。	資料工程師

任務	Description	所需的技能
	2. 根據您的資料表設計調整業務模組和 GSIs。	
在 Excel 範本中輸入資訊。	1. 更新工作表中的項目資訊。 2. 調整物件號碼：每月可新增多少項目至資料表？ 3. 更新您每月 GB 的儲存價格 AWS 區域。	資料工程師

相關資源

- Amazon DynamoDB 的隨需容量定價
- AWS DynamoDB 定價計算器

其他資訊

請注意，連接的範本只會預測標準儲存資料表類別的儲存大小和成本。根據儲存成本的預測，以及考量個別項目大小和產品或物件成長率，您可以預估下列項目：

- 資料匯出成本
- 備份和復原成本
- 資料儲存需求。

Amazon DynamoDB 資料儲存成本

DynamoDB 會持續監控資料表的大小，以判斷您的儲存費用。DynamoDB 會根據您啟用的功能，將資料的原始位元組大小與每個項目的儲存額外負荷相加，以計算可計費資料大小。如需詳細資訊，請參閱 DynamoDB 開發人員指南。

資料儲存的價格取決於您的資料表類別。如果您使用 DynamoDB Standard 資料表類別，則每個月存放的前 25 GB 是免費的。如需不同標準和標準不常存取資料表類別儲存成本的詳細資訊 AWS 區域，請參閱隨需容量定價。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWR 報告估計 Oracle 資料庫的 Amazon RDS 引擎大小

Abhishek Verma 和 Eduardo Valentim，Amazon Web Services

總結

當您將 Oracle 資料庫遷移至 Amazon Relational Database Service (Amazon RDS) 或 Amazon Aurora 時，計算目標資料庫的 CPU、記憶體和磁碟 I/O 是關鍵需求。您可以透過分析 Oracle 自動工作負載儲存庫 (AWR) 報告來估計目標資料庫所需的容量。此模式說明如何使用 AWR 報告來估計這些值。

來源 Oracle 資料庫可以是內部部署或在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上託管，也可以是 Amazon RDS for Oracle 資料庫執行個體。目標資料庫可以是任何 Amazon RDS 或 Aurora 資料庫。

> **Note**
>
> 如果您的目標資料庫引擎是 Oracle，容量預估會更精確。對於其他 Amazon RDS 資料庫，引擎大小可能會因資料庫架構的差異而有所不同。

我們建議您在遷移 Oracle 資料庫之前執行效能測試。

先決條件和限制

先決條件

- 下載 AWR 報告的 Oracle Database Enterprise Edition 授權和 Oracle Diagnostics Pack 授權。

產品版本

- 版本 11g （版本 11.2.0.3.v1 及更新版本） 和最高 12.2 和 18c、19c 的所有 Oracle 資料庫版本。
- 此模式不包含 Oracle Engineered Systems 或 Oracle Cloud Infrastructure (OCI)。

Architecture

來源技術堆疊

下列其中一項：

- 內部部署 Oracle 資料庫
- EC2 執行個體上的 Oracle 資料庫
- Amazon RDS for Oracle 資料庫執行個體

目標技術堆疊

- 任何 Amazon RDS 或 Amazon Aurora 資料庫

目標架構

如需完整遷移程序的資訊，請參閱使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫遷移至 Aurora PostgreSQL 的模式。

自動化和擴展

如果您有多個 Oracle 資料庫要遷移，而且想要使用其他效能指標，您可以依照部落格文章中所述的步驟，根據 Oracle 效能指標大規模調整 Amazon RDS 執行個體的大小，來自動化程序。

工具

- Oracle 自動工作負載儲存庫 (AWR) 是內建於 Oracle 資料庫的儲存庫。它會定期收集和存放系統活動和工作負載資料，然後由自動資料庫診斷監控 (ADDM) 進行分析。AWR 會定期擷取系統效能資料的快照（預設為每 60 分鐘），並儲存資訊（預設為最多 8 天）。 您可以使用 AWR 檢視和報告來分析此資料。

最佳實務

- 若要計算目標資料庫的資源需求，您可以使用單一 AWR 報告、多個 AWR 報告或動態 AWR 檢視。我們建議您在尖峰負載期間使用多個 AWR 報告來估計處理這些尖峰負載所需的資源。此外，動態檢視提供了更多資料點，可協助您更精確地計算資源需求。
- 您應該預估 IOPS 僅適用於您計劃遷移的資料庫，而不是使用磁碟的其他資料庫和程序。
- 若要計算資料庫正在使用多少 I/O，請勿使用 AWR 報告負載設定檔區段中的資訊。如果可用，請改用 I/O 設定檔區段，或跳至執行個體活動統計資料區段，並查看實體讀取和寫入操作的總值。
- 當您估計 CPU 使用率時，我們建議您使用資料庫指標方法，而不是作業系統 (OS) 統計資料，因為它是以資料庫使用的 CPU 為基礎。(OS 統計資料也包含其他程序的 CPU 用量。) 您也應該檢查 ADDM 報告中與 CPU 相關的建議，以改善遷移後的效能。

- 當您判斷正確的執行個體類型時，請考慮特定執行個體大小的 I/O 輸送量限制：Amazon Elastic Block Store (Amazon EBS) 輸送量和網路輸送量。
- 在遷移之前執行效能測試，以驗證引擎大小。

史詩

建立 AWR 報告

任務	Description	所需的技能
啟用 AWR 報告。	若要啟用報告，請遵循 Oracle 文件中的指示。	DBA
檢查保留期間。	若要檢查 AWR 報告的保留期間，請使用下列查詢。 `SQL> SELECT snap_interval,retention FROM dba_hist_wr_control;`	DBA
產生快照。	如果 AWR 快照間隔不夠精細，無法擷取尖峰工作負載的峰值，您可以手動產生 AWR 報告。若要產生手動 AWR 快照，請使用下列查詢。 `SQL> EXEC dbms_workload_repository.create_snapshot;`	DBA
檢查最近的快照。	若要檢查最近的 AWR 快照，請使用下列查詢。 `SQL> SELECT snap_id, to_char(begin_interval_time,'dd/MON/yy hh24:mi') Begin_Interval,`	DBA

任務	Description	所需的技能
	to_char(end_interv al_time,'dd/MON/yy hh24:mi') End_Interval FROM dba_hist_snapshot ORDER BY 1;	

估計磁碟 I/O 需求

任務	Description	所需的技能
選擇一個方法。	IOPS 是儲存裝置上每秒輸入和輸出操作的標準測量，包括讀取和寫入操作。 如果您要將現場部署資料庫遷移至 AWS，您需要判斷資料庫使用的尖峰磁碟 I/O。您可以使用下列方法來預估目標資料庫的磁碟 I/O： • AWR 報告的載入設定檔區段 • AWR 報告的執行個體活動統計資料區段（針對 Oracle 資料庫 12c 或更新版本使用此區段） • AWR 報告的 I/O 設定檔區段（針對 12c 之前的 Oracle 資料庫版本使用此區段） • AWR 檢視 下列步驟說明這四種方法。	DBA
選項 1：使用負載描述檔。	下表顯示 AWR 報告的 Load Profile 區段範例。	DBA

任務	Description	所需的技能
	Important 如需更準確的資訊，建議您使用選項 2 (I/O 設定檔）或選項 3（執行個體活動統計資料），而非負載設定檔。 每秒 / 每個交易 / 每個執行 / 每次呼叫 資料庫時間 (s)： 26.6 / 0.2 / 0.00 / 0.02 資料庫 CPU (s)： 18.0 / 0.1 / 0.00 / 0.01 背景 CPU (s)： 0.2 / 0.0 / 0.00 / 0.00 重做 2，4 / 17，.9	

任務	Description	所需的技能
	大小（位元組） 邏輯讀取（區塊） 3，3 23，.5 區塊變更： 21， 150.5 實體讀取（區塊） 13， 94.4 實體寫入（區塊） 3，4 24.1 讀取 IO 3，5 24.9	

任務	Description	所需的技能
	請求： 寫入 IO 請求： 574.1 4.0 讀取 IO (MB) 106.1 0.7 寫入 IO (MB) 27.1 0.2 IM 掃描資料列： 0.0 0.0 工作階段邏輯讀取即時	

任務	Description	所需的技能
	通訊： 使用者呼叫： 剖析 (SQL 硬剖析 (SQL SQL 工作區 (MB) 登入： 執行 (SQL 轉返： 交易：	

任務	Description	所需的技能
	根據此資訊，您可以計算 IOPs 和輸送量，如下所示： IOPS = 讀取 I/O 請求：+ 寫入 I/O 請求 = 3，586.8 + 574.7 = 4134.5 輸送量 = 實體讀取 （區塊） + 實體寫入 （區塊） = 13，575.1 + 3，467.3 = 17，042.4 由於 Oracle 中的區塊大小為 8 KB，因此您可以計算總輸送量，如下所示： MB 的總輸送量為 17042.4 * 8 * 1024 / 1024 / 1024 = 133.2 MB **Warning** 請勿使用負載描述檔來估計執行個體大小。它不如執行個體活動統計資料或 I/O 設定檔精確。	

任務	Description	所需的技能
選項 2：使用執行個體活動統計資料。	如果您使用的是 12c 之前的 Oracle 資料庫版本，則可以使用 AWR 報告的執行個體活動統計資料區段來估計 IOPS 和輸送量。下表顯示本節的範例。 統計數字 / 總計 / 每秒 / 每個交易 實體讀取總 IO 請求 / 2，54 ，217 / 3，61 / 25.11 實體讀取總位元組數 / 80，7 6，12 / 114，4 26.26 / 796，14 8 實體寫入總 IO 請求 / 534， 08 / 757.1 / 5.27 實體寫入總位元組數 / 25，5 8，84 / 36，1 1.84 / 251，50 8	DBA

任務	Description	所需的技能
	根據此資訊，您可以計算總 IOPS 和輸送量，如下所示： 總 IOPS = 3，610.28 + 757.11 = 4367 總 Mbps = 114，482，4 26.26 + 36，165，631.84 = 150648058.1 / 1024 / 1024 = 143 Mbps	

任務	Description	所需的技能
選項 3：使用 I/O 設定檔。	在 Oracle Database 12c 中，AWR 報告包含 I/O 設定檔區段，可在單一資料表中呈現所有資訊，並提供更準確的資料庫效能資料。下表顯示本節的範例。 		每秒讀寫數	每秒讀取數	每秒寫入數
---	---	---	---
請求總數：	4，36	3，61	757.1
資料庫請求：	4，16	3，58	574.7
最佳化請求：	0.0	0.0	0.0
重做請求：	179.3	2.8	176.6
總計 (MB)：	143.7	109.2	34.5
資料庫 (MB)：	133.1	106.1	27.1
最佳化總	0.0	0.0	0.0		DBA

任務	Description	所需的技能
	計 (MB)： 重做 7.6 2.7 4.9 (MB)： 資料 17，0 13，5 3，467. 庫 （區 塊）： 透過 5，89 5，36 537.6 緩衝 區快 取 （區 塊）： 直接 11，1 8，21 2，929. （區 塊）： 下表提供輸送量和總 IOPS 的 下列值： 輸送量 = 143 MBPS（從第五 列，標記為總計，第二欄） IOPS = 4，367.4（從第一 列，標記為請求總數，第二 欄）	

任務	Description	所需的技能
選項 4：使用 AWR 檢視。	您可以使用 AWR 檢視來查看相同的 IOPS 和輸送量資訊。若要取得此資訊，請使用下列查詢： `break on report compute sum of Value on report select METRIC_NAME,avg(AVERAGE) as "Value" from dba_hist_sysmetric_summary where METRIC_NAME in ('Physical Read Total IO Requests Per Sec','Physical Write Total IO Requests Per Sec') group by metric_name;`	DBA

估算 CPU 需求

任務	Description	所需的技能
選擇一個方法。	您可以透過三種方式估計目標資料庫所需的 CPU： • 透過使用處理器的實際可用核心 • 根據作業系統統計資料使用使用的核心 • 根據資料庫統計資料使用使用的核心	DBA

任務	Description	所需的技能
	如果您正在尋找已使用的核心，我們建議您使用資料庫指標方法，而不是作業系統統計資料，因為它是根據您計劃遷移的資料庫所使用的 CPU。(OS 統計資料也包含其他程序的 CPU 用量。) 您也應該檢查 ADDM 報告中與 CPU 相關的建議，以改善遷移後的效能。 您也可以根據 CPU 產生來預估需求。如果您使用的是不同的 CPU 世代，您可以依照白皮書中的說明來預估目標資料庫所需的 CPU 以獲得最佳工作負載效能的 vCPUs 數量。	

任務	Description	所需的技能

選項 1：根據可用核心估計需求。	在 AWR 報告中：
	• CPUs 是指邏輯和虛擬 CPUs。
	• 核心是實體 CPU 晶片組的處理器數量。
	• 插槽是將晶片連接到電路板的實體裝置。多核心處理器具有多個 CPU 核心的通訊端。
	
	您可以透過兩種方式預估可用的核心：
	• 使用作業系統命令
	• 使用 AWR 報告
	
	使用作業系統命令估計可用的核心
	
	使用以下命令計算處理器中的核心。
	`$ cat /proc/cpuinfo | grep "cpu cores"|uniq`
	`cpu cores : 4`
	`cat /proc/cpuinfo | egrep "core id|physical id" | tr -d "\n" | sed s/physical/\\nphysical/g | grep -v ^$ | sort | uniq | wc -l`
	
	使用下列命令計算處理器中的通訊端數量。
		DBA

任務	Description	所需的技能

	`grep "physical id" /proc/cpuinfo | sort -u`
	`physical id : 0`
	`physical id : 1`
	
	Note
	我們不建議使用 nmon 和 sar 等作業系統命令來擷取 CPU 使用率。這是因為這些計算包含其他程序的 CPU 使用率，可能無法反映資料庫使用的實際 CPU。
	
	使用 AWR 報告預估可用的核心
	
	您也可以從 AWR 報告的第一個區段衍生 CPU 使用率。以下是報告中的摘錄。
	
	資料庫 ID 執行個體 Inst num 啟動時間 發行版本 RA
	XX <DB XX> 1 205 Sep 0 0 日 12.1 0 NO	

任務	Description	所需的技能
	23：09 Hos 平台 CPU 核心 通訊端 記憶體 Nar (GB) (主機名稱) <ho Linu 80 80 2 441.7 e> x86 64 位 元 在此範例中，CPUs計數為 80，這表示這些是邏輯（虛擬）CPUs。您也可以看到此組態有兩個通訊端、每個通訊端上的一個實體處理器（總共兩個實體處理器），以及每個實體處理器或通訊端的 40 個核心。	

任務	Description	所需的技能
選項 2：使用作業系統統計資料估計 CPU 使用率。	您可以直接在作業系統（使用 sar 或其他主機作業系統公用程式）中檢查作業系統 CPU 用量統計資料，也可以從 AWR 報告的作業系統統計資料區段檢閱 IDLE/(IDLE+BUSY) 值。您可以查看直接從 v$osstat 消耗的 CPU 秒數。AWR 和 Statspack 報告也會在作業系統統計資料區段中顯示此資料。 如果同一個方塊中有多個資料庫，則它們對於 BUSY_TIME 都有相同的 v$osstat 值。 統計數字 / Value / 結束值 FREE_MEMORY_BYT / 6，810，，248 / 12，280，9，232 INACTIVE_MEMORY_BYTES / 175，627，33，632 / 160，380，53，568 SWAP_FREE_BYTES / 17，145，4，336 / 17，145，2，384 BUSY_TIME / 1，305，，937 IDLE_TIME / 4，312，，839 IOWAIT_TIME / 53，417，4	DBA

任務	Description	所需的技能
	NICE_TI 29，815 SYS_TIM 148，56 70 USER_T 1，146， ，783 LOAD 25 29 VM_IN_E 593，92 ES VM_OUT 327，68 TES PHYSIC/ 474，36 MEMOR 17，152 TES NUM_CF 80 NUM_CF 80 ORES NUM_CF 2 OCKETS GLOBAL 4，194， CEIVE_S E_MAX GLOBAL 2，097， ND_SIZE AX	

任務	Description	所需的技能
	TCP_RE 87，380 VE_SIZE EFAULT TCP_RE 6，291， VE_SIZE AX TCP_RE 4,096 VE_SIZE IN TCP_SE 16,384 SIZE_DE ULT TCP_SE 4，194， SIZE_MA TCP_SE 4,096 SIZE_MI 如果系統中沒有其他主要 CPU 取用者，請使用下列公式來計算 CPU 使用率的百分比： 使用率 = 忙碌時間/總時間 忙碌時間 = 需求 = v$osstat.BUSY_TIME C = 總時間 （忙碌 + 閒置） C = 容量 = v$ostat.BUSY_TIME + v$ostat.IDLE_TIME	

任務	Description	所需的技能
	使用率 = BUSY_TIME / (BUSY_TIME + IDLE_TIME) = -1，305，569，937 / (1，305，569，937 + 4，312，718，839) = 使用 23%	

任務	Description	所需的技能
選項 3：使用資料庫指標估計 CPU 使用率。	如果系統中有多個資料庫正在執行，您可以使用出現在報告開頭的資料庫指標。 快照 ID / 快照時間 / 工作階段 / 游標/工作階段 開始 Snap 1846 28-Sep-日 09：00：42 1226 35.8 結束快照： 1854 06-Oct-2 日 13：00：20 1876 41.1 已過： 11，（分鐘） 資料庫時間： 312 0（分鐘）	DBA

任務	Description	所需的技能
	若要取得 CPU 使用率指標，請使用此公式： 資料庫 CPU 用量 （ 可用 CPU 功率的 %) = CPU 時間 / NUM_CPUS / 經過時間 其中 CPU 用量是以 CPU 時間描述，並代表在 CPU 上花費的時間，而不是等待 CPU 的時間。此計算會導致： = 312，625.40 / 11，759.64 /80 = 正在使用 33% 的 CPU 核心數量 (33%) * 80 = 26.4 個核心 總核心 = 26.4 * (120%) = 31.68 個核心 您可以使用這兩個值中的較大值來計算 Amazon RDS 或 Aurora 資料庫執行個體的 CPU 使用率。 **Note** 在 IBM AIX 上，計算的使用率不符合作業系統或資料庫的值。這些值確實符合其他作業系統。	

估計記憶體需求

任務	Description	所需的技能
使用記憶體統計資料估計記憶體需求。	您可以使用 AWR 報告來計算來源資料庫的記憶體，並比對目標資料庫中的記憶體。您也應該檢查現有資料庫的效能，並減少記憶體需求以節省成本，或提高需求以改善效能。這需要詳細分析應用程式的 AWR 回應時間和服務層級協議 (SLA)。使用 Oracle 系統全域區域 (SGA) 和程式全域區域 (PGA) 用量的總和作為 Oracle 的估計記憶體使用率。為作業系統新增額外 20%，以判斷目標記憶體大小需求。對於 Oracle RAC，請使用所有 RAC 節點的估計記憶體使用率總和，並降低總記憶體，因為它存放在常見區塊上。 1. 檢查執行個體效率百分比資料表中的指標。表中使用下列用語： • Buffer Hit % 是在緩衝快取中找到特定區塊而非執行實體 I/O 的次數百分比。為了獲得更好的效能，請將目標設為 100%。 • 緩衝 Nowait % 應接近 100%。 • Latch Hit % 應接近 100%。	DBA

任務	Description	所需的技能
	• % 非稀疏 CPU 是用於非剖析活動的 CPU 時間百分比。此值應接近 100%。 執行個體效率百分比（目標 100%) 緩衝區 Nowa %：99.99 重做 NoWa %：100.00 緩衝區命中 %：99.84 記憶體內排序 %：100.00 Librar Hit %：748.7 軟剖析 %：99.81 執行以剖析 %：96.61 Latch Hit %：100.00 剖析 72.73 非稀 99.21	

任務	Description	所需的技能
	CPU 到剖析重疊 %： 疏 CPU %： Flash Cache Hit %： 0.00 在此範例中，所有指標看起來都沒問題，因此您可以將現有資料庫的 SGA 和 PGA 用作容量規劃需求。 2. 檢查記憶體統計資料區段並計算 SGA/PGA。 Begin 結束 主機記憶體 (MB)： 452，38 452，387 SGA 使用 (MB)： 220，54 220，544 PGA 使用 (MB)： 36，874 45，270.(	

任務	Description	所需的技能
	使用中的執行個體記憶體總數 = SGA + PGA = 220 GB + 45 GB = 265 GB 新增 20% 的緩衝區： 總執行個體記憶體 = 1.2 * 265 GB = 318 GB 由於 SGA 和 PGA 佔主機記憶體的 70%，因此總記憶體需求為： 主機記憶體總數 = 318/0.7 = 464 GB **Note** 當您遷移至 Amazon RDS for Oracle 時，PGA 和 SGA 會根據預先定義的公式預先計算。請確定預先計算的值接近您的預估值。	

判斷目標資料庫的資料庫執行個體類型

任務	Description	所需的技能
根據磁碟 I/O、CPU 和記憶體預估值判斷資料庫執行個體類型。	根據先前步驟中的預估值，目標 Amazon RDS 或 Aurora 資料庫的容量應為： • CPU 的 68 個核心 • 143 MBPS 的輸送量	DBA

任務	Description	所需的技能
	• 磁碟 I/O 的 4367 IOPS • 464 GB 的記憶體 在目標 Amazon RDS 或 Aurora 資料庫中，您可以將這些值對應至 db.r5.16xlarge 執行個體類型，其容量為 32 個核心、512 GB RAM 和 13，600 Mbps 的輸送量。如需詳細資訊，請參閱 AWS 部落格文章根據 Oracle 效能指標大規模調整大小的 Amazon RDS 執行個體。	

相關資源

- Aurora 資料庫執行個體類別 (Amazon Aurora 文件）
- Amazon RDS 資料庫執行個體儲存體 (Amazon RDS 文件）
- AWS Miner 工具 (GitHub 儲存庫）

使用 AWS DMS 將 Amazon RDS for SQL Server 資料表匯出至 S3 儲存貯體

Subhani Shaik，Amazon Web Services

總結

SQL Server 的 Amazon Relational Database Service (Amazon RDS) 不支援將資料載入 Amazon Web Services (AWS) 雲端上的其他資料庫引擎連結伺服器。反之，您可以使用 AWS Database Migration Service (AWS DMS) 將 Amazon RDS for SQL Server 資料表匯出至 Amazon Simple Storage Service (Amazon S3) 儲存貯體，其中資料可供其他資料庫引擎使用。

AWS DMS 可協助您快速且安全地將資料庫遷移至 AWS。來源資料庫在遷移期間保持完全運作，將依賴資料庫的應用程式停機時間降至最低。AWS DMS 可以在最廣泛使用的商業和開放原始碼資料庫之間遷移您的資料。

此模式會在設定 AWS DMS 端點時使用 AWS Secrets Manager。Secrets Manager 可協助您保護存取應用程式、服務和 IT 資源所需的秘密。您可以使用 服務在整個生命週期輪換、管理和擷取資料庫登入資料、API 金鑰和其他秘密。使用者和應用程式透過呼叫 Secrets Manager 來擷取秘密，減少對敏感資訊進行硬式編碼的需求。Secrets Manager 提供秘密輪換與 Amazon RDS、Amazon Redshift 和 Amazon DocumentDB 的內建整合。此外，此服務可延伸至其他類型的秘密，包括 API 金鑰和 OAuth 權杖。透過 Secrets Manager，您可以針對 AWS 雲端、第三方服務和內部部署中的資源，集中使用精細的許可和稽核秘密輪換來控制對秘密的存取。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- S3 儲存貯體
- 虛擬私有雲端 (VPC)
- 資料庫子網路
- Amazon RDS for SQL Server
- 代表 Amazon RDS 執行個體存取（列出、取得和放置物件）至 S3 儲存貯體的 AWS Identity and Access Management (IAM) 角色。
- Secrets Manager 存放 RDS 執行個體登入資料。

Architecture

技術堆疊

- Amazon RDS for SQL Server
- AWS DMS
- Amazon S3
- AWS Secrets Manager

目標架構

下圖顯示 AWS DMS 協助將資料從 Amazon RDS 執行個體匯入 S3 儲存貯體的架構。

1. 透過來源端點連線至來源 Amazon RDS 執行個體的 AWS DMS 遷移任務
2. 從來源 Amazon RDS 執行個體複製資料
3. 透過目標端點連線至目標 S3 儲存貯體的 AWS DMS 遷移任務
4. 以逗號分隔值 (CSV) 格式將複製的資料匯出至 S3 儲存貯體

工具

AWS 服務

- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Secrets Manager 可協助您以 API 呼叫 Secrets Manager，以程式設計方式擷取秘密，取代程式碼中的硬式編碼登入資料，包括密碼。

其他服務

- Microsoft SQL Server Management Studio (SSMS) 是一種用於管理 SQL Server 的工具，包括存取、設定和管理 SQL Server 元件。

史詩

設定 Amazon RDS for SQL Server 執行個體

任務	Description	所需的技能
建立 Amazon RDS for SQL Server 執行個體。	1. 開啟 AWS 管理主控台，選擇 RDS，然後使用標準建立選項來建立具有所需版本的 Amazon RDS 執行個體，例如 SQL Server Express Edition、SQL Server Standard Edition 或 SQL Server Enterprise Edition。針對 版本，選擇 2016 或更新版本。 2. 在範本下，選擇開發/測試。	DBA，DevOps 工程師
設定執行個體的登入資料。	1. 輸入執行個體的名稱。 2. 提供 Amazon RDS 執行個體的使用者名稱和密碼。	DBA，DevOps 工程師
設定執行個體類別、儲存體、自動擴展和可用性。	1. 從清單中選取資料庫執行個體類別：標準、記憶體最佳化和爆量類別。選擇資料庫執行個體類型，以配置為此資料庫執行個體規劃之工作負載所需的運算、網路和記憶體容量。如需詳細資訊，請參閱 AWS 文件。 2. 從清單中選取儲存類型：一般用途 SSD、佈建 IOPS SSD 或磁性。視需要配置預設儲存體大小。	DBA，DevOps 工程師

任務	Description	所需的技能
	3. 選擇啟用儲存體自動調整規模，以根據您的容量規劃增加 Amazon RDS 儲存體。 4. AWS DMS 支援具有複寫執行個體的異地同步備份部署。如果可用區域、內部硬體或網路發生中斷，AWS DMS 會建立待命執行個體，並透過自動容錯移轉至待命複本來提供高可用性 (HA)。根據匯入的大小，選取適當的選項。	
指定 VPC、子網路群組、公有存取和安全群組。	視需要選取 VPC、資料庫子網路群組和 VPC 安全群組，以建立 Amazon RDS 執行個體。遵循最佳實務，例如： • 請勿啟用 RDS 資料庫執行個體的公開存取。 • 請勿在安全群組中使用 CIDR 0.0.0.0/0。 • 僅使用必要的 IP 地址和連接埠詳細資訊來存取 RDS 執行個體。	DBA，DevOps 工程師
設定監控、備份和維護。	1. 指定您想要的備份選項。根據預設，自動備份會以 7 天的保留期間啟用。 2. 選擇適當的自動次要版本升級和維護時段設定，由 Amazon RDS 將待定修改或維護套用至資料庫。 3. 選擇建立資料庫。	DBA，DevOps 工程師

設定資料庫和範例資料

任務	Description	所需的技能
建立資料表並載入範例資料。	在新資料庫中，建立資料表。使用其他資訊區段中的範例程式碼，將資料載入資料表。	DBA，DevOps 工程師

設定登入資料

任務	Description	所需的技能
建立機密。	1. 在 主控台上，選擇 Secrets Manager，然後選擇儲存新的秘密。 2. 輸入 Amazon RDS for SQL Server 資料庫的使用者名稱和密碼。 此秘密將用於 AWS DMS 來源端點。	DBA，DevOps 工程師

設定資料庫與 S3 儲存貯體之間的存取權

任務	Description	所需的技能
建立 IAM 角色以存取 Amazon RDS。	1. 在 主控台上，選擇 IAM，然後建立 IAM 角色，讓 S3 儲存貯體讀取/寫入存取 Amazon RDS。 2. 在特徵下，選取 S3 整合。	DBA，DevOps 工程師

建立 S3 儲存貯體

任務	Description	所需的技能
建立 S3 儲存貯體。	若要從 Amazon RDS for SQL Server 儲存資料，請在主控台上選擇 S3，然後選擇建立儲存貯體。請確定 S3 儲存貯體未公開提供。	DBA，DevOps 工程師

設定 AWS DMS 和 S3 儲存貯體之間的存取權

任務	Description	所需的技能
為 AWS DMS 建立 IAM 角色以存取 Amazon S3。	建立 IAM 角色，允許 AWS DMS 從 S3 儲存貯體列出、取得和放置物件。	DBA，DevOps 工程師

設定 AWS DMS

任務	Description	所需的技能
建立 AWS DMS 來源端點。	1. 在 主控台上，選擇 Database Migration Service，然後選擇端點。建立來源端點，選取選取 RDS 資料庫執行個體核取方塊。 2. 針對來源引擎，選取 Microsoft SQL Server。 3. 在存取端點資料庫下，選擇 AWS Secrets Manager，然後輸入您先前建立的秘密和 IAM 角色，以及資料庫名稱。 4. 測試來源端點。	DBA，DevOps 工程師

任務	Description	所需的技能
建立 AWS DMS 目標端點。	建立目標端點，選取 Amazon S3 做為目標引擎。 提供您先前建立之 IAM 角色的 S3 儲存貯體名稱和資料夾名稱。	DBA，DevOps 工程師
建立 AWS DMS 複寫執行個體。	在相同的 VPC、子網路和安全群組中，建立 AWS DMS 複寫執行個體。如需選擇執行個體類別的詳細資訊，請參閱 AWS 文件。	DBA，DevOps 工程師
建立 AWS DMS 遷移任務。	若要將資料從 Amazon RDS for SQL Server 匯出到 S3 儲存貯體，請建立資料庫遷移任務。針對遷移類型，選擇遷移現有資料。選取您建立的 AWS DMS 端點和複寫執行個體。	DBA，DevOps 工程師

將資料匯出至 S3 儲存貯體

任務	Description	所需的技能
執行資料庫遷移任務。	若要匯出 SQL Server 資料表資料，請啟動資料庫遷移任務。任務將以 CSV 格式將資料從 Amazon RDS for SQL Server 匯出至 S3 儲存貯體。	DBA，DevOps 工程師

清除資源

任務	Description	所需的技能
刪除 資源。	若要避免產生額外費用，請使用 主控台以下列順序刪除資源： 1. 遷移任務 2. Replication instance (複寫執行個體) 3. 端點 4. S3 bucket (S3 儲存貯體) 5. 資料庫執行個體	DBA，DevOps 工程師

相關資源

- AWS DMS
- Amazon S3
- Amazon RDS for SQL Server
- Amazon S3 整合

其他資訊

若要建立資料庫和資料表，以及載入範例資料，請使用下列程式碼。

```
--Step1: Database creation in RDS SQL Server
CREATE DATABASE [Test_DB]
 ON  PRIMARY
( NAME = N'Test_DB', FILENAME = N'D:\rdsdbdata\DATA\Test_DB.mdf' , SIZE = 5120KB ,
 FILEGROWTH = 10%)
 LOG ON
( NAME = N'Test_DB_log', FILENAME = N'D:\rdsdbdata\DATA\Test_DB_log.ldf' , SIZE =
 1024KB , FILEGROWTH = 10%)
GO

--Step2: Create Table
```

```
USE Test_DB
GO
Create Table Test_Table(ID int, Company Varchar(30), Location Varchar(20))

--Step3: Load sample data.
USE Test_DB
GO
Insert into Test_Table values(1,'AnyCompany','India')
Insert into Test_Table values(2,'AnyCompany','USA')
Insert into Test_Table values(3,'AnyCompany','UK')
Insert into Test_Table values(4,'AnyCompany','Hyderabad')
Insert into Test_Table values(5,'AnyCompany','Banglore')
```

在 Aurora PostgreSQL 中處理動態 SQL 陳述式中的匿名區塊

anuradha chintha，Amazon Web Services

總結

注意：Amazon Cloud Directory 不再開放給新客戶。如需 Cloud Directory 的替代方案，請探索 Amazon DynamoDB 和 Amazon Neptune。如果您需要協助為您的使用案例選擇正確的替代方案，或有其他問題，請聯絡 AWS 支援。

此模式說明如何避免在動態 SQL 陳述式中處理匿名區塊時遇到的錯誤。當您使用 AWS Schema Conversion Tool 將 Oracle 資料庫轉換為 Aurora PostgreSQL 相容版本資料庫時，會收到錯誤訊息。為了避免錯誤，您必須知道0UT綁定變數的值，但在執行 SQL 陳述式之前，您無法知道0UT綁定變數的值。AWS Schema Conversion Tool (AWS SCT) 無法了解動態 SQL 陳述式中的邏輯所產生的錯誤。AWS SCT 無法轉換 PL/SQL 程式碼中的動態 SQL 陳述式 （即函數、程序和套件）。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- Aurora PostgreSQL 資料庫 （ 資料庫 ） 執行個體
- Oracle 資料庫執行個體的 Amazon Relational Database Service (Amazon RDS)
- PostgreSQLinteractive 終端機 (psql)
- SQL *Plus
- AWS_ORACLE_EXT 目標資料庫中的結構描述 (AWS SCT 延伸套件的一部分)
- 最新版 AWS Schema Conversion Tool (AWS SCT) 及其必要的驅動程式

Architecture

來源技術堆疊

- 內部部署 Oracle 資料庫 10g 及更新版本

目標技術堆疊

- Amazon Aurora PostgreSQL

- Amazon RDS for PostgreSQL
- AWS Schema Conversion Tool (AWS SCT)

遷移架構

下圖顯示如何使用 AWS SCT 和 Oracle 0UT繫結變數來掃描應用程式程式碼以取得內嵌 SQL 陳述式，並將程式碼轉換為 Aurora 資料庫可以使用的相容格式。

該圖顯示以下工作流程：

1. 使用 Aurora PostgreSQL 做為目標資料庫，為來源資料庫產生 AWS SCT 報告。
2. 識別動態 SQL 程式碼區塊中的匿名區塊 (AWS SCT 為此引發錯誤）。
3. 手動轉換程式碼區塊，並在目標資料庫上部署程式碼。

工具

AWS 服務

- Amazon Aurora PostgreSQL 相容版本是完全受管的 ACID 相容關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。
- Amazon Relational Database Service (Amazon RDS) for Oracle 可協助您在 AWS 雲端中設定、操作和擴展 Oracle 關聯式資料庫。
- AWS Schema Conversion Tool (AWS SCT) 可將來源資料庫結構描述和大部分資料庫程式碼物件自動轉換為與目標資料庫相容的格式，協助您預測異質資料庫遷移。

其他工具

- pgAdmin 可讓您連線至資料庫伺服器並與之互動。
- Oracle SQL Developer 是一種整合的開發環境，可用來開發和管理 Oracle Database 中的資料庫。您可以針對此模式使用 SQL *Plus 或 Oracle SQL Developer。

史詩

設定 Oracle 來源資料庫

任務	Description	所需的技能
在 Amazon RDS 或 Amazon EC2 上建立 Oracle 執行個體。	若要在 Amazon RDS 上建立 Oracle 資料庫執行個體，請參閱《Amazon RDS 文件》中的建立 Oracle 資料庫執行個體並連線至 Oracle 資料庫執行個體上的資料庫。 若要在 Amazon Elastic Compute Cloud (Amazon EC2) 上建立 Oracle 資料庫執行個體，請參閱 AWS 規範指引文件中的 Amazon EC2 for Oracle。	DBA
建立資料庫結構描述和物件以進行遷移。	您可以使用 Amazon Cloud Directory 來建立資料庫結構描述。如需詳細資訊，請參閱 Cloud Directory 文件中的建立結構描述。	DBA
設定傳入和傳出安全群組。	若要建立和設定安全群組，請參閱 Amazon RDS 文件中的使用安全群組控制存取。	DBA
確認資料庫正在執行。	若要檢查資料庫的狀態，請參閱 Amazon RDS 文件中的檢視 Amazon RDS 事件。	DBA

設定目標 Aurora PostgreSQL 資料庫

任務	Description	所需的技能
在 Amazon RDS 中建立 Aurora PostgreSQL 執行個體。	若要建立 Aurora PostgreSQL 執行個體，請參閱 Amazon RDS 文件中的建立資料庫叢集並連線至 Aurora PostgreSQL 資料庫叢集上的資料庫。	DBA
設定傳入和傳出安全群組。	若要建立和設定安全群組，請參閱 Aurora 文件中的透過建立安全群組來提供 VPC 中資料庫叢集的存取權。	DBA
確認 Aurora PostgreSQL 資料庫正在執行。	若要檢查資料庫的狀態，請參閱 Aurora 文件中的檢視 Amazon RDS 事件。	DBA

設定 AWS SCT

任務	Description	所需的技能
將 AWS SCT 連線至來源資料庫。	若要將 AWS SCT 連線至來源資料庫，請參閱 AWS SCT 文件中的連線至 PostgreSQL 做為來源。	DBA
將 AWS SCT 連線至目標資料庫。	若要將 AWS SCT 連線至目標資料庫，請參閱《AWS Schema Conversion Tool 使用者指南》中的什麼是 AWS Schema Conversion Tool？。Schema Conversion Tool	DBA

任務	Description	所需的技能
在 AWS SCT 中轉換資料庫結構描述，並將自動轉換的程式碼儲存為 SQL 檔案。	若要儲存 AWS SCT 轉換的檔案，請參閱《AWS 結構描述轉換工具使用者指南》中的在 AWS SCT 中儲存和套用轉換Schema Conversion Tool。	DBA

遷移程式碼

任務	Description	所需的技能
取得 SQL 檔案以進行手動轉換。	在 AWS SCT 轉換檔案中，提取需要手動轉換的 SQL 檔案。	DBA
更新指令碼。	手動更新 SQL 檔案。	DBA

相關資源

- Amazon RDS
- Amazon Aurora 功能

其他資訊

下列範例程式碼示範如何設定 Oracle 來源資料庫：

```
CREATE or replace PROCEDURE calc_stats_new1 (
  a NUMBER,
  b NUMBER,
  result out NUMBER)
IS
BEGIN
result:=a+b;
END;
/
```

```
set serveroutput on ;
```

```
DECLARE
  a NUMBER := 4;
  b NUMBER := 7;
  plsql_block VARCHAR2(100);
  output number;
BEGIN
  plsql_block := 'BEGIN calc_stats_new1(:a, :b,:output); END;';
  EXECUTE IMMEDIATE plsql_block USING a, b,out output;
  DBMS_OUTPUT.PUT_LINE('output:'||output);

END;
```

下列範例程式碼示範如何設定目標 Aurora PostgreSQL 資料庫：

```
 w integer,
 x integer)
RETURNS integer
AS
$BODY$
DECLARE
begin
return w + x ;
end;
$BODY$
LANGUAGE  plpgsql;


CREATE OR REPLACE FUNCTION test_pg.init()
RETURNS void
AS
$BODY$
BEGIN
if aws_oracle_ext.is_package_initialized
      ('test_pg' ) then
      return;
    end if;
    perform aws_oracle_ext.set_package_initialized
      ('test_pg' );

PERFORM aws_oracle_ext.set_package_variable('test_pg', 'v_output', NULL::INTEGER);
PERFORM aws_oracle_ext.set_package_variable('test_pg', 'v_status', NULL::text);
END;
```

```
$BODY$
LANGUAGE  plpgsql;


DO $$
declare
v_sql text;
v_output_loc int;
a integer :=1;
b integer :=2;
BEGIN
perform  test_pg.init();
--raise notice 'v_sql %',v_sql;
execute 'do $a$ declare v_output_l int; begin select * from test_pg.calc_stats_new1('||
a||','||b||') into v_output_l;
PERFORM aws_oracle_ext.set_package_variable(''test_pg'', ''v_output'', v_output_l) ;
 end; $a$'  ;
v_output_loc := aws_oracle_ext.get_package_variable('test_pg', 'v_output');
raise notice 'v_output_loc %',v_output_loc;
END ;
$$
```

協助強制執行 DynamoDB 標記

Mansi Suratwala，Amazon Web Services

總結

此模式會在預先定義的 Amazon DynamoDB 標籤遺失或從 Amazon Web Services (AWS) 雲端上的 DynamoDB 資源中移除時設定自動通知。

DynamoDB 是全受管的 NoSQL 資料庫服務，可提供快速且可預測的效能與可擴展性。DynamoDB 可讓您卸載操作和擴展分散式資料庫的管理負擔。當您使用 DynamoDB 時，不必擔心硬體佈建、設定和組態、複寫、軟體修補或叢集擴展。

模式使用 AWS CloudFormation 範本，這會建立 Amazon CloudWatch Events 事件和 AWS Lambda 函數。事件會使用 AWS CloudTrail 來監控任何新的或現有的 DynamoDB 標記資訊。如果遺失或移除預先定義的標籤，CloudWatch 會觸發 Lambda 函數，這會傳送 Amazon Simple Notification Service (Amazon SNS) 通知給您，通知您違規。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Lambda .zip 檔案的 Amazon Simple Storage Service (Amazon S3) 儲存貯體，其中包含執行 Lambda 函數的 Python 指令碼

限制

- 解決方案只有在 `TagResource`或 `UntagResource` CloudTrail 事件發生時才有效。它不會為任何其他事件建立通知。

Architecture

目標技術堆疊

- Amazon DynamoDB
- AWS CloudTrail
- Amazon CloudWatch

- AWS Lambda
- Amazon S3
- Amazon SNS

目標架構

自動化和擴展

您可以針對不同的 AWS 區域和帳戶多次使用 AWS CloudFormation 範本。您只需要在每個區域或帳戶中執行範本一次。

工具

工具

- Amazon DynamoDB – DynamoDB 是全受管的 NoSQL 資料庫服務，可提供快速且可預測的效能與可擴展性。
- AWS CloudTrail – CloudTrail 是一種 AWS 服務，可協助您進行 AWS 帳戶的控管、合規以及操作和風險稽核。使用者、角色或 AWS 服務所執行的動作會在 CloudTrail 中記錄為事件。
- Amazon CloudWatch Events – Amazon CloudWatch Events 提供近乎即時的系統事件串流，說明 AWS 資源的變更。
- AWS Lambda – Lambda 是一種運算服務，支援執行程式碼，而不需要佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種高度可擴展的物件儲存服務，可用於各種儲存解決方案，包括網站、行動應用程式、備份和資料湖。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一種 Web 服務，可讓應用程式、最終使用者和裝置立即從雲端傳送和接收通知。

Code

- 專案的 .zip 檔案可做為附件使用。

史詩

定義 S3 儲存貯體

任務	Description	所需的技能
定義 S3 儲存貯體。	在 Amazon S3 主控台上，選擇或建立具有不包含正斜線之唯一名稱的 S3 儲存貯體。此 S3 儲存貯體將託管 Lambda 程式碼 .zip 檔案。您的 S3 儲存貯體必須與受監控的 DynamoDB 資源位於相同的 AWS 區域。	雲端架構師

將 Lambda 程式碼上傳至 S3 儲存貯體

任務	Description	所需的技能
將 Lambda 程式碼上傳至 S3 儲存貯體。	將附件區段中提供的 Lambda 程式碼 .zip 檔案上傳至 S3 儲存貯體。S3 儲存貯體必須與受監控的 DynamoDB 資源位於相同的區域。	雲端架構師

部署 AWS CloudFormation 範本

任務	Description	所需的技能
部署 AWS CloudFormation 範本。	在 AWS CloudFormation 主控台上，部署附件區段中提供的 AWS CloudFormation 範本。在下一個史詩中，提供參數的值。	雲端架構師

完成 AWS CloudFormation 範本中的參數

任務	Description	所需的技能
命名 S3 儲存貯體。	輸入您在第一個史詩中建立或選擇的 S3 儲存貯體名稱。	雲端架構師
提供 Amazon S3 金鑰。	提供 Lambda 程式碼 .zip 檔案在 S3 儲存貯體中的位置，不帶正斜線（例如 `<folder>/<file-name>.zip` ）。	雲端架構師
提供電子郵件地址	提供作用中的電子郵件地址以接收 Amazon SNS 通知。	雲端架構師
定義記錄層級。	定義 Lambda 函數的記錄層級和頻率。 會`Info`指定應用程式進度的詳細資訊訊息。會`Error`指定仍可允許應用程式繼續執行的錯誤事件。會`Warning`指定可能有害的情況。	雲端架構師
輸入所需的 DynamoDB 標籤金鑰。	請確定標籤以逗號分隔，中間沒有空格（例如 `ApplicationId,CreatedBy,Environment,Organization` ）。CloudWatch Events 事件會搜尋這些標籤，並在找不到它們時傳送通知。	雲端架構師

確認訂閱。

任務	Description	所需的技能
確認訂閱。	當範本成功部署時，它會傳送訂閱電子郵件到您提供的電	雲端架構師

任務	Description	所需的技能
	子郵件地址。若要接收違規通知，您必須確認此電子郵件訂閱。	

相關資源

- 建立 S3 儲存貯體
- 將檔案上傳至 S3 儲存貯體
- DynamoDB 中的標記資源
- 使用 AWS CloudTrail 建立在 AWS API 呼叫上觸發的 CloudWatch Events 規則 CloudTrail

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS DMS 和 Amazon Aurora 實作跨區域災難復原

Mark Hudson，Amazon Web Services

總結

自然或人為造成的災難可能隨時發生，並可能影響在特定 AWS 區域中執行之服務和工作負載的可用性。若要降低風險，您必須開發災難復原 (DR) 計畫，其中包含 AWS 服務的內建跨區域功能。對於本質上不提供跨區域功能的 AWS 服務，DR 計畫也必須提供解決方案來處理跨 AWS 區域的容錯移轉。

此模式會引導您完成災難復原設定，其中涉及單一區域中的兩個 Amazon Aurora MySQL 相容版本資料庫叢集。為了符合 DR 需求，資料庫叢集會設定為使用 Amazon Aurora 全域資料庫功能，而單一資料庫跨越多個 AWS 區域。AWS Database Migration Service (AWS DMS) 任務會在本機區域中的叢集之間複寫資料。不過，AWS DMS 目前不支援區域之間的任務容錯移轉。此模式包含解決該限制和在兩個區域中獨立設定 AWS DMS 所需的步驟。

先決條件和限制

先決條件

- 選取支援 Amazon Aurora 全域資料庫的主要和次要 AWS 區域。
- 主要區域中單一帳戶中的兩個獨立 Amazon Aurora MySQL 相容版本資料庫叢集。
- 資料庫執行個體類別 db.r5 或更新版本 （ 建議 ）。
- 主要區域中的 AWS DMS 任務，在現有資料庫叢集之間執行持續複寫。
- 已備妥 DR 區域資源，以符合建立資料庫執行個體的需求。如需詳細資訊，請參閱在 VPC 中使用資料庫執行個體。

限制

- 如需 Amazon Aurora 全域資料庫限制的完整清單，請參閱 Amazon Aurora 全域資料庫的限制。

產品版本

- Amazon Aurora MySQL 相容版本 5.7 或 8.0。如需詳細資訊，請參閱Amazon Aurora 版本。

Architecture

目標技術堆疊

- Amazon Aurora MySQL 相容版本全域資料庫叢集
- AWS DMS

目標架構

下圖顯示兩個 AWS 區域的全域資料庫，一個具有主要主資料庫和報告程式資料庫和 AWS DMS 複寫，另一個具有次要主資料庫和報告程式資料庫。

自動化和擴展

您可以使用 AWS CloudFormation 在次要區域中建立先決條件基礎設施，例如虛擬私有雲端 (VPC)、子網路和參數群組。您也可以使用 AWS CloudFormation 在 DR 區域中建立次要叢集，並將其新增至全域資料庫。如果您使用 CloudFormation 範本在主要區域中建立資料庫叢集，您可以使用其他範本更新或增強這些叢集，以建立全域資料庫資源。如需詳細資訊，請參閱使用兩個資料庫執行個體建立 Amazon Aurora 資料庫叢集和為 Aurora MySQL 建立全域資料庫叢集。

最後，您可以在容錯移轉和容錯回復事件發生後，使用 CloudFormation 在主要和次要區域中建立 AWS DMS 任務。如需詳細資訊，請參閱AWS::DMS::ReplicationTask。

工具

- Amazon Aurora 是全受管關聯式資料庫引擎，與 MySQL 和 PostgreSQL 相容。此模式使用 Amazon Aurora MySQL 相容版本。
- Amazon Aurora 全域資料庫專為全域分散式應用程式而設計。單一 Amazon Aurora 全域資料庫可以跨越多個 AWS 區域。它會複寫您的資料，而不會影響資料庫效能。它還在每個區域中啟用具有低延遲的快速本機讀取，並從整個區域的中斷提供災難復原。
- AWS DMS 提供一次性遷移或持續複寫。持續複寫任務可讓您的來源和目標資料庫保持同步。設定之後，進行中的複寫任務會持續將來源變更套用至目標，並將延遲降至最低。資料驗證和轉換等所有 AWS DMS 功能都可用於任何複寫任務。

史詩

準備主要區域中現有的資料庫叢集

任務	Description	所需的技能
修改資料庫叢集參數群組。	在現有的資料庫叢集參數群組中，將 `binlog_format` 參數設定為資料列的值，以啟用資料列層級二進位記錄。 執行持續複寫或變更資料擷取 (CDC) 時，AWS DMS 需要 MYSQL 相容資料庫的資料列層級二進位記錄。如需詳細資訊，請參閱使用 AWS 受管 MySQL 相容資料庫做為 AWS DMS 的來源。	AWS 管理員

更新資料庫二進位日誌保留期。	使用安裝在最終使用者裝置或 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的 MySQL 用戶端，在主要資料庫叢集的寫入器節點上執行 Amazon Relational Database Service (Amazon RDS) 提供的下列預存程序，其中 XX是保留日誌的時數。
	
	`call mysql.rds_set_configuration('binlog retention hours', XX)`
	
	執行下列命令以確認設定。
	
	`call mysql.rds_show_configuration;`
		DBA

任務	Description	所需的技能
	AWS 管理的 MySQL 相容資料庫會盡快清除二進位日誌。因此，保留期間必須足夠長，以確保日誌不會在 AWS DMS 任務執行之前清除。值為 24 小時通常就已足夠，但值應以在 DR 區域中設定 AWS DMS 任務所需的時間為基礎。	

更新主要區域中現有的 AWS DMS 任務

任務	Description	所需的技能

記錄 AWS DMS 任務 ARN。	使用 Amazon Resource Name (ARN) 取得 AWS DMS 任務名稱以供日後使用。若要擷取 AWS DMS 任務 ARN，請在主控台中檢視任務或執行下列命令。
	`aws dms describe-replication-tasks`
	ARN 如下所示。
	`arn:aws:dms:us-east-1:<accountid>:task:AN6HFFMPM246XOZVEUHCNSOVF7MQCLTOZUIRAMY`
	最後一個冒號後面的字元對應於後續步驟中使用的任務名稱。
		AWS 管理員

任務	Description	所需的技能
修改現有的 AWS DMS 任務以記錄檢查點。	AWS DMS 會建立包含資訊的檢查點，以便複寫引擎知道變更串流的復原點。若要記錄檢查點資訊，請在 主控台中執行下列步驟： 1. 停止 AWS DMS 任務。 2. 使用任務中的 JSON 編輯器，將 TaskRecoveryTableEnabled 參數設定為 true。 3. 啟動 AWS DMS 任務。	AWS 管理員
驗證檢查點資訊。	使用連線至叢集寫入器端點的 MySQL 用戶端，查詢報告器資料庫叢集中的新中繼資料表，以確認其存在並包含複寫狀態資訊。執行下列命令。 `select * from awsdms_control.awsdms_txn_state;` 來自 ARN 的任務名稱應該在 `Task_Name` 欄中的此表格中找到。	DBA

將兩個 Amazon Aurora 叢集展開至 DR 區域

任務	Description	所需的技能
在 DR 區域中建立基礎基礎設施。	建立和存取 Amazon Aurora 叢集所需的基本元件： • 虛擬私有雲端 (VPC)	AWS 管理員

任務	Description	所需的技能
	• 子網路 • 安全群組 • 網路存取控制清單 • 子網路群組 • DB parameter group (資料庫參數群組) • DB cluster parameter group (資料庫叢集參數群組) 確定兩個參數群組的組態都符合主要區域中的組態。	
將 DR 區域新增至兩個 Amazon Aurora 叢集。	將次要區域 (DR 區域） 新增至主要和報告者 Amazon Aurora 叢集。如需詳細資訊，請參閱將 AWS 區域新增至 Amazon Aurora 全域資料庫。	AWS 管理員

執行容錯移轉

任務	Description	所需的技能
停止 AWS DMS 任務。	容錯移轉發生後，主要區域中的 AWS DMS 任務將無法正常運作，且應停止以避免錯誤。	AWS 管理員
執行受管容錯移轉。	執行主資料庫叢集的受管容錯移轉至 DR 區域。如需說明，請參閱執行 Amazon Aurora 全域資料庫的受管計劃容錯移轉。主要資料庫叢集上的容錯移轉完成後，請在報告者資料庫叢集上執行相同的活動。	AWS 管理員，DBA

任務	Description	所需的技能
將資料載入主資料庫。	將測試資料插入 DR 資料庫叢集中主要資料庫的寫入器節點。此資料將用於驗證複寫是否正常運作。	DBA
建立 AWS DMS 複寫執行個體。	若要在 DR 區域中建立 AWS DMS 複寫執行個體，請參閱建立複寫執行個體。	AWS 管理員，DBA
建立 AWS DMS 來源和目標端點。	若要在 DR 區域中建立 AWS DMS 來源和目標端點，請參閱建立來源和目標端點。來源應指向主要資料庫叢集的寫入器執行個體。目標應指向報告者資料庫叢集的寫入器執行個體。	AWS 管理員，DBA
取得複寫檢查點。	若要取得複寫檢查點，請使用 MySQL 用戶端，針對 DR 區域中報告者資料庫叢集中的寫入器節點執行下列動作來查詢中繼資料表。 `select * from awsdms_control.awsdms_txn_state;` 在表格中，尋找對應於 AWS DMS 任務 ARN 的 task_name 值，該 ARN 存在於您在第二個特徵中取得的主要區域中。	DBA

任務	Description	所需的技能
建立 AWS DMS 任務。	使用 主控台，在 DR 區域中建立 AWS DMS 任務。在任務中，指定僅複寫資料變更的遷移方法。如需詳細資訊，請參閱建立任務。 1. 在任務設定中，使用精靈指定下列項目： • 來源交易的 CDC 啟動模式 – 啟用自訂 CDC 啟動模式 • 來源交易的自訂 CDC 起點 – 指定復原檢查點 2. 在復原檢查點方塊中，輸入先前透過`awsdms_txn_state` 資料表上的資料庫查詢取得的複寫檢查點值。 3. 在任務設定區段中，選取 JSON 編輯器，並將 TaskRecoveryTableEnabled 參數設為 true。 將 AWS DMS 任務開始遷移任務設定為建立時自動。	AWS 管理員，DBA
記錄 AWS DMS 任務 ARN。	使用 ARN 取得 AWS DMS 任務名稱以供日後使用。若要擷取 AWS DMS 任務 ARN，請執行下列命令。 `aws dms describe-replication-tasks`	AWS 管理員，DBA

任務	Description	所需的技能
驗證複寫的資料。	查詢 DR 區域中的報告程式資料庫叢集，確認您載入主資料庫叢集的測試資料已複寫。	DBA

執行容錯回復

任務	Description	所需的技能
停止 AWS DMS 任務。	發生容錯移轉後，DR 區域中的 AWS DMS 任務將無法正常運作，且應停止以避免錯誤。	AWS 管理員
執行受管容錯回復。	將主要資料庫叢集容錯移轉回主要區域。如需說明，請參閱執行 Amazon Aurora 全域資料庫的受管計劃容錯移轉。主資料庫叢集上的容錯回復完成後，請在報告者資料庫叢集上執行相同的活動。	AWS 管理員，DBA
取得複寫檢查點。	若要取得複寫檢查點，請使用 MySQL 用戶端，針對 DR 區域中報告者資料庫叢集中的寫入器節點執行下列動作來查詢中繼資料表。 `select * from awsdms_control.awsdms_txn_state;` 在表格中，尋找對應到 AWS DMS 任務 ARN `task_name` 的值，該 ARN 存在於您在第四個 epic 中取得的 DR 區域中。	DBA

任務	Description	所需的技能
更新 AWS DMS 來源和目標端點。	資料庫叢集故障後，請檢查主要區域中的叢集，以判斷哪些節點是寫入器執行個體。然後，確認主要區域中現有的 AWS DMS 來源和目標端點指向寫入器執行個體。如果沒有，請使用寫入器執行個體網域名稱系統 (DNS) 名稱更新端點。	AWS 管理員
建立 AWS DMS 任務。	使用 主控台，在主要區域中建立 AWS DMS 任務。在任務中，指定僅複寫資料變更的遷移方法。如需詳細資訊，請參閱建立任務。 1. 在任務設定中，使用精靈並指定下列項目： • 來源交易的 CDC 啟動模式 – 啟用自訂 CDC 啟動模式 • 來源交易的自訂 CDC 起點 – 指定復原檢查點 2. 在復原檢查點方塊中，輸入先前透過資料表上的資料庫查詢取得的 `awsdms_txn_state` 複寫檢查點值。 3. 此外，在任務設定區段中，選取 JSON 編輯器，並將TaskRecoveryTableEnabled 參數設定為 true。 4. 最後，將 AWS DMS 任務開始遷移任務設定設為自動建立。	AWS 管理員，DBA

任務	Description	所需的技能
記錄 AWS DMS 任務 Amazon Resource Name (ARN)。	使用 ARN 取得 AWS DMS 任務名稱以供日後使用。若要擷取 AWS DMS 任務 ARN，請執行下列命令： `aws dms describe-replication-tasks` 執行另一個受管容錯移轉或在 DR 案例期間，將需要任務名稱。	AWS 管理員，DBA
刪除 AWS DMS 任務。	刪除主要區域中的原始（目前停止）AWS DMS 任務，以及次要區域中的現有 AWS DMS 任務（目前停止）。	AWS 管理員

相關資源

- 設定 Amazon Aurora 資料庫叢集
- 使用 Amazon Aurora 全球資料庫
- 使用 Amazon Aurora MySQL
- 使用 AWS DMS 複寫執行個體
- 使用 AWS DMS 端點
- 使用 AWS DMS 任務
- 什麼是 AWS CloudFormation？

其他資訊

Amazon Aurora 全域資料庫在此範例中用於 DR，因為它們提供 1 秒的有效復原時間目標 (RTO) 和不到 1 分鐘的復原點目標 (RPO)，兩者都低於傳統的複寫解決方案，非常適合 DR 案例。

Amazon Aurora 全域資料庫提供許多其他優點，包括下列項目：

- 具有本機延遲的全域讀取 – 全球消費者可以使用本機延遲存取本機區域中的資訊。
- 可擴展的次要 Amazon Aurora 資料庫叢集 – 次要叢集可以獨立擴展，最多可新增 16 個唯讀複本。
- 從主要叢集快速複寫到次要 Amazon Aurora 資料庫叢集 – 複寫對主要叢集的效能影響很小。它發生在儲存層，典型的跨區域複寫延遲少於 1 秒。

此模式也使用 AWS DMS 進行複寫。Amazon Aurora 資料庫提供建立僅供讀取複本的功能，可簡化複寫程序和 DR 設定。不過，當需要資料轉換或目標資料庫需要來源資料庫沒有的其他索引時，AWS DMS 通常用於複寫。

將具有超過 100 個引數的 Oracle 函數和程序遷移至 PostgreSQL

Srinivas Potlachervoo，Amazon Web Services

總結

此模式說明如何將具有超過 100 個引數的 Oracle 資料庫函數和程序遷移至 PostgreSQL。例如，您可以使用此模式將 Oracle 函數和程序遷移至下列其中一個 PostgreSQL 相容 AWS 資料庫服務：

- PostgreSQL 的 Amazon Relational Database Service (Amazon RDS)
- Amazon Aurora PostgreSQL-Compatible Edition

PostgreSQL 不支援具有超過 100 個引數的函數或程序。做為解決方法，您可以定義具有符合來源函數引數之類型欄位的新資料類型。然後，您可以建立並執行使用自訂資料類型做為引數的 PL/pgSQL 函數。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Amazon RDS Oracle 資料庫 (DB) 執行個體
- Amazon RDS for PostgreSQL 資料庫執行個體或 Aurora PostgreSQL 相容資料庫執行個體

產品版本

- Amazon RDS Oracle 資料庫執行個體 10.2 版及更新版本
- Amazon RDS PostgreSQL 資料庫執行個體 9.4 版及更新版本，或 Aurora PostgreSQL 相容資料庫執行個體 9.4 版及更新版本
- Oracle SQL Developer 18 版及更新版本
- pgAdmin 第 4 版及更新版本

Architecture

來源技術堆疊

- Amazon RDS Oracle 資料庫執行個體 10.2 版及更新版本

目標技術堆疊

- Amazon RDS PostgreSQL 資料庫執行個體 9.4 版及更新版本，或 Aurora PostgreSQL 相容資料庫執行個體 9.4 版及更新版本

工具

AWS 服務

- 適用於 PostgreSQL 的 Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展 PostgreSQL 關聯式資料庫。
- Amazon Aurora PostgreSQL 相容版本是完全受管的 ACID 相容關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。

其他服務

- Oracle SQL Developer 是一種整合的開發環境，可簡化傳統和雲端部署中 Oracle 資料庫的開發和管理。
- pgAdmin 是 PostgreSQL 的開放原始碼管理工具。它提供圖形界面，可協助您建立、維護和使用資料庫物件。

最佳實務

請確定您建立的資料類型符合來源 Oracle 函數或程序中包含的類型欄位。

史詩

執行具有超過 100 個引數的 Oracle 函數或程序

任務	Description	所需的技能
建立或識別具有超過 100 個引數的現有 Oracle/PLSQL 函數或程序。	建立具有超過 100 個引數的 Oracle/PLSQL 函數或程序。 -或-	Oracle/PLSQL 知識

任務	Description	所需的技能
	識別具有超過 100 個引數的現有 Oracle/PLSQL 函數或程序。 如需詳細資訊，請參閱 Oracle 資料庫文件中的第 14.7 節 CREATE FUNCTION 陳述式和 14.11 CREATE PROCEDURE 陳述式。	
編譯 Oracle/PLSQL 函數或程序。	編譯 Oracle/PLSQL 函數或程序。 如需詳細資訊，請參閱 Oracle 資料庫文件中的編譯函數。	Oracle/PLSQL 知識
執行 Oracle/PLSQL 函數。	執行 Oracle/PLSQL 函數或程序。然後，儲存輸出。	Oracle/PLSQL 知識

定義符合來源函數或程序引數的新資料類型

任務	Description	所需的技能
在 PostgreSQL 中定義新的資料類型。	在 PostgreSQL 中定義新的資料類型，其中包含出現在來源 Oracle 函數或程序引數中的所有相同欄位。 如需詳細資訊，請參閱 PostgreSQL 文件中的 CREATE TYPE。	PostgreSQL PL/pgSQL 知識

建立 PostgreSQL 函數，其中包含新的 TYPE 引數

任務	Description	所需的技能
建立包含新資料類型的 PostgreSQL 函數。	建立包含新TYPE引數的 PostgreSQL 函數。 若要檢閱範例函數，請參閱此模式的其他資訊區段。	PostgreSQL PL/pgSQL 知識
編譯 PostgreSQL 函數。	在 PostgreSQL 中編譯 函數。如果新的資料類型欄位符合來源函數的 或程序的引數，則函數會成功編譯。	PostgreSQL PL/pgSQL 知識
執行 PostgreSQL 函數。	執行 PostgreSQL 函數。	PostgreSQL PL/pgSQL 知識

疑難排解

問題	解決方案
函數會傳回下列錯誤： 錯誤：「<statement>」附近的語法錯誤	確定所有函數的陳述式都以分號 () 結尾;。
函數會傳回下列錯誤： 錯誤：「<variable>」不是已知的變數	確定函數內文中使用的變數已列在函數的 DECLARE區段中。

相關資源

- 使用 Amazon Aurora PostgreSQL (Amazon Aurora 使用者指南)
- CREATE TYPE (PostgreSQL 文件)

其他資訊

包含 TYPE 引數的範例 PostgreSQL 函數

```
CREATE OR REPLACE FUNCTION test_proc_new
(
    IN p_rec type_test_proc_args
)
RETURNS void
AS
$BODY$
BEGIN

    /*
    **************
    The body would contain code to process the input values.
    For our testing, we will display couple of values.
    ***************
    */
    RAISE NOTICE USING MESSAGE = CONCAT_WS('', 'p_acct_id: ', p_rec.p_acct_id);
    RAISE NOTICE USING MESSAGE = CONCAT_WS('', 'p_ord_id: ', p_rec.p_ord_id);
    RAISE NOTICE USING MESSAGE = CONCAT_WS('', 'p_ord_date: ', p_rec.p_ord_date);

END;
$BODY$
LANGUAGE plpgsql
COST 100;
```

將 Redis 工作負載遷移至 AWS 上的 Redis Enterprise Cloud

Antony Prasad Thevaraj，Amazon Web Services

Srinivas Pendyala，Redis

總結

此模式討論將 Redis 工作負載遷移至 Redis Enterprise Cloud on Amazon Web Services (AWS) 的高階程序。它描述了遷移步驟，提供有關選擇可用工具的資訊，並討論了使用每個工具的優點、缺點和步驟。或者，如果您需要從 Redis 遷移工作負載的其他協助，您可以與 Redis Professional Services 互動。

如果您在內部部署執行 Redis OSS 或 Redis Enterprise Software，您會熟悉在資料中心維護 Redis 資料庫的重大管理開銷和操作複雜性。透過將工作負載遷移至雲端，您可以大幅減輕營運負擔，並利用 Redis Enterprise Cloud，這是 Redis 的全託管資料庫即服務 (DBaaS) 產品。此遷移有助於提高業務敏捷性、改善應用程式可靠性，並降低整體成本，同時您可以存取最新的 Redis Enterprise Cloud on AWS 功能，例如 99.999% 的可用性、架構簡單性和擴展性。

金融服務、零售、醫療保健和遊戲領域以及需要詐騙偵測、即時清查、索賠處理和工作階段管理解決方案的使用案例中，都有 Redis Enterprise Cloud 的潛在應用程式。您可以使用 Redis Enterprise Cloud 連線至您的 AWS 資源，例如，連線至在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上執行的應用程式伺服器，或連線到部署為 AWS Lambda 服務的微服務。

先決條件和限制

假設

- 您目前正在操作要遷移至雲端的內部部署資料庫系統。
- 您已識別工作負載的遷移需求，包括：
  - 資料一致性需求
  - 基礎設施和系統環境需求
  - 資料映射和轉換需求
  - 功能測試要求
  - 效能測試要求
  - 驗證要求
  - 定義的切換策略

- 您已評估遷移所需的時間表和成本估算。
- 您的需求會考量工作範圍，以及您識別為遷移一部分的系統和資料庫。
- 您已在負責、負責、諮詢、告知 (RACI) 矩陣中識別利益相關者及其角色和責任。
- 您已收到所有利益相關者的必要協議和核准。

成本

根據現有來源資料庫的技術規格（例如記憶體大小、輸送量和總資料大小），Redis 解決方案架構師可以在 Redis Enterprise Cloud 上調整目標系統的大小。 如需一般定價資訊，請參閱 Redis 網站上的 Redis 定價。

人員和技能

遷移程序涉及下列角色和責任。

Role	Description	所需的技能
遷移解決方案架構師	具備定義、規劃和實作遷移策略專業知識的技術架構師	了解來源和目標系統的技術和應用程式層級；具備將工作負載遷移至雲端的經驗
資料架構師	技術架構師，在定義、實作和交付各種資料庫的資料解決方案方面擁有廣泛的經驗	結構化和非結構化資料的資料建模、對為企業實作資料庫的深入了解和經驗
Redis 解決方案架構師	技術架構師，可協助針對適當的使用案例建構大小最佳的 Redis 叢集	精通各種使用案例的架構和部署 Redis 解決方案
雲端解決方案架構師	技術架構師對雲端解決方案有更深入的了解，尤其是在 AWS	精通雲端架構解決方案；工作負載遷移和應用程式現代化體驗
企業架構師	技術架構師，完全了解組織的技術格局、擁有未來藍圖的共同願景，以及在整個組織中所有團隊中實作和建立標準化架構最佳實務	軟體架構認證，例如 TOGAF、基礎軟體工程技能，以及解決方案架構和企業架構專業知識

IT 或 DevOps 工程師	負責建立和維護基礎設施的工程師，包括監控基礎設施是否有問題、執行維護任務，以及視需要進行更新。	充分了解各種技術，包括作業系統、聯網和雲端運算；熟悉程式設計語言，例如 Python、Bash 和 Ruby，以及 Docker、Kubernetes 和 Ansible 等工具

Architecture

遷移選項

下圖顯示將內部部署 (Redis 型或其他） 資料來源遷移至 AWS 的選項。它會顯示數種您可以選擇的遷移工具，例如使用 Redis 複寫功能或使用 AWS DMS 將 Redis 資料庫 (RDB) 檔案匯出至 Amazon Simple Storage Service (Amazon S3)。

1. 內部部署資料來源：非以 Redis 為基礎的資料庫，例如 MySQL、PostgreSQL、Oracle、SQL Server 或 MariaDB。
2. 內部部署資料來源：Redis 通訊協定型資料庫，例如 Redis OSS 和 Redis Enterprise Software。
3. 從 Redis 型資料庫遷移資料最簡單的方法是匯出 RDB 檔案，並將其匯入目標 Redis Enterprise Cloud on AWS。
4. 或者，您可以使用 Redis 中的複寫功能 (`REPLICAOF`)，將資料從來源遷移到目標。
5. 如果您的資料遷移需求包含資料轉換，您可以採用 Redis 輸入/輸出工具 (RIOT) 來遷移資料。
6. 或者，您可以使用 AWS Data Migration Service (AWS DMS) 從 SQL 型資料庫遷移資料。
7. 您必須使用 AWS DMS 的虛擬私有雲端 (VPC) 對等互連，才能成功將資料遷移至目標 Redis Enterprise Cloud on AWS。

目標架構

下圖顯示 Redis Enterprise Cloud on AWS 的典型部署架構，並說明如何與金鑰 AWS 服務搭配使用。

1. 您可以連線到由 Redis Enterprise Cloud on AWS 支援的商業應用程式。
2. 您可以在自己的 AWS 帳戶中，在該帳戶中的 VPC 中執行商業應用程式。

3. 您可以使用 Redis Enterprise Cloud 資料庫端點來連線至您的應用程式。範例包括在 EC2 執行個體上執行的應用程式伺服器、部署為 AWS Lambda 服務的微服務、Amazon Elastic Container Service (Amazon ECS) 應用程式或 Amazon Elastic Kubernetes Service (Amazon EKS) 應用程式。
4. 在 VPC 中執行的商業應用程式需要與 Redis Enterprise Cloud VPC 的 VPC 對等連線。這可讓商業應用程式透過私有端點安全地連線。
5. Redis Enterprise Cloud on AWS 是一種記憶體內 NoSQL 資料庫平台，部署為 AWS 上的DBaaS，並由 Redis 完全管理。
6. Redis Enterprise Cloud 會部署在由 Redis 建立的標準 AWS 帳戶中的 VPC 內。
7. 基於安全考量，Redis Enterprise Cloud 會部署在可在私有和公有端點存取的私有子網路中。建議您將用戶端應用程式連接到私有端點上的 Redis。如果您打算使用公有端點，強烈建議您啟用 TLS 來加密用戶端應用程式與 Redis Enterprise Cloud 之間的資料。

Redis 遷移方法與 AWS 遷移方法保持一致，如 AWS 規範指引網站上的調動您的組織以加速大規模遷移。

自動化和擴展

遷移的環境設定任務可以透過 AWS 登陸區域和基礎設施即程式碼 (IaC) 範本進行自動化和擴展。這些會在此模式的 Epics 區段中討論。

工具

根據您的資料遷移需求，您可以選擇各種技術選項，將資料遷移至 Redis Enterprise Cloud on AWS。下表說明並比較這些工具。

工具	Description	優點	缺點
RDB 匯出和匯入	您從來源（例如 Redis OSS 或 Redis Enterprise Software) 資料庫匯出資料，格式為 RDB 檔案。如果您的資料庫是透過 Redis OSS 叢集提供，您可以將每個主碎片匯出至 RDB。	• 簡單。 • 可與任何以 Redis 為基礎的解決方案搭配使用，這些解決方案可將 RDB 格式的資料匯出為來源（包括 Redis OSS 和 Redis Enterprise Software)。	• 不處理資料轉換需求或支援邏輯資料庫合併。 • 大型資料集耗時。 • 沒有差異遷移支援會導致更長的停機時間。

然後，您只需一個步驟即可匯入所有 RDB 檔案。如果您的來源資料庫是以 OSS 叢集為基礎，但您的目標資料庫未使用 OSS 叢集 API，您必須將應用程式原始碼變更為使用標準 Redis 用戶端程式庫。 資料轉換需求或邏輯資料庫合併需要更複雜的程序，此表稍後會在邏輯資料庫合併下說明。	• 透過簡單的程序實現資料一致性。

Redis 複寫功能（主動-被動）	您可以持續將資料從 Redis OSS、Enterprise Software 或 Enterprise Cloud 資料庫複寫到 Redis Enterprise Cloud 資料庫。初始同步後，Redis 複寫功能 (`ReplicaOf`) 會執行差異遷移，這表示幾乎沒有觀察到應用程式停機時間。 Redis 複寫功能旨在以主動-被動的方式使用。假設目標是被動的，並完全重新同步（從來源資料庫漂移和同步）。因此，在來源和目標之間切換有些複雜。 您可以將 OSS 叢集的所有主碎片指定為來源，以從 Redis OSS 叢集複寫到標準叢集 Redis Enterprise Cloud 資料庫。不過，Redis 複寫功能最多允許 32 個來源資料庫。	• 支援連續複寫（初始資料載入，後面接著差異）。 • 幾乎沒有停機時間（取決於複寫延遲）。 • 實現資料一致性。	• 只有一個站台是作用中的，因此在站台之間切換會更複雜。 • 當您從 OSS 叢集遷移時，最多支援 32 個主碎片。

AWS DMS	您可以使用 AWS DMS，以最短的停機時間將資料從任何支援的來源資料庫遷移到目標 Redis 資料存放區。如需詳細資訊，請參閱 AWS DMS 文件中的使用 Redis 做為 AWS DMS 的目標。	• 支援 NoSQL 和 SQL 資料來源的遷移。 • 與其他 AWS 服務搭配運作良好。 • 支援即時遷移和變更資料擷取 (CDC) 使用案例。	• Redis 鍵值不能包含特殊字元，例如 %。 • 不支援遷移資料列或欄位名稱中具有特殊字元的資料。 • 不支援完整大型二進位物件 (LOB) 模式。
邏輯資料庫合併	特殊資料庫合併需求可能需要自訂資料遷移解決方案。例如，您可能在 Redis OSS 中有四個邏輯資料庫 (`SELECT 0..3`)，但您可能想要使用單一資料庫端點，而不是將資料移至多個 Redis Enterprise Cloud 資料庫。Redis Enterprise 不支援可選取的邏輯資料庫，因此您必須轉換來源資料庫的實體資料模型。例如，您可以將每個資料庫索引映射到字首 (`0` 到 `cmp`、`usr1`到 等），然後使用遷移指令碼或擷取、轉換和載入 (ETL) 工具來輸出 RDB 檔案，然後您可以將其匯入目標資料庫。	• 使用自訂指令碼，在遷移至目標系統期間精細控制資料形狀。	• 如果您決定不完成遷移，則轉返可能非常具有挑戰性，特別是如果必須將較新的資料轉返至來源系統時。 • 如果目標是為一次性遷移建置一次性解決方案，則建置成本可能會很高。 • 如果遷移需求經常變更，程式碼、基礎設施、開發時間和其他區域的維護成本可能會很高。

此外，您可以從 AWS 使用下列工具和服務。

評估和探索工具：

- 遷移評估器

應用程式和伺服器遷移工具：

- AWS Application Migration Service

資料庫遷移工具：

- AWS Schema Conversion Tool (AWS SCT)
- AWS Database Migration Service (AWS DMS)

資料遷移工具：

- AWS Storage Gateway
- AWS DataSync
- AWS Direct Connect
- AWS Snowball
- Amazon Data Firehose

AWS 合作夥伴解決方案：

- AWS 遷移能力合作夥伴

史詩

完成探索和評估任務

任務	Description	所需的技能
識別工作負載。	識別您要遷移的適當候選工作負載。在選擇工作負載進行遷移之前，請考慮下列事項：	資料架構師、商業擁護者、遷移專案發起人

任務	Description	所需的技能
	• 遷移或未遷移此工作負載的商業價值為何？ • 如果此工作負載未成功遷移至目標系統，是否有緊急應變計畫？ 理想情況下，請選擇具有最大業務影響且涉及最低風險的工作負載。保持整體程序反覆運算並以小幅度遷移。	
識別資料來源和需求；設計資料模型。	Redis 會執行研討會來加速探索並定義專案的遷移規劃。在本研討會中，Redis 團隊會識別資料來源和來源資料模型需求，並分析如何在 Redis Enterprise Cloud 中重新建模。 Redis 遷移團隊（專業服務）會與您的組織執行詳細的資料模型設計練習。在本練習中，Redis 團隊會： • 識別目標 Redis 資料結構。 • 定義資料映射策略。 • 記錄遷移方法和建議。 • 與利益相關者一起檢閱和完成資料模型。	Redis 解決方案架構師

任務	Description	所需的技能
識別來源資料庫的特性。	識別來源和目標環境中使用的 Redis 產品。例如： • 來源資料庫是 OSS 叢集資料庫、獨立 Redis 資料庫還是 Redis Enterprise 資料庫？ • 目標資料庫是 Redis Enterprise 標準資料庫還是 OSS 叢集相容的資料庫？ • 對應用程式原始碼有何影響？	資料架構師
收集目前的系統 SLA 和其他調整規模指標。	判斷目前服務層級協議 (SLAs)，以輸送量（每秒操作數）、延遲、每個資料庫的整體記憶體大小，以及高可用性 (HA) 需求表示。	資料架構師

任務	Description	所需的技能
識別目標系統的特性。	判斷這些問題的答案： • 需要遷移多少資料？ • 遷移給定的資料量需要多長時間？ • 遷移的停機時間需求為何？您的服務或應用程式是否可以在特定期間內無法使用？如果是這樣，持續多久？ • 遷移的資料應該有多一致？目標資料庫是否可以處於稍微不一致（過時）狀態？ • 資料是否必須先轉換，才能載入目標資料庫？（例如，您可能想要在遷移之前將可選取的資料庫索引轉換為字首。） • 來源資料庫是否可從目標資料庫的主機（例如，從對等 VPC 或使用加密從公有端點）存取？ • 使用 Redis 技術架構師完成資料大小調整和 Redis 叢集大小調整練習。 • 識別聯網需求、基礎設施需求、軟體版本和軟體授權，並在遷移之前採購任何元件。 • 傳輸此資料是否有任何相關的安全問題？	資料架構師、Redis 解決方案架構師（選用）

任務	Description	所需的技能
識別相依性。	識別要遷移之目前系統的上游和下游相依性。確定遷移工作與其他相依系統遷移一致。例如，如果您打算將其他商業應用程式從內部部署遷移到 AWS 雲端，請識別這些應用程式，並根據專案目標、時間表和利益相關者進行調整。	資料架構師、企業架構師
識別遷移工具。	根據您的資料遷移需求（例如來源資料或停機時間需求），您可以使用先前在工具章節中所述的任何工具。此外，您可以使用： • 使用 CRDB 部署進行雙向（主動）複寫。 • 自訂匯出/匯入指令碼（例如，使用 DUMP/RESTORE 命令）。 • 其他匯出/匯入工具和協助工具，例如 RIOT、ECstats2 或 ETL 工具。 • Terraform 或 AWS CloudFormation 範本等 IaC 工具。	遷移解決方案架構師，Redis 解決方案架構師
建立應變計畫。	建立應變計畫以復原，以防您在遷移期間遇到問題。	專案管理、技術團隊，包括架構師

完成安全與合規任務

任務	Description	所需的技能
保護 Redis 管理主控台。	若要保護管理主控台，請遵循 Redis 文件中的指示。	IT 基礎設施管理員
保護 Redis 資料庫。	請參閱 Redis 文件中的下列頁面以： • 定義角色型存取控制。 • 定義網路安全。 • 啟用 TLS。	
保護 Redis 雲端 APIs。	啟用 API 時，您可以管理 Redis Cloud 帳戶所有擁有者的 API 金鑰。如需 API 安全功能的概觀，請參閱 Redis 網站上的 API 身分驗證文件。	IT 基礎設施管理員

設定新環境

任務	Description	所需的技能

在 AWS 上設定新的環境。	此任務包括：
	• AWS 登陸區域設定活動。登陸區域支援：
	  • 多帳戶部署
	  • 最低安全性基準
	  • 以安全基準和 ISV 先決條件（網路、安全組態等）自動佈建新帳戶的方式
	  • 通知、集中式記錄和監控
	• ISV 軟體組態活動。這包括需要包含在遷移中的組態，
		IT 或 DevOps 工程師

任務	Description	所需的技能
	例如產品和工作負載設定和變更。 • IaC 活動，例如設定或自訂 AWS CloudFormation 或 Terraform 範本。	
部署遷移架構。	1. 在 AWS 上設定 Redis Enterprise Cloud。 2. 安裝遷移工具，例如 RIOT 或 AWS DMS。如需可用工具的清單，請參閱工具一節。 3. 在應用程式、遷移和資料庫層之間建立連線。 4. 建立可流經每一層的範例工作負載，並遷移少量的範例資料。 您現在已準備好執行實際的資料遷移管道並進行測試。	IT 或 DevOps 工程師

設定聯網

任務	Description	所需的技能
建立連線。	在內部部署基礎設施和 AWS 雲端資源之間建立連線。使用安全群組、AWS Direct Connect 和其他資源來實現此功能。如需詳細資訊，請參閱 AWS 網站上的將您的資料中心連線至 AWS。	IT 或 DevOps 工程師

任務	Description	所需的技能
設定 VPC 對等互連。	在 VPCs 執行商業應用程式的 VPC （或執行遷移工具的 EC2 執行個體或 AWS DMS 複寫伺服器）與執行 Redis Enterprise Cloud 的 VPC 之間建立 VPC 對等互連。如需說明，請參閱 Amazon VPC 文件中的開始使用 Amazon VPC，以及 Redis 文件中的啟用 VPC 對等互連。	IT 或 DevOps 工程師

遷移資料

任務	Description	所需的技能
選擇資料遷移工具。	檢閱工具區段中的表格，以查看這些工具的說明、優點和缺點： • RDS 匯出和匯入 • Redis 複寫功能 (`ReplicaOf`) • AWS DMS • 邏輯資料庫合併 下列資料列說明與每個工具相關聯的資料遷移任務。	遷移解決方案架構師
選項 1：使用 RDB 匯出和匯入。	1. 中斷連線來源：停止來源資料庫上的流量（例如，中斷連線商業應用程式）。 2. 匯出：將來源資料庫的資料匯出為 RDB 檔案。	遷移解決方案架構師，Redis 解決方案架構師

任務	Description	所需的技能
	3. 階段：將資料上傳到 AWS 上 Redis Enterprise Cloud 執行個體可存取的位置（例如，您可以將它們上傳到 S3 儲存貯體或 FTP 伺服器）。 4. 匯入：將 RDB 檔案（透過在一個匯入步驟中列出所有檔案）匯入 Redis Enterprise Cloud 目標資料庫。 5. 切換：移至目標資料庫（例如，透過將應用程式連線至資料庫）。 如需詳細資訊，請參閱 Redis 文件。	

任務	Description	所需的技能
選項 2：使用 Redis 複寫功能（主動-被動）。	1. 連接資料庫：在來源和目標資料庫之間建立`ReplicaOf`連結。 2. 執行初始同步：等待來源和目標資料庫之間的初始同步完成。 3. 中斷連線來源：停止來源資料庫上的流量（例如中斷連線應用程式）。 4. 執行差異複寫：等待目標資料庫複寫差異。 5. 切換：移至目標資料庫（例如，將您的應用程式連接到該資料庫）。 6. 刪除：移除來源和目標資料庫之間的`ReplicaOf`連結。 如需詳細資訊，請參閱 Redis 文件。	遷移解決方案架構師，Redis 解決方案架構師

任務	Description	所需的技能
選項 3：使用 AWS DMS。	1. 設定 AWS DMS 複寫執行個體：此執行個體會執行所有遷移程序。如需指示：使用 AWS DMS 文件中的 AWS DMS 複寫執行個體。 2. 定義來源資料庫：定義來源端點。測試來源端點與 AWS DMS 複寫伺服器之間的連線。如需說明：在 AWS DMS 文件中建立來源和目標端點。 3. 設定目標資料庫：在 AWS 上設定 Redis Enterprise Cloud 並設定要遷移至的資料庫。 4. 定義目標資料庫：定義目標端點。確定已在執行 AWS DMS 的 VPC 與託管 AWS 上 Redis Enterprise Cloud 的 VPC 之間建立 VPC 對等互連。測試 AWS DMS 複寫伺服器與目標資料庫之間的連線。 5. 建立 AWS DMS 任務：建立任務或一組任務，以定義您要用來遷移資料的資料表和複寫程序。如需指示：在 AWS DMS 文件中使用 AWS DMS 任務。 6. 遷移：執行 AWS DMS 任務來遷移資料。	遷移解決方案架構師，Redis 解決方案架構師

任務	Description	所需的技能
	7. 切換：移至目標資料庫（例如，將您的應用程式連接到該資料庫）。	
選項 4：使用邏輯資料庫合併。	此選項涉及使用遷移指令碼或 ETL 工具來轉換來源資料庫的實體資料模型，以及產生 RDB 檔案。Redis Professional Services 會視需要協助此步驟。	遷移解決方案架構師，Redis 解決方案架構師

遷移您的應用程式

任務	Description	所需的技能
協調專案管理時間表和目標。	將應用程式層的遷移專案目標、里程碑和時間表與 Redis 資料遷移專案的目標、里程碑和時間表保持一致。	專案管理
對齊測試活動。	在 AWS 雲端中遷移和現代化應用程式層之後，將應用程式層指向新遷移的 Redis Enterprise Cloud on AWS 進行測試。	測試

測試

任務	Description	所需的技能
實作測試計畫。	在測試環境中根據測試需求，在您的站點執行資料遷移常式和在實作階段開發的指令碼。	測試

任務	Description	所需的技能
測試資料品質。	遷移資料後測試資料品質。	測試
測試功能。	測試資料查詢和應用程式層，以確保應用程式在與來源系統中相同的層級執行。	測試

切換

任務	Description	所需的技能
做出切換決策。	在所有應用程式層級和資料庫層級測試完成後，執行領導團隊和利益相關者會根據測試團隊確認的最終結果，決定是否在 AWS 上切換到新環境。	專案管理、商業擁護者
切換到 AWS 雲端。	當您確認一切就緒時，請將應用程式層指向新遷移的資料，並將用戶端指向根據 AWS 上新的 Redis Enterprise Cloud 系統執行的新應用程式層。	IT 或 DevOps 工程師、資料架構師、遷移解決方案架構師、Redis 解決方案架構師

相關資源

Redis 資源

- Redis Enterprise Cloud 文件
- RIOT 工具 (GitHub 儲存庫）
- Terraform 提供者 （下載）

AWS 資源

- 示範遷移
- AWS 合作夥伴解決方案

- 文件
- 部落格文章
- 白皮書
- 教學課程和影片
- AWS 雲端遷移
- AWS 方案指引

其他資訊

如需將 Redis 工作負載遷移至 AWS 雲端的標準安全需求，請參閱 AWS 網站上的安全、身分和合規最佳實務，以及 Redis 網站上的 Redis 信任中心。

使用具有相同主機名稱的 SAP HSR 將 SAP HANA 遷移至 AWS

Pradeep Puliyampatta，Amazon Web Services

總結

SAP HANA 遷移至 Amazon Web Services (AWS) 可以使用多個選項執行，包括備份和還原、匯出和匯入，以及 SAP HANA 系統複寫 (HSR)。特定選項的選擇取決於來源和目標 SAP HANA 資料庫之間的網路連線、來源資料庫的大小、停機時間考量事項和其他因素。

當來源和目標系統與整個資料庫 (SAP HANA 資料庫複寫快照） 之間有穩定的網路時，將 SAP HANA 工作負載遷移至 AWS 的 SAP HSR 選項可在 1 天內完全複寫，如 SAP 針對 SAP HSR 的網路輸送量需求所規定。此方法的停機時間需求僅限於對目標 AWS 環境、SAP HANA 資料庫備份和遷移後任務執行接管。

SAP HSR 支援將不同的主機名稱 （映射至不同 IP 地址的主機名稱） 用於主要或來源與次要或目標系統之間的複寫流量。您可以透過在 的 `[system_replication_hostname_resolution]`區段下定義這些特定的主機名稱集來執行此操作`global.ini`。在本節中，必須在每個主機上定義主要和次要網站的所有主機。如需詳細的組態步驟，請參閱 SAP 文件。

此設定的一個關鍵要點是，主要系統中的主機名稱必須與次要系統中的主機名稱不同。否則，可以觀察到下列錯誤。

- `"each site must have a unique set of logical hostnames"`
- `"remoteHost does not match with any host of the source site. All hosts of source and target site must be able to resolve all hostnames of both sites correctly"`

不過，遷移後步驟的數量可以透過在目標 AWS 環境中使用相同的 SAP HANA 資料庫主機名稱來減少。

此模式提供了在使用 SAP HSR 選項時，在來源和目標環境中使用相同主機名稱的解決方法。透過此模式，您可以使用 SAP HANA 主機名稱重新命名選項。您可以將暫時主機名稱指派給目標 SAP HANA 資料庫，以促進 SAP HSR 的主機名稱唯一性。遷移完成目標 SAP HANA 環境的接管里程碑後，您可以將目標系統主機名稱還原為來源系統的主機名稱。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 具有虛擬私有網路 (VPN) 端點或路由器的虛擬私有雲端 (VPC)。
- AWS Client VPN 或 AWS Direct Connect 設定為將檔案從來源傳輸到目標。
- 來源和目標環境中的 SAP HANA 資料庫。目標 SAP HANA 資料庫修補程式層級應該等於或高於相同 SAP HANA 平台版本中的來源 SAP HANA 資料庫修補程式層級。例如，複寫無法在 HANA 1.0 和 HANA 2.0 系統之間設定。如需詳細資訊，請參閱 SAP 備註：1999880 – 常見問答集：SAP HANA 系統複寫中的問題 15。
- 目標環境中的 SAP 應用程式伺服器。
- 目標環境中的 Amazon Elastic Block Store (Amazon EBS) 磁碟區。

限制

下列 SAP 文件清單涵蓋與此因應措施相關的已知問題，包括 SAP HANA 動態分層和橫向擴展遷移的相關限制：

- 2956397 – SAP HANA 資料庫系統重新命名失敗
- 2222694 – 嘗試重新命名 HANA 系統時，出現以下錯誤「原始 sidadm 使用者不擁有來源檔案 (uid = xxxx)」
- 2607227 – hdblcm：register_rename_system：重新命名 SAP HANA 執行個體失敗
- 2630562 – HANA 主機名稱重新命名失敗，且 HANA 未啟動
- 2935639 – sr_register 未使用 global.ini 區段中 system_replication_hostname_resolution 指定的主機名稱
- 2710211 – 錯誤：來源系統和目標系統具有重疊的邏輯主機名稱
- 2693441 – 由於錯誤而無法重新命名 SAP HANA 系統
- 2519672 – HANA 主要和次要具有不同的系統 PKI SSFS 資料和金鑰，或無法檢查
- 2457129 – 當動態分層是橫向的一部分時，不允許 SAP HANA 系統主機重新命名
- 2473002 – 使用 HANA 系統複寫來遷移向外擴展系統（在 中，針對向外擴展 SAP HANA 系統使用此主機名稱重新命名方法，SAP 沒有提供限制。 不過，每個個別主機都必須重複此程序。 其他橫向擴展遷移限制也適用於此方法。）

產品版本

- 此解決方案適用於 SAP HANA 資料庫平台版本 1.0 和 2.0。

Architecture

來源設定

SAP HANA 資料庫安裝在來源環境中。所有 SAP 應用程式伺服器連線和資料庫介面都使用相同的主機名稱進行用戶端連線。下圖顯示範例來源主機名稱hdbhost及其對應的 IP 地址。

目標設定

AWS 雲端 目標環境使用相同的主機名稱來執行 SAP HANA 資料庫。AWS 上的目標環境包括下列項目：

- SAP HANA 資料庫
- SAP 應用程式伺服器
- EBS 磁碟區

中繼組態

在下圖中， AWS 目標環境上的主機名稱會暫時重新命名，temp-host因此來源和目標上的主機名稱是唯一的。遷移完成目標環境的接管里程碑後，會使用原始名稱 重新命名目標系統虛擬主機名稱hdbhost。

中繼組態包含下列其中一個選項：

- AWS Client VPN 使用 Client VPN 端點
- Direct Connect 連線至路由器

AWS 目標環境上的 SAP 應用程式伺服器可以在複寫設定之前或之後安裝。不過，在複寫設定之前安裝應用程式伺服器，有助於減少安裝期間的停機時間、高可用性組態和備份。

工具

AWS 服務

- AWS Client VPN 是一種受管的用戶端型 VPN 服務，可讓您安全地存取內部部署網路中的 AWS 資源。

- AWS Direct Connect 會透過標準乙太網路光纖纜線將您的內部網路連結至 Direct Connect 位置。透過此連線，您可以直接建立與公有的虛擬介面 AWS 服務，繞過網路路徑中的網際網路服務供應商。
- Amazon Elastic Block Store (Amazon EBS) 提供區塊層級儲存磁碟區，可與 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體搭配使用。EBS 磁碟區的行為與未格式化的原始區塊型儲存設備相似。您可以將這些磁碟區做為裝置，掛載在您的執行個體上。

其他工具

- SAP 應用程式伺服器 – SAP 應用程式伺服器提供程式設計人員表達商業邏輯的方式。SAP 應用程式伺服器會根據商業邏輯執行資料處理。實際資料會存放在資料庫中，這是個別的元件。
- SAP HANA 駕駛艙和 SAP HANA Studio – SAP HANA 駕駛艙和 SAP HANA Studio 都提供 SAP HANA 資料庫的管理界面。在 SAP HANA Studio 中，SAP HANA 管理主控台是提供 SAP HANA 資料庫管理相關內容的系統檢視。
- SAP HANA 系統複寫 – SAP HANA 系統複寫 (SAP HSR) 是 SAP 提供的標準程序，用於複寫 SAP HANA 資料庫。SAP HSR 所需的可執行檔是 SAP HANA 伺服器核心本身的一部分。

史詩

準備來源和目標環境

任務	Description	所需的技能
安裝和設定 SAP HANA 資料庫。	在來源和目標環境中，確保 SAP HANA 資料庫已安裝並根據 SAP HANA on 最佳實務進行設定。如需詳細資訊，請參閱 SAP HANA on AWS。	SAP Basis 管理
映射 IP 地址。	在目標環境中，確保將暫時主機名稱指派給內部 IP 地址。 1. 透過導覽至 EC2、執行個體、動作、聯網、管理 IP 地址、指派新的 IP 地址，將次要 IPv4 地址指派給 AWS 管理主控台上的 EC2 執行個體。	AWS 管理

任務	Description	所需的技能
	2. 若要將相同的地址指派給 EC2 網路轉接器 (NIC)，請從作業系統中，以根使用者身分執行命令 `ip addr add <IP>/32 dev eth0`，`<IP>`將 取代為步驟 1 的 IP 地址。	
解決目標主機名稱。	在次要 SAP HANA 資料庫上，更新 `/etc/hosts` 檔案中的相關主機名稱，確認已解析 SAP HANA 複寫網路的兩個主機名稱 (`hdbhost` 和 `temp-host`)。	Linux 管理
備份來源和目標 SAP HANA 資料庫。	使用 SAP HANA Studio 或 SAP HANA 駕駛艙在 SAP HANA 資料庫上執行備份。	SAP Basis 管理
Exchange 系統 PKI 憑證。	（僅適用於 SAP HANA 2.0 和更新版本） 在主要和次要資料庫之間的檔案系統 (SSFS) 存放區中的系統公有金鑰基礎設施 (PKI) 安全存放區中交換憑證。如需詳細資訊，請參閱 SAP Note 2369981 – 使用 SAP HANA 系統複寫進行身分驗證所需的組態步驟。	SAP Basis 管理

重新命名目標 SAP HANA 資料庫

任務	Description	所需的技能
停止目標用戶端連線。	在目標環境中，關閉 SAP 應用程式伺服器和其他用戶端連線。	SAP Basis 管理

將目標 SAP HANA 資料庫重新命名為暫時主機名稱。	1. 身為根使用者，請使用常駐將目標 SAP HANA 資料庫主機名稱重新命名為暫時主機名稱hdblcm。
	`root $> cd /hana/shared/<SID/hdblcm`
	`root $> ./hdblcm`
	2. 選擇選項9 | rename_system | Rename the SAP HANA Database System。
	3. 提供新名稱： temp-host 。
	4. 您可以視需要驗證其他選項。不過，請確定您不會將主機重新命名與 SID 變更混淆 (SAP Note 2598814 – hdblcm：SID 重新命名失敗）。
	
	SAP HANA 資料庫停止和啟動將由 控制hdblcm。
		SAP Basis 管理

指派複寫網路。	在來源系統的 global.ini 檔案中，在 [system_replication_hostname_resolution] 標頭下，	SAP Basis 管理

任務	Description	所需的技能
	提供來源和目標複寫網路詳細資訊。然後將項目複製到目標系統上`global.ini` 的檔案。	
在主要 上啟用複寫。	若要在來源 SAP HANA 資料庫上啟用複寫，請執行下列命令。 `hdbnsutil -sr_enable --name=siteA`	SAP Basis 管理

將目標 SAP HANA 資料庫註冊為次要系統。	若要將目標 SAP HANA 資料庫註冊為用於 SAP HSR 來源的次要系統，請選擇非同步複寫。
	`(sid)adm $> HDB stop`
	`(sid)adm $> hdbnsutil -sr_register –name=siteB –remotehost=hdbhost /`
	`--remoteInstance=00 –replicationMode=async –operationMode=logreplay`
	`(sid)adm $> HDB start`
	或者，您可以選擇要註冊–`online`的選項。在這種情況下，您不需要停止和啟動 SAP HANA 資料庫。
		SAP Basis 管理

任務	Description	所需的技能

驗證同步。	在來源 SAP HANA 資料庫上，確認所有日誌都套用到目標系統（因為它是非同步複寫）。
	
	若要驗證複寫，請在來源上執行下列命令。
	
	`(sid)adm $> cdpy`
	`(sidadm $> python systemReplicationStatus.py`
		SAP Basis 管理

關閉來源 SAP 應用程式和 SAP HANA 資料庫。	在遷移切換期間，請關閉來源系統 (SAP 應用程式和 SAP HANA 資料庫。	SAP Basis 管理
在目標執行接管。	若要在 AWS 上執行目標接管，請執行命令 `hdbnsutil -sr_takeover` 。	SAP Basis 管理
在目標 SAP HANA 資料庫上，關閉複寫。	若要清除複寫中繼資料，請執行命令 來停止目標系統的複寫`hdbnsutil -sr_disable` 。 **Note** 這符合 SAP Note 2693441 – 由於錯誤而無法重新命名 SAP HANA 系統。	SAP Basis 管理
備份目標 SAP HANA 資料庫。	接管成功後，建議您執行完整的 SAP HANA 資料庫備份。	SAP Basis 管理

還原至目標系統中的原始主機名稱

任務	Description	所需的技能

將目標 SAP HANA 資料庫主機名稱還原為原始。	1. 若要將目標 SAP HANA 資料庫主機名稱還原為原始虛擬主機名稱，請使用常駐 `hdblcm`。
	`root $> cd /hana/shared/<SID>/hdblcm`
	`root $> ./hdblcm`
	2. 選擇選項`9 | rename_system | Rename the SAP HANA Database System`。
	3. 提供新名稱：`hdbhost`。
	
	您可以視需要驗證其他選項。不過，請確定您不會將主機重新命名與 SID 變更混淆 (SAP Note 2598814 – hdblcm：SID 重新命名失敗）。
		SAP Basis 管理

調整 hdbuserstore。	調整指向來源`hdbuserstore` 詳細資訊`schema/user` 的詳細資訊。如需詳細步驟，請參閱 SAP 文件。 若要驗證此步驟，請執行命令 `R3trans -d`。結果應該反映 SAP HANA 資料庫的成功連線。	SAP Basis 管理

任務	Description	所需的技能
啟動用戶端連線。	在目標環境中，啟動 SAP 應用程式伺服器和其他用戶端連線。	SAP Basis 管理

相關資源

SAP 參考

SAP 文件參考經常由 SAP 更新。若要隨時掌握最新資訊，請參閱 SAP HANA 高可用性的 SAP Note 2407186 – 操作指南與白皮書。

其他 SAP 備註

- 2550327 – 如何重新命名 SAP HANA 系統
- 1999880 – 常見問答集：SAP HANA 系統複寫
- 2078425 – SAP HANA 平台生命週期管理工具 hdblcm 的故障診斷備註
- 2592227 – HANA 系統中的 FQDN 尾碼變更
- 2048681 – 在沒有 SSH 或根登入資料的多主機系統上執行 SAP HANA 平台生命週期管理管理任務

SAP 文件

- 系統複寫網路連線
- 系統複寫的主機名稱解析

AWS 參考

- 將 SAP HANA 從其他平台遷移至 AWS

其他資訊

`hdblcm` 作為主機名稱重新命名活動的一部分執行的變更會合併在下列詳細日誌中。

使用 遷移 Microsoft SQL Server Always On 可用性群組 AWS Application Migration Service

Sreenivas Nettem、Bharath Kumar Pammi Ramesh、Anantharaman Seshadri 和 Gireesh Sreekantan，Amazon Web Services

總結

AWS Application Migration Service (AWS MGN) 是重新託管 中現有環境的偏好工具 AWS 雲端，可讓客戶遠離內部部署資料中心。此模式概述使用 AWS MGN 透過 Microsoft SQL Server Always On 可用性群組遷移 Windows 叢集的程序。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- MGN 協同運作的 AWS AWS Identity and Access Management (IAM) 角色。
- 存取來源資料庫伺服器 (SQL Server Always On 可用性群組）。
- AWS 登陸區域中用於保留 DNS 名稱的 Active Directory。
- 與 Active Directory 進行封閉網路通訊的預備子網路。
- 可與 Active Directory 通訊的目標子網路。
- 目標子網路中 Windows 叢集的兩個預留 IP 地址（每個可用區域一個）。
- 目標子網路中 SQL Always On 接聽程式的兩個預留 IP 地址（每個可用區域一個）。

產品版本

- Windows Server 2012 或更新版本
- SQL Server 2012 或更新版本

Architecture

來源技術堆疊

Microsoft Windows 叢集（內部部署實體或虛擬機器） Microsoft SQL Server Always On 可用性群組

目標技術堆疊

Amazon EC2 Windows 執行個體

目標架構

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- AWS Application Migration Service 可協助您將應用程式重新託管（提升和轉移）到 ，AWS 雲端無需變更，且停機時間最短。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。

其他工具

- Microsoft SQL Server Management Studio (SSMS) 是一種用於管理 SQL Server 的工具，包括存取、設定和管理 SQL Server 元件。

最佳實務

對於 AWS MGN，請參閱 的最佳實務 AWS Application Migration Service。

史詩

準備目標帳戶

任務	Description	所需的技能
初始化 AWS MGN。	在目標中初始化 AWS MGN AWS 區域。這會建立所需的 IAM 角色和政策。如需詳細資訊，請參閱使用 主控台初始化 Application Migration Service。	雲端管理員

任務	Description	所需的技能
建立複寫和啟動範本。	設定複寫和啟動範本以搭配 AWS MGN 使用。如需詳細資訊，請參閱 AWS 文件中的設定範本。	雲端管理員
允許通訊連接埠。	若要啟用 AWS MGN 的網路通訊，請允許透過 TCP 連接埠 443 和 1500 的流量。如需詳細資訊，請參閱 AWS 文件中的 Application Migration Service 的網路需求。	雲端管理員、網路管理員

準備來源伺服器

任務	Description	所需的技能
驗證 AWS MGN 先決條件。	確認來源伺服器符合 AWS MGN 代理程式安裝的先決條件。如需詳細資訊，請參閱 AWS 文件中的安裝需求。	遷移工程師
安裝 AWS MGN 代理程式。	在來源伺服器上安裝 AWS MGN 代理程式。在安裝期間，選擇將遷移伺服器的 AWS 區域 。安裝後，代理程式會與服務通訊並開始複寫。如需詳細資訊，請參閱在 Windows 伺服器上安裝複 AWS 寫代理程式。	遷移工程師
檢查來源伺服器狀態。	在 AWS MGN 主控台中，檢查來源伺服器的狀態。伺服器會在複寫開始時顯示準備進行測試。	雲端管理員、遷移工程師

任務	Description	所需的技能
	如果您遇到任何錯誤，請參閱 AWS MGN 文件中的疑難排解通訊錯誤。	
最佳化複寫設定。	SQL Always On 叢集使用從主要伺服器到次要伺服器的高 I/O 同步複寫。若要最佳化複寫並避免延遲，請為每個 SQL Always On 伺服器使用專用的複寫伺服器。 如果資料庫大於 5 TB，請考慮選擇較大的複寫伺服器執行個體大小，例如 m5.large，而非預設的 t3.small。	雲端管理員、遷移工程師
更新啟動範本。	更新啟動設定，並選擇 SQL Always On 伺服器的子網路。SQL Always On 叢集伺服器會分散到不同的 ， AWS 可用區域 以獲得高可用性。	遷移工程師，遷移負責人
更新啟動設定。	根據您的大小和效能需求，更新啟動設定中的執行個體類型和每秒輸入/輸出操作 (IOPS)。 （選用）在啟動設定中選擇現有的彈性網路界面。	遷移工程師，遷移負責人

測試切換

任務	Description	所需的技能
驗證來源伺服器。	在 AWS MGN 主控台中，確認來源伺服器狀態為準備好進行測試。	雲端管理員、遷移工程師
啟動測試執行個體。	1. 啟動測試執行個體，並在 Amazon EC2 主控台中驗證自動檢查是否通過。 2. 選取並啟動見證伺服器的測試執行個體。 3. 確認您可以透過 AWS MGN 主控台登入伺服器。 4. 選取 SQL Always On 叢集伺服器，並一起啟動測試執行個體。	雲端管理員、遷移工程師
測試連線能力和資料庫完整性。	測試測試執行個體的連線和資料庫完整性。然後在 AWS MGN 主控台中將來源伺服器標記為準備好進行切換。	雲端管理員、遷移工程師

預遷移任務

任務	Description	所需的技能
測試資料庫完整性。	這有助於確保在遷移之前，來源沒有資料庫完整性問題。執行 `DBCC CHECKDB` 並指定 `WITH_PHYSICAL_ONLY` 。在沒有 的情況下執行此檢查 `WITH_PHYSICAL_ONLY` 可能會導致來源的效能問題。若	資料工程師，DBA

任務	Description	所需的技能
	要維護資料庫完整性，請每週對資料庫執行完整檢查。 這些命令會偵測潛在的損毀問題，以檢查資料庫的邏輯和實體完整性。檢查會驗證資料庫的結構，例如頁面、資料列、索引和系統資料表。	
測試與連結伺服器的連線。	測試所有現有伺服器之間的連線，並記錄其狀態。這有助於確保連結的伺服器在遷移後可如預期運作。	資料工程師，DBA
驗證備份。	確認來源備份的完整性。	資料工程師，DBA

AWS MGN 切換

任務	Description	所需的技能
停止 SQL Server 和叢集服務。	停止所有 SQL 叢集節點上的 SQL Server 和 Microsoft 叢集服務。	DBA，遷移工程師
驗證伺服器。	在 AWS MGN 主控台中，確認來源伺服器的狀態已就緒可供切換，且資料複寫狀態為良好。	遷移工程師
啟動切換。	1. 啟動見證伺服器的 AWS MGN 切換。 2. 啟動 SQL Always On 叢集執行個體的 AWS MGN 切換。	遷移工程師

任務	Description	所需的技能
	3. 確認狀態變更為正在進行中的切換。 如需詳細資訊，請參閱 AWS MGN 文件中的啟動切換執行個體。	
測試啟動的伺服器。	登入已啟動的 Amazon EC2 執行個體，並驗證叢集的運作狀態。確認伺服器位於正確的子網路中、執行個體大小和 IOPS 設定正確，而且可以存取見證伺服器。	DBA，遷移工程師

資料庫切換後任務

任務	Description	所需的技能
更新叢集 IP 地址。	使用目標子網路中的兩個預留 IP 地址更新 Windows 叢集的叢集 IP 地址。如需詳細資訊，請參閱變更容錯移轉叢集執行個體的 IP 地址。	DBA，遷移工程師
更新 Always On 可用性群組接聽程式 IPs。	1. 開啟容錯移轉叢集管理員。 2. 選擇 Always On 可用性群組角色。 3. 展開可用性群組接聽程式名稱。 4. 在內容（按一下滑鼠右鍵）功能表中，選擇 IP 地址屬性。	DBA，遷移工程師

任務	Description	所需的技能
	5. 使用目標子網路中預留給接聽程式的地址來更新 IP 地址。 6. 使用 SSMS 連線至 SQL Server 主要執行個體，並確認 Always On 接聽程式正在使用兩個子網路 IPs。	
驗證連線。	使用 SSMS，連線至 Always On 可用性群組接聽程式，並確認連線成功。	DBA，遷移工程師
檢查 Always On 可用性群組的運作狀態。	1. 導覽至可用群組資料夾，開啟內容（按一下滑鼠右鍵）選單，然後選擇顯示儀表板。 2. 對於所有複本，請確認同步狀態已同步。	DBA，遷移工程師
檢查錯誤日誌。	開啟錯誤日誌，並驗證針對 SQL Server 執行個體報告的任何錯誤。確保所有資料庫的復原都已完成。	DBA，遷移工程師
測試連結的伺服器。	測試任何連結伺服器的連線能力。如果發生任何連線問題，請確定可存取目標伺服器和連接埠。	DBA，遷移工程師

完成切換

任務	Description	所需的技能
完成切換。	驗證目標 SQL Always On 叢集之後，請透過 AWS MGN 主控	雲端管理員、遷移工程師

任務	Description	所需的技能
	台完成切換。這會停止來源伺服器的資料複寫，並捨棄複寫伺服器的資料。它也會移除複寫伺服器及其相關聯的資源。	

疑難排解

問題	解決方案
AWS MGN 疑難排解	如需常見問題和解決方案，請參閱 AWS MGN 文件中的故障診斷和常見問答集一節。

相關資源

AWS resources

- Option-1 Rehost - AWS Application Migration Service (AWS)
- 什麼是 AWS Application Migration Service？

SQL Server 資源

- 什麼是 SQL Server Management Studio (SSMS)？

其他資訊

如需將工作負載遷移至 的標準安全需求 AWS 雲端，請參閱 AWS 網站上的安全、身分和合規最佳實務。

使用分散式可用性群組將 SQL Server 遷移至 AWS

Praveen Marthala，Amazon Web Services

總結

Microsoft SQL Server Always On 可用性群組為 SQL Server 提供高可用性 (HA) 和災難復原 (DR) 解決方案。可用性群組包含接受讀取/寫入流量的主要複本，以及最多八個接受讀取流量的次要複本。可用性群組是在具有兩個或多個節點的 Windows Server 容錯移轉叢集 (WSFC) 上設定。

Microsoft SQL Server Always On 分散式可用性群組提供解決方案，可在兩個獨立的 WFSCs之間設定兩個不同的可用性群組。屬於分散式可用性群組的可用性群組不必位於相同的資料中心。一個可用性群組可以是內部部署，另一個可用性群組可以是位於不同網域中 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的 Amazon Web Services (AWS) Cloud。

此模式概述使用分散式可用性群組將屬於現有可用性群組一部分的內部部署 SQL Server 資料庫遷移至 SQL Server，並在 Amazon EC2 上設定可用性群組的步驟。透過遵循此模式，您可以將資料庫遷移至 AWS 雲端，並在切換期間將停機時間降至最低。在切換之後，AWS 會立即提供資料庫。您也可以使用此模式，將基礎作業系統從現場部署變更為 AWS，同時保留相同版本的 SQL Server。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- AWS Direct Connect 或 AWS Site-to-Site VPN
- 在 AWS 的兩個節點上安裝的相同 SQL Server 版本

產品版本

- SQL Server 2016 版及更新版本
- SQL Server Enterprise Edition

Architecture

來源技術堆疊

- 內部部署中具有 Always On 可用性群組的 Microsoft SQL Server 資料庫

目標技術堆疊

- AWS 雲端上 Amazon EC2 上具有 Always On 可用性群組的 Microsoft SQL Server 資料庫

遷移架構

術語

- WSFC 1 – 內部部署的 WSFC
- WSFC 2 – AWS 雲端上的 WSFC
- AG 1 – 第一個可用性群組，位於 WSFC 1 中
- AG 2 – 第二個可用性群組，位於 WSFC 2 中
- SQL Server 主要複本 – AG 1 中的節點，被視為所有寫入的全域主要節點
- SQL Server 轉送器 – AG 2 中的節點，以非同步方式從 SQL Server 主要複本接收資料
- SQL Server 次要複本 – AG 1 或 AG 2 中從主要複本或轉送器同步接收資料的節點

工具

- AWS Direct Connect – AWS Direct Connect 透過標準乙太網路光纖纜線，將您的內部網路連結至 AWS Direct Connect 位置。透過此連線，您可以直接建立與公有 AWS 服務的虛擬介面，繞過網路路徑中的網際網路服務供應商。
- Amazon EC2 – Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以使用 Amazon EC2 視需要啟動任意數量或任意數量的虛擬伺服器，也可以向外擴展或向內擴展。
- AWS Site-to-Site VPN – AWS Site-to-Site VPN 支援建立site-to-site虛擬私有網路 (VPN)。您可以設定 VPN 在 AWS 上啟動的執行個體與您自己的遠端網路之間傳遞流量。
- Microsoft SQL Server Management Studio – Microsoft SQL Server Management Studio (SSMS) 是管理 SQL Server 基礎設施的整合環境。它提供使用者介面和一組工具，其中包含與 SQL Server 互動的豐富指令碼編輯器。

史詩

在 AWS 上設定第二個可用性群組

任務	Description	所需的技能
在 AWS 上建立 WSFC。	在具有兩個 HA 節點的 Amazon EC2 執行個體上建立 WSFC 2。您將使用此容錯移轉叢集在 AWS 上建立第二個可用性群組 (AG 2)。	系統管理員、SysOps 管理員
在 WSFC 2 上建立第二個可用性群組。	使用 SSMS，在 WSFC 2 中的兩個節點上建立 AG 2。WSFC 2 中的第一個節點將做為轉送器。WSFC 2 中的第二個節點將充當 AG 2 的次要複本。 在此階段，AG 2 中沒有可用的資料庫。這是設定分散式可用性群組的起點。	DBA、開發人員
在 AG 2 上建立沒有復原選項的資料庫。	備份現場部署可用性群組 (AG 1) 上的資料庫。 在沒有復原選項的情況下，將資料庫還原至 AG 2 的轉寄站和次要複本。還原資料庫時，請為資料庫資料檔案和日誌檔案指定磁碟空間足夠的位置。 在此階段，資料庫處於還原狀態。它們不屬於 AG 2 或分散式可用性群組，而且不會同步。	DBA、開發人員

設定分散式可用性群組

任務	Description	所需的技能
在 AG 1 上建立分散式可用性群組。	若要在 AG 1 上建立分散式可用性群組，請使用 CREATE AVAILABILITY GROUP 搭配 DISTRIBUTED 選項。 1. 使用 AG 1 和 AG 2 的LISTENER_URL 端點地址。 2. 對於 AVAILABILITY-MODE，如果有的話，請使用 ASYNCHRONOUS_COMMIT 來避免網路延遲。這不會影響資料庫的效能。 3. 對於 FAILOVER_MODE ，請使用 MANUAL。這是唯一可與分散式可用性群組搭配使用的可用性模式。 4. 若要在 AG 2 上手動還原資料庫，並對大型資料庫有更多控制權，請MANUAL針對使用 SEEDING_MODE 。	DBA、開發人員
在 AG 2 上建立分散式可用性群組。	若要在 AG 2 上建立分散式可用性群組，請使用 ALTER AVAILABILITY GROUP 搭配 DISTRIBUTED 選項。 1. 使用 AG 1 和 AG 2 的LISTENER_URL 端點地址。 2. 對於 AVAILABILITY-MODE，如果有的話，請使用 ASYNCHRONOUS_COMMI	DBA、開發人員

任務	Description	所需的技能
	T 來避免網路延遲。這不會影響資料庫的效能。 3. 對於 FAILOVER_MODE ，請使用 MANUAL。這是唯一可與分散式可用性群組搭配使用的可用性模式。 4. 若要在 AG 2 上手動還原資料庫，並對大型資料庫有更多控制權，請MANUAL針對使用 SEEDING_MODE 。 分散式可用性群組是在 AG 1 和 AG 2 之間建立。 AG 2 中的資料庫尚未設定為參與從 AG 1 到 AG 2 的資料流程。	
將資料庫新增至 AG 2 上的轉送器和次要複本。	使用 ALTER DATABASE 搭配 AG 2 上轉送器和次要複本中的 SET HADRAVAILABILITY GROUP選項，將資料庫新增至分散式可用性群組。 這會在 AG 1 和 AG 2 上的資料庫之間啟動非同步資料流程。 全域主要會進行寫入、同步傳送資料至 AG 1 上的次要複本，以及非同步傳送資料至 AG 2 上的轉送器。AG 2 上的轉送器會將資料同步傳送至 AG 2 上的次要複本。	DBA、開發人員

監控 AG 1 和 AG 2 之間的非同步資料流程

任務	Description	所需的技能
使用 DMVs和 SQL Server 日誌。	使用動態管理檢視 (DMVs) 和 SQL Server 日誌，監控兩個可用群組之間的資料流程狀態。 需要監控的 DMVs 包括 `sys.dm_hadr_availability_replica_states` 和 `sys.dm_hadr_automatic_seeding` 。 對於轉送器同步狀態，請監控轉送器上 SQL Server 日誌中的同步狀態。	DBA、開發人員

執行最終遷移的切換活動

任務	Description	所需的技能
停止所有流向主要複本的流量。	停止傳入至 AG 1 中主要複本的流量，以便在資料庫上不會發生寫入活動，且資料庫已準備好進行遷移。	應用程式擁有者、開發人員
變更 AG 1 上分散式可用性群組的可用性模式。	在主要複本上，將分散式可用性群組的可用性模式設定為同步。 將可用性模式變更為同步後，資料會從 AG 1 中的主要複本同步傳送至 AG 2 中的轉寄站。	DBA、開發人員

任務	Description	所需的技能
檢查兩個可用群組中的 LSNs。	檢查 AG 1 和 AG 2 中的最後一個日誌序號 (LSNs)。由於 AG 1 的主要複本中沒有發生寫入，因此資料會同步，且兩個可用性群組的最後一個 LSNs 應相符。	DBA、開發人員
將 AG 1 更新為次要角色。	當您將 AG 1 更新為次要角色時，AG 1 會失去主要複本角色，且不接受寫入，且兩個可用群組之間的資料流程會停止。	DBA、開發人員

容錯移轉至第二個可用性群組

任務	Description	所需的技能
手動容錯移轉至 AG 2。	在 AG 2 中的轉送器上，變更分散式可用性群組以允許資料遺失。由於您已檢查並確認 AG 1 和 AG 2 上的最後一個 LSNs 相符，因此資料遺失並非問題。 當您在 AG 2 中允許轉送器上的資料遺失時，AG 1 和 AG 2 的角色會變更： • AG 2 會成為具有主要複本和次要複本的可用性群組。 • AG 1 會成為具有轉送器和次要複本的可用性群組。	DBA、開發人員
變更 AG 2 上分散式可用性群組的可用性模式。	在 AG 2 的主要複本上，將可用性模式變更為非同步。	DBA、開發人員

任務	Description	所需的技能
	這會將資料從 AG 2 移動到 AG 1，從同步移動到非同步。此步驟是必要的，以避免 AG 2 和 AG 1 之間的網路延遲，如果有的話，而且不會影響資料庫的效能。	
開始將流量傳送至新的主要複本。	更新連線字串，以使用 AG 2 上的接聽程式 URL 端點將流量傳送至資料庫。 AG 2 現在接受寫入並將資料傳送至 AG 1 中的轉寄站，以及將資料傳送至 AG 2 中自己的次要複本。資料從 AG 2 非同步移動到 AG 1。	應用程式擁有者、開發人員

執行切換後活動

任務	Description	所需的技能
在 AG 2 上捨棄分散式可用性群組。	監控遷移的計劃時間量。然後捨棄 AG 2 上的分散式可用性群組，以移除 AG 2 和 AG 1 之間的分散式可用性群組設定。這會移除分散式可用性群組組態，而且從 AG 2 到 AG 1 的資料流程會停止。 此時，AG 2 可在 AWS 上高度使用，主要複本需要寫入，次要複本位於相同的可用性群組中。	DBA、開發人員

任務	Description	所需的技能
停用內部部署伺服器。	在屬於 AG 1 的 WSFC 1 中停用內部部署伺服器。	系統管理員、SysOps 管理員

相關資源

- 分散式可用性群組
- SQL 文件：分散式可用性群組
- SQL 文件： Always On 可用性群組：高可用性和災難復原解決方案

在 上將關聯式資料庫遷移至 MongoDB Atlas AWS

Battulga Purevragchaa 和 Igor Alekseev，Amazon Web Services

Babu Srinivasan，MongoDB

摘要

此模式說明從關聯式資料庫遷移到 中 MongoDB Atlas 的步驟，例如 SQL Server、MySQL 或 PostgreSQL AWS 雲端。它使用 MongoDB Relational Migrator 協助加速從關聯式資料庫到 MongoDB Atlas 的資料遷移。

模式隨附於 AWS 上的遷移至 MongoDB Atlas AWS 指南指南網站上的指南。它提供該指南中討論的其中一個遷移案例的實作步驟。如需其他遷移案例，請參閱 AWS 規範指引網站上的下列模式：

- 在 上將自我託管的 MongoDB 環境遷移至 MongoDB Atlas AWS
- 在 上將資料從 IBM Db2、SAP、Sybase 和其他資料庫串流至 MongoDB Atlas AWS

此模式 適用於 AWS System Integrator (SI) 合作夥伴和 AWS 使用者。

先決條件和限制

先決條件

- 要遷移至 MongoDB Atlas 的來源關聯式資料庫 (Oracle Database、SQL Server、PostgreSQL、MySQL、SAP/Sybase ASE 等）。
- 熟悉關聯式資料庫、MongoDB Atlas 和 AWS 服務。此模式說明一些高層級的遷移步驟。其他詳細資訊將在未來的版本中新增。

產品版本

- MongoDB 5.0 版或更新版本

Architecture

下圖顯示從關聯式資料庫管理系統 (RDBMS) 資料庫遷移至 MongoDB Atlas 的作業 AWS。

如需支援不同使用案例的 MongoDB Atlas 參考架構，請參閱 AWS Prescriptive Guidance 網站上的遷移至 上的 MongoDB Atlas AWS。

工具

- MongoDB Atlas 是全受管資料庫即服務 (DBaaS)，用於在雲端中部署和管理 MongoDB 資料庫。
- MongoDB Relational Migrator 可將資料從傳統關聯式資料庫順暢地轉換為 MongoDB。它有助於自動化轉換程序，並將關聯式資料庫的結構化資料模型轉換為 MongoDB 提供的彈性文件格式。Relational Migrator 會保留資料完整性和關係，以簡化遷移。組織可以利用 MongoDB 提供的可擴展性、效能和多樣性優勢，同時保持現有資料的熟悉度。

最佳實務

如需在 上使用 MongoDB 的最佳實務 AWS，請參閱 AWS 合作夥伴網路部落格上的文章。

史詩

探索和評估

任務	Description	所需的技能
決定關聯式資料庫的參數和大小。	使用關聯式 Migrator 建議和來自 的資訊來估計工作集大小，`db.stats()` 以取得總索引空間。假設經常存取您資料空間的百分比。此任務大約需要一週的時間。如需此範例和其他案例的詳細資訊和範例，請參閱相關資源一節。	應用程式擁有者，DBA
估計網路頻寬需求。	若要估算您的網路頻寬需求，請將平均文件大小乘以每秒提供的文件數量。請考慮叢集上任何節點所承擔的最大流量。若要計算從叢集到用戶端應用程式的下游資料傳輸率，請使用一段時間內傳回文件總數	DBA

任務	Description	所需的技能
	的總和。如果您的應用程式從次要節點讀取，請將總文件數除以可提供讀取操作的節點數。若要尋找資料庫的平均文件大小，請使用 `db.stats().avgObjSize` 命令。此任務通常需要一天的時間。	
選取 Atlas 層。	遵循 MongoDB 文件中的指示，選取正確的 Atlas 叢集層。	DBA
規劃切換。	規劃應用程式切換。	DBA、應用程式擁有者

在 上設定新的 MongoDB Atlas 環境 AWS

任務	Description	所需的技能
在 上建立新的 MongoDB Atlas 叢集 AWS。	在 MongoDB Atlas 中，選擇建置叢集。在建立新叢集對話方塊中，選取 AWS 做為雲端供應商。	DBA
選取 AWS 區域 和 全域叢集組態。	從 AWS 區域 Atlas 叢集可用的清單中選取 。視需要設定全域叢集。	DBA
選取叢集層。	選取您偏好的叢集層。您的方案選擇會決定記憶體、儲存體和 IOPS 規格等因素。	DBA
設定其他叢集設定。	設定其他叢集設定，例如 MongoDB 版本、備份和加密選項。如需這些選項的詳細資訊，請參閱相關資源一節。	DBA

設定安全與合規

任務	Description	所需的技能
設定存取清單。	若要連線至 Atlas 叢集，您必須將項目新增至專案的存取清單。Atlas 使用 TLS/SSL 來加密與資料庫虛擬私有雲端 (VPC) 的連線。若要設定專案的存取清單，以及有關此史詩中故事的詳細資訊，請參閱相關資源一節。	DBA
驗證和授權使用者。	您必須建立和驗證將存取 MongoDB Atlas 叢集的資料庫使用者。若要存取專案中的叢集，使用者必須屬於該專案，而且可以屬於多個專案。	DBA
建立自訂角色。	（選用）在內建 Atlas 資料庫使用者許可未涵蓋您想要的許可集的情況下，Atlas 支援建立自訂角色。	DBA
設定 VPC 對等互連。	（選用）Atlas 支援與其他 VPC 對等互連 AWS。 VPCs	AWS 管理員
設定 AWS PrivateLink 端點。	（選用）您可以使用 在 上設定私有端點 AWS AWS PrivateLink。如需詳細資訊，請參閱 Amazon VPC 文件。	AWS 管理員
啟用雙重驗證。	（選用）Atlas 支援雙重驗證 (2FA)，以協助使用者控制對其 Atlas 帳戶的存取。	AWS 管理員

任務	Description	所需的技能
使用 LDAP 設定使用者身分驗證和授權。	（選用） Atlas 支援使用輕量型目錄存取通訊協定 (LDAP) 執行使用者身分驗證和授權。	DBA
設定統一 AWS 存取。	（選用） 某些 Atlas 功能，包括使用客戶金鑰管理的 Atlas Data Lake 和靜態加密，使用 AWS Identity and Access Management (IAM) 角色進行身分驗證。	AWS 管理員
使用 設定靜態加密 AWS KMS。	（選用） Atlas 支援使用 AWS Key Management Service (AWS KMS) 加密儲存引擎和雲端供應商備份。	AWS 管理員
設定用戶端欄位層級加密。	（選用） Atlas 支援用戶端欄位層級加密，包括欄位的自動加密。	AWS 管理員

遷移資料

任務	Description	所需的技能
將 MongoDB Relational Migrator 新增至存取清單。	將 Relational Migrator 新增至來源資料庫的存取清單。這有助於準備來源環境以連線到目標 Atlas 叢集。	DBA
評估關聯式資料庫物件。	啟動 MongoDB Relational Migrator 並連線至您的關聯式資料庫。開始評估。	DBA
接受遷移模式，或選擇根據您的業務需求進行修改。	您可以接受 Relational Migrator 根據初始評估和效能參數建議	DBA

任務	Description	所需的技能
	的資料庫模式，或選擇根據您的業務需求進行變更。	
在 MongoDB Atlas 中啟動您的目標複本集。	在 MongoDB Atlas 中啟動您的目標複本集。在關聯式 Migrator 中，選擇我已準備好遷移。	DBA

設定操作整合

任務	Description	所需的技能
連線至 MongoDB Atlas 叢集。	請確定 MongoDB Atlas 叢集連線如預期般運作。	應用程式擁有者
與叢集資料互動。	驗證叢集資料。	DBA
監控您的叢集。	確認您的叢集設定正確。	DBA
備份和還原叢集資料。	定期排程叢集資料的備份。	DBA

相關資源

除非另有說明，否則下列所有連結都會移至 MongoDB 文件中的網頁。

遷移指南

- 在 上遷移至 MongoDB Atlas AWS (AWS 方案指引)

探索和評估

- 記憶體
- 使用 Atlas 範例資料集調整大小範例
- 行動應用程式的大小調整範例
- 網路流量

- 叢集自動擴展
- Atlas 大小調整範本

設定安全與合規

- 設定 IP 存取清單項目
- 設定資料庫使用者
- 設定對 Atlas UI 的存取
- 設定自訂資料庫角色
- 設定資料庫使用者
- 設定網路對等連線
- 了解 Atlas 中的私有端點
- 管理您的多重要素驗證選項
- 使用 LDAP 設定使用者身分驗證和授權
- Atlas Data Lake
- 使用客戶金鑰管理進行靜態加密
- 擔任角色的方法 (IAM 文件)
- 用戶端欄位層級加密
- 自動加密
- MongoDB Atlas 安全控制
- MongoDB 信任中心
- 設定叢集的安全功能

在 上設定新的 MongoDB Atlas 環境 AWS

- 雲端供應商和區域
- 管理全域叢集
- 選取叢集層
- 設定其他設定
- Atlas 入門
- 設定對 Atlas UI 的存取

遷移資料

- 遷移或匯入資料

監控叢集

- 監控您的叢集

整合 操作

- 連線至叢集
- 與您的資料互動
- 監控您的叢集
- 備份、還原和封存資料

在 上將自我託管的 MongoDB 環境遷移至 MongoDB Atlas AWS

Battulga Purevragchaa 和 Igor Alekseev，Amazon Web Services

Babu Srinivasan，MongoDB

摘要

此模式說明從自我管理的 MongoDB 環境 （ 包括 MongoDB Community Server、Enterprise Server、Enterprise Advanced、mLab 或任何受管 MongoDB 叢集 ） 遷移至 中的 MongoDB Atlas 的步驟 AWS 雲端。它使用 Atlas Live Migration Service 來協助加速從 MongoDB 到 MongoDB Atlas 的資料遷移。

模式隨附於 AWS 上的遷移至 MongoDB Atlas AWS 指南指南網站上的指南。它提供該指南中討論的其中一個遷移案例的實作步驟。如需其他遷移案例，請參閱 AWS 規範指引網站上的下列模式：

- 在 上將關聯式資料庫遷移至 MongoDB Atlas AWS
- 在 上將資料從 IBM Db2、SAP、Sybase 和其他資料庫串流至 MongoDB Atlas AWS

此模式適用於 AWS Systems Integrator (SI) 合作夥伴和 AWS 使用者。

先決條件和限制

先決條件

- 要遷移至 MongoDB Atlas 的來源 MongoDB Enterprise Advanced、Community Server 或其他自我管理的 MongoDB 環境。
- 熟悉 MongoDB、MongoDB Atlas 和 AWS 服務。此模式說明一些高層級的遷移步驟。其他詳細資訊將在未來的版本中新增。

產品版本

- MongoDB 6.0.13 版或更新版本

Architecture

下圖顯示 Atlas Live Migration Service，用於將資料從 MongoDB Enterprise Advanced 資料庫和 MongoDB Community 資料庫遷移至 MongoDB Atlas AWS。當您需要將大型、複雜的資料庫遷移

至 MongoDB Atlas 時，請使用此服務，將停機時間和持續資料同步降到最低。此模式使用 Atlas Live Migration Service。

下圖顯示 MongoDB 鏡像服務 (mongomirror)，您也可以用來 AWS 透過安全AWS PrivateLink連線將資料從 MongoDB Enterprise Advanced 資料庫和 MongoDB Community 資料庫遷移到 上的 MongoDB Atlas。mongomirror 用於內部部署 MongoDB 和 MongoDB Atlas 之間的持續資料複寫。此工具非常適合災難復原或分階段遷移，但超出此模式的範圍。

如需支援不同使用案例的更多 MongoDB Atlas 參考架構，請參閱 AWS 《 方案指引》網站上的遷移至 MongoDB Atlas AWS。

工具

- MongoDB Atlas 是全受管資料庫即服務 (DbaaS)，用於在雲端中部署和管理 MongoDB 資料庫。
- Atlas Live Migration Service 是免費的 MongoDB 公用程式，可協助將資料庫遷移至 Atlas。此服務會讓來源資料庫與目的地資料庫保持同步，直到切換為止。當您準備好切換時，您可以停止應用程式執行個體，將它們指向目的地 Atlas 叢集，然後重新啟動它們。若要存取此服務，請從 MongoDB Atlas 叢集中選擇資料庫選項。
- mongomirror 是一種工具，可將現有 MongoDB 複本集的資料手動遷移至 MongoDB Atlas 複本集。mongomirror不需要您關閉現有的複本集或應用程式、不匯入使用者或角色資料，或複製組態資料庫。您可以從 mongomirror MongoDB 文件下載 。

最佳實務

如需在 上使用 MongoDB 的最佳實務 AWS，請參閱 AWS 合作夥伴網路部落格上的文章。

史詩

探索和評估

任務	Description	所需的技能
決定叢集大小。	使用來自 的資訊估計工作集大小db.stats() ，以取得總索引空間。假設經常存取您資料空間的百分比。或者，您可以	DBA、應用程式擁有者

任務	Description	所需的技能
	根據自己的假設來預估記憶體需求。此任務大約需要一週的時間。如需此範例和其他案例的詳細資訊和範例，請參閱相關資源一節。	
估計網路頻寬需求。	若要估算您的網路頻寬需求，請將平均文件大小乘以每秒提供的文件數量。請考慮叢集上任何節點所承擔的最大流量。若要計算從叢集到用戶端應用程式的下游資料傳輸率，請使用一段時間內傳回文件總數的總和。如果您的應用程式從次要節點讀取，請將總文件數除以可提供讀取操作的節點數。若要尋找資料庫的平均文件大小，請使用 `db.stats().avgObjSize` 命令。此任務通常需要一天的時間。	DBA
選取 Atlas 層。	遵循 MongoDB 文件中的指示，選取正確的 Atlas 叢集層。	DBA
規劃切換。	規劃應用程式切換。	DBA、應用程式擁有者

在 AWS 上設定新的 MongoDB Atlas 環境

任務	Description	所需的技能
在 上建立新的 MongoDB Atlas 叢集 AWS。	登入 Atlas 並開啟專案的概觀頁面。選擇建立按鈕來建立叢集。如需詳細資訊，請參閱 MongoDB 文件。	DBA

任務	Description	所需的技能
選取 AWS 區域 和 全域叢集組態。	從 AWS 區域 Atlas 叢集可用的清單中選取 。視需要設定全域叢集。如需詳細資訊，請參閱 MongoDB 文件。	DBA
選取叢集層。	選取您偏好的叢集層。您的方案選擇會決定記憶體、儲存體和 IOPS 規格等因素。	DBA
設定其他叢集設定。	設定其他叢集設定，例如 MongoDB 版本、備份和加密選項。如需這些選項的詳細資訊，請參閱相關資源一節。	DBA

設定安全與合規

任務	Description	所需的技能
驗證和授權使用者。	您必須建立和驗證將存取 MongoDB Atlas 叢集的資料庫使用者。若要存取專案中的叢集，使用者必須屬於該專案，而且可以屬於多個專案。 Atlas 也支援以 AWS Identity and Access Management (IAM) 為基礎的身分驗證。如需詳細資訊，請參閱 MongoDB 文件。	DBA
建立自訂角色。	（選用）在內建 Atlas 資料庫使用者許可未涵蓋您想要的許可集的情況下，Atlas 支援建立自訂角色。	DBA

任務	Description	所需的技能
設定 VPC 對等互連。	（選用）Atlas 支援虛擬私有雲端 (VPC) 與其他 VPC 對等互連 AWS。 VPCs	AWS 管理員
設定 AWS PrivateLink 端點。	（選用）您可以使用 在 上設定私有端點 AWS AWS PrivateLink。如需詳細資訊，請參閱 Amazon VPC 文件。	AWS 管理員
啟用雙重驗證。	（選用）Atlas 支援雙重驗證 (2FA)，以協助使用者控制對其 Atlas 帳戶的存取。	AWS 管理員
使用 LDAP 設定使用者身分驗證和授權。	（選用）Atlas 支援使用輕量型目錄存取通訊協定 (LDAP) 執行使用者身分驗證和授權。	AWS 管理員
設定統一 AWS 存取。	（選用）某些 Atlas 功能，包括使用客戶金鑰管理的 Atlas Data Lake 和靜態加密，請使用 IAM 角色進行身分驗證。	AWS 管理員
使用 設定靜態加密 AWS KMS。	（選用）Atlas 支援使用 AWS Key Management Service (AWS KMS) 加密儲存引擎和雲端供應商備份。	AWS 管理員
設定用戶端欄位層級加密。	（選用）Atlas 支援用戶端欄位層級加密，包括欄位的自動加密。	AWS 管理員

遷移資料

任務	Description	所需的技能
在 MongoDB Atlas 中選取您的目標複本集。	導覽至目的地 Atlas 叢集，然後選擇省略符號 (...) 按鈕。在叢集清單中，此按鈕會出現在叢集名稱下方。在叢集詳細資訊中，按鈕會出現在右側連線和組態按鈕旁。如需詳細資訊，請參閱 MongoDB 文件。	DBA
將 Atlas Live Migration Service 新增至存取清單。	將 Atlas Live Migration Service 新增至 AWS 來源叢集中的存取清單。這有助於準備來源環境以連線到目標 Atlas 叢集。	DBA
使用 Atlas Live Migration Service 執行遷移。	選擇開始遷移。當準備切換按鈕變成綠色時，請執行切換。檢閱 Atlas 叢集效能指標。考慮更新所有應用程式層中的資料庫連線，以指向新的資料庫。	DBA

設定操作整合

任務	Description	所需的技能
連線至 MongoDB Atlas 叢集。	確定 MongoDB Atlas 叢集連線如預期般運作。	應用程式擁有者
與叢集資料互動。	測試叢集資料。	DBA
監控您的叢集。	確認您的叢集已正確設定。	DBA
備份和還原叢集資料。	定期排程叢集資料的備份。	DBA

疑難排解

問題	解決方案

錯誤：無法連線到指定的來源	• 請確定您已將正確的子網路範圍新增至來源叢集上的 IP 存取清單。您可以在即時遷移模態視窗中找到四個必要的子網路範圍。
	• 確認您指定的主機名稱解析為公有 IP 地址。在命令提示字元中，使用下列其中一個命令：
	`nslookup <hostname>`
	`ping <hostname>`
	• 請確定您沒有使用與提取即時遷移不相容的 VPC 互連連線。如果 VPC 互連連線是您唯一的選項，請`mongomirror` 改用 。

錯誤：無法解析主機名稱	找不到指定主機名稱的 IP 地址。確認指定的主機名稱正確且可公開存取。
任何其他錯誤	如果您遇到任何其他錯誤，請參閱 MongoDB 文件中的即時遷移 (Pull) 故障診斷。

相關資源

除非另有說明，否則下列所有連結都會移至 MongoDB 文件中的網頁。

遷移指南

- 在 上遷移至 MongoDB Atlas AWS (AWS 方案指引)

舊版遷移

- MongoDB 舊版的遷移

探索和評估

- 記憶體

- 使用 Atlas 範例資料集調整大小範例
- 行動應用程式的大小調整範例
- 網路流量
- 叢集自動擴展
- Atlas 大小調整範本

設定安全與合規

- 設定 IP 存取清單項目
- 設定資料庫使用者
- 設定對 Atlas UI 的存取
- 設定自訂資料庫角色
- 設定資料庫使用者
- 設定網路對等連線
- 了解 Atlas 中的私有端點
- 管理您的多重要素驗證選項
- 使用 LDAP 設定使用者身分驗證和授權
- Atlas Data Lake
- 使用客戶金鑰管理進行靜態加密
- 擔任角色的方法 (IAM 文件)
- 用戶端欄位層級加密
- 自動加密
- MongoDB Atlas 安全控制
- MongoDB 信任中心
- 設定叢集的安全功能

在 上設定新的 MongoDB Atlas 環境 AWS

- 雲端供應商和區域
- 管理全域叢集
- 選取叢集層
- 設定其他設定

- Atlas 入門
- 設定對 Atlas UI 的存取

遷移資料

- 遷移或匯入資料

監控叢集

- 監控您的叢集

整合 操作

- 連線至叢集
- 與您的資料互動
- 監控您的叢集
- 備份、還原和封存資料

訓練

- 使用 MongoDB Atlas 即時遷移

其他資訊

如需詳細資訊，請參閱 MongoDB 文件中的下列主題：

- 若要將資料移至無伺服器執行個體，請使用 Compass 匯出和匯入資料，或使用自我管理工具遷移資料。若要進一步了解，請參閱無伺服器執行個體限制。
- 若要將資料載入 Atlas 中的新叢集，請參閱將資料載入 Atlas。
- 若要複製叢集以供測試之用，請參閱自我管理部署的備份方法。
- 如果您想要遷移的應用程式需要接近連續的執行時間，請聯絡 MongoDB Support 並分享您的執行時間需求和叢集組態。
- 如需詳細資訊，請參閱遷移或匯入資料。

使用 AWS DMS 將 Oracle 資料庫遷移至 Amazon DynamoDB

Rambabu Karnena，Amazon Web Services

總結

此模式會逐步引導您使用 AWS Database Migration Service (AWS DMS) 將 Oracle 資料庫遷移至 Amazon DynamoDB 的步驟。它涵蓋三種類型的來源資料庫：

- 內部部署 Oracle 資料庫
- Amazon Elastic Compute Cloud (Amazon EC2) 上的 Oracle 資料庫
- Oracle 資料庫執行個體的 Amazon Relational Database Service (Amazon RDS)

在此概念驗證中，此模式著重於從 Amazon RDS for Oracle 資料庫執行個體遷移。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 連線至 Amazon RDS for Oracle 資料庫的應用程式
- 在來源 Amazon RDS for Oracle 資料庫中建立的資料表，其中包含主索引鍵和範例資料

限制

- 由於 Amazon DynamoDB 不支援這些資料庫物件，因此不會考慮遷移 Oracle 資料庫物件，例如程序、函數、套件和觸發程序。

產品版本

- 此模式適用於 AWS DMS 支援的所有 Oracle 資料庫版本。如需詳細資訊，請參閱使用 Oracle 資料庫做為 AWS DMS 的來源，以及使用 Amazon DynamoDB 資料庫做為 AWS DMS 的目標。我們建議您使用最新版本的 AWS DMS，以獲得最全面的版本和功能支援。

Architecture

來源技術堆疊

- Amazon RDS for Oracle 資料庫執行個體、Amazon EC2 上的 Oracle 或內部部署 Oracle 資料庫

目標技術堆疊

- Amazon DynamoDB

AWS 資料遷移架構

工具

- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。此模式使用 Amazon RDS for Oracle。

史詩

規劃遷移

任務	Description	所需的技能
建立 VPC。	在您的 AWS 帳戶中，建立虛擬私有雲端 (VPC) 和私有子網路。	系統管理員
建立安全群組和網路存取控制清單。	如需詳細資訊，請參閱 AWS 文件。	系統管理員
設定和啟動 Amazon RDS for Oracle 資料庫執行個體。	如需詳細資訊，請參閱 AWS 文件。	DBA，系統管理員

遷移資料

任務	Description	所需的技能
建立 IAM 角色以存取 DynamoDB。	在 AWS Identity and Access Management (IAM) 主控台中，建立角色、連接政策 `AmazonDynamoDBFullAccess to it` ，然後選取 AWS DMS 做為服務。	系統管理員
建立 AWS DMS 複寫執行個體以進行遷移。	複寫執行個體應與來源資料庫位於相同的可用區域和 VPC 中。	系統管理員
在 AWS DMS 中建立來源和目標端點。	若要建立來源資料庫端點，您有兩個選項： • 在 Amazon RDS 主控台上，選擇資料庫、資料庫識別符、連線與安全性，然後選擇端點。 • 在 AWS DMS 主控台上，選擇選取 RDS 資料庫執行個體。 若要建立目標資料庫端點，請從先前的任務中選擇角色 Amazon Resource Name (ARN)，以存取 DynamoDB。	系統管理員
建立 AWS DMS 任務，將來源 Oracle 資料庫資料表載入 DynamoDB。	從先前的步驟中選擇來源和目的地端點名稱，以及複寫執行個體。類型可以是完全載入。選擇 Oracle 結構描述，並指定 % 來選取所有資料表。	系統管理員

任務	Description	所需的技能
驗證 DynamoDB 中的資料表。	若要檢視遷移結果，請從 DynamoDB 主控台的左側導覽窗格中選擇資料表。	DBA

遷移應用程式

任務	Description	所需的技能
修改應用程式程式碼。	若要連線至 DynamoDB 並從 DynamoDB 擷取資料，請更新應用程式程式碼。	應用程式擁有者、DBA、系統管理員

切換

任務	Description	所需的技能
切換應用程式用戶端以使用 DynamoDB。		DBA、應用程式擁有者、系統管理員

關閉專案

任務	Description	所需的技能
關閉 AWS 資源。	例如， 關閉 Amazon RDS for Oracle 執行個體、DynamoDB 和 AWS DMS 複寫執行個體。	DBA，系統管理員
收集指標。	指標包括遷移時間、手動工作和工具執行工作的百分比，以及節省成本。	DBA、應用程式擁有者、系統管理員

相關資源

- AWS Database Migration Service 和 Amazon DynamoDB：須知事項（部落格文章）
- 使用 Oracle 資料庫做為 AWS DMS 的來源
- 使用 Amazon DynamoDB 資料庫做為 AWS Database Migration Service 的目標
- 從 RDBMS 遷移至 Amazon DynamoDB 的最佳實務（白皮書）

使用 SharePlex 和 AWS DMS 從 Oracle 8i 或 9i 遷移至 Amazon RDS for Oracle

Ramu Jagini，Amazon Web Services

總結

此模式說明如何將內部部署 Oracle 8i 或 9i 資料庫遷移至適用於 Oracle 資料庫的 Amazon Relational Database Service (Amazon RDS)。您可以使用此模式，透過使用 Quest SharePlex 進行同步複寫，以縮短停機時間來完成遷移。

您必須使用中繼 Oracle 資料庫執行個體進行遷移，因為 AWS Database Migration Service (AWS DMS) 不支援 Oracle 8i 或 9i 作為來源環境。您可以使用 SharePlex 7.6.3 從先前的 Oracle 資料庫版本複寫到更新的 Oracle 資料庫版本。中繼 Oracle 資料庫執行個體與 SharePlex 7.6.3 的目標相容，並支援做為 AWS DMS 的來源或較新版本的 SharePlex。此支援可將資料後續複寫至 Amazon RDS for Oracle 目標環境。

請考慮數種已棄用資料類型和功能可能會影響從 Oracle 8i 或 9i 遷移至最新版 Oracle 資料庫。為了減輕此影響，此模式使用 Oracle 11.2.0.4 做為中繼資料庫版本，以協助在遷移至 Amazon RDS for Oracle 目標環境之前最佳化結構描述程式碼。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署環境中的來源 Oracle 8i 或 9i 資料庫
- Amazon Elastic Compute Cloud (Amazon EC2) 上的預備 Oracle 資料庫 12c 版本 2 (12CR2)Amazon EC2
- Quest SharePlex 7.6.3 （商業等級）

限制

- RDS for Oracle 限制

產品版本

- 來源資料庫的 Oracle 8i 或 9i

- 預備資料庫的 Oracle 12CR2 （必須與 Amazon RDS for Oracle 版本相符）
- 目標資料庫的 Oracle 12CR2 或更新版本 (Amazon RDS for Oracle)

Architecture

來源技術堆疊

- Oracle 8i 或 9i 資料庫
- SharePlex

目標技術堆疊

- Amazon RDS for Oracle

遷移架構

下圖顯示如何將 Oracle 8i 或 9i 資料庫從內部部署環境遷移至 AWS 雲端中的 Amazon RDS for Oracle 資料庫執行個體。

該圖顯示以下工作流程：

1. 使用封存日誌模式、強制記錄和補充記錄來啟用 Oracle 來源資料庫。
2. 使用 Recovery Manager (RMAN) point-in-time復原和 FLASHBACK_SCN，從 Oracle 來源資料庫還原 Oracle 預備資料庫。
3. 設定 SharePlex 使用 FLASHBACK_SCN（用於 RMAN) 從 Oracle 來源資料庫讀取重做日誌。
4. 啟動 SharePlex 複寫，將資料從 Oracle 來源資料庫同步到 Oracle 預備資料庫。
5. 使用 EXPDP 和 IMPDP 搭配 來還原 Amazon RDS for Oracle 目標資料庫FLASHBACK_SCN。
6. 使用 FLASHBACK_SCN（用於 EXPDP) 將 AWS DMS 及其來源任務設定為 Oracle 預備資料庫，並將 Amazon RDS for Oracle 設定為目標資料庫。
7. 啟動 AWS DMS 任務，將資料從 Oracle 預備資料庫同步至 Oracle 目標資料庫。

工具

- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。

- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。
- Quest SharePlex 是一種 Oracle-to-Oracle的資料複寫工具，可在最短的停機時間下移動資料，而不會遺失資料。
- Recovery Manager (RMAN) 是 Oracle 資料庫用戶端，可在資料庫上執行備份和復原任務。它可大幅簡化資料庫檔案的備份、還原和復原。
- Data Pump Export 可協助您將資料和中繼資料上傳到一組稱為傾印檔案集的作業系統檔案中。傾印檔案集只能由 Data Pump Import 公用程式或 DBMS_DATAPUMP 套件匯入。

史詩

在 Amazon EC2 上設定 SharePlex 和 Oracle 預備資料庫

任務	Description	所需的技能
建立 EC2 執行個體。	1. 建立 EC2 執行個體。 2. 在 EC2 執行個體上安裝 Oracle 12CR2，以做為 Oracle 預備資料庫。 EC2	Oracle 管理
準備預備資料庫。	透過從 Oracle 8i 或 9i 資料庫來源環境取得 RMAN 備份，在 Oracle 12CR2 上準備還原為升級的 Oracle 預備資料庫。 如需詳細資訊，請參閱 Oracle 文件中的 Oracle 9i Recovery Manager 使用者指南和資料庫備份與復原使用者指南。	Oracle 管理
設定 SharePlex。	將 SharePlex 來源設定為內部部署 Oracle 8i 或 9i 資料庫，並將目標設定為 Amazon EC2 上託管的 Oracle 12CR2 預備資料庫。 Amazon EC2	SharePlex、Oracle 管理

將 Amazon RDS for Oracle 設定為您的目標環境

任務	Description	所需的技能
建立 Oracle 資料庫執行個體。	建立 Amazon RDS for Oracle 資料庫，然後將 Oracle 12CR2 連線至資料庫。 如需詳細資訊，請參閱《Amazon RDS 文件》中的建立 Oracle 資料庫執行個體並連線至 Oracle 資料庫執行個體上的資料庫。	DBA
從預備資料庫還原 Amazon RDS for Oracle。	1. 使用 從 Oracle 預備資料庫伺服器取得 EXPDP 備份`FLASHBACK_SCN` 。 2. 從預備資料庫還原 Amazon RDS for Oracle。 如需詳細資訊，請參閱 Oracle 文件中的 54 DBMS_DATA PUMP。	DBA

設定 AWS DMS

任務	Description	所需的技能
建立資料庫的端點。	為 Oracle 預備資料庫建立來源端點，並為 Amazon RDS for Oracle 資料庫建立目標端點。 如需詳細資訊，請參閱 AWS 知識中心中的如何使用 AWS DMS 建立來源或目標端點？。	DBA

任務	Description	所需的技能
建立複寫執行個體。	使用 AWS DMS 將 Oracle 預備資料庫的複寫執行個體啟動至 Amazon RDS for Oracle 資料庫。 如需詳細資訊，請參閱 AWS 知識中心中的如何建立 AWS DMS 複寫執行個體？。	DBA
建立並啟動複寫任務。	使用 FLASHBACK_SCN EXPDP （因為已透過 EXPDP 完全載入） 中的 建立變更資料擷取 (CDC) 的 AWS DMS 複寫任務。 如需詳細資訊，請參閱 AWS DMS 文件中的建立任務。	DBA

切換到 Amazon RDS for Oracle

任務	Description	所需的技能
停止應用程式工作負載。	在計劃的切換時段期間停止應用程式伺服器及其應用程式。	應用程式開發人員，DBA
驗證現場部署 Oracle 預備資料庫與 EC2 執行個體的同步。	透過在內部部署來源資料庫上執行幾個日誌切換，確認從 SharePlex 複寫執行個體複寫任務的所有訊息都已張貼到 Amazon EC2 上的 Oracle 預備資料庫。 如需詳細資訊，請參閱 Oracle 文件中的 6.4.2 切換日誌檔案。	DBA

任務	Description	所需的技能
使用 Amazon RDS for Oracle 資料庫驗證 Oracle 預備資料庫的同步。	確認您的所有 AWS DMS 任務沒有延遲和錯誤，然後檢查任務的驗證狀態。	DBA
停止 SharePlex 和 Amazon RDS 的複寫。	如果 SharePlex 和 AWS DMS 複寫都未顯示任何錯誤，則停止這兩個複寫。	DBA
將應用程式重新映射至 Amazon RDS。	與應用程式伺服器及其應用程式共用 Amazon RDS for Oracle 端點詳細資訊，然後啟動應用程式以繼續業務操作。	應用程式開發人員，DBA

測試 AWS 目標環境

任務	Description	所需的技能
在 AWS 上測試 Oracle 預備資料庫環境。	1. 測試 SharePlex 複寫，並確認 Oracle 預備資料庫中沒有同步差距或複寫錯誤。 2. 透過內部部署環境中定義的基準，確認應用程式的行為符合預期。	SharePlex、Oracle 管理
測試 Amazon RDS 環境。	1. 確認複寫後傳播至 Amazon RDS 的所有資料均無錯誤。 2. 將另一個應用程式指向 Amazon RDS 資料庫執行個體，然後執行效能測試以驗證預期的行為。 如需詳細資訊，請參閱 Amazon RDS 文件中的 Amazon RDS for Oracle。	Oracle 管理

相關資源

- 安心遷移
- Amazon EC2
- Amazon RDS for Oracle
- AWS Database Migration Service
- 偵錯 AWS DMS 遷移：發生錯誤時該怎麼做（第 1 部分）
- 偵錯 AWS DMS 遷移：發生錯誤時該怎麼做（第 2 部分）
- 偵錯 AWS DMS 遷移：發生問題時該怎麼辦？（第 3 部分）
- 用於資料庫複寫的 SharePlex
- SharePlex：任何環境的資料庫複寫

將內部部署 MySQL 資料庫遷移至 Amazon EC2

Lorenzo Mota，Amazon Web Services

摘要

此模式提供將內部部署 MySQL 資料庫遷移至 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上 MySQL 資料庫的指引。模式討論使用 AWS Database Migration Service (AWS DMS) 或原生 MySQL 工具進行遷移，例如 mysqldump。它著重於完整資料庫遷移至 MySQL 資料庫執行個體。

模式主要用於DBAs解決方案架構師。它可用於小型或大型專案、測試或最終遷移階段。建議您在生產環境中使用此模式之前，至少執行一個測試週期。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署資料中心中的 MySQL 來源資料庫

產品版本

- MySQL 5.5 版及更新版本
- Amazon EC2 支援的目標作業系統；請參閱 Amazon EC2 FAQs

Architecture

來源 技術堆疊

- 內部部署 MySQL 資料庫

目標 技術堆疊

- Amazon EC2 上的 MySQL 資料庫執行個體

AWS 資料遷移方法

- AWS DMS

- 原生 MySQL 工具，例如 mysqldump，或第三方工具，例如 Percona XtraBackup

目標架構

下圖說明切換後的目標 Amazon EC2 實作。

AWS 資料遷移架構

使用 AWS DMS：

下圖說明資料遷移工作流程，以 AWS DMS 為基礎，將完整和增量變更傳送至目標 MySQL 資料庫，直到切換為止。從內部部署到 的網路連線 AWS 取決於 SQL 用戶端的需求，且超出此模式的範圍。

使用其他 MySQL 工具：

下圖說明使用 MySQL 工具從內部部署資料庫產生匯出傾印檔案的資料遷移工作流程。這些檔案會移至 Amazon Simple Storage Service (Amazon S3)，並在切換之前匯入目標 MySQL 資料庫。從內部部署到 的網路連線 AWS 取決於 SQL 用戶端的需求，且超出此模式的範圍。

備註：

- 根據停機時間考量和最終切換的資料庫大小，您可以使用 AWS DMS 或其他變更資料擷取 (CDC) 工具，將切換時間降至最低。當您使用 等 CDC 工具時 AWS DMS，您可以在幾分鐘內遷移至目標資料庫。
- 如果資料庫大小和網路延遲允許短暫的切換遷移時段，則 mysqldump 的離線策略就足夠了。（我們建議您執行測試以取得大約的時間。）
- 通常，與離線選項相比，透過 的 CDC 策略 AWS DMS 需要更多的監控和複雜性。

工具

AWS 服務

- AWS Database Migration Service (AWS DMS) 支援數個來源和目標資料庫。如需 支援的 MySQL 來源和目標資料庫的相關資訊 AWS DMS，請參閱使用 MySQL 相容資料庫做為 的來源 AWS DMS和使用 MySQL 相容資料庫做為 的目標 AWS DMS。如果您的來源資料庫不受 支援 AWS DMS，您必須選擇其他方法來遷移資料。

其他工具

- mysqldump 是一種 MySQL 公用程式，可從 MySQL 資料庫建立傾印檔案，以供備份或遷移之用。
- Percona XtraBackup 是一種開放原始碼公用程式，可在 MySQL 資料庫上執行非封鎖備份。

史詩

規劃遷移

任務	Description	所需的技能
驗證資料庫版本。	驗證來源和目標資料庫的版本。如需有關 支援的 MySQL 版本的資訊 AWS DMS，請參閱 AWS DMS 文件中的 的來源 AWS DMS和 的目標 AWS DMS。	DBA
識別目標作業系統。	判斷目標作業系統的版本。如需 Amazon EC2 支援的目標作業系統清單，請參閱 Amazon EC2 FAQs。	DBA，系統管理員
識別硬體需求。	根據 MySQL 相容性清單和容量需求，判斷目標伺服器執行個體的硬體需求。	DBA，系統管理員
識別儲存需求。	判斷目標資料庫的儲存類型和容量。	DBA，系統管理員
識別網路需求。	判斷網路需求，例如延遲和頻寬。	DBA，系統管理員
選擇目標執行個體類型。	根據容量、儲存功能和網路功能選擇目標執行個體類型。	DBA，系統管理員
識別安全需求。	判斷來源和目標資料庫的網路或主機存取安全需求。	DBA，系統管理員

任務	Description	所需的技能
識別使用者。	決定 MySQL 軟體安裝的作業系統使用者清單。如需詳細資訊，請參閱 MySQL 文件。	DBA，系統管理員
決定備份策略。		DBA
判斷可用性需求。		DBA
識別應用程式遷移或切換策略。		DBA，系統管理員

設定基礎設施

任務	Description	所需的技能
建立虛擬私有雲端 (VPC) 和子網路。	設定路由表、網際網路閘道、NAT 閘道和子網路。如需詳細資訊，請參閱 Amazon VPC 文件中的 VPC 組態選項。	系統管理員
建立安全群組和網路存取控制清單 ACLs)。	根據您的需求設定連接埠 (MySQL 的預設值為 3306) 和 CIDR 範圍或特定 IPs。	系統管理員
設定和啟動 EC2 執行個體。	如需說明，請參閱 Amazon EC2 文件中的啟動 EC2 執行個體。 Amazon EC2	系統管理員

安裝 MySQL 軟體

任務	Description	所需的技能
建立使用者和群組。	建立需要存取伺服器和資料庫的作業系統使用者和群組。如	DBA，系統管理員

任務	Description	所需的技能
	需詳細資訊，請參閱 MySQL 文件中的存取控制和帳戶管理。	
下載 MySQL。	下載 MySQL 軟體。如需指示和二進位檔，請參閱 MySQL 文件中的安裝 MySQL。	DBA，系統管理員
在 EC2 執行個體上安裝 MySQL 並設定伺服器。	連接至 EC2 執行個體並安裝 MySQL 軟體。如需詳細資訊，請參閱 Amazon EC2 文件中的連線至 EC2 執行個體。Amazon EC2	DBA，系統管理員

遷移資料 – 選項 1

任務	Description	所需的技能
使用原生 MySQL 或第三方工具遷移資料。	此選項使用原生 MySQL 工具或第三方工具來遷移資料庫物件和資料。如需說明，請參閱 mysqldump 或 Percona XtraBackup 的文件（適用於實體遷移）。如需使用這些工具的詳細資訊，請參閱 MySQL 到 Amazon RDS for MySQL 或 Amazon Aurora MySQL 的遷移選項 AWS 部落格文章。	DBA

遷移資料 – 選項 2

任務	Description	所需的技能
使用 遷移資料 AWS DMS。	如需詳細資訊，請參閱 AWS DMS 文件中的 的高階檢視 AWS DMS。	DBA

準備切換

任務	Description	所需的技能
收集物件計數。	從來源資料庫和新目標資料庫收集物件計數。修正目標資料庫中的任何差異。	DBA
檢查相依性。	確認往返其他資料庫的相依性（連結）仍然有效且正常運作。	DBA
測試。	如果這是一個測試週期，請執行查詢測試、收集指標並修正任何問題。	DBA

剪下

任務	Description	所需的技能
移動用戶端。	將應用程式用戶端切換到新的基礎設施。	DBA、應用程式擁有者、系統管理員
提供支援。	在功能應用程式測試期間提供支援。	DBA

關閉專案

任務	Description	所需的技能
關閉資源。	關閉 AWS DMS 複寫執行個體和其他暫時 AWS 資源。	DBA，系統管理員
檢閱和專案文件。	檢閱並驗證專案文件。	DBA、應用程式擁有者、系統管理員
收集指標。	收集遷移時間、與工具輔助變更相比的手動變更百分比，以及節省成本等指標。	DBA、應用程式擁有者、系統管理員
關閉專案。	關閉遷移專案並提供意見回饋。	DBA、應用程式擁有者、系統管理員
停用來源資料庫。	停用內部部署 MySQL 資料庫。	DBA，系統管理員

相關資源

參考

- Amazon EC2 文件
- AWS DMS 文件
- Amazon EC2 定價
- AWS DMS Step-by-Step演練
- mysqldump
- Percona XtraBackup

教學課程和影片

- 入門 AWS DMS
- Amazon EC2 簡介 – Elastic Cloud Server & Hosting with AWS （影片）

在沒有加密的情況下監控 Amazon Aurora 是否有執行個體

Mansi Suratwala，Amazon Web Services

總結

此模式提供 Amazon Web Services (AWS) CloudFormation 範本，您可以部署該範本，以在未開啟加密的情況下建立 Amazon Aurora 執行個體時設定自動通知。

Aurora 為全受管關聯式資料庫引擎，可與 MySQL 和 PostgreSQL 相容。透過一些工作負載，Aurora 可提供 MySQL 最多五倍的輸送量和 PostgreSQL 最多三倍的輸送量，而不需變更您的多數現有應用程式。

CloudFormation 範本會建立 Amazon CloudWatch Events 事件和 AWS Lambda 函數。事件使用 AWS CloudTrail 來監控任何 Aurora 執行個體的建立或現有執行個體的時間點還原。Cloudwatch Events 事件會啟動 Lambda 函數，以檢查是否啟用加密。如果未開啟加密，Lambda 函數會傳送 Amazon Simple Notification Service (Amazon SNS) 通知，通知您違規。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶

限制

- 此服務控制僅適用於 Amazon Aurora 執行個體。它不支援其他 Amazon Relational Database Service (Amazon RDS) 執行個體。
- CloudFormation 範本必須僅針對 `CreateDBInstance`和 `RestoreDBClusterToPointInTime` 部署。

產品版本

- Amazon Aurora 支援的 PostgreSQL 版本
- Amazon Aurora 支援的 MySQL 版本

Architecture

目標技術堆疊

- Amazon Aurora
- AWS CloudTrail
- Amazon CloudWatch
- AWS Lambda
- Amazon Simple Storage Service (Amazon S3)
- Amazon SNS

目標架構

自動化和擴展

您可以針對不同的區域和帳戶多次使用 CloudFormation 範本。您只需要在每個區域或帳戶中執行一次。

工具

工具

- Amazon Aurora – Amazon Aurora 是全受管關聯式資料庫引擎，與 MySQL 和 PostgreSQL 相容。
- AWS CloudTrail – AWS CloudTrail 可協助您管理 AWS 帳戶的控管、合規以及操作和風險稽核。使用者、角色或 AWS 服務採取的動作會在 CloudTrail 中記錄為事件。
- Amazon CloudWatch Events – Amazon CloudWatch Events 提供near-real-time的系統事件串流，說明 AWS 資源的變更。
- AWS Lambda – AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種高度可擴展的物件儲存服務，可用於各種儲存解決方案，包括網站、行動應用程式、備份和資料湖。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一種受管服務，使用 Lambda、HTTP、電子郵件、行動推播通知和行動文字訊息 (SMS) 提供訊息傳遞。

Code

專案的 .zip 檔案可做為附件使用。

史詩

建立 Lambda 指令碼的 S3 儲存貯體

任務	Description	所需的技能
定義 S3 儲存貯體。	開啟 Amazon S3 主控台，然後選擇或建立 S3 儲存貯體。此 S3 儲存貯體將託管 Lambda 程式碼 .zip 檔案。您的 S3 儲存貯體必須與 Aurora 位於相同的區域。S3 儲存貯體名稱不能包含正斜線。	雲端架構師

將 Lambda 程式碼上傳至 S3 儲存貯體

任務	Description	所需的技能
上傳 Lambda 程式碼。	將附件區段中提供的 Lambda 程式碼 .zip 檔案上傳至您定義的 S3 儲存貯體。	雲端架構師

部署 CloudFormation 範本

任務	Description	所需的技能
部署 CloudFormation 範本。	在 CloudFormation 主控台上，部署做為此模式附件提供的 `RDS_Aurora_Encryption_At_Rest.yml` CloudFormation 範本。在下一個 Epic 中，提供範本參數的值。	雲端架構師

完成 CloudFormation 範本中的參數

任務	Description	所需的技能
提供 S3 儲存貯體名稱。	輸入您在第一個特徵中建立或選擇的 S3 儲存貯體名稱。	雲端架構師
提供 S3 金鑰。	提供 Lambda 程式碼 .zip 檔案在 S3 儲存貯體中的位置，不帶正斜線（例如 `<directory>/<file-name>.zip`）。	雲端架構師
提供電子郵件地址。	提供作用中的電子郵件地址以接收 Amazon SNS 通知。	雲端架構師
定義記錄層級。	定義 Lambda 函數的記錄層級和頻率。 會`Info`指定應用程式進度的詳細資訊訊息。 會`Error`指定仍可允許應用程式繼續執行的錯誤事件。 會`Warning`指定可能有害的情況。	雲端架構師

確認訂閱

任務	Description	所需的技能
確認訂閱。	當範本成功部署時，它會傳送訂閱電子郵件訊息到提供的電子郵件地址。若要接收通知，您必須確認此電子郵件訂閱。	雲端架構師

相關資源

- 建立 S3 儲存貯體

- 將檔案上傳至 S3 儲存貯體
- 建立 Amazon Aurora 資料庫叢集
- 建立使用 AWS CloudTrail 在 AWS API 呼叫上觸發的 CloudWatch Events 規則 CloudTrail

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Amazon CloudWatch 監控 Oracle GoldenGate 日誌

Chithra Krishnamurthy，Amazon Web Services

摘要

Oracle GoldenGate 可在 Amazon Relational Database Service (Amazon RDS) for Oracle 資料庫之間或在 Amazon Elastic Compute Cloud (Amazon EC2) 上託管的 Oracle 資料庫之間提供即時複寫。它支援單向和雙向複寫。

當您使用 GoldenGate 進行複寫時，監控對於驗證 GoldenGate 程序是否啟動並執行至關重要，以確保來源和目標資料庫處於同步狀態。

此模式說明針對 GoldenGate 錯誤日誌實作 Amazon CloudWatch 監控的步驟，以及如何設定警示來傳送特定事件的通知，例如 `STOP`或 ，`ABEND`以便您可以採取適當動作快速恢復複寫。

先決條件和限制

先決條件

- 在 EC2 執行個體上安裝和設定 GoldenGate，因此您可以在這些 EC2 執行個體上設定 CloudWatch 監控。如果您想要監控跨 AWS 區域的 GoldenGate 以進行雙向複寫，您必須在執行 GoldenGate 程序的每個 EC2 執行個體中安裝 CloudWatch 代理程式。

限制

- 此模式說明如何使用 CloudWatch 監控 GoldenGate 程序。CloudWatch 不會在複寫期間監控複寫延遲或資料同步問題。您必須執行個別的 SQL 查詢來監控複寫延遲或資料相關錯誤，如 GoldenGate 文件所述。

產品版本

- 本文件是以 Oracle GoldenGate 19.1.0.0.4 for Oracle on Linux x86-64 的實作為基礎。不過，此解決方案適用於所有主要版本的 GoldenGate。

Architecture

目標技術堆疊

- EC2 執行個體上安裝的 Oracle GoldenGate 二進位檔
- Amazon CloudWatch
- Amazon Simple Notification Service (Amazon SNS)

目標架構

工具

AWS 服務

- Amazon CloudWatch 是一項監控服務，用於此模式來監控 GoldenGate 錯誤日誌。
- Amazon SNS 是一項訊息通知服務，用於此模式來傳送電子郵件通知。

其他工具

- Oracle GoldenGate 是一種資料複寫工具，可用於 Amazon RDS for Oracle 資料庫或託管於 Amazon EC2 的 Oracle 資料庫。

高階實作步驟

1. 為 CloudWatch 代理程式建立 AWS Identity and Access Management (IAM) 角色。
2. 將 IAM 角色連接至產生 GoldenGate 錯誤日誌的 EC2 執行個體。
3. 在 EC2 執行個體上安裝 CloudWatch 代理程式。
4. 設定 CloudWatch 代理程式組態檔案： `awscli.conf`和 `awslogs.conf`。
5. 啟動 CloudWatch 代理程式。
6. 在日誌群組中建立指標篩選條件。
7. 設定 Amazon SNS。
8. 建立指標篩選條件的警示。當這些篩選條件擷取事件時，Amazon SNS 會傳送電子郵件提醒。

如需詳細說明，請參閱下一節。

史詩

步驟 1. 為 CloudWatch 代理程式建立 IAM 角色

任務	Description	所需的技能
建立 IAM 角色。	存取 AWS 資源需要許可，因此您可以建立 IAM 角色，以包含每個伺服器執行 CloudWatch 代理程式所需的許可。 若要建立 IAM 角色： 1. 登入 AWS 管理主控台，然後前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。 2. 在導覽窗格中，選擇角色，然後選擇建立角色。 3. 針對信任的實體類型，選擇 AWS 服務。 4. 針對常用案例，選擇 EC2，然後選擇下一步。 5. 在政策清單中，選取 CloudWatchAgentServerPolicy 旁的核取方塊。如有需要，請使用搜尋方塊來尋找政策。 6. 選擇下一步。 7. 對於 Role name (角色名稱)，輸入新角色的名稱，例如 `goldengate-cw-monitoring-role` 或另一個您喜好的名稱。 8. (選用) 針對 Role description (角色描述)，輸入描述。	AWS 一般

任務	Description	所需的技能
	9. 確認 CloudWatchAgentServerPolicy 出現在政策名稱下。 10. (選用) 新增一或多個標籤 (鍵/值對) 來組織、追蹤或控制此角色的存取，然後選擇建立角色。	

步驟 2. 將 IAM 角色連接至 GoldenGate EC2 執行個體

任務	Description	所需的技能
將 IAM 角色連接至產生 GoldenGate 錯誤日誌的 EC2 執行個體。	GoldenGate 產生的錯誤日誌必須填入 CloudWatch 並進行監控，因此您需要將您在步驟 1 中建立的 IAM 角色連接到 GoldenGate 執行所在的 EC2 執行個體。 若要將 IAM 角色連接至執行個體： 1. 前往 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。 2. 在導覽窗格中，選擇執行個體，然後尋找執行 GoldenGate 的執行個體。 3. 選取執行個體，然後選擇動作、安全性、修改 IAM 角色。 4. 選取要連接到執行個體的第一個步驟中建立的 IAM 角色，然後選擇儲存。	AWS 一般

步驟 3-5。在 Goldengate EC2 執行個體上安裝和設定 CloudWatch 代理程式

任務	Description	所需的技能
在 GoldenGate EC2 執行個體上安裝 CloudWatch 代理程式。	若要安裝代理程式，請執行 命令： `sudo yum install -y awslogs`	AWS 一般

編輯代理程式組態檔案。	1. 執行下列命令。
	`sudo su -`
	2. 編輯此檔案以視需要更新 AWS 區域。
	`cat /etc/awslogs/conf [plugins] cwlogs = cwlogs [default] region = us-east-1`
	3. 編輯 `/etc/awslogs/awslogs.conf` 檔案以更新檔案名稱、日誌群組名稱和日期/時間格式。您必須指定日期/時間以符合 中的日期格式`ggserror.log` ；否則，日誌串流不會流入 CloudWatch。例如：
	`datetime_format = %Y-%m-%dT%H:%M:%S%z file = /u03/oracle/oragg/ggserr.log log_group_name = goldengate_monitor`
		AWS 一般

任務	Description	所需的技能
啟動 CloudWatch 代理程式。	若要啟動代理程式，請使用下列命令。 `$ sudo service awslogsd start` 啟動代理程式後，您可以在 CloudWatch 主控台中檢視日誌群組。日誌串流將具有 檔案的內容。	AWS 一般

步驟 6. 建立日誌群組的指標篩選條件

任務	Description	所需的技能
建立關鍵字 ABEND 和 STOPPED 的指標篩選條件。	當您為日誌群組建立指標篩選條件時，只要在錯誤日誌中識別篩選條件，就會啟動警示，並根據 Amazon SNS 組態傳送電子郵件通知。 若要建立指標篩選條件： 1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。 2. 選擇日誌群組的名稱。 3. 選擇 Actions (動作)，然後選擇 Create metric filter (建立指標篩選條件)。 4. 針對篩選條件模式，指定模式，例如 ABEND。	CloudWatch

任務	Description	所需的技能
	5. 選擇 Next (下一步)，然後輸入指標篩選條件的名稱。 6. 在 Metric details (指標詳細資訊) 下的 Metric namespace (指標命名空間) 中，輸入將發佈指標的 CloudWatch 命名空間名稱。如果命名空間不存在，請務必選取 Create new (新建)。 7. 對於指標值，請輸入 1，因為您的指標篩選條件正在計算篩選條件中關鍵字的出現次數。 8. 將單位設定為無。 9. 選擇 Create metric filter (建立指標篩選條件)。可以從導覽窗格中找到您建立的指標篩選條件。 10為STOPPED模式建立另一個指標篩選條件。在一個日誌群組中，您可以建立多個指標篩選條件，並個別設定警示。	

步驟 7. 設定 Amazon SNS

任務	Description	所需的技能
建立 SNS 主題。	在此步驟中，您會設定 Amazon SNS 為指標篩選條件建立警示。 若要建立 SNS 主題：	Amazon SNS

任務	Description	所需的技能
	1. 登入 Amazon SNS 主控台，網址為 https：//https://console.aws.amazon.com/sns/home。 2. 在建立主題方塊中，輸入主題名稱，例如 `goldengate-alert` ，然後選擇下一步。 3. 針對類型，選擇標準。 4. 捲動到表單結尾，然後選擇 Create topic (建立主題)。主控台會開啟新主題的 Details (詳細資料) 頁面。	

任務	Description	所需的技能
建立訂閱。	若要建立 主題的訂閱： 1. 在左導覽窗格中，選擇訂閱。 2. 在訂閱頁面，選擇建立訂閱。 3. 在建立訂閱頁面上，選擇主題 ARN 欄位，以查看您 AWS 帳戶中主題的清單。 4. 選擇您在之前步驟所建立的主題。 5. 對於 Protocol (通訊協定)，選擇 Email (電子郵件)。 6. 針對 Endpoint (端點)，請輸入可用於接收通知的電子郵件地址。 7. 選擇建立訂閱。 主控台會開啟新訂閱的詳細資訊頁面。 8. 檢查您的電子郵件收件匣是否有來自 AWS Notifications 的訊息，然後在電子郵件中選擇確認訂閱。 Amazon SNS 會開啟您的 web 瀏覽器，並顯示含有您的訂閱 ID 的訂閱確認。	Amazon SNS

步驟 8. 建立警示以傳送指標篩選條件的通知

任務	Description	所需的技能
建立 SNS 主題的警示。	若要根據日誌群組指標篩選條件建立警示： 1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。 2. 在導覽窗格中，選擇 Logs (日誌)，然後選擇 Log groups (日誌群組)。 3. 選擇包含指標篩選條件的日誌群組。 4. 選擇 Metric filters (指標篩選條件)。 5. 在指標篩選條件索引標籤中，選取您要作為警示基礎之指標篩選條件的核取方塊。 6. 選擇 Create alarm (建立警示)。 7. 針對條件，在每個區段中指定下列項目： • 對於閾值類型，選擇靜態。 • 對於每當 <metric-name> 為 . . 時，請選擇較大。 • 對於超過 . . . ，請指定 0。 8. 選擇下一步。 9. 在通知下：	CloudWatch

任務	Description	所需的技能
	• 針對 Alarm state trigger (警示狀態觸發)，選擇 In Alarm (警示中)。 • 針對將通知傳送至下列 SNS 主題，選擇選取現有主題。 • 在電子郵件方塊中，選取您在上一個步驟中建立的 Amazon SNS 主題。 10選擇下一步。 11.在 Name and description (名稱和描述) 中，輸入警示的名稱和描述。 **Note** 對於描述，您可以指定執行個體名稱，以便通知電子郵件具有描述性。 12對於預覽和建立，請檢查您的組態是否正確，然後選擇建立警示。 完成這些步驟後，只要在您監控的 GoldenGate 錯誤日誌檔 (`ggserr.log`) 中偵測到這些模式，您就會收到電子郵件通知。	

疑難排解

問題	解決方案
GoldenGate 錯誤日誌中的日誌串流不會流入 CloudWatch。	檢查 `/etc/awslogs/awslogs.conf` 檔案以驗證檔案名稱、日誌群組名稱和日期/時間格式。您必須指定日期/時間，以符合 中的日期格式`ggserror.log` 。否則，日誌串流不會流入 CloudWatch。

相關資源

- Amazon CloudWatch 文件
- 使用 CloudWatch 代理程式收集指標和日誌
- Amazon SNS 文件

在 Amazon RDS for Oracle 上將 Oracle Database Enterprise Edition 轉換為 Standard Edition 2

Lanre (Lan-Ray) showunmi 和 Tarun Chawla，Amazon Web Services

總結

Oracle Database Enterprise Edition (EE) 是許多企業中執行應用程式的熱門選擇。不過，在某些情況下，應用程式使用很少或沒有 Oracle 資料庫 EE 功能，因此缺乏產生大量授權成本的正當理由。當您遷移至 Amazon RDS 時，您可以將此類資料庫降級為 Oracle Database Standard Edition 2 (SE2)，以節省成本。

此模式說明如何在從現場部署遷移至 Amazon RDS for Oracle 時，從 Oracle 資料庫 EE 降級至 Oracle 資料庫 SE2。如果您的 EE Oracle 資料庫已在 Amazon RDS 或 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上執行，則此模式中顯示的步驟也適用。

如需詳細資訊，請參閱 AWS 規範指引指南，了解如何評估 Oracle 資料庫降級至 AWS 上的 Standard Edition 2。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Oracle 資料庫企業版
- 用戶端工具，例如 Oracle SQL Developer 或 SQL*Plus，用於在 Oracle 資料庫上連接和執行 SQL 命令
- 執行評估的資料庫使用者；例如，下列其中一項：
  - 具有足夠權限執行 AWS Schema Conversion Tool (AWS SCT) 評估的使用者
  - 具有足夠權限可在 Oracle 資料庫字典資料表上執行 SQL 查詢的使用者
- 執行資料庫遷移的資料庫使用者；例如，下列其中一項：
  - 具有足夠權限執行 AWS Database Migration Service (AWS DMS) 的使用者
  - 具有足夠權限可執行 Oracle Data Pump 匯出和匯入的使用者
  - 具有足夠權限可執行 Oracle GoldenGate 的使用者

限制

- Amazon RDS for Oracle 具有資料庫大小上限。如需詳細資訊，請參閱 Amazon RDS 資料庫執行個體儲存體。

產品版本

本文件所述的一般邏輯適用於 9i 及更新版本的 Oracle。如需自我管理和 Amazon RDS for Oracle 資料庫的支援版本，請參閱 AWS DMS 文件。

若要在不支援 AWS SCT 的情況下識別功能用量，請在來源資料庫上執行 SQL 查詢。若要從不支援 AWS DMS 和 Oracle Data Pump 的舊版 Oracle 遷移，請使用 Oracle Export and Import 公用程式。

如需支援版本的最新清單，請參閱 AWS 文件中的 Oracle on Amazon RDS。如需定價和支援執行個體類別的詳細資訊，請參 Amazon RDS for Oracle 定價。

Architecture

來源技術堆疊

- 在內部部署或 Amazon EC2 上執行的 Oracle Database Enterprise Edition

使用原生 Oracle 工具鎖定技術堆疊

- 執行 Oracle Database SE2 的 Amazon RDS for Oracle

1. 使用 Oracle Data Pump 匯出資料。
2. 透過資料庫連結將傾印檔案複製到 Amazon RDS。
3. 使用 Oracle Data Pump 將傾印檔案匯入 Amazon RDS。

使用 AWS DMS 鎖定技術堆疊

- 執行 Oracle Database SE2 的 Amazon RDS for Oracle
- AWS DMS

1. 搭配 FLASHBACK_SCN 使用 Oracle Data Pump 匯出資料。
2. 透過資料庫連結將傾印檔案複製到 Amazon RDS。
3. 使用 Oracle Data Pump 將傾印檔案匯入 Amazon RDS。
4. 使用 AWS DMS 變更資料擷取 (CDC)。

工具

AWS 服務

- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。此模式使用 Amazon RDS for Oracle。
- AWS SCT 提供專案型使用者介面，可自動評估、轉換來源 Oracle 資料庫的資料庫結構描述，並將其複製到與 Amazon RDS for Oracle 相容的格式。AWS SCT 可讓您分析可透過將授權類型從 Enterprise 變更為 Standard Edition of Oracle 來實現的潛在成本節省。AWS SCT 報告的授權評估和雲端支援區段提供使用中 Oracle 功能的詳細資訊，讓您可以在遷移至 Amazon RDS for Oracle 時做出明智的決定。

其他工具

- 原生 Oracle 匯入和匯出公用程式支援將 Oracle 資料移入和移出 Oracle 資料庫。Oracle 提供兩種類型的資料庫匯入和匯出公用程式：原始匯出和匯入（適用於舊版）和 Oracle Data Pump 匯出和匯入（適用於 Oracle Database 10g 版本 1 和更新版本）。
- Oracle GoldenGate 提供即時複寫功能，讓您可以在初始載入後同步目標資料庫。此選項有助於減少應用程式上線期間的停機時間。

史詩

進行遷移前評估

任務	Description	所需的技能
驗證應用程式的資料庫需求。	確保您的應用程式已通過認證，可在 Oracle Database	應用程式開發人員、DBA、應用程式擁有者

任務	Description	所需的技能
	SE2 上執行。直接與軟體廠商、開發人員或應用程式文件確認。	
直接在資料庫中調查 EE 功能的使用。	若要判斷 EE 功能使用情況，請執行下列其中一項操作： • 為您的 Oracle EE 資料庫產生 AWS SCT 評估報告。如果您想要變更授權類型，報告會告訴您應從目前 EE 資料庫移除哪些功能。 • 如果您有 Oracle Support 帳戶，請取得並執行支援文件 1317265.1 `options_packs_usage_statistics.sql` 中的指令碼，以產生 Oracle 資料庫正在使用的選項和功能報告。 https://support.oracle.com/knowledge/Oracle%20Database%20Products/1317265_1.html • 查詢 DBA_FEATURE_USAGE_STATISTICS 以顯示所有使用中功能的詳細資訊。	應用程式擁有者、DBA、應用程式開發人員

任務	Description	所需的技能
識別操作活動使用 EE 功能。	資料庫或應用程式管理員有時會依賴僅限 EE 的功能來進行操作活動。常見範例包括線上維護活動（索引重建、資料表移動），以及依批次任務使用平行處理。 可以盡可能修改您的操作來減輕這些相依性。識別這些功能的使用方式，並根據成本與優點做出決策。 使用比較 Oracle 資料庫 EE 和 SE2 功能資料表做為指南，以識別 Oracle Database SE2 中可用的功能。	應用程式開發人員、DBA、應用程式擁有者
檢閱 EE Oracle 資料庫的工作負載模式。	Oracle Database SE2 會隨時自動將用量限制為最多 16 個 CPU 執行緒。 如果您的 Oracle EE 資料庫已獲授權使用 Oracle Diagnostic Pack，請使用自動工作負載儲存庫 (AWR) 工具或 DBA_HIST_* 檢視來分析資料庫工作負載模式，以判斷當您降級至 SE2 時，16 個 CPU 執行緒的上限是否會對服務層級造成負面影響。 確保您的評估涵蓋尖峰活動期間，例如打烊、月或年處理。	應用程式擁有者、DBA、應用程式開發人員

在 AWS 上準備目標基礎設施

任務	Description	所需的技能
部署和設定聯網基礎設施。	建立虛擬私有雲端 (VPC) 和子網路、安全群組和網路存取控制清單。	AWS 管理員、雲端架構師、網路管理員、DevOps 工程師
佈建 Amazon RDS for Oracle SE2 資料庫。	佈建目標 Amazon RDS for Oracle SE2 資料庫，以符合應用程式的效能、可用性和安全性需求。我們建議生產工作負載使用異地同步備份組態。不過，為了改善遷移效能，您可以延遲啟用異地同步備份，直到資料遷移之後。	雲端管理員、雲端架構師、DBA、DevOps 工程師、AWS 管理員
自訂 Amazon RDS 環境。	設定自訂參數和選項，並啟用其他監控。如需詳細資訊，請參閱遷移至 Amazon RDS for Oracle 的最佳實務。	AWS 管理員、AWS 系統管理員、雲端管理員、DBA、雲端架構師

執行遷移試轉和應用程式測試

任務	Description	所需的技能
遷移資料（試執行）。	使用最適合您特定環境的方法，將資料從來源 Oracle EE 資料庫遷移至 Amazon RDS for Oracle SE2 資料庫執行個體。根據大小、複雜性和可用停機時間時段等因素，選取遷移策略。使用下列其中一項或組合： • 原生 Oracle 工具，例如 Oracle Data Pump（建	DBA

任務	Description	所需的技能
	議）、Oracle Import-Export 公用程式和 Oracle GoldenGate。 • AWS DMS，透過 CDC 使用完整負載搭配連續複寫。	
驗證目標資料庫。	執行資料庫儲存體和程式碼物件的遷移後驗證。檢閱遷移日誌，並修正任何已識別的問題。如需詳細資訊，請參閱將 Oracle 資料庫遷移至 AWS 雲端指南。	DBA
測試應用程式。	應用程式和資料庫管理員應視需要執行功能、效能和操作測試。如需詳細資訊，請參閱遷移至 Amazon RDS for Oracle 的最佳實務。 最後，從利益相關者取得測試結果的簽署。	應用程式開發人員、應用程式擁有者、DBA、遷移工程師、遷移負責人

切換

任務	Description	所需的技能
從 Oracle 資料庫 EE 重新整理資料。	根據應用程式可用性需求選取資料重新整理方法。如需詳細資訊，請參閱將 Oracle 資料庫遷移至 AWS 的策略中的遷移方法。 例如，您可以使用 Oracle GoldenGate 或 AWS DMS 等工具搭配持續複寫，達到幾近	應用程式擁有者、Cutover Lead、DBA、遷移工程師、遷移負責人

任務	Description	所需的技能
	零的停機時間。如果停機時間時段允許，您可以使用 Oracle Data Pump 或原始匯出匯入公用程式等離線方法執行最終資料切換。	
將應用程式指向目標資料庫執行個體。	更新應用程式和其他用戶端中的連線參數，以指向 Amazon RDS for Oracle SE2 資料庫。	應用程式開發人員、應用程式擁有者、遷移工程師、遷移負責人、切換負責人
執行遷移後活動。	執行資料遷移後任務，例如啟用異地同步備份、資料驗證和其他檢查。	DBA，遷移工程師
執行切換後監控。	使用 Amazon CloudWatch 和 Amazon RDS Performance Insights 等工具來監控 Amazon RDS for Oracle SE2 資料庫。	應用程式開發人員、應用程式擁有者、AWS 管理員、DBA、遷移工程師

相關資源

AWS 方案指引

- 將 Oracle 資料庫遷移至 AWS 雲端（指南）
- 評估在 AWS 上將 Oracle 資料庫降級至 Standard Edition 2（指南）
- 將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle（模式）
- 使用 Oracle Data Pump 將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle (模式)

部落格文章

- 使用 AWS DMS 以接近零的停機時間遷移 Oracle 資料庫
- 使用 Amazon RDS for Oracle 在 Oracle SE 中分析效能管理
- 使用 Amazon RDS for Oracle 在 Oracle SE 中管理您的 SQL 計劃
- 在 Oracle Standard Edition 中實作資料表分割：第 1 部分

使用 Precisely Connect 將大型主機資料庫複寫至 AWS

Lucio Pereira、Sayantan Giri 和 Balaji Mohan，Amazon Web Services

總結

此模式概述使用 Precisely Connect，以近乎即時的方式將資料從大型主機資料庫複寫至 Amazon 資料存放區的步驟。它使用 Amazon Managed Streaming for Apache Kafka (Amazon MSK) 和雲端中的自訂資料庫連接器實作事件型架構，以改善可擴展性、彈性和效能。

Precisely Connect 是一種複寫工具，可從舊版大型主機系統擷取資料並將其整合到雲端環境中。使用具有低延遲和高輸送量異質資料管道的近乎即時訊息流程，透過變更資料擷取 (CDC)，將資料從大型主機複寫到 AWS。

此模式也涵蓋具有多區域資料複寫和容錯移轉路由的彈性資料管道的災難復原策略。

先決條件和限制

先決條件

- 您要複寫至 AWS 雲端的現有大型主機資料庫，例如 IBM DB2、IBM 資訊管理系統 (IMS) 或虛擬儲存存取方法 (VSAM)
- 作用中的 AWS 帳戶
- 從公司環境到 AWS 的 AWS Direct Connect 或 AWS Virtual Private Network (AWS VPN)
- 具有可由舊版平台連線之子網路的虛擬私有雲端

Architecture

來源技術堆疊

大型主機環境，至少包含下列其中一個資料庫：

- IBM IMS 資料庫
- IBM DB2 資料庫
- VSAM 檔案

目標技術堆疊

- Amazon MSK

- Amazon Elastic Kubernetes Service (Amazon EKS) 和 Amazon EKS Anywhere
- Docker
- AWS 關聯式或 NoSQL 資料庫，如下所示：
  - Amazon DynamoDB
  - Oracle、Amazon RDS for PostgreSQL 或 Amazon Aurora 的 Amazon Relational Database Service (Amazon RDS)
  - Amazon ElastiCache for Redis
  - Amazon Keyspaces (適用於 Apache Cassandra)

目標架構

將大型主機資料複寫至 AWS 資料庫

下圖說明將大型主機資料複寫至 DynamoDB、Amazon RDS、Amazon ElastiCache 或 Amazon Keyspaces 等 AWS 資料庫。在內部部署大型主機環境中使用 Precisely Capture 和 Publisher、在內部部署分散式環境中使用 Precisely Dispatcher on Amazon EKS Anywhere，以及在 AWS 雲端中精確套用引擎和資料庫連接器，即可近乎即時地進行複寫。

該圖顯示以下工作流程：

1. 精確擷取會從 CDC 日誌取得大型主機資料，並在內部暫時性儲存體中維護資料。
2. Precisely Publisher 會監聽內部資料儲存體中的變更，並透過 TCP/IP 連線將 CDC 記錄傳送至 Precisely Dispatcher。
3. Precisely Dispatcher 會從發佈者接收 CDC 記錄，並將其傳送至 Amazon MSK。Dispatcher 會根據使用者組態和多個工作者任務建立 Kafka 金鑰，以平行推送資料。當記錄存放在 Amazon MSK 中時，發送者會將確認回發佈者。
4. Amazon MSK 會將 CDC 記錄保留在雲端環境中。主題的分割區大小取決於您的交易處理系統 (TPS) 對輸送量的需求。Kafka 金鑰是進一步轉換和交易排序的必要項目。
5. 精確套用引擎會接聽來自 Amazon MSK 的 CDC 記錄，並根據目標資料庫需求轉換資料 （例如，透過篩選或映射）。您可以將自訂邏輯新增至 Precisely SQD 指令碼。(SQD 是 Precisely 的專屬語言。) 精確套用引擎會將每個 CDC 記錄轉換為 Apache Avro 或 JSON 格式，並根據您的需求將其分發至不同的主題。
6. 目標 Kafka 主題會根據目標資料庫在多個主題中保留 CDC 記錄，而 Kafka 會根據定義的 Kafka 金鑰促進交易排序。分割區索引鍵與對應的分割區對齊，以支援循序程序。

7. 資料庫連接器（自訂 Java 應用程式）從 Amazon MSK 接聽 CDC 記錄，並將其存放在目標資料庫中。
8. 您可以根據您的需求選取目標資料庫。此模式同時支援 NoSQL 和關聯式資料庫。

災難復原

業務持續性是組織成功的關鍵。AWS 雲端提供高可用性 (HA) 和災難復原 (DR) 的功能，並支援組織的容錯移轉和備用計畫。此模式遵循主動/被動 DR 策略，並提供實作符合您 RTO 和 RPO 需求的 DR 策略的高階指引。

下圖說明 DR 工作流程。

上圖顯示以下項目：

1. 如果 AWS 區域 1 發生任何故障，則需要半自動容錯移轉。如果區域 1 發生故障，系統必須啟動路由變更，才能將 Precisely Dispatcher 連線至區域 2。
2. Amazon MSK 透過區域之間的鏡像複寫資料。因此，在容錯移轉期間，區域 2 中的 Amazon MSK 叢集必須提升為主要領導者。
3. 精確套用引擎和資料庫連接器是可在任何區域中運作的無狀態應用程式。
4. 資料庫同步處理取決於目標資料庫。例如，DynamoDB 可以使用全域資料表，而 ElastiCache 可以使用全域資料存放區。

透過資料庫連接器進行低延遲和高輸送量處理

資料庫連接器是此模式中的關鍵元件。連接器遵循以接聽程式為基礎的方法來從 Amazon MSK 收集資料，並透過關鍵任務應用程式（第 0 層和第 1 層）的高輸送量和低延遲處理將交易傳送至資料庫。下圖說明此程序。

此模式支援透過多執行緒處理引擎開發具有單一執行緒耗用量的自訂應用程式。

1. 連接器主執行緒會使用來自 Amazon MSK 的 CDC 記錄，並將其傳送至執行緒集區進行處理。
2. 執行緒集區中的執行緒會處理 CDC 記錄，並將其傳送至目標資料庫。
3. 如果所有執行緒都忙碌，執行緒佇列會保留 CDC 記錄。
4. 主要執行緒會等待從執行緒佇列清除所有記錄，並將偏移遞交至 Amazon MSK。

5. 子執行緒處理失敗。如果在處理期間發生失敗，則失敗的訊息會傳送至 DLQ （無效字母佇列） 主題。
6. 子執行緒會根據大型主機時間戳記啟動條件式更新 （請參閱 DynamoDB 文件中的條件表達式），以避免資料庫中有任何重複或out-of-order的更新。

如需有關如何使用多執行緒功能實作 Kafka 消費者應用程式的資訊，請參閱 Confluent 網站上的部落格文章 Apache Kafka 消費者的多執行緒訊息使用。

工具

AWS 服務

- Amazon Managed Streaming for Apache Kafka (Amazon MSK) 是一種全受管服務，可協助您建置和執行使用 Apache Kafka 處理串流資料的應用程式。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 AWS 上執行 Kubernetes，而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- Amazon EKS Anywhere 可協助您部署、使用和管理在您自己的資料中心執行的 Kubernetes 叢集。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。
- Amazon ElastiCache 可協助您在 AWS 雲端中設定、管理和擴展分散式記憶體內快取環境。
- Amazon Keyspaces （適用於 Apache Cassandra) 是一種受管資料庫服務，可協助您在 AWS 雲端中遷移、執行和擴展 Cassandra 工作負載。

其他工具

- Precisely Connect 將 VSAM 資料集或 IBM 大型主機資料庫等舊版大型主機系統的資料整合到新一代雲端和資料平台。

最佳實務

- 尋找 Kafka 分割區和多執行緒連接器的最佳組合，以平衡最佳效能和成本。由於較高的 MIPS （每秒百萬個指令） 使用量，多個精確擷取和發送器執行個體可能會增加成本。
- 避免將資料處理和轉換邏輯新增至資料庫連接器。為此，請使用精確套用引擎，以微秒為單位提供處理時間。

- 在資料庫連接器中建立資料庫 (心跳） 的定期請求或運作狀態檢查呼叫，以經常暖機連線並降低延遲。
- 實作執行緒集區驗證邏輯，以了解執行緒佇列中的待定任務，並等待所有執行緒完成，再進行下一次 Kafka 輪詢。這有助於避免節點、容器或程序當機時遺失資料。
- 透過運作狀態端點公開延遲指標，透過儀表板和追蹤機制增強可觀測性功能。

史詩

準備來源環境 （內部部署）

任務	Description	所需的技能
設定大型主機程序 （批次或線上公用程式），從大型主機資料庫啟動 CDC 程序。	1. 識別大型主機環境。 2. 識別將參與 CDC 程序的大型主機資料庫。 3. 在大型主機環境中，開發啟動 CDC 工具的程序，以擷取來源資料庫中的變更。如需說明，請參閱您的大型主機文件。 4. 記錄 CDC 程序，包括組態。 5. 在測試和生產環境中部署程序。	大型主機工程師
啟用大型主機資料庫日誌串流。	1. 在大型主機環境中設定日誌串流以擷取 CDC 日誌。如需說明，請參閱您的大型主機文件。 2. 測試日誌串流，以確保其擷取必要的資料。 3. 在測試和生產環境中部署日誌串流。	大型主機資料庫專家

任務	Description	所需的技能
使用擷取元件來擷取 CDC 記錄。	1. 在大型主機環境中安裝和設定精確擷取元件。如需說明，請參閱精確文件。 2. 測試組態，以確保擷取元件正常運作。 3. 設定複寫程序，透過擷取元件複寫擷取的 CDC 記錄。 4. 記錄每個來源資料庫的擷取組態。 5. 開發監控系統，以確保擷取元件隨著時間正確收集日誌。 6. 在測試和生產環境中部署安裝和組態。	大型主機工程師，精確連線 SME
設定發佈者元件以接聽擷取元件。	1. 在大型主機環境中安裝和設定 Precisely Publisher 元件。如需說明，請參閱精確文件。 2. 測試組態，以確保發佈者元件正常運作。 3. 設定複寫程序，將 CDC 記錄從發佈者發佈至 Precisely Dispatcher 元件。 4. 記錄發佈者組態。 5. 開發監控系統，以確保發布者元件隨著時間正常運作。 6. 在測試和生產環境中部署安裝和組態。	大型主機工程師，精確連線 SME

任務	Description	所需的技能
在內部部署分散式環境中佈建 Amazon EKS Anywhere。	1. 在內部部署基礎設施上安裝 Amazon EKS Anywhere，並確保已正確設定。如需說明，請參閱 Amazon EKS Anywhere 文件。 2. 為 Kubernetes 叢集設定安全的網路環境，包括防火牆。 3. 實作並測試 Amazon EKS Anywhere 叢集的範例應用程式部署。 4. 實作叢集的自動擴展功能。 5. 開發並實作備份和災難復原程序。	DevOps 工程師
在分散式環境中部署和設定 Dispatcher 元件，以在 AWS 雲端中發佈主題。	1. 設定和容器化 Precisely Dispatcher 元件。如需說明，請參閱精確文件。 2. 將 Dispatcher Docker 映像部署至內部部署 Amazon EKS Anywhere 環境。 3. 設定 AWS 雲端和 Dispatcher 之間的安全連線。 4. 開發監控系統，以確保 Dispatcher 元件隨著時間正常運作。 5. 在測試和生產環境中部署安裝和組態。	DevOps 工程師，精確連線中小企業

準備目標環境 (AWS)

任務	Description	所需的技能
在指定的 AWS 區域中佈建 Amazon EKS 叢集。	1. 登入您的 AWS 帳戶並進行設定，以確保具備建立和管理 Amazon EKS 叢集所需的許可。 2. 在選取的 AWS 區域中建立虛擬私有雲端 (VPC) 和子網路。如需說明，請參閱 Amazon EKS 文件。 3. 建立並設定必要的網路安全群組，以允許 Amazon EKS 叢集與 VPC 中其他資源之間的通訊。如需詳細資訊，請參閱 Amazon EKS 文件。 4. 建立 Amazon EKS 叢集，並使用正確的節點群組大小和執行個體類型進行設定。 5. 部署範例應用程式來驗證 Amazon EKS 叢集。	DevOps 工程師、網路管理員
佈建 MSK 叢集並設定適用的 Kafka 主題。	1. 設定您的 AWS 帳戶，以確保具備建立和管理 MSK 叢集所需的許可。 2. 建立並設定必要的網路安全群組，以允許 MSK 叢集與 VPC 中其他資源之間的通訊。如需詳細資訊，請參閱 Amazon VPC 文件。 3. 建立 MSK 叢集並將其設定為包含應用程式將使用的 Kafka 主題。如需詳細資訊	DevOps 工程師、網路管理員

任務	Description	所需的技能
	，請參閱 Amazon MSK 文件。	
設定 Apply Engine 元件以聆聽複寫的 Kafka 主題。	1. 設定和容器化精確套用引擎元件。 2. 將 Apply Engine Docker 映像部署到 AWS 帳戶中的 Amazon EKS 叢集。 3. 設定套用引擎以聆聽 MSK 主題。 4. 在套用引擎中開發和設定 SQD 指令碼，以處理篩選和轉換。如需詳細資訊，請參閱精確文件。 5. 在測試和生產環境中部署 Apply Engine。	精確連線 SME

任務	Description	所需的技能
在 AWS 雲端中佈建資料庫執行個體。	1. 設定您的 AWS 帳戶，以確保擁有建立和管理資料庫叢集和資料表所需的許可。如需說明，請參閱您要使用的 AWS 資料庫服務的 AWS 文件。（如需連結，請參閱資源一節。） 2. 在選取的 AWS 區域中建立 VPC 和子網路。 3. 建立並設定必要的網路安全群組，以允許資料庫執行個體與 VPC 中其他資源之間的通訊。 4. 建立資料庫並將其設定為包含應用程式將使用的資料表。 5. 設計和驗證資料庫結構描述。	資料工程師、DevOps 工程師
設定和部署資料庫連接器，以聆聽套用引擎發佈的主題。	1. 設計資料庫連接器，將 Kafka 主題連接至您在先前步驟中建立的 AWS 資料庫。 2. 根據目標資料庫開發連接器。 3. 設定連接器以聆聽套用引擎發佈的 Kafka 主題。 4. 將連接器部署到 Amazon EKS 叢集。	應用程式開發人員、雲端架構師、資料工程師

設定業務持續性和災難復原

任務	Description	所需的技能
為您的業務應用程式定義災難復原目標。	1. 根據您的業務需求和影響分析，定義 CDC 管道的 RPO 和 RTO 目標。 2. 定義通訊和通知程序，以確保所有利益相關者都知道災難復原計畫。 3. 決定實作災難復原計劃所需的預算和資源。 4. 記錄災難復原目標，包括 RPO 和 RTO 目標。	雲端架構師、資料工程師、應用程式擁有者
根據定義的 RTO/RPO 設計災難復原策略。	1. 根據您的關鍵性和復原需求，決定最適合 CDC 管道的災難復原策略。 2. 定義災難復原架構和拓撲。 3. 定義 CDC 管道的容錯移轉和容錯回復程序，以確保它們可以快速無縫地切換到備份區域。 4. 記錄災難復原策略和程序，並確保所有利益相關者都清楚地了解設計。	雲端架構師、資料工程師
佈建災難復原叢集和組態。	1. 佈建次要 AWS 區域以進行災難復原。 2. 在次要 AWS 區域中，建立與主要 AWS 區域相同的環境。 3. 在主要和次要區域之間設定 Apache Kafka MirrorMaker。如需詳細資訊，請參閱 Amazon MSK 文件。	DevOps 工程師、網路管理員、雲端架構師

任務	Description	所需的技能
	4. 在次要區域中設定待命應用程式。 5. 設定主要和次要區域之間的資料庫複寫。	
測試 CDC 管道的災難復原。	1. 定義 CDC 管道災難復原測試的範圍和目標，包括要實現的測試案例和 RTO。 2. 識別用於執行災難復原測試的測試環境和基礎設施。 3. 準備測試資料集和指令碼來模擬失敗案例。 4. 驗證資料完整性和一致性，以確保沒有資料遺失。	應用程式擁有者、資料工程師、雲端架構師

相關資源

AWS 資源

- Amazon DynamoDB
- Amazon DynamoDB 的條件表達式
- Amazon EKS
- Amazon EKS Anywhere
- Amazon ElasticCache
- Amazon Keyspaces
- Amazon MSK
- Amazon RDS 和 Amazon Aurora
- Amazon VPC

精確連線資源

- **精確連線概觀**

- 使用 Precisely Connect 變更資料擷取

Confluent 資源

- Apache Kafka 消費者的多執行緒訊息使用量

使用 Lambda 和 Secrets Manager 來排程 Amazon RDS for PostgreSQL 和 Aurora PostgreSQL 的任務

Yaser Raja，Amazon Web Services

總結

對於內部部署資料庫和託管在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的資料庫，資料庫管理員通常會使用 cron 公用程式來排程任務。

例如，可以使用 Cron 輕鬆排程資料擷取任務或資料清除任務。對於這些任務，資料庫登入資料通常是硬式編碼或存放在屬性檔案中。不過，當您遷移至 Amazon Relational Database Service (Amazon RDS) 或 Amazon Aurora PostgreSQL 相容版本時，將無法登入主機執行個體來排程 Cron 任務。

此模式說明如何使用 AWS Lambda 和 AWS Secrets Manager 在遷移後排程 Amazon RDS for PostgreSQL 和 Aurora PostgreSQL 相容資料庫的任務。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 相容資料庫

限制

- 任務必須在 15 分鐘內完成，也就是 Lambda 函數逾時限制。如需其他限制，請參閱 AWS Lambda 文件。
- 任務程式碼必須以 Lambda 支援的語言撰寫。

Architecture

來源技術堆疊

此堆疊以 Bash、Python 和 Java 等語言撰寫的任務為特色。資料庫登入資料會存放在 屬性檔案中，並使用 Linux cron 排程任務。

目標技術堆疊

此堆疊具有 Lambda 函數，使用存放在 Secrets Manager 中的登入資料來連線至資料庫並執行活動。Lambda 函數是使用 Amazon CloudWatch Events 在排程間隔啟動。

目標架構

工具

- Amazon CloudWatch Events 提供近乎即時的系統事件串流，描述 AWS 資源的變更。使用您可以快速設定的簡單規則，您可以比對事件並將它們路由到一或多個目標函數或串流。CloudWatch Events 在操作變更時會查覺到。它會回應這些操作變更，並視需要採取修正動作，透過傳送訊息來回應環境、啟用 函數、進行變更，以及擷取狀態資訊。您也可以使用 CloudWatch Events 來排程使用 Cron 或 Rate 表達式在特定時間自行啟動的自動化動作。
- AWS Lambda 是一種運算服務，讓您無需設定或管理伺服器即可運行程式碼。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。您只需為使用的運算時間付費；程式碼未執行時無需付費。使用 Lambda，您可以為幾乎任何類型的應用程式或後端服務執行程式碼，無需管理。Lambda 在高可用性運算基礎設施上執行程式碼，並管理所有運算資源，包括伺服器和作業系統維護、容量佈建和自動擴展、程式碼監控和記錄。您只需使用 Lambda 支援的語言之一提供程式碼即可。
- AWS Secrets Manager 可協助您保護存取應用程式、服務和 IT 資源的秘密。您可以在資料庫憑證、API 金鑰和其他秘密的整個生命週期中輕鬆輪換、管理和擷取。使用者和應用程式透過呼叫 Secrets Manager APIs 來擷取秘密，無需以純文字硬式編碼敏感資訊。Secrets Manager 提供秘密輪換與 Amazon RDS、Amazon Redshift 和 Amazon DocumentDB 的內建整合。此服務可延伸至其他類型的秘密，包括 API 金鑰和 OAuth 權杖。Secrets Manager 可讓您使用精細的許可控制對秘密的存取，並針對 AWS 雲端、第三方服務和內部部署中的資源集中稽核秘密輪換。

史詩

在 Secrets Manager 中存放資料庫登入資料

任務	Description	所需的技能
為 Lambda 函數建立資料庫使用者。	最佳實務是針對應用程式的不同部分使用不同的資料庫使用者。如果您的 Cron 任務已存在不同的資料庫使用者，請使用它。否則，請建立新的資料	DBA

任務	Description	所需的技能
	庫使用者。如需詳細資訊，請參閱管理 PostgreSQL 使用者和角色 (AWS 部落格文章）。	
在 Secrets Manager 中將資料庫登入資料儲存為秘密。	遵循建立資料庫秘密 (Secrets Manager 文件） 中的指示。	DBA、DevOps

編寫 Lambda 函數的程式碼

任務	Description	所需的技能
選擇 Lambda 支援的程式設計語言。	如需支援的語言清單，請參閱 Lambda 執行時間 (Lambda 文件）。	開發人員
撰寫邏輯以從 Secrets Manager 擷取資料庫登入資料。	如需範例程式碼，請參閱如何使用 安全地提供資料庫登入資料給 Lambda 函數 AWS Secrets Manager(AWS 部落格文章）。	開發人員
撰寫邏輯以執行排定的資料庫活動。	將您現場部署使用之排程任務的現有程式碼遷移至 Lambda 函數。如需詳細資訊，請參閱部署 Lambda 函數 (Lambda 文件）。	開發人員

部署程式碼並建立 Lambda 函數

任務	Description	所需的技能
建立 Lambda 函數部署套件。	此套件包含程式碼及其相依性。如需詳細資訊，請參閱部署套件 (Lambda 文件）。	開發人員

任務	Description	所需的技能
建立 Lambda 函數。	在 Lambda 主控台中，選擇建立函數，輸入函數名稱，選擇執行時間環境，然後選擇建立函數。	DevOps
上傳部署套件。	選擇您建立的 Lambda 函數以開啟其組態。您可以直接在程式碼區段中撰寫程式碼，或上傳部署套件。若要上傳套件，請前往函數程式碼區段，選擇程式碼項目類型以上傳 .zip 檔案，然後選取套件。	DevOps
根據您的需求設定 Lambda 函數。	例如，您可以將逾時參數設定為您預期 Lambda 函數需要的持續時間。如需詳細資訊，請參閱設定函數選項 (Lambda 文件）。	DevOps
設定 Lambda 函數角色的許可，以存取 Secrets Manager。	如需說明，請參閱在 AWS Lambda 函數中使用秘密 (Secrets Manager 文件）。	DevOps
測試 Lambda 函數。	手動啟動 Lambda 函數，以確保其如預期般運作。	DevOps

使用 CloudWatch Events 排程 Lambda 函數

任務	Description	所需的技能
建立規則以依排程執行 Lambda 函數。	使用 CloudWatch Events 排程 Lambda 函數。如需說明，請參閱使用 CloudWatch Events 排程 Lambda 函數 (CloudWatch Events 教學課程）。	DevOps

相關資源

- AWS Secrets Manager
- Lambda 入門
- 建立在事件上觸發的 CloudWatch 事件規則
- AWS Lambda 限制
- 從無伺服器應用程式查詢 AWS 資料庫（部落格文章）

使用內部部署 SMTP 伺服器和 Database Mail 傳送 Amazon RDS for SQL Server 資料庫執行個體的通知

Nishad Mankar，Amazon Web Services

總結

Database Mail (Microsoft 文件） 使用 Simple Mail Transfer Protocol (SMTP) 伺服器，從 Microsoft SQL Server 資料庫傳送電子郵件訊息，例如通知或提醒。Amazon Relational Database Service (Amazon RDS) for Microsoft SQL Server 文件提供使用 Amazon Simple Email Service (Amazon SES) 做為 Database Mail SMTP 伺服器的說明。如需詳細資訊，請參閱在 Amazon RDS for SQL Server 上使用 Database Mail。做為替代組態，此模式說明如何使用內部部署 SMTP 伺服器做為郵件伺服器，將 Database Mail 設定為從 Amazon RDS for SQL Server 資料庫 (DB) 執行個體傳送電子郵件。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 執行 SQL Server 標準版或企業版的 Amazon RDS 資料庫執行個體
- 內部部署 SMTP 伺服器的 IP 地址或主機名稱
- 傳入安全群組規則，允許從 SMTP 伺服器的 IP 地址連線至 Amazon RDS for SQL Server 資料庫執行個體
- 內部部署網路與包含 Amazon RDS 資料庫執行個體的虛擬私有雲端 (VPC) 之間的連線，例如 AWS Direct Connect 連線

限制

- 不支援 SQL Server 的 Express 版本。
- 如需限制的詳細資訊，請參閱《Amazon RDS 文件》中的在 Amazon RDS for SQL Server 上使用資料庫郵件中的限制。

產品版本

- RDS 中支援的 SQL Server 版本標準版和企業版

Architecture

目標技術堆疊

- Amazon RDS for SQL Server 資料庫執行個體
- Amazon Route 53 轉送規則
- 資料庫郵件
- 內部部署 SMTP 伺服器
- Microsoft SQL Server Management Studio (SSMS)

目標架構

下圖顯示此模式的目標架構。發生事件或動作啟動有關資料庫執行個體的通知或提醒時，Amazon RDS for SQL Server 會使用 Database Mail 傳送電子郵件通知。Database Mail 使用內部部署 SMTP 伺服器來傳送電子郵件。

工具

AWS 服務

- 適用於 Microsoft SQL Server 的 Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展 SQL Server 關聯式資料庫。
- Amazon Route 53 是一種可用性高、可擴展性強的 DNS Web 服務。

其他工具

- Database Mail 是一種工具，可將通知和提醒等電子郵件訊息從 SQL Server 資料庫引擎傳送給使用者。
- Microsoft SQL Server Management Studio (SSMS) 是一種用於管理 SQL Server 的工具，包括存取、設定和管理 SQL Server 元件。在此模式中，您可以使用 SSMS 執行 SQL 命令，在 Amazon RDS for SQL Server 資料庫執行個體上設定 Database Mail。

史詩

啟用與內部部署 SMTP 伺服器的網路連線

任務	Description	所需的技能
從 RDS 資料庫執行個體移除異地同步備份。	如果您使用的是多區域 RDS 資料庫執行個體，請將多可用區域執行個體轉換為單一可用區域執行個體。當您完成設定 Database Mail 後，會將資料庫執行個體轉換回異地同步備份部署。Database Mail 組態接著可在主要節點和次要節點中運作。如需說明，請參閱從 Microsoft SQL Server 資料庫執行個體移除異地同步備份。	DBA
在內部部署 SMTP 伺服器上建立 Amazon RDS 端點或 IP 地址的允許清單。	SMTP 伺服器位於 AWS 網路之外。在內部部署 SMTP 伺服器上，建立允許清單，允許伺服器與 Amazon RDS 執行個體或 Amazon RDS 上託管的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的傳出端點或 IP 地址進行通訊。此程序因組織而異。如需資料庫執行個體端點的詳細資訊，請參閱尋找資料庫執行個體端點和連接埠號碼。	DBA
移除連接埠 25 限制。	根據預設，AWS 會限制 EC2 執行個體上的連接埠 25。若要移除連接埠 25 限制，請執行下列動作：	一般 AWS

任務	Description	所需的技能
	1. 使用您的 AWS 帳戶登入，然後開啟移除電子郵件傳送限制請求表單。 2. 輸入您的電子郵件地址，讓 AWS Support 可以與您聯絡，告知您請求的更新。 3. 在使用案例描述欄位中提供必要資訊。 4. 選擇提交。 **Note** • 如果您在多個 AWS 區域中有執行個體，請為每個區域提交個別的請求。 • 處理您的請求最多可能需要 48 小時。	
新增 Route 53 規則來解析 SMTP 伺服器的 DNS 查詢。	使用 Route 53 解析 AWS 資源與內部部署 SMTP 伺服器之間的 DNS 查詢。您必須建立將 DNS 查詢轉送至 SMTP 伺服器網域的規則，例如 `example.com` 。如需說明，請參閱 Route 53 文件中的建立轉送規則。	網路管理員

在 Amazon RDS for SQL Server 資料庫執行個體上設定 Database Mail

任務	Description	所需的技能
啟用資料庫郵件。	建立 Database Mail 的參數群組，將 `database mail xps` 參數設定為 `1`，然後將 Database Mail 參數群組與目標 RDS 資料庫執行個體建立關聯。如需說明，請參閱 Amazon RDS 文件中的啟用資料庫郵件。請勿繼續進行這些說明中的設定資料庫郵件一節。內部部署 SMTP 伺服器的組態與 Amazon SES 不同。	DBA
連線到資料庫執行個體。	從堡壘主機，使用 Microsoft SQL Server Managemen t Studio (SSMS) 連線至 Amazon RDS for SQL Server 資料庫執行個體。如需說明，請參閱連線至執行 Microsoft SQL Server 資料庫引擎的資料庫執行個體。如果您遇到任何錯誤，請參閱相關資源區段中的連線故障診斷參考。	DBA
建立設定檔。	在 SSMS 中，輸入下列 SQL 陳述式來建立 Database Mail 設定檔。取代以下的值： • 針對 `profile_name` ，輸入新設定檔的名稱。 • 針對 `description` ，輸入新設定檔的簡短描述。	DBA

任務	Description	所需的技能
	如需此預存程序及其引數的詳細資訊，請參閱 Microsoft 文件中的 sysmail_add_profile_sp。 `EXECUTE msdb.dbo.sysmail_add_profile_sp @profile_name = 'SQL Alerts profile', @description = 'Profile used for sending outgoing notifications using OM SMTP Server.';`	

任務	Description	所需的技能
將主體新增至設定檔。	輸入下列 SQL 陳述式，將公有或私有主體新增至 Database Mail 設定檔。主體是可以要求 SQL Server 資源的實體。取代以下的值： • 針對 `profile_name` ，輸入您先前建立的設定檔名稱。 • 針對 `principal_name` ，輸入資料庫使用者或角色的名稱。此值必須對應至 SQL Server 身分驗證使用者、Windows 身分驗證使用者或 Windows 身分驗證群組。 如需此預存程序及其引數的詳細資訊，請參閱 Microsoft 文件中的 sysmail_add_principalprofile_sp。 `EXECUTE msdb.dbo.sysmail_add_principalprofile_sp @profile_name = 'SQL Alerts profile', @principal_name = 'public', @is_default = 1 ;`	DBA

任務	Description	所需的技能
建立 帳戶。	輸入下列 SQL 陳述式來建立 Database Mail 帳戶。取代以下的值： • 針對 `account_name` ，輸入新帳戶的名稱。 • 針對 `description` ，輸入新帳戶的簡短描述。 • 針對 `email_address` ，輸入要從中傳送 Database Mail 訊息的電子郵件地址。 • 針對 `display_address` ，輸入要用於此帳戶傳出訊息的顯示名稱，例如 `SQL Server Automated Notification` 。您也可以使用為 輸入的值`email_address` 。 • 針對 `mailserver_name` ，輸入 SMTP 郵件伺服器的名稱或 IP 地址。 • 對於 `port`，請保留 的值`25`。 • 對於 `enable_ssl` ，`0`如果您不希望 Database Mail 使用 SSL 加密通訊，請將值保留在 `1`或輸入 。 • 針對 `username`，輸入登入 SMTP 郵件伺服器的使用者名稱。如果伺服器不需要身分驗證，請輸入 `NULL`。 • 針對 `password`，輸入登入 SMTP 郵件伺服器的密碼。	DBA

任務	Description	所需的技能

	如果伺服器不需要身分驗證，請輸入 NULL。
	
	如需此預存程序及其引數的詳細資訊，請參閱 Microsoft 文件中的 sysmail_add_account_sp。
	
	`EXECUTE msdb.dbo.sysmail_add_account_sp`
	`@account_name = 'SQL Alerts account',`
	`@description = 'Database Mail account for sending outgoing notifications.',`
	`@email_address = 'xyz@example.com',`
	`@display_name = 'xyz@example.com',`
	`@mailserver_name = 'test_smtp.example.com',`
	`@port = 25,`
	`@enable_ssl = 1,`
	`@username = 'SMTP-username',`
	`@password = 'SMTP-password';`	

任務	Description	所需的技能
將帳戶新增至設定檔。	輸入下列 SQL 陳述式，將 Database Mail 帳戶新增至 Database Mail 設定檔。取代以下的值： • 針對 `profile_name` ，輸入您先前建立的設定檔名稱。 • 針對 `account_name` ，輸入您先前建立的帳戶名稱。 如需此預存程序及其引數的詳細資訊，請參閱 Microsoft 文件中的 sysmail_add_profileaccount_sp。 `EXECUTE msdb.dbo.sysmail_add_profileaccount_sp @profile_name = 'SQL Alerts profile', @account_name = 'SQL Alerts account', @sequence_number = 1;`	DBA
（選用）將異地同步備份新增至 RDS 資料庫執行個體。	如果您想要使用資料庫鏡像 (DBM) 或 Always On 可用性群組 (AGs) 新增異地同步備份，請參閱將異地同步備份新增至 Microsoft SQL Server 資料庫執行個體中的指示。	DBA

相關資源

- 在 Amazon RDS for SQL Server 上使用資料庫郵件 (Amazon RDS 文件）

- 使用檔案附件 (Amazon RDS 文件）
- 針對 SQL Server 資料庫執行個體的連線進行故障診斷 (Amazon RDS 文件）
- 無法連線至 Amazon RDS 資料庫執行個體 (Amazon RDS 文件）

在 AWS 上設定 SAP on IBM Db2 的災難復原

Ambarish Satarkar and Debasis Sahoo，Amazon Web Services

總結

此模式概述使用 IBM Db2 做為資料庫平台，在 Amazon Web Services (AWS) 雲端上執行，為 SAP 工作負載設定災難復原 (DR) 系統的步驟。目標是提供低成本的解決方案，以便在發生中斷時提供業務持續性。

模式使用指示燈方法。透過在 AWS 上實作指示燈 DR，您可以減少停機時間並維持業務連續性。指示燈方法著重於在 AWS 中設定最小 DR 環境，包括與生產環境同步的 SAP 系統和待命 Db2 資料庫。

此解決方案可擴展。您可以視需要將其擴展到完整規模的災難復原環境。

先決條件和限制

先決條件

- 在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上執行的 SAP 執行個體
- IBM Db2 資料庫
- SAP 產品可用性矩陣 (PAM) 支援的作業系統
- 生產和待命資料庫主機的不同實體資料庫主機名稱
- 每個啟用跨區域複寫 (CRR) 的 AWS 區域中的 Amazon Simple Storage Service (Amazon S3) 儲存貯體 https://docs.aws.amazon.com/AmazonS3/latest/userguide/replication.html

產品版本

- IBM Db2 資料庫版本 11.5.7 或更新版本

Architecture

目標技術堆疊

- Amazon EC2
- Amazon Simple Storage Service (Amazon S3)
- Amazon Virtual Private Cloud (VPC 對等互連）
- Amazon Route 53

- IBM Db2 高可用性災難復原 (HADR)

目標架構

此架構會使用 Db2 做為資料庫平台，為 SAP 工作負載實作 DR 解決方案。生產資料庫部署在 AWS 區域 1，待命資料庫則部署在第二個區域。待命資料庫稱為 DR 系統。Db2 資料庫支援多個待命資料庫（最多三個）。它使用 Db2 HADR 來設定 DR 資料庫，並在生產和待命資料庫之間自動化日誌運送。

如果發生使區域 1 無法使用的災難，DR 區域中的待命資料庫會接管生產資料庫角色。SAP 應用程式伺服器可以事先建置，或使用 AWS Elastic Disaster Recovery 或 Amazon Machine Image (AMI) 來滿足復原時間目標 (RTO) 需求。此模式使用 AMI。

Db2 HADR 實作生產待命設定，其中生產做為主要伺服器，且所有使用者都與其連線。所有交易都會寫入日誌檔案，這些檔案會使用 TCP/IP 傳輸到待命伺服器。待命伺服器透過轉傳傳輸的日誌記錄來更新其本機資料庫，這有助於確保其與生產伺服器保持同步。

使用 VPC 對等互連，以便生產區域和 DR 區域中的執行個體可以互相通訊。Amazon Route 53 會將最終使用者路由至網際網路應用程式。

1. 在區域 1 中建立應用程式伺服器的 AMI，並將 AMI 複製到區域 2。發生災難時，使用 AMI 在區域 2 中啟動伺服器。
2. 在生產資料庫 （ 區域 1) 和待命資料庫 （ 區域 2) 之間設定 Db2 HADR 複寫。
3. 變更 EC2 執行個體類型，以符合發生災難時的生產執行個體。
4. 在區域 1 中， `LOGARCHMETH1` 設定為 `db2remote: S3 path`。
5. 在區域 2 中， `LOGARCHMETH1` 設定為 `db2remote: S3 path`。
6. 跨區域複寫會在 S3 儲存貯體之間執行。

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon Route 53 是一種可用性高、可擴展性強的 DNS Web 服務。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路類似於您在自己的資料中心內操作的傳統網路，具有使用可擴展的 AWS 基礎設施的優勢。此模式使用 VPC 對等互連。

最佳實務

- 網路在決定 HADR 複寫模式時扮演重要角色。對於跨 AWS 區域的 DR，我們建議您使用 Db2 HADR ASYNC 或 SUPERASYNC 模式。
- 如需 Db2 HADR 複寫模式的詳細資訊，請參閱 IBM 文件。
- 您可以使用 AWS 管理主控台或 AWS 命令列界面 (AWS CLI) 來建立現有 SAP 系統的新 AMI。然後，您可以使用 AMI 來復原現有的 SAP 系統或建立複製。
- AWS Systems Manager Automation 可協助 EC2 執行個體和其他 AWS 資源的常見維護和部署任務。
- AWS 提供多個原生服務來監控和管理 AWS 上的基礎設施和應用程式。Amazon CloudWatch 和 AWS CloudTrail 等服務可以分別用來監控基礎基礎設施和 API 操作。如需詳細資訊，請參閱 SAP on AWS – IBM Db2 HADR with Pacemaker。

史詩

準備環境

任務	Description	所需的技能
檢查系統和日誌。	1. 確認已設定 Db2 上的生產 SAP。 2. 確認日誌備份已開啟並設定為將日誌儲存在 S3 儲存貯體中。這可由 Db2 參數 檢查`LOGARCHMETH1` 。 3. 建立其他應用程式伺服器的 AMI。	AWS 管理員、SAP Basis 管理員

設定伺服器和複寫

任務	Description	所需的技能
建立 SAP 和資料庫伺服器。	1. 若要部署 DR 區域的基礎設施，請使用 AWS CloudFormation 指令碼或使用生產執行個體的 AMI。做為指示燈方法的一部分，您可以在與生產執行個體相同的系列中使用較小的 EC2 執行個體。例如，如果您的生產執行個體類型為 `r6i.12xlarge` ，您可以使用 DR 組建的`r6i.xlarge` 執行個體類型。不過，請確定您在 DR 執行個體上配置相同的儲存容量，以還原生產資料庫備份。 2. 為 建立 Amazon Elastic File System (Amazon EFS) 掛載點`/sapmnt/<SID>/` ，並確認它已設定為從主要系統複寫。 3. 從生產系統進行 FULL 資料庫備份 （線上或離線）。您將使用此備份來建置 DR 資料庫。 4. 在 DR 系統中，使用 SAP 軟體佈建管理員 (SWPM) 系統複製方法搭配 使用系統複製搭配備份/還原用於 HA/DR，以建置 DR SAP 系統。 5. 當收到請求時，請使用您從生產環境中取得的備份在	SAP Basis 管理員

任務	Description	所需的技能
	DR 中還原資料庫。DR 資料庫將處於向前滾動擱置狀態。 在還原完整備份之後，依預設會設定向前滾動擱置狀態。向前移動擱置狀態表示資料庫正在還原中，而且可能需要套用一些變更。如需詳細資訊，請參閱 IBM 文件。	

任務	Description	所需的技能

檢查組態。	1. 若要設定 HADR 的日誌封存，生產和 DR 資料庫都必須能夠自動從所有日誌封存位置擷取日誌。確認 DR 資料庫中的 LOGARCHMETH1 參數已設定為與生產資料庫中相同的位置。如果因為區域限制而無法存取相同的位置，請確定 DR 系統可以從主要系統自動擷取日誌。
	2. 若要啟用資料庫複寫啟用的 TCP/IP 連接埠，請在生產和 DR 主機/etc/services 中新增下列兩個項目來修改 。在程式碼中，<SID>是指 Db2 資料庫的系統 ID (SID) （例如 PR1)。
	`<SID>_HADR_1 55001/tcp # DB2 HADR Port1`
	`<SID>_HADR_2 55002/tcp # DB2 HADR Port2`
	確認兩個連接埠都允許主要和待命之間的傳入和傳出流量。
	3. /etc/hosts 檢查生產和 DR 主機，確認生產和待命主機的主機名稱都指向正確的 IP 地址。
		AWS 管理員、SAP Basis 管理員

任務	Description	所需的技能

設定從生產資料庫到 DR 資料庫的複寫（使用 ASYNC 模式）。	1. 在生產資料庫中，執行下列命令來更新參數。
	`db2 UPDATE DB CFG FOR <SID> USING HADR_LOCAL_HOST HOST1`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_LOCAL_SVC <SID>_HADR_1`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_REMOTE_HOST HOST2`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_REMOTE_SVC <SID>_HADR_2`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_REMOTE_INST db2<sid>`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_TIMEOUT 120`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_SYNCMODE ASYNC`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_SPOOL_LIMIT 1000`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_PEER_WINDOW 240`
	`db2 UPDATE DB CFG FOR <SID> USING indexrec RESTART logindexbuild ON`
	2. 在 DR 資料庫中，執行下列命令來更新參數。
		SAP Basis 管理員

任務	Description	所需的技能

	`db2 UPDATE DB CFG FOR <SID> USING HADR_LOCAL_HOST HOST2`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_LOCAL_SVC <SID>_HADR_2`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_REMOTE_HOST HOST1`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_REMOTE_SVC <SID>_HADR_1`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_REMOTE_INST db2<sid>`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_TIMEOUT 120`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_SYNCMODE ASYNC`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_SPOOL_LIMIT 1000`
	`db2 UPDATE DB CFG FOR <SID> USING HADR_PEER_WINDOW 240`
	`db2 UPDATE DB CFG FOR <SID> USING indexrec RESTART logindexbuild ON`
	
	需要這些參數才能將 HADR 相關資訊提供給這兩個資料庫。在 Db2 資料庫中，HADR 會根據先前設定的每個參數的值啟動。如需這些參數的詳細資訊，請參閱 IBM 文件。	

任務	Description	所需的技能

	3. 使用以下命令，先在新建立的待命資料庫上啟動 HADR。
	
	`db2 deactivate db <SID>`
	`db2 start hadr on db <SID> as standby`
	
	4. 使用下列命令啟動生產資料庫上的 HADR。
	
	`db2 deactivate db <SID>`
	`db2 start hadr on db <SID> as primary`
	
	5. 檢查生產和待命 Db2 資料庫是否同步，以及日誌運送是否持續進行。
	
	若要監控 HADR 複寫狀態，請使用下列db2pd命令。
	
	`db2pd -d <SID> -hadr`
	
	如需監控 HADR 的詳細資訊，請參閱 IBM 文件。	

測試 DR 容錯移轉任務

任務	Description	所需的技能
規劃 DR 測試的生產業務停機時間。	請務必在生產環境中規劃必要的業務停機時間，以測試 DR 容錯移轉案例。	SAP Basis 管理員

任務	Description	所需的技能
建立測試使用者。	建立可在 DR 主機中驗證的測試使用者（或任何測試變更），以確認 DR 容錯移轉後的日誌複寫。	SAP Basis 管理員
在 主控台上，停止生產 EC2 執行個體。	在此步驟中會啟動不規律關機，以模擬災難案例。	AWS 系統管理員

擴展 DR EC2 執行個體以符合需求。	在 EC2 主控台上，變更 DR 區域中的執行個體類型。
	1. 停止執行個體：如果執行個體正在執行，您必須先停止它，才能變更其執行個體類型。在 EC2 主控台上，選取執行個體，然後選擇停止。
	2. 修改執行個體類型：在 EC2 主控台上，選取執行個體，然後選擇動作、執行個體設定、變更執行個體類型。選取符合主要執行個體的執行個體類型，然後選擇套用。
	3. 啟動執行個體：在執行個體類型變更完成後，從 EC2 主控台選取執行個體並選擇啟動，以啟動執行個體。
	4. 若要啟動 Db2 資料庫，請使用下列命令。
	`db2start`
	`db2 start HADR on db <SID> as standby`
		SAP 基礎管理員

任務	Description	所需的技能

啟動接管。	從 DR 系統 (host2) 啟動接管程序，並將 DR 資料庫做為主要資料庫。
	`db2 takeover hadr on database <SID> by force`
	或者，您可以設定下列參數，根據執行個體類型自動調整資料庫記憶體配置。值INSTANCE_MEMORY 可以根據要配置給 Db2 資料庫的記憶體專用部分來決定。
	`db2 update db cfg for <SID> using INSTANCE_MEMORY <FIXED VALUE> IMMEDIATE;`
	`db2 get db cfg for <SID> | grep -i DATABASE_MEMORY AUTOMATIC IMMEDIATE;`
	`db2 update db cfg for <SID> using self_tuning_mem ON IMMEDIATE;`
	使用以下命令驗證變更。
	`db2 get db cfg for <SID> | grep -i MEMORY`
	`db2 get db cfg for <SID> | grep -i self_tuning_mem`
		SAP Basis 管理員

在 DR 區域中啟動 SAP 的應用程式伺服器。	使用您從生產系統建立的 AMI，在 DR 區域中啟動新的額外應用程式伺服器。	SAP Basis 管理員

任務	Description	所需的技能
在啟動 SAP 應用程式之前執行驗證。	1. 驗證 /etc/hosts 和 /etc/fstab 項目。 2. 在 DR 系統上掛/sapmnt/<SID>/ 載。 3. 驗證 DR 檔案系統/sapmnt/<SID>/ 已與生產同步/sapmnt/<SID>/ 。 4. 登入<sid>adm使用者、執行 R3trans -d，並驗證 trans.log 檔案中的輸出。trans.log 檔案會在您執行 R3trans -d命令的相同位置產生。	AWS 管理員、SAP Basis 管理員

在 DR 系統上啟動 SAP 應用程式。	使用 <sid>adm使用者在 DR 系統上啟動 SAP 應用程式。使用下列程式碼，其中 XX代表 SAP ABAP SAP Central Services (ASCS) 伺服器的執行個體編號，而 YY代表 SAP 應用程式伺服器的執行個體編號。
	`sapconrol -nr XX -function StartService <SID>`
	`sapconrol -nr XX -function StartSystem`
	`sapconrol -nr YY -function StartService <SID>`
	`sapconrol -nr YY -function StartSystem`
		SAP Basis 管理員

任務	Description	所需的技能
執行 SAP 驗證。	這是做為 DR 測試來執行，以提供證據或檢查 DR 區域的資料複寫成功。	測試工程師

執行 DR 容錯回復任務

任務	Description	所需的技能
啟動生產 SAP 和資料庫伺服器。	在 主控台上，啟動在生產系統中託管 SAP 和資料庫的 EC2 執行個體。	SAP Basis 管理員

啟動生產資料庫並設定 HADR。	使用下列命令登入生產系統 (host1) 並確認資料庫處於復原模式。
	`db2start`
	`db2 start HADR on db P3V as standby`
	`db2 connect to <SID>`
	確認 HADR 狀態為 `connected` 。複寫狀態應為 `peer`。
	`db2pd -d <SID> -hadr`
	如果資料庫不不一致，且未處於 `connected` 和 `peer` 狀態，則備份和還原可能需要讓資料庫（位於 host1) 與目前作用中的資料庫 (host2位於 DR 區域中）同步。在這種情況下，請將資料庫備份從 DR host2 區域中的資料庫還原
		SAP Basis 管理員

任務	Description	所需的技能
	至host1生產區域中的資料庫。	

將資料庫容錯移轉回生產區域。	在正常business-as-usual情況下，此步驟會在排程的停機時間中執行。在 DR 系統上執行的應用程式會停止，且資料庫會失敗回生產區域（區域 1），以從生產區域恢復操作。
	1. 登入 DR 區域中的 SAP 應用程式伺服器，然後停止 SAP 應用程式。
	2. /sapmnt/<SID> 從 DR 系統卸載，確保變更已反向複寫至/sapmnt/<SID> 生產系統的 。
	3. 登入生產區域中的資料庫伺服器 (host1)，然後執行接管。
	`db2 takeover hadr on database <SID>`
	4. 檢查 HADR 狀態：HADR_ROLE 應該在 PRIMARY host1和 StandBy上host2。
	`db2pd -d <SID> -hadr`
		SAP Basis 管理員

任務	Description	所需的技能
在啟動 SAP 應用程式之前執行驗證。	1. 驗證 /etc/hosts 和 /etc/fstab 項目。 2. 在生產系統上掛/sapmnt/<SID>/ 載。 3. 請確定它與 DR 系統 同步/sapmnt/<SID>/ 。 4. 登入<sid>adm使用者、執行 R3trans -d，並驗證 trans.log 檔案中的輸出。trans.log 檔案會在您執行 R3trans -d命令的相同位置產生。	AWS 管理員、SAP Basis 管理員

啟動 SAP 應用程式。	1. 使用 <sid>adm使用者在生產系統上啟動 SAP 應用程式。使用下列程式碼，其中 XX代表 SAP ASCS 伺服器的執行個體編號，而 YY代表 SAP 應用程式伺服器的執行個體編號。
	`sapconrol -nr XX -function StartService <SID>`
	`sapconrol -nr XX -function StartSystem`
	`sapconrol -nr YY -function StartService <SID>`
	`sapconrol -nr YY -function StartSystem`
	2. 若要確認應用程式伺服器是否可用，請登入 SAP 並使用 SICK 和 SM51 交易執行檢查。
		SAP Basis 管理員

疑難排解

問題	解決方案
用於排除 HADR 相關問題的金鑰日誌檔案和命令	• `db2 get db cfg | grep -i hadr` • `db2pd -d sid -hadr` • `Db2diag.log` （此檔案通常位於 `db2dump`目錄中，`db2dump`路徑由 參數 定義`DIAGPATH`。）
疑難排解 Db2 UDB 上 HADR 問題的 SAP 注意事項	請參閱 SAP Note 1154013 - DB6：HADR 環境中的資料庫問題。（您需要 SAP 入口網站登入資料才能存取此備註。）

相關資源

- AWS 上 Db2 資料庫的災難復原方法（部落格文章）
- SAP on AWS – IBM Db2 HADR 搭配 Pacemaker
- 在DB2 資料庫之間設定 HADR 複寫的逐步程序
- Db2 HADR Wiki

其他資訊

使用此模式，您可以為在 Db2 資料庫上執行的 SAP 系統設定災難復原系統。在災難情況下，業務應該能夠在您定義的復原時間目標 (RTO) 和復原點目標 (RPO) 要求內繼續：

- RTO 是服務中斷和服務還原之間的最大可接受延遲。這會決定可接受的服務無法使用之時間長度。
- RPO 是自上次資料復原點以來可接受的時間上限。這會決定最後一個復原點與服務中斷之間可接受的資料遺失。

如需 HADR 相關的FAQs，請參閱 SAP 備註 #1612105 - DB6：Db2 高可用性災難復原 (HADR) 的常見問答集。（您需要 SAP 入口網站登入資料才能存取此備註。）

使用 Terraform 設定資料庫遷移的 CI/CD 管道

Rahul Sharad Gaikwad、Ashish Bhatt、Aniket Dekate、Ruchika Modi、Tamilselvan P、Nadeem Rahaman、Aarti Rajput 和 Naveen Suthar、Amazon Web Services

總結

此模式旨在建立持續整合和持續部署 (CI/CD) 管道，以可靠且自動化的方式管理資料庫遷移。它涵蓋了佈建必要基礎設施、遷移資料以及使用 Terraform 自訂結構描述變更的程序，Terraform 是一種基礎設施即程式碼 (IaC) 工具。

具體而言， 模式會設定 CI/CD 管道，將內部部署 Microsoft SQL Server 資料庫遷移至 上的 Amazon Relational Database Service (Amazon RDS) AWS。您也可以使用此模式，將虛擬機器 (VM) 或其他雲端環境中的 SQL Server 資料庫遷移至 Amazon RDS。

此模式可解決下列與資料庫管理和部署相關的挑戰：

- 手動資料庫部署非常耗時、容易出錯，而且缺乏跨環境的一致性。
- 協調基礎設施佈建、資料遷移和結構描述變更可能很複雜且難以管理。
- 確保資料完整性並將資料庫更新期間的停機時間降至最低，對生產系統至關重要。

此模式提供下列優點：

- 透過實作資料庫遷移的 CI/CD 管道，簡化更新和部署資料庫變更的程序。這可降低錯誤風險、確保跨環境的一致性，並將停機時間降至最低。
- 有助於改善可靠性、效率和協同合作。可在資料庫更新期間加快上市時間並減少停機時間。
- 協助您針對資料庫管理採用現代 DevOps 實務，進而在軟體交付程序中提高敏捷性、可靠性和效率。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 本機電腦上已安裝 Terraform 0.12 或更新版本 （ 如需說明，請參閱 Terraform 文件)
- 來自 HashiCorp 的 Terraform AWS Provider 3.0.0 版或更新版本 （ 請參閱此供應商的 GitHub 儲存庫)

- 最低權限 AWS Identity and Access Management (IAM) 政策 （請參閱部落格文章撰寫最低權限 IAM 政策的技術)

Architecture

此模式實作下列架構，為資料庫遷移程序提供完整的基礎設施。

在此架構中：

- 來源資料庫是現場部署、虛擬機器 (VM) 或由其他雲端供應商託管的 SQL Server 資料庫。此圖假設來源資料庫位於內部部署資料中心。
- 內部部署資料中心和 透過 VPN 或 AWS 連線進行 AWS Direct Connect 連線。這可提供來源資料庫與 AWS 基礎設施之間的安全通訊。
- 目標資料庫是在資料庫佈建管道的協助 AWS 下，託管在 虛擬私有雲端 (VPC) 內的 Amazon RDS 資料庫。
- AWS Database Migration Service (AWS DMS) 會將您的內部部署資料庫複寫到 AWS。它用於設定來源資料庫到目標資料庫的複寫。

下圖顯示設定不同層級資料庫遷移程序的基礎設施，其中包含佈建、 AWS DMS 設定和驗證。

在此程序中：

- 驗證管道會驗證所有檢查。當所有必要的驗證完成時，整合管道會移至下一個步驟。
- 資料庫佈建管道由各種階段組成，這些 AWS CodeBuild 階段會對資料庫提供的 Terraform 程式碼執行 Terraform 動作。當這些步驟完成時，它會在目標中部署資源 AWS 帳戶。
- AWS DMS 管道包含各種 CodeBuild 階段，這些階段會執行測試，然後使用 IaC 佈建執行遷移的 AWS DMS 基礎設施。

工具

AWS 服務 和 工具

- AWS CodeBuild 是一種全受管持續整合服務，可編譯原始程式碼、執行測試，並產生ready-to-deploy的軟體套件。

- AWS CodePipeline 是一種全受管持續交付服務，可協助您自動化發行管道，以實現快速可靠的應用程式和基礎設施更新。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 中設定、操作和擴展關聯式資料庫 AWS 雲端。
- Amazon Simple Storage Service (Amazon S3) 是一種物件儲存服務，可提供可擴展性、資料可用性、安全性和效能。
- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移到 AWS 雲端 或在雲端和內部部署設定的組合之間遷移。

其他服務

- Terraform 是 HashiCorp 的 IaC 工具，可協助您建立和管理雲端和內部部署資源。

程式碼儲存庫

此模式的程式碼可在 GitHub Database Migration DevOps Framework 中使用 Terraform 範例儲存庫。

最佳實務

- 實作資料庫遷移的自動化測試，以驗證結構描述變更和資料完整性的正確性。這包括單元測試、整合測試和end-to-end測試。
- 為您的資料庫實作強大的備份和還原策略，特別是在遷移之前。這可確保資料完整性，並在發生故障時提供備用選項。
- 實作強大的復原策略，以在遷移期間發生故障或問題時還原資料庫變更。這可能包括轉返到先前的資料庫狀態或還原個別遷移指令碼。
- 設定監控和記錄機制，以追蹤資料庫遷移的進度和狀態。這可協助您快速識別和解決問題。

史詩

設定您的本機工作站

任務	Description	所需的技能
在本機工作站上設定 Git。	遵循 Git 文件中的指示，在本機工作站上安裝和設定 Git。	DevOps 工程師

任務	Description	所需的技能

建立專案資料夾，並從 GitHub 儲存庫新增檔案。	1. 開啟此模式的 GitHub 儲存庫。
	2. 選擇程式碼以查看複製選項，然後複製 HTTPS 索引標籤中提供的 URL。
	3. 在工作站上為您的專案建立資料夾。
	4. 開啟終端機並導覽至此資料夾。
	5. 複製 GitHub 儲存庫：
	`git clone <github-repository-url>`
	其中 `<github-repository-url>` 是您在步驟 2 中複製的 URL。
	6. 複製完成後，請前往專案資料夾中的複製儲存庫：
	`cd <folder-name>/aws-terraform-db-migration-framework-samples`
	7. 在您選擇的整合式開發環境 (IDE) 中開啟此專案。
		DevOps 工程師

佈建目標架構

任務	Description	所需的技能
更新必要的參數。	`ssm-parameters.sh` 檔案會存放所有必要 AWS Systems	DevOps 工程師

任務	Description	所需的技能
	Manager 的參數。您可以使用專案的自訂值來設定這些參數。 在本機工作站的 `setup/db-ssm-params` 資料夾中，開啟 `ssm-parameters.sh` 檔案並設定這些參數，再執行 CI/CD 管道。	
初始化 Terraform 組態。	在 `db-cicd-integration` 資料夾中，輸入下列命令來初始化包含 Terraform 組態檔案的工作目錄： `terraform init`	DevOps 工程師
預覽 Terraform 計劃。	若要建立 Terraform 計劃，請輸入下列命令： `terraform plan -var-file="terraform.sample"` Terraform 會評估組態檔案，以判斷宣告資源的目標狀態。然後，它會比較目標狀態與目前狀態，並建立計劃。	DevOps 工程師
驗證計劃。	檢閱計劃並確認其已在您的目標中設定所需的架構 AWS 帳戶。	DevOps 工程師

任務	Description	所需的技能
部署解決方案。	1. 輸入下列命令以套用計劃： `terraform apply -var-file="terraform.sample"` 2. 輸入 `yes` 以確認。Terraform 會建立、更新或銷毀基礎設施，以達到組態檔案中宣告的目標狀態。如需序列的詳細資訊，請參閱此模式的架構區段。	DevOps 工程師

驗證部署

任務	Description	所需的技能
驗證部署。	驗證`db-cicd-integration` 管道的狀態，以確認資料庫遷移已完成。 1. 登入 AWS 管理主控台，然後開啟 AWS CodePipeline 主控台。 2. 在導覽窗格中，選擇管道。 3. 選擇`db-cicd-integration` 管道。 4. 驗證管道執行是否成功完成。	DevOps 工程師

使用後清除基礎設施

任務	Description	所需的技能
清除基礎設施。	1. 專案完成後，請使用 命令清除您建立的基礎設施： `terraform destroy --var-file=terrafo rm.sample` 2. 輸入 `yes` 以確認。	DevOps 工程師

相關資源

AWS 文件

- Terraform 產品入門

Terraform 文件

- Terraform 安裝
- Terraform 後端組態
- Terraform AWS 提供者文件

使用 FSx for Windows File Server 在 Amazon EC2 上設定 Microsoft SQL Server 容錯移轉叢集

Sweta Krishna 和 Ramesh Babu Donti，Amazon Web Services

總結

具有容錯移轉叢集執行個體 (FCI) 的 Microsoft SQL Server Standard 版可以為 SQL Server Enterprise 提供更具成本效益的替代方案。設定 SQL FCI 需要在節點之間共用檔案儲存，Amazon FSx for Windows File Server 提供全受管儲存，可自動跨可用區域同步複寫。Amazon FSx 透過對一般用途檔案共享使用內建的重複資料刪除來降低儲存成本，因此無需維護第三方解決方案。Amazon FSx 也支援下列項目：

- 只需為您使用的項目付費，無需預付費用或承諾。
- 使用 FSx 手動設定 FCI 做為共用儲存。
- 使用 FSx 做為 SQL 叢集的檔案共享見證。
- Amazon FSx for Windows File Server 支援持續可用的檔案共用的伺服器訊息區塊 (SMB) 3.0，因此適合 SQL Server FCI 部署。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 建立和管理 Amazon Virtual Private Cloud (Amazon VPC) 資源、Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、安全群組和 AWS Identity and Access Management (IAM) 角色的許可。
- AWS Managed Microsoft AD 或您自己的內部部署 Active Directory。
- 具備設定容錯移轉叢集必要許可的 Active Directory 網域使用者。
- SQL Server FCI 和 Microsoft Active Directory 連接埠的安全群組規則，用於安全混合連線。
- Active Directory for SQL Server 中的服務帳戶，已跨 SQL 節點設定適當的許可。
- 容錯移轉叢集中的 Amazon FSx for Windows File Server。
- SQL Server 安裝二進位檔。

限制

- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

產品版本

- Amazon EC2 for Windows Server 2012 R2 或更新版本
- 具有所有目前 Windows Server 版本的 Amazon FSx for Windows File Server
- Amazon FSx for NetApp ONTAP 作為共用儲存的替代方案
- SQL Server 2012/2016/2019/2022

Architecture

技術堆疊

- Amazon EC2
- Amazon FSx for Windows File Server
- Amazon VPC
- AWS Directory Service
- AWS Systems Manager
- IAM

目標架構

下圖顯示使用 Amazon FSx for Windows File Server 在 Amazon EC2 上 Microsoft SQL Server FCI 的高階架構。 FSx

網路基礎設施

- Amazon VPC 提供跨越三個可用區域的網路容器。
- 私有子網路在每個可用區域中提供隔離的子網路，以部署資源。

運算層

- Amazon EC2 包含 SQL Server 叢集節點 1，部署在可用區域 1 中，做為 Windows Server 容錯移轉叢集 (WSFC) 的一部分。

- Amazon EC2 包含 SQL Server 叢集節點 2，部署在可用區域 2 中做為 WSFC 的一部分。
- WSFC 叢集會連接兩個 SQL Server 節點以進行容錯移轉功能。

Amazon FSx for Windows File Server 的儲存層

異地同步備份 FSx 部署 （跨越可用區域 1 和 2)

- 可用區域 1 中的主要 FSx 檔案系統託管作用中的 SQL Server 資料和日誌檔案。
- 可用區域 2 中的次要 FSx 檔案系統提供自動容錯移轉功能。
- 共用 SMB 檔案共用 (\\fsx.domain\sqlshare)，可供 SQL Server 資料庫的兩個叢集節點存取。

單一可用區 FSx 部署 （在 AZ3 中）

- 可用區域 3 中的 Amazon FSx 檔案伺服器見證充當叢集仲裁見證。
- 檔案共享見證 (\\fsx.domain\witness) 會維護叢集仲裁，並防止大腦分割案例。

目錄服務

- AWS Managed Microsoft AD 提供叢集功能所需的 Windows 身分驗證和網域服務。

高可用性功能

- 多可用區域元件提供跨可用區域的容錯能力。
- 如果主要伺服器故障，FSx 待命檔案伺服器會提供自動容錯移轉。
- 檔案共享見證會在可用區域 3 中提供叢集仲裁管理，以協助確保在故障期間執行適當的叢集操作。
- 網域已與 整合 AWS Managed Microsoft AD ，以進行無縫 Windows 身分驗證。

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon FSx 提供檔案系統，可支援業界標準的連線通訊協定，並提供高可用性和跨的複寫功能 AWS 區域。

- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。
- AWS Directory Service for Microsoft Active Directory 可讓您的目錄感知工作負載 AWS 和資源在 中使用 Microsoft Active Directory AWS 雲端。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Systems Manager 可協助您管理在 中執行的應用程式和基礎設施 AWS 雲端。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。

最佳實務

- 將資料庫執行個體放在私有子網路中，以防止它們從網際網路公開存取，同時仍允許它們連線到 AWS 服務 並執行更新。
- 如需一般最佳實務，請參閱在 Amazon EC2 上部署 Microsoft SQL Server 的最佳實務。
- 若要使用 PowerShell 來管理 Amazon FSx for Windows File Server，請參閱管理 FSx for Windows 檔案系統。

史詩

建立和設定 SQL Server 的 Amazon EC2 節點

任務	Description	所需的技能
新增名稱和標籤。	1. 在 中 AWS 管理主控台，導覽至 Amazon EC2。 2. 選擇啟動執行個體。 3. 新增必要的標籤。	DBA
選擇 Windows AMI。	選擇符合 SQL Server 需求的 Amazon Machine Image (AMI) for Windows。	DBA
選取執行個體類型。	選取符合您需求的 Amazon EC2 執行個體類型。	DBA

任務	Description	所需的技能
使用金鑰對。	您可以使用金鑰對安全地連線至執行個體。在啟動執行個體之前，請確定您能夠存取選取的金鑰對。	DBA
設定網路設定。	1. 選擇您用來設定 Active Directory 的 VPC。 2. 選擇 VPC 和私有子網路。 3. 選擇現有的安全群組。如果您沒有安全群組，請選擇建立安全群組。	DBA
設定進階網路設定。	1. 選擇進階網路設定。 2. 選取次要 IP。 3. 選取自動指派核取方塊，以從子網路指派次要 IP。	DBA
設定儲存體。	設定所需的總儲存體，然後選擇所需的儲存體類型。	DBA
設定進階詳細資訊並啟動執行個體。	1. 從目錄清單中選擇 Active Directory 網域。 2. 選擇已連接 `AmazonSSMManagedInstanceCore` 和 `AmazonSSMDirectoryServiceAccess` 政策的 IAM 角色。 3. （選用）設定其他可用的選項。 4. 啟動執行個體	DBA
建立節點 2。	重複這些步驟來建立和設定節點 2。	DBA

在節點 1 和 2 上安裝和設定 Windows Server 容錯移轉叢集

任務	Description	所需的技能
登入節點 1。	以管理員身分登入 Windows Amazon EC2 執行個體。	DBA
在節點 1 上安裝 FCI 功能。	1. 在 PowerShell 中，執行下列指令碼： `Install-WindowsFeature -Name Failover-Clustering -IncludeManagementTools` 1. 安裝後，請檢查並安裝任何未完成的 Windows 更新。 2. 重新啟動執行個體。	DBA
登入節點 2。	以管理員身分登入 Windows Amazon EC2 執行個體。	DBA
在節點 2 上安裝 FCI 功能。	1. 在 PowerShell 中，執行下列指令碼： `Install-WindowsFeature -Name Failover-Clustering -IncludeManagementTools` 1. 安裝後，請檢查並安裝任何未完成的 Windows 更新。 2. 重新啟動執行個體。	DBA
將節點新增至叢集。	1. 使用遠端桌面連線至節點 1，然後選擇啟動伺服器管理員。	DBA

任務	Description	所需的技能
	2. 在工具功能表中，選擇容錯移轉叢集管理員。 3. 在容錯移轉叢集管理員窗格的管理下，選擇建立叢集以開啟建立叢集精靈。 4. 在 Before you begin (開始之前) 頁面上，選擇 Next (下一步)。 5. 如果顯示選取伺服器頁面，請在輸入名稱欄位中，選擇瀏覽並搜尋您的伺服器，然後選擇新增。針對您要新增的每個伺服器重複此步驟，然後選擇下一步。 6. 在用於管理叢集的存取點頁面上，選擇叢集，然後選擇下一步。 7. 確認頁面會顯示叢集名稱、節點和其他網域資訊。檢閱這些詳細資訊，然後選擇下一步。 8. 在摘要頁面上，確認已成功建立 FCI。如果有任何警告或錯誤，請檢視摘要輸出或選擇檢視報告。 9. 選擇完成。 10設定叢集時，系統會顯示您已完成建立叢集精靈。 11針對節點 2 重複這些步驟。	

任務	Description	所需的技能
讓叢集上線。	若要讓叢集上線，請更新兩個節點的靜態 IP 地址： 1. 選取叢集物件的名稱。 2. 開啟內容（按一下滑鼠右鍵）選單，然後選擇屬性。 3. 從 EC2 執行個體中選取子網路和未使用的次要 IP 地址。 4. 更新次要 IP 地址後，開啟內容（按一下滑鼠右鍵）選單，然後選取叢集物件名稱。 5. 選擇上線。	DBA
驗證叢集。	導覽至容錯移轉叢集管理員，並確認叢集核心資源在線上。	DBA

在節點 1 和 2 上安裝 SQL Server

任務	Description	所需的技能
登入 伺服器。	以管理員身分登入 Amazon EC2 執行個體。	DBA
掛載 SQL 二進位檔。	1. 啟動 SQL Server 安裝中心，然後選擇安裝。 2. 選擇新的 SQL Server 容錯移轉叢集安裝。 3. 依照提示操作以設定檔案。 4. 在全域規則對話方塊中，確認所有檢查都成功。 5. 在安裝容錯移轉叢集規則對話方塊中，確認所有檢查都	

任務	Description	所需的技能
	成功。在繼續之前解決任何警告。 6. 在功能選擇對話方塊中，選取資料庫引擎服務和用戶端工具連線的核取方塊。 7. 在特徵規則對話方塊中，驗證傳遞的所有規則。 8. 在執行個體組態對話方塊中，輸入 SQL Server 網路名稱。 9. 在叢集資源群組中，指定資源群組的名稱。從下拉式清單中選取群組，或輸入自訂名稱。 10.在叢集網路組態對話方塊中，選取 IPv4 並清除 DHCP。為每個節點提供次要私有 IP 地址。 11.在伺服器組態對話方塊中： • 在服務帳戶索引標籤上，提供 SQL Server 的 SQL Service 帳戶名稱和登入資料的必要詳細資訊。 • 確認代理程式和資料庫引擎的啟動類型設定為手動。 • 選取授予對 SQL Server 資料庫引擎服務執行磁碟區維護任務權限的核取方塊。這可啟用 SQL Server 的即時檔案初始化。 12.在資料庫引擎組態對話方塊中：	

任務	Description	所需的技能
	• 在 Windows 身分驗證模式下的伺服器組態索引標籤上，確認已選取身分驗證模式。 • 在資料目錄索引標籤上，指定資料、日誌、tempdb 和備份檔案 （例如檔案 `.mdf` 和 `.ldf` 檔案）的 `\\fsx.domain\sqlshare\logs` Amazon `\\fsx.domain\sqlshare\data` FSx 路徑位置。 13.在功能組態規則對話方塊中，確認所有檢查都成功，然後選擇下一步。 14.在準備安裝對話方塊中，驗證所有組態設定。 15.選擇安裝以繼續安裝。 16.在完成對話方塊中，選擇關閉。	

任務	Description	所需的技能
將節點 2 新增至容錯移轉叢集。	1. 導覽至 SQL Server 安裝中心。 2. 選擇安裝。 3. 選擇將節點新增至 SQL Server 容錯移轉叢集，然後遵循叢集節點組態的提示。 4. 在叢集節點組態對話方塊中，檢查 IPv4 並清除 DHCP。 5. 新增節點 2 的 IP 地址。 6. 在服務帳戶下，確認節點 1 和 2 的詳細資訊相符。 7. 提供對應 SQL Server 服務帳戶的登入資料。 8. 在功能規則對話方塊中，確認所有檢查都成功。 9. 選擇下一步，然後遵循其餘提示。 10選擇安裝以繼續安裝。 11.在完成對話方塊中，選擇關閉。	DBA

設定 Amazon FSx 檔案共享見證

任務	Description	所需的技能
設定規定人數設定。	1. 使用 RDP 連線至您的 Amazon EC2 執行個體。 2. 導覽至容錯移轉叢集管理員。	DBA

任務	Description	所需的技能
	3. 在叢集上開啟內容（按一下滑鼠右鍵）選單，然後選擇更多動作。 4. 選擇設定叢集規定人數設定。 5. 設定規定人數設定。	
擷取 DNS 詳細資訊。	在 Amazon FSx 主控台中，選擇 Managed AD，然後選擇連接。DNS 應該具有下列格式：`\\example.example.net\share`	DBA
設定檔案共享見證。	選擇 Amazon FSx 檔案共用路徑，然後選擇完成。	DBA

相關資源

AWS resources

- Amazon FSx for Windows File Server（影片）
- 深入了解 Amazon FSx for Windows File Server（影片）
- 如何在 Windows Server 上使用 Amazon EBS Multi-Attach 部署 SQL Server 容錯移轉叢集 (AWS 部落格文章）
- 使用 Amazon FSx for Windows File Server 簡化 Microsoft SQL Server 高可用性部署 (AWS 部落格文章）
- 使用 Amazon FSx for NetApp ONTAP 的 SQL Server 高可用性部署 (AWS 部落格文章）
- 搭配 Microsoft SQL Server 使用 FSx for Windows File Server
- 什麼是 FSx for Windows File Server？

其他資源

- 建立容錯移轉叢集

其他資訊

設定檔案共享見證

透過在允許傳入連線的 Amazon FSx 安全群組中新增規則，確保您從兩個節點連線到檔案系統。應允許 SMB 連接埠。例如，如果 DNS 名稱為 `\\example.example.com\share`，請使用 `\\example.example.com\share`。對 Always On 可用性叢集中的檔案共享見證使用相同的值。完成下列步驟以設定檔案共享見證：

1. 使用 RDP 連線到您的 Amazon EC2 執行個體。
2. 導覽至容錯移轉叢集管理員。
3. 開啟內容（按一下滑鼠右鍵）選單，然後選擇更多動作。
4. 選擇設定叢集規定人數設定。
5. 選擇下一步。
6. 選取配額組態並設定檔案共享見證。
7. 提供 DNS 名稱。
8. 檢閱摘要，然後選擇完成。檔案共享見證應該在線上的叢集核心資源區段中。

使用作用中待命資料庫為 Amazon RDS Custom 上的 Oracle 電子商務套件設定 HA/DR 架構

Simon Cunningham、Jaydeep Nandy 和 Nitin Saxena，Amazon Web Services

總結

此模式說明如何在 Amazon Relational Database Service (Amazon RDS) Custom 上架構 Oracle E-Business 解決方案，以便在另一個 Amazon Web Services (AWS) 可用區域中設定 Amazon RDS Custom 僅供讀取複本資料庫，並將其轉換為作用中待命資料庫，以獲得高可用性 (HA) 和災難復原 (DR)。Amazon RDS Custom 僅供讀取複本的建立是透過 AWS 管理主控台完全自動化。

此模式不會討論新增其他應用程式層和共用檔案系統的步驟，這也可能是 HA/DR 架構的一部分。如需這些主題的詳細資訊，請參閱下列 Oracle 支援備註：1375769.1、1375670.1 和 1383621.1（第 5 節，進階複製選項)。（存取需要 Oracle Support 帳戶。）

若要將 E-Business Suite 系統遷移至 Amazon Web Services (AWS) 上的單一層級單一可用區架構，請參閱將 Oracle E-Business Suite 遷移至 Amazon RDS Custom 模式。

Oracle E-Business Suite 是一種企業資源規劃 (ERP) 解決方案，可自動化整個企業的流程，例如財務、人力資源、供應鏈和製造。它具有三層架構：用戶端、應用程式和資料庫。先前，您必須在自我管理的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上執行 E-Business Suite 資料庫，但您現在可以受益於 Amazon RDS Custom。

先決條件和限制

先決條件

- Amazon RDS Custom 上現有的 E-Business Suite 安裝；請參閱將 Oracle E-Business Suite 遷移至 Amazon RDS Custom 的模式
- 如果您想要將僅供讀取複本變更為唯讀，並使用它將報告卸載至待命，則為 Oracle Active Data Guard 資料庫授權（請參閱 Oracle Technology 商業價目表)

限制

- Amazon RDS Custom 上 Oracle 資料庫的限制和不支援的組態
- 與 Amazon RDS Custom for Oracle 僅供讀取複本相關聯的限制

產品版本

對於 Amazon RDS Custom 支援的 Oracle 資料庫版本和執行個體類別，請參閱 Amazon RDS Custom for Oracle 的要求和限制。

Architecture

下圖說明 AWS 上 E-Business Suite 的代表性架構，其中包含作用中/被動設定中的多個可用區域和應用程式層。資料庫使用 Amazon RDS Custom 資料庫執行個體和 Amazon RDS Custom 僅供讀取複本。僅供讀取複本使用 Active Data Guard 複寫到另一個可用區域。您也可以使用僅供讀取複本卸載主要資料庫上的讀取流量，並用於報告目的。

如需詳細資訊，請參閱《Amazon RDS 文件》中的使用 Amazon RDS Custom for Oracle 的僅供讀取複本。

根據預設，Amazon RDS Custom 僅供讀取複本會建立為掛載。不過，如果您想要將部分唯讀工作負載卸載至待命資料庫，以減少主要資料庫的負載，您可以依照 Epics 區段中的步驟，手動將掛載複本的模式變更為唯讀。典型的使用案例是從待命資料庫執行您的報告。變更為唯讀需要作用中的待命資料庫授權。

當您在 AWS 上建立僅供讀取複本時，系統會在封面下使用 Oracle Data Guard 代理程式。 此組態會自動產生並在最高效能模式中設定，如下所示：

```
DGMGRL> show configuration
Configuration - rds_dg
  Protection Mode: MaxPerformance
  Members:
  vis_a - Primary database
    vis_b - Physical standby database
Fast-Start Failover: DISABLED
Configuration Status:
SUCCESS   (status updated 58 seconds ago)
```

工具

AWS 服務

- Amazon RDS Custom for Oracle 是一種受管資料庫服務，適用於需要存取基礎作業系統和資料庫環境的舊版、自訂和封裝應用程式。它可自動化資料庫管理任務和操作，同時讓身為資料庫管理員的您能夠存取和自訂資料庫環境和作業系統。

其他工具

- Oracle Data Guard 是一種工具，可協助您建立和管理 Oracle 待命資料庫。此模式使用 Oracle Data Guard 在 Amazon RDS Custom 上設定作用中待命資料庫。

史詩

建立僅供讀取複本

任務	Description	所需的技能
建立 Amazon RDS Custom 資料庫執行個體的僅供讀取複本。	若要建立僅供讀取複本，請遵循 Amazon RDS 文件中的指示，並使用您建立的 Amazon RDS Custom 資料庫執行個體（請參閱先決條件一節）做為來源資料庫。 根據預設，Amazon RDS Custom 僅供讀取複本會建立為實體待命，且處於掛載狀態。這是為了確保符合 Oracle Active Data Guard 授權。請依照下列步驟，將僅供讀取複本轉換為唯讀模式。	DBA

將僅供讀取複本變更為唯讀作用中待命

任務	Description	所需的技能
連線至 Amazon RDS Custom 僅供讀取複本。	使用以下命令將實體待命資料庫轉換為作用中待命資料庫。 ⚠ Important 這些命令需要 Oracle 作用中待命授權。若要	DBA

任務	Description	所需的技能

	取得授權，請聯絡您的 Oracle 代表。
	
	`$ sudo su - rdsdb`
	`-bash-4.2$ sql`
	`SQL> select process,s`
	`tatus,sequence# from v`
	`$managed_standby;`
	
	`PROCESS STATUS`
	`SEQUENCE#`
	`--------- ------------`
	`----------`
	`ARCH CLOSING`
	`3956`
	`ARCH CONNECTED`
	`0`
	`ARCH CLOSING`
	`3955`
	`ARCH CLOSING`
	`3957`
	`RFS IDLE`
	`0`
	`RFS IDLE`
	`3958`
	`MRP0 APPLYING_LOG`
	`3958`
	`SQL> select name,`
	`database_role,`
	`open_mode from v`
	`$database;`
	
	`NAME DATABASE_`
	`ROLE OPEN_MODE`
	`--------- ---------`
	`------- ---------`
	`----------`
	`VIS PHYSICAL`
	`STANDBY MOUNTED`	

任務	Description	所需的技能
	SQL> alter database recover managed standby database cancel; Database altered. Open the standby database SQL> alter database open; Database altered. SQL> select name, database_role, open_mode from v$database; NAME DATABASE_ROLE OPEN_MODE --------- ---------- ---------- VIS PHYSICAL STANDBY READ ONLY	
使用即時日誌啟動媒體復原。	若要啟用即時日誌套用功能，請使用下列命令。這些轉換和驗證待命（僅供讀取複本）為作用中待命資料庫，因此您可以連接和執行唯讀查詢。 SQL> alter database recover managed standby database using current logfile disconnect from session; Database altered	DBA

任務	Description	所需的技能

檢查資料庫狀態。	若要檢查資料庫的狀態，請使用下列命令。
	
	`SQL> select name, database_role, open_mode from v$database;`
	`NAME DATABASE_ROLE OPEN_MODE`
	`--------- ---------- ----------`
	`VIS PHYSICAL STANDBY READ ONLY WITH APPLY`
		DBA

任務	Description	所需的技能

檢查重做套用模式。	若要檢查重做套用模式，請使用下列命令。
	
	`SQL> select process,status,sequence# from v$managed_standby;`
	`PROCESS STATUS SEQUENCE#`
	`--------- ------------ ----------`
	`ARCH CLOSING 3956`
	`ARCH CONNECTED 0`
	`ARCH CLOSING 3955`
	`ARCH CLOSING 3957`
	`RFS IDLE 0`
	`RFS IDLE 3958`
	`MRP0 APPLYING_LOG 3958`
	
	`SQL> select open_mode from v$database;`
	`OPEN_MODE`
	`--------------------`
	`READ ONLY WITH APPLY`
		DBA

相關資源

- 將 Oracle 電子商務套件遷移至 Amazon RDS Custom (AWS 方案指引）
- 使用 Amazon RDS Custom (Amazon RDS 文件）
- 使用 Amazon RDS Custom for Oracle 的僅供讀取複本 (Amazon RDS 文件）
- Amazon RDS Custom for Oracle – 資料庫環境中的新控制功能 (AWS 新聞部落格）

- 在 AWS 上遷移 Oracle 電子商務套件 (AWS 白皮書)
- AWS 上的 Oracle E-Business Suite 架構 (AWS 白皮書)

在 上將資料從 IBM Db2、SAP、Sybase 和其他資料庫串流至 MongoDB Atlas AWS

Battulga Purevragchaa 和 Igor Alekseev，Amazon Web Services

Babu Srinivasan，MongoDB

摘要

此模式說明將資料從 IBM Db2 和其他資料庫遷移至 上的 MongoDB Atlas 的步驟，例如大型主機資料庫和 Sybase AWS 雲端。它使用 AWS Glue 來協助加速資料遷移至 MongoDB Atlas。

模式隨附於 上的遷移至 MongoDB Atlas AWS 指南，請參閱 AWS 方案指引網站。它提供該指南中討論的其中一個遷移案例的實作步驟。如需其他遷移案例，請參閱 AWS 規範指引網站上的下列模式：

- 在 上將自我託管的 MongoDB 環境遷移至 MongoDB Atlas AWS
- 在 上將關聯式資料庫遷移至 MongoDB Atlas AWS

模式 適用於 AWS Managed Services 合作夥伴和 AWS 使用者。

先決條件和限制

先決條件

- 要遷移至 MongoDB Atlas 的來源資料庫，例如 SAP、Sybase、IBM Db2 等。
- 熟悉 SAP、Sybase、IBM Db2、MongoDB Atlas 和 等資料庫 AWS 服務。

產品版本

- MongoDB 5.0 版或更新版本。

Architecture

下圖說明使用 AWS Glue Studio、Amazon Kinesis Data Streams 和 MongoDB Atlas 的批次資料載入和資料串流。

此參考架構使用 AWS Glue Studio 建立擷取、轉換和載入 (ETL) 管道，將資料遷移至 MongoDB Atlas。與 MongoDB Atlas AWS Glue 編目程式 整合，以促進資料控管。資料可以使用 Amazon Kinesis Data Streams 批次移植或串流至 MongoDB Atlas。

批次資料載入

如需批次資料遷移的詳細資訊，請參閱 AWS 部落格文章使用 編寫 MongoDB Atlas 的 ETL 任務 AWS Glue。

資料串流

如需支援不同使用案例的 MongoDB Atlas 參考架構，請參閱 AWS Prescriptive Guidance 網站上的遷移至 上的 MongoDB Atlas AWS。

工具

- AWS Glue 是全受管 ETL 服務。它可協助您可靠地分類、清理、擴充和移動資料存放區和資料串流之間的資料。
- Amazon Kinesis Data Streams 可協助您即時收集和處理大型資料記錄串流。
- MongoDB Atlas 是全受管資料庫即服務 (DbaaS)，用於在雲端中部署和管理 MongoDB 資料庫。

最佳實務

如需指導方針，請參閱 MongoDB GitHub 儲存庫中的 MongoDB 最佳實務指南。 MongoDB GitHub

史詩

探索和評估

任務	Description	所需的技能
決定叢集大小。	使用來自 的資訊估計工作集大小`db.stats()` ，以取得總索引空間。假設經常存取您資料空間的百分比。或者，您可以根據您的假設預估記憶體需求。此任務大約需要一週的時間。如需此範例和其他案例的詳細資訊和範例，請參閱相關資源區段中的連結。	MongoDB DBA，應用程式架構師

任務	Description	所需的技能
估計網路頻寬需求。	若要估算您的網路頻寬需求，請將平均文件大小乘以每秒提供的文件數量。請考慮叢集上任何節點所承擔的最大流量。若要計算從叢集到用戶端應用程式的下游資料傳輸率，請使用一段時間內傳回文件總數的總和。如果您的應用程式從次要節點讀取，請將總文件數除以可提供讀取操作的節點數。若要尋找資料庫的平均文件大小，請使用 `db.stats().avgObjSize` 命令。此任務通常需要一天的時間。	MongoDB DBA
選取 Atlas 層。	遵循 MongoDB 文件中的指示，選取正確的 Atlas 叢集層。	MongoDB DBA
規劃切換。	規劃應用程式切換。	MongoDB DBA，應用程式架構師

在 AWS 上設定新的 MongoDB Atlas 環境

任務	Description	所需的技能
在 上建立新的 MongoDB Atlas 叢集 AWS。	在 MongoDB Atlas 中，選擇建置叢集，然後選取 AWS 做為雲端供應商。	MongoDB DBA
選取 AWS 區域 和 全域叢集組態。	從 AWS 區域 Atlas 叢集可用的清單中選取 。視需要設定全域叢集。	MongoDB DBA

任務	Description	所需的技能
選取叢集層。	選取您偏好的叢集層。您的方案選擇會決定記憶體、儲存體和 IOPS 規格等因素。	MongoDB DBA
設定其他叢集設定。	設定其他叢集設定，例如 MongoDB 版本、備份和加密選項。如需這些選項的詳細資訊，請參閱相關資源一節。	MongoDB DBA

設定安全與合規

任務	Description	所需的技能
設定存取清單。	若要連線至 Atlas 叢集，您必須將項目新增至專案的存取清單。Atlas 使用 Transport Layer Security (TLS) / Secure Sockets Layer (SSL) 來加密資料庫虛擬私有雲端 (VPC) 的連線。若要設定專案的存取清單，以及有關此史詩中案例的詳細資訊，請參閱相關資源區段中的連結。	MongoDB DBA
驗證和授權使用者。	您必須建立和驗證將存取 MongoDB Atlas 叢集的資料庫使用者。若要存取專案中的叢集，使用者必須屬於該專案，而且可以屬於多個專案。 您也可以使用 AWS Identity and Access Management (IAM) 啟用授權。如需詳細資訊，請參閱 MongoDB 文件中的使用 IAM 設定身分驗證。	MongoDB DBA

任務	Description	所需的技能
建立自訂角色。	（選用）如果內建的 Atlas 資料庫使用者權限未涵蓋您所需的權限集，則 Atlas 支援建立自訂角色。	MongoDB DBA
設定 VPC 對等互連。	（選用）Atlas 支援 VPC 與其他 AWS VPC 對等互連。VPCs	MongoDB DBA
設定 AWS PrivateLink 端點。	（選用）您可以使用 AWS 在上設定私有端點AWS PrivateLink。	MongoDB DBA
啟用雙重驗證。	（選用）Atlas 支援雙重驗證 (2FA)，以協助使用者控制對其 Atlas 帳戶的存取。	MongoDB DBA
使用 LDAP 設定使用者身分驗證和授權。	（選用）Atlas 支援使用輕量型目錄存取通訊協定 (LDAP) 執行使用者身分驗證和授權。	MongoDB DBA
設定統一 AWS 存取。	（選用）某些 Atlas 功能，包括使用客戶金鑰管理的 Atlas Data Lake 和靜態加密，請使用 IAM 角色進行身分驗證。	MongoDB DBA
使用 設定靜態加密 AWS KMS。	（選用）Atlas 支援使用 AWS Key Management Service (AWS KMS) 加密儲存引擎和雲端供應商備份。	MongoDB DBA
設定 CSFLE。	（選用）Atlas 支援用戶端欄位層級加密 (CSFLE)，包括欄位的自動加密。	MongoDB DBA

遷移資料

任務	Description	所需的技能
在 MongoDB Atlas 中啟動您的目標複本集。	在 MongoDB Atlas 中啟動您的目標複本集。在 Atlas Live Migration Service 中，選擇我已準備好遷移。	MongoDB DBA
建立 AWS Glue 與 MongoDB Atlas 的連線。	使用 AWS Glue 編目程式 與 MongoDB Atlas （目標資料庫） AWS Glue 連線。此步驟有助於準備目標環境以進行遷移。如需詳細資訊，請參閱 AWS Glue 文件。	MongoDB DBA
建立 AWS Glue 與來源資料庫或來源串流的連線。	這有助於準備目標環境以進行遷移。	MongoDB DBA
設定資料轉換。	設定轉換邏輯，將資料從舊版結構化結構描述遷移至 MongoDB 的彈性結構描述。	MongoDB DBA
遷移資料。	排程 中的遷移 AWS Glue Studio。	MongoDB DBA

設定操作整合

任務	Description	所需的技能
連線至叢集。	連線至 MongoDB Atlas 叢集。	應用程式開發人員
與資料互動。	與叢集資料互動。	應用程式開發人員
監控叢集。	監控您的 MongoDB Atlas 叢集。	MongoDB DBA
備份和還原資料。	備份和還原叢集資料。	MongoDB DBA

疑難排解

問題	解決方案
如果您遇到問題	請參閱 MongoDB Atlas CloudFormation 資源儲存庫中的故障診斷。

相關資源

除非另有說明，否則下列所有連結都會移至 MongoDB 文件中的網頁。

遷移指南

- 在 上遷移至 MongoDB Atlas AWS (AWS 方案指引)

探索和評估

- 記憶體
- 使用 Atlas 範例資料集調整大小範例
- 行動應用程式的大小調整範例
- 網路流量
- 叢集自動擴展
- Atlas 大小調整範本

設定安全與合規

- 設定 IP 存取清單項目
- 設定資料庫使用者
- 設定對 Atlas UI 的存取
- 設定自訂資料庫角色
- 設定資料庫使用者
- 設定網路對等連線
- 了解 Atlas 中的私有端點
- 管理您的多重要素驗證選項

- 使用 LDAP 設定使用者身分驗證和授權
- Atlas Data Lake
- 使用客戶金鑰管理進行靜態加密
- 擔任角色的方法 (IAM 文件)
- 用戶端欄位層級加密
- 自動加密
- MongoDB Atlas 安全控制
- MongoDB 信任中心
- 設定叢集的安全功能

在 上設定新的 MongoDB Atlas 環境 AWS

- 雲端供應商和區域
- 管理全域叢集
- 選取叢集層
- 設定其他設定
- Atlas 入門
- 設定對 Atlas UI 的存取

遷移資料

- 遷移或匯入資料

監控叢集

- 監控您的叢集

整合 操作

- 連線至叢集
- 與您的資料互動
- 監控您的叢集
- 備份、還原和封存資料

GitHub 儲存庫

- 使用 將資料串流至 MongoDB Atlas AWS Glue

Amazon RDS Custom for Oracle 上 Oracle PeopleSoft 應用程式的轉換角色

sampath kathirvel，Amazon Web Services

總結

若要在 Amazon Web Services (AWS) 上執行 Oracle PeopleSoft 企業資源規劃 (ERP) 解決方案，您可以使用 Amazon Relational Database Service (Amazon RDS) 或 Amazon RDS Custom for Oracle，以支援需要存取基礎作業系統 (OS) 和資料庫環境的舊版、自訂和封裝應用程式。如需規劃遷移時要考慮的關鍵因素，請參閱 AWS 方案指引中的 Oracle 資料庫遷移策略。

此模式著重於為在 Amazon RDS Custom 上執行的 PeopleSoft 應用程式資料庫執行 Oracle Data Guard 切換或角色轉換的步驟，做為具有僅供讀取複本資料庫的主要資料庫。模式包含設定快速啟動容錯移轉 (FSFO) 的步驟。在此過程中，Oracle Data Guard 組態中的資料庫會繼續在其新角色中運作。Oracle Data Guard 轉換的典型使用案例包括災難復原 (DR) 演練、資料庫上的排程維護活動，以及待命優先修補程式套用滾動修補程式。如需詳細資訊，請參閱部落格文章 減少 Amazon RDS Custom 中的資料庫修補停機時間。

先決條件和限制

先決條件

- 使用僅供讀取複本模式完成在 Amazon RDS Custom 上將 HA 新增至 Oracle PeopleSoft。

限制

- RDS Custom for Oracle 的限制和不支援的組態
- 與 Amazon RDS Custom for Oracle 僅供讀取複本相關聯的限制

產品版本

- 如需 Amazon RDS Custom 支援的 Oracle 資料庫版本，請參閱 RDS Custom for Oracle。
- 如需 Amazon RDS Custom 支援的 Oracle 資料庫執行個體類別，請參閱 RDS Custom for Oracle 的資料庫執行個體類別支援。

Architecture

技術堆疊

- Amazon RDS Custom for Oracle

目標架構

下圖顯示 Amazon RDS Custom 資料庫執行個體和 Amazon RDS Custom 僅供讀取複本。Oracle Data Guard 在 DR 的容錯移轉期間提供角色轉換。

如需在 AWS 上使用 Oracle PeopleSoft 的代表性架構，請參閱在 AWS 上設定高度可用的 PeopleSoft 架構。

工具

AWS 服務

- Amazon RDS Custom for Oracle 是一種受管資料庫服務，適用於需要存取基礎作業系統和資料庫環境的舊版、自訂和封裝應用程式。
- AWS Secrets Manager 可協助您使用以程式設計方式擷取秘密的 API 呼叫取代程式碼中的硬式編碼登入資料，包括密碼。在此模式中，您會從秘密名稱RDS_DATAGUARD為 的 Secrets Manager 擷取資料庫使用者密碼`do-not-delete-rds-custom-+<<RDS Resource ID>>+-dg`。

其他服務

- Oracle Data Guard 可協助您建立、維護、管理和監控待命資料庫。此模式使用 Oracle Data Guard 最高效能來轉換角色 (Oracle Data Guard 切換)。

最佳實務

對於您的生產部署，我們建議您在第三個可用區域中啟動觀察器執行個體，與主要節點和僅供讀取複本節點分開。

史詩

啟動角色轉換

任務	Description	所需的技能
暫停主要和複本的資料庫自動化。	雖然 RDS Custom 自動化架構不會干擾角色轉換程序，但最	雲端管理員，DBA

任務	Description	所需的技能
	好在 Oracle Data Guard 切換期間暫停自動化。 若要暫停和繼續 RDS Custom 資料庫自動化，請遵循暫停和繼續 RDS Custom 自動化中的指示。	

任務	Description	所需的技能

檢查 Oracle Data Guard 狀態。	若要檢查 Oracle Data Guard 狀態，請登入主要資料庫。此模式包含使用多租戶容器資料庫 (CDB) 或非 CDB 執行個體的程式碼。
	
	非 CDB
	
	`-bash-4.2$ dgmgrl RDS_DATAGUARD@RDS_CUSTOM_ORCL_A`
	`DGMGRL for Linux: Release 19.0.0.0.0 - Production on Mon Nov 28 20:55:50 2022`
	`Version 19.10.0.0.0`
	`Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved.`
	`Welcome to DGMGRL, type "help" for information.`
	`Password:`
	`Connected to "ORCL_A"`
	`Connected as SYSDG.`
	`DGMGRL> show configuration`
	`Configuration - rds_dg`
	`Protection Mode: MaxAvailability`
	`Members:`
	`orcl_a - Primary database`
	`orcl_d - Physical standby database`
	`Fast-Start Failover: Disabled`
	`Configuration Status:`
		DBA

任務	Description	所需的技能
	SUCCESS (status updated 59 seconds ago) DGMGRL> CDB CDB-bash-4.2$ dgmgrl C##RDS_DATAGUARD@RDS_CUSTOM_RDSCDB_A DGMGRL for Linux: Release 19.0.0.0.0 - Production on Wed Jan 18 06:13:07 2023 Version 19.16.0.0.0 Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved. Welcome to DGMGRL, type "help" for information. Password: Connected to "RDSCDB_A" Connected as SYSDG. DGMGRL> show configuration Configuration - rds_dg Protection Mode: MaxAvailability Members: rdscdb_a - Primary database rdscdb_b - Physical standby database Fast-Start Failover: Disabled Configuration Status: SUCCESS (status updated 52 seconds ago)	

任務	Description	所需的技能
	`DGMGRL>`	
驗證執行個體角色。	開啟 AWS 管理主控台，然後導覽至 Amazon RDS 主控台。在資料庫的複寫區段的連線與安全索引標籤上，驗證主要和複本的執行個體角色。 主要角色應與 Oracle Data Guard 主要資料庫相符，而複本角色應與 Oracle Data Guard 實體待命資料庫相符。	雲端管理員，DBA

任務	Description	所需的技能

執行切換。	若要執行切換，DGMGRL請從主節點連線至 。
	
	非 CDB
	
	`DGMGRL> switchover to orcl_d;`
	`Performing switchover NOW, please wait...`
	`Operation requires a connection to database "orcl_d"`
	`Connecting ...`
	`Connected to "ORCL_D"`
	`Connected as SYSDG.`
	`New primary database "orcl_d" is opening...`
	`Operation requires start up of instance "ORCL" on database "orcl_a"`
	`Starting instance "ORCL"...`
	`Connected to an idle instance.`
	`ORACLE instance started.`
	`Connected to "ORCL_A"`
	`Database mounted.`
	`Database opened.`
	`Connected to "ORCL_A"`
	`Switchover succeeded, new primary is "orcl_d"`
	`DGMGRL>`
	
	CDB
	
	`DGMGRL> switchover to rdscdb_b`
	`Performing switchover NOW, please wait...`
		DBA

任務	Description	所需的技能
	New primary database "rdscdb_b" is opening... Operation requires start up of instance "RDSCDB" on database "rdscdb_a" Starting instance "RDSCDB"... Connected to an idle instance. ORACLE instance started. Connected to "RDSCDB_A" Database mounted. Database opened. Connected to "RDSCDB_A" Switchover succeeded, new primary is "rdscdb_b"	

任務	Description	所需的技能

驗證 Oracle Data Guard 連線。	切換後，驗證從主節點到的 Oracle Data Guard 連線DGMGRL。
	
	非 CDB
	
	`DGMGRL> show configura tion;`
	`Configuration - rds_dg`
	`Protection Mode:`
	`MaxAvailability`
	`Members:`
	`orcl_d - Primary`
	`database`
	`orcl_a - Physical`
	`standby database`
	`Fast-Start Failover:`
	`Disabled`
	`Configuration Status:`
	`SUCCESS (status updated`
	`60 seconds ago)`
	`DGMGRL>`
	
	`DGMGRL> show configura tion lag;`
	`Configuration - rds_dg`
	`Protection Mode:`
	`MaxAvailability`
	`Members:`
	`orcl_d - Primary`
	`database`
	`orcl_a - Physical`
	`standby database`
	`Transport Lag: 0`
	`seconds (computed 0`
	`seconds ago)`
	`Apply Lag: 0 seconds`
	`(computed 0 seconds`
	`ago)`
	`Fast-Start Failover:`
	`Disabled`
		DBA

任務	Description	所需的技能
	Configuration Status: SUCCESS (status updated 44 seconds ago) DGMGRL> CDB DGMGRL> show configura tion DGMGRL> show configura tion Configuration - rds_dg Protection Mode: MaxAvailability Members: rdscdb_b - Primary database rdscdb_a - Physical standby database Fast-Start Failover: Disabled Configuration Status: SUCCESS (status updated 52 seconds ago) DGMGRL> DGMGRL> show configura tion lag Configuration - rds_dg Protection Mode: MaxAvailability Members: rdscdb_b - Primary database rdscdb_a - Physical standby database Transport Lag: 0 seconds (computed 0 seconds ago)	

任務	Description	所需的技能
	Apply Lag: 0 seconds (computed 0 seconds ago) Fast-Start Failover: Disabled Configuration Status: SUCCESS (status updated 53 seconds ago) DGMGRL>	
在 Amazon RDS 主控台上驗證執行個體角色。	在您執行角色切換後，Amazon RDS 主控台會在資料庫連線與安全索引標籤的複寫區段下顯示新角色。複寫狀態可能需要幾分鐘的時間才能從空白更新為複寫。	DBA

設定 FSFO

任務	Description	所需的技能
重設切換。	將切換設回主節點。	DBA
安裝並啟動觀察者。	觀察者程序是DGMGRL用戶端元件，通常在與主要和待命資料庫不同的機器中執行。觀察者的 ORACLE HOME 安裝可以是 Oracle Client Administrator 安裝，也可以安裝 Oracle Database Enterprise Edition 或 Personal Edition。如需資料庫版本的觀察程式安裝詳細資訊，請參閱安裝和啟動觀察程式。若要設定觀察者程序的	DBA

任務	Description	所需的技能
	高可用性，建議您執行下列動作： • 為執行觀察程式的 EC2 執行個體啟用 EC2 執行個體自動復原。 EC2 在作業系統啟動過程中，您需要自動化觀察者啟動程序。 • 在 EC2 執行個體中部署觀察者，並設定大小為一 (1) 的 Amazon EC2 Auto Scaling 群組。發生 EC2 執行個體故障時，自動擴展群組會自動啟動另一個 EC2 執行個體。 對於 Oracle 12c 版本 2 和更新版本，您最多可以部署三個觀察者。一個觀察者是主要觀察者，其餘則是備份觀察者。當主要觀察者失敗時，其中一個備份觀察者會擔任主要角色。	

任務	Description	所需的技能

從觀察者主機連線至 DGMGRL。	觀察者主機設定了主要和待命資料庫連線`tnsnames.ora` 的項目。只要資料遺失在 FastStartFailoverLagLimit 組態（以秒為單位的值）內，您就可以啟用具有最大效能保護模式的 FSFO。不過，您必須使用最大可用性保護模式才能達到零資料遺失 (RPO=0)。
	
	非 CDB
	
	`DGMGRL> show configuration;`
	`Configuration - rds_dg`
	`Protection Mode: MaxAvailability`
	`Members:`
	`orcl_a - Primary database`
	`orcl_d - Physical standby database`
	`Fast-Start Failover: Disabled`
	`Configuration Status:`
	`SUCCESS (status updated 58 seconds ago)`
	`DGMGRL> show configuration lag`
	`Configuration - rds_dg`
	`Protection Mode: MaxAvailability`
	`Members:`
	`orcl_a - Primary database`
	`orcl_d - Physical standby database`
		DBA

任務	Description	所需的技能

```
Transport Lag: 0
 seconds (computed 1
 second ago)
Apply Lag: 0 seconds
 (computed 1 second ago)
Fast-Start Failover:
 Disabled
Configuration Status:
SUCCESS (status updated
 5 seconds ago)
DGMGRL>
```

CDB

```
-bash-4.2$ dgmgrl
 C##RDS_DATAGUARD@R
DS_CUSTOM_RDSCDB_A
DGMGRL for Linux:
 Release 19.0.0.0.0 -
 Production on Wed Jan
 18 06:55:09 2023
Version 19.16.0.0.0
Copyright (c) 1982,
 2019, Oracle and/or
 its affiliates.  All
 rights reserved.
Welcome to DGMGRL, type
 "help" for informati
on.
Password:
Connected to "RDSCDB_A
"
Connected as SYSDG.
DGMGRL> show configura
tion
Configuration - rds_dg
  Protection Mode:
 MaxAvailability
  Members:
  rdscdb_a - Primary
 database
```

任務	Description	所需的技能
	rdscdb_b - Physical standby database Fast-Start Failover: Disabled Configuration Status: SUCCESS (status updated 18 seconds ago) DGMGRL>	

任務	Description	所需的技能

將待命資料庫修改為容錯移轉目標。	從主節點或觀察者節點連線到一個待命資料庫。（雖然您的初始化可能具有 mulitiple 待命資料庫，但您目前只需要連線到一個。）
	
	非 CDB
	
	`DGMGRL> edit database orcl_a set property FastStartFailoverTarget='orcl_d';`
	`Property "faststartfailovertarget" updated`
	`DGMGRL> edit database orcl_d set property FastStartFailoverTarget='orcl_a';`
	`Property "faststartfailovertarget" updated`
	`DGMGRL> show database orcl_a FastStartFailoverTarget;`
	`FastStartFailoverTarget = 'orcl_d'`
	`DGMGRL> show database orcl_d FastStartFailoverTarget;`
	`FastStartFailoverTarget = 'orcl_a'`
	`DGMGRL>`
	
	CDB
	
	`DGMGRL> edit database orcl_a set property`
		DBA

任務	Description	所需的技能
	FastStartFailoverT arget='rdscdb_b'; Object "orcl_a" was not found DGMGRL> edit database rdscdb_a set property FastStartFailoverT arget='rdscdb_b'; Property "faststar tfailovertarget" updated DGMGRL> edit database rdscdb_b set property FastStartFailoverT arget='rdscdb_a'; Property "faststar tfailovertarget" updated DGMGRL> show database rdscdb_a FastStart FailoverTarget; FastStartFailoverT arget = 'rdscdb_b' DGMGRL> show database rdscdb_b FastStart FailoverTarget; FastStartFailoverT arget = 'rdscdb_a' DGMGRL>	

任務	Description	所需的技能

為 DGMGRL 的連線設定 FastStartFailoverThreshold。	在 Oracle 19c 中，預設值為 30 秒，最小值為 6 秒。較低的值可能會縮短容錯移轉期間的復原時間目標 (RTO)。較高的值有助於降低主要資料庫發生不必要的容錯移轉暫時性錯誤的機率。
	
	RDS Custom for Oracle 自動化架構會監控資料庫運作狀態，並每隔幾秒執行修正動作。因此，我們建議將 FastStartFailoverThreshold 設定為高於 10 秒的值。下列範例會將閾值設定為 35 秒。
	
	非 CBD 或 CDB
	
	`DGMGRL> edit configuration set property FastStartFailoverThreshold=35;`
	`Property "faststartfailoverthreshold" updated`
	`DGMGRL> show configuration FastStartFailoverThreshold;`
	`FastStartFailoverThreshold = '35'`
	`DGMGRL>`
		DBA

任務	Description	所需的技能

從主要節點或觀察者節點連線至 DGMGRL 以啟用 FSFO。	如果資料庫未啟用 Flashback 資料庫，則ORA-16827 會顯示警告訊息。如果 FastStart FailoverAutoReinstate 組態屬性設定為 TRUE（這是預設值），則選用的閃存資料庫有助於自動將失敗的主要資料庫恢復到容錯移轉之前的某個時間點。
	
	非 CDB
	
	`DGMGRL> enable fast_start failover;`
	`Warning: ORA-16827: Flashback Database is disabled`
	`Enabled in Zero Data Loss Mode.`
	`DGMGRL>`
	`DGMGRL> show configuration`
	`Configuration - rds_dg`
	`Protection Mode: MaxAvailability`
	`Members:`
	`orcl_a - Primary database`
	`Warning: ORA-16819: fast-start failover observer not started`
	`orcl_d - (*) Physical standby database`
	`Warning: ORA-16819: fast-start failover observer not started`
	`Fast-Start Failover: Enabled in Zero Data Loss Mode`
		DBA

任務	Description	所需的技能
	Configuration Status: WARNING (status updated 29 seconds ago) DGMGRL> CDB DGMGRL> enable fast_star t failover; Warning: ORA-16827: Flashback Database is disabled Enabled in Zero Data Loss Mode. DGMGRL> show configura tion; Configuration - rds_dg Protection Mode: MaxAvailability Members: rdscdb_a - Primary database Warning: ORA-16819 : fast-start failover observer not started rdscdb_b - (*) Physical standby database Fast-Start Failover: Enabled in Zero Data Loss Mode Configuration Status: WARNING (status updated 11 seconds ago) DGMGRL>	

任務	Description	所需的技能

啟動觀察者以進行 FSFO 監控，並驗證狀態。	您可以在啟用 FSFO 之前或之後啟動觀察者。如果已啟用 FSFO，觀察者會立即開始監控主要和目標待命資料庫的狀態和連線。如果未啟用 FSFO，在啟用 FSFO 之前，觀察者不會開始監控。
	
	當您啟動觀察者時，主要資料庫組態會顯示為沒有任何錯誤訊息，如上一個`show configuration` 命令所證明。
	
	非 CDB
	
	`DGMGRL> start observer;`
	`[W000 2022-12-0`
	`1T06:16:51.271+00:00]`
	` FSFO target standby is`
	` orcl_d`
	`Observer 'ip-10-0-`
	`1-89' started`
	`[W000 2022-12-0`
	`1T06:16:51.352+00:00]`
	` Observer trace level is`
	` set to USER`
	
	`DGMGRL> show configura`
	`tion`
	`Configuration - rds_dg`
	`Protection Mode:`
	` MaxAvailability`
	`Members:`
	`orcl_a - Primary`
	` database`
	`orcl_d - (*) Physical`
	` standby database`
		DBA

任務	Description	所需的技能
	Fast-Start Failover: Enabled in Zero Data Loss Mode Configuration Status: SUCCESS (status updated 56 seconds ago) DGMGRL> DGMGRL> show observer Configuration - rds_dg Primary: orcl_a Active Target: orcl_d Observer "ip-10-0- 1-89" - Master Host Name: ip-10-0-1 -89 Last Ping to Primary: 1 second ago Last Ping to Target: 1 second ago DGMGRL> CDB DGMGRL> start observer; Succeeded in opening the observer file "/home/oracle/fsfo _ip-10-0-1-56.dat". [W000 2023-01-1 8T07:31:32.589+00:00] FSFO target standby is rdscdb_b Observer 'ip-10-0- 1-56' started The observer log file is '/home/oracle/obse rver_ip-10-0-1-56. log'.	

任務	Description	所需的技能

	DGMGRL> show configura
	tion
	Configuration - rds_dg
	  Protection Mode:
	 MaxAvailability
	  Members:
	  rdscdb_a - Primary
	 database
	  rdscdb_b - (*)
	 Physical standby
	 database
	Fast-Start Failover:
	 Enabled in Zero Data
	 Loss Mode
	Configuration Status:
	SUCCESS (status
	 updated 12 seconds ago)
	DGMGRL>
	
	DGMGRL> show observer;
	Configuration - rds_dg
	  Primary:
	 rdscdb_a
	  Active Target:
	 rdscdb_b
	Observer "ip-10-0-
	1-56" - Master
	  Host Name:
	  ip-10-0-1-56
	  Last Ping to Primary:
	  1 second ago
	  Last Ping to Target:
	  2 seconds ago
	DGMGRL>	

任務	Description	所需的技能

驗證容錯移轉。	在此案例中，可以透過手動停止主要 EC2 執行個體來執行容錯移轉測試。在停止 EC2 執行個體之前，請使用 `tail`命令，根據您的組態監控觀察者日誌檔案。使用 DGMGRL以`orcl_d`使用者登入待命資料庫RDS_DATAGUARD ，並檢查 Oracle Data Guard 狀態。它應該顯示`orcl_d`是新的主要資料庫。
	
	Note
	在此容錯移轉測試案例中， `orcl_d`是非 CDB 資料庫。
	
	在容錯移轉之前，已在 上啟用快閃記憶體資料庫`orcl_a`。在前一個主要資料庫返回線上並開始處於 MOUNT 狀態後，觀察者會將其恢復為新的待命資料庫。恢復的資料庫可做為新主要資料庫的 FSFO 目標。您可以在觀察者日誌中驗證詳細資訊。
	
	`DGMGRL> show configuration`
	`Configuration - rds_dg`
	`Protection Mode: MaxAvailability`
	`Members:`
		DBA

任務	Description	所需的技能
	orcl_d - Primary database Warning: ORA-16824 : multiple warnings, including fast-star t failover-related warnings, detected for the database orcl_a - (*) Physical standby database (disabled) ORA-16661: the standby database needs to be reinstated Fast-Start Failover: Enabled in Zero Data Loss Mode Configuration Status: WARNING (status updated 25 seconds ago) DGMGRL> 以下顯示 中的範例輸出observer.log 。 $ tail -f /tmp/obse rver.log Unable to connect to database using rds_custom_orcl_a [W000 2023-01-1 8T07:50:32.589+00:00] Primary database cannot be reached. [W000 2023-01-1 8T07:50:32.589+00:00] Fast-Start Failover threshold has expired. [W000 2023-01-1 8T07:50:32.590+00:00]	

任務	Description	所需的技能
	Try to connect to the standby. [W000 2023-01-18T07:50:32.590+00:00] Making a last connection attempt to primary database before proceeding with Fast-Start Failover. [W000 2023-01-18T07:50:32.591+00:00] Check if the standby is ready for failover. [S002 2023-01-18T07:50:32.591+00:00] Fast-Start Failover started... 2023-01-18T07:50:32.591+00:00 Initiating Fast-Start Failover to database "orcl_d"... [S002 2023-01-18T07:50:32.592+00:00] Initiating Fast-start Failover. Performing failover NOW, please wait... Failover succeeded, new primary is "orcl_d" 2023-01-18T07:55:32.101+00:00 [S002 2023-01-18T07:55:32.591+00:00] Fast-Start Failover finished... [W000 2023-01-18T07:55:32.591+00:00] Failover succeeded. Restart pinging. [W000 2023-01-18T07:55:32.603+00:00]	

任務	Description	所需的技能
	Primary database has changed to orcl_d. [W000 2023-01-18T07:55:33.618+00:00] Try to connect to the primary. [W000 2023-01-18T07:55:33.622+00:00] Try to connect to the primary rds_custom_orcl_d. [W000 2023-01-18T07:55:33.634+00:00] The standby orcl_a needs to be reinstated [W000 2023-01-18T07:55:33.654+00:00] Try to connect to the new standby orcl_a. [W000 2023-01-18T07:55:33.654+00:00] Connection to the primary restored! [W000 2023-01-18T07:55:35.654+00:00] Disconnecting from database rds_custom_orcl_d. [W000 2023-01-18T07:55:57.701+00:00] Try to connect to the new standby orcl_a. ORA-12170: TNS:Connect timeout occurred	

設定 Oracle Peoplesoft 應用程式與資料庫之間的連線

任務	Description	所需的技能
在主要資料庫中建立和啟動服務。	您可以使用同時包含組態中主要和待命資料庫端點的 TNS 項目，以避免角色轉換期間的應用程式組態變更。您可以定義兩個角色型資料庫服務，以支援讀取/寫入和唯讀工作負載。在下列範例中，`orcl_rw`是主要資料庫上作用中的讀取/寫入服務。`orcl_ro`是唯讀服務，並且是已在唯讀模式中開啟的待命資料庫上作用中。 `SQL> select name,open _mode from v$database; NAME OPEN_MODE --------- --------- ----------- ORCL READ WRITE SQL> exec dbms_serv ice.create_service ('orcl_rw','orcl_r w'); PL/SQL procedure successfully completed . SQL> exec dbms_serv ice.create_service ('orcl_ro','orcl_r o'); PL/SQL procedure successfully completed . SQL> exec dbms_serv ice.start_service('orcl_rw');`	DBA

任務	Description	所需的技能
	PL/SQL procedure successfully completed . SQL>	
在待命資料庫中啟動服務。	若要在唯讀待命資料庫中啟動服務，請使用下列程式碼。 SQL> select name,open _mode from v$database; NAME OPEN_MODE --------- --------- ----------- ORCL READ ONLY WITH APPLY SQL> exec dbms_serv ice.start_service('orcl_ro'); PL/SQL procedure successfully completed . SQL>	DBA

任務	Description	所需的技能
重新啟動主要資料庫時，自動啟動服務。	若要在重新啟動時在主要資料庫中自動啟動服務，請使用下列程式碼。 `SQL> CREATE OR REPLACE TRIGGER TrgDgServices after startup on database DECLARE db_role VARCHAR(30); db_open_mode VARCHAR(30); BEGIN SELECT DATABASE_ROLE, OPEN_MODE INTO db_role, db_open_mode FROM V$DATABASE; IF db_role = 'PRIMARY' THEN DBMS_SERV 2 ICE.START_SERVICE('orcl_rw'); END IF; IF db_role = 'PHYSICAL STANDBY' AND db_open_mode LIKE 'READ ONLY%' THEN DBMS_SERVICE.START_SERVICE('orcl_ro'); END IF; END; / Trigger created. SQL>`	DBA

任務	Description	所需的技能
設定讀取/寫入和唯讀資料庫之間的連線。	您可以針對讀取/寫入和唯讀連線使用下列應用程式組態範例。 `ORCL_RW = (DESCRIPTION = (CONNECT_TIMEOUT= 120)(RETRY_COUNT=20)(RETRY_DELAY=3)(TRANSPORT_CONNECT_ TIMEOUT=3) (ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST=devpsftdb.******.us-west-2 .rds.amazonaws.com) (PORT=1521)) (ADDRESS = (PROTOCOL = TCP)(HOST=psftread .******.us-west-2. rds.amazonaws.com) (PORT=1521))) (CONNECT_DATA=(SERVIC E_NAME = orcl_rw))) ORCL_RO = (DESCRIPTION = (CONNECT_TIMEOUT= 120)(RETRY_COUNT=20)(RETRY_DELAY=3)(TRANSPORT_CONNECT_ TIMEOUT=3) (ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST=devpsftdb.******.us-west-2 .rds.amazonaws.com) (PORT=1521)) (ADDRESS = (PROTOCOL = TCP)(HOST=psftread`	DBA

任務	Description	所需的技能
	.******.us-west-2. rds.amazonaws.com) (PORT=1521))) (CONNECT_DATA=(SERVIC E_NAME = orcl_ro)))	

相關資源

- 使用 Amazon RDS Custom for Oracle 上的 Data Guard 啟用高可用性 (AWS 技術指南)
- 將 Amazon RDS 設定為 Oracle PeopleSoft 資料庫 (AWS 白皮書)
- Oracle Data Guard Broker 指南 (Oracle 參考文件)
- Data Guard 概念和管理 (Oracle 參考文件)
- Oracle Data Guard 特定 FAN 和 FCF 組態需求 (Oracle 參考文件)

將資料從 Amazon Redshift 叢集跨帳戶卸載至 Amazon S3

Andrew Kamel，Amazon Web Services

總結

當您測試應用程式時，在測試環境中擁有生產資料會很有幫助。使用生產資料可讓您更準確地評估正在開發的應用程式。

此模式會將生產環境中 Amazon Redshift 叢集的資料擷取到 Amazon Web Services () 上開發環境中的 Amazon Simple Storage Service (Amazon S3) 儲存貯體AWS。

模式會逐步完成 DEV 和 PROD 帳戶的設定，包括下列項目：

- 必要的資源
- AWS Identity and Access Management (IAM) 角色
- 子網路、安全群組和虛擬私有雲端 (VPC) 的網路調整，以支援 Amazon Redshift 連線
- 具有 Python 執行時間以測試架構的範例 AWS Lambda 函數

若要授予 Amazon Redshift 叢集的存取權，模式會使用 AWS Secrets Manager 來存放相關登入資料。優點是擁有直接連線至 Amazon Redshift 叢集所需的所有必要資訊，而不需要知道 Amazon Redshift 叢集所在的位置。此外，您可以監控秘密的使用。

儲存在 Secrets Manager 中的秘密包括 Amazon Redshift 叢集的主機、資料庫名稱、連接埠和相關登入資料。

如需使用此模式時安全性考量的相關資訊，請參閱最佳實務一節。

先決條件和限制

先決條件

- 在 PROD 帳戶中執行的 Amazon Redshift 叢集
- 在 DEV 帳戶中建立的 S3 儲存貯體
- DEV 和 PROD 帳戶之間的 VPC 對等互連，並相應地調整路由表
- 同時為對等 VPCs 啟用 DNS 主機名稱和 DNS 解析

限制

- 根據您要查詢的資料量，Lambda 函數可能會逾時。

  如果您的執行需要的時間超過 Lambda 逾時上限 (15 分鐘），請針對 Lambda 程式碼使用非同步方法。此模式的程式碼範例使用適用於 Python 的 psycopg2 程式庫，目前不支援非同步處理。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

Architecture

下圖顯示具有 DEV 和 PROD 帳戶的目標架構。

該圖顯示以下工作流程：

1. DEV 帳戶中的 Lambda 函數會擔任存取 PROD 帳戶中 Secrets Manager 中的 Amazon Redshift 登入資料所需的 IAM 角色。

   Lambda 函數接著會擷取 Amazon Redshift 叢集秘密。
2. DEV 帳戶中的 Lambda 函數會使用資訊，透過對等 VPCs 連線至 PROD 帳戶中的 Amazon Redshift 叢集。

   然後，Lambda 函數會傳送卸載命令來查詢 PROD 帳戶中的 Amazon Redshift 叢集。
3. PROD 帳戶中的 Amazon Redshift 叢集會擔任相關的 IAM 角色，以存取 DEV 帳戶中的 S3 儲存貯體。

   Amazon Redshift 叢集會將查詢的資料卸載至 DEV 帳戶中的 S3 儲存貯體。

從 Amazon Redshift 查詢資料

下圖顯示用來擷取 Amazon Redshift 登入資料並連線至 Amazon Redshift 叢集的角色。工作流程是由 Lambda 函數啟動。

該圖顯示以下工作流程：

1. DEV 帳戶中`CrossAccount-SM-Read-Role`的 會擔任 PROD 帳戶中`SM-Read-Role`的 。
2. `SM-Read-Role` 角色使用附加的政策從 Secrets Manager 擷取秘密。
3. 登入資料用於存取 Amazon Redshift 叢集。

將資料上傳至 Amazon S3

下圖顯示擷取資料並將其上傳至 Amazon S3 的跨帳戶讀寫程序。工作流程是由 Lambda 函數啟動。模式會鏈結 Amazon Redshift 中的 IAM 角色。來自 Amazon Redshift 叢集的卸載命令會擔任 `CrossAccount-S3-Write-Role`，然後擔任 `S3-Write-Role`。此角色鏈結可讓 Amazon Redshift 存取 Amazon S3。

工作流程包含下列步驟：

1. DEV 帳戶中`CrossAccount-SM-Read-Role`的 會擔任 PROD 帳戶中`SM-Read-Role`的 。
2. 會從 Secrets Manager `SM-Read-Role`擷取 Amazon Redshift 登入資料。
3. Lambda 函數會連線至 Amazon Redshift 叢集並傳送查詢。
4. Amazon Redshift 叢集會擔任 `CrossAccount-S3-Write-Role`。
5. `CrossAccount-S3-Write-Role` 假設 DEV 帳戶中`S3-Write-Role`的 。
6. 查詢結果會卸載至 DEV 帳戶中的 S3 儲存貯體。

工具

AWS 服務

- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- Amazon Redshift 是 AWS 雲端中的受管 PB 級資料倉儲服務。
- AWS Secrets Manager 可協助您將程式碼中的硬式編碼憑證 (包括密碼) 取代為 Secrets Manager 的 API 呼叫，以便透過程式設計方法來擷取機密。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

程式碼儲存庫

此模式的程式碼可在 GitHub unload-redshift-to-s3-python 儲存庫中使用。

最佳實務

安全免責聲明

實作此解決方案之前，請考慮下列重要的安全建議：

- 請記住，連接開發和生產帳戶可以增加範圍並降低整體安全狀態。我們建議僅暫時部署此解決方案，擷取所需的資料部分，然後立即銷毀已部署的資源。若要銷毀資源，您應該刪除 Lambda 函數、移除為此解決方案建立的任何 IAM 角色和政策，以及撤銷帳戶之間授予的任何網路存取權。
- 將任何資料從生產複製到開發環境之前，請先諮詢您的安全與合規團隊。通常不應以此方式複製個人身分識別資訊 (PII)、受保護醫療資訊 (PHI) 和其他機密或管制資料。僅複製公開可用的非機密資訊（例如，來自商店前端的公有股票資料）。考慮權杖化或匿名化資料，或產生合成測試資料，而不是盡可能使用生產資料。其中一個AWS 安全原則是讓人員遠離資料。換句話說，開發人員不應該在生產帳戶中執行操作。
- 限制對開發帳戶中 Lambda 函數的存取，因為它可以從生產環境中的 Amazon Redshift 叢集讀取資料。
- 若要避免中斷生產環境，請實作下列建議：
  - 使用單獨的專用開發帳戶進行測試和開發活動。
  - 實作嚴格的網路存取控制，並將帳戶之間的流量限制為僅必要。
  - 監控和稽核對生產環境和資料來源的存取。
  - 為所有涉及的資源和服務實作最低權限的存取控制。
  - 定期檢閱和輪換登入資料，例如 AWS Secrets Manager 秘密和 IAM 角色存取金鑰。
- 請參閱本文所用服務的下列安全文件：
  - AWS Lambda 安全性
  - Amazon Redshift 安全性
  - Amazon S3 安全性
  - AWS Secrets Manager 安全性
  - IAM 安全最佳實務

安全性是存取生產資料和資源時的首要考量。一律遵循最佳實務、實作最低權限的存取控制，並定期檢閱和更新您的安全措施。

史詩

從 Amazon Redshift 查詢資料

任務	Description	所需的技能
建立 Amazon Redshift 叢集的秘密。	若要建立 Amazon Redshift 叢集的秘密，請執行下列動作： 1. 在 PROD 帳戶中，登入 AWS 管理主控台，然後開啟位於 https：//https://console.aws.amazon.com/secretsmanager/ 的 Secrets Manager 主控台。 2. 選擇存放新的秘密。 3. 選取 Amazon Redshift 資料倉儲的登入資料。 4. 針對使用者名稱和密碼，輸入執行個體的值，並確認或選擇加密金鑰的值。 5. 選擇秘密將存取的 Amazon Redshift 資料倉儲。 6. 輸入 `Redshift-Creds-Secret` 做為秘密名稱。 7. 使用預設選項完成剩餘的建立步驟，然後選擇儲存。 8. 檢視您的秘密，並記下為識別秘密而產生的秘密 ARN 值。	DevOps 工程師
建立 角色以存取 Secrets Manager。	若要建立角色，請執行下列動作： 1. 在 PROD 帳戶中，開啟位於 https：//https://console.aws	DevOps 工程師

任務	Description	所需的技能
	.amazon.com/iam/ 的 IAM 主控台。 2. 選擇政策。 3. 選擇建立政策。 4. 選擇 JSON 索引標籤，然後輸入如下所示的 IAM 政策： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "secretsmanager:GetResourcePolicy", "secretsmanager:GetSecretValue", "secretsmanager:DescribeSecret", "secretsmanager:ListSecretVersionIds" ], "Resource": [ "<Redshift-Creds-Secret-ARN>" ] }, { "Effect": "Allow",`	

任務	Description	所需的技能

	<pre>"Action":
	"secretsmanager:ListSecrets",
	"Resource": "*"
	}
	]
	}</pre>
	`Redshift-Creds-Secret-ARN` 將取代為 Secrets Manager 秘密的 Amazon Resource Name (ARN)，其中包含 Amazon Redshift 叢集的資訊和憑證。	

將資料上傳至 Amazon S3

任務	Description	所需的技能

建立角色以存取 S3 儲存貯體。	若要建立存取 S3 儲存貯體的角色，請執行下列動作：
	1. 在 DEV 帳戶中，開啟 IAM 主控台。
	2. 選擇政策。
	3. 選擇建立政策。
	4. 選擇 JSON 索引標籤，然後輸入如下所示的 IAM 政策：
	<pre>{
	"Version": "2012-10-17",
	"Statement": [
	{</pre>
		DevOps 工程師

任務	Description	所需的技能

	"Sid": "kmsstmt"
	,
	"Effect":
	"Allow",
	"Action": [
	"kms:Decr
	ypt",
	"kms:Encr
	ypt",
	"kms:Gene
	rateDataKey"
	],
	"Resource": [
	"<kms-key-
	arn>"
	]
	},
	{
	"Sid":
	"s3stmt",
	"Effect":
	"Allow",
	"Action": [
	"s3:PutOb
	ject",
	"s3:Get*",
	"s3:List*"
	],
	"Resource": [
	"arn:aws:
	s3:::mybucket",
	"arn:aws:
	s3:::mybucket/*"
	]
	}
	]
	}
	mybucket 以您要存取的 S3 儲存貯體名稱取代 。此外，如果 S3 儲存貯體已加	

任務	Description	所需的技能
	密，請將 `kms-key-arn` 取代為用於加密 S3 儲存貯體之 AWS Key Management Service (AWS KMS) 金鑰的 ARN。否則，您不需要政策中的 AWS KMS 區段。 5. 選擇檢閱政策，輸入 `S3-Write-Policy` 做為政策名稱，然後選擇建立政策。 6. 在導覽窗格中，選擇 Roles (角色)。 7. 選擇 Create Role (建立角色)。 8. 針對信任的實體角色，選擇自訂信任政策。 9. 選擇下一步：許可，然後選取您建立的 S3-Write-Policy 政策。 10輸入 `S3-Write-Role` 做為角色名稱，然後選擇建立角色。	

任務	Description	所需的技能
建立 Amazon Redshift 角色。	若要建立 Amazon Redshift 角色，請執行下列動作： 1. 在 PROD 帳戶中，開啟 IAM 主控台。 2. 選擇政策。 3. 選擇建立政策。 4. 選擇 JSON 索引標籤，然後輸入如下所示的 IAM 政策： `{ "Version": "2012-10-17", "Statement": [ { "Sid": "CrossAccountPolicy", "Effect": "Allow", "Action": "sts:AssumeRole", "Resource": "S3-Write-Role-ARN" } ] }` `S3-Write-Role-ARN` 將 取代為 DEV 帳戶中 `S3-Write-Role` 的 ARN。 5. 選擇檢閱政策，輸入 `S3-Write-Role-Assume-Policy` 做為政策名稱，然後選擇建立政策。	DevOps 工程師

任務	Description	所需的技能
	6. 在導覽窗格中，選擇角色，然後選擇建立角色。 7. 選擇 AWS 服務做為信任的實體類型，然後選擇 Redshift、Redshift - 可自訂。 8. 選擇下一步：許可，然後選取您建立的 S3-Write-Role-Assume-Policy 政策。 9. 輸入 `CrossAccount-S3-Write-Role` 做為角色名稱，然後選擇建立角色。 10. 將 IAM 角色與您的 Amazon Redshift 叢集建立關聯。	

部署 Lambda 函數。

任務	Description	所需的技能
部署 Lambda 函數。	若要在對等 VPC 中部署 Lambda 函數，請執行下列動作： 1. 在 https：//https://console.aws.amazon.com/lambda/ 開啟 Lambda 主控台。 2. 選擇 函數。 3. 選擇 Create function (建立函數)。 4. 在 Basic information (基本資訊) 下，對於 Function	DevOps 工程師

任務	Description	所需的技能
	name (函數名稱)，為您的函數輸入名稱。 5. 針對執行期，選擇 Python 3.8。 6. 展開變更預設執行角色，然後執行下列動作： a. 選擇使用現有角色。 b. 針對現有角色，選取您先前建立的 CrossAccount-RM-Read-Role Lambda 角色。 7. 展開進階設定，並執行下列動作： a. 選取啟用 VPC 核取方塊。 b. 針對 VPC，選取 DEV 帳戶中的對等 VPC。 c. 針對子網路，選取私有子網路。 d. 針對 Security Groups (安全群組)，請選取預設安全群組。 8. 選擇 Create function (建立函數)。 9. 將 psycopg2 程式庫新增為 Lambda 函數的圖層。 **Note** 您可以從 psycopg2-lambda-layer 儲存庫使用已部署的 layer。請務必根據	

任務	Description	所需的技能
	您的 AWS 區域 和 Python 執行時間使用 URL。	

測試架構

任務	Description	所需的技能

匯入所需的資源。	若要匯入所需的資源，請執行下列命令：
	`import ast`
	`import boto3`
	`import psycopg2`
	`import base64`
	`from botocore.exceptions import ClientError`
		應用程式開發人員

執行 Lambda 處理常式函數。	Lambda 函數使用 AWS Security Token Service (AWS STS) 進行跨帳戶存取和臨時憑證管理。函數使用 AssumeRole API 操作暫時取得 IAM `sm_read_role` 角色的許可。
	若要執行 Lambda 函數，請使用下列範例程式碼：
	`def lambda_handler(event, context):`
	`  sts_client = boto3.client('sts')`
		應用程式開發人員

任務	Description	所需的技能

	`    # Secrets Manager Configurations`
	`    secret_name = "redshift_creds"`
	`    sm_region = "eu-west-1"`
	`    sm_read_role = "arn:aws:iam::PROD_ACCOUNT_NUMBER:role/SM-Read-Role"`
	
	`    # S3 Bucket Configurations`
	`    s3_bucket_path = "s3://mybucket/"`
	`    s3_bucket_region = "eu-west-1"`
	`    s3_write_role = "arn:aws:iam::DEV_ACCOUNT_NUMBER:role/S3-Write-Role"`
	
	`    # Redshift Configurations`
	`    sql_query = "select * from category"`
	`    redshift_db = "dev"`
	`    redshift_s3_write_role = "arn:aws:iam::PROD_ACCOUNT_NUMBER:role/CrossAccount-S3-Write-Role"`
	
	`    chained_s3_write_role = "%s,%s" % (redshift_s3_write_role, s3_write_role)`
	
	`    assumed_role_object = sts_client.assume_role(`	

任務	Description	所需的技能

	RoleArn=s
	m_read_role,
	RoleSessi
	onName="CrossAccou
	ntRoleAssumption",
	ExternalI
	d="YOUR_EXTERNAL_ID",
	)
	credentials =
	assumed_role_objec
	t['Credentials']
	
	secret_dict =
	ast.literal_eval(g
	et_secret(credenti
	als, secret_name,
	sm_region))
	execute_query(secr
	et_dict, sql_query
	, s3_bucket_path,
	chained_s3_write_r
	ole, s3_bucket_region,
	redshift_db)
	
	return {
	'statusCode':
	200
	}	

任務	Description	所需的技能

取得秘密。	若要取得 Amazon Redshift 秘密，請使用下列範例程式碼：
	
	`def get_secret(credentials, secret_name, sm_region):`
	`  # Create a Secrets Manager client`
	`  session = boto3.session.Session()`
	`  sm_client = session.client(`
	`  service_name='secretsmanager',`
	`  aws_access_key_id=credentials['AccessKeyId'],`
	`  aws_secret_access_key=credentials['SecretAccessKey'],`
	`  aws_session_token=credentials['SessionToken'],`
	`  region_name=sm_region`
	`  )`
	
	`  try:`
	`  get_secret_value_response = sm_client.get_secret_value(`
	`  SecretId=secret_name`
	`  )`
	`  except ClientError as e:`
	`  print(e)`
	`  raise e`
	`  else:`
		應用程式開發人員

任務	Description	所需的技能

	<pre> if 'SecretSt
	ring' in get_secre
	t_value_response:
	            return
	 get_secret_value_r
	esponse['SecretStr
	ing']
	        else:
	            return
	 base64.b64decode(g
	et_secret_value_re
	sponse['SecretBina
	ry'])</pre>	

任務	Description	所需的技能

執行卸載命令。	若要將資料卸載至 S3 儲存貯體，請使用下列範例程式碼。
	
	`def execute_query(secret_dict, sql_query, s3_bucket_path, chained_s3_write_role, s3_bucket_region, redshift_db):`
	`conn_string = "dbname='%s' port='%s' user='%s' password='%s' host='%s'" \`
	`% (redshift_db,`
	`secret_dict["port"],`
	`secret_dict["username"],`
	`secret_dict["password"],`
	`secret_dict["host"])`
	`con = psycopg2.connect(conn_string)`
	`unload_command = "UNLOAD ('{}') TO '{}' IAM_ROLE '{}' DELIMITER '|' REGION '{}';" \`
	`.format(sql_query,`
	`s3_bucket_path + str(datetime.datetime.now()) + ".csv",`
		應用程式開發人員

任務	Description	所需的技能

	``
	chained_s3_write_role,
	
	s3_bucket_region)
	
	# Opening a cursor and run query
	cur = con.cursor()
	cur.execute(unload_command)
	
	print(cur.fetchone())
	cur.close()
	con.close()
	``	

清除

任務	Description	所需的技能
刪除 Lambda 函數。	為了避免產生意外成本，請移除資源以及 DEV 和 PROD 帳戶之間的連線。 若要移除 Lambda 函數，請執行下列動作： 1. 在 https：//https://console.aws.amazon.com/lambda/ 開啟 AWS Lambda 主控台。 2. 尋找並選取您建立的 Lambda 函數。 3. 選擇動作，然後選擇刪除。 4. 確認刪除。	DevOps 工程師

任務	Description	所需的技能
移除 IAM 角色和政策。	從 DEV 和 PROD 帳戶移除 IAM 角色和政策。 在 DEV 帳戶中，執行下列動作： 1. 開啟 IAM 主控台。 2. 刪除下列角色： • `S3-Write-Role` • `CrossAccount-RM-Read-Role` (Lambda 角色) 3. 刪除相關聯的政策： • `S3-Write-Policy` • 擔任 PROD 帳戶角色的 CrossAccount 政策 在 PROD 帳戶中，執行下列動作： 1. 開啟 IAM 主控台。 2. 刪除下列角色： • `SM-Read-Role` • `CrossAccount-S3-Write-Role` 3. 刪除相關聯的政策： • 用於存取 Secrets Manager 的 CrossAccount 政策 • `S3-Write-Role-Assume-Policy`	DevOps 工程師

任務	Description	所需的技能
在 Secrets Manager 中刪除秘密。	若要刪除秘密，請執行下列動作： 1. 在 PROD 帳戶中，開啟 Secrets Manager 主控台。 2. 找到並選取名為 的秘密`Redshift-Creds-Secret` 。 3. 選擇 Actions (動作)，然後選擇 Delete secret (刪除秘密)。 4. 確認刪除。	DevOps 工程師
移除 VPC 對等互連和安全群組規則。	若要移除 VPC 對等互連和安全群組規則，請執行下列動作： 1. 在 PROD 帳戶中，開啟位於 https：//https://console.aws.amazon.com/ec2/ 的 Amazon EC2 主控台。 2. 導覽至安全群組。 3. 尋找 Amazon Redshift 叢集使用的安全群組。 4. 編輯傳入規則，並移除允許從 DEV 帳戶 Lambda VPC 連線的規則。 5. 導覽至 VPC 對等連線，然後刪除對等連線。	DevOps 工程師

任務	Description	所需的技能
從 S3 儲存貯體移除資料。	若要從 Amazon S3 移除資料，請執行下列動作： 1. 在 DEV 帳戶中，開啟位於 https：//https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。 2. 找出您用於資料儲存的儲存貯體。 3. 刪除儲存貯體中的物件，或在不再需要時刪除整個儲存貯體。	DevOps 工程師
清除 AWS KMS 金鑰。	如果您已為加密建立任何自訂 AWS KMS 金鑰，請執行下列動作： 1. 在 https：//https://console.aws.amazon.com/kms/ 開啟 AWS KMS 主控台。 2. 找到為此模式建立的任何金鑰。 3. 排程要刪除的金鑰。（刪除金鑰有強制性的等待期間）。	DevOps 工程師

任務	Description	所需的技能
檢閱和刪除 Amazon CloudWatch logs。	若要刪除 CloudWatch 日誌，請執行下列動作： 1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。 2. 檢查您的 Lambda 函數或 Amazon Redshift 叢集建立的任何日誌群組。 3. 如果不再需要這些日誌群組，請將其刪除。	DevOps 工程師

相關資源

- Amazon CloudWatch 文件
- IAM 文件
- Lambda 文件
- Amazon Redshift 文件
- Amazon S3 文件
- AWS Secrets Manager 文件
- AWS 安全原則

其他資訊

將資料從 Amazon Redshift 卸載至 Amazon S3 之後，您可以使用 Amazon Athena 進行分析。

當您需要存取大量資料時，Amazon Athena 是一種大數據查詢服務非常有用。您可以使用 Athena，而無需佈建伺服器或資料庫。Athena 支援複雜的查詢，您可以在不同的物件上執行它。

與大多數一樣 AWS 服務，使用 Athena 的主要好處是它在執行查詢時提供了很大的靈活性，而不會增加複雜性。當您使用 Athena 時，您可以在 Amazon S3 中查詢不同的資料類型，例如 CSV 和 JSON，而無需變更資料類型。您可以從各種來源查詢資料，包括外部 AWS。Athena 可降低複雜性，因為您不必管理伺服器。Athena 會在您執行查詢之前直接從 Amazon S3 讀取資料，而不會載入或變更資料。

依工作負載的資料庫遷移模式

主題

- IBM
- Microsoft
- N/A
- 開放原始碼
- Oracle
- SAP

IBM

- 使用 AWS DMS 將 Db2 資料庫從 Amazon EC2 遷移至 Aurora MySQL 相容
- 使用日誌運送將 LUW 的 Db2 遷移至 Amazon EC2，以減少中斷時間
- 將 LUW 的 Db2 遷移至具有高可用性災難復原的 Amazon EC2
- 使用 AWS DMS 和 AWS SCT 從 Amazon EC2 上的 IBM Db2 遷移至 Aurora PostgreSQL 相容 Amazon EC2
- 從 IBM WebSphere Application Server 遷移至 Amazon EC2 上的 Apache Tomcat
- 在 上將資料從 IBM Db2、SAP、Sybase 和其他資料庫串流至 MongoDB Atlas AWS

Microsoft

- 加速探索 Microsoft 工作負載並將其遷移至 AWS
- 使用連結的伺服器從 Amazon EC2 上的 Microsoft SQL Server 存取內部部署 Microsoft SQL Server 資料表
- 評估將 SQL Server 資料庫遷移至 AWS 上的 MongoDB Atlas 的查詢效能
- 使用 AWS Lambda 和 任務排程器，在 Amazon EC2 上執行的 SQL Server Express 版本中自動化資料庫任務
- 變更 Python 和 Perl 應用程式，以支援從 Microsoft SQL Server 到 Amazon Aurora PostgreSQL 相容版本的資料庫遷移
- 在 AWS 上 SQL Server 的 Always On 可用性群組中設定唯讀路由
- 使用 設定 Amazon RDS for Microsoft SQL Server 的 Windows 身分驗證 AWS Managed Microsoft AD
- 使用 Microsoft Excel 和 Python 為 AWS DMS 任務建立 AWS CloudFormation 範本
- 使用 Terraform 在 Amazon EC2 和 Amazon FSx 上部署 SQL Server 容錯移轉叢集執行個體
- 使用 AWS DMS 將 Microsoft SQL Server 資料庫匯出至 Amazon S3
- 使用 AWS DMS 將 Amazon RDS for SQL Server 資料表匯出至 S3 儲存貯體
- 從 SQL Server 遷移至 PostgreSQL 時，實作 PII 資料的 SHA1 雜湊
- 將 EC2 Windows 執行個體擷取並遷移至 AWS Managed Services 帳戶
- 使用 FSx for Windows File Server 在 Amazon EC2 上設定 Microsoft SQL Server 容錯移轉叢集
- 將訊息佇列從 Microsoft Azure Service Bus 遷移至 Amazon SQS
- 使用 AWS DMS 將 Microsoft SQL Server 資料庫從 Amazon EC2 遷移至 Amazon DocumentDB
- 使用 AWS DMS 和 AWS SCT 將 Microsoft SQL Server 資料庫遷移至 Aurora MySQL
- 將 .NET 應用程式從 Microsoft Azure App Service 遷移至 AWS Elastic Beanstalk
- 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon EC2
- 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon RDS for SQL Server
- 使用連結的伺服器將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon RDS for SQL Server
- 使用原生備份和還原方法，將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon RDS for SQL Server
- 使用 AWS DMS 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon Redshift
- 使用 AWS SCT 資料擷取代理程式將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon Redshift

- 將內部部署 Microsoft SQL Server 資料庫遷移至執行 Linux 的 Amazon EC2 上的 Microsoft SQL Server
- 使用 Rclone 將資料從 Microsoft Azure Blob 遷移至 Amazon S3
- 使用 appcmd.exe 將 IIS 託管應用程式遷移至 Amazon EC2
- 使用 遷移 Microsoft SQL Server Always On 可用性群組 AWS Application Migration Service
- 使用 Application Migration Service 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon EC2
- 在 上將關聯式資料庫遷移至 MongoDB Atlas AWS
- 使用分散式可用性群組將 SQL Server 遷移至 AWS
- 使用 ACM 將 Windows SSL 憑證遷移至 Application Load Balancer
- 在 AWS 雲端中重新託管內部部署工作負載：遷移檢查清單
- 解決將 Microsoft SQL Server 遷移至 AWS 雲端後的連線錯誤
- 使用內部部署 SMTP 伺服器和 Database Mail 傳送 Amazon RDS for SQL Server 資料庫執行個體的通知
- 使用 Terraform 設定資料庫遷移的 CI/CD 管道
- 使用 Amazon FSx 設定 SQL Server Always On FCI 的異地同步備份基礎設施

N/A

- 在重新託管遷移至 期間建立防火牆請求的核准程序 AWS
- 加密現有的 Amazon RDS for PostgreSQL 資料庫執行個體
- Amazon DynamoDB 資料表的預估儲存成本
- 使用 AWS DMS 和 Amazon Aurora 實作跨區域災難復原

開放原始碼

- 使用 Python 應用程式自動產生 Amazon DynamoDB 的 PynamoDB 模型和 CRUD 函數 DynamoDB
- 在 pgAdmin 中使用 SSH 通道進行連線
- 在 Aurora PostgreSQL 相容中建立應用程式使用者和角色
- 在 Amazon RDS 中啟用 PostgreSQL 資料庫執行個體的加密連線
- 使用 AWS SCT 和 AWS DMS 將 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL AWS CLI CloudFormation
- 使用原生工具將內部部署 MariaDB 資料庫遷移至 Amazon RDS for MariaDB
- 將內部部署 MySQL 資料庫遷移至 Amazon EC2
- 將內部部署 MySQL 資料庫遷移至 Amazon RDS for MySQL
- 將內部部署 MySQL 資料庫遷移至 Aurora MySQL
- 將內部部署 PostgreSQL 資料庫遷移至 Aurora PostgreSQL
- 將 Couchbase Server 資料庫遷移至 Amazon EC2
- 使用 Auto Scaling 從 IBM WebSphere Application Server 遷移至 Amazon EC2 上的 Apache Tomcat
- 使用 SharePlex 和 AWS DMS 從 Oracle 8i 或 9i 遷移至 Amazon RDS for Oracle
- 使用 pglogical 從 Amazon EC2 上的 PostgreSQL 遷移至 Amazon RDS for PostgreSQL Amazon EC2
- 使用 AWS App2Container 將內部部署 Java 應用程式遷移至 AWS
- 使用 Percona XtraBackup、Amazon EFS 和 Amazon S3 將內部部署 MySQL 資料庫遷移至 Aurora MySQL
- 將 Oracle 外部資料表遷移至 Amazon Aurora PostgreSQL 相容
- 將具有超過 100 個引數的 Oracle 函數和程序遷移至 PostgreSQL
- 將 Redis 工作負載遷移至 AWS 上的 Redis Enterprise Cloud
- 在沒有加密的情況下監控 Amazon Aurora 是否有執行個體
- 重新啟動 RHEL 來源伺服器後，在不停用 SELinux 的情況下自動重新啟動 AWS 複寫代理程式
- 使用 Lambda 和 Secrets Manager 來排程 Amazon RDS for PostgreSQL 和 Aurora PostgreSQL 的任務
- 使用 pg_transport 在兩個 Amazon RDS 資料庫執行個體之間傳輸 PostgreSQL 資料庫
- 將資料從 Amazon Redshift 叢集跨帳戶卸載至 Amazon S3

Oracle

- 使用僅供讀取複本將 HA 新增至 Amazon RDS Custom 上的 Oracle PeopleSoft
- 將 JSON Oracle 查詢轉換為 PostgreSQL 資料庫 SQL
- 將 Oracle 的 VARCHAR2(1) 資料類型轉換為 Amazon Aurora PostgreSQL 的布林資料類型
- 使用 PostgreSQL 相容 Aurora 全域資料庫模擬 Oracle DR
- 在 Amazon Aurora PostgreSQL 和 Amazon RDS for PostgreSQL 中模擬 Oracle PL/SQL 關聯陣列
- 使用 AWR 報告估計 Oracle 資料庫的 Amazon RDS 引擎大小
- 在 Aurora PostgreSQL 中處理動態 SQL 陳述式中的匿名區塊
- 使用 Oracle SQL Developer 和 AWS SCT，逐步從 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL
- 在 Aurora PostgreSQL 相容中使用檔案編碼將 BLOB 檔案載入 TEXT
- 使用 AWS DMS，以 SSL 模式將 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL
- 將 Amazon RDS for Oracle 資料庫遷移至另一個資料庫 AWS 帳戶 ，AWS 區域 並使用 AWS DMS 進行持續複寫
- 使用 Oracle Data Pump 將內部部署 Oracle 資料庫遷移至 Amazon EC2
- 使用 Logstash 將內部部署 Oracle 資料庫遷移至 Amazon OpenSearch Service
- 使用 AWS DMS 和 AWS SCT 將內部部署 Oracle 資料庫遷移至 Amazon RDS for MySQL
- 將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 透過資料庫連結使用直接 Oracle Data Pump Import，將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 使用 Oracle Data Pump 將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 使用 Oracle 旁觀者和 AWS DMS 將內部部署 Oracle 資料庫遷移至 Amazon RDS for PostgreSQL
- 將內部部署 Oracle 資料庫遷移至 Amazon EC2 上的 Oracle
- 使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫從 Amazon EC2 遷移至 Amazon RDS for MariaDB
- 使用 AWS DMS 將 Oracle 資料庫從 Amazon EC2 遷移至 Amazon RDS for Oracle
- 使用 AWS DMS 將 Oracle 資料庫遷移至 Amazon DynamoDB
- 使用 Oracle GoldenGate 平面檔案轉接器，將 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫遷移至 Amazon Redshift
- 使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫遷移至 Aurora PostgreSQL
- 使用 Oracle Data Pump 和 AWS DMS 將 Oracle JD Edwards EnterpriseOne 資料庫遷移至 AWS

- 使用 AWS DMS 將 Oracle 分割的資料表遷移至 PostgreSQL
- 使用 AWS DMS 將 Oracle PeopleSoft 資料庫遷移至 AWS
- 將資料從現場部署 Oracle 資料庫遷移至 Aurora PostgreSQL
- 從 Amazon RDS for Oracle 遷移至 Amazon RDS for MySQL
- 使用具體化視觀表和 AWS DMS，從 Oracle 8i 或 9i 遷移至 Amazon RDS for PostgreSQL
- 使用 SharePlex 和 AWS DMS 從 Oracle 8i 或 9i 遷移至 Amazon RDS for PostgreSQL
- 使用 Oracle GoldenGate 從 Oracle 資料庫遷移至 Amazon RDS for PostgreSQL
- 使用 AWS DMS 和 AWS SCT 從 Oracle on Amazon EC2 遷移至 Amazon RDS for MySQL
- 在 Amazon ECS 上從 Oracle WebLogic 遷移至 Apache Tomcat (TomEE)
- 將函數型索引從 Oracle 遷移至 PostgreSQL
- 將舊版應用程式從 Oracle Pro*C 遷移至 ECPG
- 將 Oracle CLOB 值遷移至 AWS 上的 PostgreSQL 中的個別資料列
- 將 Oracle 資料庫錯誤代碼遷移至與 Amazon Aurora PostgreSQL 相容的資料庫
- 使用延伸模組將 Oracle 原生函數遷移至 PostgreSQL
- 將 Oracle PeopleSoft 遷移至 Amazon RDS Custom
- 將 Oracle ROWID 功能遷移至 AWS 上的 PostgreSQL
- 將 Oracle SERIALLY_REUSABLE pragma 套件遷移至 PostgreSQL
- 將虛擬產生的資料欄從 Oracle 遷移至 PostgreSQL
- 使用 Amazon CloudWatch 監控 Oracle GoldenGate 日誌
- 分析從 Oracle 到 PostgreSQL 的部分資料庫遷移的物件相依性
- 在 Amazon RDS for Oracle 上將 Oracle Database Enterprise Edition 轉換為 Standard Edition 2
- 使用作用中待命資料庫為 Amazon RDS Custom 上的 Oracle 電子商務套件設定 HA/DR 架構
- 在 Aurora PostgreSQL 相容上設定 Oracle UTL_FILE 功能
- Amazon RDS Custom for Oracle 上 Oracle PeopleSoft 應用程式的轉換角色
- 從 Oracle 遷移到 Amazon Aurora PostgreSQL 後驗證資料庫物件

SAP

- 使用 Systems Manager 和 EventBridge 自動備份 SAP HANA 資料庫
- 使用 AWS DMS 從 SAP ASE 遷移至 Amazon RDS for SQL Server
- 使用 AWS SCT 和 AWS DMS 將 Amazon EC2 上的 SAP ASE 遷移至與 Amazon Aurora PostgreSQL 相容
- 使用具有相同主機名稱的 SAP HSR 將 SAP HANA 遷移至 AWS
- 在 AWS 上設定 SAP on IBM Db2 的災難復原

更多模式

- 從 Amazon EKS 容器存取 Amazon Neptune 資料庫
- 使用 Athena 存取、查詢和聯結 Amazon DynamoDB 資料表
- 允許 EC2 執行個體寫入存取 AMS 帳戶中的 S3 儲存貯體
- 使用 Amazon Athena 和 Amazon Quick Sight 分析並視覺化巢狀 JSON 資料
- 使用 AWS Batch 自動化 Amazon RDS for PostgreSQL 資料庫執行個體的備份
- 使用 DynamoDB TTL 自動將項目封存至 Amazon S3
- 自動修復未加密的 Amazon RDS 資料庫執行個體和叢集
- 使用 AWS Systems Manager 維護 Windows 自動停止和啟動 Amazon RDS 資料庫執行個體
- 建置包含 MongoDB Atlas 的 AWS 登陸區域
- 使用 AWS Mainframe Modernization 和 建置 COBOL Db2 程式 AWS CodeBuild
- 使用 Amazon DataZone 建置企業資料網格 AWS CDK，以及 AWS CloudFormation
- 變更 Python 和 Perl 應用程式，以支援從 Microsoft SQL Server 到 Amazon Aurora PostgreSQL 相容版本的資料庫遷移
- 使用 Python 將 EBCDIC 資料轉換為 AWS 上的 ASCII
- 將 Teradata NORMALIZE 暫時功能轉換為 Amazon Redshift SQL
- 將 Teradata RESET WHEN 功能轉換為 Amazon Redshift SQL
- 將 Oracle 的 VARCHAR2(1) 資料類型轉換為 Amazon Aurora PostgreSQL 的布林資料類型
- 在 Aurora PostgreSQL 相容中建立應用程式使用者和角色
- 使用 Microsoft Excel 和 Python 為 AWS DMS 任務建立 AWS CloudFormation 範本
- 使用 Kinesis Data Streams 和 Firehose 搭配 將 DynamoDB 記錄交付至 Amazon S3 AWS CDK
- 使用私有靜態 IPs 在 Amazon EC2 上部署 Cassandra 叢集，以避免重新平衡
- 使用 Terraform 在 Amazon EKS 中部署 CockroachDB 叢集
- 使用 RAG 和 ReAct 提示，開發進階生成式 AI 聊天式助理
- 使用 PostgreSQL 相容 Aurora 全域資料庫模擬 Oracle DR
- 在 IBM Db2 資料庫中啟用直接封存至 Amazon S3 的 DB2 日誌 Amazon S3 Db2
- 在 Amazon RDS for SQL Server 中啟用透明資料加密
- 使用 AWS DMS 將 Microsoft SQL Server 資料庫匯出至 Amazon S3
- 從 SQL Server 遷移至 PostgreSQL 時，實作 PII 資料的 SHA1 雜湊

- 使用 Oracle SQL Developer 和 AWS SCT，逐步從 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL
- 在 Aurora PostgreSQL 相容中使用檔案編碼將 BLOB 檔案載入 TEXT
- 使用 AWS Secrets Manager 管理登入資料
- 使用 AWS DMS 將 Db2 資料庫從 Amazon EC2 遷移至 Aurora MySQL 相容
- 使用 AWS DMS 將 Microsoft SQL Server 資料庫從 Amazon EC2 遷移至 Amazon DocumentDB
- 使用 AWS DMS 和 AWS SCT 將 Microsoft SQL Server 資料庫遷移至 Aurora MySQL
- 使用 AWS DMS，以 SSL 模式將 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL
- 使用 AWS SCT 和 AWS DMS 將 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL AWS CLI CloudFormation
- 將 Amazon RDS 資料庫執行個體遷移至另一個 VPC 或帳戶
- 將 Amazon RDS for Oracle 資料庫遷移至另一個資料庫 AWS 帳戶 ， AWS 區域 並使用 AWS DMS 進行持續複寫
- 將 Amazon Redshift 叢集遷移至中國的 AWS 區域
- 使用原生工具將內部部署 MariaDB 資料庫遷移至 Amazon RDS for MariaDB
- 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon EC2
- 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon RDS for SQL Server
- 使用連結的伺服器將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon RDS for SQL Server
- 使用原生備份和還原方法，將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon RDS for SQL Server
- 使用 AWS DMS 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon Redshift
- 使用 AWS SCT 資料擷取代理程式將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon Redshift
- 將內部部署 Microsoft SQL Server 資料庫遷移至執行 Linux 的 Amazon EC2 上的 Microsoft SQL Server
- 將內部部署 MySQL 資料庫遷移至 Amazon RDS for MySQL
- 將內部部署 MySQL 資料庫遷移至 Aurora MySQL
- 使用 Oracle Data Pump 將內部部署 Oracle 資料庫遷移至 Amazon EC2
- 使用 Logstash 將內部部署 Oracle 資料庫遷移至 Amazon OpenSearch Service
- 使用 AWS DMS 和 AWS SCT 將內部部署 Oracle 資料庫遷移至 Amazon RDS for MySQL
- 將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle

- 透過資料庫連結使用直接 Oracle Data Pump Import，將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 使用 Oracle Data Pump 將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 使用 Oracle 旁觀者和 AWS DMS 將內部部署 Oracle 資料庫遷移至 Amazon RDS for PostgreSQL
- 將內部部署 Oracle 資料庫遷移至 Amazon EC2 上的 Oracle
- 將內部部署 PostgreSQL 資料庫遷移至 Aurora PostgreSQL
- 將內部部署 ThoughtSpot Falcon 資料庫遷移至 Amazon Redshift
- 使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫從 Amazon EC2 遷移至 Amazon RDS for MariaDB
- 使用 AWS DMS 將 Oracle 資料庫從 Amazon EC2 遷移至 Amazon RDS for Oracle
- 使用 Oracle GoldenGate 平面檔案轉接器，將 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫遷移至 Amazon Redshift
- 使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫遷移至 Aurora PostgreSQL
- 使用 Oracle Data Pump 和 AWS DMS 將 Oracle JD Edwards EnterpriseOne 資料庫遷移至 AWS
- 使用 AWS DMS 將 Oracle 分割的資料表遷移至 PostgreSQL
- 使用 AWS DMS 將 Oracle PeopleSoft 資料庫遷移至 AWS
- 將 Couchbase Server 資料庫遷移至 Amazon EC2
- 將資料從現場部署 Oracle 資料庫遷移至 Aurora PostgreSQL
- AWS 雲端 使用 Starburst 將資料遷移至
- 使用日誌運送將 LUW 的 Db2 遷移至 Amazon EC2，以減少中斷時間
- 將 LUW 的 Db2 遷移至具有高可用性災難復原的 Amazon EC2
- 從 Amazon RDS for Oracle 遷移至 Amazon RDS for MySQL
- 從 Couchbase Server 遷移至 AWS 上的 Couchbase Capella
- 使用 AWS DMS 和 AWS SCT 從 Amazon EC2 上的 IBM Db2 遷移至 Aurora PostgreSQL 相容 Amazon EC2
- 使用具體化視觀表和 AWS DMS，從 Oracle 8i 或 9i 遷移至 Amazon RDS for PostgreSQL
- 使用 SharePlex 和 AWS DMS 從 Oracle 8i 或 9i 遷移至 Amazon RDS for PostgreSQL
- 使用 Oracle GoldenGate 從 Oracle 資料庫遷移至 Amazon RDS for PostgreSQL
- 使用 AWS DMS 和 AWS SCT 從 Oracle on Amazon EC2 遷移至 Amazon RDS for MySQL
- 使用 pglogical 從 Amazon EC2 上的 PostgreSQL 遷移至 Amazon RDS for PostgreSQL Amazon EC2
- 使用 AWS DMS 從 SAP ASE 遷移至 Amazon RDS for SQL Server

- 將函數型索引從 Oracle 遷移至 PostgreSQL
- 將舊版應用程式從 Oracle Pro*C 遷移至 ECPG
- 使用 Application Migration Service 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon EC2
- 將內部部署 Cloudera 工作負載遷移至 AWS 上的 Cloudera 資料平台
- 使用 Percona XtraBackup、Amazon EFS 和 Amazon S3 將內部部署 MySQL 資料庫遷移至 Aurora MySQL
- 從內部部署伺服器將 Oracle Business Intelligence 12c 遷移至 AWS 雲端
- 將 Oracle CLOB 值遷移至 AWS 上的 PostgreSQL 中的個別資料列
- 將 Oracle 資料庫錯誤代碼遷移至與 Amazon Aurora PostgreSQL 相容的資料庫
- 將 Oracle 外部資料表遷移至 Amazon Aurora PostgreSQL 相容
- 使用延伸模組將 Oracle 原生函數遷移至 PostgreSQL
- 將 Oracle PeopleSoft 遷移至 Amazon RDS Custom
- 將 Oracle ROWID 功能遷移至 AWS 上的 PostgreSQL
- 將 Oracle SERIALLY_REUSABLE pragma 套件遷移至 PostgreSQL
- 使用 AWS SCT 和 AWS DMS 將 Amazon EC2 上的 SAP ASE 遷移至與 Amazon Aurora PostgreSQL 相容
- 將虛擬產生的資料欄從 Oracle 遷移至 PostgreSQL
- 設定最低可行的資料空間以在組織之間共用資料
- 監控 Amazon ElastiCache 叢集的靜態加密
- 使用 Amazon Athena 查詢具有 SQL 的 Amazon DynamoDB 資料表
- 使用信任的內容來保護和簡化 AWS 上 Db2 聯合資料庫中的使用者存取
- 在 AWS 上設定高度可用的 PeopleSoft 架構
- 在 Aurora PostgreSQL 相容上設定 Oracle UTL_FILE 功能
- 使用 PGO 在 Amazon EKS 上簡化 PostgreSQL 部署
- 以 CSV 檔案將大規模 Db2 z/OS 資料傳輸至 Amazon S3
- 使用 pg_transport 在兩個 Amazon RDS 資料庫執行個體之間傳輸 PostgreSQL 資料庫
- 從 Oracle 遷移到 Amazon Aurora PostgreSQL 後驗證資料庫物件

儲存和備份

主題

- 允許 EC2 執行個體寫入存取 AMS 帳戶中的 S3 儲存貯體
- 使用 Snowflake Snowpipe、Amazon S3、Amazon SNS 和 Amazon Data Firehose 將資料串流擷取自動化至 Snowflake 資料庫
- 自動加密現有和新的 Amazon EBS 磁碟區
- 在 上的 Stromasys Charon-SSP 模擬器中備份 Sun SPARC 伺服器 AWS 雲端
- 使用 Veeam Backup & Replication 備份資料並存檔至 Amazon S3
- 使用 將資料從 Amazon S3 儲存貯體複製到另一個帳戶和區域 AWS CLI
- 在 IBM Db2 資料庫中啟用直接封存至 Amazon S3 的 DB2 日誌 Amazon S3 Db2
- 使用 DistCp 搭配適用於 Amazon S3 的 AWS PrivateLink，將資料從內部部署 Hadoop 環境遷移至 Amazon S3
- 更多模式

允許 EC2 執行個體寫入存取 AMS 帳戶中的 S3 儲存貯體

Mansi Suratwala，Amazon Web Services

總結

AWS Managed Services (AMS) 可協助您更有效率且安全地操作 AWS 基礎設施。AMS 帳戶具有用於標準化 AWS 資源管理的安全防護機制。一種護欄是預設的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體描述檔不允許對 Amazon Simple Storage Service (Amazon S3) 儲存貯體進行寫入存取。不過，您的組織可能有多個 S3 儲存貯體，並且需要對 EC2 執行個體的存取進行更多控制。例如，您可能想要將來自 EC2 執行個體的資料庫備份存放在 S3 儲存貯體中。

此模式說明如何使用變更請求 (RFCs) 來允許 EC2 執行個體寫入您 AMS 帳戶中 S3 儲存貯體的存取權。RFC 是您或 AMS 建立的請求，可在受管環境中進行變更，並包含特定操作的變更類型 (CT) ID。

先決條件和限制

先決條件

- AMS 進階帳戶。如需詳細資訊，請參閱 AMS 文件中的 AMS 操作計劃。
- 存取 AWS Identity and Access Management (IAM) `customer-mc-user-role`角色以提交 RFCs。
- AWS Command Line Interface (AWS CLI)，已安裝並使用 AMS 帳戶中的 EC2 執行個體進行設定。
- 了解如何在 AMS 中建立和提交 RFCs。如需詳細資訊，請參閱 AMS 文件中的什麼是 AMS 變更類型？。
- 了解手動和自動變更類型 (CTs)。如需詳細資訊，請參閱 AMS 文件中的自動化和手動 CTs。

Architecture

技術堆疊

- AMS
- AWS CLI
- Amazon EC2
- Amazon S3
- IAM

工具

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Managed Services (AMS) 可協助您更有效率且安全地操作 AWS 基礎設施。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。

史詩

使用 RFC 建立 S3 儲存貯體

任務	Description	所需的技能
使用自動化 RFC 建立 S3 儲存貯體。	1. 登入您的 AMS 帳戶，選擇選擇變更類型頁面，選擇 RFCs，然後選擇建立 RFC。 2. 提交建立 S3 儲存貯體自動化 RFC。 **Note** 請務必記錄 S3 儲存貯體的名稱。	AWS 系統管理員、AWS 開發人員

建立 IAM 執行個體描述檔並將其與 EC2 執行個體建立關聯

任務	Description	所需的技能
提交手動 RFC 以建立 IAM 角色。	加入 AMS 帳戶時，`customer-mc-ec2-instance-profile` 會建立名為 的預設 IAM 執行個體描述檔，並與 AMS 帳戶中的每個 EC2 執行個體建立關聯。不過，執行個體描述檔沒有寫入 S3 儲存貯體的許可。 若要新增寫入許可，請提交建立 IAM 資源手動 RFC，以建立具有下列三個政策的 IAM 角色：`customer_ec2_instance_` 、 `customer_deny_policy` 和 `customer_ec2_s3_integration_policy` 。 **Important** `customer_ec2_instance_` 和 `customer_deny_policy` 政策已存在於您的 AMS 帳戶中。不過，您需要使用下列範例政策`customer_ec2_s3_integration_policy` 來建立：	AWS 系統管理員、AWS 開發人員

任務	Description	所需的技能

	{
	  "Version": "2012-10-17",
	  "Statement": [
	  {
	  "Sid": "",
	  "Effect": "Allow",
	  "Principal": {
	  "Service": "ec2.amazonaws.com"
	  },
	  "Action": "sts:AssumeRole"
	  }
	  ]
	}
	
	Role Permissions:
	{
	  "Version": "2012-10-17",
	  "Statement": [
	  {
	  "Action": [
	
	  "s3:ListBucket",
	
	  "s3:GetBucketLocation"
	  ],
	  "Resource": "arn:aws:s3:::",
	  "Effect": "Allow"
	  },
	  {
	  "Action": [
	
	  "s3:GetObject",	

任務	Description	所需的技能

	<pre>"s3:PutObject",
	
	"s3:ListMultipartUploadParts",
	
	"s3:AbortMultipartUpload"
	],
	"Resource": "arn:aws:s3:::/*",
	"Effect": "Allow"
	}
	]
	}</pre>	

提交手動 RFC 以取代 IAM 執行個體描述檔。	提交手動 RFC，將目標 EC2 執行個體與新的 IAM 執行個體設定檔建立關聯。	AWS 系統管理員、AWS 開發人員
測試 S3 儲存貯體的複製操作。	在 中執行下列命令，以測試 S3 儲存貯體的複製操作 AWS CLI： <pre>aws s3 cp test.txt s3://<S3 bucket>/test2.txt</pre>	AWS 系統管理員、AWS 開發人員

相關資源

- 為您的 Amazon EC2 執行個體建立 IAM 執行個體描述檔
- 建立 S3 儲存貯體 （ 使用 Amazon S3 主控台、 AWS SDKs或 AWS CLI)

使用 Snowflake Snowpipe、Amazon S3、Amazon SNS 和 Amazon Data Firehose 將資料串流擷取自動化至 Snowflake 資料庫

Bikash Chandra Rout，Amazon Web Services

總結

此模式說明如何在 Amazon Web Services (AWS) 雲端上使用 服務來處理持續的資料串流，並將其載入 Snowflake 資料庫。模式使用 Amazon Data Firehose 將資料交付至 Amazon Simple Storage Service (Amazon S3)、Amazon Simple Notification Service (Amazon SNS) 在收到新資料時傳送通知，以及 Snowflake Snowpipe 將資料載入 Snowflake 資料庫。

透過遵循此模式，您可以在幾秒鐘內持續產生可用於分析的資料、避免多個手動COPY命令，以及完全支援載入時的半結構化資料。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 持續將資料傳送至 Firehose 交付串流的資料來源。
- 從 Firehose 交付串流接收資料的現有 S3 儲存貯體。
- 作用中的 Snowflake 帳戶。

限制

- Snowflake Snowpipe 不會直接連線至 Firehose。

Architecture

技術堆疊

- Amazon Data Firehose
- Amazon SNS
- Amazon S3

- Snowflake Snowpipe
- Snowflake 資料庫

工具

- Amazon Data Firehose 是一項全受管服務，可將即時串流資料交付至目的地，例如 Amazon S3、Amazon Redshift、Amazon OpenSearch Service、Splunk，以及受支援的第三方服務供應商擁有的任何自訂 HTTP 端點或 HTTP 端點。
- Amazon Simple Storage Service (Amazon S3) 是網際網路的儲存體。
- Amazon Simple Notification Service (Amazon SNS) 會協調和管理消息傳遞或傳送到訂閱端點或客戶端。
- Snowflake – Snowflake 是以Software-as-a-Service(SaaS) 提供的分析資料倉儲。
- Snowflake Snowpipe – Snowpipe 會在 Snowflake 階段提供檔案時立即從檔案載入資料。

史詩

設定 Snowflake Snowpipe

任務	Description	所需的技能
在 Snowflake 中建立 CSV 檔案。	登入 Snowflake 並執行 `CREATE FILE FORMAT`命令，以使用指定的欄位分隔符號建立 CSV 檔案。如需此命令和其他 Snowflake 命令的詳細資訊，請參閱其他資訊一節。	開發人員
建立外部 Snowflake 階段。	執行 `CREATE STAGE`命令來建立參考您先前建立之 CSV 檔案的外部 Snowflake 階段。重要：您將需要 S3 儲存貯體的 URL、 AWS 存取金鑰和 AWS 私密存取金鑰。執行 `SHOW STAGES`命令以確認已建立 Snowflake 階段。	開發人員

任務	Description	所需的技能
建立 Snowflake 目標資料表。	執行 `CREATE TABLE`命令來建立 Snowflake 資料表。	開發人員
建立管道。	執行 `CREATE PIPE`命令；確定 `auto_ingest=true` 位於 命令中。執行 `SHOW PIPES`命令以確認已建立管道。複製並儲存`notification_channel` 資料欄值。此值將用於設定 Amazon S3 事件通知。	開發人員

設定 S3 儲存貯體

任務	Description	所需的技能
建立 S3 儲存貯體的 30 天生命週期政策。	登入 AWS 管理主控台 並開啟 Amazon S3 主控台。選擇包含來自 Firehose 資料的 S3 儲存貯體。然後選擇 S3 儲存貯體中的管理索引標籤，然後選擇新增生命週期規則。在生命週期規則對話方塊中輸入規則的名稱，並為儲存貯體設定 30 天的生命週期規則。如需此案例和其他案例的協助，請參閱相關資源一節。	系統管理員、開發人員
為 S3 儲存貯體建立 IAM 政策。	開啟 AWS Identity and Access Management (IAM) 主控台，然後選擇政策。選擇 Create policy (建立政策)，然後選擇 JSON 標籤。將政策從其他資訊區段複製並貼到 JSON 欄	系統管理員、開發人員

任務	Description	所需的技能
	位。此政策將授予 PutObject 和 DeleteObject 許可，以及 GetObjectVersion 、GetObject 和 ListBucket 許可。選擇檢閱政策，輸入政策名稱，然後選擇建立政策。	
將政策指派給 IAM 角色。	開啟 IAM 主控台，選擇角色，然後選擇建立角色。選擇另一個 AWS 帳戶做為信任的實體。輸入您的 AWS 帳戶 ID，然後選擇需要外部 ID。輸入您稍後要變更的預留位置 ID。選擇下一步，並指派您先前建立的 IAM 政策。然後建立 IAM 角色。	系統管理員、開發人員
複製 IAM 角色的 Amazon Resource Name (ARN)。	開啟 IAM 主控台，然後選擇角色。選擇您先前建立的 IAM 角色，然後複製並存放角色 ARN。	系統管理員、開發人員

在 Snowflake 中設定儲存整合

任務	Description	所需的技能
在 Snowflake 中建立儲存整合。	登入 Snowflake 並執行 CREATE STORAGE INTEGRATION 命令。這將修改信任關係、授予對 Snowflake 的存取權，並提供 Snowflake 階段的外部 ID。	系統管理員、開發人員

任務	Description	所需的技能
擷取 Snowflake 帳戶的 IAM 角色。	執行 `DESC INTEGRATION` 命令來擷取 IAM 角色的 ARN。 **Important** `<integration_name>` 是您先前建立的 Snowflake 儲存整合的名稱。	系統管理員、開發人員
記錄兩個資料欄值。	複製並儲存 `storage_aws_iam_user_arn` 和 `storage_aws_external_id` 資料欄的值。	系統管理員、開發人員

允許 Snowflake Snowpipe 存取 S3 儲存貯體

任務	Description	所需的技能
修改 IAM 角色政策。	開啟 IAM 主控台，然後選擇角色。選擇您先前建立的 IAM 角色，然後選擇信任關係索引標籤。選擇編輯信任關係。`snowflake_external_id` 將 取代為您先前複製`storage_aws_external_id` 的值。`snowflake_user_arn` 將 取代為您先前複製`storage_aws_iam_user_arn` 的值。然後選擇更新信任政策。	系統管理員、開發人員

開啟並設定 S3 儲存貯體的 SNS 通知

任務	Description	所需的技能
開啟 S3 儲存貯體的事件通知。	開啟 Amazon S3 主控台並選擇您的儲存貯體。選擇屬性，然後在進階設定 下，選擇事件。選擇新增通知，然後輸入此事件的名稱。如果您未輸入名稱，則會使用全域唯一識別碼 (GUID)。	系統管理員、開發人員
設定 S3 儲存貯體的 Amazon SNS 通知。	在事件下，選擇 ObjectCreate（全部），然後在傳送至下拉式清單中選擇 SQS 佇列。在 SNS 清單中，選擇新增 SQS 佇列 ARN，然後貼上您先前複製`notification_channel` 的值。然後選擇 Save (儲存)。	系統管理員、開發人員
訂閱 Snowflake SQS 佇列至 SNS 主題。	訂閱 Snowflake SQS 佇列至您建立的 SNS 主題。如需此步驟的說明，請參閱相關資源一節。	系統管理員、開發人員

檢查 Snowflake 階段整合

任務	Description	所需的技能
檢查並測試 Snowpipe。	登入 Snowflake 並開啟 Snowflake 階段。將檔案放入 S3 儲存貯體，並檢查 Snowflake 資料表是否載入它們。當新物件出現在 SAmazon	系統管理員、開發人員

任務	Description	所需的技能
	S33 會將 SNS 通知傳送至 Snowpipe。	

相關資源

- 管理儲存生命週期
- 訂閱 Snowflake SQS 佇列至 Amazon SNS 主題

其他資訊

建立檔案格式：

```
CREATE FILE FORMAT <name>
TYPE = 'CSV'
FIELD_DELIMITER = '|'
SKIP_HEADER = 1;
```

建立外部階段：

```
externalStageParams (for Amazon S3) ::=
  URL = 's3://[//]'

  [ { STORAGE_INTEGRATION =  } | { CREDENTIALS = ( {  { AWS_KEY_ID = `` AWS_SECRET_KEY
 = `` [ AWS_TOKEN = `` ] } | AWS_ROLE = ``  } ) ) }` ]
  [ ENCRYPTION = ( [ TYPE = 'AWS_CSE' ] [ MASTER_KEY = '' ] |
                   [ TYPE = 'AWS_SSE_S3' ] |
                   [ TYPE = 'AWS_SSE_KMS' [ KMS_KEY_ID = '' ] |
                   [ TYPE = NONE ] )
```

建立資料表：

```
CREATE [ OR REPLACE ] [ { [ LOCAL | GLOBAL ] TEMP[ORARY] | VOLATILE } | TRANSIENT ]
 TABLE [ IF NOT EXISTS ]
  <table_name>
    ( <col_name> <col_type> [ { DEFAULT <expr>
                              | { AUTOINCREMENT | IDENTITY } [ ( <start_num> ,
 <step_num> ) | START <num> INCREMENT <num> ] } ]
```

```
                                  /* AUTOINCREMENT / IDENTITY supported only for numeric
 data types (NUMBER, INT, etc.) */
                               [ inlineConstraint ]
     [ , <col_name> <col_type> ... ]
     [ , outoflineConstraint ]
     [ , ... ] )
  [ CLUSTER BY ( <expr> [ , <expr> , ... ] ) ]
  [ STAGE_FILE_FORMAT = ( { FORMAT_NAME = '<file_format_name>'
                           | TYPE = { CSV | JSON | AVRO | ORC | PARQUET | XML }
 [ formatTypeOptions ] } ) ]
  [ STAGE_COPY_OPTIONS = ( copyOptions ) ]
  [ DATA_RETENTION_TIME_IN_DAYS = <num> ]
  [ COPY GRANTS ]
  [ COMMENT = '<string_literal>' ]
```

顯示階段：

```
SHOW STAGES;
```

建立管道：

```
CREATE [ OR REPLACE ] PIPE [ IF NOT EXISTS ]
  [ AUTO_INGEST = [ TRUE | FALSE ] ]
  [ AWS_SNS_TOPIC =  ]
  [ INTEGRATION = '' ]
  [ COMMENT = '' ]
  AS
```

顯示管道：

```
SHOW PIPES [ LIKE '<pattern>' ]
           [ IN { ACCOUNT | [ DATABASE ] <db_name> | [ SCHEMA ] <schema_name> } ]
```

建立儲存整合：

```
CREATE STORAGE INTEGRATION <integration_name>
  TYPE = EXTERNAL_STAGE
  STORAGE_PROVIDER = S3
  ENABLED = TRUE
  STORAGE_AWS_ROLE_ARN = '<iam_role>'
  STORAGE_ALLOWED_LOCATIONS = ('s3://<bucket>/<path>/', 's3://<bucket>/<path>/')
```

```
  [ STORAGE_BLOCKED_LOCATIONS = ('s3://<bucket>/<path>/', 's3://<bucket>/<path>/') ]
```

範例：

```
create storage integration s3_int
  type = external_stage
  storage_provider = s3
  enabled = true
  storage_aws_role_arn = 'arn:aws:iam::001234567890:role/myrole'
  storage_allowed_locations = ('s3://amzn-s3-demo-bucket1/mypath1/', 's3://amzn-s3-
demo-bucket2/mypath2/')
  storage_blocked_locations = ('s3://amzn-s3-demo-bucket1/mypath1/sensitivedata/',
 's3://amzn-s3-demo-bucket2/mypath2/sensitivedata/');
```

如需此步驟的詳細資訊，請參閱設定 Snowflake 儲存整合以從 Snowflake 文件存取 Amazon S3。

描述整合：

```
DESC INTEGRATION <integration_name>;
```

S3 儲存貯體政策：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "s3:PutObject",
              "s3:GetObject",
              "s3:GetObjectVersion",
              "s3:DeleteObject",
              "s3:DeleteObjectVersion"
            ],
            "Resource": "arn:aws:s3:::/*"
        },
        {
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::",
            "Condition": {
                "StringLike": {
```

```
                    "s3:prefix": [
                        "/*"
                    ]
                }
            }
        }
    ]
}
```

自動加密現有和新的 Amazon EBS 磁碟區

Tony DeMarco 和 Josh Joy，Amazon Web Services

總結

Amazon Elastic Block Store (Amazon EBS) 磁碟區的加密對於組織的資料保護策略至關重要。這是建立架構良好的環境的重要步驟。雖然無法直接加密現有的未加密 EBS 磁碟區或快照，但您可以透過建立新的磁碟區或快照來加密它們。如需詳細資訊，請參閱 Amazon EC2 文件中的加密 EBS 資源。Amazon EC2 此模式提供預防性和偵測性控制，用於加密新的和現有的 EBS 磁碟區。在此模式中，您可以設定帳戶設定、建立自動修復程序，以及實作存取控制。

先決條件和限制

先決條件

- 作用中的 Amazon Web Services (AWS) 帳戶
- 在 macOS、Linux 或 Windows 上安裝和設定 AWS Command Line Interface (AWS CLI) macOS
- 在 macOS、Linux 或 Windows 上安裝和設定 jq
- AWS Identity and Access Management (IAM) 許可佈建為具有 AWS CloudFormation、Amazon Elastic Compute Cloud (Amazon EC2)、AWS Systems Manager、AWS Config 和 AWS Key Management Service (AWS KMS) 的讀取和寫入存取權
- AWS Organizations 設定已啟用所有功能，這是服務控制政策的需求
- 目標帳戶中已啟用 AWS Config

限制

- 在您的目標 AWS 帳戶中，不得有名為加密磁碟區的 AWS Config 規則。此解決方案會部署具有此名稱的規則。具有此名稱的預先存在規則可能會導致部署失敗，並導致與多次處理相同規則相關的不必要的費用。
- 此解決方案會使用相同的 AWS KMS 金鑰加密所有 EBS 磁碟區。
- 如果您為帳戶啟用 EBS 磁碟區加密，則此設定為區域特定。如果您為 AWS 區域啟用它，則無法為該區域中的個別磁碟區或快照停用它。如需詳細資訊，請參閱 Amazon EC2 文件中的預設加密。
- 當您修復現有的未加密 EBS 磁碟區時，請確保 EC2 執行個體未使用。此自動化會關閉執行個體，以分離未加密的磁碟區並連接加密的磁碟區。修復正在進行時會有停機時間。如果這是您組織的關鍵基礎設施，請確定有手動或自動高可用性組態，以免影響執行個體上執行的任何應用程式的可用性。建議您只在標準維護時段內修復關鍵資源。

Architecture

自動化工作流程

1. AWS Config 偵測到未加密的 EBS 磁碟區。
2. 管理員使用 AWS Config 將修復命令傳送至 Systems Manager。
3. Systems Manager 自動化會拍攝未加密 EBS 磁碟區的快照。
4. Systems Manager 自動化使用 AWS KMS 來建立快照的加密複本。
5. Systems Manager 自動化會執行下列動作：
   a. 如果受影響的 EC2 執行個體正在執行，則停止該執行個體
   b. 將磁碟區的新加密複本連接至 EC2 執行個體
   c. 將 EC2 執行個體傳回其原始狀態

工具

AWS 服務

- AWS CLI – AWS 命令列界面 (AWS CLI) 可讓您直接存取 AWS 服務的公有應用程式程式設計界面 (APIs)。您可以使用 AWS CLI 探索服務的功能，並開發 shell 指令碼來管理您的 資源。除了低階 API 同等命令之外，數個 AWS 服務還提供 AWS CLI 的自訂功能。自訂功能可能包括較高階的命令，可簡化具有複雜 API 的服務使用。
- AWS CloudFormation – AWS CloudFormation 是一項服務，可協助您建立 AWS 資源的模型和設定。您可以建立範本，描述您想要的所有 AWS 資源 （ 例如 Amazon EC2 執行個體 ），而 CloudFormation 會為您佈建和設定這些資源。
- AWS Config – AWS Config 提供 AWS 帳戶中 AWS 資源組態的詳細檢視。這包含資源彼此之間的關係和之前的組態方式，所以您可以看到一段時間中組態和關係的變化。
- Amazon EC2 – Amazon Elastic Compute Cloud (Amazon EC2) 是一種 Web 服務，可提供可調整大小的運算容量，讓您用來建置和託管軟體系統。
- AWS KMS – AWS Key Management Service (AWS KMS) 是一種針對雲端擴展的加密和金鑰管理服務。其他 AWS 服務會使用 AWS KMS 金鑰和功能，您可以使用它們來保護 AWS 環境中的資料。
- AWS Organizations – AWS Organizations 是一種帳戶管理服務，可讓您將多個 AWS 帳戶合併到您建立並集中管理的組織。
- AWS Systems Manager Automation – Systems Manager Automation 可簡化 Amazon EC2 執行個體和其他 AWS 資源的常見維護和部署任務。

其他服務

- jq – jq 是輕量且靈活的命令列 JSON 處理器。您可以使用此工具從 AWS CLI 輸出擷取金鑰資訊。

Code

- 此模式的程式碼可在 GitHub 中使用客戶 KMS 金鑰儲存庫自動修復未加密的 EBS 磁碟區。

史詩

自動化未加密磁碟區的修復

任務	Description	所需的技能
下載指令碼和 CloudFormation 範本。	從 GitHub 下載 shell 指令碼、JSON 檔案和 CloudFormation 範本 使用客戶 KMS 金鑰儲存庫自動修復未加密的 EBS 磁碟區。	AWS 管理員，一般 AWS
識別 AWS KMS 金鑰的管理員。	1. 登入 AWS 管理主控台，然後前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。 2. 識別將成為 AWS KMS 金鑰管理員的使用者或角色。如果需要為此目的建立新的使用者或角色，請立即建立。如需詳細資訊，請參閱 IAM 文件中的 IAM 身分。此自動化會建立新的 AWS KMS 金鑰。 3. 識別之後，請複製使用者或角色的 Amazon Resource Name (ARN)。如需詳細資訊，請參閱 IAM 文件中的	AWS 管理員，一般 AWS

任務	Description	所需的技能
	IAM ARNs。您可以在下一個步驟中使用此 ARN。	
部署 Stack1 CloudFormation 範本。	1. 開啟位在 AWS CloudFormationcloudformation/ 的 https://console.aws.amazon.com/ 主控台。 2. 在 CloudFormation 中，部署 `Stack1.yaml` 範本。請注意下列部署詳細資訊： • 為堆疊提供清晰且描述性的名稱。請記下堆疊名稱，因為您在下一個步驟中需要此值。 • 將金鑰管理員的 ARN 貼到 Stack1 中唯一的參數欄位。此使用者或角色會成為堆疊所建立 AWS KMS 金鑰的管理員。 如需部署 CloudFormation 範本的詳細資訊，請參閱 CloudFormation 文件中的使用 AWS CloudFormation CloudFormation 範本。	AWS 管理員，一般 AWS

任務	Description	所需的技能
部署 Stack2 CloudFormation 範本。	在 CloudFormation 中，部署 `Stack2.yaml` 範本。請注意下列部署詳細資訊： • 為堆疊提供清晰且描述性的名稱。 • 對於 Stack2 的唯一參數，輸入您在上一個步驟中建立的堆疊名稱。這可讓 Stack2 參考堆疊在上一個步驟中部署的新 AWS KMS 金鑰和角色。	AWS 管理員，一般 AWS
建立未加密的磁碟區進行測試。	使用未加密的 EBS 磁碟區建立 EC2 執行個體。如需說明，請參閱《Amazon EC2 文件》中的建立 Amazon EBS 磁碟區。Amazon EC2 執行個體類型並不重要，而且不需要存取執行個體。您可以建立 t2.micro 執行個體以保留在免費方案中，而且不需要建立金鑰對。	AWS 管理員，一般 AWS

任務	Description	所需的技能
測試 AWS Config 規則。	1. 開啟位於 https：//https://console.aws.amazon.com/config/ 的 AWS Config 主控台。在規則頁面上，選擇加密磁碟區規則。 2. 確認您的新未加密測試執行個體出現在不合規資源清單中。如果磁碟區未立即顯示，請等待幾分鐘並重新整理結果。AWS Config 規則會在建立執行個體和磁碟區後立即偵測資源變更。 3. 選取資源，然後選擇修復。 您可以在 Systems Manager 中檢視修復進度和狀態，如下所示： 1. 開啟位於 https：//https://console.aws.amazon.com/systems-manager/ 的 AWS Systems Manager 主控台。 2. 在導覽窗格中，選擇 Automation (自動化)。 3. 選擇執行 ID 連結以檢視步驟和狀態。	AWS 管理員，一般 AWS
設定其他帳戶或 AWS 區域。	根據您的使用案例，為任何其他帳戶或 AWS 區域重複此史詩。	AWS 管理員，一般 AWS

啟用 EBS 磁碟區的帳戶層級加密

任務	Description	所需的技能
執行啟用指令碼。	1. 在 bash shell 中，使用 cd命令導覽至複製的儲存庫。 2. 輸入以下命令以執行 enable-ebs-encryption-for-account 指令碼。 `./Bash/enable-ebs-encryption-for-account.sh`	AWS 管理員、一般 AWS、Bash
確認設定已更新。	1. 前往 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。 2. 在畫面右側的設定下，選擇資料保護和安全性。 3. 在 EBS 加密區段下，確認一律加密新的 EBS 磁碟區已開啟，且預設加密金鑰已設定為您先前指定的 ARN。 **Note** 如果一律關閉加密新的 EBS 磁碟區設定，或金鑰仍設定為alias/aws/ebs，請確認您已登入執行 shell 指令碼的相同帳戶和 AWS 區域，	AWS 管理員，一般 AWS

任務	Description	所需的技能
	並檢查 shell 是否有錯誤訊息。	
設定其他帳戶或 AWS 區域。	根據您的使用案例，為任何其他帳戶或 AWS 區域重複此史詩。	AWS 管理員，一般 AWS

防止建立未加密的執行個體

任務	Description	所需的技能
建立服務控制政策。	1. 開啟位於 https：//https://console.aws.amazon.com/organizations/v2/ 的 AWS Organizations 主控台。 2. 建立新的服務控制政策。如需詳細資訊，請參閱 AWS Organizations 文件中的建立服務控制政策。 3. 將 的內容`DenyUnencryptedEC2.json` 新增至政策並儲存。您已從第一個史詩中的 GitHub 儲存庫下載此 JSON 檔案。 4. 將此政策連接至組織根目錄或任何必要的組織單位 OUs)。如需詳細資訊，請參閱 AWS Organizations 文件中的連接和分離服務控制政策。	AWS 管理員，一般 AWS

相關資源

AWS 服務文件

- AWS CLI
- AWS Config
- AWS CloudFormation
- Amazon EC2
- AWS KMS
- AWS Organizations
- AWS Systems Manager 自動化

其他資源

- jq 手冊 (jq 網站)
- jq 下載 (GitHub)

在 上的 Stromasys Charon-SSP 模擬器中備份 Sun SPARC 伺服器 AWS 雲端

Kevin Yung 和 Rohit Darji，Amazon Web Services

Luis Ramos，Stromasys

摘要

此模式提供四種選項，可讓您在從內部部署環境遷移至 Amazon Web Services (AWS) 雲端後備份 Sun Microsystems SPARC 伺服器。這些備份選項可協助您實作備份計劃，以符合組織的復原點目標 (RPO) 和復原時間目標 (RTO)、使用自動化方法，並降低整體營運成本。模式提供四個備份選項的概觀，以及實作這些選項的步驟。

如果您在 Stromasys Charon-SSP 模擬器上使用託管為訪客的 Sun SPARC 伺服器，您可以使用下列三個備份選項之一：

- 備份選項 1：Stromasys 虛擬磁帶 – 使用 Charon-SSP 虛擬磁帶功能在 Sun SPARC 伺服器中設定備份設施，並使用AWS Systems Manager 自動化將您的備份檔案存檔到 Amazon Simple Storage Service (Amazon S3)。
- 備份選項 2：Stromasys 快照 – 使用 Charon-SSP 快照功能為 Charon-SSP 中的 Sun SPARC 訪客伺服器設定備份設施。
- 備份選項 3：Amazon Elastic Block Store (Amazon EBS) 磁碟區快照 – 如果您在 Amazon Elastic Compute Cloud (Amazon EC2) 上託管 Charon-SSP 模擬器，您可以使用 Amazon EBS 磁碟區快照來建立 Sun SPARC 檔案系統的備份。

如果您使用託管為硬體訪客和 Amazon EC2 上的 Charon-SSP 的 Sun SPARC 伺服器，您可以使用下列備份選項：

- 備份選項 4： AWS Storage Gateway 虛擬磁帶庫 (VTL) – 使用備份應用程式搭配 Storage Gateway VTL 磁帶閘道來備份 Sun SPARC 伺服器。

如果您使用託管為 Sun SPARC 伺服器中品牌區域的 Sun SPARC 伺服器，您可以使用備份選項 1、2 和 4。

Stromasys 提供軟體和服務來模擬舊版 SPARC、Alpha、VAX 和 PA-RISC 關鍵系統。如需 AWS 雲端 使用 Stromasys 模擬遷移至 的詳細資訊，請參閱 AWS 部落格上的AWS 使用 Stromasys 將 SPARC、Alpha 或其他舊版系統重新託管至 。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 現有的 Sun SPARC 伺服器。
- Charon-SSP 的現有授權。Charon-SSP 的授權可從 取得 AWS Marketplace ，而 Stromasys 虛擬環境 (VE) 的授權可從 Stromasys 取得。如需詳細資訊，請聯絡 Stromasys 銷售。
- 熟悉 Sun SPARC 伺服器和 Linux 備份。
- 熟悉 Charon-SSP 模擬技術。如需詳細資訊，請參閱 Stromasys 文件中的 Stromasys 舊版伺服器模擬。
- 如果您想要為 Sun SPARC 伺服器檔案系統使用虛擬磁帶設施或備份應用程式，您必須為 Sun SPARC 伺服器檔案系統建立和設定備份設施。
- 了解 RPO 和 RTO。如需詳細資訊，請參閱 AWS Well-Architected Framework 文件中的可靠性支柱白皮書中的災難復原目標。
- 若要使用 Backup 選項 4，您必須具有下列項目：
  - 支援 Storage Gateway VTL 磁帶閘道的軟體型備份應用程式。如需詳細資訊，請參閱 AWS Storage Gateway 文件中的使用 VTL 裝置。
  - 安裝和設定 Bacula Director 或類似的備份應用程式。如需詳細資訊，請參閱 Bacula Director 文件。

下表提供此模式中四個備份選項的相關資訊。

備份選項	實現當機一致性？	實現應用程式一致性？	虛擬備份設備解決方案？	典型 使用案例
選項 1 – Stromasys 虛擬磁帶	是 您可以自動化 Sun SPARC 檔案系統快照，以備份虛擬磁帶中的資料。例如，您可以使用 UFS 或 ZFS 快照。	是 此備份選項需要自動化指令碼來排清傳輸中的交易、在檔案系統快照期間設定唯讀或暫時離線模式，或取得應用	是	使用 .tar 或 .zip 檔案備份 Sun SPARC 伺服器檔案系統 應用程式資料備份

		程式資料傾印。您可能還需要應用程式停機時間或唯讀模式。		
選項 2 – Stromasys 快照	是 您必須設定 Charon-SSP Manager 或使用命令列啟動引數來啟用此功能。 您也必須執行 Linux 命令，要求 Charon-SSP 模擬器將 Sun SPARC 訪客伺服器狀態儲存到快照檔案中。 **Important** 您必須關閉 Sun SPARC 訪客伺服器。	是 此備份選項會建立模擬訪客伺服器的快照，包括其虛擬磁碟和記憶體傾印。 **Important** 您必須在快照期間關閉 Sun SPARC 訪客伺服器。	否	Sun SPARC 伺服器快照 應用程式資料備份

選項 3 – Amazon EBS 磁碟區快照	是 您可以使用 AWS Backup 來自動化 Amazon EBS 快照。	是 此備份選項需要自動化指令碼來排清傳輸中的交易，並在 Amazon EBS 磁碟區快照期間設定 Amazon EC2 執行個體的唯讀或暫時停止。 **Important** 此備份選項可能需要應用程式停機時間或唯讀模式，才能達到應用程式的一致性。	否	Sun SPARC 伺服器檔案系統快照 應用程式資料備份

選項 4 – AWS Storage Gateway VTL	是 您可以使用備份代理程式，自動將 Sun SPARC 檔案系統備份資料備份至 VTL。	是 此備份選項需要自動化指令碼來排清傳輸中的交易，並在檔案系統快照或應用程式資料傾印期間設定唯讀或暫時離線模式。 **Important** 此備份選項可能需要應用程式停機時間或唯讀模式。	是	Sun SPARC 伺服器檔案系統備份的大型機群 應用程式資料備份

限制

- 您可以使用此模式的方法來備份個別的 Sun SPARC 伺服器，但如果您有在叢集中執行的應用程式，您也可以將這些備份選項用於共用資料。

工具

備份選項 1： Stromasys 虛擬磁帶

- Stromasys Charon-SSP 模擬器會在標準 64 位元 x86 相容電腦系統內建立原始 SPARC 硬體的虛擬複本。它執行原始 SPARC 二進位程式碼，包括作業系統 (OSs)，例如 SunOS 或 Solaris、其分層產品和應用程式。
- Amazon Elastic Compute Cloud (Amazon EC2) 是一種 Web 服務，可提供可調整大小的運算容量，讓您用來建置和託管軟體系統。
- Amazon Elastic File System (Amazon EFS) 提供簡單、無伺服器、set-and-forget的彈性檔案系統，可與 AWS 服務 和內部部署資源搭配使用。

- Amazon Simple Storage Service (Amazon S3) 是網際網路的儲存體。
- AWS Systems Manager 自動化可簡化 Amazon EC2 執行個體和其他 AWS 資源的常見維護和部署任務。

備份選項 2：Stromasys 快照

- Stromasys Charon-SSP 模擬器會在標準 64 位元 x86 相容電腦系統內建立原始 SPARC 硬體的虛擬複本。它執行原始 SPARC 二進位程式碼，包括 SunOS 或 Solaris 等OSs、其分層產品和應用程式。
- Amazon Elastic Compute Cloud (Amazon EC2) 是一種 Web 服務，可提供可調整大小的運算容量，讓您用來建置和託管軟體系統。
- Amazon Elastic File System (Amazon EFS) 提供簡單、無伺服器、set-and-forget的彈性檔案系統，可與 AWS 服務 和內部部署資源搭配使用。
- Amazon Simple Storage Service (Amazon S3) 是網際網路的儲存體。
- AWS Systems Manager 自動化可簡化 Amazon EC2 執行個體和其他 AWS 資源的常見維護和部署任務。

備份選項 3：Amazon EBS 磁碟區快照

- Stromasys Charon-SSP 模擬器會在標準 64 位元 x86 相容電腦系統內建立原始 SPARC 硬體的虛擬複本。它執行原始 SPARC 二進位程式碼，包括 SunOS 或 Solaris 等OSs、其分層產品和應用程式。
- AWS Backup 是一種全受管的資料保護服務，可讓您輕鬆集中和自動化跨 AWS 服務雲端、雲端和內部部署。
- Amazon Elastic Block Store (Amazon EBS) 提供區塊層級儲存磁碟區，可與 Amazon EC2 執行個體搭配使用。
- Amazon Elastic Compute Cloud (Amazon EC2) 是一種 Web 服務，可提供可調整大小的運算容量，讓您用來建置和託管軟體系統。

備份選項 4：AWS Storage GatewayVTL

- Stromasys Charon-SSP 模擬器會在標準 64 位元 x86 相容電腦系統內建立原始 SPARC 硬體的虛擬複本。它執行原始 SPARC 二進位程式碼，包括 SunOS 或 Solaris 等OSs、其分層產品和應用程式。
- Bacula 是開放原始碼的企業級電腦備份系統。如需現有備份應用程式是否支援磁帶閘道的詳細資訊，請參閱 AWS Storage Gateway 文件中的磁帶閘道支援的第三方備份應用程式。
- Amazon Elastic Compute Cloud (Amazon EC2) 是一種 Web 服務，可提供可調整大小的運算容量，讓您用來建置和託管軟體系統。
- 適用於 MySQL 的 Amazon Relational Database Service (Amazon RDS) MySQL 支援執行多個 MySQL 版本的資料庫執行個體。
- Amazon Simple Storage Service (Amazon S3) 是網際網路的儲存體。
- AWS Storage Gateway 將內部部署軟體設備與雲端儲存連線，以便在內部部署 IT 環境與 AWS 儲存基礎設施之間，提供與資料安全功能的無縫整合。

史詩

備份選項 1 – 建立 Stromasys 虛擬磁帶備份

任務	Description	所需的技能
建立虛擬磁帶檔案儲存的 Amazon EFS 共用檔案系統。	登入 AWS 管理主控台 或使用 AWS Command Line Interface (AWS CLI) 來建立 Amazon EFS 檔案系統。 如需詳細資訊，請參閱 Amazon EFS 文件中的建立 Amazon EFS 檔案系統。 EFS	雲端架構師
設定 Linux 主機以掛載共用檔案系統。	在 Amazon EC2 Linux 執行個體上安裝 Amazon EFS 驅動程式，並設定 Linux 作業系統在啟動期間掛載 Amazon EFS 共用檔案系統。 如需詳細資訊，請參閱《Amazon EFS 文件》中的使	DevOps 工程師

任務	Description	所需的技能
	用 Amazon EFS 掛載協助程式掛載檔案系統。 EFS	
安裝 Charon-SSP 模擬器。	在 Amazon EC2 Linux 執行個體上安裝 Charon-SSP 模擬器。 如需詳細資訊，請參閱 Stromasys 文件中的為 Charon-SSP 設定 AWS 雲端執行個體。	DevOps 工程師
為每個 Sun SPARC 訪客伺服器在共用檔案系統中建立虛擬磁帶檔案容器。	執行 `touch <vtape-container-name>` 命令，針對 Charon-SSP 模擬器中部署的每個 Sun SPARC 訪客伺服器，在共用檔案系統中建立虛擬磁帶檔案容器。	DevOps 工程師
設定 Charon-SSP Manager 為 Sun SPARC 訪客伺服器建立虛擬磁帶裝置。	登入 Charon-SSP Manager，建立虛擬磁帶裝置，並將其設定為為每個 Sun SPARC 訪客伺服器使用虛擬磁帶容器檔案。 如需詳細資訊，請參閱 Stromasys 文件中的適用於 Linux 的 Charon-SSP 5.2 使用者指南。	DevOps 工程師
驗證虛擬磁帶裝置是否可在 Sun SPARC 訪客伺服器中使用。	登入每個 Sun SPARC 訪客伺服器並執行 `mt -f /dev/rmt/1` 命令，以驗證虛擬磁帶裝置已在作業系統中設定。	DevOps 工程師

任務	Description	所需的技能
開發 Systems Manager Automation Runbook 和自動化。	開發 Systems Manager Automation Runbook，並在 Systems Manager 中設定維護時段和關聯，以排程備份程序。 如需詳細資訊，請參閱 AWS Systems Manager 文件中的自動化演練和設定維護時段。	雲端架構師
設定 Systems Manager 自動化來封存輪換的虛擬磁帶容器檔案。	使用其他資訊區段中返回選項 1 的程式碼範例，開發 Systems Manager Automation Runbook，將輪換的虛擬磁帶容器檔案封存至 Amazon S3。	雲端架構師
部署 Systems Manager Automation Runbook 以進行封存和排程。	部署 Systems Manager Automation Runbook 並排程在 Systems Manager 中自動執行。 如需詳細資訊，請參閱 Systems Manager 文件中的自動化演練。	雲端架構師

備份選項 2 – 建立 Stromasys 快照

任務	Description	所需的技能
建立虛擬磁帶檔案儲存的 Amazon EFS 共用檔案系統。	登入 AWS 管理主控台 或使用 AWS CLI 建立 Amazon EFS 檔案系統。	雲端架構師

任務	Description	所需的技能
	如需詳細資訊，請參閱 Amazon EFS 文件中的建立 Amazon EFS 檔案系統。 EFS	
設定 Linux 主機以掛載共用檔案系統。	在 Amazon EC2 Linux 執行個體中安裝 Amazon EFS 驅動程式，並設定 Linux 作業系統在啟動期間掛載 Amazon EFS 共用檔案系統。 如需詳細資訊，請參閱 Amazon EFS 文件中的使用 Amazon EFS 掛載協助程式掛載檔案系統。 EFS	DevOps 工程師
安裝 Charon-SSP 模擬器。	在 Amazon EC2 Linux 執行個體上安裝 Charon-SSP 模擬器。 如需詳細資訊，請參閱 Stromasys 文件中的為 Charon-SSP 設定 AWS 雲端執行個體。	DevOps 工程師
設定 Sun SPARC 訪客伺服器以使用快照選項啟動。	使用 Charon-SSP Manager 為每個 Sun SPARC 訪客伺服器設定快照選項。 如需詳細資訊，請參閱 Stromasys 文件中的適用於 Linux 的 Charon-SSP 5.2 使用者指南。	DevOps 工程師

任務	Description	所需的技能
開發 Systems Manager Automation Runbook。	使用額外資訊區段中備份選項 2 的程式碼範例，開發 Systems Manager Automation Runbook，以在維護時段期間在 Sun SPARC 訪客伺服器上遠端執行快照命令。	雲端架構師
部署 Systems Manager Automation Runbook 並設定與 Amazon EC2 Linux 主機的關聯。	部署 Systems Manager Automation Runbook，並在 Systems Manager 中設定維護時段和關聯，以排程備份程序。 如需詳細資訊，請參閱 AWS Systems Manager 文件中的自動化演練和設定維護時段。	雲端架構師
將快照封存至長期儲存。	使用其他資訊區段中的 Runbook 範例程式碼來開發 Systems Manager Automation Runbook，將快照檔案封存至 Amazon S3。	雲端架構師

備份選項 3 – 建立 Amazon EBS 磁碟區快照

任務	Description	所需的技能
安裝 Charon-SSP 模擬器。	在 Amazon EC2 Linux 執行個體上安裝 Charon-SSP 模擬器。 如需詳細資訊，請參閱 Stromasys 文件中的為 Charon-SSP 設定 AWS 雲端執行個體。	DevOps 工程師

任務	Description	所需的技能
為 Sun SPRAC 訪客伺服器建立 Amazon EBS 磁碟區。	登入 AWS 管理主控台，開啟 Amazon EBS 主控台，然後為 Sun SPRAC 訪客伺服器建立 Amazon EBS 磁碟區。 如需詳細資訊，請參閱 Stromasys 文件中的為 Charon-SSP 設定 AWS 雲端執行個體。	雲端架構師
將 Amazon EBS 磁碟區連接至 Amazon EC2 Linux 執行個體。	在 Amazon EC2 主控台上，將 Amazon EBS 磁碟區連接至 Amazon EC2 Linux 執行個體。 如需詳細資訊，請參閱 Amazon EC2 文件中的將 Amazon EBS 磁碟區連接至執行個體。 Amazon EC2	AWS DevOps
在 Charon-SSP 模擬器中將 Amazon EBS 磁碟區對應為 SCSI 磁碟機。	設定 Charon-SSP Manager 將 Amazon EBS 磁碟區映射為 Sun SPARC 訪客伺服器中的 SCSI 磁碟機。 如需詳細資訊，請參閱 Stromasys 文件中適用於 Linux 的 Charon-SSP V5.2 指南的 SCSI 儲存組態一節。	AWS DevOps

任務	Description	所需的技能
設定快照 Amazon EBS 磁碟區的 AWS Backup 排程。	設定 AWS Backup 政策和排程以快照 Amazon EBS 磁碟區。 如需詳細資訊，請參閱 AWS 開發人員中心文件中使用教學課程的 Amazon EBS 備份和還原 AWS Backup。	AWS DevOps

備份選項 4 – 建立 AWS Storage Gateway VTL

任務	Description	所需的技能
建立磁帶閘道裝置。	登入 AWS 管理主控台，開啟 AWS Storage Gateway 主控台，然後在 VPC 中建立磁帶閘道裝置。 如需詳細資訊，請參閱 AWS Storage Gateway 文件中的建立閘道。	雲端架構師
為 Bacula Catalog 建立 Amazon RDS 資料庫執行個體。	開啟 Amazon RDS 主控台並建立 Amazon RDS for MySQL 資料庫執行個體。 如需詳細資訊，請參閱《Amazon RDS 文件》中的建立 MySQL 資料庫執行個體並連線至 MySQL 資料庫執行個體上的資料庫。	雲端架構師
在 VPC 中部署備份應用程式控制器。	在 Amazon EC2 執行個體上安裝 Bacula、部署備份應用程式控制器，然後設定備份儲存體以與磁帶閘道裝置連線。您可以在 `Bacula-storage-`	AWS DevOps

任務	Description	所需的技能
	`daemon-config.txt` 檔案（已連接）中使用範例 Bacula Director 儲存協助程式組態。 如需詳細資訊，請參閱 Bacula 文件。	
在 Sun SPARC 訪客伺服器上設定備份應用程式。	設定第二個用戶端，使用 `SUN-SPARC-Guest-Bacula-Config.txt` 檔案（已連接）中的範例 Bacula 組態，在 Sun SPARC 訪客伺服器上安裝和設定備份應用程式。	DevOps 工程師
設定備份組態和排程。	使用 `Bacula-Director-Config.txt` 檔案（已連接）中的範例 Bacula Director 組態，在備份應用程式控制器中設定備份組態和排程。 如需詳細資訊，請參閱 Bacula 文件。	DevOps 工程師
驗證備份組態和排程是否正確。	遵循 Bacula 文件的指示，在 Sun SPARC 訪客伺服器中執行安裝的驗證和備份測試。 例如，您可以使用下列命令來驗證組態檔案： • `bacula-dir -t -c bacula-dir.conf` • `bacula-fd -t -c bacula-fd.conf` • `bacula-sd -t -c bacula-sd.conf`	DevOps 工程師

相關資源

- 具有 VE 授權的 Charon Virtual SPARC
- Charon Virtual SPARC
- 搭配 Bacula Enterprise Edition 使用雲端服務和物件儲存
- 災難復原 (DR) 目標
- Charon 傳統系統模擬解決方案

其他資訊

備份選項 1 – 建立 Stromasys 虛擬磁帶

您可以使用下列範例 Systems Manager Automation Runbook 程式碼自動啟動備份，然後交換磁帶：

```
...
# example backup script saved in SUN SPARC Server
 #!/usr/bin/bash
 mt -f  rewind
 tar -cvf
 mt -f  offline
...
         mainSteps:
         - action: aws:runShellScript
           name:
           inputs:
             onFailure: Abort
             timeoutSeconds: "1200"
             runCommand:
             - |
               # Validate tape backup container file exists
               if [ ! -f {{TapeBackupContainerFile}} ]; then
                 logger -s -p local3.warning "Tape backup container file is not exists
 - {{TapeBackupContainerFile}}, create a new one"
                 touch {{TapeBackupContainerFile}}
               fi
         - action: aws:runShellScript
           name: startBackup
           inputs:
             onFailure: Abort
             timeoutSeconds: "1200"
             runCommand:
```

```
            - |
              user={{BACKUP_USER}}
              keypair={{KEYPAIR_PATH}}
              server={{SUN_SPARC_IP}}
              backup_script={{BACKUP_SCRIPT}}
              ssh -i $keypair $user@$server -c "/usr/bin/bash $backup_script"
        - action: aws:runShellScript
          name: swapVirtualDiskContainer
          inputs:
            onFailure: Abort
            timeoutSeconds: "1200"
            runCommand:
            - |
              mv {{TapeBackupContainerFile}} {{TapeBackupContainerFile}}.$(date +%s)
              touch {{TapeBackupContainerFile}}
        - action: aws:runShellScript
          name: uploadBackupArchiveToS3
          inputs:
            onFailure: Abort
            timeoutSeconds: "1200"
            runCommand:
            - |
              aws s3 cp {{TapeBackupContainerFile}} s3://{{BACKUP_BUCKET}}/
{{SUN_SPARC_IP}}/$(date '+%Y-%m-%d')/
 ...
```

備份選項 2 – Stromasys 快照

您可以使用下列範例 Systems Manager Automation Runbook 程式碼來自動化備份程序：

```
      ...

         mainSteps:
         - action: aws:runShellScript
           name: startSnapshot
           inputs:
             onFailure: Abort
             timeoutSeconds: "1200"
             runCommand:
             - |
               # You may consider some graceful stop of the application before taking a
 snapshot
               # Query SSP PID by configuration file
```

```
              # Example: ps ax | grep ssp-4 | grep Solaris10.cfg | awk '{print $1"
 "$5}' | grep ssp4 | cut -f1 -d" "
              pid=`ps ax | grep ssp-4 | grep {{SSP_GUEST_CONFIG_FILE}} | awk '{print
 $1" "$5}' | grep ssp4 | cut -f1 -d" "`
              if [ -n "${pid}" ]; then
                kill -SIGTSTP ${pid}
              else
                echo "No PID found for SPARC guest with config
 {{SSP_GUEST_CONFIG_FILE}}"
                exit 1
              fi
        - action: aws:runShellScript
          name: startBackup
          inputs:
            onFailure: Abort
            timeoutSeconds: "1200"
            runCommand:
            - |
              # upload snapshot and virtual disk files into S3
              aws s3 sync {{SNAPSHOT_FOLDER}} s3://{{BACKUP_BUCKET}}/$(date '+%Y-%m-
%d')/
              aws s3 cp {{VIRTUAL_DISK_FILE}} s3://{{BACKUP_BUCKET}}/$(date '+%Y-%m-
%d')/
        - action: aws:runShellScript
          name: restratSPARCGuest
          inputs:
            onFailure: Abort
            timeoutSeconds: "1200"
            runCommand:
            - |
              /opt/charon-ssp/ssp-4u/ssp4u -f {{SSP_GUEST_CONFIG_FILE}} -d -a
 {{SPARC_GUEST_NAME}} --snapshot {{SNAPSHOT_FOLDER}}
 ...
```

備份選項 4 – AWS Storage Gateway VTL

如果您使用 Solaris 非全域區域來執行虛擬化舊版 Sun SPARC 伺服器，備份應用程式方法可以套用至在 Sun SPARC 伺服器中執行的非全域區域（例如，備份用戶端可以在非全域區域內執行）。不過，備份用戶端也可以在 Solaris 主機中執行，並拍攝非全域區域的快照。然後，可以在磁帶上備份快照。

下列範例組態會將託管 Solaris 非全域區域的檔案系統新增至 Solaris 主機的備份組態：

```
FileSet {
   Name = "Branded Zones"
   Include {
     Options {
       signature = MD5
     }
     File = /zones
   }
 }
```

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Veeam Backup & Replication 備份資料並存檔至 Amazon S3

Jeanna James，Anthony Fiore (AWS) 和 William Quigley，Amazon Web Services

總結

此模式詳細說明使用 Veeam 向外擴展備份儲存庫功能，將 Veeam Backup & Replication 建立的備份傳送至支援的 Amazon Simple Storage Service (Amazon S3) 物件儲存類別的程序。

Veeam 支援多個 Amazon S3 儲存類別，最適合您的特定需求。您可以根據備份或封存資料的資料存取、彈性和成本需求，選擇儲存體類型。例如，您可以將您不打算在 Amazon S3 不常存取 (IA) 中使用的資料存放 30 天或更長時間，以降低成本。如果您打算將資料封存 90 天或更長時間，您可以使用 S3 Glacier Flexible Retrieval 或 S3 Glacier Deep Archive 搭配 Veeam 的封存層。您也可以使用 S3 物件鎖定，在 Amazon S3 中進行不可變的備份。

此模式不包含如何使用磁帶閘道設定 Veeam Backup & Replication AWS Storage Gateway。如需有關該主題的資訊，請參閱 Veeam 網站上的使用 AWS VTL 閘道的 Veeam 備份與複寫 - 部署指南。

警告：此案例需要具有程式設計存取和長期登入資料的 AWS Identity and Access Management (IAM) 使用者，這會造成安全風險。為了協助降低此風險，建議您只為這些使用者提供執行任務所需的許可，並在不再需要這些使用者時將其移除。必要時可以更新存取金鑰。如需詳細資訊，請參閱《IAM 使用者指南》中的更新存取金鑰。

先決條件和限制

先決條件

- 已安裝 Veeam Backup & Replication，包括 Veeam Availability Suite 或 Veeam Backup Essentials（您可以註冊免費試用）
- 具有 Enterprise 或 Enterprise Plus 功能的 Veeam Backup & Replication 授權，其中包括 Veeam Universal License (VUL)
- 可存取 Amazon S3 儲存貯體的作用中 IAM 使用者
- 如果使用封存層，則可以存取 Amazon Elastic Compute Cloud (Amazon EC2) 和 Amazon Virtual Private Cloud (Amazon VPC) 的作用中 IAM 使用者
- 從內部部署到 的網路連線 AWS 服務 ，具有可用的頻寬，可透過公有網際網路連線或 AWS Direct Connect 公有虛擬介面 (VIF) 備份和還原流量

- 開啟下列網路連接埠和端點，以確保與物件儲存庫進行適當的通訊：
  - Amazon S3 儲存 – TCP – 連接埠 443：用於與 Amazon S3 儲存通訊。
  - Amazon S3 儲存 – 雲端端點 – `*.amazonaws.com` 適用於 AWS 區域 和 AWS GovCloud (US) Regions，或`*.amazonaws.com.cn`適用於中國區域：用於與 Amazon S3 儲存通訊。如需連線端點的完整清單，請參閱 AWS 文件中的 Amazon S3 端點。
  - Amazon S3 儲存 – TCP HTTP – 連接埠 80：用於驗證憑證狀態。請考慮憑證驗證端點 — 憑證撤銷清單 (CRL) URLs和線上憑證狀態通訊協定 (OCSP) 伺服器 — 可能會有所變更。您可以在憑證本身中找到實際的地址清單。
  - Amazon S3 儲存 – 憑證驗證端點 – `*.amazontrust.com`：用於驗證憑證狀態。請考慮憑證驗證端點 (CRL URLs和 OCSP 伺服器） 可能會有所變更。您可以在憑證本身中找到實際的地址清單。

限制

- Veeam 不支援在任何做為 Veeam 物件儲存庫的 S3 儲存貯體上使用 S3 S3生命週期政策。其中包括具有 Amazon S3 儲存類別轉換的政策，以及 S3 生命週期過期規則。Veeam 必須是管理這些物件的唯一實體。啟用 S3 生命週期政策可能會有非預期的結果，包括資料遺失。

產品版本

- Veeam Backup & Replication v9.5 Update 4 或更新版本（僅限備份或容量層）
- Veeam Backup & Replication v10 或更新版本（備份或容量層和 S3 物件鎖定）
- Veeam Backup & Replication v11 或更新版本（備份或容量層、封存或封存層，以及 S3 物件鎖定）
- Veeam Backup & Replication v12 或更新版本（效能層、備份或容量層、封存或封存層，以及 S3 物件鎖定）
- S3 Standard
- S3 標準 – IA
- S3 單區域 – IA
- S3 Glacier Flexible Retrieval（僅限 v11 和更新版本）
- S3 Glacier Deep Archive（僅限 v11 和更新版本）
- S3 Glacier Instant Retrieval（僅限 v12 和更新版本）

Architecture

來源技術堆疊

- 內部部署 Veeam Backup & Replication 安裝，可從 Veeam 備份伺服器或 Veeam 閘道伺服器連線至 Amazon S3

目標技術堆疊

- Amazon S3
- Amazon VPC 和 Amazon EC2 （ 如果使用封存層 ）

目標架構：SOBR

下圖顯示橫向擴展備份儲存庫 (SOBR) 架構。

Veeam Backup and Replication 軟體可保護資料免於發生邏輯錯誤，例如系統故障、應用程式錯誤或意外刪除。在此圖表中，備份會先在內部部署執行，次要副本會直接傳送至 Amazon S3。備份代表資料的point-in-time副本。

工作流程包含分層或複製備份至 Amazon S3 所需的三個主要元件，以及一個選用元件：

- Veeam Backup & Replication (1) – 負責協調、控制和管理備份基礎設施、設定、任務、復原任務和其他程序的備份伺服器。
- Veeam 閘道伺服器 （ 圖中未顯示 ） – 如果 Veeam 備份伺服器沒有與 Amazon S3 的傳出連線，則需要選用的內部部署閘道伺服器。
- 橫向擴展備份儲存庫 (2) – 支援資料多層儲存的儲存庫系統。向外擴展備份儲存庫包含一或多個備份儲存庫，可提供對資料的快速存取，並且可以使用 Amazon S3 物件儲存庫進行擴充，以進行長期儲存 （ 容量層 ） 和封存 （ 封存層 ）。Veeam 使用向外擴展備份儲存庫，在本機 （ 效能層 ） 和 Amazon S3 物件儲存 （ 容量和封存層 ） 之間自動分層資料。

> **Note**
>
> 從 Veeam Backup & Replication v12.2 開始，Direct to S3 Glacier 功能可讓 S3 容量層變成選用。SOBR 可以設定效能層和 S3 Glacier 封存層。此組態適用於對容量層的本機 （ 內部部署 ） 儲存有大量投資，且只需要在雲端中長期保留封存的使用者。如需詳細資訊，請參閱 Veeam Backup & Replication 文件。

- Amazon S3 (3) – AWS 物件儲存服務，可提供可擴展性、資料可用性、安全性和效能。

目標架構：DTO

下圖顯示direct-to-object(DTO) 架構。

在此圖表中，備份資料會直接傳送至 Amazon S3，而不先存放在內部部署。次要副本可以存放在 S3 Glacier 中。

自動化和擴展

您可以使用 VeeamHub GitHub 儲存庫中提供的 CloudFormation 範本，自動建立 IAM 資源和 S3 儲存貯體。範本包含標準和不可變的選項。

工具

工具和 AWS 服務

- Veeam Backup & Replication 是 Veeam 的解決方案，用於保護、備份、複寫和還原您的虛擬和實體工作負載。
- CloudFormation 可協助您建立和設定 AWS 資源的模型、快速且一致地佈建資源，以及在整個生命週期中進行管理。您可以使用範本來描述資源及其相依性，並將它們一起啟動和設定為堆疊，而不是個別管理資源。您可以跨多個 和 管理和佈建堆疊 AWS 帳戶 AWS 區域。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以使用 Amazon EC2 視需要啟動任意數量或任意數量的虛擬伺服器，也可以向外擴展或向內擴展。
- AWS Identity and Access Management (IAM) 是一種 Web 服務，可安全地控制對 的存取 AWS 服務。透過 IAM，您可以集中管理使用者、存取金鑰等安全登入資料，以及控制使用者和應用程式可存取之 AWS 資源的許可。
- Amazon Simple Storage Service (Amazon S3) 是一種物件儲存服務。您可以使用 Amazon S3 隨時從 Web 任何地方存放和擷取任意資料量。
- Amazon S3 Glacier (S3 Glacier) 是一種安全且耐用的服務，用於低成本的資料封存和長期備份。
- Amazon Virtual Private Cloud (Amazon VPC) 會佈建 的邏輯隔離區段 AWS 雲端 ，您可以在已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路與您在資料中心中操作的傳統網路非常相似，且具備使用 AWS可擴展基礎設施的優勢。

Code

使用 VeeamHub GitHub 儲存庫中提供的 CloudFormation 範本，自動為此模式建立 IAM 資源和 S3 儲存貯體。如果您想要手動建立這些資源，請遵循 Epics 區段中的步驟。

最佳實務

根據 IAM 最佳實務，強烈建議您定期輪換長期 IAM 使用者憑證，例如用於將 Veeam Backup & Replication 備份寫入 Amazon S3 的 IAM 使用者。如需詳細資訊，請參閱 IAM 文件中的安全最佳實務。

史詩

在帳戶中設定 Amazon S3 儲存體

任務	Description	所需的技能
建立 IAM 使用者。	遵循 IAM 文件中的指示來建立 IAM 使用者。此使用者不應擁有 AWS 主控台存取權，而且您將需要為此使用者建立存取金鑰。Veeam 使用此實體向進行身分驗證 AWS ，以讀取和寫入 S3 儲存貯體。您必須授予最低權限（也就是僅授予執行任務所需的許可），讓使用者沒有比所需更多的授權。如需連接至 Veeam IAM 使用者的 IAM 政策範例，請參閱其他資訊一節。 **Note** 或者，您可以使用 VeeamHub GitHub 儲存庫中提供的 CloudFormation 範本，為此模式建立 IAM	AWS 管理員

任務	Description	所需的技能
	使用者和 S3 儲存貯體。	
建立 S3 儲存貯體。	1. 登入 AWS 管理主控台 並開啟 Amazon S3 主控台。 2. 如果您還沒有可用作目標儲存的現有 S3 儲存貯體，請選擇建立儲存貯體，並指定儲存貯體名稱 AWS 區域和儲存貯體設定。 • 我們建議您為 S3 儲存貯體啟用封鎖公開存取選項，並設定存取和使用者許可政策，以符合組織的需求。如需範例，請參閱 Amazon S3documentation。 • 我們建議您啟用 S3 物件鎖定，即使您不打算立即使用它。此設定只能在建立 S3 儲存貯體時啟用。 如需詳細資訊，請參閱 Amazon S3 文件中的建立儲存貯體。	AWS 管理員

將 Amazon S3 和 S3 Glacier Flexible Retrieval （或 S3 Glacier Deep Archive) 新增至 Veeam Backup & Replication

任務	Description	所需的技能
啟動新的物件儲存庫精靈。	在 Veeam 中設定物件儲存和橫向擴展備份儲存庫之前，您	AWS 管理員、應用程式擁有者

任務	Description	所需的技能
	必須新增要用於容量和封存層的 Amazon S3 和 S3 Glacier 儲存庫。在下一個史詩中，您將將這些儲存庫連接到橫向擴展備份儲存庫。 1. 在 Veeam 主控台上，開啟 Backup Infrastructure 檢視。 2. 在庫存窗格中，選擇備份儲存庫節點，然後選擇新增儲存庫。 3. 在新增備份儲存庫對話方塊中，選擇物件儲存，Amazon S3。	

任務	Description	所需的技能
為容量層新增 Amazon S3 儲存體。	1. 在 Amazon Cloud Storage Services 對話方塊中，選擇 Amazon S3。 2. 在精靈的名稱步驟中，指定物件儲存體名稱和簡短描述，例如建立者和建立日期。 3. 在精靈的帳戶步驟中，指定物件儲存體帳戶。 • 針對登入資料，選擇您在第一個 epic 中建立的 IAM 使用者，以存取您的 Amazon S3 物件儲存。 • 針對 AWS 區域，選擇 S3 AWS 區域 儲存貯體所在的 。 4. 在精靈的儲存貯體步驟中，指定物件儲存設定。 • 針對資料中心區域，選擇 S3 AWS 區域 儲存貯體所在的 。 • 針對儲存貯體，選擇您在第一個特徵中建立的 S3 儲存貯體。 • 針對資料夾，建立或選取要映射物件儲存庫的雲端資料夾。 • 如果您想要啟用不可變性，請選擇讓最近的備份在 X 天內不可變，並設定備份應該鎖定的期間。請注意，啟用不可變性會導致成本增加，因為 Veeam	AWS 管理員、應用程式擁有者

任務	Description	所需的技能
	對 Amazon S3 的 API 呼叫數量增加。 5. 在精靈的摘要步驟中，檢閱組態資訊，然後選擇完成。	

任務	Description	所需的技能
新增封存層的 S3 Glacier 儲存體。	如果您想要建立封存層，請使用其他資訊區段中詳述的 IAM 許可。 1. 如前所述啟動新物件儲存庫精靈。 2. 在 Amazon Cloud Storage Services 對話方塊中，選擇 Amazon S3 Glacier。 3. 在精靈的名稱步驟中，指定物件儲存體名稱和簡短描述，例如建立者和建立日期。 4. 在精靈的帳戶步驟中，指定物件儲存體帳戶。 • 針對登入資料，選擇您在第一個 epic 中建立的 IAM 使用者，以存取您的 S3 Glacier 物件儲存。 • 針對 AWS 區域，選擇 S3 AWS 區域 儲存貯體所在的 。 5. 在精靈的儲存貯體步驟中，指定物件儲存設定。 • 針對資料中心區域，選擇 AWS 區域。 • 針對儲存貯體，選擇 S3 儲存貯體來存放備份資料。這可以與您用於容量層的儲存貯體相同。 • 針對資料夾，建立或選取要映射物件儲存庫的雲端資料夾。	AWS 管理員、應用程式擁有者

任務	Description	所需的技能
	• 如果您想要啟用不可變性，請選擇讓最近的備份在保留政策的整個期間不可變。請注意，啟用不可變性會導致成本增加，因為 Veeam 對 Amazon S3 的 API 呼叫數量增加。 • 如果您想要使用 S3 Glacier Deep Archive 做為封存儲存類別，請選擇使用 Deep Archive 儲存類別。 6. 在精靈的 Proxy Appliance 步驟中，設定用於將資料從 Amazon S3 傳輸至 S3 Glacier 的輔助執行個體。您可以使用預設設定或手動設定每個設定。若要手動設定設定： • 請選擇 Customize (自訂)。 • ForEC2 執行個體類型，根據將備份檔案傳輸至橫向擴展備份儲存庫的封存層的速度和成本需求，選擇代理設備的執行個體類型。 • 針對 Amazon VPC，選擇目標執行個體的 VPC。 • 針對子網路，選擇代理設備的子網路。	

任務	Description	所需的技能
	• 針對安全群組，選擇要與代理設備建立關聯的安全群組。 • 針對重新導向連接埠，指定 TCP 連接埠，以在代理設備與備份基礎設施元件之間路由請求。 • 選擇確定以確認您的設定。 7. 在精靈的摘要步驟中，檢閱組態資訊，然後選擇完成。	

新增橫向擴展備份儲存庫

任務	Description	所需的技能
啟動新的橫向擴展備份儲存庫精靈。	1. 在 Veeam 主控台上，開啟 Backup Infrastructure 檢視。 2. 在庫存窗格中，選擇橫向擴展儲存庫，然後選擇新增橫向擴展儲存庫。	應用程式擁有者、AWS 系統管理員
新增橫向擴展備份儲存庫，並設定容量和封存層。	1. 在精靈的名稱步驟中，指定向外擴展備份儲存庫的名稱和簡短描述。 2. 如有需要，請新增效能範圍。您也可以使用現有的 Veeam 本機備份儲存庫做為效能層。從 Veeam 第 12 版開始，您可以新增 S3 儲存貯體做為direct-to-object	應用程式擁有者、AWS 系統管理員

任務	Description	所需的技能
	(DTO) 備份的效能範圍，略過本機效能層。 3. 選擇進階，並指定向外擴展備份儲存庫的其他選項。 • 選擇使用每個機器備份檔案，為每個機器建立單獨的備份檔案，並同時將這些檔案寫入多個串流中的備份儲存庫。建議使用此選項來提高儲存和運算資源使用率。 • 選擇在必要範圍離線時執行完整備份，以建立完整備份檔案，以防包含增量備份還原點的範圍離線。此選項需要橫向擴展備份儲存庫中的可用空間，才能託管完整的備份檔案。 4. 在精靈的政策步驟中，指定儲存庫的備份置放政策。 • 選擇資料地區性，將屬於相同鏈的完整和增量備份檔案一起存放至相同的效能範圍。您可以將屬於新備份鏈的檔案存放至相同效能範圍，或存放至另一個備份鏈（除非您使用重複資料刪除儲存設備作為效能範圍）。 • 選擇效能，將完整和增量備份檔案存放到不同的效能範圍。此選項需要快速且可靠的網路連線。如果您選擇效能，您可以限制	

任務	Description	所需的技能
	要存放在每個效能範圍的備份檔案類型。例如，您可以將完整備份檔案存放在一個範圍，將增量備份檔案存放在其他範圍。若要選擇檔案類型： • 請選擇 Customize (自訂)。 • 在備份配置設定對話方塊中，選擇效能範圍，然後選擇編輯。 • 選擇您要存放範圍的備份檔案類型。 5. 在精靈的容量層步驟中，設定您要連接到橫向擴展備份儲存庫的長期儲存層。 • 選擇使用物件儲存擴展備份儲存庫容量。針對物件儲存庫，選擇您在上一個特徵中新增的容量層的 Amazon S3 儲存體。 • 選擇視窗以選取移動或複製資料的時段。 • 選擇建立物件儲存時立即將備份複製到物件儲存體，以將所有或僅最近建立的備份檔案複製到容量範圍。 • 選擇在物件儲存體超出操作還原時段時，將備份移至物件儲存體，以將非作用中備份鏈傳輸到容量範圍。在移動早於 X 天的備	

任務	Description	所需的技能
	份檔案欄位中，指定備份檔案應卸載的持續時間。（若要在建立非作用中備份鏈當天卸載，請指定 0 天。）如果向外擴展備份儲存庫已達到您指定的閾值，您也可以選擇覆寫以更快地移動備份檔案。 • 選擇加密上傳至物件儲存的資料，並指定密碼來加密所有資料及其中繼資料以進行卸載。選擇新增或管理密碼以指定新密碼。 6. 在精靈的封存層步驟中，設定您要連接至橫向擴展備份儲存庫的封存儲存層。（如果您略過新增 Amazon S3 Glacier 儲存體，則不會顯示此步驟。） • 選擇封存 GFS 完整備份至物件儲存。針對物件儲存庫，選擇您在上一個史詩中新增的 Amazon Glacier 儲存體。 • 對於超過 N 天的封存 GFS 備份，請選擇將檔案移至封存範圍的時間範圍。（若要在建立非作用中的備份鏈當天封存，請指定 0 天。）	

任務	Description	所需的技能
	7. 在精靈的摘要步驟中，檢閱向外擴展備份儲存庫的組態，然後選擇完成。	

相關資源

- 在 (IAM 文件） 中建立 IAM 使用者 AWS 帳戶
- 建立儲存貯體 (Amazon S3 文件）
- 封鎖對 Amazon S3 儲存體的公開存取 (Amazon S3 文件）
- 使用 S3 物件鎖定 (Amazon S3 文件）
- Veeam 技術文件
- 如何建立安全 IAM 政策以連線至 S3 物件儲存 (Veeam 文件）

其他資訊

下列各節提供範例 IAM 政策，您可以在此模式的 Epics 區段中建立 IAM 使用者時使用。

容量層的 IAM 政策

> **Note**
>
> 將範例政策中的 S3 儲存貯體名稱從 <yourbucketname> 變更為您要用於 Veeam 容量層備份的 S3 儲存貯體名稱。另請注意，政策應限於用於 Veeam 的特定資源 （以下政策中的Resource規格表示），並且政策的第一部分會停用用戶端加密，如 AWS 部落格文章防止 Amazon S3 物件的意外加密所述。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RestrictSSECObjectUploads",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
```

```
            "Resource": "arn:aws:s3:::<your-bucket-name>/*",
            "Condition": {
                "Null": {
                    "s3:x-amz-server-side-encryption-customer-algorithm": "false"
                }
            }
        },
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:GetObjectVersion",
                "s3:ListBucketVersions",
                "s3:ListBucket",
                "s3:PutObjectLegalHold",
                "s3:GetBucketVersioning",
                "s3:GetObjectLegalHold",
                "s3:GetBucketObjectLockConfiguration",
                "s3:PutObject*",
                "s3:GetObject*",
                "s3:GetEncryptionConfiguration",
                "s3:PutObjectRetention",
                "s3:PutBucketObjectLockConfiguration",
                "s3:DeleteObject*",
                "s3:DeleteObjectVersion",
                "s3:GetBucketLocation"

            ],
            "Resource": [
                "arn:aws:s3:::<yourbucketname>",
                "arn:aws:s3:::<yourbucketname>/*"
            ]
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets",
                "s3:ListBucket"
            ],
            "Resource": "arn:aws:s3:::*"
        }
    ]
}
```

封存層的 IAM 政策

> **Note**
>
> 將範例政策中的 S3 儲存貯體名稱從 <yourbucketname> 變更為您要用於 Veeam 封存層備份的 S3 儲存貯體名稱。

若要使用現有的 VPC、子網路和安全群組：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3Permissions",
            "Effect": "Allow",
            "Action": [
                "s3:DeleteObject",
                "s3:PutObject",
                "s3:GetObject",
                "s3:RestoreObject",
                "s3:ListBucket",
                "s3:AbortMultipartUpload",
                "s3:GetBucketVersioning",
                "s3:ListAllMyBuckets",
                "s3:GetBucketLocation",
                "s3:GetBucketObjectLockConfiguration",
                "s3:PutObjectRetention",
                "s3:GetObjectVersion",
                "s3:PutObjectLegalHold",
                "s3:GetObjectRetention",
                "s3:DeleteObjectVersion",
                "s3:ListBucketVersions"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket-name>",
                "arn:aws:s3:::<bucket-name>/*"
            ]
        }
    ]
}

{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:CreateKeyPair",
                "ec2:DescribeKeyPairs",
                "ec2:RunInstances",
                "ec2:DeleteKeyPair",
                "ec2:DescribeVpcAttribute",
                "ec2:CreateTags",
                "ec2:DescribeSubnets",
                "ec2:TerminateInstances",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeImages",
                "ec2:DescribeVpcs"
            ],
            "Resource": "arn:aws:ec2:<region>:<account-id>:*"
        }
    ]
}
```

若要建立新的 VPC、子網路和安全群組：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3Permissions",
            "Effect": "Allow",
            "Action": [
                "s3:DeleteObject",
                "s3:PutObject",
                "s3:GetObject",
                "s3:RestoreObject",
                "s3:ListBucket",
                "s3:AbortMultipartUpload",
                "s3:GetBucketVersioning",
                "s3:ListAllMyBuckets",
                "s3:GetBucketLocation",
                "s3:GetBucketObjectLockConfiguration",
```

```
                "s3:PutObjectRetention",
                "s3:GetObjectVersion",
                "s3:PutObjectLegalHold",
                "s3:GetObjectRetention",
                "s3:DeleteObjectVersion",
                "s3:ListBucketVersions"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket-name>",
                "arn:aws:s3:::<bucket-name>/*"
            ]
        }
    ]
}

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:CreateKeyPair",
                "ec2:DescribeKeyPairs",
                "ec2:RunInstances",
                "ec2:DeleteKeyPair",
                "ec2:DescribeVpcAttribute",
                "ec2:CreateTags",
                "ec2:DescribeSubnets",
                "ec2:TerminateInstances",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeImages",
                "ec2:DescribeVpcs",
                "ec2:CreateVpc",
                "ec2:CreateSubnet",
                "ec2:DescribeAvailabilityZones",
                "ec2:CreateRoute",
                "ec2:CreateInternetGateway",
                "ec2:AttachInternetGateway",
                "ec2:ModifyVpcAttribute",
                "ec2:CreateSecurityGroup",
                "ec2:DeleteSecurityGroup",
                "ec2:AuthorizeSecurityGroupIngress",
```

```
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:DescribeRouteTables",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        }
    ]
}
```

使用 將資料從 Amazon S3 儲存貯體複製到另一個帳戶和區域 AWS CLI

Appasaheb Bagali 和 Purushotham G K，Amazon Web Services

摘要

此模式說明如何將資料從 AWS 帳戶中的來源 Amazon Simple Storage Service (Amazon S3) 儲存貯體遷移到另一個 AWS 帳戶中的目的地 Amazon S3 儲存貯體，無論是在相同 AWS 區域 或不同區域中。

來源 Amazon S3 儲存貯體允許 AWS Identity and Access Management (IAM) 使用連接的資源政策進行存取。目的地帳戶中的使用者必須擔任具有來源儲存貯體`PutObject` 和 `GetObject`許可的角色。最後，您會執行 `copy`和 `sync`命令，將資料從來源 Amazon S3 儲存貯體傳輸到目的地 Amazon S3 儲存貯體。

帳戶擁有上傳至 Amazon S3 儲存貯體的物件。如果您跨帳戶和區域複製物件，您可以授予所複製物件的目的地帳戶擁有權。您可以將物件的存取控制清單 (ACL) 變更為 ，以變更物件的擁有權`bucket-owner-full-control`。不過，我們建議您將程式設計跨帳戶許可授予目的地帳戶，因為 ACLs可能難以管理多個物件。

> **Warning**
>
> 此案例需要具有程式設計存取和長期登入資料的 IAM 使用者，這會造成安全風險。為了協助降低此風險，建議您只為這些使用者提供執行任務所需的許可，並在不再需要這些使用者時將其移除。必要時可以更新存取金鑰。如需詳細資訊，請參閱 IAM 文件中的更新存取金鑰。

先決條件和限制

先決條件

- 相同或不同 AWS 帳戶 中的兩個作用中 AWS 區域。
- 來源帳戶中現有的 Amazon S3 儲存貯體。
- 如果您的來源或目的地 Amazon S3 儲存貯體已啟用預設加密，您必須修改 AWS Key Management Service (AWS KMS) 金鑰許可。如需詳細資訊，請參閱本主題的 AWS re：Post 文章。
- 熟悉跨帳戶許可。

限制

- 此模式涵蓋一次性遷移。對於需要持續自動將新物件從來源儲存貯體遷移到目的地儲存貯體的情況，您可以使用 Amazon S3 批次複寫。
- 此模式使用暫時性和非持久性的工作階段登入資料 `AccessKeyId`(`SecretAccessKey`、 和 `SessionToken`)。輸出中的過期時間戳記會指出這些登入資料何時過期。角色已設定工作階段持續時間上限。如果工作階段過期，複製任務將會取消。

Architecture

工具

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

最佳實務

- IAM 中的安全最佳實務 (IAM 文件)
- 套用最低權限許可 (IAM 文件)

史詩

在目的地中建立 IAM 使用者和角色 AWS 帳戶

任務	Description	所需的技能
建立 IAM 使用者並取得存取金鑰。	1. 登入 AWS 管理主控台 並建立具有程式設計存取權的 IAM 使用者。如需詳細說明，請參閱 IAM 文件中的建立 IAM 使用者。不需要為此使用者連接任何政策。	AWS DevOps

任務	Description	所需的技能
	2. 為此使用者產生存取金鑰和私密金鑰。如需說明，請參閱 AWS 文件中的 AWS 帳戶 和 存取金鑰。	

任務	Description	所需的技能
建立 IAM 身分型政策。	使用以下許可建立名為 `S3MigrationPolicy` 的 IAM 身分型政策。根據您的使用案例修改來源和目的地儲存貯體名稱。此身分型政策允許擔任此角色的使用者存取來源儲存貯體和目的地儲存貯體。如需詳細說明，請參閱 IAM 文件中的建立 IAM 政策。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "s3:ListBucket", "s3:ListObjectsV2", "s3:GetObject", "s3:GetObjectTagging", "s3:GetObjectVersion", "s3:GetObjectVersionTagging" ], "Resource": [ "arn:aws:s3:::amaz`	AWS DevOps

任務	Description	所需的技能
	on-s3-demo-source-bucket", "arn:aws:s3:::amazon-s3-demo-source-bucket/*"] }, { "Effect": "Allow", "Action": ["s3:ListBucket", "s3:PutObject", "s3:PutObjectAcl", "s3:PutObjectTagging", "s3:GetObjectTagging", "s3:ListObjectsV2", "s3:GetObjectVersion", "s3:GetObjectVersionTagging"], "Resource": ["arn:aws:s3:::amazon-s3-demo-destination-bucket", "arn:aws:s3:::amazon-	

任務	Description	所需的技能

	<pre>s3-demo-destination-bucket/*"
	            ]
	        }
	    ]
	}</pre>	

任務	Description	所需的技能
建立 IAM 角色。	使用以下信任政策建立名為 `S3MigrationRole` 的 IAM 角色。根據您的使用案例修改信任政策中目的地 IAM 角色或使用者名稱的 Amazon Resource Name (ARN)。此信任政策允許新建立的 IAM 使用者擔任 `S3MigrationRole` 。連接先前建立的 `S3MigrationPolicy` 。如需詳細步驟，請參閱《IAM 文件》中的建立角色以將許可委派給 IAM 使用者。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "AWS": "arn:aws:iam::<destination_account>:user/<user_name>" }, "Action": "sts:AssumeRole", "Condition": {} } ] }`	AWS DevOps

在來源帳戶中建立並連接 Amazon S3 儲存貯體政策

任務	Description	所需的技能
建立並連接 Amazon S3 儲存貯體政策。	1. 登入 AWS 管理主控台 來源帳戶的 ，然後開啟 Amazon S3 主控台。 2. 選擇來源 Amazon S3 儲存貯體，然後選擇許可。 3. 在 Bucket policy (儲存貯體政策) 下方，選擇 Edit (編輯)。 4. 貼上下列儲存貯體政策。請務必包含目的地帳戶的 AWS 帳戶 ID，並根據您的需求設定儲存貯體政策範本。此資源型政策允許目的地角色`S3MigrationRole` 存取來源帳戶中的 Amazon S3 物件。 `{ "Version": "2012-10-17", "Statement": [ { "Sid": "DelegateS3Access", "Effect": "Allow", "Principal": {"AWS": "arn:aws:iam::<destination_account>:role/<RoleName>"}, "Action": ["s3:ListBucket", "s3:GetObject",`	雲端管理員

任務	Description	所需的技能

	<pre>"s3:ListObjectsV2",
	"s3:GetObjectTagging",
	"s3:GetObjectVersion",
	"s3:GetObjectVersionTagging"
	],
	"Resource": [
	"arn:aws:s3:::amazon-s3-demo-source-bucket/*",
	"arn:aws:s3:::amazon-s3-demo-source-bucket"
	]
	}
	]
	}</pre> 5. 選擇儲存。	

設定目的地 Amazon S3 儲存貯體

任務	Description	所需的技能
建立目的地 Amazon S3 儲存貯體。	1. 登入 AWS 管理主控台 目的地帳戶的 ，然後開啟 Amazon S3 主控台。 2. 選擇建立儲存貯體。	雲端管理員

任務	Description	所需的技能
	3. 根據您的需求建立 Amazon S3 儲存貯體。如需詳細資訊，請參閱 Amazon S3 文件中的建立儲存貯體。	

將資料複製到目的地 Amazon S3 儲存貯體

任務	Description	所需的技能
AWS CLI 使用新建立的使用者登入資料設定 。	1. 安裝最新版本的 AWS CLI。如需說明，請參閱 AWS CLI 文件中的安裝或更新 的最新版本 AWS CLI。 2. AWS CLI 使用您建立之 IAM 使用者的 AWS 存取金鑰執行`$ aws configure` 和更新。如需詳細資訊，請參閱 AWS CLI 文件中的組態和登入資料檔案設定。	AWS DevOps
擔任 Amazon S3 遷移角色。	1. 使用 AWS CLI 來假設 `S3MigrationRole` ： `aws sts assume-role \ --role-arn "arn:aws:iam::<destination_account>:role/S3MigrationRole" \ --role-session-name AWSCLI-Session`	AWS 管理員

任務	Description	所需的技能
	Note 此命令會輸出數個資訊片段。在登入資料區塊中，您需要 `AccessKeyId` 、 `SecretAccessKey` 和 `SessionToken` 。此範例使用環境變數 `RoleAccessKeyID` 、 `RoleSecretKey` 和 `RoleSessionToken` 。：工作階段登入資料 (`AccessKeyId` 、 `SecretAccessKey` 和 `SessionToken`) 是暫時性和非持久性。輸出中的過期時間戳記會指出這些登入資料何時過期。角色已設定工作階段持續時間上限。如果登入資料過期，您必須`sts:AssumeRole` 再次呼叫 以取得新的臨時登入資料。	

任務	Description	所需的技能

	2. 建立三個環境變數以擔任 IAM 角色。這些環境變數會以下列輸出完成：
	
	`# Linux`
	`export AWS_ACCES`
	`S_KEY_ID=<RoleAcce`
	`ssKeyID from command`
	`output>`
	`export AWS_SECRE`
	`T_ACCESS_KEY=<Role`
	`SecretKey from`
	`command output>`
	`export AWS_SESSI`
	`ON_TOKEN=<RoleSess`
	`ionToken from command`
	`output>`
	`# Windows`
	`set AWS_ACCES`
	`S_KEY_ID=<RoleAcce`
	`ssKeyID from command`
	`output>`
	`set AWS_SECRE`
	`T_ACCESS_KEY=<Role`
	`SecretKey from`
	`command output>`
	`set AWS_SESSI`
	`ON_TOKEN=<RoleSess`
	`ionToken from command`
	`output>`
	
	3. 執行下列命令，確認您已擔任 IAM 角色：
	
	`aws sts get-caller-identity`
	
	如需詳細資訊，請參閱如何使用 AWS CLI 擔任 IAM 角色？	

任務	Description	所需的技能

將來源儲存貯體中的資料複製並同步到目的地儲存貯體。	當您擔任角色時S3MigrationRole ，您可以使用複製 (cp) 或同步 (sync) 命令來複製資料。
	
	複製：
	
	`aws s3 cp s3://amazon-s3-demo-source-bucket/ \ s3://amazon-s3-demo-destination-bucket/ \ --recursive --source-region SOURCE-REGION-NAME --region DESTINATION-REGION-NAME`
	
	同步：
	
	`aws s3 sync s3://amazon-s3-demo-source-bucket/ \ s3://amazon-s3-demo-destination-bucket/ \ --source-region SOURCE-REGION-NAME --region DESTINATION-REGION-NAME`
		雲端管理員

疑難排解

問題	解決方案
呼叫 `ListObjects` 操作時發生錯誤 (`AccessDenied`)	1. 請確定您已擔任角色 `S3MigrationRole` 。 2. 執行 `aws sts get-caller-identity` 以檢查使用的角色。如果輸出未顯示 的 ARN`S3MigrationRole` ，請再次擔任 IAM 角色，然後重試。

相關資源

- 建立 Amazon S3 儲存貯體 (Amazon S3 文件)
- Amazon S3 儲存貯體政策和使用者政策 (Amazon S3 文件)
- IAM 身分 （使用者、群組和角色） (IAM 文件)
- cp 命令 (AWS CLI 文件)
- sync 命令 (AWS CLI 文件)

在 IBM Db2 資料庫中啟用直接封存至 Amazon S3 的 DB2 日誌 Amazon S3 Db2

Ambarish Satarkar，Amazon Web Services

總結

此模式說明如何使用 Amazon Simple Storage Service (Amazon S3) 做為 IBM Db2 產生的封存日誌的目錄儲存，而不使用預備區域。

您可以為 logarchmeth1 和 logarchmeth2 log archive 方法組態參數指定 DB2REMOTE Amazon S3 儲存體。您可以使用 `logarchmeth1` 參數來指定從目前日誌路徑封存之日誌的主要目的地。使用此功能，您可以直接從 Amazon S3 封存和擷取交易日誌，而無需使用暫存區域。

Amazon S3 會將上傳至其中的資料存放在單一裝置中至少三個裝置 AWS 區域。考量到企業備份的高可用性、彈性儲存選項、生命週期政策和安全性，所有規模和產業的數百萬客戶都使用 Amazon S3 來儲存企業備份。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上執行的 IBM Db2 資料庫。Amazon EC2
- AWS Command Line Interface (AWS CLI) 已安裝
- libcurl 和 libxml2 安裝在 Db2 EC2 執行個體上。

限制

- 只有 Db2 11.5.7 或更新版本允許將日誌直接封存至 Amazon S3 儲存體。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。
- 在所有組態中，Amazon S3 存在下列限制：
  - AWS Key Management Service 不支援 (AWS KMS)。
  - AWS 不支援角色型 (AWS Identity and Access Management (IAM)) 或字符型 (AWS Security Token Service (AWS STS)) 登入資料。

產品版本

- AWS CLI 第 2 版或更新版本
- IBM Db2 11.5.7 或更新版本
- Linux SUSE Linux Enterprise Server (SLES) 11 或更新版本
- Red Hat Enterprise Linux (RHEL) 6 或更新版本
- Windows Server 2008 R2、2012 (R2)、2016 或 2019

Architecture

下圖顯示此模式的元件和工作流程。

上的架構 AWS 雲端 包括下列項目：

- 虛擬私有雲端 (VPC) – AWS 雲端 您啟動資源之 的邏輯隔離區段。
- 可用區域 – 在 內隔離的資料中心執行 Db2 LUW (Linux、Unix、Windows) 工作負載，以提供高可用性 AWS 區域。
- 公有子網路 – 透過 NAT 閘道為管理員和網際網路連線提供 RDP （遠端桌面通訊協定） 存取。
- 私有子網路 – 託管 Db2 LUW 資料庫。Db2 LUW 執行個體是使用 LOGARCHMETH1 參數設定。參數會透過閘道端點將資料庫日誌封存檔案直接寫入 Amazon S3 路徑。

以下 AWS 服務 提供支援：

- Amazon S3 – 做為 Db2 日誌封存檔案的耐用、可擴展儲存位置。
- Amazon Elastic File System (Amazon EFS) – 提供 Db2 可用於資料庫備份和預備的共用、全受管檔案系統。Db2 也可以在日誌檔案封存至 Amazon S3 之前，使用 Amazon EFS 做為日誌檔案的掛載點。
- Amazon CloudWatch – 從 Db2 和基礎 EC2 執行個體收集和監控指標、日誌和事件。您可以使用 CloudWatch 建立警示、儀表板和自動回應效能或可用性問題。

自動化和擴展

- 此模式提供全自動化解決方案來存放 Db2 日誌封存備份。
- 您可以使用相同的 Amazon S3 儲存貯體來啟用多個 Db2 資料庫的日誌封存。

工具

AWS 服務

- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您執行的應用程式。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon Elastic File System (Amazon EFS) 協助您在 AWS 雲端中建立和設定共用檔案系統。
- AWS IAM Identity Center 可協助您集中管理所有 AWS 帳戶 和雲端應用程式的單一登入 (SSO) 存取。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。

其他工具

- libcurl 是免費的用戶端 URL 傳輸程式庫。
- libxml2 是免費的 XML C 剖析器和工具組。

最佳實務

- 遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。

史詩

設定 AWS 服務

任務	Description	所需的技能
設定 AWS CLI。	若要下載並安裝 AWS CLI，請使用下列命令：	AWS 系統管理員、AWS 管理員

任務	Description	所需的技能
	i) curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip" ii) unzip awscliv2.zip iii) sudo ./aws/install	
設定 AWS CLI。	若要設定 AWS CLI，請使用下列命令： $ aws configure AWS Access Key ID [None]:********** ********** AWS Secret Access Key [None]: ********* ********** Default region name [None]: < aws region > Default output format [None]: text	AWS 系統管理員、AWS 管理員

任務	Description	所需的技能

建立 IAM 使用者。	若要建立 IAM 使用者以稍後用於與 Amazon S3 的 Db2 資料庫連線，請使用下列命令：
	
	`aws iam create-user --user-name <unique username>`
	
	以下是 命令的範例：
	
	`aws iam create-user --user-name db_backup_user`
	
	Warning
	此種情況需要具備程式化存取權限且擁有長期憑證的 IAM 使用者，此舉將構成安全風險。為了降低此風險，建議您只為這些使用者提供執行任務所需的許可，並在不再需要這些使用者時將其移除。必要時可以更新存取金鑰。如需詳細資訊，請參閱 IAM 文件中的AWS 安全登入資料和管理 IAM 使用者的存取金鑰。 https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_acc
		AWS 系統管理員

任務	Description	所需的技能
	ess-keys.html#Using_RotateAccessKey	

建立 Amazon S3 儲存貯體。	若要建立用於存放資料庫備份的 Amazon S3 儲存貯體，請使用下列命令：
	
	`aws s3api create-bucket --bucket <unique bucket name> --region <aws region>`
	
	下列是 命令範例。
	
	`aws s3api create-bucket --bucket myfirstbucket --region af-south-1`
		AWS 系統管理員

任務	Description	所需的技能
授權 IAM 使用者。	若要授權新建立的 IAM 使用者擁有 Amazon S3 許可，請使用下列步驟： 1. 若要db2bucket.json 為 Amazon S3 儲存貯體產生新的 IAM 政策檔案，請使用下列命令： { "Version": "2012-10-17", "Statement": [{ "Sid": "ListObjectsInBucket", "Effect": "Allow", "Action": "s3:ListBucket", "Resource": ["arn:aws:s3:::myfirstbucket"] }, { "Sid": "AllObjectActions", "Effect": "Allow", "Action": ["s3:PutObject", "s3:GetObject",	AWS 系統管理員、AWS 管理員

任務	Description	所需的技能

	<pre>"s3:DeleteObject"
	            ],
	            "Resource
	": [
	
	"arn:aws:s3:::myfi
	rstbucket/*"
	            ]
	        }
	    ]
	}</pre>	

	2. 根據 db2bucket .json 檔案建立 IAM 政策。JSON 輸出會顯示政策的 Amazon Resource Name (ARN)：`"Arn": "arn:aws:iam::111122223333:policy/db2s3policy"` 。
	<pre>aws iam create-policy
	\
	      --policy-name
	db2s3policy \
	      --policy-
	document file://db
	2bucket.json</pre>	

	3. 使用政策 ARN 將 IAM 政策連接至 IAM 使用者。
	<pre>aws iam attach-user-
	policy --user-name
	db_backup_user --
	policy-arn "arn:aws:
	iam::111122223333:</pre>	

任務	Description	所需的技能
	`policy/db2s3policy"`	

任務	Description	所需的技能

建立存取金鑰。	若要產生存取金鑰，以程式設計方式從 DB2 執行個體存取 Amazon S3，請使用下列命令：
	
	`aws iam create-access-key --user-name <username>`
	
	以下是 命令的範例：
	
	`aws iam create-access-key --user-name db_backup_user`
	
	⚠ Warning
	此種情況需要具備程式化存取權限且擁有長期憑證的 IAM 使用者，此舉將構成安全風險。為了降低此風險，建議您只為這些使用者提供執行任務所需的許可，並在不再需要這些使用者時將其移除。必要時可以更新存取金鑰。如需詳細資訊，請參閱 IAM 文件中的AWS 安全登入資料和管理 IAM 使用者的存取金鑰。 https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_acc
		AWS 系統管理員

任務	Description	所需的技能
	ess-keys.html#Using_RotateAccessKey	
建立 PKCS 金鑰存放區。	若要建立 PKCS 金鑰存放區以存放金鑰，並建立私密存取金鑰以將資料傳輸至 Amazon S3，請使用下列命令： `gsk8capicmd_64 -keydb -create -db "/db2/db2<sid>/.keystore/db6-s3.p12" -pw "<password>" -type pkcs12 -stash`	AWS 系統管理員
設定 DB2 以使用金鑰存放區。	若要將 DB2 設定為搭配 `keystore_location` 和 `keystore_type` 參數使用金鑰存放區，請使用下列命令： `db2 "update dbm cfg using keystore_location /db2/db2<sid>/.keystore/db6-s3.p12 keystore_type pkcs12"`	AWS 系統管理員

任務	Description	所需的技能

建立 DB2 儲存體存取別名。	儲存存取別名會指定要使用的 Amazon S3 儲存貯體。它也提供連線詳細資訊，例如以加密格式存放在本機金鑰存放區的使用者名稱和密碼。如需詳細資訊，請參閱 IBM Db2 文件中的 CATALOG STORAGE ACCESS 命令。
	
	若要建立儲存存取別名，請使用下列語法：
	
	`db2 "catalog storage access alias <alias_name> vendor S3 server <S3 endpoint> user '<access_key>' password '<secret_access_key>' container '<bucket_name>'"`
	
	以下是範例：
	
	`db2 "catalog storage access alias DB2BKPS3 vendor S3 server s3.us-west-2.amazonaws.com user '**********' password '**********' container 'myfirstbucket'"`
		AWS 系統管理員

更新DB2 中的 logarchmeth1 位置並重新啟動DB2

任務	Description	所需的技能

更新LOGARCHMETH1 位置。	若要使用您先前定義的儲存存取別名，請更新LOGARCHMETH1 資料庫參數，請使用下列命令：
	`db2 update db cfg for <DBNAME> using LOGARCHMETH1 'DB2REMOTE://<storage_alias_name>//<sub folder>'`
	若要將日誌與其他檔案分開，請指定要將日誌儲存在 S3 儲存貯體中的子目錄（即 Amazon S3 儲存貯體字首）。TESTDB_LOGS
	以下是範例：
	`db2 update db cfg for ABC using LOGARCHMETH1 'DB2REMOTE://DB2BKPS3//TESTDB_LOGS/'`
	您應該會看到下列訊息：DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
		AWS 系統管理員

重新啟動 DB2。	重新設定 DB2 執行個體以進行日誌封存後，請重新啟動該執行個體。	AWS 管理員、AWS 系統管理員

任務	Description	所需的技能
	不過，如果 LOGARCHMETH1 先前已設定為任何檔案系統位置，則不需要重新啟動。	

檢查 Amazon S3 和 db2diag.log 中的封存日誌路徑

任務	Description	所需的技能
檢查 Amazon S3 中的封存日誌。	此時，您的資料庫已完全設定為將交易日誌直接封存至 Amazon S3 儲存體。若要確認組態，請開始在資料庫上執行交易活動，以開始耗用（和封存）日誌空間。然後，檢查 Amazon S3 中的封存日誌。	AWS 管理員、AWS 系統管理員
檢查 中的封存日誌組態db2diag.log 。	在 Amazon S3 中檢查封存日誌後，請在 DB2 診斷日誌中尋找下列訊息db2diag.log ： `MESSAGE : ADM1846I Completed archive for log file "S0000079.LOG" to Completed archive for log file S0000080.LOG to DB2REMOTE://<AWS S3 Bucket Name>/<SID>/log1/db2<sid>/<SID>/NODE0000/LOGSTREAM0000/C0000001/ from /db2/<SID>/log_dir/NODE0000/LOGSTREAM0000/. M`	AWS 系統管理員

任務	Description	所需的技能
	`ESSAGE : ADM1846I Completed archive for log file "S0000080 .LOG" to Comple ted archive for log file S0000081.LOG to DB2REMOTE://<AWS S3 Bucket Name> / <SID>/log1/db2<si d>/<SID>/NODE0000/ LOGSTREAM0000/C000 0001/ from /db2/<SID >/log_dir/NODE0000/ LOGSTREAM0000/.` 此訊息確認已關閉的 DB2 交易日誌檔案正在封存至（遠端）Amazon S3 儲存體。	

相關資源

AWS 服務文件

- AWS 安全登入資料 (IAM 文件)
- 授予最低權限 (IAM 文件)
- 管理 IAM 使用者的存取金鑰 (IAM 文件)
- IAM 中的安全最佳實務 (IAM 文件)

IBM 資源

- IBM Db2 資料庫
- logarchmeth1 - 主要日誌封存方法組態參數
- logarchmeth2 - 次要日誌封存方法組態參數
- 遠端儲存

使用 DistCp 搭配適用於 Amazon S3 的 AWS PrivateLink，將資料從內部部署 Hadoop 環境遷移至 Amazon S3

Jason Owens、Andres Cantor、Jeff Klopfenstein、Bruno Rocha Oliveira 和 Samuel Schmidt，Amazon Web Services

總結

此模式示範如何使用 Apache 開放原始碼工具 DistCp 搭配 AWS PrivateLink for Amazon Simple Storage Service (Amazon S3)，將幾乎任何數量的資料從現場部署 Apache Hadoop 環境遷移至 Amazon Web Services (AWS) 雲端。您可以使用適用於 Amazon S3 的 AWS PrivateLink，透過內部部署資料中心和 Amazon Virtual Private Cloud (Amazon VPC) 之間的私有網路連線將資料遷移至 Amazon S3，而不是使用公有網際網路或代理解決方案來遷移資料。 Amazon Virtual Private Cloud 如果您在 Amazon Route 53 中使用 DNS 項目，或在內部部署 Hadoop 叢集的所有節點的 /etc/hosts 檔案中新增項目，則會自動將您導向正確的介面端點。

本指南提供使用 DistCp 將資料遷移至 AWS 雲端的說明。DistCp 是最常用的工具，但還有其他遷移工具可用。例如，您可以使用離線 AWS 工具，例如 AWS Snowball 或 AWS Snowmobile，或線上 AWS 工具，例如 AWS Storage Gateway 或 AWS DataSync。此外，您可以使用其他開放原始碼工具，例如 Apache NiFi。

先決條件和限制

先決條件

- 在內部部署資料中心與 AWS 雲端之間具有私有網路連線的作用中 AWS 帳戶
- Hadoop，使用 DistCp 安裝在內部部署
- 可存取 Hadoop 分散式檔案系統 (HDFS) 中遷移資料的 Hadoop 使用者
- AWS Command Line Interface (AWS CLI)，已安裝並設定
- 將物件放入 S3 儲存貯體的許可

限制

虛擬私有雲端 (VPC) 限制適用於 Amazon S3 的 AWS PrivateLink。如需詳細資訊，請參閱介面端點屬性和限制以及 AWS PrivateLink 配額 (AWS PrivateLink 文件）。

Amazon S3 的 AWS PrivateLink 不支援下列項目：

- 聯邦資訊處理標準 (FIPS) 端點

- 網站端點
- 舊版全域端點

Architecture

來源技術堆疊

- 安裝 DistCp 的 Hadoop 叢集

目標技術堆疊

- Amazon S3
- Amazon VPC

目標架構

下圖顯示 Hadoop 管理員如何使用 DistCp，透過私有網路連線，例如 AWS Direct Connect，透過 Amazon S3 介面端點將資料從內部部署環境複製到 Amazon S3。

工具

AWS 服務

- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路類似於您在自己的資料中心內操作的傳統網路，具有使用可擴展的 AWS 基礎設施的優勢。

其他工具

- Apache Hadoop DistCp （分散式複本）是一種用於複製大型叢集間和叢集內的工具。DistCp 使用 Apache MapReduce 進行分佈、錯誤處理和復原，以及報告。

史詩

將資料遷移至 AWS 雲端

任務	Description	所需的技能
為 Amazon S3 的 AWS PrivateLink 建立端點。	1. 登入 AWS 管理主控台並開啟 Amazon VPC 主控台。 2. 在導覽窗格中，選擇端點，然後選擇建立端點。 3. 在 Service category (服務類別) 中，選擇 AWS services (AWS 服務)。 4. 在搜尋方塊中，輸入 s3，然後按 Enter。 5. 在搜尋結果中，選擇 com.amazonaws.<your-aws-region>.s3 服務名稱，其中 Type 欄中的值為 Interface。 6. 在 VPC 中，選擇您的 VPC。針對子網路，選擇您的子網路。 7. 針對安全群組，選擇或建立允許 TCP 443 的安全群組。 8. 根據您的需求新增標籤，然後選擇建立端點。	AWS 管理員
驗證端點並尋找 DNS 項目。	1. 開啟 Amazon VPC 主控台，選擇端點，然後選取您先前建立的端點。 2. 在詳細資訊索引標籤上，尋找 DNS 名稱的第一個 DNS 項目。這是區域 DNS 項目。當您使用此 DNS 名稱	AWS 管理員

任務	Description	所需的技能
	時，會在可用區域特定的 DNS 項目之間請求替換。 3. 選擇子網路索引標籤。您可以在每個可用區域中找到端點彈性網路界面的地址。	

任務	Description	所需的技能

檢查防火牆規則和路由組態。	若要確認您的防火牆規則已開啟且您的聯網組態已正確設定，請使用 Telnet 在連接埠 443 上測試端點。例如：
	
	`$ telnet vpce-<your-VPC-endpoint-ID>.s3.us-east-2.vpce.amazonaws.com 443`
	
	`Trying 10.104.88.6...`
	
	`Connected to vpce-<your-VPC-endpoint-ID>.s3.us-east-2.vpce.amazonaws.com.`
	
	`...`
	
	`$ telnet vpce-<your-VPC-endpoint-ID>.s3.us-east-2.vpce.amazonaws.com 443`
	
	`Trying 10.104.71.141...`
	
	`Connected to vpce-<your-VPC-endpoint-ID>.s3.us-east-2.vpce.amazonaws.com.`
	
	Note
	如果您使用區域項目，則成功測試會顯示 DNS 正在交替兩個 IP 地址，您可以在 Amazon VPC 主控台
		網路管理員、AWS 管理員

任務	Description	所需的技能
	中所選端點的子網路索引標籤上看到這些地址。	

任務	Description	所需的技能
設定名稱解析。	您必須設定名稱解析，以允許 Hadoop 存取 Amazon S3 介面端點。您無法使用端點名稱本身。反之，您必須解析 `<your-bucket-name>.s3.<your-aws-region>.amazonaws.com` 或 `*.s3.<your-aws-region>.amazonaws.com`。如需此命名限制的詳細資訊，請參閱 Hadoop S3A 用戶端簡介 (Hadoop 網站）。 選擇下列其中一個組態選項： • 使用內部部署 DNS 來解析端點的私有 IP 地址。您可以覆寫所有儲存貯體或所選儲存貯體的行為。如需詳細資訊，請參閱 AWS PrivateLink 安全混合存取 Amazon S3 中的「選項 2：使用網域名稱系統回應政策區域 (DNS RPZ) 存取 Amazon S3」(AWS 部落格文章）。Amazon S3 PrivateLink • 設定內部部署 DNS，以有條件地將流量轉送至 VPC 中的解析程式傳入端點。流量會轉送至 Route 53。如需詳細資訊，請參閱 AWS PrivateLink 安全混合存取中的「選項 3：使用 Amazon Route 53 Resolver 傳入端點從內部部署轉送 DNS 請	AWS 管理員

任務	Description	所需的技能
	求」(AWS 部落格文章）。Amazon S3 PrivateLink • 編輯 Hadoop 叢集中所有節點上的 /etc/hosts 檔案。這是用於測試的暫時性解決方案，不建議用於生產。若要編輯 /etc/hosts 檔案，請新增 `<your-bucket-name>.s3.<your-aws-region>.amazonaws.com` 或的項目`s3.<your-aws-region>.amazonaws.com` 。/etc/hosts 檔案不能有項目的多個 IP 地址。您必須從其中一個可用區域選擇單一 IP 地址，這會變成單一失敗點。	

任務	Description	所需的技能

設定 Amazon S3 的身分驗證。	若要透過 Hadoop 驗證 Amazon S3，建議您將暫時角色登入資料匯出至 Hadoop 環境。如需詳細資訊，請參閱使用 S3 驗證 (Hadoop 網站）。對於長時間執行的任務，您可以建立使用者，並指派具有將資料僅放入 S3 儲存貯體之許可的政策。存取金鑰和私密金鑰可以存放在 Hadoop 上，僅供 DistCp 任務本身和 Hadoop 管理員存取。如需存放秘密的詳細資訊，請參閱使用 Hadoop 登入資料提供者存放秘密 (Hadoop 網站）。如需其他身分驗證方法的詳細資訊，請參閱 AWS IAM Identity Center (AWS Single Sign-On 的後續版本）文件中的如何取得 IAM 角色的登入資料，以搭配 CLI 存取 AWS 帳戶。 AWS Single Sign-On
	
	若要使用臨時登入資料，請將臨時登入資料新增至您的登入資料檔案，或執行下列命令將登入資料匯出至您的環境：
	
	`export AWS_SESSION_TOKEN=SECRET-SESSION-TOKEN`
	`export AWS_ACCESS_KEY_ID=SESSION-ACCESS-KEY`
		AWS 管理員

任務	Description	所需的技能

	`export AWS_SECRE T_ACCESS_KEY=SESSION-SECRET-KEY`
	
	如果您有傳統的存取金鑰和私密金鑰組合，請執行下列命令：
	
	`export AWS_ACCES S_KEY_ID=my.aws.key export AWS_SECRE T_ACCESS_KEY=my.se cret.key`
	
	Note
	如果您使用存取金鑰和私密金鑰組合，請將 DistCp 命令中的登入資料提供者從變更為 `"org.apac he.hadoop .fs.s3a.T emporaryA WSCredent ialsProvi der" "org.apac he.hadoop .fs.s3a.S impleAWSC redential sProvider"` 。	

任務	Description	所需的技能
使用 DistCp 傳輸資料。	若要使用 DistCp 傳輸資料，請執行下列命令： `hadoop distcp -Dfs.s3a.aws.credentials.provider=\"org.apache.hadoop.fs.s3a.TemporaryAWSCredentialsProvider" \ -Dfs.s3a.access.key="${AWS_ACCESS_KEY_ID}" \ -Dfs.s3a.secret.key="${AWS_SECRET_ACCESS_KEY}" \ -Dfs.s3a.session.token="${AWS_SESSION_TOKEN}" \ -Dfs.s3a.path.style.access=true \ -Dfs.s3a.connection.ssl.enabled=true \ -Dfs.s3a.endpoint=s3.<your-aws-region>.amazonaws.com \ hdfs:///user/root/ s3a://<your-bucket-name>` **Note** 當您搭配 Amazon S3 的 AWS PrivateLink 使用 DistCp 命令時，不會自動探索端點的 AWS 區域。Hadoop 3.3.2 和更新版本透過	遷移工程師、AWS 管理員

任務	Description	所需的技能
	啟用 選項明確設定 S3 儲存貯體的 AWS 區域來解決此問題。如需詳細資訊，請參閱 S3A 以新增選項 fs.s3a.endpoint.region 以設定 AWS 區域 (Hadoop 網站）。 如需其他 S3A 提供者的詳細資訊，請參閱一般 S3A 用戶端組態 (Hadoop 網站）。例如，如果您使用加密，您可以根據您的加密類型，將下列選項新增至上述一系列命令： `-Dfs.s3a.server-side-encryption-algorithm=AES-256 [or SSE-C or SSE-KMS]` **Note** 若要搭配 S3A 使用介面端點，您必須為介面端點的 S3 區域名稱（例如 `s3.<your-aws-region>.amazonaws.com`）建立 DNS 別名項目。如需說明，請參閱設定 Amazon S3 的身分驗證一節。Hadoop 3.3.2 和	

任務	Description	所需的技能
	舊版需要此解決方法。未來的 S3A 版本不需要此解決方法。 如果您有 Amazon S3 的簽章問題，請新增選項以使用簽章版本 4 (SigV4) 簽署： `-Dmapreduce.map.java.opts="-Dcom.amazonaws.services.s3.enableV4=true"`	

更多模式

- 安裝 以 AWS 服務 從 IBM z/OS 存取 AWS CLI
- 使用 DynamoDB TTL 自動將項目封存至 Amazon S3
- 使用 Systems Manager 和 EventBridge 自動備份 SAP HANA 資料庫
- 使用 BMC AMI Cloud Data 將大型主機資料備份並封存至 Amazon S3
- 建置 ETL 服務管道，使用 AWS Glue 從 Amazon S3 遞增載入資料至 Amazon Redshift
- 使用 在 Amazon Bedrock 中設定模型調用記錄 AWS CloudFormation
- 使用 Python 將 EBCDIC 資料轉換為 AWS 上的 ASCII
- 將 Oracle 的 VARCHAR2(1) 資料類型轉換為 Amazon Aurora PostgreSQL 的布林資料類型
- 使用 Amazon EFS 在 EC2 執行個體上建立 Amazon ECS 任務定義並掛載檔案系統
- 使用 Kinesis Data Streams 和 Firehose 搭配 將 DynamoDB 記錄交付至 Amazon S3 AWS CDK
- 使用 Terraform 和 DRA 部署 Lustre 檔案系統以進行高效能資料處理
- Amazon DynamoDB 資料表的預估儲存成本
- 使用 Security Hub CSPM 識別 中的公 AWS Organizations 有 Amazon S3 儲存貯體
- AWS 使用 將內部部署 SFTP 伺服器遷移至 AWS Transfer for SFTP
- 使用 AWS DMS 將 Oracle 分割的資料表遷移至 PostgreSQL
- 使用 Rclone 將資料從 Microsoft Azure Blob 遷移至 Amazon S3
- 將 Oracle CLOB 值遷移至 AWS 上的 PostgreSQL 中的個別資料列
- 在 AWS 大型遷移中遷移共用檔案系統
- 使用 將小型資料集從內部部署遷移至 Amazon S3 AWS SFTP
- 在沒有加密的情況下監控 Amazon Aurora 是否有執行個體
- 使用 Transfer 系列將大型主機檔案直接移至 Amazon S3
- 使用 AWS Lambda 自動化 AWS 帳戶 從 移除相同 中的 Amazon EC2 AWS Managed Microsoft AD 項目
- 使用 Amazon EFS on Amazon EKS 搭配 AWS Fargate，以持久性資料儲存來執行具狀態工作負載
- 成功將 S3 儲存貯體匯入為 AWS CloudFormation 堆疊
- 使用 AWS DataSync 同步不同 AWS 區域中 Amazon EFS 檔案系統之間的資料 DataSync
- 檢視 AWS 帳戶或組織的 EBS 快照詳細資訊

開發人員工具

主題

- DevOps
- 基礎設施
- Web 和行動應用程式

DevOps

主題

- 使用後台和自助式 Amazon SageMaker AI 範本加速 MLOps
- 使用 Amazon Bedrock 自動化 AWS 基礎設施操作
- 使用 Terraform 自動化負載平衡器端點變更時的 CloudFront 更新
- 使用 GitHub 動作自動化 AWS CDK Python 應用程式的 Amazon CodeGuru 檢閱
- 使用 GitHub Actions、Artifactory 和 Terraform 在多儲存庫設定中自動化 AWS Supply Chain 資料湖的部署
- 自動化 AWS 資源評估
- 使用開放原始碼工具自動安裝 SAP 系統
- 使用 AWS CDK 自動化 AWS Service Catalog 產品組合和產品部署
- 使用 AWS Service Catalog 和 自動化動態管道管理，以在 Gitflow 環境中部署 Hotfix 解決方案 AWS CodePipeline
- 自動化刪除 AWS CloudFormation 堆疊和相關聯的資源
- 使用 Terraform 在 Amazon Managed Grafana 上自動化 Amazon MWAA 自訂指標的擷取和視覺化
- 使用 AWS CodePipeline 和 AWS CodeBuild 自動化堆疊集部署
- 使用 Cloud Custodian 和 AWS CDK 將 Systems Manager 的 AWS 受管政策自動連接至 EC2 執行個體設定檔
- 使用 AWS CDK 自動為微服務建置 CI/CD 管道和 Amazon ECS 叢集
- 使用 GitHub 動作和 Terraform 建置 Docker 映像並將其推送至 Amazon ECR
- 使用 AWS CodeCommit、AWS CodePipeline 和 AWS Device Farm 建置和測試 iOS 應用程式
- 為在 Amazon EKS 上執行的應用程式設定交互 TLS 身分驗證
- 使用 自動建立 Amazon WorkSpaces 應用程式資源 AWS CloudFormation
- 使用 Firelens 日誌路由器為 Amazon ECS 建立自訂日誌剖析器
- 使用 GitHub Actions 和 Terragrunt 建立 API 驅動的資源協同運作架構
- 使用 GitHub Actions 建立 Terraform 受管 AWS 基礎設施的自動提取請求
- 自動為 Java 和 Python 專案建立動態 CI 管道
- 使用 Terraform 部署 CloudWatch Synthetics Canary
- 在聊天應用程式自訂動作和 中使用 Amazon Q Developer 部署 ChatOps 解決方案來管理 SAST 掃描結果 CloudFormation

- 使用 Terraform 在 Amazon Bedrock 上部署代理程式系統，搭配CrewAI 架構
- 使用 AWS CodePipeline CI/CD 管道部署 AWS Glue 任務 AWS CodePipeline
- 使用 AWS CodePipeline、AWS CodeCommit 和 AWS CodeBuild 在多個 AWS 區域中部署程式碼
- 將工作負載從 Azure DevOps 管道部署到私有 Amazon EKS 叢集
- 使用 Terraform 執行 Amazon Redshift SQL 查詢
- 從 AWS Organizations 中的整個組織將 AWS Backup 報告匯出為 CSV 檔案 AWS Organizations
- 將 Amazon EC2 執行個體清單的標籤匯出至 CSV 檔案
- 使用 Troposphere 產生包含 AWS Config 受管規則的 AWS CloudFormation 範本
- 讓 SageMaker 筆記本執行個體暫時存取另一個 AWS 帳戶中的 CodeCommit 儲存庫
- 為多帳戶 DevOps 環境實作 GitHub 流程分支策略
- 為多帳戶 DevOps 環境實作 Gitflow 分支策略
- 實作多帳戶 DevOps 環境的主體分支策略
- 實作集中式自訂 Checkov 掃描，以在部署 AWS 基礎設施之前強制執行政策
- 使用 K8sGPT 和 Amazon Bedrock 整合實作採用 AI 技術的 Kubernetes 診斷和故障診斷
- 在 CodeCommit 中自動偵測變更並啟動單一儲存庫的不同 CodePipeline 管道 CodeCommit
- 使用 AWS CloudFormation 將 Bitbucket 儲存庫與 AWS Amplify 整合
- 使用 Step Functions 和 Lambda 代理函數跨 AWS 帳戶啟動 CodeBuild 專案
- 使用應用程式復原控制器管理 EMR 叢集的多可用區域容錯移轉
- 使用 AWS 程式碼服務和 AWS KMS 多區域金鑰，管理將微服務部署至多個帳戶和區域的藍/綠部署
- 使用 AWS CloudFormation 和 AWS Config 監控 Amazon ECR 儲存庫是否有萬用字元許可
- 使用 AWS CDK 和 GitHub Actions 工作流程最佳化多帳戶無伺服器部署
- 使用 GitHub 動作根據 AWS CloudFormation 範本佈建 AWS Service Catalog 產品
- 透過部署角色販賣機解決方案來佈建最低權限的 IAM 角色
- 將 Amazon CloudWatch 指標發佈至 CSV 檔案
- 使用 AWS Lambda 自動化 AWS 帳戶 AWS Managed Microsoft AD 從 移除 Amazon EC2 項目
- 使用 AWS Lambda 自動化 AWS 帳戶 從 移除相同 中的 Amazon EC2 AWS Managed Microsoft AD 項目
- AWS Glue 使用 pytest 架構在 中執行 Python ETL 任務的單元測試
- 使用 AWS CodePipeline 和 AWS CDK 設定 CI/CD 管道
- 使用 Terraform 設定企業規模的集中式記錄
- 使用 cert-manager 和 Let's Encrypt 為 Amazon EKS 上的應用程式設定end-to-end加密

- 使用 Flux 簡化 Amazon EKS 多租戶應用程式部署
- 使用自動化工作流程簡化 Amazon Lex 機器人開發和部署
- 使用 AWS Fargate WaitCondition 勾點建構來協調資源相依性和任務執行
- 在 AWS CodePipeline 中使用第三方 Git 來源儲存庫
- 使用 AWS CodePipeline 建立 CI/CD 管道來驗證 Terraform 組態
- 更多模式

使用後台和自助式 Amazon SageMaker AI 範本加速 MLOps

Ashish Bhatt、Shashank hirematt 和 Shivanshu Suryakar，Amazon Web Services

總結

使用機器學習操作 (MLOps) 系統的組織在擴展、標準化和保護其 ML 基礎設施方面面臨重大挑戰。此模式引進一種轉型方法，結合了開放原始碼開發人員入口網站 Backstage 與 Amazon SageMaker AI 和強化基礎設施即程式碼 (IaC) 模組，以改善資料科學團隊如何開發、部署和管理 ML 工作流程。

此模式的 IaC 模組會在 GitHub AWS AIOps 模組儲存庫中提供。這些模組提供預先建置的範本，用於設定 ML 基礎設施和建立一致的 ML 環境。不過，資料科學家通常很難直接使用這些範本，因為他們需要基礎設施專業知識。新增開發人員入口網站，例如 Backstage，為資料科學家建立易於使用的方式，以部署標準化的 ML 環境，而無需了解基礎基礎設施詳細資訊。

透過使用 Backstage 做為自助式平台並整合預先設定的 SageMaker AI 範本，您可以：

- 加速 ML 計畫的價值實現時間。
- 協助強制執行一致的安全性和控管。
- 為資料科學家提供標準化、合規的環境。
- 降低營運開銷和基礎設施複雜性。

此模式提供解決 MLOps 重大挑戰的解決方案，也提供可擴展、可重複的架構，在維持組織標準的同時實現創新。

目標對象

此模式適用於組織內涉及 ML、雲端架構和平台工程的廣泛受眾。其中包含：

- 想要標準化和自動化 ML 工作流程部署的 ML 工程師。
- 希望自助存取預先設定且合規 ML 環境的資料科學家。
- 負責建置和維護內部開發人員平台和共用基礎設施的平台工程師。
- 為 MLOps 設計可擴展、安全且符合成本效益之雲端解決方案的雲端架構師。
- 有興趣將持續整合和持續交付 (CI/CD) 實務延伸至 ML 基礎設施佈建和工作流程的 DevOps 工程師。
- 技術主管和經理負責監督 ML 計畫，並希望改善團隊生產力、控管和上市時間。

如需有關 MLOps 挑戰、SageMaker AI MLOps 模組，以及此模式提供的解決方案如何滿足 ML 團隊需求的詳細資訊，請參閱其他資訊一節。

先決條件和限制

先決條件

- AWS Identity and Access Management 將資源佈建至 的 (IAM) 角色和許可 AWS 帳戶
- 了解 Amazon SageMaker Studio、SageMaker Projects、SageMaker Pipelines 和 SageMaker Model Registry 概念
- 了解 IaC 原則和使用 等工具的經驗 AWS Cloud Development Kit (AWS CDK)

限制

- 範本涵蓋範圍有限。目前，該解決方案僅支援更廣泛的 AIOps 解決方案中的 SageMaker AI 相關 AIOps 模組。其他模組，例如 Ray on Amazon Elastic Kubernetes Service (Amazon EKS)、MLflow、Apache Airflow，以及 Amazon Bedrock 的微調，尚無法做為後台範本使用。
- 不可設定的預設設定。範本使用來自 AIOps SageMaker 模組的固定預設組態，無需自訂。您無法透過 Backstage 介面修改執行個體類型、儲存體大小、聯網組態或安全政策，這會限制特定使用案例的彈性。
- AWS僅限 支援。平台專為 AWS 部署而設計，不支援多雲端案例。在 外部使用雲端服務的組織 AWS 雲端 ，無法將這些範本用於其 ML 基礎設施需求。
- 手動憑證管理。您必須為每個部署手動提供登入 AWS 資料。此解決方案不提供與公司身分提供者 AWS IAM Identity Center或自動憑證輪換的整合。
- 有限的生命週期管理。範本缺少全面的資源生命週期管理功能，例如自動清除政策、成本最佳化建議和基礎設施偏離偵測。建立後，您必須手動管理和監控部署的資源。

Architecture

下圖顯示統一開發人員入口網站的解決方案架構，該入口網站使用 SageMaker AI 標準化和加速跨環境的 ML 基礎設施部署。

在此架構中：

1. AWS 應用程式現代化藍圖使用 Amazon EKS 叢集佈建基礎設施設定，做為雲端原生卓越營運 (CNOE) 架構的基礎。此全方位解決方案提供可擴展的內部開發人員平台 (IDP)，可解決複雜的雲端

原生基礎設施管理挑戰。藍圖提供結構化方法來設定強大、彈性的基礎設施，以適應不斷發展的組織需求。

2. CNOE 開放原始碼架構整合 DevOps 工具，並透過統一的平台工程方法解決生態系統分段問題。透過結合不同的工具和技術，可簡化雲端原生開發的複雜環境，讓您的團隊可以專注於創新，而不是工具鏈管理。此架構提供標準化方法來選取、整合和管理開發工具。
3. 透過 CNOE，Backstage 會部署為 out-of-the-box解決方案。Backstage 透過Keycloak 綁定了強大的身分驗證，並透過Argo CD 綁定了全面的部署工作流程。此整合式平台會建立集中式環境來管理開發程序，並為團隊提供單一位置，以跨多個環境存取、部署和監控其基礎設施和應用程式。
4. GitHub 儲存庫包含預先設定的 AIOps 軟體範本，涵蓋整個 SageMaker AI 生命週期。這些範本可解決重要的 ML 基礎設施需求，包括 SageMaker Studio 佈建、模型訓練、推論管道和模型監控。這些範本可協助您加速 ML 計畫，並確保不同專案和團隊的一致性。
5. GitHub Actions 實作自動化工作流程，透過 Seed-Farmer 公用程式動態觸發資源佈建。此方法將 Backstage 目錄與 AIOps 模組儲存庫整合，並建立簡化的基礎設施部署程序。自動化可減少手動介入、將人為錯誤降到最低，並確保跨不同環境快速、一致的基礎設施建立。
6. AWS CDK 可協助您將基礎設施定義為程式碼並進行佈建，並確保跨指定資源的可重複、安全且合規的部署 AWS 帳戶。這種方法以最少的手動介入提供最大的控管，因此您可以建立標準化的基礎設施範本，以便輕鬆複寫、版本控制和稽核。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端 基礎設施。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 上執行 Kubernetes， AWS 而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- Amazon SageMaker AI 是一種受管 ML 服務，可協助您建置和訓練 ML 模型，然後將模型部署到生產就緒的託管環境中。

其他工具

- Backstage 是一種開放原始碼架構，可協助您建置內部開發人員入口網站。
- GitHub Actions 是一種 CI/CD 平台，可自動化軟體開發工作流程，包括建置、測試和部署程式碼等任務。

程式碼儲存庫

此模式使用來自下列 GitHub 儲存庫的程式碼和範本：

- AIOps 內部開發人員平台 (IDP) 與 Backstage 儲存庫
- 來自 AIOps 模組儲存庫的 SageMaker AI 相關模組 AWS AIOps
- 儲存庫上的現代工程 AWS

實作

此實作使用 儲存庫上現代工程 AWS後台的生產級部署模式。此方法可大幅簡化設定程序，同時整合安全性和可擴展性的 AWS 最佳實務。

此模式的 Epics 區段概述了實作方法。如需詳細的step-by-step部署說明，請參閱 AIOps 內部開發人員平台 (IDP) 與 Backstage 儲存庫中提供的完整部署指南。實作包括：

- 初始後台平台部署
- 將 SageMaker 軟體範本與 Backstage 整合
- 使用和維護後台範本

部署指南也包含持續維護、疑難排解和平台擴展的指引。

最佳實務

遵循這些最佳實務，以協助確保 MLOps 基礎設施實作的安全性、控管和卓越營運。

範本管理

- 絕不對即時範本進行重大變更。
- 在生產部署之前，請務必徹底測試更新。
- 維護清楚且記錄良好的範本版本。

安全性

- 將 GitHub 動作釘選至特定遞交安全雜湊演算法 (SHAs)，以協助防止供應鏈攻擊。
- 使用具有精細許可的最低權限 IAM 角色。
- 將敏感登入資料儲存在 GitHub Secrets 和 中AWS Secrets Manager。

- 絕不在 範本中硬式編碼登入資料。

控管與追蹤

- 實作全面的資源標記標準。
- 啟用精確的成本追蹤和合規監控。
- 維護基礎設施變更的清晰稽核線索。

本指南為使用 Backstage、SageMaker AI 和 IaC 模組實作這些最佳實務提供了堅實的基礎。

史詩

設定您的 ML 環境

任務	Description	所需的技能
部署 Backstage。	此步驟使用 儲存庫上現代工程 AWS的藍圖，建置強大且可擴展的基礎設施，整合多個 AWS 服務 以建立 ML 工作流程的集中式 IDP。遵循部署指南後台部署一節中的指示，複製儲存庫、安裝相依性、引導 AWS CDK 設定環境變數，以及部署後台平台。 基礎設施使用 Amazon EKS 做為容器協同運作平台，以部署 IDP 元件。Amazon EKS 架構包含安全的聯網組態，可建立嚴格的網路隔離並控制存取模式。平台與身分驗證機制整合，以協助保護使用者跨服務和環境的存取。	平台工程師
設定 SageMaker AI 範本。	此步驟使用 GitHub AIOps 內部開發人員平台 (IDP) 中的指	平台工程師

任務	Description	所需的技能
	令碼搭配 Backstage 儲存庫。遵循部署指南的 SageMaker 範本設定區段中的指示，複製儲存庫、設定先決條件，以及執行設定指令碼。 此程序會建立儲存庫，其中包含與 Backstage 整合所需的 SageMaker AI 範本。	
將 SageMaker AI 範本與 Backstage 整合。	請遵循部署指南的 SageMaker 範本整合區段中的指示來註冊 SageMaker AI 範本。 此步驟會將 AIOps 模組 （最後一個步驟的 SageMaker AI 範本） 整合到您的後台部署中，讓您可以自行滿足 ML 基礎設施需求。	平台工程師

任務	Description	所需的技能
使用 Backstage 中的 SageMaker AI 範本。	遵循部署指南的使用 SageMaker 範本一節中的指示，存取 Backstage 入口網站並在 SageMaker Studio 中建立 ML 環境。 在 Backstage 入口網站中，您可以從可用的 SageMaker AI 範本中選擇，包括 SageMaker Studio 環境、SageMaker 筆記本、自訂 SageMaker 專案範本和模型部署管道的選項。在您提供組態參數之後，平台會自動建立專用儲存庫，並透過 GitHub Actions 和 Seed-Farmer 佈建 AWS 資源。您可以透過 GitHub 動作日誌和 Backstage 元件目錄來監控進度。	資料科學家、資料工程師、開發人員

管理用於控管和合規的範本

任務	Description	所需的技能
更新 SageMaker AI 範本。	若要更新後台中的 SageMaker AI 範本，請遵循下列步驟。 1. 修改範本內容： a. 在 `template.yaml` 檔案中進行必要的變更，或編輯 `skeleton/` 目錄中的檔案。	平台工程師

任務	Description	所需的技能
	b. 在本機或開發環境中測試任何新參數、動作或檔案結構。 2. 測試變更： a. 使用 Backstage UI 或 CLI (`@backstage/create-app`)，使用更新後的範本來堆疊測試元件。 b. 驗證所有步驟是否成功執行，以及產生的程式碼是否符合您的期望。 3. 遞交和推送變更： a. 將變更推送至儲存範本的 Git 儲存庫。 如果範本已在特定分支中註冊（例如 `main`），則會自動反映更新。 b. 如果您使用的是版本控制（請參閱下一個步驟），請確定已更新正確的版本或標籤。	

任務	Description	所需的技能
建立和管理範本的多個版本。	對於重大變更或升級，您可能想要建立 SageMaker AI 範本的多個版本。 1. 為每個版本使用 Git 標籤或分支；例如： ``` git checkout -b v2.0.0 git push origin v2.0.0 ``` 2. （選用但建議）分別註冊每個版本。 在後台中，您可以將不同版本的範本註冊為目錄中的個別實體，其中每個實體指向特定的分支或標籤。例如（檔案）.yaml： ``` metadata: name: node-service- template-v2 description: Node.js service template - Version 2 spec: type: template lifecycle: experimen tal version: '2.0.0' ``` 3. 透過在範本儲存庫中維護CHANGELOG.md 檔案，清楚傳達變更。在此檔案中，記錄在範本的每個版本中引入了哪些功能或變更。	平台工程師

任務	Description	所需的技能
	4. 如有必要，請棄用範本的較舊版本： a. 在範本描述中將其標記為已棄用。 b. 如果不再需要版本，請從目錄中移除版本。	

擴展您的 ML 環境

任務	Description	所需的技能
將範本涵蓋範圍擴展到 SageMaker AI 之外。	目前的解決方案只會實作 SageMaker AI 相關的 AIOps 範本。您可以新增 AIOps 模組並整合其他 AWS 服務 和應用程式的自訂軟體範本，以擴展 ML 環境。您可以使用 Backstage 中的範本設計工具界面、實作自訂 scaffolder 動作，或使用標準中繼資料維護範本儲存庫，來建立這些項目。平台支援範本版本控制、跨團隊共用和驗證工作流程，以確保一致性。如需詳細資訊，請參閱 Backstage 文件。 您也可以實作範本繼承模式來建立基礎範本的專用版本。此可擴展性可讓您管理 SageMaker AI 以外的各種 AWS 資源和應用程式，同時保留簡化的開發人員體驗和維護組織的標準。	平台工程師

任務	Description	所需的技能
使用動態參數注入。	目前範本使用預設組態而不進行自訂，並執行 Seed-Farmer CLI 以部署具有預設變數的資源。您可以針對模組特定的組態使用動態參數注入來擴展預設組態。	平台工程師
增強安全性和合規性。	若要增強建立 AWS 資源的安全性，您可以啟用角色型存取控制 (RBAC) 與單一登入 (SSO)、SAML、OpenID Connect (OIDC) 和政策整合，做為程式碼強制執行。	平台工程師
新增自動資源清除。	您可以啟用自動清除政策的功能，也可以新增基礎設施偏離偵測和修復。	平台工程師

清除資源

任務	Description	所需的技能
移除後台基礎設施和 SageMaker AI 資源。	使用完 ML 環境後，請遵循部署指南的清除和資源管理一節中的指示，移除後台基礎設施並刪除 ML 環境中的 SageMaker AI 資源。	平台工程師

疑難排解

問題	解決方案
AWS CDK 引導失敗	驗證 AWS 登入資料和區域組態。

問題	解決方案
Amazon EKS 叢集存取問題	檢查 kubectl 組態和 IAM 許可。
Application Load Balancer 連線問題	確定安全群組允許連接埠 80/443 上的傳入流量。
GitHub 整合問題	驗證 GitHub 字符許可和組織存取。
SageMaker AI 部署失敗	檢查AWS 服務 配額和 IAM 許可。

相關資源

- 平台工程（在 AWS Cloud Adoption Framework： Platform perspective 指南中）
- Amazon SageMaker AI 文件
- 後台軟體範本（後台網站）
- AIOps 模組儲存庫 (ML 可重複使用 IaC 模組的集合）
- AIOps 內部開發人員平台 (IDP) 與 Backstage 儲存庫
- 儲存庫上的現代工程 AWS
- 雲端原生卓越營運 (CNOE) 網站

其他資訊

業務挑戰

開始或擴展 MLOps 計畫的組織經常遇到以下業務和技術挑戰：

- 環境不一致。缺乏標準化的開發和部署環境會使協同合作變得困難，並增加部署風險。
- 手動佈建額外負荷。使用 SageMaker Studio、Amazon Simple Storage Service (Amazon S3) 儲存貯體、IAM 角色和 CI/CD 管道手動設定 ML 基礎設施非常耗時且容易出錯，並從其模型開發的核心任務中轉移資料科學家。
- 缺乏可探索性和重複使用性。如果缺少集中式目錄，就很難找到現有的 ML 模型、資料集和管道。這會導致多餘的工作和錯失重複使用的機會。
- 複雜的控管和合規。確保 ML 專案遵守組織安全政策、資料隱私權法規和合規標準，例如健康保險流通與責任法案 (HIPAA) 和一般資料保護法規 (GDPR)，在沒有自動化防護機制的情況下可能具有挑戰性。

- 值時間變慢。這些挑戰的累積效果會導致長期的 ML 專案生命週期，並延遲從 ML 投資實現商業價值。
- 安全風險。不一致的組態和手動程序可能會引入安全漏洞，導致難以強制執行最低權限和網路隔離。

這些問題會延長開發週期、增加營運開銷，並帶來安全風險。ML 的反覆性質需要可重複的工作流程和高效的協同合作。

截至 2026 年，Gartner 預測 80% 的軟體工程組織將擁有平台團隊。（請參閱《Gartner 網站上的平台工程讓開發人員變得更好、更快、更快樂。）此預測強調 IDP 如何加速軟體交付。做為 IDP，Backstage 有助於將訂單還原至複雜的基礎設施，以便團隊快速安全地交付高品質的程式碼。將 Backstage 與強化的 AIOps 模組整合，可協助您從被動故障診斷轉變為主動預防。

MLOps SageMaker 模組

用於此模式的 GitHub 儲存庫中的 AIOps 模組為 AWS 透過可重複使用且強化的 IaC 在 上標準化 MLOps 提供了寶貴的基礎。這些模組封裝佈建 SageMaker 專案、管道和相關聯網和儲存資源的最佳實務，目標是降低複雜性並加速 ML 環境的設定。您可以將這些範本用於各種 MLOps 使用案例，以建立一致且安全的部署模式，促進更受管且更有效率的 ML 工作流程方法。

直接使用 AIOps 模組通常需要平台團隊部署和管理這些 IaC 範本，這可能為希望自助存取的資料科學家帶來挑戰。探索和了解可用的範本、設定必要的參數，以及觸發其部署，可能需要導覽 AWS 服務主控台或直接與 IaC 工具互動。這可能會為偏好專注於 ML 任務的資料科學家產生摩擦、增加認知負載，如果這些範本不是透過集中且易於使用的界面進行管理，可能會導致參數化不一致或偏離組織標準。將這些強大的 AIOps 模組與 Backstage 等 IDP 整合，可提供簡化的自助式體驗、增強的可探索性和更強大的控管控制，以使用這些標準化的 MLOps 建置區塊，協助解決這些挑戰。

後台做為 IDP

內部開發人員平台 (IDP) 是由平台團隊建置的自助服務層，可簡化和標準化開發人員建置、部署和管理應用程式的方式。它可消除基礎設施的複雜性，並讓開發人員透過統一的界面輕鬆存取工具、環境和服務。

IDP 的主要目標是透過以下方式增強開發人員體驗和生產力：

- 為服務建立和部署等任務啟用自助式服務。
- 透過標準範本提升一致性和合規性。
- 整合整個開發生命週期 (CI/CD、監控和文件） 的工具。

Backstage 是由 Spotify 建立的開放原始碼開發人員入口網站，現在是雲端原生運算基金會 (CNCF) 的一部分。它透過提供集中、可擴展的平台來管理軟體元件、工具和文件，協助組織建置自己的 IDP。透過 Backstage，開發人員可以：

- 透過軟體目錄探索和管理所有內部服務。
- 透過 scaffolder 外掛程式使用預先定義的範本建立新專案。
- 從單一位置存取整合式工具，例如 CI/CD 管道、Kubernetes 儀表板和監控系統。
- 透過 TechDocs 維護一致的 Markdown 型文件。

常見問答集

使用此後台範本與透過 SageMaker 主控台手動部署 SageMaker Studio 有何不同？

Backstage 範本提供優於手動 AWS 主控台部署的多種優勢，包括遵循組織最佳實務的標準化組態、使用 Seed-Farmer 和 的自動化 IaC 部署 AWS CDK、內建安全政策和合規措施，以及透過 GitHub 與您組織的開發人員工作流程整合。範本也會使用版本控制建立可重現的部署，讓您更輕鬆地跨不同階段（開發、預備、生產）複寫環境，並維持團隊之間的一致性。此外，範本包含自動清除功能，並透過 Backstage 與您組織的身分管理系統整合。透過主控台手動部署需要深入的 AWS 專業知識，並且不提供版本控制或範本提供的相同層級的標準化和管理。因此，相較於生產 ML 環境，主控台部署更適合一次性實驗。

什麼是 Seed-Farmer？為什麼使用此解決方案？

Seed-Farmer 是一種 AWS 部署協調工具，可透過使用 管理基礎設施模組 AWS CDK。此模式使用 Seed-Farmer，因為它提供標準化、可重複使用的基礎設施元件，專為 AI/ML 工作負載而設計， AWS 服務 自動處理之間的複雜相依性，並確保在不同環境中進行一致的部署。

我需要安裝 AWS CLI 才能使用這些範本嗎？

否，您不需要 AWS CLI 在電腦上安裝 。範本完全透過雲端中的 GitHub 動作執行。您可以透過 Backstage 介面提供 AWS 登入資料（存取金鑰、私密金鑰和工作階段字符），而部署會自動在 GitHub Actions 環境中進行。

部署 SageMaker Studio 環境需要多長時間？

典型的 SageMaker Studio 部署需要 15-25 分鐘才能完成。這包括 AWS CDK 引導 (2-3 分鐘）、Seed-Farmer 工具鏈設定 (3-5 分鐘） 和資源建立 (10-15 分鐘）。確切時間取決於您的 AWS 區域 和聯網設定的複雜性。

我可以在相同的 中部署多個 SageMaker 環境 AWS 帳戶嗎？

是，您可以。每個部署都會根據您在範本中提供的元件名稱來建立具有唯一名稱的資源。不過，請注意 AWS 服務 配額：每個帳戶每個區域的 SageMaker 網域數量有限，因此請在建立多個環境之前檢查您的配額。

使用 Amazon Bedrock 自動化 AWS 基礎設施操作

Ishwar Chauthaiwale 和 Anand Bukkapatnam Tirumala，Amazon Web Services

摘要

在雲端原生解決方案中，自動化常見的基礎設施操作在維護有效率、安全且符合成本效益的環境方面扮演重要角色。手動處理操作耗時且容易發生人為錯誤。此外，具有不同 AWS 專業知識層級的團隊成員需要執行這些任務，同時確保遵守安全通訊協定。此模式示範如何使用 Amazon Bedrock 透過自然語言處理 (NLP) 自動化常見的 AWS 基礎設施操作。

此模式可協助組織開發可重複使用、模組化且安全的程式碼，以便在多個環境中部署生成式 AI 型基礎設施。透過專注於基礎設施即程式碼 (IaC) 和自動化，它提供了重要的 DevOps 優勢，包括版本控制、一致的部署、減少錯誤、更快速的佈建和改善的協同合作。

模式實作安全架構，讓團隊能夠管理與金鑰相關的操作， AWS 服務 包括：

- Amazon Simple Storage Service (Amazon S3) 儲存貯體版本控制管理
- 建立 Amazon Relational Database Service (Amazon RDS) 快照
- Amazon Elastic Compute Cloud (Amazon EC2) 執行個體管理

架構採用 Amazon Virtual Private Cloud (Amazon VPC) 端點和私有聯網進行安全通訊，其 AWS Lambda 函數在私有子網路中做為任務執行器運作。Amazon S3 提供資料管理並實作全方位 AWS Identity and Access Management (IAM) 角色和許可，以確保適當的存取控制。此解決方案不包含聊天歷史記錄功能，也不會儲存聊天。

先決條件和限制

- 作用中 AWS 帳戶。
- 應採取適當的存取控制措施，以協助保護和控制存取。存取控制的範例包括使用 AWS Systems Manager、基礎模型存取、用於部署的 IAM 角色，以及以服務為基礎的角色、停用對 Amazon S3 儲存貯體的公開存取，以及設定無效字母佇列。
- AWS Key Management Service (AWS KMS) 客戶受管金鑰。
- AWS Command Line Interface (AWS CLI) 第 2 版或更新版本，已在部署環境中安裝和設定。
- 已安裝並設定 Terraform AWS Provider 第 4 版或更新版本。
- 安裝https://developer.hashicorp.com/terraform/install並設定 Terraform 1.5.7 版或更新版本。

- 在 Amazon Bedrock 中檢閱和定義代理程式動作群組的 OpenAPI 結構描述，以協助防止未經授權的存取和維護資料完整性。
- 在 中針對所需的 Amazon Titan Text Embeddings v2 和 Claude 3.5 Sonnet 或 Claude 3 Haiku 基礎模型啟用存取。 AWS 帳戶 https://docs.aws.amazon.com/bedrock/latest/userguide/models-supported.html為了避免部署失敗，請確認您的目標部署 AWS 區域 支援所需的模型。
- 設定虛擬私有雲端 (VPC)，遵循 AWS Well Architected Framework 最佳實務。
- 已完成 Amazon Responsible AI 政策的檢閱。

產品版本

- Amazon Titan Text Embeddings v2
- Anthropic Claude 3.5 Sonnet 或 Claude 3 Haiku
- Terraform AWS Provider 第 4 版或更新版本
- Terraform 1.5.7 版或更新版本

Architecture

下圖顯示此模式的工作流程和架構元件。

解決方案架構包含多個層，可共同處理自然語言請求並執行對應的 AWS 操作：

1. 使用者透過 Amazon Bedrock 聊天主控台提出操作請求。
2. 聊天機器人使用 Amazon Bedrock 知識庫來處理請求。它實作 Amazon Titan Text Embeddings v2 模型進行自然語言處理。
3. 如果使用者提示包含動作請求，Amazon Bedrock 動作群組會使用 Anthropic Claude 3 Haiku 或 Claude 3.5 Sonnet 模型（取決於您的選擇）執行邏輯，並透過 OpenAPI 結構描述定義操作。
4. 動作群組使用 到達 Amazon VPC 端點， AWS PrivateLink 以安全進行服務通訊。
5. 透過 Amazon Bedrock 服務的 Amazon VPC 端點達到 AWS Lambda 函數。
6. Lambda 函數是主要執行引擎。根據請求，Lambda 函數會呼叫 API 以在 上執行動作 AWS 服務。Lambda 函數也會處理操作路由和執行。
7. 從 Lambda 函數 AWS 服務 取得 API 請求並執行對應的操作。
8. Lambda 函數會計算 Amazon Bedrock 理解的輸出承載。

9. 此承載會透過使用 PrivateLink 進行安全的服務通訊，傳送至 Amazon Bedrock。Amazon Bedrock 使用的大型語言模型 (LLM) 了解此承載，並將其轉換為人類可理解的格式。
10然後，輸出會在 Amazon Bedrock 聊天主控台上向使用者顯示。

解決方案會啟用下列主要操作：

- Amazon S3 – 啟用儲存貯體版本控制以進行版本控制。
- Amazon RDS – 建立資料庫快照以進行備份。
- Amazon EC2 – 列出執行個體並控制執行個體的啟動和停止。

工具

AWS 服務

- Amazon Bedrock 是一項全受管服務，可讓您透過統一 API 使用來自領導 AI 新創公司的高效能基礎模型 (FMs) 和 Amazon。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon OpenSearch Serverless 是 Amazon OpenSearch Service 的隨需無伺服器組態。
- AWS PrivateLink 可協助您建立從虛擬私有雲端 (VPCs) 到 VPC 外部服務的單向私有連線。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 中設定、操作和擴展關聯式資料庫 AWS 雲端。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Systems Manager 可協助您管理在 中執行的應用程式和基礎設施 AWS 雲端。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。

其他工具

- Git 是開放原始碼的分散式版本控制系統。
- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。

程式碼儲存庫

此模式的程式碼可在 GitHub aws-samples/infra-ops-orchestrator 儲存庫中使用。

最佳實務

- 定期監控 Lambda 執行日誌。如需詳細資訊，請參閱監控和疑難排解 Lambda 函數。如需最佳實務的詳細資訊，請參閱使用 AWS Lambda 函數的最佳實務。
- 定期檢閱安全組態，以確保符合組織的需求。如需詳細資訊，請參閱安全最佳實務。
- 遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。

史詩

部署解決方案

任務	Description	所需的技能

複製儲存庫。	若要在本機電腦上複製儲存庫，請執行下列命令：
	`git clone "git@github.com:aws-samples/infra-ops-orchestrator.git"`
	`cd infra-ops-orchestrator`
		AWS DevOps，DevOps 工程師

編輯環境變數。	在複製儲存庫的根目錄中編輯 `terraform.tfvars` 檔案。檢閱 指示的預留位	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	置[XXXXX]，並根據您的環境進行更新。	

建立基礎設施。	若要建立基礎設施，請執行下列命令：
	`terraform init`
	`terraform plan`
	請仔細檢閱執行計畫。如果計劃的變更是可接受的，請執行下列命令：
	`terraform apply --auto-approve`
		AWS DevOps，DevOps 工程師

存取解決方案

任務	Description	所需的技能
存取解決方案。	成功部署後，請依照下列步驟使用聊天型界面： 1. 若要存取 Infrastructure Orchestrator Assistant，AWS 管理主控台 請使用具有 Amazon Bedrock 許可的 IAM 角色登入 ，然後開啟位於 https：//https://console.aws.amazon.com/bedrock/ 的 Amazon Bedrock 主控台。從左側導覽窗格選取代理程式。然後，在客服人	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	員區段中選擇 Infrastructure Orchestrator Assistant。 2. 針對下列建議，請確定目標資源存在於您的 AWS 環境中，然後嘗試這些範例操作： • 建立 Amazon RDS 執行個體的快照備份，方法是詢問：'建立 RDS 執行個體的快照 【instance-name】' • 在 Amazon S3 儲存貯體上啟用版本控制，方法是詢問：「啟用儲存貯體的版本控制 【儲存貯體名稱】」 • 透過詢問 來列出 Amazon EC2 執行個體：「列出所有 EC2 執行個體」 • 開始或停止 Amazon EC2 執行個體，方法是詢問：'Start EC2 執行個體 【instance-id】' 或 'Stop EC2 執行個體 【instance-id】' 注意：將括號中的值取代為 AWS 環境中的實際資源名稱或 IDs。	

清除資源

任務	Description	所需的技能

刪除建立的資源。	若要刪除此模式建立的所有基礎設施，請執行下列命令：
	`terraform plan -destroy`
	請仔細檢閱銷毀計畫。如果計劃刪除是可接受的，請執行下列命令：
	`terraform destroy`
	注意：此命令將永久刪除此模式建立的所有資源。命令會在移除任何資源之前提示確認。
		AWS DevOps，DevOps 工程師

疑難排解

問題	解決方案
代理程式行為	如需此問題的相關資訊，請參閱 Amazon Bedrock 文件中的測試和疑難排解代理程式行為。
Lambda 網路問題	如需有關這些問題的資訊，請參閱 Lambda 文件中的對 Lambda 中的聯網問題進行故障診斷。
IAM 許可	如需這些問題的相關資訊，請參閱 IAM 文件中的疑難排解 IAM。

相關資源

- 為 Amazon RDS 的單一可用區域資料庫執行個體建立資料庫快照

- 在 Amazon Bedrock 中為代理程式的動作群組定義 OpenAPI 結構描述
- 在儲存貯體上啟用版本控制
- Amazon Bedrock 代理程式的運作方式
- 使用 Amazon Bedrock 知識庫擷取資料並產生 AI 回應
- 透過 安全地存取服務 AWS PrivateLink
- 停止和啟動 Amazon EC2 執行個體
- 使用動作群組來定義代理程式要執行的動作

使用 Terraform 自動化負載平衡器端點變更時的 CloudFront 更新

Tamilselvan P、Mohan Annam 和 Naveen Suthar，Amazon Web Services

總結

當 Amazon Elastic Kubernetes Service (Amazon EKS) 的使用者透過 Helm Chart 刪除和重新安裝其輸入組態時，會建立新的 Application Load Balancer (ALB)。這會造成問題，因為 Amazon CloudFront 會繼續參考舊 ALB 的 DNS 記錄。因此，無法連線到目標為此端點的服務。（如需此有問題工作流程的詳細資訊，請參閱其他資訊。）

為了解決此問題，此模式描述使用使用 Python 開發的自訂 AWS Lambda 函數。此 Lambda 函數會自動偵測何時透過 Amazon EventBridge 規則建立新的 ALB。然後 適用於 Python (Boto3) 的 AWS SDK， 函數會使用新的 ALB DNS 地址更新 CloudFront 組態，以確保流量路由至正確的端點。

此自動化解決方案可維持服務連續性，而不會產生額外的路由或延遲。即使基礎基礎設施變更，此程序也有助於確保 CloudFront 一律參考正確的 ALB DNS 端點。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 使用 Helm 在 Amazon EKS 上部署用於測試和驗證的範例 Web 應用程式。如需詳細資訊，請參閱《Amazon EKS 文件》中的在 Amazon EKS 上使用 Helm 部署應用程式。
- 設定 CloudFront 將呼叫路由到由 Helm 輸入控制器建立的 ALB。如需詳細資訊，請參閱 Amazon EKS 文件中的 Install AWS Load Balancer 控制器與 Helm，以及 CloudFront 文件中的限制對 Application Load Balancer 的存取。
- 在本機工作區中安裝和設定 Terraform。

限制

- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

- Terraform 1.0.0 版或更新版本
- Terraform AWS Provider 4.20 版或更新版本

Architecture

下圖顯示此模式的工作流程和架構元件。

此解決方案會執行下列步驟：

1. 每當 Helm 重新啟動或部署時，Amazon EKS 輸入控制器都會建立新的 Application Load Balancer (ALB)。
2. EventBridge 會尋找 ALB 建立事件。
3. ALB 建立事件會觸發 Lambda 函數。
4. Lambda 函數已根據 python 3.9 部署，並使用 boto3 API 進行呼叫。 AWS 服務 Lambda 函數會使用從建立負載平衡器事件接收的最新負載平衡器 DNS 名稱來更新 CloudFront 項目。

工具

AWS 服務

- Amazon CloudFront 透過全球資料中心網路提供 Web 內容，從而降低延遲並改善效能，從而加快 Web 內容的發佈速度。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 上執行 Kubernetes， AWS 而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如， AWS Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他事件匯流排 AWS 帳戶。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- 適用於 Python (Boto3) 的 AWS SDK 是一種軟體開發套件，可協助您整合 Python 應用程式、程式庫或指令碼 AWS 服務。

其他工具

- Python 是一種一般用途的電腦程式設計語言。
- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。

程式碼儲存庫

此模式的程式碼可在 GitHub aws-cloudfront-automation-terraform-samples 儲存庫中使用。

史詩

設定本機工作站

任務	Description	所需的技能
設定 Git CLI。	若要在本機工作站中安裝和設定 Git 命令列界面 (CLI)，請遵循 Git 文件中的入門 – 安裝 Git 說明。	DevOps 工程師
建立專案資料夾並新增檔案。	1. 前往模式的 GitHub 儲存庫，然後選擇程式碼按鈕。 2. 在複製對話方塊中，選擇 HTTPS 索引標籤。在使用 Web URL 複製中，複製顯示的 URL。 3. 在本機電腦上建立資料夾。將其命名為您的專案名稱。 4. 在本機電腦上開啟終端機，然後導覽至此資料夾。 5. 若要複製此模式的 git 儲存庫，請執行下列命令：`git clone https://github.com/aws-samples/aws-cloudfront-automation-terraform-samples` 6. 複製儲存庫之後，請使用下列命令前往複製的目錄：`cd <directory name>/cloudfront-update`	DevOps 工程師

任務	Description	所需的技能
	在您選擇的整合式開發環境 (IDE) 中開啟此專案。	

使用 Terraform 組態佈建目標架構

任務	Description	所需的技能

部署解決方案。	若要在目標中部署資源 AWS 帳戶，請使用下列步驟：
	1. 前往 `cloudfront-update` 資料夾。
	2. 使用 更新 `terraform.tfvars` 檔案`cloudfront_distribution_id` 。
	3. 若要 AWS 區域 為您的 AWS 設定檔設定 ，請執行下列命令：
	`export AWS_REGION={{ REGION }}`
	4. 若要初始化 Terraform，請執行下列命令：
	`terraform init`
	5. 若要驗證 Terraform，請執行下列命令：
	`terraform validate`
	6. 若要建立 Terraform 執行計劃，請執行下列命令：
		DevOps 工程師

任務	Description	所需的技能

	`terraform plan`
	7. 若要從 套用動作`terraform plan`，請執行下列命令：
	`terraform apply`	

驗證部署

任務	Description	所需的技能
驗證部署。	1. 登入 AWS 管理主控台 ，並在 https://console.aws.amazon.com/cloudfront/v4/home：// 開啟 Amazon CloudFront 主控台。 2. 在左側導覽窗格中，選擇分佈，然後開啟 CloudFront 分佈。 3. 在原始伺服器索引標籤上，確認原始伺服器名稱和原始伺服器映射具有更新的 ALB DNS 記錄。	DevOps 工程師

清除基礎設施

任務	Description	所需的技能
清除基礎設施。	若要清除您先前建立的基礎設施，請使用下列步驟：	DevOps 工程師

任務	Description	所需的技能
	1. 執行下列命令：`terraform destroy` 2. 若要確認銷毀命令，請輸入 `yes`。	

疑難排解

問題	解決方案
驗證供應商登入資料時發生錯誤	當您從本機電腦執行 Terraform `apply`或 `destroy` 命令時，您可能會遇到類似以下的錯誤： `Error: configuring Terraform AWS Provider: error validating provider credentials: error calling sts:GetCallerIdentity: operation error STS: GetCallerIdentity, https response error StatusCode: 403, RequestID: 123456a9-fbc1-40ed-b8d8-513d0133ba7f, api error InvalidClientTokenId: The security token included in the request is invalid.` 此錯誤是由本機電腦組態中使用的登入資料的安全字符過期所造成。 若要解決錯誤，請參閱 AWS Command Line Interface (AWS CLI) 文件中的設定和檢視組態設定。

相關資源

AWS resources

- 限制對 Application Load Balancer 的存取

- 使用 AWS Load Balancer控制器路由網際網路流量

Terraform 文件

- AWS 供應商
- 安裝 Terraform
- 遠端狀態

其他資訊

有問題的工作流程

該圖顯示以下工作流程：

1. 當使用者存取應用程式時，呼叫會前往 CloudFront。
2. CloudFront 會將呼叫路由至個別的 Application Load Balancer (ALB)。
3. ALB 包含目標 IP 地址，即應用程式 Pod 的 IP 地址。從那裡，ALB 會將預期結果提供給使用者。

不過，此工作流程會示範問題。應用程式部署是透過 Helm Chart 進行。每當有部署或有人重新啟動 Helm 時，也會重新建立個別的輸入。因此，外部負載平衡器控制器會重新建立 ALB。此外，在每次重新建立期間，會使用不同的 DNS 名稱重新建立 ALB。因此，CloudFront 在原始伺服器設定中會有過時的項目。由於此項目過時，使用者將無法連線應用程式。此問題會導致使用者停機。

替代解決方案

另一個可能的解決方法是為 ALB 建立外部 DNS，然後將其指向 CloudFront 中的 Amazon Route 53 私有託管區域端點。不過，此方法會在應用程式流程中新增另一個跳轉，這可能會導致應用程式延遲。此模式的 Lambda 函數解決方案不會中斷目前的流程。

使用 GitHub 動作自動化 AWS CDK Python 應用程式的 Amazon CodeGuru 檢閱

Vanitha Dontireddy 和 Sarat Chandra Pothula，Amazon Web Services

摘要

注意：自 2025 年 11 月 7 日起，您無法在 Amazon CodeGuru Reviewer 中建立新的儲存庫關聯。若要了解具有類似 CodeGuru Reviewer 功能的服務，請參閱 CodeGuru Reviewer 文件中的 Amazon CodeGuru Reviewer 可用性變更。 CodeGuru

此模式展示透過 GitHub 動作協調的 AWS Cloud Development Kit (AWS CDK) Python 應用程式的 Amazon CodeGuru 自動化程式碼檢閱整合。解決方案會部署 AWS CDK Python 中定義的無伺服器架構。透過在開發管道中自動化專家程式碼分析，此方法可以對 AWS CDK Python 專案執行下列動作：

- 增強程式碼品質。
- 簡化工作流程。
- 最大化無伺服器運算的優勢。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- AWS Command Line Interface (AWS CLI) 2.9.11 版或更新版本，已安裝並設定。
- 作用中的 GitHub 帳戶和 GitHub 儲存庫，具有讀取和寫入工作流程許可，以及 GitHub Actions 建立提取請求 (PR)，以確保 PR 工作流程正常運作。
- GitHub 動作中的 OpenID Connect (OIDC) 角色，可在 中部署解決方案 AWS 帳戶。若要建立角色，請使用 AWS CDK 建構。

限制

- Amazon CodeGuru Profiler 支援以所有 Java 虛擬機器 (JVM) 語言 （ 例如 Scala 和 Kotlin) 撰寫的應用程式，以及執行時間和 Python 3.6 或更新版本。
- Amazon CodeGuru Reviewer 僅支援來自下列來源提供者的 Java 和 Python 程式碼儲存庫關聯：Bitbucket AWS CodeCommit、GitHub、GitHub Enterprise Cloud 和 GitHub Enterprise Server。此外，僅透過 GitHub 動作支援 Amazon Simple Storage Service (Amazon S3) 儲存庫。

- 在持續整合和持續部署 (CI/CD) 管道期間，沒有自動列印問題清單的方法。反之，此模式使用 GitHub 動作做為處理和顯示問題清單的替代方法。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱依區域的 AWS 服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

下圖顯示此解決方案的架構。

如圖所示，當開發人員建立提取請求 (PR) 以供檢閱時，GitHub Actions 會觸發下列步驟：

1. IAM 角色假設 – 管道使用 GitHub Secrets 中指定的 IAM 角色來執行部署任務。
2. 程式碼分析
   - CodeGuru Reviewer 會分析存放在 Amazon S3 儲存貯體中的程式碼。它可識別瑕疵並提供修正和最佳化的建議。
   - CodeGuru Security 會掃描政策違規和漏洞。
3. 調查結果檢閱
   - 管道會在主控台輸出中列印問題清單儀表板的連結。
   - 如果偵測到關鍵問題清單，管道會立即失敗。
   - 對於高、正常或低嚴重性的問題清單，管道會繼續進行下一個步驟。
4. PR 核准
   - 檢閱者必須手動核准 PR。
   - 如果 PR 遭拒，管道會失敗並停止進一步的部署步驟。
5. CDK 部署 – 在 PR 核准時，CDK 部署程序就會開始。它會設定下列 AWS 服務 和資源：
   - CodeGuru Profiler
   - AWS Lambda 函數
   - Amazon Simple Queue Service (Amazon SQS) 佇列
6. 分析資料產生 – 若要為 CodeGuru Profiler 產生足夠的分析資料：
   - 管道會定期傳送訊息至 Amazon SQS 佇列，多次叫用 Lambda 函數。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端 基礎設施。
- CDK Toolkit 是命令列雲端開發套件，可協助您與 AWS CDK 應用程式互動。
- Amazon CodeGuru Profiler 會從即時應用程式收集執行期效能資料，並提供可協助您微調應用程式效能的建議。
- Amazon CodeGuru Reviewer 使用程式分析和機器學習來偵測開發人員難以找到的潛在瑕疵。然後，CodeGuru Profiler 會提供改善 Java 和 Python 程式碼的建議。
- Amazon CodeGuru Security 是一種靜態應用程式安全工具，使用機器學習來偵測安全政策違規和漏洞。它提供解決安全風險的建議，並產生指標，讓您可以追蹤應用程式的安全狀態。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Queue Service (Amazon SQS) 提供安全、耐用且可用的託管佇列，可協助您整合和分離分散式軟體系統和元件。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他工具

- GitHub Actions 是與 GitHub 儲存庫緊密整合的持續整合和持續交付 (CI/CD) 平台。您可以使用 GitHub 動作來自動化建置、測試和部署管道。

程式碼儲存庫

此模式的程式碼可在 GitHub amazon-codeguru-suite-cdk-python 儲存庫中使用。

最佳實務

- 遵循使用 開發和部署雲端基礎設施的最佳實務 AWS CDK。
- 在 GitHub Actions 工作流程 中使用 時，請遵循 IAM 中的安全最佳實務，包括： AWS 服務
  - 請勿將登入資料存放在您的儲存庫程式碼中。
  - 擔任 IAM 角色 來接收臨時登入資料，並盡可能使用臨時登入資料。
  - 將最低權限授予 GitHub 動作工作流程中使用的 IAM 角色。僅授予在 GitHub 動作工作流程中執行動作所需的許可。

- 監控 GitHub 動作工作流程中使用的 IAM 角色活動 。 GitHub
- 定期輪換您使用的任何長期登入資料。

史詩

設定您的環境

任務	Description	所需的技能

設定 AWS 登入資料。	若要匯出定義 AWS 帳戶 和您要部署堆疊 AWS 區域 之位置的變數，請執行下列命令：
	`export CDK_DEFAULT_ACCOUNT=<12-digit AWS account number>`
	`export CDK_DEFAULT_REGION=<AWS Region>`
	的 AWS 登入資料 AWS CDK 是透過環境變數提供。
		AWS DevOps，DevOps 工程師

複製儲存庫。	若要在本機電腦上複製儲存庫，請執行下列命令： `git clone https://github.com/aws-samples/amazon-codeguru-suite-cdk-python.git`	AWS DevOps，DevOps 工程師
安裝 CDK Toolkit。	若要確認已安裝 CDK Toolkit 並檢查版本，請執行下列命令： `cdk --version`	AWS DevOps，DevOps 工程師

任務	Description	所需的技能

	如果 CDK Toolkit 版本早於 2.27.0，請輸入下列命令將其更新至 2.27.0 版：
	`npm install -g aws-cdk@2.27.0`
	如果未安裝 CDK Toolkit，請執行下列命令來安裝它：
	`npm install -g aws-cdk@2.27.0 --force`	

安裝所需的依存項目。	若要安裝所需的專案相依性，請執行下列命令：
	`python -m pip install --upgrade pip`
	`pip install -r requirements.txt`
		AWS DevOps，DevOps 工程師

任務	Description	所需的技能

引導 CDK 環境。	若要引導 AWS CDK 環境，請執行下列命令：
	`npm install`
	`npm run cdk bootstrap "aws://${ACCOUNT_NUMBER}/${AWS_REGION}"`
	成功引導環境後，應該會顯示下列輸出：
	`# Bootstrapping environment aws://{account}/{region}...`
	`# Environment aws://{account}/{region} bootstrapped`
		AWS DevOps，DevOps 工程師

部署 CDK 應用程式

任務	Description	所需的技能
合成 AWS CDK 應用程式。	若要合成 AWS CDK 應用程式，請執行下列命令： `cdk synth` 如需此命令的詳細資訊，請參閱 AWS CDK 文件中的 cdk 合成。	AWS DevOps，DevOps 工程師
部署 資源。	若要部署資源，請執行下列命令：	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	`cdk deploy --require-approval never` **Note** `--require-approval never` 旗標表示 CDK 將自動核准和執行所有變更。這包括 CDK 通常會標記為需要手動檢閱的變更（例如 IAM 政策變更或移除資源）。在生產環境中使用 `--require-approval never`旗標之前，請確定您的 CDK 程式碼和 CI/CD 管道已經過良好測試且安全。	

建立 GitHub 秘密和個人存取字符

任務	Description	所需的技能
在 GitHub 中建立所需的秘密。	若要允許 GitHub Actions 工作流程安全地存取 AWS 資源，而不會暴露儲存庫程式碼中的敏感資訊，請建立秘密。若要在 GitHub 中為 `ROLE_TO_ASSUME` 、`CodeGuruReviewArtifactBucketName` 和 建立秘密`AWS_ACCOUNT_ID` ，請	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	遵循 GitHub 動作文件中為儲存庫建立秘密中的指示。 以下是有關變數的詳細資訊： • AWS_ACCOUNT_ID – 執行管道的 AWS 帳戶 ID。 • CodeGuruReviewArtifactBucketName – 存放 CodeGuru Reviewer 成品的 S3 儲存貯體名稱。此模式使用儲存貯體名稱 codeguru-reviewer-build-artifacts-<ACCOUNT_ID>-<REGION> 。 • AWS_REGION – AWS 區域資源所在的 。 • ROLE_TO_ASSUME – 管道擔任的 IAM 角色名稱。此模式使用角色名稱 githubActionsDeployRole 。	

任務	Description	所需的技能
建立 GitHub 個人存取字符。	若要為您的 GitHub 動作工作流程設定安全的方式來驗證身分並與 GitHub 互動，請執行下列動作： 1. 若要建立對儲存庫具有讀取和寫入存取權的 GitHub 個人存取字符，請遵循 GitHub 文件中管理個人存取字符的指示。 2. 若要將此字符儲存為 GitHub 動作的儲存庫秘密，請遵循 GitHub 動作文件中為儲存庫建立秘密中的指示。	AWS DevOps，DevOps 工程師

清除

任務	Description	所需的技能
清除資源。	若要清除 AWS CDK Python 應用程式，請執行下列命令： `cdk destroy --all`	DevOps 工程師

疑難排解

問題	解決方案
顯示儀表板問題清單的連結。	在 CI/CD 管道期間，無法列印問題清單。反之，此模式使用 GitHub 動作做為處理和顯示問題清單的替代方法。

相關資源

AWS resources

- AWS 雲端開發套件
- Amazon CodeGuru 文件
- Amazon S3
- AWS Identity and Access Management
- Amazon Simple Queue Service
- 什麼是 AWS Lambda？

GitHub 文件

- 在 Amazon Web Services 中設定 OpenID Connect
- GitHub 動作
- 重複使用工作流程
- 觸發工作流程

使用 GitHub Actions、Artifactory 和 Terraform 在多儲存庫設定中自動化 AWS Supply Chain 資料湖的部署

Keshav Ganesh，Amazon Web Services

摘要

此模式提供自動化方法來使用多儲存庫持續整合和持續部署 (CI/CD) 管道來部署和管理 AWS Supply Chain 資料湖。它示範了兩種部署方法：使用 GitHub Actions 工作流程自動部署，或直接使用 Terraform 手動部署。這兩種方法都使用 Terraform for infrastructure as code (IaC)，搭配自動方法新增 GitHub Actions 和 JFrog Artifactory 來增強 CI/CD 功能。

解決方案利用 AWS Supply Chain AWS Lambda和 Amazon Simple Storage Service (Amazon S3) 來建立資料湖基礎設施，同時使用任一部署方法來自動化組態和資源建立。此自動化可消除手動組態步驟，並確保跨環境進行一致的部署。此外， AWS Supply Chain 不需要在擷取、轉換和載入 (ETL) 方面擁有深入的專業知識，並且可以提供採用 Amazon Quick Sight 技術的洞見和分析。

透過實作此模式，組織可以縮短部署時間、將基礎設施維護為程式碼，並透過版本控制的自動化程序來管理供應鏈資料湖。多儲存庫方法提供精細的存取控制，並支援獨立部署不同的元件。團隊可以選擇最適合其現有工具和程序的部署方法。

先決條件和限制

先決條件

請確定本機電腦上已安裝下列項目：

- AWS Command Line Interface (AWS CLI) 第 2 版
- GitHub CLI
- Python 3.13 版
- Terraform 1.12 版或更新版本

在部署之前，請確定已備妥下列項目：

- 作用中 AWS 帳戶。
- 虛擬私有雲端 (VPC)， AWS 區域 在您的 中選擇的 AWS 帳戶 中有兩個私有子網路。
- 用於部署至下列服務的 AWS Identity and Access Management (IAM) 角色的足夠許可：

- AWS Supply Chain – 偏好完整存取來部署其元件，例如資料集和整合流程，以及從 存取。 AWS 管理主控台
- Amazon CloudWatch Logs – 用於建立和管理 CloudWatch 日誌群組。
- Amazon Elastic Compute Cloud (Amazon EC2) – 適用於 Amazon EC2 安全群組和 Amazon Virtual Private Cloud (Amazon VPC) 端點。
- Amazon EventBridge – 供 使用 AWS Supply Chain。
- IAM – 用於建立 AWS Lambda 服務角色。
- AWS Key Management Service (AWS KMS) – 用於存取 AWS KMS keys 用於 Amazon S3 成品儲存貯體和 Amazon S3 AWS Supply Chain 預備儲存貯體的 。
- AWS Lambda – 用於建立部署 AWS Supply Chain 元件的 Lambda 函數。
- Amazon S3 – 用於存取 Amazon S3 成品儲存貯體、伺服器存取記錄儲存貯體和 AWS Supply Chain 預備儲存貯體。如果您使用手動部署，也需要 Amazon S3 Terraform 成品儲存貯體的許可。
- Amazon VPC – 用於建立和管理 VPC。

如果您偏好使用 GitHub 動作工作流程進行部署，請執行下列動作：

- 為具有先前提及許可的 IAM 角色設定 OpenID Connect (OIDC)。
- 建立具有類似許可的 IAM 角色來存取 AWS 管理主控台。如需詳細資訊，請參閱 IAM 文件中的建立角色以授予許可給 IAM 使用者。

如果您偏好手動部署，請執行下列動作：

- 建立 IAM 使用者以擔任具有先前提及許可的 IAM 角色。如需詳細資訊，請參閱 IAM 文件中的建立角色以授予許可給 IAM 使用者。
- 在您的本機終端機中擔任 角色。

如果您偏好使用 GitHub 動作工作流程進行部署，請設定下列項目：

- 取得主機名稱、登入使用者名稱和登入存取權杖的 JFrog Artifactory 帳戶。
- 用於存放成品的 JFrog 專案金鑰和儲存庫。

限制

- AWS Supply Chain 執行個體不支援複雜的資料轉換技術。
- AWS Supply Chain 最適合供應鏈網域，因為它提供內建的分析和洞見。對於任何其他網域， AWS Supply Chain 可以用作資料湖架構中的資料存放區。
- 此解決方案中使用的 Lambda 函數可能需要增強，才能在生產規模部署中處理 API 重試和記憶體管理。
- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

您可以使用自動化 GitHub 動作工作流程或使用 Terraform 手動部署此解決方案。

使用 GitHub 動作自動部署

下圖顯示使用 GitHub 動作工作流程的自動化部署選項。JFrog Artifactory 用於成品管理。它存放資源資訊和輸出，以用於多儲存庫部署。

使用 Terraform 手動部署

下圖顯示透過 Terraform 的手動部署選項。Amazon S3 用於成品管理，而不是 JFrog Artifactory。

部署工作流程

圖表顯示下列工作流程：

1. 使用下列其中一種部署方法部署 AWS Supply Chain 服務資料集基礎設施和資料庫：
   - 自動化部署 – 使用 GitHub Actions 工作流程協調所有部署步驟，並使用 JFrog Artifactory 進行成品管理。
   - 手動部署 – 直接針對每個部署步驟執行 Terraform 命令，並使用 Amazon S3 進行成品管理。
2. 建立 AWS Supply Chain 服務操作所需的支援 AWS 資源：
   - Amazon VPC 端點和安全群組
   - AWS KMS keys
   - CloudWatch Logs 日誌群組
3. 建立和部署下列基礎設施資源：

- 管理（建立、更新和刪除）AWS Supply Chain 服務執行個體、命名空間和資料集的 Lambda 函數。
- AWS Supply Chain 暫存 Amazon S3 儲存貯體以進行資料擷取

4. 部署 Lambda 函數，管理預備儲存貯體和 AWS Supply Chain 資料集之間的整合流程。部署完成後，剩餘的工作流程步驟會管理資料擷取和分析。
5. 設定 AWS Supply Chain 暫存 Amazon S3 儲存貯體的來源資料擷取。
6. 將資料新增至 AWS Supply Chain 預備 Amazon S3 儲存貯體後，服務會自動觸發至 AWS Supply Chain 資料集的整合流程。
7. AWS Supply Chain 與 Quick Sight Analytics 整合，以根據擷取的資料產生儀表板。

工具

AWS 服務

- Amazon CloudWatch Logs 可協助您集中所有系統、應用程式的日誌， AWS 服務 以便您可以監控日誌並將其安全地存檔。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如， AWS Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他事件匯流排 AWS 帳戶。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS IAM Identity Center 可協助您集中管理所有 AWS 帳戶 和雲端應用程式的單一登入 (SSO) 存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Q in AWS Supply Chain 是互動式生成式 AI 助理，可透過分析 AWS Supply Chain 資料湖中的資料，協助您更有效率地操作供應鏈。

- Amazon Quick Sight 是一種雲端規模的商業智慧 (BI) 服務，可協助您在單一儀表板中視覺化、分析和報告您的資料。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Supply Chain 是以雲端為基礎的受管應用程式，可作為供應鏈網域組織中的資料存放區，可用於產生洞見並對擷取的資料進行分析。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。Amazon VPC 端點是一種虛擬裝置，可協助您將 VPC 私下連線至支援的 ， AWS 服務 而不需要網際網路閘道、NAT 裝置、VPN 連線或 AWS Direct Connect 連線。

其他工具

- GitHub Actions 是與 GitHub 儲存庫緊密整合的持續整合和持續交付 (CI/CD) 平台。您可以使用 GitHub 動作來自動化建置、測試和部署管道。
- HashiCorp Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。
- JFrog Artifactory 透過應用程式交付程序，提供二進位檔和成品的end-to-end自動化和管理。
- Python 是一種一般用途的電腦程式設計語言。此模式使用 Python 讓 AWS 函數的程式碼與 互動 AWS Supply Chain

.

最佳實務

- 實作此模式時，請盡可能維持最高的安全性。如先決條件中所述，請確定具有兩個私有子網路的虛擬私有雲端 (VPC) 位於 AWS 區域 您選擇的 AWS 帳戶 中的 中。
- 盡可能使用 AWS KMS 客戶受管金鑰，並授予他們有限的存取許可。
- 若要設定具有擷取此模式資料所需最低存取權的 IAM 角色，請參閱此模式儲存庫中的從來源系統安全資料擷取至 Amazon S3。

史詩

（兩個選項）設定本機工作站

任務	Description	所需的技能

複製儲存庫。	若要複製此模式的儲存庫，請在本機工作站中執行下列命令：
	
	`git clone https://github.com/aws-samples/sample-automate-aws-supply-chain-deployment.git`
	`cd ASC-Deployment`
		AWS DevOps

（自動化選項）驗證部署的先決條件。	請確定自動化部署的先決條件已完成。	應用程式擁有者

（手動選項）準備部署 AWS Supply Chain 資料集。	若要前往 的`terraform-deployment` 目錄`ASC-Datasets` ，請執行下列命令：
	
	`cd ASC-Datasets/terraform-deployment`
	
	若要擔任在先決條件中建立的角色 ARN，請執行下列命令：
	
	`aws sts assume-role --role-arn <enter AWS user role ARN> --role-session-name <your-session-name>`
	
	若要設定和匯出環境變數，請執行下列命令：
		AWS DevOps

任務	Description	所需的技能

	`# Export Environment variables`
	`export REGION=<Enter deployment region>`
	`export REPO_NAME=<Enter Current ASC Datasets dir name>`
	`export PROJECT_NAME="asc-deployment-poc"`
	`export ACCOUNT_ID=<Enter deployment Account ID>`
	`export ENVIRONMENT="dev"`
	`export LAMBDA_LAYER_TEMP_DIR_TERRAFORM="layerOutput"`
	`export LAMBDA_FUNCTION_TEMP_DIR_TERRAFORM="lambdaOutput"`
	`export AWS_USER_ROLE=<Enter user role ARN for AWS Console access and deployment>`
	`export S3_TERRAFORM_ARTIFACTS_BUCKET_NAME="$PROJECT_NAME-$ACCOUNT_ID-$REGION-terraform-artifacts-$ENVIRONMENT"`	

任務	Description	所需的技能

（手動選項）準備管理部署中的 AWS Supply Chain 整合流程。	若要前往 的terraform-deployment 目錄ASC-Integration-Flows ，請執行下列命令：
	`cd ASC-Integration-Flows/terraform-deployment`
	若要擔任先前建立的角色 ARN，請執行下列命令：
	`aws sts assume-role --role-arn <enter AWS user role ARN> --role-session-name <your-session-name>`
	若要設定和匯出環境變數，請執行下列命令：
	`# Export Environment variables`
	`export REGION=<Enter deployment region>`
	`export REPO_NAME=<Enter Current ASC Integration Flows dir name>`
	`export ASC_DATASET_VARS_REPO=<Enter Current ASC Datasets dir name>  #Must be the same directory name used for ASC Datasets deployment`
		應用程式擁有者

任務	Description	所需的技能
	export PROJECT_NAME="asc-deployment-poc" export ACCOUNT_ID=<Enter deployment Account ID> export ENVIRONMENT="dev" export LAMBDA_LAYER_TEMP_DIR_TERRAFORM="layerOutput" export LAMBDA_FUNCTION_TEMP_DIR_TERRAFORM="lambdaOutput" export S3_TERRAFORM_ARTIFACTS_BUCKET_NAME="$PROJECT_NAME-$ACCOUNT_ID-$REGION-terraform-artifacts-$ENVIRONMENT"	

（自動化選項）使用 GitHub 動作工作流程部署 AWS Supply Chain 資料集

任務	Description	所需的技能
複製 ASC-Datasets 目錄。	若要將ASC-Datasets 目錄複製到新位置，請使用下列步驟： 1. 若要前往 ASC-Datasets 目錄，請執行下列命令： cd ASC-Datasets	AWS DevOps

任務	Description	所需的技能
	2. 若要將ASC-Datasets 目錄複製到新位置，請執行下列命令： `cp -r ASC-Datas ets ../ASC-Datasets- standalone cd ../ASC-Datasets- standalone`	
設定 ASC-Datasets 目錄。	若要在您的組織中ASC-Datas ets 將 設定為獨立儲存庫，請執行下列命令： `git init git add . git commit -m "Initial commit: ASC-Datasets standalone repositor y" git remote add origin <INSERT_ASC_DATASE TS_GITHUB_URL> git branch -M dev`	AWS DevOps
在 .github 工作流程檔案中設定分支名稱。	在部署工作流程檔案中設定分支名稱，如下列範例所示： `on: workflow_ dispatch: push: branches: - dev #Change to any other branch preferred for deployment`	應用程式擁有者

任務	Description	所需的技能
設定 GitHub 環境並設定環境值。	若要在 GitHub 組織中設定 GitHub 環境，請使用此模式儲存庫中設定 GitHub 環境的指示。 若要在工作流程檔案中設定環境值，請使用此模式儲存庫中工作流程檔案中設定環境值中的指示。	應用程式擁有者
觸發工作流程。	若要將您的變更推送到您的 GitHub 組織並觸發部署工作流程，請執行下列命令： `git push -u origin dev`	AWS DevOps

（自動化選項）使用 GitHub 動作工作流程部署 AWS Supply Chain 整合流程

任務	Description	所需的技能
複製 ASC-Integration-Flows 目錄。	若要將ASC-Integration-Flows 目錄複製到新位置，請使用下列步驟： 1. 若要前往 ASC-Integration-Flows 目錄，請執行下列命令： `cd ASC-Integration-Flows` 2. 若要將ASC-Integration-Flows 目錄複製到新位置，請執行下列命令：	AWS DevOps

任務	Description	所需的技能
	`cp -r ASC-Integration-Flows ../ASC-Integration-Flows-standalone` `cd ../ASC-Integration-Flows-standalone`	

設定 ASC-Integration-Flows 目錄。	若要將ASC-Integration-Flows 目錄設定為組織中的獨立儲存庫，請執行下列命令：
	`git init`
	`git add .`
	`git commit -m "Initial commit: ASC-Integration-Flows standalone repository"`
	`git remote add origin <INSERT_ASC_Integration_Flows_GITHUB_URL>`
	`git branch -M dev`
		AWS DevOps

在 .github 工作流程檔案中設定分支名稱。	在部署工作流程檔案中設定分支名稱，如下列範例所示：
	`on:`
	`  workflow_dispatch:`
	`  push:`
	`  branches:`
	`  - dev`
	`#Change to any other branch preferred for deployment`
		應用程式擁有者

任務	Description	所需的技能
設定 GitHub 環境並設定環境值。	若要在 GitHub 組織中設定 GitHub 環境，請使用此模式儲存庫中設定 GitHub 環境的指示。 若要在工作流程檔案中設定環境值，請使用此模式儲存庫中工作流程檔案中設定環境值中的指示。	應用程式擁有者
觸發工作流程。	若要將您的變更推送到您的 GitHub 組織並觸發部署工作流程，請執行下列命令： `git push -u origin dev`	AWS DevOps

（手動選項）使用 Terraform 部署 AWS Supply Chain 資料集

任務	Description	所需的技能
導覽至 `terraform-deployment` 目錄。	若要前往 的`terraform-deployment` 目錄`ASC-Datasets` ，請執行下列命令： `cd ASC-Datasets/terraform-deployment`	AWS DevOps

設定 Terraform 狀態 Amazon S3 儲存貯體。	若要設定 Terraform 狀態 Amazon S3 儲存貯體，請使用下列指令碼：
	
	`# Setup terraform bucket`
	`chmod +x ../scripts/setup-terraform.sh`
		AWS DevOps

任務	Description	所需的技能
	`../scripts/setup-terraform.sh`	

設定 Terraform 成品 Amazon S3 儲存貯體。	若要設定 Terraform 成品 Amazon S3 儲存貯體，請使用下列指令碼：
	`# Setup terraform artifacts bucket`
	`chmod +x ../scripts/setup-terraform-artifacts-bucket.sh`
	`../scripts/setup-terraform-artifacts-bucket.sh`
		AWS DevOps

設定 Terraform 後端和提供者組態。	若要設定 Terraform 後端和提供者組態，請使用下列指令碼：
	`# Setup terraform backend and providers config if they don't exist`
	`chmod +x ../scripts/generate-terraform-config.sh`
	`../scripts/generate-terraform-config.sh`
		AWS DevOps

任務	Description	所需的技能

產生部署計畫。	若要產生部署計畫，請執行下列命令：
	
	`# Run terraform init and validate`
	`terraform init`
	`terraform validate`
	
	`# Run terraform plan`
	`terraform plan \`
	`-var-file="tfInputs/$ENVIRONMENT.tfvars" \`
	`-var="project_name=$PROJECT_NAME" \`
	`-var="environment=$ENVIRONMENT" \`
	`-var="user_role=$AWS_USER_ROLE" \`
	`-var="lambda_temp_dir=$LAMBDA_FUNCTION_TEMP_DIR_TERRAFORM" \`
	`-var="layer_temp_dir=$LAMBDA_LAYER_TEMP_DIR_TERRAFORM" \`
	`-parallelism=40 \`
	`-out='tfplan.out'`
		AWS DevOps

部署組態。	若要部署組態，請執行下列命令：
	
	`# Run terraform apply`
	`terraform apply tfplan.out`
		AWS DevOps

任務	Description	所需的技能

更新其他組態並存放輸出。	若要更新 AWS KMS 金鑰政策，並將套用的組態輸出存放在 Terraform 成品 Amazon S3 儲存貯體中，請執行下列命令：
	`# Update AWS Supply Chain KMS Key policy with the service's requirements`
	`chmod +x ../scripts/update-asc-kms-policy.sh`
	`../scripts/update-asc-kms-policy.sh`
	`# Update AWS KMS Keys' policy with IAM roles`
	`chmod +x ../scripts/update-kms-policy.sh`
	`../scripts/update-kms-policy.sh`
	`# Create terraform outputs file to be used as input variables`
	`terraform output -json > raw_output.json`
	`jq -r 'to_entries | map( if .value.type == "string" then "\(.key) = \"\(.value.value)\"" else "\(.key) = \(.value.value | tojson)"`
		AWS DevOps

任務	Description	所需的技能
	end) | .[]' raw_outpu t.json > $REPO_NAME- outputs.tfvars # Upload reformed outputs file to Amazon S3 terraform artifacts bucket (For retrieval from other repositor ies) aws s3 cp $REPO_NAME- outputs.tfvars s3:// $S3_TERRAFORM_ARTIFACT S_BUCKET_NAME/$REP O_NAME-outputs.tfvars rm -f raw_output.json rm -f $REPO_NAME- outputs.tfvars	

（手動選項）使用 Terraform 部署 AWS Supply Chain 服務整合流程

任務	Description	所需的技能
導覽至 terraform-deployment 目錄。	若要前往 的terraform-deployment 目錄ASC-Integration-Flows ，請執行下列命令： cd ASC-Integration-Fl ows/terraform-depl oyment	AWS DevOps
設定 Terraform 後端和提供者組態。	若要設定 Terraform 後端和提供者組態，請使用下列指令碼：	AWS DevOps

任務	Description	所需的技能

	`# Setup terraform backend and providers config if they don't exist`
	`chmod +x ../scripts/generate-terraform-config.sh`
	`../scripts/generate-terraform-config.sh`	

任務	Description	所需的技能

產生部署計畫。	若要產生部署計畫，請執行下列命令。這些命令會初始化您的 Terraform 環境、將來自 的組態變數ASC-Datasets 與現有的 Terraform 組態合併，以及產生部署計畫。
	`# Run terraform init and validate`
	`terraform init`
	`terraform validate`
	`# Download and merge ASC DATASET tfvars`
	`chmod +x ../scripts/download-vars-through-s3.sh`
	`../scripts/download-vars-through-s3.sh $ASC_DATASET_VARS_REPO`
	`# Run terraform plan`
	`terraform plan \`
	`-var-file="tfInputs/$ENVIRONMENT.tfvars" \`
	`-var="project_name=$PROJECT_NAME" \`
	`-var="environment=$ENVIRONMENT" \`
	`-var="lambda_temp_dir=$LAMBDA_FUNCTION_TEMP_DIR_TERRAFORM" \`
	`-var="layer_temp_dir=$LAMBDA_LAYER_TEMP_DIR_TERRAFORM" \`
	`-parallelism=40 \`
		AWS DevOps

任務	Description	所需的技能
	`-out='tfplan.out'`	

部署組態。	若要部署組態，請執行下列命令：
	
	`# Run terraform apply`
	`terraform apply`
	`tfplan.out`
		AWS DevOps

任務	Description	所需的技能

更新其他組態。	若要更新 AWS KMS 金鑰政策，並將套用的組態輸出存放在 Terraform 成品 Amazon S3 儲存貯體中，請執行下列命令：
	`# Update AWS KMS Keys' policy with IAM roles`
	`chmod +x ../scripts/update-kms-policy-through-s3.sh`
	`../scripts/update-kms-policy-through-s3.sh $ASC_DATA SET_VARS_REPO`
	`# Create terraform outputs file to be used as input variables`
	`terraform output -json > raw_output.json`
	`jq -r 'to_entries | map( if .value.type == "string" then "\(.key) = \"\(.value.value)\"" else "\(.key) = \(.value.value | tojson)" end ) | .[]' raw_output.json > $REPO_NAME-outputs.tfvars`
	`# Upload reformed outputs file to Amazon S3 terraform artifacts`
		AWS DevOps

任務	Description	所需的技能
	bucket (For retrieval from other repositories) aws s3 cp $REPO_NAME-outputs.tfvars s3://$S3_TERRAFORM_ARTIFACTS_BUCKET_NAME/$REPO_NAME-outputs.tfvars rm -f raw_output.json rm -f $REPO_NAME-outputs.tfvars	

（兩個選項）擷取資料

任務	Description	所需的技能
上傳範例 CSV 檔案。	若要上傳資料集的範例 CSV 檔案，請使用下列步驟： 1. 針對部署中建立的行事曆和傳出訂單行資料集，使用不同的資料建立範例 CSV 檔案。 2. `asc_instance_id` 從 terraform 輸出目錄擷取 AWS Supply Chain 執行個體 ID。 3. 請注意在部署中建立 AWS Supply Chain 的 Amazon S3 儲存貯體名稱： `aws-supply-chain-data-<Instance_ID>` 4. 若要使用 上傳檔案 AWS CLI，請執行下列命令：	資料工程師

任務	Description	所需的技能

	`# Upload Calendar CSV file`
	`aws s3 cp calendar_sample.csv s3://aws-supply-chain-data-<Instance_ID>/calendar-data/`
	
	`# Upload Outbound Order Line CSV file`
	`aws s3 cp outbound_order_line_sample.csv s3://aws-supply-chain-data-<Instance_ID>/outbound-order-line-data/`	

（兩個選項）設定 AWS Supply Chain 存取權

任務	Description	所需的技能
設定 AWS Supply Chain 存取權。	若要從 設定 AWS Supply Chain 存取權 AWS 管理主控台，請使用下列步驟： 1. 登入 AWS 管理主控台 並搜尋 AWS Supply Chain 服務。 2. 前往執行個體 `asc-deployment-poc-dev-asc-instance` 。 3. 此模式使用 IAM Identity Center 來管理使用者對 AWS Supply Chain 執行個體的存取。若要確保完整存	應用程式擁有者

任務	Description	所需的技能
	取此解決方案，請以應用程式的管理員身分登入。	

（自動化選項）使用 GitHub 動作工作流程清除所有資源

任務	Description	所需的技能
觸發整合流程資源的銷毀工作流程。	`ASC-Integration-Flows` 從 GitHub 組織中的部署分支觸發 的銷毀工作流程。	AWS DevOps
觸發資料集資源的銷毀工作流程。	`ASC-Datasets` 從 GitHub 組織中的部署分支觸發 的銷毀工作流程。	AWS DevOps

（手動選項）使用 Terraform 清除 AWS Supply Chain 整合流程的資源

任務	Description	所需的技能
導覽至 `terraform-deployment` 目錄。	若要前往 的`terraform-deployment` 目錄`ASC-Integration-Flows` ，請執行下列命令： `cd ASC-Integration-Flows/terraform-deployment`	AWS DevOps
設定 Terraform 後端和提供者組態。	若要設定 Terraform 後端和提供者組態，請使用下列指令碼： `# Setup terraform backend and providers`	AWS DevOps

任務	Description	所需的技能
	config if they don't exist chmod +x ../scripts/generate-terraform-config.sh ../scripts/generate-terraform-config.sh	

任務	Description	所需的技能

產生基礎設施銷毀計畫。	若要透過產生詳細的捨棄計畫來準備 AWS 基礎設施的受控銷毀，請執行下列命令。程序會初始化 Terraform、整合 AWS Supply Chain 資料集組態，並建立銷毀計畫，供您在執行之前檢閱。
	`# Run terraform init and validate`
	`terraform init`
	`terraform validate`
	`# Download and merge ASC DATASET tfvars`
	`chmod +x ../scripts/download-vars-through-s3.sh`
	`../scripts/download-vars-through-s3.sh $ASC_DATASET_VARS_REPO`
	`# Run terraform plan`
	`terraform plan -destroy \`
	`-var-file="tfInputs/$ENVIRONMENT.tfvars" \`
	`-var="project_name=$PROJECT_NAME" \`
	`-var="environment=$ENVIRONMENT" \`
	`-var="lambda_temp_dir=$LAMBDA_FUNCTION_TEMP_DIR_TERRAFORM" \`
		AWS DevOps

任務	Description	所需的技能
	`-var="layer_temp_dir=$LAMBDA_LAYER_TEMP_DIR_TERRAFORM" \ -parallelism=40 \ -out='tfplan.out'`	
執行基礎設施銷毀計畫。	若要執行基礎設施的計劃銷毀，請執行下列命令： `# Run terraform apply terraform apply tfplan.out`	AWS DevOps
從 Amazon S3 儲存貯體移除 Terraform 輸出。	若要移除部署 期間上傳的輸出檔案`ASC-Integration-Flows` ，請執行下列命令： `# Delete the outputs file aws s3 rm s3://$S3_TERRAFORM_ARTIFACTS_BUCKET_NAME/$REPO_NAME-outputs.tfvars`	AWS DevOps

（手動選項）使用 Terraform 清除 AWS Supply Chain 服務資料集的資源

任務	Description	所需的技能
導覽至 `terraform-deployment` 目錄。	若要前往 的`terraform-deployment` 目錄`ASC-Datasets` ，請執行下列命令： `cd ASC-Datasets/terraform-deployment`	AWS DevOps

任務	Description	所需的技能

設定 Terraform 後端和提供者組態。	若要設定 Terraform 後端和提供者組態，請使用下列指令碼：
	
	`# Setup terraform backend and providers config if they don't exist`
	`chmod +x ../scripts/generate-terraform-config.sh`
	`../scripts/generate-terraform-config.sh`
		AWS DevOps

任務	Description	所需的技能

產生基礎設施銷毀計畫。	若要建立銷毀 AWS Supply Chain 資料集資源的計劃，請執行下列命令：
	
	`# Run terraform init and validate`
	`terraform init`
	`terraform validate`
	
	`# Run terraform plan`
	`terraform plan -destroy \`
	`-var-file="tfInputs/$ENVIRONMENT.tfvars" \`
	`-var="project_name=$PROJECT_NAME" \`
	`-var="environment=$ENVIRONMENT" \`
	`-var="user_role=$AWS_USER_ROLE" \`
	`-var="lambda_temp_dir=$LAMBDA_FUNCTION_TEMP_DIR_TERRAFORM" \`
	`-var="layer_temp_dir=$LAMBDA_LAYER_TEMP_DIR_TERRAFORM" \`
	`-parallelism=40 \`
	`-out='tfplan.out'`
		AWS DevOps

任務	Description	所需的技能

清空 Amazon S3 儲存貯體。	若要清空所有 Amazon S3 儲存貯體 （為 設定的伺服器存取記錄儲存貯體除外`force-destroy` ），請使用下列指令碼：
	`# Delete S3 buckets excluding server access logging bucket`
	`chmod +x ../scripts/empty-s3-buckets.sh`
	`../scripts/empty-s3-buckets.sh tfplan.out`
		AWS DevOps

執行基礎設施銷毀計畫。	若要使用產生的計劃執行 AWS Supply Chain 資料集基礎設施的計劃銷毀，請執行下列命令：
	`# Run terraform apply`
	`terraform apply tfplan.out`
		AWS DevOps

從 Amazon S3 Terraform 成品儲存貯體移除 Terraform 輸出。	若要完成清除程序，`ASC-Datasets` 請執行下列命令，移除部署 期間上傳的輸出檔案：
	`# Delete the outputs file`
	`aws s3 rm s3://$S3_TERRAFORM_ARTIFACTS_BUCKET_NAME/$REPO_NAME-outputs.tfvars`
		AWS DevOps

疑難排解

問題	解決方案
由於 AWS Supply Chain 內部錯誤或服務角色的 IAM 許可不足， AWS Supply Chain 資料集或整合流程無法正確部署。	首先，清除所有資源。然後，重新部署 AWS Supply Chain 資料集資源，然後重新部署 AWS Supply Chain 整合流程資源。
AWS Supply Chain 整合流程不會擷取為 AWS Supply Chain 資料集上傳的新資料檔案。	1. 檢查 AWS Supply Chain 整合流程組態的字首是否符合上傳範例資料檔案時使用的字首。 2. 如果 AWS Supply Chain 資料集的資源已重新建立，其相關聯的 Amazon Resource Name (ARNs) 會在內部變更。因此，請重新部署 AWS Supply Chain 服務整合流程的資源。

相關資源

AWS 文件

- AWS Supply Chain

其他資源

- 了解 GitHub 動作工作流程 (GitHub 文件)

其他資訊

此解決方案可以透過 提供的預先建置儀表板 AWS Supply Chain 或與 Amazon Quick Sight 的自訂整合，複寫更多資料集，並查詢以供進一步分析。此外，您可以使用 Amazon Q 來詢問與 AWS Supply Chain 執行個體相關的問題。

使用 AWS Supply Chain Analytics 分析資料

如需設定 AWS Supply Chain Analytics 的說明，請參閱 AWS Supply Chain 文件中的設定 AWS Supply Chain Analytics。

此模式示範了建立行事曆和 Outbound_Order_Line 資料集。若要建立使用這些資料集的分析，請使用下列步驟：

1. 若要分析資料集，請使用季節性分析儀表板。若要新增儀表板，請遵循 AWS Supply Chain 文件中預先建置儀表板中的步驟。
2. 選擇儀表板以查看以行事曆資料和外撥訂單行資料的範例 CSV 檔案為基礎的分析。

儀表板會根據資料集的擷取資料，提供多年來的需求洞察。您可以進一步指定 ProductID、CustomerID、年份和其他參數進行分析。

使用 Amazon Q 詢問與 AWS Supply Chain 執行個體相關的問題

Amazon Q in AWS Supply Chain 是互動式生成式 AI 助理，可協助您更有效率地操作供應鏈。Amazon Q 可以執行下列動作：

- 分析資料湖中的 AWS Supply Chain 資料。
- 提供營運和財務洞察。
- 回答您的立即供應鏈問題。

如需使用 Amazon Q 的詳細資訊，請參閱 AWS Supply Chain 文件中的在 中啟用 AWS Supply Chain Amazon Q 和在 中使用 Amazon Q AWS Supply Chain。

自動化 AWS 資源評估

Naveen Suthar、Arun Bagal、Manish Garg 和 Sandeep Gawande，Amazon Web Services

摘要

此模式描述使用 AWS 雲端開發套件 (AWS CDK) 設定資源評估功能的自動化方法。透過使用此模式，營運團隊會以自動化方式收集資源稽核詳細資訊，並在單一儀表板上檢視 AWS 帳戶中部署的所有資源詳細資訊。這在下列使用案例中很有用：

- 將基礎設施識別為程式碼 (IaC) 工具，並隔離由不同 IaC 解決方案建立的資源，例如 HashiCorp Terraform、AWS CloudFormation、AWS CDK 和 AWS Command Line Interface (AWS CLI)
- 擷取資源稽核資訊

此解決方案也會協助領導團隊從單一儀表板取得 AWS 帳戶中資源和活動的洞見。

注意：Amazon Quick Sight 是付費服務。在執行它來分析資料並建立儀表板之前，請檢閱 Amazon Quick Sight 定價。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 具有佈建資源存取權的 AWS Identity and Access Management (IAM) 角色和許可
- 建立的 Amazon Quick 帳戶可存取 Amazon Simple Storage Service (Amazon S3) 和 Amazon Athena
- 已安裝 AWS CDK 2.55.1 版或更新版本
- 已安裝 Python 3.9 版或更新版本

限制

- 此解決方案會部署到單一 AWS 帳戶。
- 除非 AWS CloudTrail 已設定並將資料儲存在 S3 儲存貯體中，否則解決方案不會追蹤部署之前發生的事件。

產品版本

- AWS CDK 2.55.1 版或更新版本
- Python 3.9 版或更新版本

Architecture

目標技術堆疊

- Amazon Athena
- AWS CloudTrail
- AWS Glue
- AWS Lambda
- Amazon Quick Sight
- Amazon S3

目標架構

AWS CDK 程式碼會部署在 AWS 帳戶中設定資源評估功能所需的所有資源。下圖顯示將 CloudTrail 日誌傳送至 AWS Glue、Amazon Athena 和 Quick Sight 的程序。

1. CloudTrail 會將日誌傳送至 S3 儲存貯體以進行儲存。
2. 事件通知會叫用處理日誌並產生篩選資料的 Lambda 函數。
3. 篩選的資料會存放在另一個 S3 儲存貯體中。
4. AWS Glue 爬蟲程式是在 S3 儲存貯體中篩選的資料上設定，以在 AWS Glue Data Catalog 資料表中建立結構描述。
5. 篩選的資料已準備好供 Amazon Athena 查詢。
6. Quick Sight 會存取查詢的資料以進行視覺化。

自動化和擴展

- 如果 AWS Organizations 中有全組織的 CloudTrail 追蹤，此解決方案可以從一個 AWS 帳戶擴展到多個 AWS 帳戶。 AWS Organizations 透過在組織層級部署 CloudTrail，您也可以使用此解決方案來擷取所有必要資源的資源稽核詳細資訊。

- 此模式使用 AWS 無伺服器資源來部署解決方案。

工具

AWS 服務

- Amazon Athena 是一種互動式查詢服務，可協助您使用標準 SQL 直接在 Amazon S3 中分析資料。
- AWS 雲端開發套件 (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。
- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在 AWS 帳戶和 AWS 區域的整個生命週期中管理這些資源。
- AWS CloudTrail 可協助您稽核 AWS 帳戶的控管、合規和營運風險。
- AWS Glue 是全受管的擷取、轉換和載入 (ETL) 服務。它可協助您可靠地分類、清理、擴充和移動資料存放區和資料串流之間的資料。此模式使用 AWS Glue 爬蟲程式和 AWS Glue Data Catalog 資料表。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而不需要佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Quick 是一種雲端規模的商業智慧 (BI) 服務，可協助您在單一儀表板中視覺化、分析和報告您的資料。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

程式碼儲存庫

此模式的程式碼可在 GitHub infrastructure-assessment-iac-automation 儲存庫中使用。

程式碼儲存庫包含下列檔案和資料夾：

- `lib` 資料夾 – 用來建立 AWS 資源的 AWS CDK 建構 Python 檔案
- `src/lambda_code` – 在 Lambda 函數中執行的 Python 程式碼
- `requirements.txt` – 必須安裝的所有 Python 相依性清單
- `cdk.json` – 輸入檔案，用來提供啟動資源所需的值

最佳實務

設定 Lambda 函數的監控和提醒。如需詳細資訊，請參閱監控和疑難排解 Lambda 函數。如需使用 Lambda 函數的一般最佳實務，請參閱 AWS 文件。

史詩

設定您的環境

任務	Description	所需的技能
在本機電腦上複製儲存庫。	若要複製儲存庫，請執行 `git clone https://github.com/aws-samples/infrastructure-assessment-iac-automation.git` 命令。	AWS DevOps，DevOps 工程師

設定 Python 虛擬環境並安裝必要的相依性。	若要設定 Python 虛擬環境，請執行下列命令。
	`cd infrastructure-assessment-iac-automation`
	`python3 -m venv .venv`
	`source .venv/bin/activate`
	若要設定所需的相依性，請執行命令 `pip install -r requirements.txt` 。
		AWS DevOps，DevOps 工程師

設定 AWS CDK 環境並合成 AWS CDK 程式碼。	1. 若要在您的 AWS 帳戶中設定 AWS CDK 環境，請執行命令 `cdk bootstrap aws://ACCOUNT-NUMBER/REGION` 。	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	2. 若要將程式碼轉換為 AWS CloudFormation 堆疊組態，請執行命令 `cdk synth`。	

在本機電腦上設定 AWS 登入資料

任務	Description	所需的技能

匯出要部署堆疊之帳戶和區域的變數。	若要使用環境變數為 AWS CDK 提供 AWS 登入資料，請執行下列命令。
	`export CDK_DEFAULT_ACCOUNT=<12 Digit AWS Account Number>`
	`export CDK_DEFAULT_REGION=<region>`
		AWS DevOps，DevOps 工程師

設定 AWS CLI 設定檔。	若要設定帳戶的 AWS CLI 設定檔，請遵循 AWS 文件中的指示。	AWS DevOps，DevOps 工程師

設定和部署資源評估工具

任務	Description	所需的技能
在帳戶中部署資源。	若要使用 AWS CDK 在 AWS 帳戶中部署資源，請執行下列動作： 1. 在複製儲存庫的根目錄中，在 `cdk.json` 檔案中，提供下列參數的輸入： • `s3_context`	AWS DevOps

任務	Description	所需的技能
	• ct_context • kms_context • lambda_context • glue_context • qs_context 這些值定義資源組態和命名法。預設值已設定，可視需要變更。 **Note** 若要避免顯示 S3 儲存貯體已存在的錯誤，請務必在 ct 和 output區段s3_context 中提供 的唯一名稱。 2. 若要部署資源，請執行命令 cdk deploy。 cdk deploy 命令會建立 CloudTrail 資源來記錄事件，並將日誌檔案儲存在輸入 S3 儲存貯體中。線索的日誌檔案將由 Lambda 函數處理。篩選結果會存放在輸出 S3 儲存貯體中，並準備好供 Amazon Athena 和 Amazon Quick Sight 使用。	

任務	Description	所需的技能
執行 AWS Glue 爬蟲程式並建立資料目錄資料表。	AWS Glue 爬蟲程式用於保持資料結構描述動態。解決方案會依 AWS Glue 爬蟲程式排程器所定義定期執行爬蟲程式，在 AWS Glue Data Catalog 資料表中建立和更新分割區。AWS Glue 在輸出 S3 儲存貯體中提供資料後，請使用下列步驟執行 AWS Glue 爬蟲程式並建立資料目錄資料表結構描述以進行測試： 1. 登入 AWS 管理主控台並導覽至 AWS Glue 主控台。 2. 在導覽窗格中的資料目錄下，選擇爬蟲程式。 3. 選取`iac-tool-qa-resource-iac-json-crawler` 爬蟲程式。 4. 執行爬蟲程式。 5. 爬蟲程式成功執行後，會建立 AWS Glue Data Catalog 資料表。Quick Sight 將使用資料表來視覺化資料。 **Note** AWS CDK 程式碼會將 AWS Glue 爬蟲程式設定為在特定時間執行，但您也可以隨需執行。	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
部署 Quick Sight 建構。	1. 若要部署 Quick Sight 建構，請在 #QuickSight setup – ends中取消 #QuickSight setup - start和 之間的程式碼註解resource_iac_tool_stack.py 。 2. 取消註解後，請執行 cdk deploy命令以在 QuickSight DataSet Quick Sight 帳戶中建立 QuickSight DataSource 和 。	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
建立 Quick Sight 儀表板。	若要建立 Quick Sight 儀表板和分析範例，請執行下列動作： 1. 導覽至 Quick Sight 主控台，然後選取部署資源的 AWS 區域。 2. 在導覽窗格中，選擇資料集，並驗證`ct-operations-iac-ds` 已在 Quick Sight 資料集中建立名為 的資料集。 如果您沒有看到資料集，請重新部署 Quick Sight 建構。 3. 選取`ct-operations-iac-ds` 資料集，然後選擇在分析中使用。 4. 選取預設工作表。 5. 從左側的欄位清單中選取個別的資料欄。 6. 選取所需的資料欄後，請選取適當的視覺效果類型以檢視資料。 如需詳細資訊，請參閱在 Amazon Quick Sight 中啟動分析和在 Amazon Quick Sight 中啟動視覺化類型。	AWS DevOps，DevOps 工程師

清除解決方案中的所有 AWS 資源

任務	Description	所需的技能
移除 AWS 資源。	1. 若要移除解決方案部署的 AWS 資源，請執行命令 `cdk destroy`。 2. 刪除兩個 S3 儲存貯體中的所有物件，然後移除儲存貯體。 如需詳細資訊，請參閱刪除儲存貯體。	AWS DevOps，DevOps 工程師

在 AWS 資源評估工具自動化上設定其他功能

任務	Description	所需的技能
監控並清除手動建立的資源。	（選用）如果您的組織有使用 IaC 工具建立資源的合規要求，您可以使用 AWS 資源評估工具自動化來擷取手動佈建的資源，以實現合規。您也可以使用 工具將資源匯入 IaC 工具，或重新建立資源。若要監控手動佈建的資源，請執行下列高階任務： 1. 部署 AWS 資源評估工具自動化。 2. 設定 Lambda 函數以每天查詢 Athena 資料表、尋找有關手動佈建資源的相關資料，並將其匯出至逗號分隔值 (CSV) 檔案。	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	3. Lambda 函數執行後，會將包含所需資料的通知傳送給個別利益相關者。 4. 對於更長的保留期，.csv 檔案可以存放在 S3 儲存貯體中。 5. 根據 .csv 檔案中的資訊，刪除手動建立的資源，或將其匯入現有的 IaC 解決方案。	

疑難排解

問題	解決方案
AWS CDK 傳回錯誤。	如需 AWS CDK 問題的協助，請參閱疑難排解常見的 AWS CDK 問題。

相關資源

- 使用 Python 建置 Lambda 函數
- 開始使用 AWS CDK
- 在 Python 中使用 AWS CDK
- 建立 CloudTrail 日誌追蹤
- Amazon Quick Sight 入門

其他資訊

多個帳戶

若要設定多個帳戶的 AWS CLI 登入資料，請使用 AWS 設定檔。如需詳細資訊，請參閱設定 AWS CLI 中的設定多個設定檔一節。

AWS CDK 命令

使用 AWS CDK 時，請記住下列有用的命令：

- 列出應用程式中的所有堆疊

```
cdk ls
```

- 發出合成的 AWS CloudFormation 範本

```
cdk synth
```

- 將堆疊部署到您的預設 AWS 帳戶和區域

```
cdk deploy
```

- 比較已部署堆疊與目前狀態

```
cdk diff
```

- 開啟 AWS CDK 文件

```
cdk docs
```

使用開放原始碼工具自動安裝 SAP 系統

Guilherme Sesterheim，Amazon Web Services

總結

此模式說明如何使用開放原始碼工具來建立下列資源，以自動化 SAP 系統安裝：

- SAP S/4HANA 1909 資料庫
- SAP ABAP 中央服務 (ASCS) 執行個體
- SAP 主要應用程式伺服器 (PAS) 執行個體

HashiCorp Terraform 會建立 SAP 系統的基礎設施，而 Ansible 會設定作業系統 (OS) 並安裝 SAP 應用程式。Jenkins 會執行 安裝。

此設定會將 SAP 系統安裝轉換為可重複的程序，這有助於提高部署效率和品質。

> **Note**
>
> 此模式中提供的範例程式碼適用於高可用性 (HA) 系統和非 HA 系統。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 包含所有 SAP 媒體檔案的 Amazon Simple Storage Service (Amazon S3) 儲存貯體
- 具有存取金鑰和私密金鑰，且具有下列許可的 AWS Identity and Access Management (IAM) 主體：
  - 唯讀許可：Amazon Route 53、AWS Key Management Service (AWS KMS)
  - 讀取和寫入許可：Amazon S3、Amazon Elastic Compute Cloud (Amazon EC2)、Amazon Elastic File System (Amazon EFS)、IAM、Amazon CloudWatch、Amazon DynamoDB
- Route 53 私有託管區域
- 在 Amazon Marketplace 中使用 HA 和更新服務 8.2 Amazon Machine Image (AMI) 的 Red Hat Enterprise Linux for SAP 訂閱
- AWS KMS 客戶受管金鑰
- 安全殼層 (SSH) 金鑰對

- Amazon EC2 安全群組，允許從您安裝 Jenkins 的主機名稱（主機名稱很可能是 localhost) 在連接埠 22 上進行 SSH 連線
- HashiCorp 的 Vagrant 已安裝並設定
- 已安裝並設定 VirtualBox by Oracle
- 熟悉 Git、Terraform、Ansible 和 Jenkins

限制

- 只有 SAP S/4HANA 1909 已針對此特定案例進行完整測試。如果您使用其他版本的 SAP HANA，此模式中的範例 Ansible 程式碼需要修改。
- 此模式中的範例程序適用於 Mac OS 和 Linux 作業系統。某些命令只能在 Unix 型終端機中執行。不過，您可以使用不同的命令和 Windows 作業系統來達成類似的結果。

產品版本

- SAP S/4HANA 1909
- Red Hat Enterprise Linux (RHEL) 8.2 或更新版本

Architecture

下圖顯示使用開放原始碼工具自動化 AWS 帳戶中 SAP 系統安裝的範例工作流程：

該圖顯示以下工作流程：

1. Jenkins 透過執行 Terraform 和 Ansible 程式碼來協調執行 SAP 系統安裝。
2. Terraform 程式碼會建置 SAP 系統的基礎設施。
3. Ansible 程式碼會設定作業系統並安裝 SAP 應用程式。
4. Amazon EC2 執行個體上安裝 SAP S/4HANA 1909 資料庫、ASCS 執行個體和包含所有已定義先決條件的 PAS 執行個體。

> **Note**
>
> 此模式中的範例設定會自動在您的 AWS 帳戶中建立 Amazon S3 儲存貯體，以存放 Terraform 狀態檔案。

技術堆疊

- Terraform
- Ansible
- Jenkins
- SAP S/4HANA 1909 資料庫
- SAP ASCS 執行個體
- SAP PAS 執行個體
- Amazon EC2

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，並快速向上或向下擴展。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以保護資料。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路類似於您在自己的資料中心內操作的傳統網路，具有使用可擴展的 AWS 基礎設施的優勢。

其他工具

- HashiCorp Terraform 是一種命令列界面應用程式，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。
- Ansible 是一種開放原始碼組態即程式碼 (CaC) 工具，可協助自動化應用程式、組態和 IT 基礎設施。
- Jenkins 是一種開放原始碼自動化伺服器，可讓開發人員建置、測試和部署其軟體。

Code

此模式的程式碼可在 GitHub aws-install-sap-with-jenkins-ansible 儲存庫中使用。

史詩

設定先決條件

任務	Description	所需的技能
將您的 SAP 媒體檔案新增至 Amazon S3 儲存貯體。	建立包含所有 SAP 媒體檔案的 Amazon S3 儲存貯體。 **Important** 請務必遵循啟動精靈文件中適用於 S/4HANA 的 AWS Launch Wizard 資料夾階層。 https://docs.aws.amazon.com/launchwizard/latest/userguide/launch-wizard-sap-software-install-details.html	雲端管理員
安裝 VirtualBox。	安裝和設定 VirtualBox by Oracle。	DevOps 工程師
安裝 Vagrant。	安裝和設定 HashiCorp 的 Vagrant。	DevOps 工程師
設定您的 AWS 帳戶。	1. 確認您有具有存取金鑰和私密金鑰的 IAM 主體，且具有下列許可： • 唯讀許可：Amazon Route 53、AWS Key Management Service (AWS KMS) • 讀取和寫入許可：Amazon	一般 AWS

任務	Description	所需的技能
	S3、Amazon Elastic Compute Cloud (Amazon EC2)、Amazon Elastic File System (Amazon EFS)、IAM、Amazon CloudWatch、Amazon DynamoDB 2. 儲存 IAM 主體的存取金鑰和私密金鑰以供日後參考。 3. 如果您還沒有 Route 53 私有託管區域，請建立該區域。儲存區域名稱（例如 sapteam.net) 以供日後參考。 4. 訂閱 Red Hat Enterprise Linux for SAP with HA and Update Services 8.2 AMI in Amazon Marketplace。儲存 AMI ID（例如 ami-0000000) 以供日後參考。 5. 建立 AWS KMS 客戶受管金鑰。儲存 KMS 金鑰的 Amazon Resource Name (ARN) 以供日後參考。 **Note** 以下是 AWS KMS 客戶受管金鑰 ARN 範例：arn：aws：kms:us-east-1：123412341234：key/uuid	

任務	Description	所需的技能
	6. 建立 SSH 金鑰對。儲存金鑰對的名稱和 .pem 檔案以供日後參考。 7. 建立 Amazon EC2 安全群組，允許從您安裝 Jenkins 的主機名稱對連接埠 22 進行 SSH 連線。儲存安全群組 ID 以供日後參考。 **Note** 主機名稱很可能是 localhost。	

建置並執行 SAP 安裝

任務	Description	所需的技能
從 GitHub 複製程式碼儲存庫。	在 GitHub 上複製 aws-install-sap-with-jenkins-ansible 儲存庫。	DevOps 工程師
啟動 Jenkins 服務。	開啟 Linux 終端機。然後，導覽至包含複製程式碼儲存庫資料夾的本機資料夾，並執行下列命令： `sudo vagrant up` **Note** Jenkins 啟動大約需要 20 分鐘。命令會傳	DevOps 工程師

任務	Description	所需的技能
	回服務啟動，並在成功時執行訊息。	
在 Web 瀏覽器中開啟 Jenkins 並登入。	1. 在 Web 瀏覽器中，輸入 http：//localhost：5555。Jenkins 開啟。 2. 使用使用者名稱的 admin 和密碼的 my_secret_pass_from_vault 登入 Jenkins。	DevOps 工程師

任務	Description	所需的技能
設定 SAP 系統安裝參數。	1. 在 Jenkins 中，選擇管理 Jenkins。然後，選擇管理登入資料。您可以設定的登入資料變數清單隨即出現。 2. 設定下列所有登入資料變數： • 針對 AWS_ACCOUNT_CREDENTIALS，輸入 IAM 主體的存取金鑰 ID 和私密存取金鑰 ID。 • 針對 AMI_ID，輸入 Red Hat Enterprise Linux for SAP 搭配 HA 和更新服務 8.2 AMI 的 AMI ID。 • 對於 KMS_KEY_ARN，輸入您的 AWS KMS 客戶受管金鑰的 ARN。 • 對於 SSH_KEYPAIR_NAME，輸入 SSH 金鑰對的名稱，而不輸入 .pem 檔案類型。 • 對於 SSH_KEYPAIR_FILE，輸入金鑰對的 .pem 檔案（例如 mykeypair.pem) 的完整名稱。請確定您也將金鑰對的 .pem 檔案上傳至 Jenkins。 • 對於 S3_ROOT_FOLDER_INSTALL_FILES，輸入包含 SAP 媒體檔案的 Amazon S3 儲存貯體和資料	AWS 系統管理員、DevOps 工程師

任務	Description	所需的技能
	夾的名稱（例如 s3：//my-media-bucket/S4H1909)。 • 針對 PRIVATE_DNS_ZONE_NAME，輸入 Route 53 私有託管區域的名稱（例如 myprivatecompanyurl.net)。 • 針對 VPC_ID，輸入您要建立 SAP 資源之 Amazon VPC 的 VPC ID（例如 vpc-12345)。 • 對於 SUBNET_IDS，如果您在測試環境中工作（適用於未來的 HA 功能），請輸入兩個公有子網路 IDs。如果您在生產環境中工作，最佳實務是搭配堡壘主機使用兩個私有子網路。 • 針對 SECURITY_GROUP_ID，輸入 Amazon EC2 安全群組的 ID，該群組允許從您安裝 Jenkins 的主機名稱對連接埠 22 進行 SSH 連線。 **Note** 您可以根據您的使用案例，視需要設定其他非必要參數。例如，您可以變更執行個體的 SAP 系統 ID (SID)、預設密碼、名稱和 SAP 系統標籤。所有必要	

任務	Description	所需的技能
	的變數名稱開頭都有（必要）。	
執行 SAP 系統安裝。	1. 在 Jenkins 中，選擇 Jenkins 首頁。然後，選擇 SAP Hana+ASCS+PAS 3 執行個體。 2. 選擇啟動並安裝。然後，選擇主要。 3. 立即選擇建置。 如需管道步驟的資訊，請參閱 AWS 部落格上的了解使用開放原始碼工具自動化 SAP 安裝的管道步驟一節。 **Note** 如果發生錯誤，請將游標移到出現的紅色錯誤方塊，然後選擇日誌。出現發生錯誤之管道步驟的日誌。大多數錯誤是因為參數設定不正確而發生。	DevOps 工程師、AWS 系統管理員

相關資源

- DevOps for SAP – SAP 安裝：從 2 個月到 2 小時 (DevOps Enterprise Summit Video Library)

使用 AWS CDK 自動化 AWS Service Catalog 產品組合和產品部署

Sandeep Gawande、Vyoma Sachdeva 和 RAJNEESH TYAGI，Amazon Web Services

總結

AWS Service Catalog 可協助您集中管理已核准在組織的 AWS 環境中使用的 IT 服務或產品的目錄。產品組合稱為產品組合，而產品組合也包含組態資訊。使用 AWS Service Catalog，您可以為組織中的每種類型的使用者建立自訂產品組合，然後授予適當產品組合的存取權。這些使用者可以從產品組合中快速部署所需的任何產品。

如果您有複雜的聯網基礎設施，例如多區域和多帳戶架構，建議您在單一中央帳戶中建立和管理 Service Catalog 產品組合。此模式說明如何使用 AWS Cloud Development Kit (AWS CDK) 在中央帳戶中自動建立 Service Catalog 產品組合、授予最終使用者對它們的存取權，然後選擇性地在一或多個目標 AWS 帳戶中佈建產品。此ready-to-use型解決方案會在來源帳戶中建立 Service Catalog 產品組合。它也可以選擇性地使用 AWS CloudFormation 堆疊在目標帳戶中佈建產品，並協助您為產品設定 TagOptions：

- AWS CloudFormation StackSets – 您可以使用 StackSets 跨多個 AWS 區域和帳戶啟動 Service Catalog 產品。在此解決方案中，您可以選擇在部署此解決方案時自動佈建 產品。如需詳細資訊，請參閱使用 AWS CloudFormation StackSets (Service Catalog 文件） 和 StackSets 概念 (CloudFormation 文件）。
- TagOption 程式庫 – 您可以使用 TagOption 程式庫來管理佈建產品的標籤。TagOption 是在 AWS Service Catalog 中管理的鍵/值對。它不是 AWS 標籤，但可做為根據 TagOption 建立 AWS 標籤的範本。如需詳細資訊，請參閱 TagOption 程式庫 (Service Catalog 文件）。

先決條件和限制

先決條件

- 您要用作來源帳戶的作用中 AWS 帳戶，用於管理 Service Catalog 產品組合。
- 如果您使用此解決方案在一或多個目標帳戶中佈建產品，則目標帳戶必須已存在且處於作用中狀態。
- 存取 AWS Service Catalog、AWS CloudFormation 和 AWS IAM 的 AWS Identity and Access Management (IAM) 許可。 AWS Service Catalog AWS CloudFormation

產品版本

- AWS CDK 2.27.0 版

Architecture

目標技術堆疊

- 集中式 AWS 帳戶中的 Service Catalog 產品組合
- 部署在目標帳戶中的 Service Catalog 產品

目標架構

1. 在產品組合 （或來源) 帳戶中，您可以使用使用案例的 AWS 帳戶、AWS 區域、IAM 角色、產品組合和產品資訊來更新 config.json 檔案。
2. 您可以部署 AWS CDK 應用程式。
3. AWS CDK 應用程式會擔任部署 IAM 角色，並建立 config.json 檔案中定義的 Service Catalog 產品組合和產品。

   如果您將 StackSets 設定為在目標帳戶中部署產品，則程序會繼續。如果您未設定 StackSets 來佈建任何產品，則程序已完成。
4. AWS CDK 應用程式會擔任 StackSet 管理員角色，並部署您在 config.json 檔案中定義的 AWS CloudFormation 堆疊集。
5. 在目標帳戶中，StackSets 會擔任 StackSet 執行角色並佈建產品。

工具

AWS 服務

- AWS 雲端開發套件 (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。
- AWS CDK Toolkit 是命令列雲端開發套件，可協助您與 AWS CDK 應用程式互動。
- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶和區域的生命週期中管理這些資源。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Service Catalog 可協助您集中管理針對 AWS 核准的 IT 服務目錄。最終使用者可在機構所設的限制範圍內，迅速地只部署自己需要且經核准的 IT 服務。

程式碼儲存庫

此模式的程式碼可在 aws-cdk-servicecatalog-automation 儲存庫的 GitHub 上取得。程式碼儲存庫包含下列檔案和資料夾：

- cdk-sevicecatalog-app – 此資料夾包含此解決方案的 AWS CDK 應用程式。
- config – 此資料夾包含 config.json 檔案和 CloudFormation 範本，用於部署 Service Catalog 產品組合中的產品。
- config/config.json – 此檔案包含所有組態資訊。您可以更新此檔案，為您的使用案例自訂此解決方案。
- config/templates – 此資料夾包含 Service Center 產品的 CloudFormation 範本。
- setup.sh – 此指令碼部署解決方案。
- uninstall.sh – 此指令碼會刪除部署此解決方案時建立的堆疊和所有 AWS 資源。

若要使用範例程式碼，請遵循 Epics 區段中的指示。

最佳實務

- 用於部署此解決方案的 IAM 角色應遵循最低權限 (IAM 文件） 原則。
- 遵守使用 AWS CDK 開發雲端應用程式的最佳實務 (AWS 部落格文章）。
- 遵守 AWS CloudFormation 最佳實務 (CloudFormation 文件）。

史詩

設定您的環境

任務	Description	所需的技能
安裝 AWS CDK Toolkit。	請確定您已安裝 AWS CDK Toolkit。輸入下列命令以確認是否已安裝並檢查版本。 `cdk --version`	AWS DevOps，DevOps 工程師

任務	Description	所需的技能

	如果未安裝 AWS CDK Toolkit，請輸入下列命令來安裝它。
	`npm install -g aws-cdk@2.27.0`
	如果 AWS CDK Toolkit 版本早於 2.27.0，請輸入下列命令將其更新至 2.27.0 版。
	`npm install -g aws-cdk@2.27.0 --force`	

複製儲存庫。	輸入以下命令。在其他資訊區段中的複製儲存庫中，您可以複製包含儲存庫 URL 的完整命令。這會從 GitHub 複製 aws-cdk-servicecatalog-automation 儲存庫。
	`git clone <repository-URL>.git`
	這會在目標目錄中建立`cd aws-cdk-servicecatalog-automation` 資料夾。輸入下列命令以導覽至此資料夾。
	`cd aws-cdk-servicecatalog-automation`
		AWS DevOps，DevOps 工程師

任務	Description	所需的技能

設定 AWS 登入資料。	輸入下列命令：這些匯出下列變數，定義您要部署堆疊的 AWS 帳戶和區域。
	`export CDK_DEFAULT_ACCOUNT=<12-digit AWS account number>`
	`export CDK_DEFAULT_REGION=<AWS Region>`
	AWS CDK 的 AWS 登入資料是透過環境變數提供。
		AWS DevOps，DevOps 工程師

設定最終使用者 IAM 角色的許可。	如果您要使用 IAM 角色來授予產品組合及其產品存取權，則這些角色必須具有由 servicecatalog.amazonaws.com 服務主體擔任的許可。如需如何授予這些許可的指示，請參閱使用 Service Catalog 啟用受信任存取 (AWS Organizations 文件）。	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
設定 StackSets 所需的 IAM 角色。	如果您使用 StackSets 在目標帳戶中自動佈建產品，則需要設定管理和執行堆疊集的 IAM 角色。 1. 在來源帳戶中，確認 是否`AWSCloudFormationStackSetAdministrationRole` 已存在。在目標帳戶中，確認`AWSCloudFormationStackSetExecutionRole` 是否已存在。如果這些角色已存在，您可以跳到下一個史詩。 2. 遵循授予自我管理許可 (IAM 文件） 中的指示，在產品組合帳戶中建立堆疊集管理角色，並在每個目標帳戶中建立執行角色。	AWS DevOps，DevOps 工程師

自訂和部署解決方案

任務	Description	所需的技能
建立 CloudFormation 範本。	在 `config/templates` 資料夾中，為您要包含在產品組合中的任何產品建立 CloudFormation 範本。如需詳細資訊，請參閱使用 AWS CloudFormation 範本 (CloudFormation 文件）。	應用程式開發人員、AWS DevOps、DevOps 工程師
自訂組態檔案。	在 `config`資料夾中，開啟 config.json 檔案，並根據您的	應用程式開發人員、DevOps 工程師、AWS DevOps

任務	Description	所需的技能
	使用案例定義適當的參數。除非另有說明，否則需要下列參數： 在 `portfolios` 區段中，定義下列參數以建立一或多個 Service Catalog 產品組合： • `portfolioName` – 產品組合的名稱。 • `providerName` – 管理產品組合的人員、團隊或組織的名稱。 • `description` – 產品組合的簡短描述。 • `roles` –（選用）應可存取此產品組合之任何 IAM 角色的名稱。具有此角色的使用者可以存取此產品組合中的產品。 • `users` –（選用）應可存取此產品組合及其產品的任何 IAM 使用者的名稱。 • `groups` –（選用）應可存取此產品組合及其產品的任何 IAM 使用者群組名稱。 **Warning** IAM 使用者具有長期憑證，這會造成安全風險。為了協助降低此風險，建議您只為這些使用者提供執行任務所	

任務	Description	所需的技能

	需的許可，並在不再需要這些使用者時將其移除。
	
	Important
	`roles`、`users`和`groups` 都是選用參數，但如果您未定義其中一個參數，則沒有人可以在 Service Catalog 主控台中檢視產品組合產品。至少定義其中一個參數。如需詳細資訊，請參閱授予 Service Catalog 最終使用者的許可 (Service Catalog 文件）。
	
	• （選用）在 `tagOption` 區段中，定義產品的 TagOptions：
	  • `key` – TagOption 金鑰的名稱
	  • `value` – TagOption 允許的字串值
	
	如需詳細資訊，請參閱 TagOption 程式庫 (Service Catalog 文件）。
	• 在 `products`區段中，定義產品的下列參數：	

任務	Description	所需的技能

	• `portfolioName` – 您要新增產品的產品組合名稱。您只能指定一個產品組合。
	• `productName` – 產品的名稱。
	• `owner` – 產品的擁有者。
	• `productVersionName` – 字串值中產品版本的名稱，例如 `v1`。
	• `templatePath` – 產品的 CloudFormation 範本的檔案路徑。
	• `deployWithStackSets` –（選用）指定您要使用 StackSets 在產品組合中自動佈建產品的一或多個帳戶和區域。如果您使用此部署選項，則需要本節中的所有下列參數：
	  • `accounts` – 目標帳戶。
	  • `regions` – 目標區域。
	  • `stackSetAdministrationRoleName` – 用於管理 StackSets 組態的 IAM 角色名稱。請不要變更此值。此角色必須具有此確切名稱。
	  • `stackSetExecutionRoleName` – 部署堆疊	

任務	Description	所需的技能
	執行個體的目標帳戶中的 IAM 角色名稱。請不要變更此值。此角色必須具有此確切名稱。 如需已完成組態檔案的範例，請參閱其他資訊區段中的範例組態檔案。	
部署解決方案。	輸入以下命令。這會部署 AWS CDK 應用程式，並佈建 Service Catalog 產品組合和產品，如 config.json 檔案中所指定。 `sh +x setup.sh`	應用程式開發人員、DevOps 工程師、AWS DevOps

任務	Description	所需的技能
驗證部署。	執行下列動作來驗證部署是否成功： 1. 使用登入資料登入 AWS 管理主控台，該登入資料可存取您在組態檔案中定義的一或多個產品組合。 2. 在 https：//https://console.aws.amazon.com/servicecatalog/ 開啟 Service Catalog 主控台。 3. 在導覽窗格的佈建下，選擇產品。確認您看到您為產品組合指定的產品清單。 4. 遵循啟動產品 (Service Catalog 文件）中的指示來啟動其中一個可用的產品。確認可用的產品版本和標籤符合您在組態檔案中提供的值。 5. 如果您選擇使用 StackSets 在一或多個目標帳戶中自動佈建產品，請執行下列動作： a. 使用登入資料登入，該登入資料可讓您檢視其中一個目標帳戶中的佈建產品。 b. 在 Service Catalog 主控台的導覽窗格中，於佈建下，選擇佈建產品。 c. 確認預期的產品出現在清單中。	一般 AWS

任務	Description	所需的技能
（選用）更新產品組合和產品。	如果您想要使用此解決方案來更新產品組合或產品，或佈建新產品： 1. 在 config.json 檔案中進行必要的變更。 2. 視需要在 `config/template` 資料夾中新增或修改任何 CloudFormation 範本。 3. 重新部署解決方案。 例如，您可以新增其他產品組合或佈建更多資源。AWS CDK 應用程式只會實作變更。如果先前部署的產品組合或產品沒有變更，則重新部署不會影響它們。	應用程式開發人員、DevOps 工程師、一般 AWS

清除解決方案

任務	Description	所需的技能
（選用）移除此解決方案部署的 AWS 資源。	如果您想要刪除佈建產品，請遵循刪除佈建產品 (Service Catalog 文件）中的指示。 如果您想要刪除此解決方案建立的所有資源，請輸入下列命令。 `sh uninstall.sh`	AWS DevOps、DevOps 工程師、應用程式開發人員

相關資源

- AWS Service Catalog Construct Library (AWS API 參考)
- StackSets 概念 (CloudFormation 文件)
- AWS Service Catalog (AWS 行銷)
- 搭配 AWS CDK 使用 Service Catalog (AWS 研討會)

其他資訊

複製儲存庫

輸入下列命令，從 GitHub 複製儲存庫。

```
git clone https://github.com/aws-samples/aws-cdk-servicecatalog-automation.git
```

範例組態檔案

以下是具有範例值的範例 config.json 檔案。

```
{
    "portfolios": [
        {
            "displayName": "EC2 Product Portfolio",
            "providerName": "User1",
            "description": "Test1",
            "roles": [
                "<Names of IAM roles that can access the products>"
            ],
            "users": [
                "<Names of IAM users who can access the products>"
            ],
            "groups": [
                "<Names of IAM user groups that can access the products>"
            ]
        },
        {
            "displayName": "Autoscaling Product Portfolio",
            "providerName": "User2",
            "description": "Test2",
            "roles": [
                "<Name of IAM role>"
```

```
            ]
        }
    ],
    "tagOption": [
        {
            "key": "Group",
            "value": [
                "finance",
                "engineering",
                "marketing",
                "research"
            ]
        },
        {
            "key": "CostCenter",
            "value": [
                "01",
                "02",
                "03",
                "04"
            ]
        },
        {
            "key": "Environment",
            "value": [
                "dev",
                "prod",
                "stage"
            ]
        }
    ],
    "products": [
        {
            "portfolioName": "EC2 Product Profile",
            "productName": "Ec2",
            "owner": "owner1",
            "productVersionName": "v1",
            "templatePath": "../../config/templates/template1.json"
        },
        {
            "portfolioName": "Autoscaling Product Profile",
            "productName": "autoscaling",
            "owner": "owner1",
            "productVersionName": "v1",
```

```
            "templatePath": "../../config/templates/template2.json",
            "deployWithStackSets": {
                "accounts": [
                    "012345678901",
                ],
                "regions": [
                    "us-west-2"
                ],
                "stackSetAdministrationRoleName":
 "AWSCloudFormationStackSetAdministrationRole",
                "stackSetExecutionRoleName": "AWSCloudFormationStackSetExecutionRole"
            }
        }
    ]
}
```

使用 AWS Service Catalog 和 自動化動態管道管理，以在 Gitflow 環境中部署 Hotfix 解決方案 AWS CodePipeline

Balaji Vedagiri、Faisal Shahdad、Shanmugam Shanker 和 Vivek Thangamuthu，Amazon Web Services

摘要

> **Note**
>
> AWS CodeCommit 不再提供給新客戶。的現有客戶 AWS CodeCommit 可以繼續正常使用服務。進一步了解

此模式解決了管理動態 Hotfix 管道的案例，該管道專門用於將 Hotfix 解決方案安全地部署到生產環境。解決方案是透過使用 AWS Service Catalog 產品組合和產品來實作和管理。Amazon EventBridge 規則用於事件自動化。限制是透過為開發人員使用 Service Catalog 產品組合限制條件和 AWS Identity and Access Management (IAM) 角色來強制執行。僅允許 AWS Lambda 函數啟動由 EventBridge 規則觸發的 Service Catalog 產品。此模式專為具有特定 Gitflow 設定的環境而設計，如其他資訊中所述。

一般而言，會部署 Hotfix 來解決即時環境中報告的關鍵或安全問題，例如生產。應僅將 Hotfix 直接部署到預備和生產環境。預備和生產管道廣泛用於定期開發請求。這些管道無法用來部署 Hotfix，因為品質保證中有持續的功能無法提升為生產。若要釋出 Hotfix，此模式會描述具有下列安全功能的動態、短期管道：

- 自動建立 – 只要在 AWS CodeCommit 儲存庫中建立 Hotfix 分支，就會自動建立 Hotfix 管道。
- 存取限制 – 開發人員無法存取 在 Hotfix 程序之外建立此管道。
- 受控階段 – 管道具有具有特殊存取字符的受控階段，確保提取請求 (PR) 只能建立一次。
- 核准階段 – 核准階段包含在管道中，以取得相關利益相關者的必要核准。
- 自動刪除 – 每當在 CodeCommit 儲存庫中將`hotfix`分支與 PR 合併後刪除，就會自動刪除 Hotfix 管道。

先決條件和限制

先決條件

- AWS 帳戶 需要三個作用中的 ，如下所示：

  - 工具帳戶 - 用於持續整合和持續交付 (CI/CD) 設定。
  - 階段帳戶 - 用於使用者接受度測試。
  - 生產帳戶 - 適用於企業最終使用者。
  - （選用）新增 AWS 帳戶 以做為 QA 帳戶。如果您想要同時使用主要管道設定，包括 QA 和 Hotfix 管道解決方案進行測試，則需要此帳戶。
- 具有選用條件的 AWS CloudFormation 堆疊，可視需要使用主要管道在 QA 帳戶中部署。透過建立和刪除`hotfix`分支，仍然可以在沒有主要管道設定的情況下測試模式。
- Amazon Simple Storage Service (Amazon S3) 儲存貯體，用於存放用於建立 Service Catalog 產品的 CloudFormation 範本。
- 根據合規要求建立 CodeCommit 儲存庫的 PR 核准規則（建立儲存庫之後）。
- 限制開發人員和團隊的 IAM 許可會導致拒絕 prcreation-lambda Lambda 函數的執行，因為它應該只從管道叫用。

限制

- CloudFormation 提供者用於部署階段，而應用程式則是使用 CloudFormation 變更集進行部署。如果您想要使用不同的部署選項，請視需要修改 CodePipeline 堆疊。
- 此模式使用 AWS CodeBuild 和其他組態檔案來部署範例微服務。如果您有不同的工作負載類型（例如，無伺服器工作負載），則必須更新所有相關組態。
- 此模式會在單一 AWS 區域（例如，美國東部（維吉尼亞北部）us-east-1) 中部署應用程式 AWS 帳戶。若要跨多個區域部署，請在命令和堆疊中變更區域參考。
- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱依區域的 AWS 服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

本節中的圖表提供建立生命週期事件和刪除生命週期事件的工作流程。

上述用於建立生命週期事件的圖表顯示下列項目：

1. 開發人員會在 CodeCommit 儲存庫中建立`hotfix-*`分支，以開發修正程式相關的解決方案。
2. `hotfix-*` 分支建立事件是透過 EventBridge 規則擷取。事件詳細資訊包括儲存庫名稱和分支名稱。

3. EventBridge 規則會叫用 AWS Lambda 函數 `hotfix-lambda-function`。EventBridge 規則會將事件資訊傳遞給 Lambda 函數做為輸入。
4. Lambda 函數會處理輸入以擷取儲存庫名稱和分支名稱。它會使用從處理過的輸入擷取的值來啟動 Service Catalog 產品。
5. Service Catalog 產品包含管道設定，可將解決方案部署至階段和生產環境。管道區塊包含來源、建置和部署階段。此外，還有手動核准階段來提升生產環境的部署。
6. 來源階段會從第一個步驟中建立的儲存庫和`hotfix-*`分支擷取程式碼。程式碼會透過用於成品的 Amazon S3 儲存貯體傳遞至建置階段。在建置階段，會建立容器映像，其中包含在`hotfix-*`分支中開發並推送至 Amazon Elastic Container Registry (Amazon ECR) 的 Hotfix。
7. 階段環境的部署階段會使用包含 Hotfix 的最新容器映像來更新 Amazon Elastic Container Service (Amazon ECS)。透過建立和執行 CloudFormation 變更集來部署 Hotfix。
8. 在階段環境中成功部署後，會叫用 `prcreation-lambda` Lambda 函數。此 Lambda 函數會建立從`hotfix-*`分支到儲存庫 `develop`和 `main`分支的 PR。Lambda 函數可確保在`hotfix-*`分支中開發的修正會反向合併並包含在後續部署中。
9. 手動核准階段有助於確保必要的利益相關者檢閱修正，並核准在生產環境中部署。
10. 生產環境的部署階段會使用包含 Hotfix 的最新容器映像來更新 Amazon ECS。透過建立和執行 CloudFormation 變更集來部署 Hotfix。

上述刪除生命週期事件的圖表顯示下列項目：

1. 開發人員成功將 Hotfix 部署至生產環境後，會刪除`hotfix-*`分支。
2. `hotfix-*` 分支刪除事件是透過 EventBridge 規則擷取。事件詳細資訊包括儲存庫名稱和分支名稱。
3. EventBridge 規則會叫用 Lambda 函數。EventBridge 規則會將事件資訊傳遞給 Lambda 函數做為輸入。
4. Lambda 函數會處理輸入以擷取儲存庫名稱和分支名稱。Lambda 函數會從傳遞的輸入決定個別的 Service Catalog 產品，然後終止產品。
5. Service Catalog 佈建產品終止會刪除先前在該產品中建立的管道和相關資源。

自動化和擴展

- 模式包含 EventBridge 規則和 Lambda 函數，可平行處理多個 Hotfix 分支建立請求。Lambda 函數會為相符的事件規則佈建 Service Catalog 產品。

- 管道設定是使用 Service Catalog 產品來處理，其提供版本控制功能。解決方案也會自動擴展，以平行處理相同應用程式的多個 Hotfix 開發。
- prcreation-lambda 函數可確保這些 Hotfix 變更也會透過自動提取請求建立合併回 `main`和`develop`分支。此方法對於讓 `main`和`develop`分支與所有修正保持最新狀態，並避免潛在的程式碼迴歸至關重要。此程序可確保所有長期分支都有最新的修正，以協助維持分支間的一致性，並防止程式碼迴歸。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。
- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。 AWS CodeCommit 不再可供新客戶使用。的現有客戶 AWS CodeCommit 可以繼續正常使用服務。如需詳細資訊，請參閱如何將 AWS CodeCommit 儲存庫遷移至另一個 Git 供應商。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- Amazon Elastic Container Service (Amazon ECS) 是快速、可擴展的容器管理服務，可協助您執行、停止和管理叢集上的容器。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- AWS Service Catalog 可協助您集中管理已核准的 IT 服務目錄 AWS。最終使用者可在機構所設的限制範圍內，迅速地只部署自己需要且經核准的 IT 服務。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他工具

- CloudFormation Linter (cfn-lint) 是一種 linter，可根據 CloudFormation 資源規格檢查 CloudFormation YAML 或 JSON 範本。它也會執行其他檢查，例如檢查資源屬性的有效值，以及是否遵守最佳實務。
- cfn-nag 是一種開放原始碼工具，可透過搜尋模式來識別 CloudFormation 範本中的潛在安全問題。
- Docker 是一組平台即服務 (PaaS) 產品，可在作業系統層級使用虛擬化在容器中交付軟體。此模式使用 Docker 在本機建置和測試容器映像。
- Git 是開放原始碼的分散式版本控制系統。

程式碼儲存庫

此模式的程式碼可在 GitHub dynamic_hotfix_codepipeline 儲存庫中使用。

最佳實務

檢閱和調整您環境中的 IAM 角色和服務控制政策 (SCP)，以確保它們適當地限制存取。這對於防止任何可以覆寫此模式中包含的安全措施的動作至關重要。遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。

史詩

設定工作環境

任務	Description	所需的技能
複製儲存庫。	若要將範例儲存庫複製到您工作位置中的新目錄，請執行下列命令： `git clone git@github.com:aws-samples/dynamic_hotfix_codepipeline.git`	AWS DevOps
匯出 CloudFormation 堆疊部署的環境變數。	定義下列環境變數，稍後在此模式中用作 CloudFormation 堆疊的輸入。 • `ApplicationName` – 此變數用於命名為應用程式	AWS DevOps

任務	Description	所需的技能

	建立的資源，讓您更輕鬆地追蹤這些資源。使用下列命令，將 取代Applicationname 為您實際的應用程式名稱：
	`export ApplicationName=<Applicationname>`
	• BucketStartName – 此變數用於命名 Amazon S3 儲存貯體。S3 儲存貯體名稱在所有 中必須是全域唯一的 AWS 帳戶。使用下列命令，BucketName 將 取代為 S3 儲存貯體的唯一名稱：
	`export BucketStartName=<BucketName>`
	• 帳戶號碼和區域 – 這些變數存放不同環境和部署區域的 AWS 帳戶 號碼。使用下列命令，將預留位置 （例如 prodaccountnumber 和 region) 取代為您實際 AWS 區域 使用的 AWS 帳戶號碼和 。
	Note
	QAAccount 是選用的。如果您想要使用 QAAccount ，	

任務	Description	所需的技能

	請使用主要管道堆疊的參數進行設定。
	
	`export ProdAccount=<prodaccountnumber>`
	`export StageAccount=<stage/preprodaccountnumber>`
	`export QAAccount=<qaccountnumber>`
	`export ToolsAccount=<toolsaccountnumber>`
	`export DepRegion=<region>`	

在 中設定所需的先決條件 AWS 帳戶

任務	Description	所需的技能

在工具帳戶中建立 CI/CD 所需的資源。	若要在工具帳戶中部署 CloudFormation 堆疊，請使用下列命令。（如果您未使用 QA 帳戶進行設定，請移除 `QAAccount` 參數。）
	
	`#InToolsAccount`
	`aws cloudformation deploy \`
	`--template-file pre-requisites/pre-reqs.yaml \`
	`--stack-name prereqs \`
		AWS DevOps

任務	Description	所需的技能
	--parameter-overrides BucketStartName=${BucketStartName} \ ApplicationName=${ApplicationName} ProdAccount=${ProdAccount} \ StageAccount=${StageAccount} ToolsAccount=${ToolsAccount} \ QAAccount=${QAAccount} \ --capabilities CAPABILITY_IAM CAPABILITY_NAMED_IAM --region ${DepRegion} 請記下 CodeCommit 儲存庫和 Amazon ECR 從上述堆疊建立的資源。在後續步驟中設定管道`main`分支時，需要這些參數。	

任務	Description	所需的技能

在工作負載帳戶中建立 CI/CD 所需的資源。	1. 若要封裝每個工作負載帳戶中的 CloudFormation 範本（階段、生產和選用的 QA)，請使用下列命令。在下列命令中，將 S3bucketpackage 取代為您要用於封裝的 Amazon S3 儲存貯體名稱。
	
	`#InStageAccount`
	`aws cloudformation package \`
	`--template-file pre-requisites/infrastack.yaml \`
	`--s3-bucket <S3bucketpackage> \`
	`--s3-prefix infraStack \`
	`--region ${DepRegion} \`
	`--output-template-file pre-requisites/infrastructure_stage.template`
	
	`#InProdAccount`
	`aws cloudformation package \`
	`--template-file pre-requisites/infrastack.yaml \`
	`--s3-bucket <S3bucketpackage> \`
	`--s3-prefix infraStack \`
	`--region ${DepRegion} \`
		AWS DevOps

任務	Description	所需的技能
	--output-template-file pre-requisites/infrastructure_prod.template 2. 若要在每個工作負載帳戶中部署 CloudFormation 範本，請使用下列命令： #InStageAccount aws cloudformation deploy --stack-name inframainstack \ --parameter-overrides ApplicationName=${ApplicationName} ToolsAccount=${ToolsAccount} DepRegion=${DepRegion} \ --template-file pre-requisites/infrastructure_stage.template --region ${DepRegion} --capabilities CAPABILITY_NAMED_IAM #InProdAccount aws cloudformation deploy --stack-name inframainstack \ --parameter-overrides ApplicationName=${ApplicationName} ToolsAccount=${ToolsAccount} DepRegion=${DepRegion} \	

任務	Description	所需的技能
	`--template-file pre-requisites/infrastructure_prod.template --region ${DepRegion} --capabilities CAPABILITY_NAMED_IAM`	

任務	Description	所需的技能

更新 S3 成品儲存貯體政策，以允許工作負載帳戶的存取。	若要更新工具帳戶中的 CloudFormation 堆疊先決條件，請使用下列命令來新增階段和生產工作負載帳戶的所有必要許可。（如果您未使用 `QAAccount` 參數進行設定，請移除 參數。）
	
	`#InToolsAccount`
	`aws cloudformation deploy \`
	`--template-file pre-requisites/pre-reqs.yaml \`
	`--stack-name prereqs \`
	`--parameter-overrides BucketStartName=${BucketStartName} \`
	`ApplicationName=${ApplicationName} ProdAccount=${ProdAccount} \`
	`StageAccount=${StageAccount} ToolsAccount=${ToolsAccount} \`
	`QAAccount=${QAAccount} PutPolicy=true \`
	`--capabilities CAPABILITY_IAM CAPABILITY_NAMED_IAM --region ${DepRegion}`
		AWS DevOps

在工具帳戶中設定 Lambda 函數和 Service Catalog 資源

任務	Description	所需的技能

設定 Service Catalog 產品組合和產品。	若要設定 Service Catalog 產品組合和產品，請執行下列動作：
	
	1. 將範本 pipeline-main.yaml 和 pipeline-hotfix.yaml 從 CodePipeline 目錄中的儲存庫上傳到您要部署到 (`Bucketname`) 的區域中現有的 Amazon S3 儲存貯體 (`DepRegion`)。
	
	`#InToolsAccount`
	`aws s3 cp ./codepipeline/pipeline-main.yaml s3://<Bucketname>/pipeline-main.yaml`
	`aws s3 cp ./codepipeline/pipeline-hotfix.yaml s3://<Bucketname>/pipeline-hotfix.yaml`
	
	2. 設定 Service Catalog 產品組合和產品，以管理 `main`和 `hotfix`分支的管道。記下`MainProductArtifactId` 下列堆疊中 `MainProductId` 和 `Outputs`區段的詳細資訊。在`main`分支的管道設定期間，後續步驟中需要此資訊。
	
	`#InToolsAccount`
		AWS DevOps

任務	Description	所需的技能

	<pre>aws cloudformation deploy \
	    --template-file pre-requisites/servicecatalogsetup.yaml \
	    --stack-name servicecatalogsetup \
	    --parameter-overrides TemplateBucket=<Bucketname> \
	    --capabilities CAPABILITY_IAM CAPABILITY_NAMED_IAM --region ${DepRegion}</pre>
	3. 提供 IAM 角色的存取權，該角色會將工具帳戶中的資源部署到 Service Catalog 產品組合的主要管道產品組合。使用此產品組合透過使用 `main`分支來部署應用程式。如需如何提供存取權的詳細資訊，請參閱 Service Catalog 文件中的授予使用者存取權。	

任務	Description	所需的技能
設定 Lambda 函數。	此解決方案使用下列 Lambda 函數來管理 Hotfix 工作流程： • `hotfix-lambda-function` 會在建立hotfix分支時處理 Service Catalog 產品佈建。 • `hotfix-cleanup-lambda-function` 會在刪除hotfix分支時管理產品終止。 • `prcreation-lambda` 會建立從hotfix分支到 `develop`和 `main`分支的提取請求。 若要讓 Lambda 函數在透過相關聯的 EventBridge 規則建立或刪除hotfix 分支時佈建和終止 Service Catalog 產品，請使用下列步驟： 1. 若要建立 Lambda 函數的 IAM 角色和許可，請使用下列命令來部署 CloudFormation 堆疊： `#InToolsAccount aws cloudformation deploy \ --template-file pre-requisites/lambdasetup.yaml \ --stack-name prsclambdasetup \ --capabilities CAPABILITY_IAM`	AWS DevOps

任務	Description	所需的技能
	CAPABILITY_NAMED_IAM --region ${DepRegion} 2. 在堆疊部署之後，使用 授予 Service Catalog 產品組合 Hotfix 管道產品組合的hotfix-lambda-execution-role 存取權AWS管理主控台。此存取權可讓 Lambda 函數啟動或終止 Hotfix 分支的 Service Catalog 產品。	

為主要分支建立管道，並在工作負載帳戶中部署應用程式

任務	Description	所需的技能
設定main分支的管道。	若要設定主要分支的管道，請在工具帳戶中執行下列命令。將 MainProductId 和 的參數取代MainProductArtifactId 為servicecatalogsetup 堆疊輸出的值。 #InToolsAccount aws servicecatalog provision-product \ --product-id <MainProductId> \ --provisioning-artifact-id <MainProductArtifactId> \ --provisioned-product-name "${Applic	AWS DevOps

任務	Description	所需的技能
	ationName}-main-pi peline" \ --provisioning- parameters Key=CodeC ommitRepoName,Valu e="${ApplicationNa me}-repository" Key=ECRRepository, Value="${Applicati onName}-app" \ --region=${DepRegi on}	

任務	Description	所需的技能
使用 `main`分支部署應用程式。	1. 若要複製在先決條件中建立的 CodeCommit 儲存庫，請使用 `git clone`命令。 如需詳細資訊，請參閱透過複製儲存庫連線至 CodeCommit 儲存庫，如 CodeCommit 文件所述。 2. 將所有應用程式檔案從儲存庫中可用的`repotemplates` 目錄複製到本機儲存庫複製 (`${ApplicationName}-repository`)。修改下列檔案以更新 `ToolsAccount` ID。在每個檔案中，找到 `RegistryAccountid` 參數並將其值設定為您的 `ToolsAccount` ID。將變更遞交至 CodeCommit 儲存庫，並將檔案同時推送至`main`和 `develop`分支。 • ecs-configuration-prod.yaml • ecs-configuration-qa.yaml • ecs-configuration-stage.yaml 3. 若要驗證應用程式部署，請使用 監控 CodePipeline 執行 AWS 管理主控台。部署完成後，請在階段環境中使用 Application Load Balancer FQDN 存取應用程	AWS DevOps

任務	Description	所需的技能
	式。確認應用程式如預期般運作。 4. 若要核准部署至生產環境，請使用 CodePipeline 主控台來尋找應用程式的管道。尋找`ApprovalToStart` 階段。檢閱變更，如果滿意，請提供手動核准以繼續生產部署。	

建立 Hotfix-* 分支的管道並部署 Hotfix

任務	Description	所需的技能
建立`hotfix-*`分支並遞交變更。	若要為`hotfix-*`分支建立管道並將 Hotfix 部署到工作負載帳戶，請執行下列動作： 1. 使用以關鍵字 開頭的名稱建立分支`hotfix`。例如，此模式使用 CodeCommit 應用程式儲存庫 () 中的`hotfix-check1` 分支`${ApplicationName}-repository` 。如需更詳細的步驟，請參閱 CodeCommit 文件中的連線至 AWS CodeCommit 儲存庫和基本 Git 命令。https://docs.aws.amazon.com/codecommit/latest/userguide/how-to-basic-git.html CodeCommit 2. 確認 Service Catalog 產品`Hotfix CICD`	AWS DevOps

任務	Description	所需的技能
	Pipeline已成功為hotfix-check1 分支動態佈建。佈建的產品名稱以此 Hotfix 分支名稱和應用程式的 CodeCommit 儲存庫名稱命名。 3. 遞交 index.html 檔案中的一些次要變更，並將其推送至 CodeCommit 儲存庫。 4. 確認 CodePipeline 在階段環境中成功執行。若要在生產環境中部署，請在 CodePipeline 中提供手動核准。 5. 使用 Application Load Balancer 完整網域名稱 (FQDN)，確認變更會顯示在應用程式首頁中。FQDN 可在 的 Outputs區段中使用inframainstack-ALB Stack-* 。	
刪除hotfix-check1 分支。	若要刪除先前建立的hotfix-check1 分支，請執行下列動作： 1. 刪除 CodeCommit 應用程式儲存庫中的hotfix-check1 分支。 2. 確認為hotfix-check1 分支佈建的 Service Catalog 產品已成功終止。	AWS DevOps

清除資源

任務	Description	所需的技能

清除已部署的資源。	若要清除先前部署的資源，請執行下列動作：
	1. 若要將工作負載帳戶中的 Amazon ECS 服務縮減為零複本，請使用下列命令：
	`aws ecs update-service --cluster ${ApplicationName}-Cluster --service ${ApplicationName}-Service-stage --desired-count 0 --region ${DepRegion}`
	`aws ecs update-service --cluster ${ApplicationName}-Cluster --service ${ApplicationName}-Service-prod --desired-count 0 --region ${DepRegion}`
	2. 終止為`main`分支佈建的 Service Catalog 產品。
	3. 清除工具帳戶中 Amazon S3 儲存貯體中建立的物件。刪除登錄檔本身之前，請先刪除 Amazon ECR 中的所有 Docker 映像。
	4. 在刪除 Service Catalog 產品組合之前，移除 Service Catalog 產品組合中授予存取權區段中的 IAM 角色。
		AWS DevOps

任務	Description	所需的技能

	5. 刪除工具帳戶和工作負載帳戶中部署的 CloudFormation 堆疊。
	
	`##In Tools Account##`
	`aws cloudformation delete-stack --stack-name servicecatalogsetup --region ${DepRegion}`
	`aws cloudformation delete-stack --stack-name prlambdasetup --region ${DepRegion}`
	`aws cloudformation delete-stack --stack-name prereqs --region ${DepRegion}`
	
	`##In Workload Accounts##`
	`aws cloudformation delete-stack --stack-name inframainstack --region ${DepRegion}`
	
	如需詳細資訊，請參閱 Service Catalog 文件中的刪除佈建的產品。	

疑難排解

問題	解決方案
您遞交給 CodeCommit 儲存庫的變更未部署。	檢查 CodeBuild 日誌是否有 Docker 建置動作中的錯誤。如需詳細資訊，請參閱 CodeBuild 文件。
未佈建 Service Catalog 產品。	檢閱相關 CloudFormation 堆疊是否有失敗的事件。如需詳細資訊，請參閱 CloudFormation 文件。

相關資源

- 基本 Git 命令
- 設定 IAM 政策以限制推送和合併至分支
- 連線至 AWS CodeCommit 儲存庫
- 將存取權授予使用者
- 將 Docker 映像推送至 Amazon ECR 私有儲存庫
- 疑難排解 AWS CodeBuild
- 什麼是 AWS CodePipeline？

其他資訊

此模式專為具有 Gitflow 設定的環境而設計，該設定在 CI/CD 程序中用於開發工作流程。這些管道遵循從開發開始的部署週期，並通過品質保證 (QA)、階段和生產環境。CI/CD 設定包含兩個 git 分支，可對環境進行促銷部署，如下所示：

- `develop` 分支會部署到開發環境。
- `main` 分支會部署到 QA、階段和生產環境。

在此設定中，在持續積極開發新功能時，套用 Hotfix 或安全修補程式的速度比平常的部署週期更快是一項挑戰。需要專用程序來處理 Hotfix 或安全請求，以確保即時環境保持正常運作和安全。

不過，如果符合下列條件，您可以使用其他可用的選項，而不需要專用的部署程序：

- CI/CD 程序配備良好的自動化測試，例如功能測試和end-to-end測試，無需手動測試並防止部署到生產環境的延遲。不過，如果自動化測試未與 CI/CD 程序充分整合，則對開發人員而言，將小型修正推送至生產環境可能會變得複雜且繁瑣。這是因為在 QA 環境中可能有新功能等待核准和簽署。Hotfix 或安全修正無法以直接的方式同時推送至生產環境。
- 開發團隊會持續將新功能部署到生產環境中，將 Hotfix 或安全修補程式整合到每個新功能的排程部署中。換句話說，生產環境的下一個功能更新包含兩個元件：新增新功能，以及包含修正程式或安全修補程式。不過，如果部署週期不連續，則可能有多個新功能已在 QA 環境中等待核准。然後，管理不同的版本並確保重新套用正確的變更可能會變得複雜且容易出錯。

> **Note**
>
> 如果您使用 第 2 版 AWS CodePipeline ，並在`hotfix`分支上設定適當的觸發條件，您仍然需要專用程序來處理未排程的請求。在第 2 版中，您可以設定推送或提取請求的觸發條件。會根據管道的先前狀態，立即將執行排入佇列或執行。不過，透過專用管道，修正會立即套用至生產環境，確保緊急問題不會延遲解決。

自動化刪除 AWS CloudFormation 堆疊和相關聯的資源

SANDEEP SINGH 和 Amazon Web Services 的 James Jacob

總結

AWS CloudFormation 是一項廣泛使用的服務，可用來管理雲端基礎設施即程式碼 (IaC)。當您使用 CloudFormation 時，請以稱為堆疊的單一單位來管理相關資源。意即您能夠建立、更新和刪除堆疊，藉此建立、更新並刪除資源集合。

有時候，您不再需要 CloudFormation 堆疊中的資源。根據資源及其組態，刪除堆疊及其相關聯的資源可能很複雜。在實際生產系統中，由於 CloudFormation 無法覆寫的條件或限制衝突，刪除有時會失敗或需要很長時間。它可能需要仔細的規劃和執行，以確保以高效且一致的方式正確刪除所有資源。此模式說明如何設定架構，以協助您管理刪除涉及下列複雜性的 CloudFormation 堆疊：

- 具有刪除保護的資源 – 有些資源可能已啟用刪除保護。常見範例為 Amazon DynamoDB 資料表和 Amazon Simple Storage Service (Amazon S3) 儲存貯體。刪除保護可防止自動刪除，例如透過 CloudFormation 刪除。如果您想要刪除這些資源，您必須手動或以程式設計方式覆寫或暫時停用刪除保護。在繼續之前，您應該仔細考慮刪除這些資源的含意。
- 具有保留政策的資源 – 某些資源，例如 AWS Key Management Service (AWS KMS) 金鑰和 Amazon S3 儲存貯體，可能具有指定在請求刪除後應保留多久的保留政策。您應該在清除策略中考慮這些政策，以保持符合組織政策和法規要求。
- 延遲刪除連接到 VPC 的 Lambda 函數 – 刪除連接到虛擬私有雲端 (VPC) 的AWS Lambda函數可能需要 5-40 分鐘，取決於程序中涉及的多個互連相依性。如果您在刪除堆疊之前從 VPC 分離函數，您可以將此延遲減少到 1 分鐘以下。
- 非由 CloudFormation 直接建立的資源 – 在某些應用程式設計中，資源可能會在原始 CloudFormation 堆疊之外建立，無論是由應用程式本身或透過堆疊佈建的資源。以下是兩個範例：
  - CloudFormation 可能會佈建執行使用者資料指令碼的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。然後，此指令碼可能會建立 AWS Systems Manager 參數來存放應用程式相關資料。此參數不是透過 CloudFormation 管理。
  - CloudFormation 可能會佈建 Lambda 函數，以自動產生 Amazon CloudWatch Logs 群組來儲存日誌。此日誌群組不是透過 CloudFormation 管理。

  即使這些資源不是由 CloudFormation 直接管理，但通常在刪除堆疊時需要清除這些資源。如果未受管，它們可能會變得孤立，並導致不必要的資源消耗。

雖然這些護欄可能會造成複雜性，但它們是有意且關鍵的。允許 CloudFormation 覆寫所有限制條件和無差別刪除資源，在許多情況下可能會導致不利和不可預見的後果。不過，身為負責管理環境的 DevOps 或雲端工程師，有時可能需要覆寫這些限制，特別是在開發、測試或預備環境中。

目標業務成果

透過實作此架構，您可以實現下列優點：

- 成本管理 – 定期且有效率地清理暫時性環境，例如end-to-end或使用者接受度測試環境，有助於防止資源執行超過必要的時間。這可以大幅降低成本。
- 安全 – 自動清除過時或未使用的資源可減少攻擊面，並有助於維護安全 AWS 的環境。
- 營運效率 – 定期和自動清理可提供下列營運優勢：
  - 移除舊日誌群組或空 Amazon S3 儲存貯體的自動化指令碼可以透過保持環境乾淨且可管理來提高營運效率。
  - 快速刪除和重新建立堆疊支援設計和實作的快速反覆運算，這可能會導致更強大和彈性的架構。
  - 定期刪除和重建環境可協助您識別和修正潛在問題。這可協助您確保基礎設施可以承受真實世界的案例。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- Python 3.6 版或更新版本，已安裝
- AWS Command Line Interface (AWS CLI)，已安裝並設定

限制

- 命名慣例用於識別應刪除的資源。此模式中的範例程式碼使用資源名稱的字首，但您可以定義自己的命名慣例。不會識別未使用此命名慣例的資源，也不會隨後刪除。

Architecture

下圖顯示此架構如何識別目標 CloudFormation 堆疊及其相關聯的其他資源。

該圖顯示以下工作流程：

1. 收集資源 – 自動化架構使用命名慣例來傳回所有相關 CloudFormation 堆疊、Amazon Elastic Container Registry (Amazon ECR) 儲存庫、DynamoDB 資料表和 Amazon S3 儲存貯體。

   > **Note**
   >
   > 此階段的函數使用分頁程式，這是 Boto3 中的一項功能，可抽象化經過截斷 API 結果集的反覆運算程序。這可確保處理所有資源。若要進一步最佳化效能，請考慮套用伺服器端篩選，或考慮使用 JMESPath 來執行用戶端篩選。

2. 預先處理 – 自動化架構可識別並解決必須覆寫的服務限制條件，以允許 CloudFormation 刪除資源。例如，它會將 DynamoDB 資料表`DeletionProtectionEnabled`的設定變更為 `False`。在命令列界面中，針對每個資源，您會收到提示，詢問您是否要覆寫限制條件。
3. 刪除堆疊 – 自動化架構會刪除 CloudFormation 堆疊。在命令列界面中，您會收到提示，詢問您是否要刪除堆疊。
4. 後置處理 – 自動化架構會刪除任何未直接透過 CloudFormation 佈建的相關資源，做為堆疊的一部分。這些資源類型的範例包括 Systems Manager 參數和 CloudWatch 日誌群組。個別函數會收集這些資源、預先處理它們，然後刪除它們。在命令列界面中，針對每個資源，您會收到提示，詢問您是否要刪除資源。

   > **Note**
   >
   > 此階段的函數使用分頁程式，這是 Boto3 中的一項功能，可抽象化經過截斷 API 結果集的反覆運算程序。這可確保處理所有資源。若要進一步最佳化效能，請考慮套用伺服器端篩選，或考慮使用 JMESPath 來執行用戶端篩選。

自動化和擴展

如果您的 CloudFormation 堆疊包含範例程式碼中未包含的其他資源，或者堆疊具有此模式中未解決的限制，則可以針對您的使用案例調整自動化架構。遵循相同的方法來收集資源、預先處理、刪除堆疊，然後進行後續處理。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。

- CloudFormation 命令列界面 (CFN-CLI) 是一種開放原始碼工具，可協助您開發和測試 AWS 和第三方延伸模組，然後註冊它們以在 CloudFormation 中使用。
- 適用於 Python (Boto3) 的 AWS SDK 是一種軟體開發套件，可協助您整合 Python 應用程式、程式庫或指令碼 AWS 服務。

其他工具

- 按一下是 Python 工具，可協助您建立命令列界面。
- Poetry 是一種在 Python 中管理相依性和封裝的工具。
- Pyenv 是一種工具，可協助您管理和切換 Python 版本。
- Python 是一種一般用途的電腦程式設計語言。

程式碼儲存庫

此模式的程式碼可在 GitHub cloudformation-stack-cleanup 儲存庫中使用。

最佳實務

- 標記資源以方便識別 – 實作標記策略來識別為不同環境和目的建立的資源。標籤可協助您根據資源的標籤篩選資源，以簡化清除程序。
- 設定資源生命週期 – 定義資源生命週期，以便在特定期間之後自動刪除資源。此實務可協助您確保暫時環境不會成為永久成本負債。

史詩

安裝工具

任務	Description	所需的技能
複製儲存庫。	1. 在虛擬環境中建立資料夾。將其命名為您的專案名稱。 2. 在本機電腦上開啟終端機，然後導覽至此資料夾。 3. 輸入下列命令，將 cloudformation-stack-cleanu	DevOps 工程師

任務	Description	所需的技能
	p 儲存庫複製到您的專案目錄： `git clone https://github.com/aws-samples/cloudformation-stack-cleanup.git`	
安裝 Poetry。	依照指示 (Poetry 文件） 在目標虛擬環境中安裝 Poetry。	DevOps 工程師

安裝依存項目。	1. 輸入下列命令以導覽至專案目錄：
	`cd cloudformation-stack-cleanup`
	2. 輸入以下命令：
	`poetry install`
	這會安裝所有必要的相依性，例如 Boto3、Click 和 CloudFormation CLI 的原始程式碼。
		DevOps 工程師

（選用） 安裝 Pyenv。	依照指示 (GitHub) 安裝 Pyenv。	DevOps 工程師

（選用） 自訂架構

任務	Description	所需的技能
建立收集、預先處理和刪除目標資源的函數。	1. 在複製的儲存庫中，輸入下列命令以導覽至 `cli`目錄：	DevOps 工程師，Python

任務	Description	所需的技能
	``` cd cfncli/cli ``` 2. 開啟 cleanup_enviornment.py 檔案。 3. 建立新的 Python 函數，收集您要修改的資源類型。如需範例，請參閱此檔案中的 `gather_ddb_tables` 函數。 4. 建立新的 Python 函數，覆寫目標資源的服務限制條件。如需範例，請參閱此檔案中的 `remove_ddb_deletion_protection` 函數。 5. 建立新的 Python 函數，以收集未受管理的目標資源。如需範例，請參閱此檔案中的 `gather_log_groups` 函數。 6. 建立新的 Python 函數，刪除未受管的目標資源。如需範例，請參閱此檔案中的 `delete_log_group` 函數。 7. 儲存並關閉 cleanup_enviornment.py 檔案。	

建立範例資源

任務	Description	所需的技能
建立 CloudFormation 堆疊。	1. 導覽至 專案目錄。	AWS DevOps

任務	Description	所需的技能
	2. 輸入下列命令來建立佈建 DynamoDB 資料表和安全群組的 CloudFormation 堆疊。更新 的值<VPCID>： `aws cloudformation create-stack \ --stack-name sampleforcleanup-Stack \ --template-body file://samples/sample-cfn-stack.yaml \ --parameters ParameterKey=VpcId,ParameterValue=<VPCID> \ --region us-east-1`	
建立 Systems Manager 參數。	輸入下列命令以建立未透過 CloudFormation 佈建的 Systems Manager 參數： `aws ssm put-parameter \ --name "/samplefforcleanup/database/password" \ --value "your_db_password" \ --type "SecureString" \ --description "Database password for my app" \ --tier "Standard" \ --region "us-east-1"`	AWS DevOps

任務	Description	所需的技能
建立 Amazon S3 儲存貯體。	輸入下列命令來建立未透過 CloudFormation 佈建的 Amazon S3 儲存貯體： `aws s3api create-bucket \ --bucket samplesorcleanup-unmanagedbucket-<UniqueIdentifier> \ --region us-east-1 \ --create-bucket-configuration LocationConstraint=us-east-1`	AWS DevOps

刪除範例資源

任務	Description	所需的技能
刪除 CloudFormation 堆疊。	1. 輸入下列命令來刪除您建立的範例 CloudFormation 堆疊、Systems Manager 參數和 Amazon S3 儲存貯體： `cfncli --region us-east-1 \ dev cleanup-env \ --prefix-list sampleforcleanup` 2. 出現提示時，請輸入 Y以繼續。	AWS DevOps
驗證資源刪除。	在輸出中，確認已刪除所有範例資源。如需範例輸出，請參閱此模式的其他資源區段。	AWS DevOps

相關資源

- 刪除堆疊 (CloudFormation 文件）
- 故障診斷 CloudFormation (CloudFormation 文件）
- 讓 Lambda 函數存取 Amazon VPC 中的資源 (Lambda 文件）
- 如何刪除卡在 DELETE_FAILED 狀態的 AWS CloudFormation 堆疊？ (AWS 知識中心）

其他資訊

以下是來自 `cfncli`命令的範例輸出：

```
cfncli --region aus-east-1  dev cleanup-env --prefix-list sampleforcleanup


https://sts.us-east-1.amazonaws.com
Cleaning up: ['sampleforcleanup'] in xxxxxxxxxx:us-east-1
Do you want to proceed? [Y/n]: Y
No S3 buckets
No ECR repositories
No Lambda functions in VPC
The following DynamoDB tables will have their deletion protection removed:
sampleforcleanup-MyDynamoDBTable
Do you want to proceed with removing deletion protection from these tables? [Y/n]: Y
Deletion protection disabled for DynamoDB table 'sampleforcleanup-MyDynamoDBTable'.
The following CloudFormation stacks will be deleted:
sampleforcleanup-Stack
Do you want to proceed with deleting these CloudFormation stacks? [Y/n]: Y
Initiated deletion of CloudFormation stack: `sampleforcleanup-Stack`
Waiting for stack `sampleforcleanup-Stack` to be deleted...
CloudFormation stack `sampleforcleanup-Stack` deleted successfully.
The following ssm_params will be deleted:
/sampleforcleanup/database/password
Do you want to proceed with deleting these ssm_params? [Y/n]: Y
Deleted SSM Parameter: /sampleforcleanup/database/password
Cleaned up: ['sampleforcleanup']
```

使用 Terraform 在 Amazon Managed Grafana 上自動化 Amazon MWAA 自訂指標的擷取和視覺化

Faisal Abdullah 和 Satya Vajrapu，Amazon Web Services

總結

此模式討論如何使用 Amazon Managed Grafana 建立和監控由 Amazon Managed Workflows for Apache Airflow (Amazon MWAA) 擷取的自訂指標。Amazon MWAA 可做為工作流程的協調器，採用以 Python 編寫指令碼的定向無環圖形 (DAGs)。此模式著重於監控自訂指標，包括過去一小時內執行DAGs 總數、每小時傳遞和失敗DAGs 計數，以及這些程序的平均持續時間。此分析顯示 Amazon Managed Grafana 如何與 Amazon MWAA 整合，以全面監控和洞察此環境中工作流程的協調。

先決條件和限制

先決條件

- 作用中 AWS 帳戶 ，具有建立和管理下列項目的必要使用者許可 AWS 服務：
  - AWS Identity and Access Management (IAM) 角色和政策
  - AWS Lambda
  - Amazon Managed Grafana
  - Amazon Managed Workflows for Apache Airflow (Amazon MWAA)
  - Amazon Simple Storage Service (Amazon S3)
  - Amazon Timestream
- 存取 Shell 環境，該環境可以是本機機器或 上的終端機AWS CloudShell。
- 安裝 Git 並安裝和設定最新版本 AWS Command Line Interface (AWS CLI) 的 shell 環境。如需詳細資訊，請參閱 AWS CLI 文件中的安裝或更新至最新版本的 AWS CLI。
- 已安裝下列 Terraform 版本：`required_version = ">= 1.6.1, < 2.0.0"`您可以使用 tfswitch 在不同的 Terraform 版本之間切換。
- 在 中 AWS IAM Identity Center 為您的 設定身分來源 AWS 帳戶。如需詳細資訊，請參閱 IAM Identity Center 文件中的在 IAM Identity Center 中確認您的身分來源。您可以選擇預設值 IAM Identity Center 目錄、Active Directory 或外部身分提供者 (IdP)，例如 Okta。如需詳細資訊，請參閱相關資源。

限制

- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

- Terraform `required_version = ">= 1.6.1, < 2.0.0"`
- Amazon Managed Grafana 9.4 版或更新版本。此模式已在 9.4 版上測試。

Architecture

下列架構圖會反白顯示解決方案 AWS 服務 中使用的 。

上述圖表會逐步執行下列工作流程：

1. Amazon MWAA 內的自訂指標來自環境中執行DAGs。指標會以 CSV 檔案格式上傳至 Amazon S3 儲存貯體。下列 DAGs 使用 Amazon MWAA 的資料庫查詢功能：
   - `run-example-dag` – 此 DAG 包含定義一或多個任務的範例 Python 程式碼。它會每 7 分鐘執行一次，並列印日期。列印日期之後，DAG 會包含要在特定期間內休眠或暫停執行的任務。
   - `other-sample-dag` – 此 DAG 每 10 分鐘執行一次，並列印日期。列印日期之後，DAG 會包含要在特定期間內休眠或暫停執行的任務。
   - `data-extract` – 此 DAG 每小時執行一次，並查詢 Amazon MWAA 資料庫並收集指標。收集指標之後，此 DAG 會將它們寫入 Amazon S3 儲存貯體，以供進一步處理和分析。
2. 為了簡化資料處理，Lambda 函數會在 Amazon S3 事件觸發時執行，這有助於將指標載入 Timestream。
3. Timestream 整合為 Amazon Managed Grafana 內的資料來源，其中存放了來自 Amazon MWAA 的所有自訂指標。
4. 使用者可以查詢資料並建構自訂儀表板，以視覺化關鍵效能指標，並深入了解 Amazon MWAA 內工作流程的協調。

工具

AWS 服務

- AWS IAM Identity Center 可協助您集中管理所有 AWS 帳戶 和雲端應用程式的單一登入 (SSO) 存取。

- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。在此模式中， 會 AWS Lambda 執行 Python 程式碼以回應 Amazon S3 事件，並自動管理運算資源。
- Amazon Managed Grafana 是一項全受管資料視覺化服務，可用來查詢、關聯和視覺化指標、日誌和追蹤，並發出警示。此模式使用 Amazon Managed Grafana 來建立指標視覺化和提醒的儀表板。
- Amazon Managed Workflows for Apache Airflow (Amazon MWAA) 是 Apache Airflow 的受管協同運作服務，可用來大規模設定和操作雲端中的資料管道。Apache Airflow 是一種開放原始碼工具，用於以程式設計方式撰寫、排程和監控稱為工作流程的程序和任務序列。在此模式中，範例 DAGs和指標擷取器 DAG 會部署在 Amazon MWAA 中。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。在此模式中，Amazon S3 用於儲存 CSV 格式DAGs、指令碼和自訂指標。
- Amazon Timestream for LiveAnalytics 是一種快速、可擴展、全受管的專用時間序列資料庫，可讓您輕鬆地每天存放和分析數兆個時間序列資料點。適用於 LiveAnalytics 的 Timestream 也與資料收集、視覺化和機器學習的常用服務整合。在此模式中，它會用來擷取產生的 Amazon MWAA 自訂指標。

其他工具

- HashiCorp Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。此模式使用 Terraform 模組自動佈建 中的基礎設施 AWS。

程式碼儲存庫

此模式的程式碼可在 GitHub 的 visualize-amazon-mwaa-custom-metrics-grafana 儲存庫中取得。`stacks/Infra` 資料夾包含下列項目：

- 所有 AWS 資源的 Terraform 組態檔案
- `grafana` 資料夾中的 Grafana 儀表板 .json 檔案
- `mwaa/dags` 資料夾中的 Amazon Managed Workflows for Apache Airflow DAGs
- Lambda 程式碼可剖析 .csv 檔案，並將指標存放在 `src` 資料夾中的 Timestream 資料庫
- `templates` 資料夾中的 IAM 政策 .json 檔案

最佳實務

Terraform 必須儲存受管基礎設施和組態的狀態，以便將實際資源映射到您的組態。根據預設，Terraform 會在本機將狀態儲存在名為 的檔案中`terraform.tfstate`。請務必確保 Terraform 狀態檔案的安全性和完整性，因為它會維護基礎設施的目前狀態。如需詳細資訊，請參閱 Terraform 文件中的遠端狀態。

史詩

使用 Terraform 部署基礎設施

任務	Description	所需的技能

部署 基礎設施。	若要部署解決方案基礎設施，請執行下列動作：
	1. 在本機電腦或使用 開啟終端機或命令提示 AWS CloudShell。
	2. 導覽至您要複製儲存庫的目錄。
	3. 若要複製儲存庫，請執行下列命令：
	`git clone https://github.com/aws-samples/visualize-amazon-mwaa-custom-metrics-grafana`
	4. 複製程序完成後，請執行下列命令以導覽至複製的儲存庫目錄：
	`cd visualize-amazon-mwaa-custom-metrics-grafana/stacks/infra`
	5. 若要下載並初始化所需的提供者，請執行下列命令：
		AWS DevOps

任務	Description	所需的技能

	`terraform init`
	6. 若要全面檢視 Terraform 將建立的所有資源，請執行下列命令：
	`terraform plan`
	Terraform 佈建下列資源：
	• Amazon Virtual Private Cloud (Amazon VPC) 和相關聯的網路元件
	• Amazon S3 資源
	• AWS Lambda 函數
	• Amazon Managed Grafana 資源 （工作區、儀表板、資料來源）
	• 支援 IAM 資源 （角色和政策）
	7. 若要從計劃輸出建立 AWS 資源，請執行下列命令：
	`terraform apply -auto-approve`
	基礎設施佈建會在大約 20 分鐘內完成。
	8. 若要根據 Terraform 檔案中定義的組態建立指定的 AWS 資源，請執行下列命令：	

任務	Description	所需的技能
	`terraform apply`	

驗證部署的基礎設施資源

任務	Description	所需的技能
驗證 Amazon MWAA 環境。	若要驗證 Amazon MWAA 環境，請執行下列動作： 1. 登入 AWS 管理主控台，導覽至 Amazon MWAA 儀表板主控台，然後選取開啟氣流使用者介面。 2. 您應該會看到以下三個處於作用中狀態DAGs： • data-extract • run-example-dag • other-sample-dag 3. 如果 DAG 未處於作用中狀態，您可以透過啟用 DAG 名稱旁的切換開關來啟用它。	AWS DevOps，資料工程師
驗證 DAG 排程。	若要檢視每個 DAG 排程，請前往 Airflow UI 中的排程索引標籤。 下列每個 DAGs都有預先設定的排程，在 Amazon MWAA 環境中執行並產生自訂指標： • run-example-dag - 每 7 分鐘執行一次	AWS DevOps 資料工程師

任務	Description	所需的技能
	• other-sample-dag - 每 10 分鐘執行一次 • data-extract - 每小時執行一次 您也可以在執行欄下看到每個 DAG 的成功執行。	

設定 Amazon Managed Grafana 環境

任務	Description	所需的技能
設定對 Amazon Managed Grafana 工作區的存取。	Terraform 指令碼建立了所需的 Amazon Managed Grafana 工作區、儀表板和指標頁面。若要設定存取權以便您可以檢視，請執行下列動作： 1. 開啟 Amazon Managed Grafana 主控台。 2. 在工作區中，選取工作區 `grafana-ws-dev` ，然後導覽至下方窗格中的身分驗證索引標籤。 3. 選擇指派新使用者或群組按鈕。 4. 在群組索引標籤中新增群組，或在使用者索引標籤中新增使用者，然後選擇指派使用者和群組按鈕。 5. 新增使用者（或群組）後，將此使用者（或群組）設為管理員。在指派的使	AWS DevOps

任務	Description	所需的技能
	用者或群組標籤中選取使用者，然後從下拉式選單中選擇進行管理員。如需詳細資訊，請參閱 Amazon Managed Grafana 文件中的搭配使用 AWS IAM Identity Center 與 Amazon Managed Grafana 工作區。 6. 導覽至工作區，然後選擇 Grafana 工作區 URL。若要以管理員身分登入 Amazon Managed Grafana，請選擇使用 登入 AWS IAM Identity Center。	

任務	Description	所需的技能
安裝 Amazon Timestream 外掛程式。	Amazon MWAA 自訂指標會載入 Timestream 資料庫。您可以使用 Timestream 外掛程式，透過 Amazon Managed Grafana 儀表板視覺化指標。 若要安裝 Timestream 外掛程式，請執行下列動作： 1. 在 Amazon Managed Grafana 主控台中，展開左側導覽窗格中的選單，然後前往管理外掛程式。 2. 搜尋並安裝最新版本的 Amazon Timestream 外掛程式。 3. 安裝外掛程式後，請前往管理、資料來源以查看 Timestream 資料來源。如果未列出資料來源，請重新整理頁面。 如需詳細資訊，請參閱《Amazon Managed Grafana 文件》中的使用外掛程式擴展工作區。	AWS DevOps，DevOps 工程師

在 Amazon Managed Grafana 儀表板中視覺化自訂指標

任務	Description	所需的技能
檢視 Amazon Managed Grafana 儀表板。	若要檢視擷取至 Amazon Managed Grafana 工作區的指標，請執行下列動作：	AWS DevOps

任務	Description	所需的技能
	1. 在 Amazon Managed Grafana 主控台中，選擇左側導覽窗格中的儀表板。 2. 若要檢視指標，請選擇 MWAA 事件儀表板，然後選取 mwaa_metrics。 儀表板指標頁面會顯示下列資訊： • 過去 1 小時內執行的總 DAG • 過去一小時成功、失敗和執行的 DAG 執行總數 • 所有、成功和失敗 DAG 執行的平均持續時間	

任務	Description	所需的技能
自訂 Amazon Managed Grafana 儀表板。	若要自訂儀表板以供日後進一步增強功能，請執行下列動作： 1. 在 Amazon Managed Grafana 儀表板`mwaa_metrics` 頁面上，選擇儀表板設定圖示。 2. 若要檢視定義儀表板的資料結構，請選擇 JSON 模型。您可以直接在主控台中編輯此 JSON 模型來自訂儀表板。 或者，此儀表板的原始碼可在 GitHub 儲存庫的 `stacks/infra/grafana` 資料夾中的 `dashboard.json` 檔案中取得。	AWS DevOps

清除 AWS 資源

任務	Description	所需的技能
暫停 Amazon MWAA DAG 執行。	若要暫停 DAG 執行，請執行下列動作： 1. 在 Amazon MWAA 主控台中，導覽至 Airflow 環境，然後選擇開啟 Airflow UI。 2. 若要暫停 DAG，請使用每個 DAG 旁的切換開關。	AWS DevOps，資料工程師

任務	Description	所需的技能
	3. 重新整理 Airflow UI 頁面，該頁面應列出已暫停區段中的三個 DAGs。	
刪除 Amazon S3 儲存貯體中的物件。	若要刪除 Amazon S3 儲存貯體 mwaa-events-bucket-* 和 mwaa-metrics-bucket-*，請遵循 Amazon S3 文件中刪除儲存貯體中的 Amazon S3 主控台使用指示。	AWS DevOps

任務	Description	所需的技能

銷毀 Terraform 建立的資源。	若要銷毀 Terraform 建立的資源和相關聯的本機 Terraform 狀態檔案，請執行下列動作：
	1. （選用）在刪除資源之前，您可以預覽 Terraform 將要進行的變更。若要產生計劃，請執行下列命令：
	`terraform plan -destroy`
	命令輸出顯示`destroy`命令將刪除先前建立的所有 AWS 資源。
	2. `terraform destroy -auto-approve`
	此命令大約需要 20 分鐘才能銷毀基礎設施。
	Note
	若要銷毀 Terraform 管理的所有資源，請執行下列命令。：`-auto-approve` 標籤不會等待使用者確認開始銷毀資源。
	3. 若要刪除本機 Terraform 狀態檔案，請執行下列命令：
	`rm .terraform.lock.hcl`
		AWS DevOps

任務	Description	所需的技能
	`rm -rf .terraform` `rm terraform.tfstate*`	

疑難排解

問題	解決方案
`null_resource.plugin_mgmt (local-exec): aws: error: argument operation: Invalid choice, valid choices are:`	將 升級至 AWS CLI 最新版本。
載入資料來源錯誤 - `Fetch error: 404 Not Found Instantiating…`	錯誤是間歇性的。等待幾分鐘，然後重新整理資料來源以檢視列出的 Timestream 資料來源。

相關資源

AWS 文件

- 用於儀表板和視覺化的 Amazon Managed Grafana
- 設定 Amazon Managed Grafana 使用 Okta
- AWS IAM Identity Center 搭配 Amazon Managed Grafana 工作區使用
- 在 Amazon MWAA 上使用 DAG

AWS 影片

- 使用 Amazon Managed Grafana 設定 IAM Identity Center 進行身分驗證，如下列影片所示。

https://www.youtube-nocookie.com/embed/XX2Xcz-Ps9U?controls=0

- 如果無法使用 IAM Identity Center，您也可以使用 Okta 等外部身分提供者 (IdP) 整合 Amazon Managed Grafana 身分驗證，如下列影片所示。

https://www.youtube-nocookie.com/embed/Z4JHxl2xpOg?controls=0

其他資訊

您可以為您的 Amazon MWAA 環境建立全面的監控和提醒解決方案，以主動管理和快速回應潛在問題或異常。Amazon Managed Grafana 包含下列功能：

警示 – 您可以根據預先定義的閾值或條件，在 Amazon Managed Grafana 中設定警示。設定電子郵件通知，在特定指標超過或低於指定閾值時提醒相關利益相關者。如需詳細資訊，請參閱 Amazon Managed Grafana 文件中的 Grafana 提醒。

整合 – 您可以將 Amazon Managed Grafana 與 OpsGenie、PagerDuty 或 Slack 等各種第三方工具整合，以增強通知功能。例如，您可以設定 Webhook 或與 APIs整合，以根據 Amazon Managed Grafana 中產生的警示觸發這些平台中的事件和通知。此外，此模式提供 GitHub 儲存庫來建立 AWS 資源。您可以進一步將此程式碼與您的基礎設施部署工作流程整合。

使用 AWS CodePipeline 和 AWS CodeBuild 自動化堆疊集部署

Thiyagarajan Mani、Mihir Borkar 和 Raghu Gowda，Amazon Web Services

總結

在持續整合和持續交付 (CI/CD) 程序中，您可能想要自動將應用程式部署到所有現有的 AWS 帳戶，以及您在 AWS Organizations 中新增至組織的新帳戶中。當您為此需求建構 CI/CD 解決方案時，AWS CloudFormation 的委派堆疊集管理員功能非常有用，因為它透過限制對管理帳戶的存取來啟用一層安全性。不過，AWS CodePipeline 會使用服務受管許可模型，將應用程式部署到多個帳戶和區域。您必須使用 AWS Organizations 管理帳戶來部署堆疊集，因為 AWS CodePipeline 不支援委派的堆疊集管理員功能。

此模式說明您可以如何解決此限制。模式使用 AWS CodeBuild 和自訂指令碼，透過 AWS CodePipeline 自動化堆疊集部署。它會自動化這些應用程式部署活動：

- 將應用程式部署為堆疊集到現有的組織單位 OUs)
- 將應用程式的部署擴展到其他 OUs和區域
- 從所有或特定 OUs 或區域移除部署的應用程式

先決條件和限制

先決條件

在您遵循此模式中的步驟之前：

- 在 AWS Organizations 管理帳戶中建立組織。如需說明，請參閱 AWS Organizations 文件。
- 啟用 AWS Organizations 和 CloudFormation 之間的受信任存取，以使用服務受管許可。如需說明，請參閱 CloudFormation 文件中的使用 AWS Organizations 啟用受信任存取。

限制

此模式隨附的程式碼具有下列限制：

- 您只能為應用程式部署單一 CloudFormation 範本；目前不支援多個範本部署。
- 自訂目前的實作需要 DevOps 專業知識。
- 此模式不使用 AWS Key Management System (AWS KMS) 金鑰。不過，您可以透過重新設定此模式隨附的 CloudFormation 範本來啟用此功能。

Architecture

CI/CD 部署管道的此架構會處理下列項目：

- 將堆疊集部署責任委派給專用 CI/CD 帳戶，做為應用程式部署的堆疊集管理員，以限制直接存取管理帳戶。
- 使用服務受管許可模型，每當在 OU 下建立新帳戶並映射時，就會自動部署應用程式。
- 確保環境層級所有帳戶的應用程式版本一致性。
- 在儲存庫和管道層級使用多個核准階段，為部署的應用程式提供額外的安全與控管層。
- 使用 CodeBuild 中的自訂建置部署指令碼來自動部署或移除堆疊集和堆疊執行個體，以克服 CodePipeline 目前的限制。 CodeBuild 如需自訂指令碼實作之 API 呼叫的流程控制和階層說明，請參閱其他資訊一節。
- 為開發、測試和生產環境建立個別堆疊集。此外，您可以建立堆疊集，在每個階段結合多個 OUs 和區域。例如，您可以在開發部署階段中結合沙盒和開發 OUs。
- 支援將應用程式部署至帳戶子集或 OUs 清單或從中排除。

自動化和擴展

您可以使用此模式隨附的程式碼，為您的應用程式建立 AWS CodeCommit 儲存庫和程式碼管道。然後，您可以將這些堆疊集部署到 OU 層級的多個帳戶。此程式碼也會自動化 Amazon Simple Notification Service (Amazon SNS) 主題等元件，以通知核准者、必要的 AWS Identity and Access Management (IAM) 角色，以及要在管理帳戶中套用的服務控制政策 (SCP)。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速一致地佈建資源，以及在整個 AWS 帳戶和區域的生命週期進行管理。
- AWS CodeBuild 是一項全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。
- AWS CodeDeploy 會自動部署到 Amazon Elastic Compute Cloud (Amazon EC2) 或內部部署執行個體、AWS Lambda 函數或 Amazon Elastic Container Service (Amazon ECS) 服務。

- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 AWS 帳戶合併到您建立並集中管理的組織。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。

程式碼儲存庫

此模式的程式碼可在 GitHub automated-code-pipeline-stackset-deployment 儲存庫中使用。如需資料夾結構和其他詳細資訊，請參閱儲存庫的讀我檔案。

最佳實務

此模式限制在 OU 層級部署應用程式時直接存取管理帳戶。將多個核准階段新增至管道和儲存庫程序，有助於為您使用此方法部署的應用程式和元件提供額外的安全性和控管。

史詩

在 AWS Organizations 中設定帳戶

任務	Description	所需的技能
啟用管理帳戶中的所有功能。	遵循 AWS Organizations 文件中的指示，為您的組織啟用管理帳戶中的所有功能。	AWS 管理員、平台管理員
建立 CI/CD 帳戶。	在 AWS Organizations 中，在您的組織中建立專用 CI/CD 帳戶，並指派團隊來擁有和控制對帳戶的存取。	AWS 管理員
新增委派管理員。	在管理帳戶中，將您在上一個步驟中建立的 CI/CD 帳戶註冊為委派堆疊集管理員。如需說明，請參閱 AWS CloudFormation 文件。	AWS 管理員、平台管理員

建立應用程式儲存庫和 CI/CD 管道

任務	Description	所需的技能
複製程式碼儲存庫。	1. 將此模式隨附的程式碼儲存庫複製到您的電腦： `git clone https://github.com/aws-samples/automated-code-pipeline-stackset-deployment.git` 2. 檢閱讀我檔案以了解目錄結構和其他詳細資訊。	AWS DevOps
建立 SNS 主題。	您可以使用 GitHub 儲存庫中提供的`sns-template.yaml` 範本來建立 SNS 主題，並設定核准請求的訂閱。 1. 在 AWS 主控台上，登入 CI/CD 帳戶。 2. 在 https：//https://console.aws.amazon.com/cloudformation 開啟 CloudFormation 主控台。 3. 使用新資源建立新的堆疊（標準選項）。 4. 針對指定範本，選擇上傳範本檔案、選擇檔案，然後從複製的 GitHub 儲存庫`templates` 資料夾中選取`sns-template.yaml` 檔案。選擇下一步。	AWS DevOps

任務	Description	所需的技能
	5. 提供有意義的應用程式堆疊名稱。 6. 指定 資源的字首。 7. 選擇下一步、下一步和提交。 8. 成功建立堆疊後，請選擇輸出索引標籤，並記下提取請求、測試環境和生產環境的 SNS 主題的 Amazon Resource Name (ARNs)。您將在後續步驟中使用此資訊。	

任務	Description	所需的技能
建立 CI/CD 元件的 IAM 角色。	您可以使用 GitHub 儲存庫中提供的`cicd-role-template.yaml` 範本來建立 CI/CD 元件所需的 IAM 角色和政策。 1. 在 AWS 主控台上，登入 CI/CD 帳戶。 2. 在 https：//https://console.aws.amazon.com/cloudformation 開啟 CloudFormation 主控台。 3. 使用新資源建立新的堆疊（標準選項）。 4. 針對指定範本，選擇上傳範本檔案、選擇檔案，然後從複製的 GitHub 儲存庫`templates` 資料夾中選取`cicd-role-template.yaml` 檔案。選擇下一步。 5. 提供有意義的應用程式堆疊名稱。 6. 輸入下列參數的值： • 許可界限政策的 ARN。您可以從 IAM 主控台上許可界限政策的政策詳細資訊區段取得此 ARN。 • 您先前記下的 SNS 生產核准主題的 ARN。 • 您先前記下的 SNS 測試核准主題的 ARN。 • 範本所建立資源的字首。	AWS DevOps

任務	Description	所需的技能
	7. 選擇下一步、下一步和提交。 8. 成功建立堆疊後，請選擇輸出索引標籤，並記下已建立之 IAM 角色的 ARNs。您將在後續步驟中使用此資訊。	

任務	Description	所需的技能
為您的應用程式建立 CodeCommit 儲存庫和程式碼管道。	您可以使用 GitHub 儲存庫中提供的`cicd-pipeline-template.yaml` 範本，為您的應用程式建立 CodeCommit 儲存庫和程式碼管道。 1. 在 AWS 主控台上，登入 CI/CD 帳戶。 2. 在 https：//https://console.aws.amazon.com/cloudformation 開啟 CloudFormation 主控台。 3. 使用新資源建立新的堆疊（標準選項）。 4. 針對指定範本，選擇上傳範本檔案、選擇檔案，然後從複製的 GitHub 儲存庫`templates` 資料夾中選取`cicd-pipeline-template.yaml` 檔案。選擇下一步。 5. 提供有意義的應用程式堆疊名稱。 6. 輸入下列參數的值： • AppRepositoryName – 將為應用程式建立的 CodeCommit 儲存庫名稱。 • AppRepositoryDescription – 將為應用程式建立之 CodeCommit 儲存庫的簡短描述。	AWS DevOps

任務	Description	所需的技能
	• ApplicationName – 您應用程式的名稱。此字串用作 CodeCommit 儲存庫的名稱和 CI/CD 管道的字首。 • CloudWatchEventRoleARN – 上一個任務中 CloudWatch 事件角色的 ARN。 • CodeBuildProjectRoleARN – 上一個任務中 CodeBuild 專案角色的 ARN。 • CodePipelineRoleARN – 上一個任務中 CodePipeline 角色的 ARN。 • DeploymentConfigBucket – 存放部署組態檔案和指令碼 .zip 檔案的 Amazon Simple Storage Service (Amazon S3) 儲存貯體名稱。 • DeploymentConfigKey – 路徑和 .zip 檔案名稱 (Amazon S3 金鑰）。 • PRApprovalSNSARN – 提取請求通知的 SNS 主題 ARN。 • ProdApprovalSNSARN – 生產核准之 SNS 主題的 ARN。	

任務	Description	所需的技能
	• TESTApprovalSNSARN – 用於測試核准的 SNS 主題 ARN。 • TemplateBucket – CI/CD 帳戶中存放 CI/CD 管道建立範本的 S3 儲存貯體名稱。 7. 選擇下一步、下一步和提交。 8. 當堆疊成功完成時，它會建立具有指定名稱和預設目錄結構的 CodeCommit 儲存庫、部署組態檔案、指令碼和儲存庫的程式碼管道。	

部署堆疊集

任務	Description	所需的技能
複製應用程式儲存庫。	您先前使用的 CI/CD 管道範本會建立範例應用程式儲存庫和程式碼管道。若要複製和驗證儲存庫： 1. 登入 CI/CD 帳戶。 2. 尋找您在上一個史詩中建立的應用程式儲存庫和 CI/CD 管道。 3. 複製儲存庫的 URL，並使用 git 複製命令在本機電腦上複製儲存庫。 4. 確認目錄結構和檔案符合下列各項：	應用程式開發人員、資料工程師

任務	Description	所需的技能

	<pre>root
	 |- deploy_configs
	 | |- deployment_config.json
	 |- parameters
	 | |- template-parameter-dev.json
	 | |- template-parameter-test.json
	 | |- template-parameter-prod.json
	 |- templates
	 | |- template.yml
	 |- buildspec.yml</pre> 其中 `deploy_configs` 資料夾包含部署組態檔案，而 `templates` 和 `parameters` 資料夾包含預設檔案，您會將其取代為您自己的 CloudFormation 範本和參數檔案。 **⚠ Important** 請勿自訂資料夾結構。 5. 建立功能分支。	

任務	Description	所需的技能
新增應用程式成品。	使用 CloudFormation 範本更新應用程式儲存庫。 **Note** 此解決方案僅支援部署單一 CloudFormation 範本。 1. 建置用於部署應用程式程式碼變更的 CloudFormation 範本，並將其命名為 `<application-name>.yaml` 。 2. 將應用程式儲存庫 `templates` 資料夾中`template.yml` 的檔案取代為您在步驟 1 中建立的 CloudFormation 範本。 3. 為每個環境準備參數檔案（開發、測試和生產）。 4. 使用格式 命名參數檔案`<cloudformation-template-name>-parameter-<environment-name>.json` 。 5. 將 `parameters` 資料夾中的預設參數檔案取代為步驟 4 中的檔案。	應用程式開發人員、資料工程師

任務	Description	所需的技能
更新部署組態檔案。	更新 deployment_config.json 檔案： 1. 在應用程式儲存庫中，導覽至 deploy_configs 資料夾。 2. 開啟 檔案deployment_config.json ： `{ "deployment_action": "<deploy/delete>", "stack_set_name": "<stack set name>", "stack_set_desciption": "<stack set description>", "deployment_targets": { "dev": { "org_units": ["list of OUs"], "regions": ["list of regions"], "filter_accounts": ["list of accounts"], "filter_type":`	應用程式開發人員、資料工程師

任務	Description	所需的技能

	"<DIFFERENCE/INTERSECTION/UNION>"
	},
	"test": {
	"org_units": ["list of OUs"],
	"regions": ["list of regions"],
	"filter_accounts": ["list of accounts"],
	"filter_type": "<DIFFERENCE/INTERSECTION/UNION>"
	},
	"prod": {
	"org_units": ["list of OUs"],
	"regions": ["list of regions"],
	"filter_accounts": ["list of accounts"],	

任務	Description	所需的技能

	<pre>"filter_type": "<DIFFERENCE/INTERSECTION/UNION>"
	}
	},
	"cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
	"auto_deployement": "<True/False>",
	"retain_stacks_on_account_removal": "<True/False>",
	"region_deployment_concurrency": "<SEQUENTIAL/PARALLEL>"
	}</pre>	

	3. 更新部署動作、堆疊集名稱、堆疊集描述和部署目標的值。 例如，您可以將 `deployment_action` 設定為 `delete`，以刪除整個堆疊集及其相關聯的堆疊執行個體。使用 `deploy` 建立新的堆疊集、更新現有的堆疊集，或新增或移除其他 OUs 或區域的堆疊執行	

任務	Description	所需的技能
	個體。如需更多範例，請參閱其他資訊一節。 此模式透過將環境名稱新增至您在部署組態檔案中提供的堆疊集名稱，為每個環境建立個別堆疊集。	

任務	Description	所需的技能
遞交變更和部署堆疊集。	遞交您在應用程式範本中指定的變更，然後依階段將堆疊集合併並部署到多個環境： 1. 儲存您的所有檔案，並將變更遞交至本機應用程式儲存庫的功能分支。 2. 將功能分支推送至遠端儲存庫。 3. 建立提取請求，將變更合併到主分支。 當提取請求已核准且變更已合併至主分支時，CI/CD 管道將會啟動。 4. 當開發部署階段成功完成時，請檢查 CloudFormation 主控台、StackSets、服務受管索引標籤。 您將看到尾碼為 的新堆疊集`dev`。 5. 檢查 CodeBuild 日誌是否有開發部署階段是否有任何問題。 6. 要求核准者核准這些階段的部署並重複步驟 5 和 6，將堆疊集部署到測試和生產環境中。測試和生產環境的堆疊集具有尾碼 `test`和 `prod`。	應用程式開發人員、資料工程師

疑難排解

問題	解決方案
部署失敗，但有例外： 將範本參數檔案的名稱變更為 <application name>-parameter-<evn>.json，不允許預設名稱	CloudFormation 範本參數檔案必須遵循指定的命名慣例。更新參數檔案名稱，然後再試一次。
部署失敗，但有例外： 將 CloudFormation 範本的名稱變更為不允許 <application name>.yml、預設 template.yml 或 template.yaml	CloudFormation 範本名稱必須遵循指定的命名慣例。更新檔案名稱，然後再試一次。
部署失敗，但有例外： 找不到 {environment name} 環境的有效 CloudFormation 範本及其參數檔案	檢查 CloudFormation 範本的檔案命名慣例及其指定環境的參數檔案。
部署失敗，但有例外： 部署組態檔案中提供的部署動作無效。有效選項為「部署」和「刪除」。	您在部署組態檔案中指定了 `deployment_action` 參數的無效值。參數有兩個有效值：`deploy`和 `delete`。使用 `deploy`建立和更新堆疊集及其相關聯的堆疊執行個體。`delete` 僅在您想要移除整個堆疊集和相關聯的堆疊執行個體時使用。

相關資源

- GitHub automated-code-pipeline-stackset-deployment儲存庫
- 啟用組織中的所有功能 (AWS Organizations 文件）
- 註冊委派管理員 (AWS CloudFormation 文件）
- 服務受管堆疊集的帳戶層級目標 (AWS CloudFormation 文件）

其他資訊

流程圖

下列流程圖說明自訂指令碼實作的 API 呼叫流程控制和階層，以自動化堆疊集部署。

部署組態檔案範例

建立新的堆疊集

下列部署組態檔案會在us-east-1三個 OUs sample-stack-set 的 AWS 區域中建立新的堆疊集，稱為 。

```
{
     "deployment_action": "deploy",
     "stack_set_name": "sample-stack-set",
     "stack_set_desciption": "this is a sample stack set",
    "deployment_targets": {
                            "dev": {
                                        "org_units": ["dev-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                            "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                            "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                           },
     "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
     "auto_deployement": "True",
     "retain_stacks_on_account_removal": "True",
     "region_deployment_concurrency": "PARALLEL"
 }
```

將現有堆疊集部署至另一個 OU

如果您部署上一個範例中顯示的組態，並且想要將堆疊集部署到dev-org-unit-2開發環境中呼叫的其他 OU，則部署組態檔案可能如下所示。

```
{
     "deployment_action": "deploy",
     "stack_set_name": "sample-stack-set",
     "stack_set_desciption": "this is a sample stack set",
    "deployment_targets": {
                              "dev": {
                                        "org_units": ["dev-org-unit-1", "dev-org-
unit-2"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                              "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                              "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                            },
     "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
     "auto_deployement": "True",
     "retain_stacks_on_account_removal": "True",
     "region_deployment_concurrency": "PARALLEL"
 }
```

將現有堆疊集部署至另一個 AWS 區域

如果您部署上一個範例中顯示的組態，而且想要將堆疊集部署到開發環境中兩個 OUs (和 us-east-2) 的其他 AWS 區域 (dev-org-unit-1dev-org-unit-2)，則部署組態檔案可能如下所示。

> **Note**
>
> CloudFormation 範本中的資源必須是有效且區域特定。

```
{
     "deployment_action": "deploy",
     "stack_set_name": "sample-stack-set",
     "stack_set_desciption": "this is a sample stack set",
    "deployment_targets": {
                            "dev": {
                                        "org_units": ["dev-org-unit-1", "dev-org-
unit-2"],
                                        "regions": ["us-east-1", "us-east-2"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                            "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                            "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                           },
     "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
     "auto_deployement": "True",
     "retain_stacks_on_account_removal": "True",
     "region_deployment_concurrency": "PARALLEL"
 }
```

從 OU 或 AWS 區域移除堆疊執行個體

假設上一個範例中顯示的部署組態已部署。下列組態檔案會從 OU 的兩個區域移除堆疊執行個體dev-org-unit-2。

```
{
     "deployment_action": "deploy",
     "stack_set_name": "sample-stack-set",
     "stack_set_desciption": "this is a sample stack set",
    "deployment_targets": {
                            "dev": {
                                        "org_units": ["dev-org-unit-1"],
```

```
                                        "regions": ["us-east-1", "us-east-2"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                                "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                                "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                            },
    "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
    "auto_deployement": "True",
    "retain_stacks_on_account_removal": "True",
    "region_deployment_concurrency": "PARALLEL"
 }
```

下列組態檔案會從開發環境中兩個 OUs us-east-1的 AWS 區域移除堆疊執行個體。

```
{
     "deployment_action": "deploy",
     "stack_set_name": "sample-stack-set",
     "stack_set_desciption": "this is a sample stack set",
    "deployment_targets": {
                                "dev": {
                                        "org_units": ["dev-org-unit-1", "dev-org-
unit-2"],
                                        "regions": ["us-east-2"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                                "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
```

```
                                    },
                                "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                            },
     "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
     "auto_deployement": "True",
     "retain_stacks_on_account_removal": "True",
     "region_deployment_concurrency": "PARALLEL"
 }
```

刪除整個堆疊集

下列部署組態檔案會刪除整個堆疊集及其所有相關聯的堆疊執行個體。

```
{
     "deployment_action": "delete",
     "stack_set_name": "sample-stack-set",
     "stack_set_desciption": "this is a sample stack set",
    "deployment_targets": {
                                "dev": {
                                        "org_units": ["dev-org-unit-1", "dev-org-
unit-2"],
                                        "regions": ["us-east-2"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                                "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                                "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                            },
     "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
```

```
    "auto_deployement": "True",
    "retain_stacks_on_account_removal": "True",
    "region_deployment_concurrency": "PARALLEL"
}
```

從部署中排除帳戶

下列部署組態檔案dev-org-unit-1會從部署中排除111122223333屬於 OU 一部分的帳戶 。

```
{
    "deployment_action": "deploy",
    "stack_set_name": "sample-stack-set",
    "stack_set_desciption": "this is a sample stack set",
   "deployment_targets": {
                            "dev": {
                                        "org_units": ["dev-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": ["111122223333"],
                                        "filter_type": "DIFFERENCE"
                                    },
                            "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                            "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                          },
    "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
    "auto_deployement": "True",
    "retain_stacks_on_account_removal": "True",
    "region_deployment_concurrency": "PARALLEL"
}
```

將應用程式部署到 OU 中的帳戶子集

下列部署組態檔案只會將應用程式部署到 OU 中的三個帳戶 (111122223333444455556666、 和 777788889999)dev-org-unit-1。

```
{
    "deployment_action": "deploy",
    "stack_set_name": "sample-stack-set",
    "stack_set_desciption": "this is a sample stack set",
   "deployment_targets": {
                            "dev": {
                                        "org_units": ["dev-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": ["111122223333",
"444455556666", "777788889999"],
                                        "filter_type": "INTERSECTION"
                                    },
                            "test": {
                                        "org_units": ["test-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    },
                            "prod": {
                                        "org_units": ["prod-org-unit-1"],
                                        "regions": ["us-east-1"],
                                        "filter_accounts": [],
                                        "filter_type": ""
                                    }
                          },
    "cft_capabilities": ["CAPABILITY_IAM", "CAPABILITY_NAMED_IAM"],
    "auto_deployement": "True",
    "retain_stacks_on_account_removal": "True",
    "region_deployment_concurrency": "PARALLEL"
}
```

使用 Cloud Custodian 和 AWS CDK 將 Systems Manager 的 AWS 受管政策自動連接至 EC2 執行個體設定檔

Ali Asfour 和 Aaron Lennon，Amazon Web Services

總結

您可以將 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體與 AWS Systems Manager 整合，以自動化操作任務並提供更高的可見性和控制。若要與 Systems Manager 整合，EC2 執行個體必須具有已安裝的 AWS Systems Manager Agent (SSM Agent) 和連接到其執行個體描述檔的 AmazonSSMManagedInstanceCore AWS Identity and Access Management (IAM) 政策。

不過，如果您想要確保所有 EC2 執行個體描述檔都已連接AmazonSSMManagedInstanceCore政策，則更新沒有執行個體描述檔的新 EC2 執行個體，或沒有執行個體描述檔但沒有AmazonSSMManagedInstanceCore政策的 EC2 執行個體時，可能會面臨挑戰。跨多個 Amazon Web Services (AWS) 帳戶和 AWS 區域新增此政策可能也很困難。

此模式透過在您的 AWS 帳戶中部署三個 Cloud Custodian 政策，協助解決這些挑戰：

- 第一個 Cloud Custodian 政策會檢查是否有具有執行個體描述檔但沒有AmazonSSMManagedInstanceCore政策的現有 EC2 執行個體。然後連接AmazonSSMManagedInstanceCore政策。
- 第二個 Cloud Custodian 政策會檢查沒有執行個體描述檔的現有 EC2 執行個體，並新增已連接AmazonSSMManagedInstanceCore政策的預設執行個體描述檔。
- 第三個 Cloud Custodian 政策會在您的帳戶中建立 AWS Lambda 函數，以監控 EC2 執行個體和執行個體描述檔的建立。這可確保在建立 EC2 執行個體時自動連接AmazonSSMManagedInstanceCore政策。

此模式使用 AWS DevOps 工具，將 Cloud Custodian 政策持續大規模部署至多帳戶環境，而無需佈建個別的運算環境。

先決條件和限制

先決條件

- 兩個或多個作用中的 AWS 帳戶。一個帳戶是安全帳戶，另一個是成員帳戶。
- 在安全帳戶中佈建 AWS 資源的許可。此模式使用管理員許可，但您應該根據組織的需求和政策授予許可。

- 能夠從安全帳戶擔任成員帳戶的 IAM 角色，並建立必要的 IAM 角色。如需詳細資訊，請參閱 IAM 文件中的使用 IAM 角色在 AWS 帳戶之間委派存取權。
-

> **Important**
>
> 安裝並設定 AWS Command Line Interface (AWS CLI)。基於測試目的，您可以使用 `aws configure`命令或設定環境變數來設定 AWS CLI。：這不建議用於生產環境，我們建議您僅將此帳戶授予最低權限存取。如需詳細資訊，請參閱 IAM 文件中的授予最低權限。

- `devops-cdk-cloudcustodian.zip` 檔案 （ 已連接 ），下載到您的本機電腦。
- 熟悉 Python。
- 安裝和設定必要的工具 (Node.js、AWS 雲端開發套件 (AWS CDK) 和 Git)。您可以使用 `devops-cdk-cloudcustodian.zip` 檔案中的 `install-prerequisites.sh` 檔案來安裝這些工具。請確定您使用根權限執行此檔案。

限制

- 雖然此模式可用於生產環境，但請確保所有 IAM 角色和政策都符合您組織的需求和政策。

套件版本

- 雲端託管人 0.9 版或更新版本
- TypeScript 3.9.7 版或更新版本
- Node.js 14.15.4 版或更新版本
- npm 7.6.1 版或更新版本
- AWS CDK 1.96.0 版或更新版本

Architecture

該圖顯示以下工作流程：

1. 雲端託管政策會推送到安全帳戶中的 AWS CodeCommit 儲存庫。Amazon CloudWatch Events 規則會自動啟動 AWS CodePipeline 管道。
2. 管道會從 CodeCommit 擷取最新的程式碼，並將其傳送至 AWS CodeBuild 處理的持續整合和持續交付 (CI/CD) 管道的持續整合部分。

3. CodeBuild 會執行完整的 DevSecOps 動作，包括 Cloud Custodian 政策的政策語法驗證，並在`--dryrun`模式下執行這些政策，以檢查識別哪些資源。
4. 如果沒有錯誤，下一個任務會提醒管理員檢閱變更，並核准成員帳戶中的部署。

技術堆疊

- AWS CDK
- CodeBuild：
- CodeCommit：
- CodePipeline
- IAM
- Cloud Custodian

自動化和擴展

除了使用 AWS CloudFormation 堆疊部署 AWS 資源之外 CodeBuild ，AWS CDK 管道模組還會佈建使用 CodePipeline 協調建置和測試原始程式碼的 CI/CD 管道。您可以針對組織中的所有成員帳戶和區域使用此模式。您也可以擴展`Roles creation`堆疊，在成員帳戶中部署其他 IAM 角色。

工具

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，用於定義程式碼中的雲端基礎設施，並透過 AWS CloudFormation 進行佈建。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可讓您使用命令列 shell 中的命令與 AWS 服務互動。
- AWS CodeBuild 是雲端中全受管的建置服務。
- AWS CodeCommit 是一種版本控制服務，可用來私下存放和管理資產。
- AWS CodePipeline 是一項持續交付服務，可用來建立模型、視覺化和自動化發行軟體所需的步驟。
- AWS Identity and Access Management 是一種 Web 服務，可協助您安全地控制對 AWS 資源的存取。
- Cloud Custodian 是一種工具，可將大多數組織用來管理公有雲端帳戶的數十種工具和指令碼統一為單一開放原始碼工具。
- Node.js 是建置在 Google Chrome 的 V8 JavaScript 引擎上的 JavaScript 執行期。 JavaScript

Code

如需此模式中使用的模組、帳戶函數、檔案和部署命令的詳細清單，請參閱 devops-cdk-cloudcustodian.zip 檔案中的 README 檔案（已連接）。

史詩

使用 AWS CDK 設定管道

任務	Description	所需的技能
設定 CodeCommit 儲存庫。	1. 在本機電腦上的工作目錄中解壓縮 devops-cdk-cloudcustodian.zip 檔案（已連接）。 2. 登入安全帳戶的 AWS 管理主控台，開啟 CodeCommit 主控台，然後建立新的devops-cdk-cloudcustodian 儲存庫。 3. 變更至專案目錄，並將 CodeCommit 儲存庫設定為原始伺服器，遞交變更，然後執行下列命令將其推送至原始伺服器分支： • cd devops-cdk-cloudcustodian • git init --initial-branch=main • git add . git commit -m 'initial commit' • git remote add origin https://git-codecommit.us-east-1.amazonaws.com/v1/devo	開發人員

任務	Description	所需的技能
	`ps-cdk-cloudcustodian` • `git push origin main` 如需詳細資訊，請參閱 AWS CodeCommit 文件中的建立 CodeCommit 儲存庫。 AWS CodeCommit	
安裝必要的工具。	使用 `install-prerequisites.sh` 檔案在 Amazon Linux 上安裝所有必要的工具。這不包含 AWS CLI，因為它已預先安裝。 如需詳細資訊，請參閱 AWS CDK 文件中 AWS CDK 入門的先決條件一節。	開發人員

任務	Description	所需的技能
安裝所需的 AWS CDK 套件。	1. 在 AWS CLI 中執行下列命令來設定您的虛擬環境：`$ python3 -m venv .env` 2. 執行下列命令來啟用您的虛擬環境：`$ source .env/bin/activate` 3. 虛擬環境啟動後，執行下列命令來安裝所需的相依性：`$ pip install -r requirements.txt` 4. 若要新增其他相依性（例如其他 AWS CDK 程式庫），請將它們新增至 `requirements.txt` 檔案，然後執行下列命令：`pip install -r requirements.txt` AWS CDK 需要下列套件，並包含在 `requirements.txt` 檔案中： • `aws-cdk.aws-cloudwatch` • `aws-cdk.aws-codebuild` • `aws-cdk.aws-codecommit` • `aws-cdk.aws-codedeploy` • `aws-cdk.aws-codepipeline`	開發人員

任務	Description	所需的技能
	• aws-cdk.aws-codepipeline-actions • aws-cdk.aws-events • aws-cdk.aws-events-targets • aws-cdk.aws-iam • aws-cdk.aws-logs • aws-cdk.aws-s3 • aws-cdk.aws-sns • aws-cdk.aws-sns-subscriptions • aws-cdk.aws-sqs • aws-cdk.core	

設定您的環境

任務	Description	所需的技能
更新所需的變數。	在 CodeCommit 儲存庫的根資料夾中開啟 vars.py 檔案，並更新下列變數： • var_deploy_region = 'us-east-1' 使用您要部署管道的 AWS 區域進行更新。 • var_codecommit_repo_name = "cdk-cloudcustodian" 使用 CodeCommit 儲存庫的名稱進行更新。	開發人員

任務	Description	所需的技能
	• `var_codecommit_branch_name = "main"` 使用 CodeCommit 分支的名稱進行更新。 • `var_adminEmail=notifyadmin@email.com’` 使用核准變更之管理員的電子郵件地址進行更新。 • 使用用於在進行變更時傳送雲端託管通知的 Slack Webhook 更新 `var_slackWebHookUrl = https://hooks.slack.com/services/T000000000/B00000000/XXXXXXXXXXXXXXXXXXXXXXXX '`。 • `var_orgId = ‘o-yyyyyyyyyy’` 使用您的組織 ID 進行更新。 • `security_account = ‘123456789011’` 使用部署管道之帳戶的 AWS 帳戶 ID 進行更新。 • `member_accounts = [‘111111111111’,’111111111112’,’111111111113’]` 使用您要引導 AWS CDK 堆疊並部署所需 IAM 角色的成員帳戶進行更新。 • `True` 如果您希望管道自動將 AWS CDK 引導至您的成	

任務	Description	所需的技能
	員帳戶，請將 `cdk_bootstrap_member_accounts = True` 設定為 。 如果設為`True`此值，則還需要成員帳戶中可從安全帳戶擔任的現有 IAM 角色的名稱。此 IAM 角色也必須具備啟動 AWS CDK 所需的許可。 • **Note** `cdk_bootstrap_role = 'AWSControlTowerExecution'` 使用可從安全帳戶擔任的成員帳戶中現有的 IAM 角色進行更新。此角色也必須具備啟動 AWS CDK 的許可。 ：這僅適用於`cdk_bootstrap_member_accounts` 將 設定為 的情況`True`。	

任務	Description	所需的技能

使用成員帳戶資訊更新 account.yml 檔案。	若要針對多個帳戶執行 c7n-org Cloud Custodian 工具，您必須將`accounts.yml` 組態檔案放在儲存庫的根目錄。以下是 AWS 的 Cloud Custodian 組態檔案範例：
	`accounts:`
	`- account_id: '123123123123'`
	`  name: account-1`
	`  regions:`
	`  - us-east-1`
	`  - us-west-2`
	`  role: arn:aws:iam::123123123123:role/CloudCustodian`
	`  vars:`
	`  charge_code: xyz`
	`  tags:`
	`  - type:prod`
	`  - division:some division`
	`  - partition:us`
	`  - scope:pci`
		開發人員

引導 AWS 帳戶

任務	Description	所需的技能
提升安全帳戶。	執行下列命令`deploy_account` ，以`cloudcustodian_stack` 應用程式引導： `cdk bootstrap -a 'python3`	開發人員

任務	Description	所需的技能
	`cloudcustodian/cloudcustodian_stack.py`	
選項 1 - 自動引導成員帳戶。	如果 `cdk_bootstrap_member_accounts` 變數`True`在 `vars.py` 檔案中設定為 ，則 `member_accounts` 變數中指定的帳戶會自動由管道引導。 如有需要，您可以使用可從安全帳戶擔任的 `*cdk_bootstrap_role*` IAM 角色進行更新，而且該角色具有啟動 AWS CDK 所需的許可。 新增至`member_accounts` 變數的新帳戶會自動由管道引導，以便部署所需的角色。	開發人員

任務	Description	所需的技能

選項 2 - 手動引導成員帳戶。	雖然我們不建議使用此方法，但您可以設定 的值`cdk_bootstrap_member_accounts` 為 `False`，並透過執行下列命令手動執行此步驟：
	`$ cdk bootstrap -a 'python3 cloudcustodian/member_account_roles_stack.py' \`
	`--trust {security_account_id} \`
	`--context assume-role-credentials:writeIamRoleName={role_name} \`
	`--context assume-role-credentials:readIamRoleName={role_name} \`
	`--mode=ForWriting \`
	`--context bootstrap=true \`
	`--cloudformation-execution-policies arn:aws:iam::aws:policy/AdministratorAccess`
	Important
	請務必使用您可以從安全帳戶取得的 IAM
		開發人員

任務	Description	所需的技能
	角色名稱，以及具有啟動 AWS CDK 所需許可的 `{security_account_id}` 和 `{role_name}` 值。 您也可以使用其他方法來引導成員帳戶，例如使用 AWS CloudFormation。如需詳細資訊，請參閱 AWS CDK 文件中的引導。	

部署 AWS CDK 堆疊

任務	Description	所需的技能
在成員帳戶中建立 IAM 角色。	執行下列命令來部署 `member_account_roles_stack` 堆疊，並在成員帳戶中建立 IAM 角色： `cdk deploy --all -a 'python3 cloudcustodian/member_account_roles_stack.py' --require-approval never`	開發人員
部署 Cloud Custodian 管道堆疊。	執行下列命令來建立部署到安全帳戶的 Cloud Custodian `cloudcustodian_stack.py` 管道：	開發人員

任務	Description	所需的技能
	`cdk deploy -a 'python3 cloudcustodian/cloudcustodian_stack.py'`	

相關資源

- AWS CDK 入門

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS CDK 自動為微服務建置 CI/CD 管道和 Amazon ECS 叢集

Varsha Raju，Amazon Web Services

摘要

此模式說明如何自動建立持續整合和持續交付 (CI/CD) 管道，以及在 Amazon Elastic Container Service (Amazon ECS) 上建置和部署微服務的基礎基礎設施。如果您想要設定proof-of-concept CI/CD 管道，向組織展示 CI/CD、微服務和 DevOps 的優勢，您可以使用此方法。您也可以使用此方法建立初始 CI/CD 管道，然後您可以根據組織的需求自訂或變更這些管道。

模式的方法會建立生產環境和非生產環境，每個環境都有虛擬私有雲端 (VPC) 和設定為在兩個可用區域中執行的 Amazon ECS 叢集。這些環境由所有微服務共用，然後您為每個微服務建立 CI/CD 管道。這些 CI/CD 管道會從 AWS CodeCommit 中的來源儲存庫提取變更、自動建置變更，然後將變更部署到您的生產和非生產環境中。當管道成功完成其所有階段時，您可以使用 URLs生產和非生產環境中存取微服務。

先決條件和限制

先決條件

- 作用中的 Amazon Web Services (AWS) 帳戶。
- 包含 `starter-code.zip` 檔案 （ 已連接 ） 的現有 Amazon Simple Storage Service (Amazon S3) 儲存貯體。
- AWS 雲端開發套件 (AWS CDK)，已安裝並設定在您的 帳戶中。如需詳細資訊，請參閱 AWS CDK 文件中的 AWS CDK 入門。
- Python 3 和 `pip`，已安裝並設定。如需詳細資訊，請參閱 Python 文件。
- 熟悉 AWS CDK、AWS CodePipeline、AWS CodeBuild、CodeCommit、Amazon Elastic Container Registry (Amazon ECR)、Amazon ECS 和 AWS Fargate。
- 熟悉 Docker。
- 了解 CI/CD 和 DevOps。

限制

- 適用一般 AWS 帳戶限制。如需詳細資訊，請參閱 AWS 一般參考文件中的 AWS 服務配額。

產品版本

- 程式碼已使用 Node.js 16.13.0 版和 AWS CDK 1.132.0 版進行測試。

Architecture

該圖顯示以下工作流程：

1. 應用程式開發人員將程式碼遞交至 CodeCommit 儲存庫。
2. 管道已啟動。
3. CodeBuild 建置 Docker 映像並將其推送至 Amazon ECR 儲存庫
4. CodePipeline 會將新映像部署到非生產 Amazon ECS 叢集中的現有 Fargate 服務。
5. Amazon ECS 會將映像從 Amazon ECR 儲存庫提取至非生產 Fargate 服務。
6. 測試是使用非生產 URL 執行。
7. 發行管理員會核准生產部署。
8. CodePipeline 會將新映像部署到生產 Amazon ECS 叢集中的現有 Fargate 服務
9. Amazon ECS 會將映像從 Amazon ECR 儲存庫提取至生產 Fargate 服務。
10生產使用者使用生產 URL 存取您的功能。

技術堆疊

- AWS CDK
- CodeBuild：
- CodeCommit：
- CodePipeline
- Amazon ECR
- Amazon ECS
- Amazon VPC

自動化和擴展

您可以使用此模式的方法，為部署在共用 AWS CloudFormation 堆疊中的微服務建立管道。自動化可以在每個 VPC 中建立多個 Amazon ECS 叢集，也可以為部署在共用 Amazon ECS 叢集中的微服務建立管道。不過，這需要您提供新的資源資訊做為管道堆疊的輸入。

工具

- AWS CDK – AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，用於定義程式碼中的雲端基礎設施，並透過 AWS CloudFormation 進行佈建。
- AWS CodeBuild – AWS CodeBuild 是雲端中全受管的建置服務。CodeBuild 可編譯原始碼、執行單元測試，並產生可立即部署的成品。
- AWS CodeCommit – AWS CodeCommit 是一種版本控制服務，可讓您在 AWS 雲端中私下存放和管理 Git 儲存庫。CodeCommit 讓您無需管理自己的來源控制系統或擔心擴展其基礎設施。
- AWS CodePipeline – AWS CodePipeline 是一種持續交付服務，可用來建立模型、視覺化和自動化發行軟體所需的步驟。您可以使用快速模型化和設定軟體發行程序的不同階段。CodePipeline 會自動化持續發佈軟體變更所需的步驟。
- Amazon ECS – Amazon Elastic Container Service (Amazon ECS) 是一種高度可擴展的快速容器管理服務，用於執行、停止和管理叢集上的容器。您可以在 AWS Fargate 管理的無伺服器基礎設施上執行任務和服務。或者，若要進一步控制您的基礎設施，您可以在您管理的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體叢集上執行任務和服務。
- Docker – Docker 可協助開發人員封裝、運送和執行任何應用程式，做為輕量、可攜式且自給自足的容器。

Code

此模式的程式碼可在 `cicdstarter.zip`和 `starter-code.zip` 檔案（已連接）中使用。

史詩

設定您的環境

任務	Description	所需的技能
設定 AWS CDK 的工作目錄。	1. 在本機電腦上建立名為 `cicdproject` 的目錄。 2. 將`cicdstarter.zip` 檔案（已連接）下載至 `cicdproject` 目錄並解壓縮。這會建立名為 的資料夾`cicdstarter` 。	AWS DevOps、雲端基礎設施

任務	Description	所需的技能
	3. 執行 `cd <user-home>/cicdproject/cicdstarter` 命令。 4. 執行 `python3 -m venv .venv`命令來設定 Python 虛擬環境。 5. 執行 `source ./.venv/bin/activate` 命令。 6. 執行 `aws configure` 命令或使用下列環境變數來設定您的 AWS 環境： • `AWS_ACCESS_KEY_ID` • `AWS_SECRET_ACCESS_KEY` • `AWS_DEFAULT_REGION`	

建立共用基礎設施

任務	Description	所需的技能
建立共用基礎設施。	1. 在您的工作目錄中，執行 `cd cicdvpcecs` 命令。 2. 執行 `pip3 install -r requirements.txt` 命令來安裝所有必要的 Python 相依性 3. 執行 `cdk bootstrap command`來設定 AWS CDK 的 AWS 環境。 4. 執行 `cdk synth --context aws_accou`	AWS DevOps、雲端基礎設施

任務	Description	所需的技能
	nt=<aws_account_ID > --context aws_region=<aws-re gion> 命令。 5. 執行 cdk deploy --context aws_accou nt=<aws_account_ID > --context aws_region=<aws-re gion> 命令。 6. AWS CloudFormation 堆疊會建立下列基礎設施： • 名為 的非生產 VPC cicd-vpc-ecs/cicd-vpc-nonprod • 名為 的生產 VPC cicd-vpc-ecs/cicd-vpc-prod • 名為 的非生產 Amazon ECS 叢集 cicd-ecs-nonprod • 名為 的生產 Amazon ECS 叢集 cicd-ecs-prod	
監控 AWS CloudFormation 堆疊。	1. 登入 AWS 管理主控台，開啟 AWS CloudFormation 主控台，然後從清單中選擇cicd-vpc-ecs 堆疊。 2. 在堆疊詳細資訊窗格中，選擇事件索引標籤，並監控堆疊建立的進度。	AWS DevOps、雲端基礎設施

任務	Description	所需的技能
測試 AWS CloudFormation 堆疊。	1. 建立 `cicd-vpc-ecs` AWS CloudFormation 堆疊之後，請確定已建立 `cicd-vpc-ecs/cicd-vpc-nonprod` 和 `cicd-vpc-ecs/cicd-vpc-prod` VPCs。 2. 確定已建立 `cicd-ecs-nonprod` 和 `cicd-ecs-prod` Amazon ECS 叢集。 **Important** 請務必記錄兩個 VPCs IDs 和兩個 VPCs 中預設安全群組的安全群組 IDs。	AWS DevOps、雲端基礎設施

為微服務建立 CI/CD 管道

任務	Description	所需的技能
建立微服務的基礎設施。	1. 為您的微服務命名。例如，此模式使用 `myservice1` 做為微服務的名稱。 2. 在您的工作目錄中執行 `cd <working-directory>/cdkpipeline` 命令。 3. 執行 `pip3 install -r requirements.txt` 命令。	AWS DevOps、雲端基礎設施

任務	Description	所需的技能
	4. 執行此模式額外資訊區段中可用的完整`cdk synth`命令。 5. 執行此模式額外資訊區段中可用的完整`cdk deploy`命令。 **Note** 您也可以使用 目錄中`cdk.json`的 檔案來提供這兩個命令的值。	
監控 AWS CloudFormation 堆疊。	開啟 AWS CloudFormation 主控台並監控`myservice1-cicd-stack` 堆疊的進度。最後，狀態會變更為 `CREATE_COMPLETE` 。	AWS DevOps、雲端基礎設施

任務	Description	所需的技能
測試 AWS CloudFormation 堆疊。	1. 在 AWS CodeCommit 主控台上，驗證名為 的儲存庫myservice1 是否存在，並包含入門程式碼。 2. 在 AWS CodeBuild 主控台上，驗證名為 的建置專案myservice1 是否存在。 3. 在 Amazon ECR 主控台上，驗證名為 的 Amazon ECR 儲存庫myservice1 是否存在。 4. 在 Amazon ECS 主控台上，驗證非生產和生產 Amazon ECS 叢集中myservice1 是否存在名為 的 Fargate 服務。 5. 在 Amazon Elastic Compute Cloud (Amazon EC2) 主控台上，確認已建立非生產和生產 Application Load Balancer。記錄 ALBs的 DNS 名稱。 6. 在 AWS CodePipeline 主控台上，驗證名為 的管道myservice1 是否存在。它必須具有 Source、Deploy-NonProd 、 Build和 Deploy-Prod 階段。管道也應該有 in progress 狀態。	

任務	Description	所需的技能
	7. 監控管道，直到所有階段都完成為止。 8. 手動核准用於生產。 9. 在瀏覽器視窗中，輸入 ALBs的 DNS 名稱。 10應用程式應該會顯示在 `Hello World`非生產和生產 URLs中。	
使用管道。	1. 開啟您先前建立的 CodeCommit 儲存庫，並開啟 `index.js` 檔案。 2. 使用 `Hello CI/CD` 取代 `Hello World`。 3. 儲存變更並將其遞交至主分支。 4. 驗證管道是否啟動，以及變更是否經過 `Build`、`Deploy-NonProd` 和 `Deploy-Prod` 階段。 5. 手動核准生產。 6. 生產和非生產 URLs 現在都應該顯示 `Hello CICD`。	AWS DevOps、雲端基礎設施
為每個微服務重複此史詩。	重複此史詩中的任務，為每個微服務建立 CI/CD 管道。	AWS DevOps、雲端基礎設施

相關資源

- 搭配 AWS CDK 使用 Python
- AWS CDK Python 參考
- 使用 AWS CDK 建立 AWS Fargate 服務

其他資訊

cdk synth 命令

```
cdk synth --context aws_account=<aws_account_number> --context
 aws_region=<aws_region> --context vpc_nonprod_id=<id_of_non_production
 VPC> --context vpc_prod_id=<id_of_production_VPC> --context
 ecssg_nonprod_id=< default_security_group_id_of_non-production_VPC>
 --context ecssg_prod_id=<default_security_group_id_of_production_VPC>
 --context code_commit_s3_bucket_for_code=<S3 bucket name> --context
 code_commit_s3_object_key_for_code=<Object_key_of_starter_code> --context
 microservice_name=<name_of_microservice>
```

cdk deploy command

```
cdk deploy --context aws_account=<aws_account_number> --context
 aws_region=<aws_region> --context vpc_nonprod_id=<id_of_non_production_VPC>
 --context vpc_prod_id=<id_of_production_VPC> --context ecssg_nonprod_id=<
 default_security_group_id_of_non-production_VPC> --context
 ecssg_prod_id=<default_security_group_id_of_production_VPC> --
context code_commit_s3_bucket_for_code=<S3 bucket name> --context
 code_commit_s3_object_key_for_code=<Object_key_of_starter_code> --context
 microservice_name=<name_of_microservice>
```

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 GitHub 動作和 Terraform 建置 Docker 映像並將其推送至 Amazon ECR

Ruchika Modi，Amazon Web Services

總結

此模式說明如何建立可重複使用的 GitHub 工作流程來建置 Dockerfile，並將產生的映像推送至 Amazon Elastic Container Registry (Amazon ECR)。模式會使用 Terraform 和 GitHub Actions 自動化 Dockerfiles 的建置程序。這可將人為錯誤的可能性降至最低，並大幅縮短部署時間。

GitHub 儲存庫主分支的 GitHub 推送動作會啟動資源部署。工作流程會根據 GitHub 組織和儲存庫名稱的組合來建立唯一的 Amazon ECR 儲存庫。然後，它會將 Dockerfile 映像推送至 Amazon ECR 儲存庫。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 作用中的 GitHub 帳戶。
- GitHub 儲存庫。
- 安裝並設定 Terraform 第 1 版或更新版本。
- Terraform 後端的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。
- 用於 Terraform 狀態鎖定和一致性的 Amazon DynamoDB 資料表。資料表必須具有名為 `LockID`且類型為 的分割區索引鍵`String`。如果未設定，則會停用狀態鎖定。
- AWS Identity and Access Management (IAM) 角色，具有為 Terraform 設定 Amazon S3 後端的許可。如需組態指示，請參閱 Terraform 文件。

限制

此可重複使用的程式碼已僅使用 GitHub 動作進行測試。

Architecture

目標技術堆疊

- Amazon ECR 儲存庫
- GitHub Actions
- Terraform

目標架構

此圖展示了以下要點：

1. 使用者將 Dockerfile 和 Terraform 範本新增至 GitHub 儲存庫。

2. 這些新增項目會啟動 GitHub 動作工作流程。

3. 工作流程會檢查 Amazon ECR 儲存庫是否存在。如果沒有，它會根據 GitHub 組織和儲存庫名稱建立儲存庫。

4. 工作流程會建置 Dockerfile，並將映像推送至 Amazon ECR 儲存庫。

工具

Amazon 服務

- Amazon Elastic Container Registry (Amazon ECR) 是安全、可擴展且可靠的受管容器登錄服務。

其他工具

- GitHub Actions 已整合至 GitHub 平台，協助您在 GitHub 儲存庫中建立、共用和執行工作流程。您可以使用 GitHub 動作來自動化任務，例如建置、測試和部署程式碼。
- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署基礎設施。

程式碼儲存庫

此模式的程式碼可在 GitHub Docker ECR 動作工作流程儲存庫中使用。

- 當您建立 GitHub 動作時，Docker 工作流程檔案會儲存在此儲存庫的 `/.github/workflows/` 資料夾中。此解決方案的工作流程位於 workflow.yaml 檔案中。
- `e2e-test` 資料夾提供範例 Dockerfile 以供參考和測試。

最佳實務

- 如需撰寫 Dockerfiles 的最佳實務，請參閱 Docker 文件。

- 使用 Amazon ECR 的 VPC 端點。VPC 端點採用 AWS PrivateLink，這項技術可讓您透過私有 IP 地址私密存取 Amazon ECR APIs。對於使用 Fargate 啟動類型的 Amazon ECS 任務，VPC 端點可讓任務從 Amazon ECR 提取私有映像，而無需將公有 IP 地址指派給任務。

史詩

設定 OIDC 提供者和 GitHub 儲存庫

任務	Description	所需的技能
設定 OpenID Connect。	建立 OpenID Connect (OIDC) 供應商。您將針對此動作中使用的 IAM 角色，使用信任政策中的提供者。如需說明，請參閱 GitHub 文件中的在 Amazon Web Services 中設定 OpenID Connect。	AWS 管理員、AWS DevOps、一般 AWS
複製 GitHub 儲存庫。	將 GitHub Docker ECR 動作工作流程儲存庫複製到您的本機資料夾： `$git clone https://github.com/aws-samples/docker-ecr-actions-workflow`	DevOps 工程師

自訂 GitHub 可重複使用的工作流程並部署 Docker 映像

任務	Description	所需的技能
自訂啟動 Docker 工作流程的事件。	此解決方案的工作流程位於 workflow.yaml。此指令碼目前設定為在收到`workflow_dispatch` 事件時部署資源。您可以將事件變更為 `workflow_call` 並從	DevOps 工程師

任務	Description	所需的技能
	另一個父工作流程呼叫工作流程，以自訂此組態。	

自訂工作流程。	workflow.yaml 檔案設定為建立動態、可重複使用的 GitHub 工作流程。您可以編輯此檔案來自訂預設組態，或者如果您使用`workflow_dispatch` 事件手動啟動部署，則可以從 GitHub Actions 主控台傳遞輸入值。
	• 請務必指定正確的 AWS 帳戶 ID 和目標區域。
	• 建立 Amazon ECR 生命週期政策（請參閱範例政策），並相應地更新預設路徑 (`e2e-test/policy.json`)。
	• 工作流程檔案需要兩個 IAM 角色做為輸入：
	  • 具有為 Terraform 設定 Amazon S3 後端許可的 IAM 角色（請參閱先決條件一節）。您可以相應地更新 `workload-assumable-role yaml`. 檔案中的預設角色名稱。
	  • 有權存取 GitHub 的 IAM 角色。Amazon ECR 政策中也會使用此角色來限制 Amazon ECR 操作。如需詳細資訊，請參閱 data.tf 檔案。
		DevOps 工程師

任務	Description	所需的技能
部署 Terraform 範本。	工作流程會根據您設定的 GitHub 事件，自動部署建立 Amazon ECR 儲存庫的 Terraform 範本。這些範本在 Github 儲存庫的根目錄以.tf檔案形式提供。	AWS DevOps，DevOps 工程師

疑難排解

問題	解決方案
將 Amazon S3 和 DynamoDB 設定為 Terraform 遠端後端時的問題或錯誤。	請遵循 Terraform 文件中的指示，為遠端後端組態設定 Amazon S3 和 DynamoDB 資源所需的許可。
無法使用 workflow_dispatch 事件執行或啟動工作流程。	設定為從workflow_dispatch 事件部署的工作流程只有在主分支上也設定工作流程時才有效。

相關資源

- 重複使用工作流程 (GitHub 文件)
- 觸發工作流程 (GitHub 文件)

使用 AWS CodeCommit、AWS CodePipeline 和 AWS Device Farm 建置和測試 iOS 應用程式

Abdullahi Olaoye，Amazon Web Services

總結

此模式概述了建立持續整合和持續交付 (CI/CD) 管道的步驟，該管道使用 AWS CodePipeline 在 AWS 上的實際裝置上建置和測試 iOS 應用程式。模式使用 AWS CodeCommit 來存放應用程式碼、Jenkins 開放原始碼工具來建置 iOS 應用程式，以及 AWS Device Farm 來測試實際裝置上建置的應用程式。這三個階段會使用 AWS CodePipeline 在管道中協調在一起。

此模式是以 AWS DevOps 部落格上的使用 AWS DevOps 和行動服務建置和測試 iOS 和 iPadOS 應用程式文章為基礎。 DevOps 如需詳細說明，請參閱部落格文章。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Apple 開發人員帳戶
- 組建伺服器 (macOS)
- Xcodeversion 11.3 （ 在建置伺服器上安裝和設定 ）
- 在工作站上安裝https://docs.aws.amazon.com/cli/latest/userguide/install-cliv1.html和設定 AWS Command Line Interface (AWS CLI)
- Git 的基本知識

限制

- 應用程式建置伺服器必須執行 macOS。
- 建置伺服器必須具有公有 IP 地址，因此 CodePipeline 可以遠端連線到它以啟動建置。

Architecture

來源技術堆疊

- 內部部署 iOS 應用程式建置程序，涉及在實體裝置上使用模擬器或手動測試

目標技術堆疊

- 用於儲存應用程式原始碼的 AWS CodeCommit 儲存庫
- 使用 Xcode 建置應用程式的 Jenkins 伺服器
- 用於在實際裝置上測試應用程式的 AWS Device Farm 裝置集區

目標架構

當使用者遞交變更至來源儲存庫時，管道 (AWS CodePipeline) 會從來源儲存庫擷取程式碼、啟動 Jenkins 組建，並將應用程式程式碼傳遞給 Jenkins。在建置之後，管道會擷取建置成品，並啟動 AWS Device Farm 任務，以針對裝置集區測試應用程式。

工具

- AWS CodePipeline 是一種全受管的持續交付服務，可協助您自動化發行管道，以實現快速可靠的應用程式和基礎設施更新。根據您定義的發行模型，CodePipeline 可以自動在每次程式碼變更時建置、測試和部署程式碼。
- AWS CodeCommit 是全受管的來源控制服務，可託管安全的 Git 型儲存庫。這可讓團隊在安全且可擴展的生態系統中輕鬆協作程式碼。CodeCommit 無需操作您自己的來源控制系統或擔心擴展其基礎設施。
- AWS Device Farm 是一種應用程式測試服務，可讓您在各種桌面瀏覽器和真實行動裝置中測試 Web 和行動應用程式，藉此改善其品質，而無需佈建和管理任何測試基礎設施。
- Jenkins 是一種開放原始碼自動化伺服器，可讓開發人員建置、測試和部署其軟體。

史詩

設定建置環境

任務	Description	所需的技能
在執行 macOS 的建置伺服器上安裝 Jenkins。	Jenkins 將用於建置應用程式，因此您必須先將其安裝在建置伺服器上。若要取得此任務和後續任務的詳細說明，請參閱此模式結尾相關資源區段中	DevOps

任務	Description	所需的技能
	的 AWS 部落格文章使用 AWS DevOps 和行動服務和其他資源建置和測試 iOS 和 iPadOS 應用程式。	
設定 Jenkins。	依照畫面上的指示來設定 Jenkins。	DevOps
安裝適用於 Jenkins 的 AWS CodePipeline 外掛程式。	此外掛程式必須安裝在 Jenkins 伺服器上，Jenkins 才能與 AWS CodePipeline 服務互動。	DevOps
建立 Jenkins 自由樣式專案。	在 Jenkins 中，建立自由樣式專案。設定專案以指定觸發條件和其他建置組態選項。	DevOps

設定 AWS Device Farm

任務	Description	所需的技能
建立 Device Farm 專案。	開啟 AWS Device Farm 主控台。建立專案和裝置集區以進行測試。如需說明，請參閱部落格文章。	開發人員

設定來源儲存庫

任務	Description	所需的技能
建立 CodeCommit 儲存庫。	建立儲存來源碼的儲存庫。	DevOps
將您的應用程式程式碼遞交至儲存庫。	連線至您建立的 CodeCommit 儲存庫。將程式碼從本機機器推送至儲存庫。	DevOps

設定管道

任務	Description	所需的技能
在 AWS CodePipeline 中建立管道。	開啟 AWS CodePipeline 主控台，並建立管道。管道會協調 CI/CD 程序的所有階段。如需說明，請參閱 AWS 部落格文章使用 AWS DevOps 和行動服務建置和測試 iOS 和 iPadOS 應用程式。	DevOps
將測試階段新增至管道。	若要新增測試階段並將其與 AWS Device Farm 整合，請編輯管道。	DevOps
啟動管道。	若要啟動管道和 CI/CD 程序，請選擇釋出變更。	DevOps

檢視應用程式測試結果

任務	Description	所需的技能
檢閱測試結果。	在 AWS Device Farm 主控台中，選取您建立的專案，並檢閱測試結果。主控台會顯示每個測試的詳細資訊。	開發人員

相關資源

此模式的Step-by-step說明

- 使用 AWS DevOps 和行動服務建置和測試 iOS 和 iPadOS 應用程式 (AWS DevOps 部落格文章)

設定 AWS Device Farm

- AWS Device Farm 主控台

設定來源儲存庫

- 建立 AWS CodeCommit 儲存庫
- 連線至 AWS CodeCommit 儲存庫

設定管道

- AWS CodePipeline 主控台

其他資源

- AWS CodePipeline 文件
- AWS CodeCommit 文件
- AWS Device Farm 文件
- Jenkins 文件
- macOS 上的 Jenkins 安裝
- Jenkins 的 AWS CodePipeline 外掛程式
- Xcode 安裝
- AWS CLI 安裝和組態
- Git 文件

為在 Amazon EKS 上執行的應用程式設定交互 TLS 身分驗證

Mahendra Revanasiddappa，Amazon Web Services

摘要

憑證型交互傳輸層安全 (TLS) 是選用的 TLS 元件，可在伺服器和用戶端之間提供雙向對等身分驗證。透過交互 TLS，用戶端必須在工作階段交涉過程中提供 X.509 憑證。伺服器使用此憑證來識別和驗證用戶端。

相互 TLS 是物聯網 (IoT) 應用程式的常見需求，可用於business-to-business應用程式或標準，例如 Open Banking。

此模式說明如何使用 NGINX 輸入控制器，為在 Amazon Elastic Kubernetes Service (Amazon EKS) 叢集上執行的應用程式設定交互 TLS。您可以為 NGINX 輸入控制器啟用內建的交互 TLS 功能，方法是註釋輸入資源。如需 NGINX 控制器上交互 TLS 註釋的詳細資訊，請參閱 Kubernetes 文件中的用戶端憑證驗證。

> **Important**
>
> 此模式使用自我簽署憑證。我們建議您僅將此模式與測試叢集搭配使用，而不是在生產環境中。如果您想要在生產環境中使用此模式，您可以使用 AWS Private Certificate Authority (AWS Private CA) 或現有的公有金鑰基礎設施 (PKI) 標準來發行私有憑證。

先決條件和限制

先決條件

- 作用中的 Amazon Web Services (AWS) 帳戶。
- 現有 Amazon EKS 叢集。
- 在 macOS、Linux 或 Windows 上安裝和設定 AWS Command Line Interface (AWS CLI) 1.7 版或更新版本。
- 安裝並設定為存取 Amazon EKS 叢集的 kubectl 命令列公用程式。如需詳細資訊，請參閱 Amazon EKS 文件中的安裝 kubectl。
- 用來測試應用程式的現有網域名稱系統 (DNS) 名稱。

限制

- 此模式使用自我簽署憑證。我們建議您僅將此模式與測試叢集搭配使用，而不是在生產環境中。

Architecture

技術堆疊

- Amazon EKS
- Amazon Route 53
- kubectl

工具

- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 AWS 上執行 Kubernetes，而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- Amazon Route 53 是一種可用性高、可擴展性強的 DNS Web 服務。
- Kubectl 是您用來與 Amazon EKS 叢集互動的命令列公用程式。

史詩

產生自我簽署憑證

任務	Description	所需的技能
產生 CA 金鑰和憑證。	執行下列命令來產生憑證授權單位 (CA) 金鑰和憑證。 `openssl req -x509 -sha256 -newkey rsa:4096 -keyout ca.key -out ca.crt -days 356 -nodes -subj '/CN=Test Cert Authority'`	DevOps 工程師
產生伺服器金鑰和憑證，並使用 CA 憑證簽署。	產生伺服器金鑰和憑證，並執行下列命令以 CA 憑證簽署。	DevOps 工程師

任務	Description	所需的技能
	`openssl req -new -newkey rsa:4096 -keyout server.key -out server.csr -nodes -subj '/CN= <your_domain_name> ' && openssl x509 -req -sha256 -days 365 -in server.csr -CA ca.crt -CAkey ca.key -set_serial 01 -out server.crt` **Important** 請確定您使用現有的網域名稱`<your_domain_name>` 取代。	
產生用戶端金鑰和憑證，並使用 CA 憑證簽署。	產生用戶端金鑰和憑證，並執行下列命令以 CA 憑證簽署。 `openssl req -new -newkey rsa:4096 -keyout client.key -out client.csr -nodes -subj '/CN=Test' && openssl x509 -req -sha256 -days 365 -in client.csr -CA ca.crt -CAkey ca.key -set_serial 02 -out client.crt`	DevOps 工程師

部署 NGINX 輸入控制器

任務	Description	所需的技能
在 Amazon EKS 叢集中部署 NGINX 輸入控制器。	使用下列命令部署 NGINX 輸入控制器。 `kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.7.0/deploy/static/provider/aws/deploy.yaml`	DevOps 工程師
確認 NGINX 輸入控制器服務正在執行。	使用以下命令，確認 NGINX 輸入控制器服務正在執行。 `kubectl get svc -n ingress-nginx` ⚠ Important 請確定服務地址欄位包含 Network Load Balancer 的網域名稱。	DevOps 工程師

在 Amazon EKS 叢集中建立命名空間，以測試相互 TLS

任務	Description	所需的技能
在 Amazon EKS 叢集中建立命名空間。	執行下列命令，在您的 Amazon EKS 叢集`mtls`中建立名為 的命名空間。	DevOps 工程師

任務	Description	所需的技能
	`kubectl create ns mtls` 這會部署範例應用程式來測試交互 TLS。	

為範例應用程式建立部署和服務

任務	Description	所需的技能

在 mtls 命名空間中建立 Kubernetes 部署和服務。	建立名為 `mtls.yaml` 的檔案。將以下程式碼貼到檔案。
	
	`kind: Deployment`
	`apiVersion: apps/v1`
	`metadata:`
	`  name: mtls-app`
	`  labels:`
	`  app: mtls`
	`spec:`
	`  replicas: 1`
	`  selector:`
	`  matchLabels:`
	`  app: mtls`
	`  template:`
	`  metadata:`
	`  labels:`
	`  app: mtls`
	`  spec:`
	`  containers:`
	`  - name: mtls-app`
	`  image: hashicorp/http-echo`
	`  args:`
	`  - "-text=mTLS is working"`
	
	`---`
		DevOps 工程師

任務	Description	所需的技能

	<pre>kind: Service
	apiVersion: v1
	metadata:
	  name: mtls-service
	spec:
	  selector:
	  app: mtls
	  ports:
	  - port: 5678 #
	 Default port for image</pre>
	執行下列命令，在 `mtls` 命名空間中建立 Kubernetes 部署和服務。
	<pre>kubectl create -f
	 mtls.yaml -n mtls</pre>	

確認已建立 Kubernetes 部署。	執行下列命令，以確認已建立部署，且有一個處於可用狀態的 Pod。
	<pre>kubectl get deploy -n
	 mtls</pre>
		DevOps 工程師

確認已建立 Kubernetes 服務。	執行下列命令，確認已建立 Kubernetes 服務。
	<pre>kubectl get service -n
	 mtls</pre>
		DevOps 工程師

在 mtls 命名空間中建立秘密

任務	Description	所需的技能
建立傳入資源的秘密。	執行 folllowing 命令，使用您先前建立的憑證，為 NGINX 輸入控制器建立秘密。 `kubectl create secret generic mtls-certs --from-file=tls.crt=server.crt --from-file=tls.key=server.key --from-file=ca.crt=ca.crt -n mtls` 您的秘密具有一個伺服器憑證供用戶端識別伺服器，以及一個 CA 憑證供伺服器驗證用戶端憑證。	DevOps 工程師

在 mtls 命名空間中建立輸入資源

任務	Description	所需的技能

在 mtls 命名空間中建立輸入資源。	建立名為 `ingress.yaml` 的檔案。將下列程式碼貼入 檔案 (`<your_domain_name>` 以您現有的網域名稱取代)。
	`apiVersion: networking.k8s.io/v1`
	`kind: Ingress`
	`metadata:`
	`  annotations:`
	`  nginx.ingress.kubernetes.io/auth-tls-verify-client: "on"`
		DevOps 工程師

任務	Description	所需的技能

	<pre> nginx.ingress.kube
	rnetes.io/auth-tls-
	secret: mtls/mtls-certs
	  name: mtls-ingress
	spec:
	  ingressClassName:
	 nginx
	  rules:
	  - host: "*.<your_
	domain_name>"
	    http:
	      paths:
	      - path: /
	        pathType: Prefix
	        backend:
	          service:
	            name: mtls-
	service
	            port:
	              number:
	 5678
	  tls:
	  - hosts:
	    - "*.<your_
	domain_name>"
	    secretName: mtls-
	certs</pre>
	執行下列命令，在mtls命名空間中建立輸入資源。
	<pre>kubectl create -f
	 ingress.yaml -n mtls</pre>
	這表示 NGINX 輸入控制器可以將流量路由到您的範例應用程式。	

任務	Description	所需的技能
確認已建立輸入資源。	執行下列命令，確認已建立輸入資源。 `kubectl get ing -n mtls` **Important** 請確定輸入資源的地址顯示為 NGINX 輸入控制器建立的負載平衡器。	DevOps 工程師

設定 DNS 將主機名稱指向負載平衡器

任務	Description	所需的技能
建立指向 NGINX 輸入控制器負載平衡器的 CNAME 記錄。	登入 AWS 管理主控台，開啟 Amazon Route 53 主控台，並建立正式名稱 (CNAME) 記錄，`mtls.<your_domain_name>` 指向 NGINX 輸入控制器的負載平衡器。 如需詳細資訊，請參閱 Route Route 53 文件中的使用 Route 53 主控台建立記錄。 Route 53	DevOps 工程師

測試應用程式。

任務	Description	所需的技能
在沒有憑證的情況下測試交互 TLS 設定。	執行下列命令。 `curl -k https://mtls.<your_domain_name>` 您應該會收到「400 未傳送必要的 SSL 憑證」錯誤回應。	DevOps 工程師
使用憑證測試交互 TLS 設定。	執行下列命令。 `curl -k https://mtls.<your_domain_name> --cert client.crt --key client.key` 您應該會收到「mTLS 正在運作」回應。	DevOps 工程師

相關資源

- 使用 Amazon Route 53 主控台建立記錄
- 在 Amazon EKS 上使用 Network Load Balancer 搭配 NGINX 輸入控制器
- 用戶端憑證驗證

使用 自動建立 Amazon WorkSpaces 應用程式資源 AWS CloudFormation

Ram Kandaswamy，Amazon Web Services

摘要

此模式提供程式碼範例和步驟， AWS 雲端 以使用 AWS CloudFormation 範本自動在 中建立 Amazon WorkSpaces 應用程式資源。模式說明如何使用 CloudFormation 堆疊自動建立 WorkSpaces 應用程式應用程式資源，包括映像建置器、映像、機群執行個體和堆疊。您可以使用桌面或應用程式交付模式，將 WorkSpaces 應用程式串流至 HTML5-compliant瀏覽器上的最終使用者。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 接受 WorkSpaces 應用程式條款與條件
- WorkSpaces 應用程式資源的基本知識，例如機群、堆疊和映像建置器

限制

- 您無法在該執行個體建立之後修改與 WorkSpaces 應用程式執行個體相關聯的 AWS Identity and Access Management(IAM) 角色。
- 建立映像建置器之後，您無法修改 WorkSpaces 應用程式映像建置器執行個體上的屬性 （例如子網路或安全群組)。

Architecture

下圖說明如何使用 CloudFormation 範本自動建立 WorkSpaces 應用程式資源。

該圖顯示以下工作流程：

1. 您可以在此模式的其他資訊???區段中，根據 YAML 程式碼建立 CloudFormation 範本。
2. CloudFormation 範本會建立 CloudFormation 測試堆疊。
   a. （選用） 您可以使用 WorkSpaces 應用程式建立映像建置器執行個體。
   b. （選用） 您可以使用自訂軟體建立 Windows 映像。

3. CloudFormation 堆疊會建立 WorkSpaces 應用程式機群執行個體和堆疊。
4. 您可以將 WorkSpaces 應用程式資源部署至 HTML5-compliant瀏覽器上的最終使用者。

工具

- Amazon WorkSpaces 應用程式是一項全受管應用程式串流服務，可讓您從任何地方立即存取桌面應用程式。WorkSpaces 應用程式會管理託管和執行應用程式所需的 AWS 資源、自動擴展，以及隨需提供使用者存取權。
- AWS CloudFormation 可協助您模型化和設定 AWS 資源、快速一致地佈建資源，以及在整個生命週期中管理資源。您可以使用範本來描述您的資源及其相依性，並將它們一起啟動和設定為堆疊，而不是個別管理資源。您可以管理和佈建跨多個 和 的堆疊 AWS 帳戶 AWS 區域。

最佳實務

- 正確設定映像建置器的網路存取 – 使用 NAT 閘道進行傳出限定網際網路存取，在具有適當網際網路存取的虛擬私有雲端 (VPC) 子網路中啟動映像建置器。

  在建立映像之前，測試與必要資源 （例如應用程式伺服器、資料庫和授權伺服器） 的網路連線。確認 VPC 路由表允許連線至所有必要的網路資源。如需詳細資訊，請參閱 WorkSpaces 應用程式文件中的網際網路存取。
- 根據服務配額主動監控機群容量 – WorkSpaces 應用程式執行個體類型和大小配額是根據 AWS 帳戶每個 AWS 區域。如果您在同一個區域中有多個使用相同執行個體類型和大小的機群，則該區域中所有機群的執行個體總數必須小於或等於適用的配額。如需詳細資訊，請參閱 WorkSpaces 應用程式文件中的故障診斷機群。

- 在機群部署之前，在映像建置器測試模式中測試應用程式 – 在建立映像和部署至機群之前，一律在映像建置器測試模式中驗證應用程式。測試模式模擬最終使用者在機群執行個體上擁有的有限許可。如需詳細資訊，請參閱 WorkSpaces 應用程式文件中的疑難排解映像建置器。

史詩

（選用）建立 WorkSpaces 應用程式映像

任務	Description	所需的技能
安裝自訂軟體並建立映像。	1. 啟動映像建置器，然後選擇相關的執行個體類型、子網路、安全群組等。 2. 連線至執行個體時使用管理員模式。 3. 使用映像助理或 PowerShell 指令碼為您的自訂軟體建立新的 Windows 映像。 **Note** 請考慮使用 Windows AppLocker 功能來進一步鎖定映像。	AWS DevOps，雲端架構師

部署 CloudFormation 範本

任務	Description	所需的技能
更新 CloudFormation 範本。	1. 使用此模式額外資訊區段中的程式碼做為 YAML 檔案。 2. 使用環境中參數的必要值來更新 YAML 檔案。	AWS 系統管理員、雲端管理員、雲端架構師、一般 AWS、AWS 管理員
使用 範本建立 CloudFormation 堆疊。	1. 登入 AWS 管理主控台 並開啟 CloudFormation 主控台。 2. 在導覽窗格中，選擇 Stacks。	應用程式擁有者、AWS 系統管理員、Windows Engineer

任務	Description	所需的技能
	3. 選擇 Create stack (建立堆疊)，然後選擇 With new resources (standard) (使用新資源 (標準))。 4. 在先決條件 – 準備範本區段中，選擇範本已就緒。 5. 在指定範本區段中，選擇上傳範本檔案。 6. 選擇選擇檔案，然後選擇更新的 CloudFormation 範本。 7. 完成精靈中的其餘步驟，以建立您的堆疊。	

疑難排解

問題	解決方案
各種問題	如需詳細資訊，請參閱 WorkSpaces 應用程式文件中的故障診斷。

相關資源

參考

- Amazon WorkSpaces 應用程式入門：使用範例應用程式設定
- 建立 Amazon WorkSpaces 應用程式機群和堆疊

教學課程和影片

- Amazon WorkSpaces 應用程式使用者工作流程
- 如何將舊版 Windows Forms 應用程式遷移至 Amazon WorkSpaces 應用程式
- AWS re：Invent 2018：使用 Amazon WorkSpaces 應用程式安全地交付桌面應用程式 (BAP201)

其他資訊

下列程式碼是 CloudFormation 範本的範例，可用來自動建立 WorkSpaces 應用程式資源。

```
AWSTemplateFormatVersion: 2010-09-09
Parameters:
  SubnetIds:
    Type: 'List<AWS::EC2::Subnet::Id>'
  testSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup::Id'
  ImageName:
    Type: String
Resources:

  AppStreamFleet:
    Type: 'AWS::AppStream::Fleet'
    Properties:
      ComputeCapacity:
        DesiredInstances: 5
      InstanceType: stream.standard.medium
      Name: appstream-test-fleet
      DisconnectTimeoutInSeconds: 1200
      FleetType: ON_DEMAND
      IdleDisconnectTimeoutInSeconds: 1200
      ImageName: !Ref ImageName
      MaxUserDurationInSeconds: 345600
      VpcConfig:
        SecurityGroupIds:
          - !Ref testSecurityGroup
        SubnetIds: !Ref SubnetIds
  AppStreamStack:
    Type: 'AWS::AppStream::Stack'
    Properties:
      Description: AppStream stack for test
      DisplayName: AppStream test Stack
      Name: appstream-test-stack
      StorageConnectors:
        - ConnectorType: HOMEFOLDERS
      UserSettings:
        - Action: CLIPBOARD_COPY_FROM_LOCAL_DEVICE
          Permission: ENABLED
        - Action: CLIPBOARD_COPY_TO_LOCAL_DEVICE
          Permission: ENABLED
        - Action: FILE_DOWNLOAD
```

```
        Permission: ENABLED
      - Action: PRINTING_TO_LOCAL_DEVICE
        Permission: ENABLED
AppStreamFleetAssociation:
  Type: 'AWS::AppStream::StackFleetAssociation'
  Properties:
    FleetName: appstream-test-fleet
    StackName: appstream-test-stack
  DependsOn:
    - AppStreamFleet
    - AppStreamStack
```

使用 Firelens 日誌路由器為 Amazon ECS 建立自訂日誌剖析器

Varun Sharma，Amazon Web Services

總結

Firelens 是 Amazon Elastic Container Service (Amazon ECS) 和 AWS Fargate 的日誌路由器。您可以使用 Firelens 將容器日誌從 Amazon ECS 路由到 Amazon CloudWatch 和其他目的地（例如 Splunk 或 Sumo Logic)。Firelens 使用 Fluentd 或 Fluent Bit 作為記錄代理程式，這表示您可以使用 Amazon ECS 任務定義參數來路由日誌。

透過選擇在來源層級剖析日誌，您可以分析記錄資料並執行查詢，以更有效率且有效地回應操作問題。由於不同的應用程式有不同的記錄模式，因此您需要使用自訂剖析器來建構日誌，並在最終目的地進行搜尋。

此模式使用 Firelens 日誌路由器搭配自訂剖析器，從在 Amazon ECS 上執行的範例 Spring Boot 應用程式將日誌推送至 CloudWatch。然後，您可以使用 Amazon CloudWatch Logs Insights 根據自訂剖析器產生的自訂欄位來篩選日誌。

先決條件和限制

先決條件

- 作用中的 Amazon Web Services (AWS) 帳戶。
- AWS Command Line Interface (AWS CLI)，安裝在本機機器上並進行設定。
- 在本機電腦上安裝和設定 Docker。
- Amazon Elastic Container Registry (Amazon ECR) 上現有的 Spring Boot 型容器化應用程式。

Architecture

技術堆疊

- CloudWatch
- Amazon ECR
- Amazon ECS
- Fargate
- Docker

- Fluent Bit

工具

- Amazon ECR – Amazon Elastic Container Registry (Amazon ECR) 是一種 AWS 受管容器映像登錄服務，安全、可擴展且可靠。
- Amazon ECS – Amazon Elastic Container Service (Amazon ECS) 是一種高度可擴展的快速容器管理服務，可讓您輕鬆地執行、停止和管理叢集上的容器。
- AWS Identity and Access Management (IAM) – IAM 是一種 Web 服務，可安全地控制對 AWS 服務的存取。
- AWS CLI – AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可讓您使用命令列 shell 中的命令與 AWS 服務互動。
- Docker – Docker 是開發、運送和執行應用程式的開放平台。

Code

下列檔案會附加至此模式：

- `customFluentBit.zip` – 包含要新增自訂剖析和組態的檔案。
- `firelens_policy.json` – 包含用來建立 IAM 政策的政策文件。
- `Task.json` – 包含 Amazon ECS 的範例任務定義。

史詩

建立自訂 Fluent Bit 映像

任務	Description	所需的技能
建立 Amazon ECR 儲存庫。	登入 AWS 管理主控台，開啟 Amazon ECR 主控台，並建立名為 的儲存庫`fluentbit_custom` 。 如需詳細資訊，請參閱 Amazon ECR 文件中的建立儲存庫。	系統管理員、開發人員

任務	Description	所需的技能
解壓縮 customFluentBit.zip 套件。	1. 將customFluentBit.zip 套件（已連接）下載到您的本機電腦。 2. 執行下列命令以解壓縮至 customFluentBit 目錄：unzip -d customFluentBit.zip 3. 目錄包含新增自訂剖析和組態所需的下列檔案： • parsers/springboot_parser.conf – 包含剖析器指令，並定義自訂剖析器的規則表達式 (regex) 模式。您可以為特定剖析器新增 regex 模式。 • conf/parse_springboot.conf – 包含篩選條件和服務指令。 • Dockerfile	

任務	Description	所需的技能
建立自訂 Docker 映像。	1. 將目錄切換至 `customFluentBit` 。 2. 開啟 Amazon ECR 主控台，選擇`fluentbit_custom` 儲存庫，然後選擇檢視推送命令。 3. 上傳您的專案。 4. 上傳完成後，複製建置的 URL。當您在 Amazon ECS 中建立容器時，需要此 URL 如需詳細資訊，請參閱 Amazon ECR 文件中的推送 Docker 映像。	系統管理員、開發人員

設定 Amazon ECS 叢集

任務	Description	所需的技能
建立 Amazon ECS 叢集	遵循 Amazon ECS 文件中建立叢集的僅限聯網範本一節中的指示，建立 Amazon ECS 叢集。 **Note** 請務必選擇建立 VPC，為您的 Amazon ECS 叢集建立新的虛擬私有雲端 (VPC)。	系統管理員、開發人員

設定 Amazon ECS 任務

任務	Description	所需的技能
設定 Amazon ECS 任務執行 IAM 角色。	使用 `AmazonECSTaskExecutionRolePolicy` 受管政策建立 Amazon ECS 任務執行 IAM 角色。如需詳細資訊，請參閱 Amazon ECS 文件中的 Amazon ECS 任務執行 IAM 角色。 **Note** 請務必記錄 IAM 角色的 Amazon Resource Name (ARN)。	系統管理員、開發人員
將 IAM 政策連接至 Amazon ECS 任務執行 IAM 角色。	1. 使用 `firelens_policy.json` （連接的）政策文件建立 IAM 政策。如需詳細資訊，請參閱 IAM 文件中的在 JSON 標籤上建立政策。 2. 將此政策連接至您先前建立的 Amazon ECS 任務執行 IAM 角色。如需詳細資訊，請參閱 IAM 文件中的新增 IAM 政策 (AWS CLI)。	系統管理員、開發人員
設定 Amazon ECS 任務定義。	1. 更新`Task.json` 範例任務定義 （已連接） 中的下列區段： • `taskRoleArn` 使用任務執行 IAM 角色的	系統管理員、開發人員

任務	Description	所需的技能
	ARN 更新 `executionRoleArn` 和 • `containerDefinitions` 使用您先前建立的自訂 Fluent Bit Docker 映像更新 中的映像 • 使用`containerDefinitions` 應用程式映像的名稱更新 中的映像 2. 開啟 Amazon ECS 主控台，選擇任務定義，選擇建立新任務定義，然後在選取相容性頁面上選擇 Fargate。 3. 選擇透過 Json 設定，將更新`Task.json` 的檔案貼到文字區域，然後選擇儲存。 4. 建立任務定義。 如需詳細資訊，請參閱 Amazon ECS 文件中的建立任務定義。	

執行 Amazon ECS 任務

任務	Description	所需的技能
執行 Amazon ECS 任務。	在 Amazon ECS 主控台上，選擇叢集，選擇您先前建立的叢集，然後執行獨立任務。	系統管理員、開發人員

任務	Description	所需的技能
	如需詳細資訊，請參閱 Amazon ECS 文件中的執行獨立任務。	

驗證 CloudWatch 日誌

任務	Description	所需的技能
驗證日誌。	1. 開啟 CloudWatch 主控台，選擇日誌群組，然後選擇 `/aws/ecs/containerinsights/{{cluster_ARN}}/firelens/application` 。 2. 驗證日誌，特別是自訂剖析器新增的自訂欄位。 3. 使用 CloudWatch 根據自訂欄位篩選日誌。	系統管理員、開發人員

相關資源

- Amazon ECS 的 Docker 基本概念
- AWS Fargate 上的 Amazon ECS
- 設定基本服務參數

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 GitHub Actions 和 Terragrunt 建立 API 驅動的資源協同運作架構

Tamilselvan P、Abhigyan Dandriyal、Sandeep Gawande 和 Akash Kumar，Amazon Web Services

總結

此模式利用 GitHub Actions 工作流程，透過標準化 JSON 承載自動化資源佈建，無需手動設定。此自動化管道會管理完整的部署生命週期，並可無縫整合各種前端系統，從自訂 UI 元件到 ServiceNow。解決方案的彈性可讓使用者透過偏好的界面與系統互動，同時維持標準化程序。

可設定的管道架構可以適應不同的組織需求。範例實作著重於 Amazon Virtual Private Cloud (Amazon VPC) 和 Amazon Simple Storage Service (Amazon S3) 佈建。模式透過標準化整個組織的請求並提供一致的整合點，有效地解決常見的雲端資源管理挑戰。這種方法可讓團隊更輕鬆地請求和管理資源，同時確保標準化。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 可存取已設定儲存庫的作用中 GitHub 帳戶

限制

- 新資源需要手動將`terragrunt.hcl`檔案新增至儲存庫組態。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

下圖顯示此模式的元件和工作流程。

架構圖顯示下列動作：

1. 使用者向 GitHub 動作提交 JSON 承載，觸發自動化管道。
2. GitHub Actions 管道會根據承載規格，從 Terragrunt 和 Terraform 儲存庫擷取所需的資源程式碼。

3. 管道會使用指定的 AWS 帳戶 ID 擔任適當的 AWS Identity and Access Management (IAM) 角色。然後，管道會將資源部署到目標， AWS 帳戶 並使用帳戶特定的 Amazon S3 儲存貯體和 Amazon DynamoDB 資料表來管理 Terraform 狀態。

每個 都 AWS 帳戶 包含用於安全存取的 IAM 角色、用於 Terraform 狀態儲存的 Amazon S3 儲存貯體，以及用於狀態鎖定的 DynamoDB 資料表。此設計可讓您跨 進行受控制、自動化的資源部署 AWS 帳戶。部署程序會透過每個帳戶中的專用 Amazon S3 儲存貯體和 IAM 角色，維護適當的狀態管理和存取控制。

工具

AWS 服務

- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。

其他工具

- GitHub Actions 是與 GitHub 儲存庫緊密整合的持續整合和持續交付 (CI/CD) 平台。您可以使用 GitHub 動作來自動化建置、測試和部署管道。
- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。
- Terragrunt 是一種協同運作工具，可同時延伸 OpenTofu 和 Terraform 功能。它可管理一般基礎設施模式的套用方式，讓您更輕鬆地擴展和維護大型基礎設施資產。

程式碼儲存庫

此模式的程式碼可在 GitHub sample-aws-orchestration-pipeline-terraform 儲存庫中使用。

最佳實務

- 使用 GitHub 儲存庫秘密存放 AWS 登入資料和敏感資料，以安全存取。

- 設定 GitHub 動作的 OpenID Connect (OIDC) 提供者以擔任 IAM 角色，避免靜態登入資料。
- 遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。

史詩

建立和設定儲存庫

任務	Description	所需的技能
初始化 GitHub 儲存庫。	若要初始化 GitHub 儲存庫，請使用下列步驟： 1. 建立新的 GitHub 儲存庫以託管管道程式碼。 2. 匯入位於來源儲存庫 .github/workflows 資料夾的 `deployment.yml` 和 `workflow-trigger.yml` 檔案。	DevOps 工程師
設定 IAM 角色和許可。	若要設定 IAM 角色和許可，請使用下列步驟： 1. 使用 OpenID Connect (OIDC) 身分提供者 (IdP)，以 GitHub 動作連線至 中的動作 AWS所需的信任關係建立 IAM 角色。 2. 將必要的許可連接到 IAM 角色，以建立後端和所需的資源。如需詳細資訊，請參閱使用 Amazon VPC 以及後端 Amazon S3 儲存貯體和 DynamoDB 資料表建立 VPC 的範例政策。	DevOps 工程師

任務	Description	所需的技能

設定 GitHub 秘密和變數。	如需如何在 GitHub 儲存庫中設定儲存庫秘密和變數的指示，請參閱 GitHub 文件中的為儲存庫建立組態變數。設定下列變數：
	• 儲存庫秘密
	  • PAT_TOKEN – 具有執行 GitHub 操作許可的個人存取字符
	• 儲存庫變數
	  • aws_role – 具有適當許可的 IAM 角色 Amazon Resource Name (ARN)，可部署所需資源並將 GitHub 動作連線至 中的動作 AWS
		DevOps 工程師

建立儲存庫結構。	若要建立儲存庫結構，請使用下列步驟： 1. 在main分支中建立新的資料夾以存放terragrunt.hcl 檔案，以及資源建立後的輸出和變更日誌。 2. 根據您要透過資料夾佈建的資源類型，使用小寫命名資料夾。資料夾稍後會在承載中依原樣使用。如需詳細資訊，請參閱 儲存庫中 Amazon S3 和 Amazon VPC 的範例結構。	DevOps 工程師

觸發管道並驗證結果

任務	Description	所需的技能
使用 curl 執行管道。	若要使用 curl 執行管道，請使用下列步驟： 1. 在本機目錄中建立 payload.json 檔案。遵循承載檔案的定義結構，並在提交請求之前將其字串化。如需詳細資訊，請參閱 儲存庫中的範例承載。 2. 使用 GitHub API 在終端機上提交請求，如下列範例所示： `curl -X POST \ -H "Accept: application/vnd.github.v3+json" \ -H "Authorization: token YOUR_GITHUB_TOKEN" \ -d @payload.json \ https://api.github.com/repos/OWNER/REPO/actions/workflows/workflow-trigger.yml/dispatches` 在範例中，為下列項目提供您自己的值： • YOUR_GITHUB_TOKEN 使用您的 GitHub 個人存取字符 • OWNER 儲存庫擁有者的名稱	DevOps 工程師

任務	Description	所需的技能
	• REPO 使用儲存庫名稱 如需管道執行程序的詳細資訊，請參閱其他資訊。	

任務	Description	所需的技能
驗證管道執行的結果	若要驗證結果，請使用下列步驟： 1. 在儲存庫的動作索引標籤中監控 GitHub 動作工作流程執行。 2. 成功執行工作流程後，請使用 驗證資源建立 AWS 管理主控台 ，如下所示： a. 對於 Amazon VPC： • 導覽至指定 中的 Amazon VPC 服務 AWS 區域。 • 檢查是否有標籤符合您請求參數的新 VPC。 • 驗證 CIDR 區塊和其他組態。 b. 對於 Amazon S3： • 導覽至 Amazon S3 儲存貯體。 • 在一般用途儲存貯體中，檢查符合您請求參數的新儲存貯體。 • 驗證 S3 儲存貯體名稱和其他組態。 您也可以使用在與 檔案相同資源內的儲存庫中建立`output.json` 的檔案，跨驗證建立的資源`terragrunt.hcl` 。	DevOps 工程師

清除資源

任務	Description	所需的技能
提交清除請求。	若要刪除不再需要的資源，請使用下列步驟： 1. 使用相同的 API 端點提交刪除請求，但修改承載，如下所示： • 在 `delete`中`RequestType` 將 變更為 `RequestParameters` 。 • 讓所有其他參數保持與`create`請求相同。 2. 在 GitHub 動作中監控刪除程序。 3. 工作流程完成後，確認資源資料夾中`changelog.json` 的檔案顯示 的狀態`deleted`。 4. 使用 驗證資源移除 AWS 管理主控台。	DevOps 工程師

相關資源

AWS 部落格

- 使用 IAM 角色將 GitHub 動作連線至 中的動作 AWS

AWS 服務 文件

- IAM 角色建立
- 使用 Amazon CloudWatch Logs 監控 CloudTrail 日誌檔案

- Amazon S3 的安全最佳實務

GitHub 資源

- 建立儲存庫分派事件
- 建立 Webhook
- 在 GitHub 儲存庫上實作強大的存取控制
- 定期稽核儲存庫存取權
- CI/CD 管道中的安全檢查
- 對 GitHub 帳戶使用多重驗證

其他資訊

管道執行程序

以下是管道執行的步驟：

1. 驗證 JSON 承載格式 - 確保傳入的 JSON 組態結構正確，並包含所有必要的參數
2. 擔任指定的 IAM 角色 - 驗證並擔任 AWS 操作所需的 IAM 角色
3. 下載必要的 Terraform 和 Terragrunt 程式碼 - 擷取指定的資源程式碼和相依性版本
4. 執行資源部署 - 套用組態以在目標環境中部署或更新 AWS 資源

用於建立 VPC 的範例承載

以下是 Terraform 後端狀態儲存貯體建立的範例程式碼：

```
state_bucket_name = "${local.payload.ApplicationName}-${local.payload.EnvironmentId}-
tfstate"
```

```
lock_table_name = "${local.payload.ApplicationName}-${local.payload.EnvironmentId}-
tfstate-lock"
```

以下是使用 Amazon VPC 建立 VPC 的範例承載，其中 `vpc_cidr` 定義 VPC 的 CIDR 區塊規格。Terraform 狀態儲存貯體會映射至`terraform`檔案中定義的變數。`ref` 參數包含要執行之程式碼的分支名稱。

```
{
    "ref": "main",
    "inputs": {
        "RequestParameters": {
            "RequestId": "1111111",
            "RequestType": "create",
            "ResourceType": "vpc",
            "AccountId": "1234567890",
            "AccountAlias": "account-alias",
            "RegionId": "us-west-2",
            "ApplicationName": "myapp",
            "DivisionName": "division-name",
            "EnvironmentId": "dev",
            "Suffix": "poc"
        },
        "ResourceParameters": [
            {
                "VPC": {
                    "vpc_cidr": "10.0.0.0/16"
                }
            }
        ]
    }
}
```

RequestParameters 用於追蹤管道區段中的請求狀態tfstate，並根據此資訊建立。下列參數包含中繼資料和控制項資訊：

- RequestId – 請求的唯一識別符
- RequestType – 操作類型 （建立、更新或刪除）
- ResourceType – 要佈建的資源類型
- AccountId – 部署 AWS 帳戶 的目標
- AccountAlias – 的易記名稱 AWS 帳戶
- RegionId – AWS 區域 用於資源部署
- ApplicationName – 應用程式的名稱
- DivisionName – 組織分處
- EnvironmentId – 環境 （例如，dev 和 prod)
- Suffix – 資源的其他識別符

ResourceParameters 包含資源特定的組態，對應至 Terraform 檔案中定義的變數。任何需要傳遞至 Terraform 模組的自訂變數都應包含在 中ResourceParameters。Amazon VPC vpc_cidr必須使用 參數。

使用 GitHub Actions 建立 Terraform 受管 AWS 基礎設施的自動提取請求

Matt Padgett、Ashish Bhatt、Ashwin Divakaran、Sandip Gangapadhyay 和 Prafful Gupta，Amazon Web Services

總結

此模式提供自動化公用程式，旨在消除在多個 Terraform 儲存庫中管理變更時涉及的手動重複工作。許多組織使用 Terraform 儲存庫以程式碼 (IaC) 的形式管理其基礎設施，通常具有數百個代表不同環境、服務或團隊的個別儲存庫。大規模管理這些儲存庫會帶來重大的操作挑戰。更新參數、升級模組版本或套用組態變更等例行任務，通常需要每天多次跨許多儲存庫建立和管理提取請求 (PRs)。

即使是簡單的變更，這個重複的手動程序也非常耗時且容易出錯。工程師必須一致地在所有目標儲存庫中套用相同的變更，並製作有意義的 PR 標題和描述。此外，他們通常必須與 Jira 等外部工具互動，以擷取或包含問題追蹤參考。這些任務雖然必要，但是無差別的繁重工作，會耗用寶貴的工程時間並降低整體效率。此工作流程中缺乏自動化會造成摩擦、減慢交付速度，並增加負責維護大規模 Terraform 基礎設施之團隊的認知負擔。

解決方案概觀

為了解決此挑戰，此模式提供完全以組態為導向的公用程式，可讓使用者在結構化組態檔案中定義所需的變更。此檔案會使用明確定義的結構描述來指定目標儲存庫、模組、參數和值。

設定完成後，公用程式會執行下列自動化步驟：

1. 讀取使用者定義的組態，以判斷變更的範圍和性質
2. 在每個目標儲存庫中建立新的分支，並套用必要的更新
3. 為每個變更產生 PR，確保所有儲存庫的一致性
4. 傳送 Slack 通知（選用），以使用建立PRs 的直接連結來提醒利益相關者

透過自動化這些重複性任務，公用程式可大幅減少與管理大規模基礎設施更新相關的時間、精力和風險。它可讓團隊專注於更高價值的工程工作，同時協助確保一致地套用變更，並可追蹤所有儲存庫。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。

- Python 3.8 版或更新版本。
- GitHub 個人存取字符 (PAT)。如需詳細資訊，請參閱 GitHub 文件中的建立個人存取字符（傳統）。
- GitHub PAT 可以存取您的目標儲存庫，讓公用程式可以執行建立分支和提取請求等操作。如需詳細資訊，請參閱此模式的 GitHub 程式碼儲存庫。

限制

- 組態複雜性代表主要挑戰。自動化的有效性受到其組態檔案的功能限制。雖然系統有效率地處理標準變更，但複雜的基礎設施修改可能需要手動介入，而且某些邊緣案例仍超出自動處理的範圍。
- 安全性和存取提供重要的考量，特別是在管理 GitHub 存取權杖和 API 速率限制時。組織必須仔細平衡自動化與安全登入資料儲存和管理的需求，確保適當的存取控制，同時維持營運效率。
- 驗證限制構成另一個值得注意的限制，因為自動化系統驗證業務邏輯和環境特定需求的能力有限。複雜的相依性和跨服務互動通常需要人工監督，因為自動化驗證無法完全擷取所有情境細節和業務規則。
- 處理大規模基礎設施變更時，會出現規模和效能問題。系統必須在 GitHub API 限制內操作，同時同時管理多個儲存庫。跨廣泛基礎設施的資源密集型操作可能會產生需要謹慎管理的效能瓶頸。
- 整合界限會限制系統的靈活性，因為它主要設計用於 GitHub 和 Slack 等特定工具。使用不同工具的組織可能需要自訂解決方案，且此模式的工作流程自訂選項僅限於支援的整合點。

Architecture

下圖顯示此解決方案的工作流程和元件。

工作流程由以下步驟組成：

1. 開發人員透過指定 Terraform 儲存庫來觸發 GitHub 動作。
2. 自動化公用程式會讀取定義的組態。
3. 自動化公用程式也會提取提供的 Terraform 儲存庫。
4. 自動化公用程式會建立新的分支，並在本機更新 Terraform 範本。
5. 自動化公用程式會將新的分支推送至儲存庫，並建立新的 PR。
6. 自動化公用程式使用包含 PR 連結的 Slack 通知來通知開發人員，並啟用 Terraform 範本以進行 AWS 雲端 部署。

工具

- GitHub 是一種開發人員平台，開發人員可用來建立、存放、管理和共用其程式碼。
- GitHub Actions 是與 GitHub 儲存庫緊密整合的持續整合和持續交付 (CI/CD) 平台。您可以使用 GitHub 動作來自動化建置、測試和部署管道。
- HashiCorp Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。
- Slack 是 Salesforce 產品，是一種採用 AI 技術的對話平台，可提供聊天和視訊協作、自動化沒有程式碼的程序，並支援資訊共用。

程式碼儲存庫

此模式的程式碼可在使用 GitHub 動作儲存庫的 GitHub 自動化 Terraform 基礎設施更新工作流程中取得。

最佳實務

- 有效的變更管理對於成功實作至關重要。組織應針對大規模變更採用逐步推展策略。維持一致的分支命名慣例和 PR 描述，並確保所有變更的完整文件。
- 必須嚴格實作安全控制，專注於最低權限的存取原則和安全登入資料管理。啟用分支保護規則以防止未經授權的變更。定期執行安全稽核以維護系統完整性。
- 強大的測試通訊協定應包括在持續整合和持續部署 (CI/CD) 管道中自動`terraform plan`執行。通訊協定也應該包含遞交前驗證檢查，以及針對重大變更的專用審核環境。這種多層測試方法有助於及早發現問題，並確保基礎設施的穩定性。
- 監控策略需要包含全面的提醒機制、詳細的成功/失敗指標追蹤，以及失敗操作的自動重試機制。此策略有助於確保營運可見性，並能夠快速回應任何出現的問題。
- 組態標準應強調所有組態的版本控制，維護模組化以實現可重複使用性和可擴展性。清楚的結構描述和範例文件，可協助團隊有效地了解和使用自動化系統。

史詩

設定和安裝

任務	Description	所需的技能

設定儲存庫。	若要設定儲存庫，請執行下列命令：
	
	`# Clone the automation tool repository`
	`git clone https://github.com/aws-samples/sample-terraform-pr-automation-utility`
	`cd sample-terraform-pr-automation-utility`
	
	`# Copy example configuration`
	`cp config.example.yaml config.yaml`
		AWS DevOps

安裝依存項目。	若要安裝和驗證 Python 相依性，請執行下列命令：
	
	`# Install Python dependencies`
	`pip3 install -r requirements.txt`
	
	`# Verify installation`
	`python3 -c "import github; import hcl2; import yaml; import requests; print('All packages installed successfully')"`
		AWS DevOps

任務	Description	所需的技能

設定 GitHub 字符。	若要設定 GitHub 字符，然後驗證其是否正常運作，請執行下列命令：
	
	`# Set GitHub token environment variable`
	`export GITHUB_TOKEN="your_github_token_here"`
	
	`# Verify token works`
	`curl -H "Authorization: token $GITHUB_TOKEN" https://api.github.com/user`
		AWS DevOps

設定 Terraform 變更的組態檔案

任務	Description	所需的技能

設定 `config.yaml` 檔案。	若要定義您的目標儲存庫和所需的變更，請編輯 `config.yaml` 檔案，如下所示：
	
	`repositories:`
	`  - owner: "your-org"`
	`  repo: "your-terraform-repo"`
	`  files:`
	`  - path: "variables.tf"`
	`  changes:`
	`  variables:`
	`  - app_version:`
	`  default:`
	
	`update:`
		AWS DevOps

任務	Description	所需的技能

	<pre>-
	from: ["1.0.0"]
	
	to: "1.1.0"
	
	settings:
	pr_title_template:
	"Infrastructure Update
	- {{timestamp}}"
	slack:
	username: "Terrafor
	m Bot"
	icon_emoji:
	":terraform:"
	notify_on_success:
	true
	notify_on_error:
	true
	notify_batch_summa
	ry: true</pre>	

測試和驗證

任務	Description	所需的技能

執行飛行前測試。	在生產儲存庫上執行組態之前，請務必先測試組態。使用下列命令：
	
	<pre># 1. Test configuration
	syntax
	python3 -c "from
	main import get_confi
	g_content; get_confi
	g_content()"
	
	# 2. Run in dry-run mode
	first</pre>
		AWS DevOps

任務	Description	所需的技能

	`DRY_RUN=true python3 main.py`
	
	`# 3. Test with minimal configuration`
	`# Use a simple config.yaml with just one repository and one change`	

驗證儲存庫存取。	若要驗證 GitHub 字符是否可以存取儲存庫，請執行下列命令：
	
	`# Test GitHub token access`
	`curl -H "Authorization: token $GITHUB_TOKEN" \`
	`  https://api.github.com/repos/owner/repo-name`
	
	`# Should return repository information, not 404`
		AWS DevOps

執行自動化公用程式

任務	Description	所需的技能
使用 GitHub Actions UI 執行自動化公用程式。	若要使用 GitHub Actions UI 執行自動化公用程式，請執行下列動作： 1. 導覽至 GitHub 上的儲存庫。	AWS DevOps

任務	Description	所需的技能
	2. 選擇動作索引標籤。 3. 選擇 Terraform Infrastructure Update Automation 工作流程。 4. 選擇執行工作流程。 5. 設定下列工作流程輸入： • 來源組態： • 選取自動化的目標分支。 • 指定包含所需變更的組態檔案 (`config.yaml`) 路徑。 • 預覽控制項： • 選擇預覽選項來檢閱變更，而不套用變更。 • 分支管理： • 輸入基本分支以建立新的功能分支。 • 輸入分支字首組態。 • 選擇自動關閉過時的提取請求核取方塊。 • 通知設定： • 輸入 Slack Webhook URL 的 URL 以取得豐富的通知 （覆寫儲存庫秘密）。 • 選擇測試 Slack 整合後再處理 （僅限試執行）核取方塊。 • 輸入覆寫預設 Slack 頻道的資訊 （例如 #infrastructure-test)。	

任務	Description	所需的技能
	• 進階設定： • 選擇啟用偵錯記錄以進行故障診斷核取方塊。	

（替代）從命令列執行自動化公用程式。	如果您願意，您可以從命令列執行自動化公用程式，而不是使用 GitHub Actions UI。使用下列命令：
	`# Run actual automation`
	`python3 main.py`
		AWS DevOps

驗證 PRs和變更

任務	Description	所需的技能
檢閱建立的 PRs和變更。	若要監控 GitHub 工作流程執行的結果，請執行下列動作： • 檢查工作流程執行日誌的處理狀態。 • 檢閱自動化公用程式建立 PRs。 • 監控 Slack 通知（如果已設定）。	AWS DevOps

清除

任務	Description	所需的技能
（選用）清除 PRs。	關閉已捨棄或不必要的 PRs。	AWS DevOps

相關資源

AWS 方案指引

- 使用 Terraform 做為 的 IaC 工具 AWS 雲端

GitHub 文件

- 關於提取請求
- 管理您的個人存取字符
- 了解 GitHub 動作
- GitHub 動作的快速入門

自動為 Java 和 Python 專案建立動態 CI 管道

Aromal Raj Jayarajan、Vijesh Vijayakumaran Nair、MAHESH RAGHUNANDANAN 和 Amarnath Reddy，Amazon Web Services

總結

此模式說明如何使用 AWS 開發人員工具，為 Java 和 Python 專案自動建立動態持續整合 (CI) 管道。

隨著技術堆疊多樣化和開發活動增加，建立和維護跨組織一致 CI 管道可能會變得困難。透過在 AWS Step Functions 中自動化程序，您可以確保 CI 管道的使用方式和方法一致。

為了自動建立動態 CI 管道，此模式使用以下變數輸入：

- 程式設計語言 （僅限 Java 或 Python)
- 管道名稱
- 必要的管道階段

> **Note**
>
> Step Functions 使用多個 AWS 服務協調管道建立。如需有關此解決方案中使用的 AWS 服務的詳細資訊，請參閱此模式的工具一節。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 正在部署此解決方案的相同 AWS 區域中的 Amazon S3 儲存貯體
- 具有建立此解決方案所需資源所需的 AWS CloudFormation 許可的 AWS Identity and Access Management (IAM) 委託人

限制

- 此模式僅支援 Java 和 Python 專案。
- 在此模式中佈建的 IAM 角色遵循最低權限原則。必須根據 CI 管道需要建立的特定資源更新 IAM 角色的許可。

Architecture

目標技術堆疊

- AWS CloudFormation
- AWS CodeBuild
- AWS CodeCommit
- AWS CodePipeline
- IAM
- Amazon Simple Storage Service (Amazon S3)
- AWS Systems Manager
- AWS Step Functions
- AWS Lambda
- Amazon DynamoDB

目標架構

下圖顯示使用 AWS 開發人員工具自動建立 Java 和 Python 專案動態 CI 管道的範例工作流程。

該圖顯示以下工作流程：

1. AWS 使用者以 JSON 格式提供 CI 管道建立的輸入參數。此輸入會啟動 Step Functions 工作流程 (狀態機器)，以使用 AWS 開發人員工具建立 CI 管道。
2. Lambda 函數會讀取名為 input-reference 的資料夾，該資料夾存放在 Amazon S3 儲存貯體中，然後產生 buildspec.yml 檔案。此產生的檔案會定義 CI 管道階段，並存放在存放參數參考的相同 Amazon S3 儲存貯體中。
3. Step Functions 會檢查 CI 管道建立工作流程的相依性是否有任何變更，並視需要更新相依性堆疊。
4. Step Functions 在 CloudFormation 堆疊中建立 CI 管道資源，包括 CodeCommit 儲存庫、CodeBuild 專案和 CodePipeline 管道。
5. CloudFormation 堆疊會將所選技術堆疊 (Java 或 Python) 的範例原始碼和 buildspec.yml 檔案複製到 CodeCommit 儲存庫。
6. CI 管道執行時間詳細資訊會存放在 DynamoDB 資料表中。

自動化和擴展

- 此模式僅適用於單一開發環境。需要在多個開發環境中使用組態變更。
- 若要新增對多個 CloudFormation 堆疊的支援，您可以建立其他 CloudFormation 範本。如需詳細資訊，請參閱 CloudFormation 文件中的 AWS CloudFormation 入門。 CloudFormation

工具

工具

- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務來建置業務關鍵應用程式。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS CodeBuild 是一項全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS CloudFormation 可協助您設定 AWS 資源、快速一致地佈建資源，以及在整個 AWS 帳戶和區域的生命週期中管理這些資源。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- AWS Systems Manager 參數存放區為組態資料管理和秘密管理提供安全的階層式儲存。

Code

此模式的程式碼可在 GitHub automated-ci-pipeline-creation 儲存庫中使用。儲存庫包含建立此模式中概述的目標架構所需的 CloudFormation 範本。

最佳實務

- 請勿直接在 CloudFormation 範本或 Step Functions 動作組態中輸入登入資料 (秘密)，例如字符或密碼。如果您這麼做，資訊會顯示在 DynamoDB 日誌中。反之，請使用 AWS Secrets Manager 來設定和存放秘密。然後，視需要參考 CloudFormation 範本和 Step Functions 動作組態中存放在 Secrets Manager 中的秘密。如需詳細資訊，請參閱 AWS Secrets Manager Secrets Manager。
- 為存放在 Amazon S3 中的 CodePipeline 成品設定伺服器端加密。如需詳細資訊，請參閱 CodePipeline 文件中的為存放在 Amazon S3 for CodePipeline 中的成品設定伺服器端加密 CodePipeline。 CodePipeline
- 設定 IAM 角色時套用最低權限許可。如需詳細資訊，請參閱 IAM 文件中的套用最低權限許可。
- 請確定您的 Amazon S3 儲存貯體不可公開存取。如需詳細資訊，請參閱《Amazon S3 S3 文件》中的設定 S3 儲存貯體的封鎖公開存取設定。
- 請務必為 Amazon S3 儲存貯體啟用版本控制。如需詳細資訊，請參閱《Amazon S3 文件》中的在 S3 儲存貯體中使用版本控制。 Amazon S3
- 設定 IAM 政策時使用 IAM Access Analyzer。此工具提供可行的建議，協助您撰寫安全且實用的 IAM 政策。如需詳細資訊，請參閱 IAM 文件中的使用 AWS Identity and Access Management Access Analyzer。
- 盡可能在設定 IAM 政策時定義特定存取條件。
- 啟用 Amazon CloudWatch 記錄以進行監控和稽核。如需詳細資訊，請參閱 CloudWatch 文件中的什麼是 Amazon CloudWatch Logs？。 CloudWatch

史詩

設定先決條件

任務	Description	所需的技能
建立 Amazon S3 儲存貯體。	建立 Amazon S3 儲存貯體（或使用現有的儲存貯體），以存放解決方案所需的 CloudFormation 範本、原始程式碼和輸入檔案。 如需詳細資訊，請參閱 Amazon S3 文件中的步驟	AWS DevOps

任務	Description	所需的技能
	1：建立您的第一個 S3 儲存貯體。 Amazon S3 **Note** Amazon S3 儲存貯體必須位於您要部署解決方案的相同 AWS 區域。	
複製 GitHub 儲存庫。	在終端機視窗中執行下列命令，複製 GitHub automated-ci-pipeline-creation 儲存庫： `git clone https://github.com/aws-samples/automated-ci-pipeline-creation.git` 如需詳細資訊，請參閱 GitHub 文件中的複製儲存庫。	AWS DevOps

任務	Description	所需的技能
將解決方案範本資料夾從複製的 GitHub 儲存庫上傳至您的 Amazon S3 儲存貯體。	從複製的 Solution-Templates 資料夾複製內容，並將其上傳至您建立的 Amazon S3 儲存貯體。 如需詳細資訊，請參閱 Amazon S3 文件中的上傳物件。 Amazon S3 **Note** 請務必僅上傳 Solution-Templates 資料夾的內容。您只能在 Amazon S3 儲存貯體的根層級上傳檔案。	AWS DevOps

部署解決方案

任務	Description	所需的技能
在複製的 GitHub 儲存庫中使用 template.yml 檔案，建立 CloudFormation 堆疊來部署解決方案。	1. 登入 AWS 管理主控台，然後開啟 AWS CloudFormation 主控台。 2. 選擇建立堆疊。下拉式清單隨即出現。 3. 在下拉式清單中，選取使用新資源 （標準）。建立堆疊頁面隨即開啟。 4. 在指定範本區段中，選取上傳範本檔案旁的核取方塊。 5. 選取 Choose file (選擇檔案)。然後，導覽至複製的	AWS 管理員、AWS DevOps

任務	Description	所需的技能
	GitHub 儲存庫的根資料夾，然後選取 template.yml 檔案。然後選擇 Open (開啟)。 6. 選擇下一步。指定堆疊詳細資訊頁面隨即開啟。 7. 在參數區段中，指定下列參數： • 針對 S3TemplateBucketName，輸入您先前建立的 Amazon S3 儲存貯體名稱，其中包含此解決方案的原始程式碼和參考。請確定儲存貯體名稱參數為小寫。 • 針對 DynamoDBTable，輸入 CloudFormation 堆疊建立的 DynamoDB 資料表名稱。 • 針對 StateMachineName，輸入 CloudFormation 堆疊建立之 Step Functions 狀態機器的名稱。 8. 選擇下一步。設定堆疊選項頁面隨即開啟。 9. 在 Configure stack options (設定堆疊選項) 頁面，選擇 Next (下一步)。請勿變更任何預設值。檢閱頁面隨即開啟。	

任務	Description	所需的技能
	10.檢閱堆疊建立設定。然後，選擇建立堆疊以啟動您的堆疊。 **Note** 建立堆疊時，它會列在 Stacks 頁面上，狀態為 CREATE_IN_PROGRESS。在完成此模式中的剩餘步驟之前，請務必等待堆疊的狀態變更為 CREATE_COMPLETE。	

測試設定

任務	Description	所需的技能
執行您建立的步驟函數。	1. 登入 AWS 管理主控台，然後開啟 Step Functions 主控台。 2. 開啟您建立的步驟函數。 3. 選擇 Start execution (開始執行)。然後，以 JSON 格式輸入工作流程的輸入值（請參閱下列範例輸入）。 4. 選擇 Start execution (開始執行)。 JSON 格式	AWS 管理員、AWS DevOps

任務	Description	所需的技能

	<pre>{
	  "details": {
	  "tech_stack":
	 "Name of the Tech Stack
	 (python/java)",
	  "project_name":
	 "Name of the Project
	 that you want to create
	 with",
	  "pre_build":
	 "Choose the step if
	 it required in the
	 buildspec.yml file
	 i.e., yes/no",
	  "build": "Choose the
	 step if it required in
	 the buildspec.yml file
	 i.e., yes/no",
	  "post_build":
	 "Choose the step if
	 it required in the
	 buildspec.yml file
	 i.e., yes/no",
	  "reports": "Choose
	 the step if it required
	 in the buildspec.yml
	 file i.e., yes/no",
	  }
	}</pre>
	Java JSON 輸入範例
	<pre>{
	  "details": {
	  "tech_stack":
	 "java",
	  "project_name":
	 "pipeline-java-pjt",
	  "pre_build": "yes",
	  "build": "yes",</pre>	

任務	Description	所需的技能
	<pre>"post_build": "yes", "reports": "yes" } }</pre> Python JSON 輸入範例 <pre>{ "details": { "tech_stack": "python", "project_name": "pipeline-python-pjt", "pre_build": "yes", "build": "yes", "post_build": "yes", "reports": "yes" } }</pre>	
確認已建立 CI 管道的 CodeCommit 儲存庫。	1. 登入 AWS 管理主控台，然後開啟 CodeCommit 主控台。 2. 在儲存庫頁面上，確認您建立的 CodeCommit 儲存庫名稱出現在儲存庫清單中。儲存庫的名稱會附加以下項目： pipeline-java-pjt-Repo 3. 開啟 CodeCommit 儲存庫，並驗證範例原始程式碼與 buildspec.yml 檔案是否一起推送至主分支。	AWS DevOps

任務	Description	所需的技能
檢查 CodeBuild 專案資源。	1. 登入 AWS 管理主控台，然後開啟 CodeBuild 主控台。 2. 在建置專案頁面上，確認您建立的 CodeBuild 專案名稱出現在專案清單中。專案的名稱會附加以下項目：pipeline-java-pjt-Build 3. 選取 CodeBuild 專案的名稱以開啟專案。然後，檢閱並驗證下列組態： • 專案組態 • 來源 • Environment (環境) • Buildspec • 批次組態 • 成品	AWS DevOps
驗證 CodePipeline 階段。	1. 登入 AWS 管理主控台，然後開啟 CodePipeline 主控台。 2. 在管道頁面上，確認您建立的管道名稱出現在管道清單中。管道的名稱會附加以下項目： pipeline-java-pjt-Pipeline 3. 選取管道的名稱以開啟管道。然後，檢閱和驗證管道的每個階段，包括遞交和部署。	AWS DevOps

任務	Description	所需的技能
確認 CI 管道已成功執行。	1. 在 CodePipeline 主控台的管道頁面上，選取管道的名稱以檢視管道的狀態。 2. 確認管道的每個階段都有成功狀態。	AWS DevOps

清除您的資源

任務	Description	所需的技能
刪除 CloudFormation 中的資源堆疊。	在 CloudFormation 中刪除 CI 管道的資源堆疊。 如需詳細資訊，請參閱 CloudFormation 文件中的刪除 AWS CloudFormation 主控台上的堆疊。 CloudFormation Note 請務必刪除名為 <project_name>-stack 的堆疊。	AWS DevOps
在 Amazon S3 和 CloudFormation 中刪除 CI 管道的相依性。	1. 清空名為 DeploymentArtifactBucket 的 Amazon S3 儲存貯體。如需詳細資訊，請參閱 Amazon S3 文件中的清空儲存貯體。 Amazon S3 2. 在 CloudFormation 中刪除 CI 管道的相依性堆疊。如需詳細資訊，請參閱 CloudFormation 文件中的刪	AWS DevOps

任務	Description	所需的技能
	除 AWS CloudFormation 主控台上的堆疊。 CloudFormation **Note** 請務必刪除名為 pipeline-creation-dependencies-stack 的堆疊。	
刪除 Amazon S3 範本儲存貯體。	刪除您在此模式的設定先決條件區段中建立的 Amazon s3 儲存貯體，該區段存放此解決方案的範本。 如需詳細資訊，請參閱 Amazon S3 文件中的刪除儲存貯體。	AWS DevOps

相關資源

- 建立使用 Lambda 的 Step Functions 狀態機器 (AWS Step Functions 文件）
- AWS Step Functions WorkFlow Studio (AWS Step Functions 文件）
- DevOps 和 AWS
- AWS CloudFormation 如何運作？ (AWS CloudFormation 文件）
- 使用 AWS CodeCommit、AWS CodeBuild、AWS CodeDeploy 和 AWS CodePipeline 完成 CI/CD (AWS 部落格文章）
- IAM 和 AWS STS 配額、名稱要求和字元限制 (IAM 文件）

使用 Terraform 部署 CloudWatch Synthetics Canary

Dhrubajyoti Mukherjee 和 Jean-Francois Landreau，Amazon Web Services

總結

請務必從客戶的角度驗證系統的運作狀態，並確認客戶能夠連線。當客戶不持續呼叫端點時，這會更困難。Amazon CloudWatch Synthetics 支援建立 Canary，可測試公有和私有端點。透過使用 Canary，即使系統未使用，您也可以知道系統的狀態。這些 Canary 可以是 Node.js Puppeteer 指令碼或 Python Selenium 指令碼。

此模式說明如何使用 HashiCorp Terraform 部署測試私有端點的 Canary。它會嵌入 Puppeteer 指令碼，以測試 URL 是否傳回 `200-OK`。然後，Terraform 指令碼可以與部署私有端點的指令碼整合。您也可以修改解決方案來監控公有端點。

先決條件和限制

先決條件

- 具有虛擬私有雲端 (VPC) 和私有子網路的作用中 Amazon Web Services (AWS) 帳戶
- 可從私有子網路到達的端點 URL
- 部署環境中安裝的 Terraform

限制

目前的解決方案適用於下列 CloudWatch Synthetics 執行時間版本：

- syn-nodejs-puppeteer-3.4
- syn-nodejs-puppeteer-3.5
- syn-nodejs-puppeteer-3.6
- syn-nodejs-puppeteer-3.7

隨著新的執行時間版本發佈，您可能需要更新目前的解決方案。您也需要修改解決方案，以跟上安全性更新。

產品版本

- Terraform 1.3.0

Architecture

Amazon CloudWatch Synthetics 是以 CloudWatch、Lambda 和 Amazon Simple Storage Service (Amazon S3) 為基礎。Amazon CloudWatch 提供精靈來建立 Canary，以及顯示 Canary 執行狀態的儀表板。Lambda 函數會執行指令碼。Amazon S3 會存放 Canary 執行的日誌和螢幕擷取畫面。

此模式透過部署在目標子網路中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體來模擬私有端點。Lambda 函數需要部署私有端點之 VPC 中的彈性網路介面。

上圖顯示以下項目：

1. Synthetics Canary 會啟動 Canary Lambda 函數。
2. Canary Lambda 函數會連接至彈性網路界面。
3. Canary Lambda 函數會監控端點的狀態。
4. Synthetics Canary 會將執行資料推送至 S3 儲存貯體和 CloudWatch 指標。
5. CloudWatch 警示會根據指標啟動。
6. CloudWatch 警示會啟動 Amazon Simple Notification Service (Amazon SNS) 主題。

工具

AWS 服務

- Amazon CloudWatch 可協助您即時監控 AWS 資源的指標，以及您在 AWS 上執行的應用程式。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而不需要佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路類似於您在自己的資料中心內操作的傳統網路，具有使用可擴展的 AWS 基礎設施的優勢。此模式使用 VPC 端點和彈性網路界面。

其他服務

- HashiCorp Terraform 是一種開放原始碼基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。此模式使用 Terraform 來部署基礎設施。
- Puppeteer 是 Node.js 程式庫。CloudWatch Synthetics 執行時間使用 Puppeteer 架構。

Code

解決方案可在 GitHub 雲端 watch-synthetics-canary-terraform 儲存庫中使用。如需詳細資訊，請參閱其他資訊一節。

史詩

實作監控私有 URL 的解決方案

任務	Description	所需的技能
收集監控私有 URL 的需求。	收集完整的 URL 定義：網域、參數和標頭。若要私下與 Amazon S3 和 Amazon CloudWatch 通訊，請使用 VPC 端點。請注意，端點如何存取 VPC 和子網路。考慮 Canary 執行的頻率。	雲端架構師、網路管理員
修改現有的解決方案以監控私有 URL。	修改 `terraform.tfvars` 檔案： • name – Canary 的名稱。 • `runtime_version` – Canary 的執行時間版本。建議使用 syn-nodejs-puppeteer-3.7。 • `take_screenshot` – 是否應擷取螢幕擷取畫面。 • `api_hostname` – 受監控端點的主機名稱。 • `api_path` – 受監控端點的路徑。	雲端架構師

任務	Description	所需的技能
	• `vpc_id` – Canary Lambda 函數使用的 VPC ID。 • `subnet_ids` – Canary Lambda 函數所使用的子網路 IDs。 • `frequency` – Canary 的執行頻率，以分鐘為單位。 • `alert_sns_topic` – 傳送 CloudWatch 警示通知的 SNS 主題。	

部署和操作解決方案。	若要部署解決方案，請執行下列動作：
	1. 從開發環境中的 `cloudwatch-synthetics-canary-terraform` 目錄中，初始化 Terraform。
	`terraform init`
	2. 規劃和檢閱變更。
	`terraform plan`
	3. 部署解決方案。
	`terraform apply`
		雲端架構師、DevOps 工程師

疑難排解

問題	解決方案
刪除佈建的資源會卡住。	依該順序手動刪除 Canary Lambda 函數、對應的彈性網路介面和安全群組。

相關資源

- 使用合成監控
- 使用 Amazon CloudWatch Synthetics 監控 API Gateway 端點（部落格文章）

其他資訊

儲存庫成品

儲存庫成品位於下列結構中。

```
.
### README.md
### main.tf
### modules
#   ### canary
#   ### canary-infra
### terraform.tfvars
### tf.plan
### variable.tf
```

`main.tf` 檔案包含核心模組，並部署兩個子模組：

- `canary-infra` 部署 Canary 所需的基礎設施。
- `canary` 部署 Canary。

解決方案的輸入參數位於 `terraform.tfvars` 檔案中。您可以使用下列程式碼範例來建立一個 Canary。

```
module "canary" {
```

```
    source = "./modules/canary"
    name   = var.name
    runtime_version = var.runtime_version
    take_screenshot = var.take_screenshot
    api_hostname = var.api_hostname
    api_path = var.api_path
    reports-bucket = module.canary_infra.reports-bucket
    role = module.canary_infra.role
    security_group_id = module.canary_infra.security_group_id
    subnet_ids = var.subnet_ids
    frequency = var.frequency
    alert_sns_topic = var.alert_sns_topic
}
```

對應的 .var 檔案如下。

```
name   = "my-canary"
runtime_version = "syn-nodejs-puppeteer-3.7"
take_screenshot = false
api_hostname = "mydomain.internal"
api_path = "/path?param=value"
vpc_id = "vpc_id"
subnet_ids = ["subnet_id1"]
frequency = 5
alert_sns_topic = "arn:aws:sns:eu-central-1:111111111111:yyyyy"
```

清除解決方案

如果您在開發環境中測試此項目，您可以清除解決方案，以避免產生成本。

1. 在 AWS 管理主控台上，導覽至 Amazon S3 主控台。清空解決方案建立的 Amazon S3 儲存貯體。如有必要，請務必備份資料。
2. 在您的開發環境中，從 `cloudwatch-synthetics-canary-terraform`目錄執行 `destroy`命令。

   ```
   terraform destroy
   ```

在聊天應用程式自訂動作和 中使用 Amazon Q Developer 部署 ChatOps 解決方案來管理 SAST 掃描結果 CloudFormation

Anand Bukkapatnam Tirumala，Amazon Web Services

總結

此模式提供全方位的解決方案，可在聊天應用程式中使用 Amazon Q Developer，以簡化透過 SonarQube 回報的靜態應用程式安全測試 (SAST) 掃描失敗的管理。這種創新方法將自訂動作和通知整合到對話界面中，從而在開發團隊中實現高效的協作和決策過程。

在現今步調快速的軟體開發環境中，有效管理 SAST 掃描結果對於維護程式碼品質和安全性至關重要。不過，許多組織都面臨下列重大挑戰：

- 由於通知系統效率低下，導致對關鍵漏洞的延遲意識
- 中斷連線核准工作流程所造成的決策程序緩慢
- 缺乏對 SAST 掃描失敗的立即、可行回應
- 有關安全調查結果的分段通訊和協作
- 安全工具的耗時且容易出錯的手動基礎設施設定

這些問題通常會導致提高安全風險、延遲發佈和降低團隊生產力。為了有效地解決這些挑戰，需要一個解決方案，可以簡化 SAST 結果管理、增強團隊協作，以及自動化基礎設施佈建。

解決方案的主要功能包括：

- 自訂通知 – 即時提醒和通知會直接交付至團隊聊天管道，以確保對 SAST 掃描漏洞或故障的及時意識和動作。
- 對話式核准 – 利益相關者可以在聊天界面中順暢地啟動和完成 SAST 掃描結果的核准工作流程，從而加速決策程序。
- 自訂動作 – 團隊可以根據 SAST 掃描結果定義和執行自訂動作，例如針對品質閘道故障自動觸發電子郵件訊息、增強對安全問題的回應能力。
- 集中式協作 – 所有與 SAST 掃描相關的討論、決策和動作都保存在統一的聊天環境中，促進團隊成員之間的協作和知識分享獲得改善。
- 基礎設施即程式碼 (IaC) – 整個解決方案都使用 AWS CloudFormation 範本包裝，可更快速、更可靠的基礎設施佈建，同時減少手動設定錯誤。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 具有許可的 AWS Identity and Access Management (IAM) 角色，可建立和管理與工具中 AWS 服務列出的 相關聯的資源。
- Slack 工作區。
- 聊天應用程式中的 Amazon Q Developer 已新增至必要的 Slack 工作區做為外掛程式。如需詳細資訊，請參閱 Slack 文件中的將應用程式新增至 Slack 工作區。註冊成功 AWS 管理主控台 後，請記下 Slack 工作區 ID，如 所示。
- 聊天應用程式用戶端中設定的 Amazon Q Developer，工作區 ID 隨時可供 CloudFormation 主控台輸入。如需說明，請參閱聊天應用程式管理員指南中的設定 Amazon Q 開發人員中的 Slack 用戶端。
- 在 Amazon Simple Email Service (Amazon SES中建立和驗證的來源電子郵件帳戶，用於傳送核准電子郵件訊息。如需設定說明，請參閱《Amazon Simple Email Service 開發人員指南》中的建立和驗證電子郵件身分。
- 用於接收核准通知的目的地電子郵件地址。此地址可以是共用的收件匣或特定的團隊分發清單。
- 可從 存取的操作型 SonarQube 執行個體 AWS 帳戶。如需詳細資訊，請參閱 SonarQube 安裝說明。
- SonarQubeuser 權杖，具有透過管道觸發和建立專案的許可。

限制

- 建立自訂動作按鈕是此解決方案中的手動程序。
- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

下圖顯示此模式的工作流程和架構元件。

圖表顯示自動化程式碼品質保證工作流程：

1. 程式碼準備和上傳：

- 開發人員會將程式碼庫壓縮為 .zip 檔案。
- 開發人員手動將 .zip 檔案上傳至指定的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。

2. Amazon S3 事件觸發和 AWS Step Functions 協調：
   - Amazon S3 上傳事件會觸發 Step Functions 工作流程。
   - Step Functions 使用 SonarQube 協調 SAST 掃描。
   - 工作流程會監控 AWS CodeBuild 任務狀態，以判斷下一個動作。如果 CodeBuild 成功（品質閘道傳遞），工作流程會終止。如果 CodeBuild 失敗，則會叫用 AWS Lambda 函數進行診斷。如需詳細資訊，請參閱本節稍後的AWS Step Functions 邏輯。
3. AWS CodeBuild 執行：
   - CodeBuild 任務會在上傳的程式碼庫上執行 SonarQube 掃描。
   - 掃描成品存放在單獨的 Amazon S3 儲存貯體中，以進行稽核和分析。
4. 失敗分析 (Lambda 函數）：
   - 在 CodeBuild 失敗時，會觸發 `CheckBuildStatus` Lambda 函數。
   - 在 CodeBuild 成功時，程序會終止，而且不需要進一步的動作。
5. Lambda 函數會分析故障原因（品質閘道故障或其他問題）
   - `CheckBuildStatus` 函數會建立自訂承載，其中包含詳細的失敗資訊。
   - `CheckBuildStatus` 函數會將自訂承載發佈至 Amazon Simple Notification Service (Amazon SNS) 主題。
6. 通知系統：
   - Amazon SNS 會將承載轉送至聊天應用程式中的 Amazon Q Developer，以進行 Slack 整合。
7. Slack 整合：
   - 聊天應用程式中的 Amazon Q Developer 會在指定的 Slack 頻道中張貼通知。
8. 核准程序：
   - 核准者會檢閱 Slack 通知中的失敗詳細資訊。
   - 核准者可以使用 Slack 中的核准按鈕啟動核准。
9. 核准處理常式：
   - 核准 Lambda 函數會從 Slack 處理核准動作。
   - 核准函數會將自訂訊息發佈至 Amazon SES。
10. 產生的訊息：
   - 核准函數會產生開發人員通知的自訂訊息。

11開發人員通知：

- Amazon SES 會傳送電子郵件訊息給開發人員，其中包含後續步驟或必要動作。

此工作流程結合了手動程式碼上傳與自動品質檢查，透過 Slack 提供立即意見回饋，並在必要時允許人工介入，確保強大且靈活的程式碼檢閱程序。

AWS Step Functions 邏輯

如先前的架構圖所示，如果 SonarQube 上的品質閘道傳遞失敗，工作流程會前往 `CheckBuildStatus` Lambda 函數。`CheckBuildStatus` 函數會在 Slack 頻道上觸發通知。每個通知都包含建議後續步驟的資訊。以下是通知的類型：

- 應用程式在程式碼安全性掃描中失敗 – 當上傳的程式碼未通過 SonarQube 安全性掃描時，使用者會收到此通知。使用者可以選擇核准以接受建置。不過，通知會建議使用者注意潛在的不良程式碼品質和安全性風險。通知包含下列詳細資訊：
  - 後續步驟：錯誤：品質閘道狀態：失敗 – 在提供的 URL 中檢視詳細資訊。
  - 在提供的 URL 中對文件中提到的漏洞進行分類。
  - CodeBuild 詳細資訊可在所提供 URL 的位置取得。
- 應用程式掃描管道因其他原因失敗 – 當管道因程式碼安全性掃描失敗以外的某些原因失敗時，使用者會收到此通知。通知包含下列詳細資訊：
  - 如需後續步驟，請前往提供的連結進行進一步故障診斷。

若要查看通知出現在 Slack 頻道中的螢幕擷取畫面，請前往 GitHub chatops-slack 儲存庫中的資產資料夾。

下圖顯示品質閘道通過失敗後 Step Functions 步驟狀態的範例。

工具

AWS 服務

- 聊天應用程式中的 Amazon Q Developer 可讓您使用 Amazon Chime、Microsoft Teams 和 Slack 聊天頻道來監控和回應 AWS 應用程式中的操作事件。終止支援通知：在 2026 年 2 月 20 日， AWS 將終止對 Amazon Chime 服務的支援。2026 年 2 月 20 日之後，您將無法再存取 Amazon Chime 主控台或 Amazon Chime 應用程式資源。如需詳細資訊，請造訪部落格文章。這不會影響 Amazon Chime SDK 服務的可用性。

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。
- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- AWS Secrets Manager 可協助您將程式碼中的硬式編碼憑證 (包括密碼) 取代為 Secrets Manager 的 API 呼叫，以便透過程式設計方法來擷取機密。
- Amazon Simple Email Service (Amazon SES) 可協助您使用自己的電子郵件地址和網域來傳送和接收電子郵件訊息。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務 來建置業務關鍵型應用程式。

其他工具

- Slack 是 Salesforce 產品，是一種採用 AI 技術的對話平台，可提供聊天和視訊協作、自動化沒有程式碼的程序，並支援資訊共用。
- SonarQube 是一種內部部署分析工具，旨在偵測超過 30 種語言、架構和 IaC 平台的編碼問題。

程式碼儲存庫

此模式的程式碼可在 GitHub chatops-slack 儲存庫中使用。

最佳實務

- CloudFormation 堆疊管理 – 如果您在 CloudFormation 堆疊執行期間遇到任何失敗，建議您刪除失敗的堆疊。然後，使用正確的參數值重新建立它。此方法支援乾淨的部署，並有助於避免潛在的衝突或部分實作。
- 共用收件匣電子郵件組態 – 當您設定 `SharedInboxEmail` 參數時，請使用可供所有相關開發人員存取的通用分發清單。此方法可提高透明度，並協助重要通知聯絡相關團隊成員。
- 生產核准工作流程 – 對於生產環境，限制對用於建置核准的 Slack 頻道的存取。只有指定的核准者才能成為此頻道的成員。此實務維持明確的責任鏈，並透過限制誰可以核准關鍵變更來增強安全性。
- IAM 許可 – 遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。

史詩

執行初始設定

任務	Description	所需的技能
複製儲存庫。	若要複製此模式的 chatops-slack 儲存庫，請使用下列命令。 `git clone "git@github.com:aws-samples/chatops-slack.git"`	AWS DevOps、建置主管、DevOps 工程師、雲端管理員

建立包含 Lambda 程式碼的 .zip 檔案。	為 `CheckBuildStatus` 和 AWS Lambda 功能的函數程式碼建立 .zip 檔案`ApprovalEmail`。若要建立 `notification.zip` 和 `approval.zip`，請使用下列命令。
	
	`cd chatops-slack/src`
	
	`chmod -R 775 *`
		AWS DevOps、建置主管、DevOps 工程師、雲端管理員

任務	Description	所需的技能

	`zip -r approval.zip approval`
	
	`zip -r notification.zip notification`	

部署 pre-requisite.yml 堆疊檔案

任務	Description	所需的技能
執行`pre-requisite.yml` 堆疊檔案。	`pre-requisite.yml` CloudFormation 堆疊檔案會部署執行`app-security.yml` 堆疊檔案之前所需的初始資源。若要執行`pre-requisite.yml` 檔案，請執行下列動作： 1. 登入 AWS 管理主控台，然後開啟 CloudFormation主控台。選擇建立堆疊，然後從下拉式清單中選擇使用新資源（標準）。 2. 在建立堆疊頁面上，選擇選擇現有範本並上傳範本檔案。然後，選擇選擇檔案並選取 pre-requisite.yml。選擇下一步。 3. 在指定堆疊詳細資訊頁面上，輸入參數的值，如其他資訊中所述。然後選擇下一步。	AWS 管理員、AWS DevOps、建置主管、DevOps 工程師

任務	Description	所需的技能
	4. 在設定堆疊選項頁面上，選擇用於建立資源的 IAM 角色，如先決條件中所述。然後選擇下一步。 5. 在檢閱和建立頁面上，選擇提交。 6. 在堆疊的詳細資訊頁面上，選擇資源和輸出索引標籤。請記下在下列步驟中使用的 `S3Lambda`、`CKMSKeyArn` 和 `CKMSKeyId` 參數值。	
將 .zip 檔案上傳至 Amazon S3 儲存貯體。	將您先前建立的 `notification.zip` 和 `approval.zip` 檔案上傳至名為 `S3LambdaBucket` 的 Amazon S3 儲存貯體。`app-security.yml` CloudFormation 堆疊檔案使用 `S3LambdaBucket` 來佈建 Lambda 函數。	AWS DevOps、建置主管、DevOps 工程師、AWS 系統管理員

執行 app-security.yml 堆疊檔案

任務	Description	所需的技能
執行`app-security.yml` 堆疊檔案。	`app-security.yml` 堆疊檔案會部署通知和核准系統的剩餘基礎設施。若要執行 `app-security.yml` 檔案，請執行下列動作：	AWS DevOps、AWS 系統管理員、DevOps 工程師、建置主管

任務	Description	所需的技能
	1. 登入 AWS 管理主控台，然後開啟 CloudFormation主控台。選擇建立堆疊，然後從下拉式清單中選擇使用新資源（標準）。 2. 在建立堆疊頁面上，選擇選擇現有範本並上傳範本檔案。然後，選擇選擇檔案，然後選取 app-security.yml。選擇下一步。 3. 在指定堆疊詳細資訊頁面上，輸入參數的值，如其他資訊中所述。然後選擇下一步。 4. 在設定堆疊選項頁面上，選擇用於建立資源的 IAM 角色，如先決條件中所述。然後選擇下一步。 5. 在檢閱和建立頁面上，選擇提交。	

任務	Description	所需的技能
測試通知設定。	若要測試通知設定，請執行下列動作： 1. 開啟 Amazon SNS 主控台。在左側導覽窗格中，選擇主題。 2. 選取以 LambdaToAWSSlackChatbot 結尾的主題名稱。 3. 在主題的詳細資訊頁面上，選擇發佈訊息。 4. 在發佈訊息至主題頁面上，針對要傳送至端點的訊息內文，輸入下列內容： `{ "version": "1.0", "source": "custom", "content": { "description": ":warning: This is a test notification" } }` 5. 選擇 Publish message (發佈訊息)。 成功傳遞測試訊息後，您應該會在 Slack 頻道上看到通知。如需詳細資訊，請參閱《聊天應用程式管理員指南》中的從 AWS 服務 到 Slack 的測試通知。	AWS DevOps、AWS 系統管理員、DevOps 工程師、建置主管

設定核准流程

任務	Description	所需的技能
設定自訂 Lambda 動作。	若要設定自訂 AWS Lambda 動作，請執行下列動作： 1. 選擇 Slack 頻道中交付通知底部的垂直省略號按鈕。 2. 在管理動作中，選擇建立。 3. 輸入自訂動作名稱，例如核准。此名稱是自訂動作的唯一識別符。 4. 輸入自訂動作按鈕的名稱，例如核准。此名稱會顯示在通知的按鈕上。此名稱應為 20 個字元或更少，並且可以包含表情符號。 5. 針對自訂動作類型，選取 Lambda 動作。 6. 選擇下一步。 7. 選取您要部署此動作 AWS 區域 的 AWS 帳戶 和 。 8. 選擇載入 Lambdas。 9. 在定義 Lambda 函數中，選取以 ApprovalEmailLambda 結尾的 Lambda 函數。 然後選擇下一步。 10.若要建立核准按鈕，請在顯示條件頁面上選擇儲存。	AWS 管理員、AWS DevOps、建置主管、DevOps 工程師、Slack Admin
驗證核准流程。	若要驗證核准流程是否如預期運作，請選擇 Slack 中的核准按鈕。	AWS 管理員、AWS DevOps、DevOps 工程師、Slack Admin

任務	Description	所需的技能
	Slackbot 應在訊息執行緒上傳送通知，並成功傳送確認字串核准電子郵件。	

疑難排解

問題	解決方案
Slack 設定錯誤	如需 Slack 設定錯誤相關問題的疑難排解資訊，請參閱聊天應用程式管理員指南中的 Amazon Q 開發人員疑難排解 Amazon Q 開發人員。
由於其他原因，掃描失敗	此錯誤表示程式碼建置任務失敗。若要對問題進行疑難排解，請前往訊息中的連結。程式碼建置任務的失敗可能原因如下： • 應用程式未正確封裝。`sonar-scanner` 命令找不到 `sonar.project.env.properties` 檔案。 • `SonarFileName` 、 `SonarFileDirectory` 或 `SonarToken` 參數的值不正確。檢查值，然後再次執行堆疊檔案。 • 無法連線 Sonar 主機。 • 您可以使用日誌進行故障診斷的其他問題。

相關資源

AWS 文件

- 設定 Slack 用戶端
- 建立自訂動作
- 建立電子郵件地址身分程序
- 教學課程：開始使用 Slack

其他資源

- 將應用程式新增至 Slack 工作區 (Slack 文件)
- 產生和使用字符 (SonarQube 文件)
- 伺服器安裝簡介 (SonarQube 文件)

其他資訊

此解決方案強調聊天應用程式中的 Amazon Q Developer 用於版本管理的自訂動作。不過，您可以修改特定使用案例的 Lambda 程式碼，並在其上建置，以重複使用解決方案。

CloudFormation 堆疊檔案的參數

下表顯示 CloudFormation 堆疊檔案 的參數及其描述`pre-requisite.yml`。

索引鍵	Description
`StackName`	CloudFormation 堆疊的名稱。
`S3LambdaBucket`	您上傳 Lambda 程式碼的 Amazon S3 儲存貯體名稱。名稱必須是全域唯一的。
`SonarToken`	SonarQube 使用者字符，如先決條件中所述。

下表顯示 CloudFormation 堆疊檔案 的參數及其描述`app-security.yml`。

索引鍵	Description
`CKMSKeyArn`	在此堆疊中建立的 AWS KMS key IAM 角色和 Lambda 函數中使用的 Amazon Resource Name (ARN)。
`CKMSKeyId`	在此堆疊中建立的 Amazon SNS 主題中使用的 AWS KMS key ID。
`EnvironmentType`	用於部署應用程式掃描管道的用戶端環境名稱。從允許的值下拉式清單中選取環境名稱。

S3LambdaBucket	包含 approval.zip 和 notification.zip 檔案的 Amazon S3 儲存貯體名稱。
SESEmail	Amazon SES 中已註冊電子郵件身分的名稱，如先決條件中所述。此身分是來源電子郵件地址。
SharedInboxMail	傳送掃描通知的目標電子郵件地址。
SlackChannelId	您要傳送通知之 Slack 頻道的頻道 ID。若要尋找頻道 ID，請在 Slack 應用程式的頻道詳細資訊中以滑鼠右鍵按一下頻道名稱。頻道 ID 位於底部。
SlackWorkspaceId	Slack 工作區 ID，如先決條件中所述。若要尋找 Slack 工作區 ID，請登入 AWS 管理主控台，在聊天應用程式主控台中開啟 Amazon Q Developer，然後選擇已設定的用戶端、Slack、WorkspaceID。
StackName	CloudFormation 堆疊的名稱。
SonarFileDirectory	包含 sonar.project.<env>.properties 檔案的目錄。
SonarFileName	sonar.project.<env>properties 檔案名稱。
SourceCodeZip	包含 檔案和原始程式碼的 .zip sonar.project.<env>properties 檔案名稱。

使用 Terraform 在 Amazon Bedrock 上部署代理程式系統，搭配CrewAI 架構

Vanitha Dontireddy，Amazon Web Services

摘要

此模式示範如何使用與 Amazon Bedrock 和 Terraform 整合的CrewAI 架構來實作可擴展的多代理程式 AI 系統。解決方案可讓組織透過基礎設施即程式碼 (IaC) 來建立、部署和管理複雜的 AI 代理程式工作流程。在此模式中，CrewAI 多代理程式協同運作功能會結合 Amazon Bedrock 基礎模型和 Terraform 基礎設施自動化。因此，團隊可以建置生產就緒的 AI 系統，以最少的人工監督來處理複雜的任務。模式實作企業級安全性、可擴展性和操作最佳實務。

先決條件和限制

先決條件

- AWS 帳戶 具備適當許可的作用中 ，可存取 Amazon Bedrock 基礎模型
- 已安裝 Terraform 1.5 版或更新版本
- 已安裝 Python 3.9 版或更新版本
- 已安裝CrewAI 架構

限制

- 客服人員互動受限於模型內容視窗。
- 大規模部署的分層狀態管理考量適用於此模式。
- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

在此模式中，會發生下列互動：

- Amazon Bedrock 透過其基礎模型 (FMs) 套件提供代理程式智慧的基礎。它可為 AI 代理器啟用自然語言處理 (NLP)、推理和決策功能，同時保持高可用性和可擴展性。
- CrewAI 架構可做為核心協同運作層，用於建立和管理 AI 代理器。它在與 Amazon Bedrock 整合時處理代理程式通訊協定、任務委派和工作流程管理。

- Terraform 透過程式碼管理整個基礎設施堆疊，包括運算資源、聯網、安全群組和 AWS Identity and Access Management (IAM) 角色。它可確保跨環境的一致、版本控制的部署。Terraform 部署會建立下列項目：
  - AWS Lambda 函數以執行CrewAI 應用程式
  - 適用於程式碼和報告的 Amazon Simple Storage Service (Amazon S3) 儲存貯體
  - 具有適當許可的 IAM 角色
  - Amazon CloudWatch 記錄
  - Amazon EventBridge 排定的執行

下圖說明使用 Amazon Bedrock 和 Terraform 來部署 CrewAI 多代理程式系統的架構。

該圖顯示以下工作流程：

1. 使用者複製儲存庫。
2. 使用者執行 命令`terraform apply`來部署 AWS 資源。
3. Amazon Bedrock 模型組態包括指定基礎模型 (FM)，以用於設定CrewAI 代理程式。
4. 建立 EventBridge 規則，根據定義的排程觸發 Lambda 函數。
5. 觸發時（透過排程或手動），Lambda 函數會初始化並擔任具有存取 AWS 服務 和 Amazon Bedrock 許可的 IAM 角色。
6. CrewAI 架構會從 YAML 檔案載入代理程式組態，並建立專門的 AI 代理程式 (AWS 基礎設施安全稽核人員）。Lambda 函數會依序執行這些代理程式，以掃描 AWS 資源、分析安全漏洞，並產生全面的稽核報告。
7. CloudWatch Logs 會擷取 Lambda 函數的詳細執行資訊，其中包含 365 天的保留期和 AWS Key Management Service (AWS KMS) 合規要求的加密。日誌提供客服人員活動、錯誤追蹤和效能指標的可見性，可有效監控和疑難排解安全稽核程序。
8. 安全性稽核報告會自動產生並存放在指定的 Amazon S3 儲存貯體中。自動化設定有助於維持一致的安全監控，並將營運開銷降至最低。

在初始部署之後，工作流程會為您的 AWS 基礎設施提供持續的安全稽核和報告，而無需手動介入。

AI 代理器概觀

此模式會建立多個 AI 代理器，每個代理器都有獨特的角色、目標和工具：

- 安全分析師代理程式會收集和分析 AWS 資源資訊。
- 滲透測試人員代理程式可識別 AWS 資源中的漏洞。
- 合規專家代理程式會根據合規標準檢查組態。
- 報告寫入器代理程式會將問題清單編譯為全面的報告。

這些客服人員會協同處理一系列任務，利用其集體技能來執行安全稽核並產生全面的報告。(`config/agents.yaml` 檔案概述了此組建中每個代理程式的功能和組態。)

安全分析處理包含下列動作：

1. 安全分析師代理程式會檢查所收集的 AWS 資源相關資料，例如下列項目：
   - Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和安全群組
   - Amazon S3 儲存貯體和組態
   - IAM 角色、政策和許可
   - 虛擬私有雲端 (VPC) 組態和網路設定
   - Amazon RDS 資料庫和安全性設定
   - Lambda 函數和組態
   - AWS 服務 稽核範圍內的其他項目
2. 滲透測試人員代理程式可識別潛在漏洞。
3. 客服人員透過CrewAI」架構協同合作來共用問題清單。

報告產生包含下列動作：

1. 報告寫入器代理程式會編譯所有其他代理程式的問題清單。
2. 安全問題會依服務、嚴重性和合規影響進行組織。
3. 系統會為每個已識別的問題產生修復建議。
4. 以 Markdown 格式建立全面的安全稽核報告，並上傳至指定的 Amazon S3 儲存貯體。保留歷史報告以進行合規追蹤和安全狀態改善。

記錄和監控活動包括：

- CloudWatch 日誌會擷取執行詳細資訊和任何錯誤。
- 系統會記錄 Lambda 執行指標以進行監控。

> **Note**
>
> 的程式碼`aws-security-auditor-crew`來自 GitHub 3P-Agentic_frameworks 儲存庫，可在 AWS 範例集合中找到。

可用性和擴展

您可以將可用的客服人員擴展到四個以上的核心客服人員。若要使用其他專業客服人員進行擴展，請考慮下列新的客服人員類型：

- 威脅情報專家代理程式可以執行下列動作：
  - 監控外部威脅饋送並與內部調查結果建立關聯
  - 提供有關與基礎設施相關的新興威脅的內容
  - 根據萬用中的主動入侵來排定漏洞的優先順序
- 合規架構代理程式可以專注於特定法規領域，例如：
  - 支付卡產業資料安全標準 (PCI DSS) 合規代理程式
  - 1996 年健康保險流通與責任法案 (HIPAA) 合規代理程式
  - 系統和組織控制 2 (SOC 2) 合規代理程式
  - 一般資料保護法規 (GDPR) 合規代理程式

透過仔細擴展可用的代理程式，此解決方案可以提供更深入、更專業的安全洞見，同時在大型 AWS 環境中維持可擴展性。如需實作方法、工具開發和擴展考量的詳細資訊，請參閱其他資訊。

工具

AWS 服務

- Amazon Bedrock 是一種全受管 AI 服務，可透過統一 API 使用高效能的基礎模型 (FMs)。
- Amazon CloudWatch Logs 可協助您集中所有系統、應用程式的日誌， AWS 服務 以便您可以監控日誌並將其安全地存檔。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如， AWS Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他事件匯流排 AWS 帳戶。在此模式中，它會用於排程和協調客服人員工作流程。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。

- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- 適用於 Python (Boto3) 的 AWS SDK 是一種軟體開發套件，可協助您整合 Python 應用程式、程式庫或指令碼 AWS 服務。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。在此模式中，它會為代理程式成品和狀態管理提供物件儲存。

其他工具

- 以 Python 為基礎的開放原始碼架構CrewAI可建置多代理程式 AI 系統。
- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。

程式碼儲存庫

此模式的程式碼可在 GitHub deploy-crewai-agents-terraform 儲存庫中使用。

最佳實務

- 使用 Amazon S3 後端搭配 Amazon DynamoDB 鎖定，為 Terraform 實作適當的狀態管理。如需詳細資訊，請參閱《使用 Terraform 提供者的最佳實務》中的後端最佳實務。 AWS
- 使用工作區來分隔開發、預備和生產環境。
- 遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。
- 透過 CloudWatch Logs 啟用詳細記錄和監控。
- 實作客服人員操作的重試機制和錯誤處理。

史詩

部署CrewAI架構

任務	Description	所需的技能
複製儲存庫。	若要在本機電腦上複製此模式的儲存庫，請執行下列命令：	DevOps 工程師

任務	Description	所需的技能
	`git clone "git@github.com:aws-samples/deploy-crewai-agents-terraform.git"` `cd deploy-crewai-agents-terraform`	
編輯環境變數。	若要編輯環境變數，請執行下列動作： 1. 從 terraform `terraform.tfvars.example` 目錄中的範例建立`terraform.tfvars` 檔案。 2. 編輯環境變數以使用您自己的資訊。	DevOps 工程師

建立基礎設施。	若要建立基礎設施，請執行下列命令：
	`cd terraform`
	`terraform init`
	`terraform plan`
	請仔細檢閱執行計畫。如果計劃的變更是可接受的，請執行下列命令：
	`terraform apply --auto-approve`
		DevOps 工程師

存取 CrewAI 代理程式

任務	Description	所需的技能
存取代理程式。	AWS 基礎設施安全稽核和報告人員中的代理程式會部署為 Lambda 函數。若要存取代理程式，請使用下列步驟： 1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/lambda/ 的 AWS Lambda 主控台。 2. 在函數頁面上，尋找並選取名為 {project_name} \function 的函數（如 Terraform 變數中所定義）。 3. 從函數的頁面，您可以採取下列動作： • 檢視組態詳細資訊。 • 監控執行指標。 • 檢視 CloudWatch 日誌。 • 手動測試函數。	DevOps 工程師
（選用）設定代理程式的手動執行。	代理程式設定為以每日排程（午夜 UTC) 自動執行。不過，您可以使用下列步驟手動觸發它們： 1. 在 Lambda 主控台中，選取名為 的函數{project_name} 。	DevOps 工程師

任務	Description	所需的技能
	2. 在函數的頁面上，選擇測試索引標籤。 3. 使用空的 JSON 物件 建立新的測試事件{}。 4. 若要執行事件，請選擇測試。 如需詳細資訊，請參閱 Lambda 文件中的在主控台中測試 Lambda 函數。	
存取代理程式日誌以進行偵錯。	CrewAI 代理程式正在 Lambda 環境中執行，具有執行安全稽核和在 Amazon S3 中存放報告的必要許可。輸出是 Markdown 報告，可提供 AWS 基礎設施的完整安全分析。 若要協助詳細偵錯代理程式行為，請執行下列動作： 1. 在 中 AWS 管理主控台，導覽至 CloudWatch Logs。 2. 尋找 Lambda 函數的日誌群組。 3. 尋找具有代理程式名稱的日誌項目 （例如，基礎設施映射專家和探索性安全分析師)。 4. 檢閱日誌，深入了解每個代理程式的動作。	DevOps 工程師

任務	Description	所需的技能
檢視代理程式執行的結果。	若要檢視代理程式執行的結果，請執行下列動作： 1. 在 中 AWS 管理主控台，導覽至 Amazon S3。 2. 開啟名為 的 Amazon S3 儲存貯體 `{project_name}-reports-{random_suffix}` （如 Terraform 變數中所定義）。 報告會以時間戳記型檔案名稱存放，如下所示： `security-audit-report-YYYY-MM-DD-HH-MM-SS.md)`	DevOps 工程師
監控代理程式執行。	若要透過 CloudWatch 日誌監控代理程式的執行，請執行下列動作： 1. 在 中 AWS 管理主控台，導覽至 CloudWatch。 2. 移至日誌群組。 3. 選取名為 的日誌群組/ `aws/lambda/{project_name}-function` 。 4. 在日誌串流中，選擇最新的日誌串流，以查看詳細的執行資訊。	DevOps 工程師

任務	Description	所需的技能

自訂代理程式行為。	若要修改客服人員或其任務，請執行下列動作：
	1. 更新本機儲存庫中的組態檔案：
	• 下列檔案會定義每個客服人員的角色、功能和設定：`src/aws_infrastructure_security_audit_and_reporting/config/agents.yaml`
	• 下列 dile 定義客服人員的任務和工作流程：
	`src/aws_infrastructure_security_audit_and_reporting/config/tasks.yaml`
	2. 若要重新封裝和更新 Lambda 函數，請使用下列命令：
	`cd terraform`
	`terraform apply`
		DevOps 工程師

清除資源

任務	Description	所需的技能
刪除建立的資源。	若要刪除此模式建立的所有基礎設施，請執行下列命令：	DevOps 工程師

任務	Description	所需的技能

	`terraform plan -destroy`
	
	⚠ Warning
	下列命令將永久刪除此模式建立的所有資源。命令會在移除任何資源之前提示確認。
	
	請仔細檢閱銷毀計畫。如果計劃刪除是可接受的，請執行下列命令：
	
	`terraform destroy`	

疑難排解

問題	解決方案
代理程式行為	如需此問題的相關資訊，請參閱 Amazon Bedrock 文件中的測試和疑難排解代理程式行為。
Lambda 網路問題	如需有關這些問題的資訊，請參閱 Lambda 文件中的對 Lambda 中的聯網問題進行故障診斷。
IAM 許可	如需這些問題的相關資訊，請參閱 IAM 文件中的疑難排解 IAM。

相關資源

AWS 部落格

- 使用CrewAI和 Amazon Bedrock 建置代理系統

AWS 文件

- Amazon Bedrock 文件
- Amazon Bedrock 代理程式的運作方式
- AWS Well-Architected 架構

其他資源

- CrewAI 文件
- Terraform AWS 提供者文件

其他資訊

本節包含實作方法、工具開發以及與自動化和擴展中先前討論相關的擴展考量的相關資訊。

實作方法

考慮以下新增客服人員的方法：

1. 代理程式組態：
   - 將新的客服人員定義新增至 `config/agents.yaml` 檔案。
   - 為每個代理程式定義專門的備份、目標和工具。
   - 根據客服人員專業設定記憶體和分析功能。
2. 任務協同運作：
   - 更新 `config/tasks.yaml` 檔案以包含新的客服人員特定任務。
   - 在任務之間建立相依性，以協助確保適當的資訊流程。
   - 適時實作平行任務執行。

技術實作

以下是提議的 Threat Intelligence Specialist 代理程式`agents.yaml`的 檔案新增項目：

```
Example new agent configuration in agents.yaml
```

```
threat_intelligence_agent:
 name: "Threat Intelligence Specialist"
 role: "Cybersecurity Threat Intelligence Analyst"
 goal: "Correlate AWS security findings with external threat intelligence"
 backstory: "Expert in threat intelligence with experience in identifying emerging
 threats and attack patterns relevant to cloud infrastructure."
verbose: true
allow_delegation: true
tools:
- "ThreatIntelligenceTool"
- "AWSResourceAnalyzer"
```

工具開發

使用CrewAI 架構，您可以採取下列動作來增強安全稽核團隊的有效性：

- 為新客服人員建立自訂工具。
- 與外部 APIs整合以進行威脅情報。
- 為不同的 開發專用分析器 AWS 服務。

擴展考量

擴展您的 AWS 基礎設施安全稽核和報告系統以處理更大的環境或更全面的稽核時，請解決下列擴展因素：

- 運算資源
  - 增加 Lambda 記憶體配置以處理其他代理程式。
  - 考慮將代理程式工作負載分割到多個 Lambda 函數。
- 成本管理
  - 在代理程式計數增加時監控 Amazon Bedrock API 用量。
  - 根據稽核範圍實作選擇性代理程式啟用。
- 協作效率
  - 最佳化客服人員之間的資訊共用。
  - 實作複雜環境的階層式代理程式結構。
- 知識庫增強功能
  - 為客服人員提供其網域的專業知識庫。
  - 使用新的安全最佳實務定期更新客服人員知識。

使用 AWS CodePipeline CI/CD 管道部署 AWS Glue 任務 AWS CodePipeline

Bruno Klein 和 Luis Henrique Massao Yamada，Amazon Web Services

總結

此模式示範如何將 AWS CodeCommit 和 AWS CodePipeline 與 AWS Glue 整合，並在開發人員將其變更推送至遠端 AWS CodeCommit 儲存庫時，立即使用 AWS Lambda 啟動任務。

當開發人員提交擷取、轉換和載入 (ETL) 儲存庫的變更，並將變更推送到 AWS CodeCommit 時，會叫用新的管道。管道會啟動 Lambda 函數，透過這些變更啟動 AWS Glue 任務。AWS Glue 任務會執行 ETL 任務。

此解決方案有助於企業、開發人員和資料工程師在遞交變更並推送至目標儲存庫後立即啟動任務的情況。它有助於實現更高層級的自動化和重現性，因此避免任務啟動和生命週期期間發生錯誤。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 安裝在本機電腦上的 Git
- 安裝在本機電腦上的 Amazon Cloud Development Kit (Amazon CDK)
- 安裝在本機電腦上的 Python
- 附件區段中的程式碼

限制

- 一旦 AWS Glue 任務成功啟動，管道就會完成。它不會等待任務完成。
- 附件中提供的程式碼僅供示範使用。

Architecture

目標技術堆疊

- AWS Glue

- AWS Lambda
- AWS CodePipeline
- AWS CodeCommit

目標架構

此程序包含下列步驟：

1. 開發人員或資料工程師會修改 ETL 程式碼、遞交變更，並將變更推送至 AWS CodeCommit。
2. 推送會啟動管道。
3. 管道會啟動 Lambda 函數，在儲存庫`codecommit:GetFile`上呼叫 ，並將檔案上傳至 Amazon Simple Storage Service (Amazon S3)。
4. Lambda 函數會使用 ETL 程式碼啟動新的 AWS Glue 任務。
5. Lambda 函數會完成管道。

自動化和擴展

範例附件示範如何將 AWS Glue 與 AWS CodePipeline 整合。它提供一個基準範例，您可以自訂或擴展供自己使用。如需詳細資訊，請參閱 Epics 區段。

工具

- AWS CodePipeline – AWS CodePipeline 是一項全受管持續交付服務，可協助您自動化發行管道，以實現快速可靠的應用程式和基礎設施更新。
- AWS CodeCommit – AWS CodeCommit 是一種全受管的來源控制服務，可託管安全的 Git 型儲存庫。
- AWS Lambda – AWS Lambda 是一種無伺服器運算服務，可讓您執行程式碼，而無需佈建或管理伺服器。
- AWS Glue – AWS Glue 是一種無伺服器資料整合服務，可讓您輕鬆探索、準備和結合資料，以進行分析、機器學習和應用程式開發。
- Git 用戶端 – Git 提供 GUI 工具，或者您可以使用命令列或桌面工具從 GitHub 檢查所需的成品。
- AWS CDK – AWS CDK 是一種開放原始碼軟體開發架構，可協助您使用熟悉的程式設計語言定義雲端應用程式資源。

史詩

部署範例程式碼

任務	Description	所需的技能
設定 AWS CLI。	將 AWS Command Line Interface (AWS CLI) 設定為目標，並使用您目前的 AWS 帳戶進行驗證。如需說明，請參閱 AWS CLI 文件。	開發人員、DevOps 工程師
解壓縮範例專案檔案。	從附件解壓縮檔案，以建立包含範例專案檔案的資料夾。	開發人員、DevOps 工程師

部署範例程式碼。	解壓縮檔案之後，請從解壓縮位置執行下列命令，以建立基準範例：
	`cdk bootstrap`
	`cdk deploy`
	`git init`
	`git remote add origin <code-commit-repository-url>`
	`git stage .`
	`git commit -m "adds sample code"`
	`git push --set-upstream origin main`
	在最後一個命令之後，您可以監控管道和 AWS Glue 任務的狀態。
		開發人員、DevOps 工程師

自訂程式碼。	根據您的業務需求自訂 etl.py 檔案的程式碼。您可以修改 ETL 程式碼、修改管道階段或擴展解決方案。	資料工程師

相關資源

- AWS CDK 入門
- 在 AWS Glue 中新增任務
- CodePipeline 中的來源動作整合
- 在 CodePipeline 的管道中調用 AWS Lambda 函數
- AWS Glue 程式設計
- AWS CodeCommit GetFile API

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS CodePipeline、AWS CodeCommit 和 AWS CodeBuild 在多個 AWS 區域中部署程式碼

Anand Krishna Varanasi，Amazon Web Services

摘要

此模式示範如何使用 AWS CloudFormation 跨多個 Amazon Web Services (AWS) 區域建置基礎設施或架構。它包含跨多個 AWS 區域的持續整合 (CI)/持續部署 (CD)，以加快部署速度。 此模式中的步驟已針對建立 AWS CodePipeline 任務進行測試，以部署至三個 AWS 區域做為範例。您可以根據您的使用案例變更區域數量。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- • 具有 AmazonS3FullAccess 和 CloudWatchFullAccess 政策的 CodeBuild 角色。這些政策可讓 CodeBuild 透過 Amazon CloudWatch 監看 AWS CodeCommit 事件，並使用 Amazon Simple Storage Service (Amazon S3) 做為成品存放區。
  - 具有下列政策的 AWS CloudFormation 角色，可讓 AWS CloudFormation 在最終建置階段中建立或更新 AWS Lambda 函數、推送或監看 Amazon CloudWatch logs，以及建立和更新變更集。
    - AWSLambdaFullAccess
    - AWSCodeDeployFullAccess
    - CloudWatchFullAccess
    - AWSCloudFormationFullAccess
    - AWSCodePipelineFullAccess

> **Note**
>
> AWS CodeBuild 和 AWS CloudFormation 的兩個 AWS Identity and Access Management (IAM) 角色具有適當的政策，可讓 CodeBuild 執行 CI 任務，以平行測試、綁定、封裝成品和部署到多個 AWS 區域。 交叉檢查 CodePipeline 建立的政策，以確認 CodeBuild 和 AWS CloudFormation 在 CI 和 CD 階段具有適當的許可。

Architecture

此模式的多區域架構和工作流程包含下列步驟。

1. 您可以將程式碼傳送至 CodeCommit 儲存庫。
2. 收到任何程式碼更新或遞交時，CodeCommit 會叫用 CloudWatch 事件，進而啟動 CodePipeline 任務。
3. CodePipeline 會參與 CodeBuild 處理的 CI。會執行下列任務。
   - 測試 AWS CloudFormation 範本（選用）
   - 部署中包含的每個區域的 AWS CloudFormation 範本封裝。例如，此模式會平行部署至三個 AWS 區域，因此 CodeBuild 會將 AWS CloudFormation 範本封裝成三個 S3 儲存貯體，每個指定區域中各一個。CodeBuild 只會使用 S3 儲存貯體做為成品儲存庫。
4. CodeBuild 會將成品封裝為下一個部署階段的輸入，並在三個 AWS 區域中平行執行。如果您指定不同數量的區域，CodePipeline 會部署到這些區域。

工具

工具

- AWS CodePipeline – CodePipeline 是一種持續交付服務，可用來建立模型、視覺化和自動化持續發佈軟體變更所需的步驟。
- AWS CodeBuild – CodeBuild 是一種全受管的建置服務，可編譯您的原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeCommit – CodeCommit 是由 Amazon Web Services 託管的版本控制服務，可用來在雲端中私下存放和管理資產（例如原始碼和二進位檔案）。
- AWS CloudFormation – AWS CloudFormation 是一項服務，可協助您建立和設定 Amazon Web Services 資源的模型，以減少管理這些資源的時間，並有更多時間專注於在 AWS 中執行的應用程式。
- AWS Identity and Access Management – AWS Identity and Access Management (IAM) 是一種 Web 服務，可協助您安全地控制對 AWS 資源的存取。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是網際網路的儲存體。此服務旨在降低開發人員進行網路規模運算的難度。

Code

下列範例程式碼適用於 BuildSpec.yaml 檔案 （建置階段）。

```
---
artifacts:
discard-paths: true
files:
- packaged-first-region.yaml
- packaged-second-region.yaml
- packaged-third-region.yaml
phases:
build:
commands:
- echo "********BUILD PHASE - CF PACKAGING**********"
- "aws cloudformation package --template-file sam-template.yaml --s3-bucket
 $S3_FIRST_REGION --output-template-file packaged-first-region.yaml --region
 $FIRST_REGION"
- "aws cloudformation package --template-file sam-template.yaml --s3-bucket
 $S3_SECOND_REGION --output-template-file packaged-second-region.yaml --region
 $SECOND_REGION"
- "aws cloudformation package --template-file sam-template-anand.yaml --s3-bucket
 $S3_THIRD_REGION --output-template-file packaged-third-region.yaml --region
 $THIRD_REGION"
install:
commands:
- echo "********BUILD PHASE - PYTHON SETUP**********"
runtime-versions:
python: 3.8
post_build:
commands:
- echo "********BUILD PHASE - PACKAGING COMPLETION**********"
pre_build:
commands:
- echo "********BUILD PHASE - DEPENDENCY SETUP**********"
- "npm install --silent --no-progress"
- echo "********BUILD PHASE - DEPENDENCY SETUP DONE**********"
version: 0.2
```

史詩

準備程式碼和 CodeCommit 儲存庫

任務	Description	所需的技能
選取部署的主要 AWS 區域。	登入您的 AWS 帳戶，然後選擇部署的主要區域。CodeCommit 儲存庫將位於主要區域。	DevOps
建立 CodeCommit 儲存庫。	建立 CodeCommit 儲存庫，並將所需的程式碼推送至其中。此程式碼通常包含 AWS CloudFormation 或 AWS SAM 範本、如果有 Lambda 程式碼，以及 CodeBuild `buildspec.yaml` 檔案做為 AWS CodePipeline 的輸入。	DevOps
將程式碼推送至 CodeCommit 儲存庫。	在附件區段中，下載此範例的程式碼，然後將所需的程式碼推送至其中。一般而言，程式碼可以包含 AWS CloudFormation 或 AWS SAM 範本、Lambda 程式碼和 CodeBuild `buildspec.yaml` 檔案做為管道的輸入。	DevOps

來源階段：建立管道

任務	Description	所需的技能
建立 CodePipeline 任務。	在 CodePipeline 主控台上，選擇建立管道。	DevOps

任務	Description	所需的技能
命名 CodePipeline 任務，然後選擇服務角色設定。	輸入任務的名稱，並保留預設服務角色設定，以便 CodePipeline 建立已連接必要政策的角色。	DevOps
指定成品存放區的位置。	在進階設定下，保留預設選項，以便 CodePipeline 建立用於程式碼成品儲存的 S3 儲存貯體。如果您改用現有的 S3 儲存貯體，儲存貯體必須位於您在第一個特徵中指定的主要區域中。	DevOps
指定加密金鑰。	保留預設選項、預設 AWS 受管金鑰，或選擇使用您自己的 AWS Key Management Service (AWS KMS) 客戶受管金鑰。	DevOps
指定來源提供者。	在來源提供者下，選擇 AWS CodeCommit。	DevOps
指定儲存庫。	選擇您在第一個 epic 中建立的 CodeCommit 儲存庫。如果您將程式碼放在分支中，請選擇分支。	DevOps
指定程式碼變更的偵測方式。	保留預設值 Amazon CloudWatch Events，做為 CodeCommit 啟動 CodePipeline 任務的變更觸發條件。	DevOps

建置階段：設定管道

任務	Description	所需的技能
指定建置提供者。	針對建置提供者，選擇 AWS CodeBuild。	DevOps
指定 AWS 區域。	選擇您在第一個史詩中指定的主要區域。	DevOps

組建階段：建立和設定專案

任務	Description	所需的技能
建立專案	選擇建立專案，然後輸入專案的名稱。	DevOps
指定環境映像。	針對此模式示範，請使用預設 CodeBuild 受管映像。如果您有自訂 Docker 映像，您也可以選擇使用自訂 Docker 映像。	DevOps
指定作業系統。	選擇 Amazon Linux 2 或 Ubuntu。 **Note** Amazon Linux 2 即將終止支援。如需詳細資訊，請參閱 Amazon Linux 2 FAQs。	DevOps
指定服務角色。	在開始建立 CodePipeline 任務之前，請選擇您為 CodeBuild 建立的角色。 CodePipeline（請參閱先決條件一節。）	DevOps

任務	Description	所需的技能
設定其他選項。	對於逾時和佇列逾時，請保留預設值。對於憑證，除非您有要使用的自訂憑證，否則請保留預設設定。	DevOps
建立環境變數。	針對您要部署的每個 AWS 區域，提供 S3 儲存貯體名稱和區域名稱（例如 us-east-1) 來建立環境變數。	DevOps
如果不是 buildspec.yml，請提供 buildspec 檔案名稱。	如果檔案名稱為預設值，請將此欄位保留空白`buildspec.yaml`。如果您重新命名 buildspec 檔案，請在此處輸入名稱。請確定它符合 CodeCommit 儲存庫中的檔案名稱。	DevOps
指定記錄。	若要查看 Amazon CloudWatch Events 的日誌，請保留預設設定。或者，您可以定義任何特定的群組或記錄器名稱。	DevOps

略過部署階段

任務	Description	所需的技能
略過部署階段並完成管道的建立。	當您設定管道時，CodePipeline 只允許您在部署階段建立一個階段。若要部署到多個 AWS 區域，請略過此階段。建立管道之後，您可以新增多個部署階段。	DevOps

部署階段：設定管道以部署到第一個區域

任務	Description	所需的技能
將階段新增至部署階段。	編輯管道，然後在部署階段中選擇新增階段。第一個階段適用於主要區域。	DevOps
提供階段的動作名稱。	輸入反映第一個（主要）階段和區域的唯一名稱。例如，輸入 primary_<region>_deploy。	DevOps
指定動作提供者。	針對動作提供者，選擇 AWS CloudFormation。	DevOps
設定第一個階段的區域。	選擇第一個（主要）區域，即設定 CodePipeline 和 CodeBuild 的相同區域。這是您要部署堆疊的主要區域。	DevOps
指定輸入成品。	選擇 BuildArtifact。這是建置階段的輸出。	DevOps
指定要採取的動作。	針對動作模式，選擇建立或更新堆疊。	DevOps
輸入 CloudFormation 堆疊的名稱。		DevOps
指定第一個區域的範本。	選取 CodeBuild 封裝並傾印至第一個（主要）區域的 S3 儲存貯體的區域特定套件名稱。	DevOps
指定 功能。	如果堆疊範本包含 IAM 資源，或者您直接從包含巨集的範本建立堆疊，則需要 功能。對於此模式，請使用 CAPABILITY_IAM、CAPABILITY_N	DevOps

任務	Description	所需的技能
	AMED_IAM、CAPABILITY_AUTO_EXPAND。	

部署階段：設定管道以部署到第二個區域

任務	Description	所需的技能
將第二個階段新增至部署階段。	若要為第二個區域新增階段，請編輯管道，然後在部署階段中選擇新增階段。重要：建立第二個區域的程序與第一個區域的程序相同，但下列值除外。	DevOps
提供第二個階段的動作名稱。	輸入反映第二個階段和第二個區域的唯一名稱。	DevOps
設定第二個階段的區域。	選擇您要部署堆疊的第二個區域。	DevOps
指定第二個區域的範本。	選取 CodeBuild 封裝並傾印至第二個區域的 S3 儲存貯體的區域特定套件名稱。	DevOps

部署階段：設定管道以部署到第三個區域

任務	Description	所需的技能
將第三個階段新增至部署階段。	若要為第三個區域新增階段，請編輯管道，然後在部署階段中選擇新增階段。重要：建立第二個區域的程序與前兩個區域的相同，但下列值除外。	DevOps

任務	Description	所需的技能
提供第三個階段的動作名稱。	輸入反映第三個階段和第三個區域的唯一名稱。	DevOps
設定第三個階段的區域。	選擇您要部署堆疊的第三個區域。	DevOps
指定第三個區域的範本。	選取 CodeBuild 封裝的區域特定套件名稱，並傾印到第三個區域的 S3 儲存貯體。	DevOps

清除部署

任務	Description	所需的技能
刪除 AWS 資源。	若要清除部署，請刪除每個區域中的 CloudFormation 堆疊。然後從主要區域刪除 CodeCommit、CodeBuild 和 CodePipeline 資源。	DevOps

相關資源

- 什麼是 AWS CodePipeline？
- AWS Serverless 應用程式模型
- AWS CloudFormation
- AWS CodePipeline 的 AWS CloudFormation 架構結構參考 AWS CodePipeline

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

將工作負載從 Azure DevOps 管道部署到私有 Amazon EKS 叢集

Mahendra Revanasiddappa，Amazon Web Services

摘要

此模式示範如何實作從 Azure DevOps 管道到私有 Amazon Elastic Kubernetes Service (Amazon EKS) 叢集的持續整合和持續交付 (CI/CD)。它解決了組織透過轉換到 Amazon EKS 叢集的私有 API 伺服器端點來增強其安全狀態面臨的關鍵挑戰。

公有端點會將 Kubernetes API 伺服器直接公開至網際網路，進而建立惡意人士可能鎖定的更大攻擊面。透過切換到私有端點，對叢集控制平面的存取僅限於客戶虛擬私有雲端 (VPC) 內。

雖然將 Amazon EKS 叢集轉換為私有 API 端點可大幅增強安全性，但它為 Azure DevOps 等外部 CI/CD 平台帶來連線挑戰。私有端點只能從叢集的 VPC 或對等網路存取。因此，在 AWS 私有網路外操作的標準 Microsoft 託管 Azure DevOps 代理程式無法直接連線到 Kubernetes API 伺服器。這會中斷依賴在這些代理程式上執行之 kubectl 或 Helm 等工具的典型部署工作流程，因為它們無法建立與叢集的連線。

為了克服這個問題，此模式展示了在私有 Amazon EKS 叢集中使用自我託管 Azure DevOps 代理程式的有效方法。此解決方案提供卓越的成本最佳化、營運效率和可擴展性，同時保留安全需求。這種方法特別有利於尋求簡化多雲端 DevOps 程序的企業，而不會影響效能或安全性。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- AWS Command Line Interface (AWS CLI) 2.13.17 版或更新版本，已安裝。
- 已安裝 kubectl 1.25.1 版或更新版本。 https://kubernetes.io/docs/tasks/tools/
- 已建立私有 Amazon EKS 叢集 1.24 版或更新版本，具有建立命名空間、秘密和部署的許可。
- Amazon EKS 叢集中的工作者節點具有網際網路的傳出連線，因此在其上執行的 Azure DevOps 代理程式可以連線至 Azure DevOps 代理程式集區。
- 已建立 GitHub 帳戶。
- 具有設定服務連線之存取權的 Azure DevOps 專案，這些連線是在 Azure Pipelines 與外部或遠端服務之間建立的驗證連線。
- 針對前一點所述的 Azure DevOps 專案安裝的 1.15 AWS Toolkit for Azure DevOps 版或更新版本。如需安裝說明，請參閱 Visual Studio Marketplace AWS Toolkit for Azure DevOps中的 。

限制

- 有些 AWS 服務 無法全部使用 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

此模式會建立下列項目：

- Amazon ECR 儲存庫 - Amazon Elastic Container Registry (Amazon ECR) 儲存庫會將 Docker 映像與 Azure DevOps 代理程式和部署的範例應用程式一起存放。
- Azure DevOps 代理程式集區 - Azure DevOps 自我託管代理程式集區會註冊在私有 Amazon EKS 叢集上執行的代理程式。
- IAM 角色 - Azure 服務連線的 AWS Identity and Access Management (IAM) 角色，可提供對私有 Amazon EKS 叢集上執行之代理程式的必要存取權。
- Azure DevOps 服務連線 - Azure DevOps 帳戶中的服務連線，可使用 IAM 角色，為管道任務提供存取所需的存取權 AWS 服務。

下圖顯示在私有 Amazon EKS 叢集上部署自我託管 Azure DevOps 代理程式，以及在同一叢集上部署範例應用程式的架構。

該圖顯示以下工作流程：

1. 將自我託管 Azure DevOps 代理程式部署為 Amazon EKS 叢集內的部署。
2. Azure DevOps 代理程式會使用個人存取字符 (PAT) 進行身分驗證，連線到 Azure DevOps 帳戶上的代理程式集區。
3. Azure Pipelines 會使用來自 GitHub 儲存庫的程式碼，設定要部署的管道。
4. 管道從管道組態中設定的代理程式集區在代理程式上執行。Azure DevOps 代理程式會持續輪詢至 Azure DevOps 帳戶，以取得管道的任務資訊。
5. Azure DevOps 代理程式會在管道任務中建置 Docker 映像，並將映像推送至 Amazon ECR 儲存庫。
6. Azure DevOps 代理程式會將範例應用程式部署在名為 的命名空間中的私有 Amazon EKS 叢集上`webapp`。

工具

工具

- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 上執行 Kubernetes， AWS 而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。

其他工具

- Docker 是一組平台即服務 (PaaS) 產品，可在作業系統層級使用虛擬化在容器中交付軟體。
- kubectl 是一種命令列界面，可協助您針對 Kubernetes 叢集執行命令。

程式碼儲存庫

- 此模式的程式碼可在 GitHub deploy-kubernetes-resources-to-amazon-eks-using-azure-devops 儲存庫中使用。

最佳實務

- 如需 Amazon EKS，請參閱 Amazon EKS 最佳實務指南。
- 遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。

史詩

建立服務連線

任務	Description	所需的技能
尋找 Azure DevOps 組織 GUID。	登入您的 Azure DevOps 帳戶，然後使用下列 URL 來尋找組織 GUID：在 URL	AWS DevOps

任務	Description	所需的技能
	https://dev.azure.com/{DevOps_Org_ID}/_apis/projectCollections?api-version=6.0 中，將 取代{DevOps_org_ID} 為您的 Azure DevOps 組織 ID。	

任務	Description	所需的技能
在 中設定 IdP AWS 帳戶。	若要在 中設定 Azure 服務連線 AWS 帳戶 的身分提供者 (IdP)，請使用下列步驟： 1. 登入 AWS 管理主控台，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。 2. 在左側窗格中，選擇身分提供者。 3. 選擇 Add Provider (新增提供商)。 4. 選擇 OpenID Connect 作為提供者類型。 5. 針對提供者 URL，輸入 Azure DevOps 發行者 URL。Azure DevOps 的每個租用戶都有一個唯一`OrganizationGUID` ，通常使用以下格式：將 `https://vstoken.dev.azure.com/{OrganizationGUID}` 取代`{OrganizationGUID}` 為您的 Azure DevOps 組織 ID。 6. 針對對象，輸入 api：//AzureADTokenExchange。這是 Azure DevOps 的固定值。 7. 選擇 Add Provider (新增提供商)。	AWS DevOps

任務	Description	所需的技能
	8. 請記下新建立供應商的 ARN，以供您在下一個任務中使用。 如需詳細資訊，請參閱如何使用 OpenID Connect AWS 從 Azure DevOps 聯合到 。	

任務	Description	所需的技能
在 中建立 IAM 政策 AWS 帳戶。	若要建立 IAM 政策，為 Azure DevOps 管道使用的 IAM 角色提供必要的許可，請使用下列步驟： 1. 在 IAM 主控台的左側窗格中，選擇政策。 2. 選擇建立政策。 3. 對於指定許可，在政策編輯器中，選取 JSON。使用下列 JSON 取代預設 JSON 政策： `{ "Version": "2012-10-17", "Statement": [ { "Sid": "Statement1", "Effect": "Allow", "Action": [ "ecr:*", "eks:DescribeCluster", "eks:ListClusters" ], "Resource": "*" } ] }`	AWS DevOps

任務	Description	所需的技能
	4. 選擇下一步。 5. 在政策名稱中，輸入 IAM 政策的名稱。此模式使用名稱 ADO-policy。 6. 選擇建立政策。	

任務	Description	所需的技能
在 中建立 IAM 角色 AWS 帳戶。	若要在 中 AWS 帳戶 為 Azure 服務連線設定 IAM 角色，請使用下列步驟： 1. 在 IAM 主控台的左側窗格中，選擇角色。 2. 選擇建立角色。 3. 針對信任的實體類型，選取 Web Identity。 4. 從下拉式清單中選取正確的 IdP。IdP 名稱以 開頭`vstoken.dev.azure.com/{OrganizationGUID}` 。 5. 在對象下拉式清單中，選取 api：//AzureADTokenExchange。 6. 若要將此角色限制為僅一個服務連線，請新增條件。在條件下，選取新增條件，然後針對金鑰選取 vstoken.dev.azure.com/{OrganizationGUID}:sub。在條件中，選取 StringEquals。對於值，請使用下列格式：`sc://{OrganizationName}/{ProjectName}/{ServiceConnectionName}` 。對於 `ServiceConnectionName` ，請使用 aws-sc。您將在下一個任務中建立此服務連線。	AWS DevOps

任務	Description	所需的技能
	7. 選擇下一步。 8. 針對新增許可，選取 ADO-policy，這是您在上一個任務中建立的政策。 9. 選擇下一步，然後在角色名稱中輸入 ado-role。對於選取信任的實體，請使用下列信任政策： `{ "Version": "2012-10-17", "Statement": [{ "Effect": "Allow", "Principal": { "Federated": "arn:aws:iam::{account_id}:oidc-provider/vstoken.dev.azure.com/{OrganizationGUID}" }, "Action": "sts:AssumeRoleWithWebIdentity", "Condition": { "StringEquals": { "vstoken.dev.azure.com/{OrganizationGUID}:aud": "api://AzureADTokenExchange", "vstoken.dev.azure.com/{OrganizationGUID}:sub": "sc://{OrganizationName}/{ProjectNam`	

任務	Description	所需的技能

	<pre>e}/{ServiceConnect
	ionName}"
	}
	}
	}
	]
	}</pre>
	在政策中，提供下列預留位置的資訊：
	• {account_id} - AWS 帳戶 ID
	• {OrganizationGUID} - Azure DevOps Organization GUID
	• {OrganizationName} - Azure DevOps Organization 名稱
	• {ProjectName} - Azure DevOps 專案名稱
	• {ServiceConnection Name} - Azure DevOps 服務連線名稱。使用 aws-sc。您將在下一個任務中建立此服務連線。	

任務	Description	所需的技能
在 Azure DevOps 帳戶中建立服務連線。	若要設定 Azure 服務連線，請使用下列步驟： 1. 在 Azure DevOps 專案中，選取專案設定、服務連線。 2. 選擇新增服務連線，選取服務連線的類型做為 aws，然後選取下一步。 3. 針對要擔任的角色，輸入 IAM 角色 ado-role 的 arn。您在上一個任務中建立 ado-role 在 中建立 IAM 角色 AWS 帳戶。 4. 選取使用 OIDC 核取方塊。 5. 針對服務連線名稱，在任務屬性中輸入 aws-sc。 6. 選擇儲存。 如需詳細資訊，請參閱 Microsoft 文件中的建立服務連線。	AWS DevOps

任務	Description	所需的技能

將 IAM 角色新增至 Amazon EKS 組態檔案。	IAM 角色必須具有必要的許可，才能在 Amazon EKS 叢集上執行必要的操作。由於它是管道角色，IAM 角色必須能夠管理叢集上幾乎所有類型的資源。因此，`system:masters` 群組許可適用於此角色。
	
	若要將必要的組態新增至 Kubernetes `aws-auth` `ConfigMap` 中的 ，請使用下列程式碼：
	
	`- groups:`
	`  - system:masters`
	`  rolearn: arn:aws:iam::{account_id}:role/ADO-role`
	`  username: ADO-role`
	
	`{account_id}` 以您的 AWS 帳戶 ID 取代 。
	
	如需詳細資訊，請參閱 Amazon EKS 文件中的 Amazon EKS 如何與 IAM 搭配使用。
		AWS DevOps

建立代理程式集區

任務	Description	所需的技能
建立自我託管代理程式集區。	若要在 Azure DevOps 帳戶中設定自我託管代理程式集區，請使用下列步驟： 1. 登入 Azure DevOps 帳戶組織。 2. 選擇 Azure DevOps Organization。 3. 選擇您的 Azure DevOps 專案。 4. 選擇 Project settings (專案設定)。 5. 選擇客服人員集區。 6. 選擇新增集區。 7. 選取自我託管。 8. 針對名稱，輸入 eks-agent。 9. 選取授予所有管道的存取許可核取方塊。 10選擇建立。 如需詳細資訊，請參閱 Microsoft 文件中的建立和管理代理程式集區。	

建置 Azure DevOps 代理程式映像並推送至 Amazon ECR

任務	Description	所需的技能
建立 Amazon ECR 儲存庫。	用於在私有 Amazon EKS 叢集上部署 Azure DevOps 代理程式和範例應用程式 (webapp) 的 Docker 映像必須存放在 Amazon ECR 儲存庫中。若要建立 Amazon ECR 儲存庫，請使用下列步驟： 1. 在 https://console.aws.amazon.com/ecr/repositories 開啟 Amazon ECR 主控台。 2. 從導覽列中，選擇要AWS區域在其中建立儲存庫的 。 3. 在儲存庫頁面上，選擇私有儲存庫，然後選擇建立儲存庫。 4. 針對儲存庫名稱，輸入 webapp。若要讓此模式中的範例應用程式運作，Amazon ECR 儲存庫名稱必須使用 webapp。如果您為儲存庫使用不同的名稱，請參閱故障診斷。 如需詳細資訊，請參閱 Amazon ECR 文件中的建立 Amazon ECR 私有儲存庫以存放映像。	AWS DevOps
建立 Dockerfile 以建置 Azure DevOps 代理程式。	建立 Dockerfile 以建置已安裝 Azure DevOps 代理程式的	AWS DevOps

任務	Description	所需的技能

	Docker 映像。將下列內容存放在名為 的檔案中Dockerfile ：
	
	`FROM ubuntu:22.04`
	`ENV TARGETARCH="linux-x64"`
	`RUN apt update && apt upgrade -y && apt install -y curl git jq libicu70 unzip wget`
	
	`RUN curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"`
	`RUN unzip awscliv2.zip`
	`RUN ./aws/install`
	`RUN rm -rf aws awscliv2.zip`
	
	`RUN curl -sSL https://get.docker.com/ | sh`
	
	`RUN curl -sL https://aka.ms/InstallAzureCLIDeb | bash`
	`RUN mkdir -p azp`
	`WORKDIR /azp/`
	
	`COPY ./start.sh ./`
	`RUN chmod +x ./start.sh`
	
	`RUN useradd -m -d /home/agent agent`
	`RUN chown -R agent:agent /azp /home/agent`
	`RUN groupadd -f docker`	

任務	Description	所需的技能
	``` RUN usermod -aG docker agent USER agent ENTRYPOINT ["./start. sh"] ```	
建立 Azure DevOps 代理程式的指令碼。	若要建立`start.sh`指令碼，請使用下列步驟： 1. 前往 Microsoft 文件中的程序 建立並建置 Dockerfile，然後捲動至步驟 5。將下列內容儲存至 `~/azp-agent-in-docker/start.sh` ，請務必使用 Unix 樣式 (LF) 行尾。 2. 複製指令碼的內容，並將其儲存在與 Dockerfile `start.sh` 相同的目錄中名為 的檔案中。	AWS DevOps

任務	Description	所需的技能
使用 Azure DevOps 代理程式建置 Docker 映像。	若要建立 Docker 映像以安裝 Azure DevOps 代理程式，請使用您先前建立的 Dockerfile 來建置映像。在存放 Dockerfile 的相同目錄中，執行下列命令： `aws ecr get-login-password --region region	docker login --username AWS --password-stdin aws_account_id.dkr.ecr.region.amazonaws.com`  `docker build --platform linux/amd64 -t ado-agent:latest .`  `docker tag ado-agent:latest aws_account_id.dkr.ecr.region.amazonaws.com/webapp:latest`  `docker push aws_account_id.dkr.ecr.region.amazonaws.com/webapp:latest`  `region` 以您的 AWS 帳戶 ID `aws_account_id` 和 取代 和 AWS 區域。	AWS DevOps

將 Azure DevOps 代理程式部署到私有 Amazon EKS 叢集

任務	Description	所需的技能

產生 Azure 個人存取字符。	在私有 Amazon EKS 叢集上執行的代理程式需要個人存取字符 (PAT)，以便可以向 Azure DevOps 帳戶進行身分驗證。若要產生 PAT，請使用下列步驟：
	1. 使用您計劃在 Azure DevOps 組織中使用的使用者帳戶登入 (`https://dev.azure.com/{Your_Organization}`)。
	1. 從您的首頁開啟您的使用者設定，然後選取個人存取字符。
	2. 選擇新權杖。
	3. 輸入字符的名稱。
	4. 選擇顯示所有範圍。
	5. 針對客服人員集區，選取讀取和管理核取方塊。
	6. 選擇建立。
	7. 若要在私有 Amazon EKS 叢集上建立秘密，請使用下列組態：
	`apiVersion: v1`
	`kind: Secret`
	`metadata:`
	`  name: azdevops-pat`
	`  namespace: default`
	`type: Opaque`
		AWS DevOps

任務	Description	所需的技能

	<pre>stringData:
	  AZP_TOKEN: <PAT
	 Token></pre>
	1. 將組態存放在名為 的檔案中ado-secret.yaml 。<PAT Token> 將 取代為您剛建立的個人存取字符。若要建立秘密，請執行下列命令：
	
	<pre>kubectl create -f ado-secret.yaml</pre>
	如需詳細資訊，請參閱 Microsoft 文件中的使用個人存取字符 (PAT) 註冊代理程式。	

任務	Description	所需的技能

使用 Kubernetes 資訊清單檔案進行代理程式部署。	若要在私有 Amazon EKS 叢集上部署 Azure DevOps 代理程式，請複製下列資訊清單檔案，並將檔案儲存為 `agent-deployment.yaml` ：
	
	`apiVersion: apps/v1`
	`kind: Deployment`
	`metadata:`
	`  name: azure-pip`
	`elines-agent-eks`
	`  labels:`
	`  app: azure-pip`
	`elines-agent`
	`spec:`
	`  replicas: 1`
	`  selector:`
	`  matchLabels:`
	`  app: azure-pip`
	`elines-agent`
	`  template:`
	`  metadata:`
	`  labels:`
	`  app: azure-pip`
	`elines-agent`
	`  spec:`
	`  containers:`
	`  - name: docker`
	`  image: docker:di`
	`nd`
	`  securityC`
	`ontext:`
	`  privileged:`
	` true`
	`  volumeMounts:`
	`  - name: shared-`
	`workspace`
	`  mountPath: /`
	`workspace`
		AWS DevOps

任務	Description	所需的技能

	<pre>- name: dind-
	storage
	mountPath: /
	var/lib/docker
	env:
	- name:
	DOCKER_TLS_CERTDIR
	value: ""
	- name: azure-pip
	elines-agent
	image: aws_accou
	nt_id.dkr.ecr.regi
	on.amazonaws.com/w
	ebapp:latest
	env:
	- name: AZP_URL
	value: "<Azure
	account URL>"
	- name: AZP_POOL
	value: "eks-
	agent"
	- name:
	AZP_TOKEN
	valueFrom:
	secretKey
	Ref:
	name:
	azdevops-pat
	key:
	AZP_TOKEN
	- name:
	AZP_AGENT_NAME
	valueFrom:
	fieldRef:
	fieldPath
	: metadata.name
	- name:
	DOCKER_HOST
	value: tcp://
	localhost:2375
	volumeMounts:</pre>	

任務	Description	所需的技能

	<pre>- mountPath: /workspace
	name: shared-workspace
	volumes:
	- name: dind-storage
	emptyDir: {}
	- name: shared-workspace
	emptyDir: {}</pre>
	將 `aws_account_id` 和 `取代<Azure account URL>`為您的 AWS 帳戶 ID 和 Azure DevOps 帳戶 URL。	

在私有 Amazon EKS 叢集上部署代理程式。	若要在私有 Amazon EKS 叢集上部署 Azure Devops 代理程式，請使用下列命令： <pre>kubectl create -f agent-deployment.tf</pre>	AWS DevOps

任務	Description	所需的技能

驗證代理程式是否正在執行。	若要驗證 Azure DevOps 代理程式是否正在執行，請使用下列命令：
	`kubectl get deploy azure-pipelines-agent-eks`
	預期的輸出應類似於以下內容：
	`NAME READY UP-TO-DATE AVAILABLE AGE azure-pipelines-agent-eks 1/1 1 1 58s`
	請確定資料READY 欄顯示1/1。
		AWS DevOps

任務	Description	所需的技能
確認代理程式已向 Azure DevOps 代理程式集區註冊。	若要確認代理程式已部署在私有 Amazon EKS 叢集上，並已向代理程式集區 註冊`eks-agent` ，請使用下列步驟： 1. 登入 Azure DevOps 組織 (`https://dev.azure.com/{Your_Organization}`)。 2. 選擇 Project settings (專案設定)。 3. 選擇客服人員集區。 4. 選取 eks-agent 集區，然後檢查 Agents 索引標籤。 您應該會看到一個以線上狀態列出的代理程式，而代理程式的名稱應以 azure-pipelines-agent-eks-* 開頭。	AWS DevOps

部署範例應用程式

任務	Description	所需的技能
將範例應用程式儲存庫授與您的 GitHub 帳戶。	將下列 AWS 範例儲存庫授與您的 GitHub 帳戶： https://github.com/aws-samples/deploy-kubernetes-resources-to-amazon-eks-using-azure-devops	AWS DevOps

任務	Description	所需的技能
建立管道。	若要在您的 Azure DevOps 帳戶中建立管道，請使用下列步驟： 1. 使用您計劃在 Azure DevOps 組織中使用的使用者帳戶登入 (`https://dev.azure.com/{Your_Organization}`)。 2. 導覽至您的專案和管道主控台。 3. 選擇新管道。 4. 針對您的程式碼所在位置，選擇 GitHub。 5. 提供管道連線至 GitHub 帳戶所需的登入資料 6. 選取儲存庫 deploy-kubernetes-resources-to-amazon-eks-using-azure-devops。 7. 針對設定管道，選取現有的 Azure 管道 YAML 檔案。 8. 針對選取現有的 YAML 檔案，選取分支的主要 ，以及選取路徑的主要 azure_pipelines.yaml。 9. 選擇繼續。 10對於檢閱管道 YAML，請將 `awsRegion` 和 的輸入參數值取代`awsEKSClusterName` 為您的資訊：	AWS DevOps

任務	Description	所需的技能

	<pre>pool:
	  name: eks-agent
	#pool: self-hosted # If you are running self-hosted Azure DevOps Agents
	
	stages:
	# Refering the pipeline template, input parameter that are not specified will be added with defaults
	- template: ./pipeline_templates/main_template.yaml
	  parameters:
	  serviceConnectionName: aws-sc
	  awsRegion: <your region>
	  awsEKSClusterName: <name of your EKS cluster>
	  projectName: webapp</pre>
	1. 選擇 RUN。	

任務	Description	所需的技能

確認範例應用程式已部署。	管道完成後，請檢查 Amazon ECR 儲存庫和 Amazon EKS 叢集，以驗證範例應用程式是否成功部署。
	
	若要驗證 Amazon ECR 儲存庫中的成品，請使用下列步驟：
	
	1. 導覽至 webapp Amazon ECR 儲存庫。
	2. 確認下列新成品是否存在：
	
	• Docker 映像 – <date>.<build_number>-image
	• Helm Chart – <date>.<build_number>-helm
	
	例如，20250501.1-image 和 20250501.1-helm 。
	
	若要驗證命名空間 中私有 Amazon EKS 叢集上的部署webapp，請使用下列命令：
	
	`kubectl get deploy -n webapp`
	
	預期的輸出如下：
	
	`NAME READY UP-TO-DATE AVAILABLE`
	`webapp 1/1 1 1`
		AWS DevOps

任務	Description	所需的技能
	注意：如果這是您的第一個管道執行，您可能需要授權服務連線和代理程式集區。在 Azure DevOps 管道界面中尋找許可請求，並核准它們以繼續。	

疑難排解

問題	解決方案
當 Amazon ECR 儲存庫名稱不相符時管道失敗 `webapp`	範例應用程式預期 Amazon ECR 儲存庫名稱符合 中的 `projectName: webapp` 參數`azure_pipeline.yml` 。 若要解決此問題，請將 Amazon ECR 儲存庫重新命名為 `webapp`，或更新下列項目： • 重新命名分叉 GitHub 儲存庫中的`webapp`目錄，以符合您的 Amazon ECR 儲存庫名稱。 • 更新 中的 `projectName` 參數`azure_pipeline.yml` ，以符合您的 Amazon ECR 儲存庫名稱。
錯誤：Kubernetes 叢集無法連線：伺服器已要求用戶端提供登入資料	如果您在 Azure 管道的「提取和部署 Helm Chart」步驟中遇到此錯誤，根本原因通常是來自 Amazon EKS 叢集 中的不正確 IAM 角色組態`aws-auth ConfigMap` 。 若要解決此問題，請檢查下列項目： • 驗證您的`aws-auth ConfigMap` 組態。 • 檢查 Amazon EKS 叢集的身分驗證設定：開啟 Amazon EKS 主控台、叢集詳細資訊、存

問題	解決方案
	取組態。確定身分驗證模式設定為 EKS API 和 ConfigMap（不只是 EKS API)。

相關資源

AWS 部落格

- 如何使用 OpenID Connect AWS 從 Azure DevOps 聯合到

AWS 服務 文件

- Amazon ECR
- Amazon EKS

Microsoft 文件

- 什麼是 Azure DevOps？
- 什麼是 Azure Pipelines？

使用 Terraform 執行 Amazon Redshift SQL 查詢

Sylvia Qi 和 Aditya Ambati，Amazon Web Services

總結

使用基礎設施做為程式碼 (IaC) 來部署和管理 Amazon Redshift 是 DevOps 中普遍的做法。IaC 可促進各種 Amazon Redshift 資源的部署和組態，例如叢集、快照和參數群組。不過，IaC 不會延伸至管理資料庫資源，例如資料表、結構描述、檢視和預存程序。這些資料庫元素是透過 SQL 查詢管理，IaC 工具不會直接支援。雖然存在管理這些資源的解決方案和工具，但您可能不想在技術堆疊中引入其他工具。

此模式概述使用 Terraform 部署 Amazon Redshift 資料庫資源的方法，包括資料表、結構描述、檢視和預存程序。模式區分兩種類型的 SQL 查詢：

- 不可重複的查詢 – 這些查詢會在初始 Amazon Redshift 部署期間執行一次，以建立必要的資料庫元件。
- 可重複的查詢 – 這些查詢是不可變的，可以在不影響資料庫的情況下重新執行。解決方案使用 Terraform 來監控可重複查詢中的變更，並相應地套用變更。

如需詳細資訊，請參閱其他資訊中的解決方案逐步解說。

先決條件和限制

先決條件

您必須有作用中的 ， AWS 帳戶 並在部署機器上安裝下列項目：

- AWS Command Line Interface (AWS CLI)
- 使用 Amazon Redshift 讀取/寫入許可設定的AWS CLI 設定檔
- Terraform 1.6.2 版或更新版本
- Python3
- Boto3

限制

- 此解決方案支援單一 Amazon Redshift 資料庫，因為 Terraform 僅允許在叢集建立期間建立一個資料庫。

- 此模式不包含在套用可重複查詢之前驗證變更的測試。我們建議您整合此類測試，以獲得增強的可靠性。
- 為了說明解決方案，此模式提供使用本機 Terraform 狀態`redshift.tf`檔案的範例檔案。不過，對於生產環境，我們強烈建議您使用具有鎖定機制的遠端狀態檔案，以增強穩定性和協同合作。
- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

此解決方案是在 Amazon Redshift 修補程式 179 上開發和測試。

程式碼儲存庫

此模式的程式碼可在 GitHub amazon-redshift-sql-deploy-terraform 儲存庫中使用。

Architecture

下圖說明 Terraform 如何透過處理不可重複和可重複的 SQL 查詢來管理 Amazon Redshift 資料庫資源。

圖表顯示下列步驟：

1. Terraform 會在初始 Amazon Redshift 叢集部署期間套用不可重複的 SQL 查詢。
2. 開發人員會將變更遞交至可重複的 SQL 查詢。
3. Terraform 會監控可重複 SQL 查詢中的變更。
4. Terraform 會將可重複的 SQL 查詢套用至 Amazon Redshift 資料庫。

此模式提供的解決方案是根據 Amazon Redshift 的 Terraform 模組所建置。Terraform 模組會佈建 Amazon Redshift 叢集和資料庫。為了增強模組，我們使用 `terraform_data` 資源，這會呼叫自訂 Python 指令碼，以使用 Amazon Redshift ExecuteStatement API 操作執行 SQL 查詢。因此，模組可以執行下列動作：

- 在佈建資料庫之後，使用 SQL 查詢部署任意數量的資料庫資源。
- 持續監控可重複 SQL 查詢中的變更，並使用 Terraform 套用這些變更。

如需詳細資訊，請參閱其他資訊中的解決方案逐步解說。

工具

AWS 服務

- Amazon Redshift 是 中全受管的 PB 級資料倉儲服務 AWS 雲端。

其他工具

- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。
- Python 是一種一般用途的程式設計語言，用於此模式來執行 SQL 查詢。

最佳實務

- Amazon Redshift 最佳實務
- 使用 Amazon Redshift Data API 與 Amazon Redshift 叢集互動

史詩

使用 Terraform 部署解決方案

任務	Description	所需的技能
複製儲存庫。	若要複製包含 Terraform 程式碼的 Git 儲存庫來佈建 Amazon Redshift 叢集，請使用下列命令。 `git clone https://github.com/aws-samples/amazon-redshift-sql-deploy-terraform.git`	DevOps 工程師
更新 Terraform 變數。	若要根據您的特定需求自訂 Amazon Redshift 叢集部署，	DevOps 工程師

任務	Description	所需的技能
	請在 terraform.tfvars 檔案中更新下列參數。 region = "<AWS_REGION>" cluster_identifier = "<REDSHIFT_CLUSTER_IDENTIFIER>" node_type = "<REDSHIFT_NODE_TYPE>" number_of_nodes = "<REDSHIFT_NODE_COUNT>" database_name = "<REDSHIFT_DB_NAME>" subnet_ids = "<REDSHIFT_SUBNET_IDS>" vpc_security_group_ids = "<REDSHIFT_SECURITY_GROUP_IDS>" run_nonrepeatable_queries = true run_repeatable_queries = true sql_path_bootstrap = "<BOOTSTRAP_SQLS_PATH>" sql_path_nonrepeatable = "<NON-REPEATABLE_SQLS_PATH>" sql_path_repeatable = "<REPEATABLE_SQLS_PATH>" sql_path_finalize = "<FINALIZE_SQLS_PATH>"	

任務	Description	所需的技能
	`create_random_password = false` `master_username = "<REDSHIFT_MASTER_USERNAME>"`	

使用 Terraform 部署資源。	1. 若要準備部署程序，請使用下列命令在複製的儲存庫中初始化 Terraform。
	`terraform init`
	2. 若要預覽 Terraform 將套用至基礎設施的變更，請使用下列命令來建立執行計畫。
	`terraform plan -var-file terraform.tfvars`
	3. 若要佈建 Amazon Redshift 叢集和相關聯的資源，請使用下列命令來套用 Terraform 執行計畫。
	`terraform apply -var-file terraform.tfvars`
		DevOps 工程師

任務	Description	所需的技能

（選用）執行其他 SQL 查詢。	範例儲存庫提供數個 SQL 查詢供示範使用。若要執行您自己的 SQL 查詢，請將它們新增至下列資料夾：
	`/bootstrap`
	`/nonrepeatable`
	`/repeatable`
	`/finalize`	

監控 SQL 陳述式的執行

任務	Description	所需的技能
監控 SQL 陳述式的部署。	您可以監控 Amazon Redshift 叢集的 SQL 執行結果。如需顯示失敗和成功 SQL 執行的輸出範例，請參閱其他資訊中的範例 SQL 陳述式。	DBA，DevOps 工程師
清除資源。	若要刪除 Terraform 部署的所有資源，請執行下列命令。 `terraform destroy`	DevOps 工程師

驗證結果

任務	Description	所需的技能
驗證 Amazon Redshift 叢集中的資料。	1. 登入 AWS 管理主控台，然後開啟 Amazon Redshift 主控台。	DBA、AWS DevOps

任務	Description	所需的技能
	2. 在導覽選單上，選擇叢集。在清單中選擇相關的叢集名稱。 3. 請遵循 Amazon Redshift 文件中的使用 Amazon Redshift 查詢編輯器 v2 查詢資料庫中的指示。	

相關資源

AWS 文件

- Amazon Redshift 佈建叢集
- 對 Amazon Redshift Data API 的問題進行故障診斷

其他資源

- 命令：套用 (Terraform 文件)

其他資訊

解決方案演練

若要使用解決方案，您必須以特定方式組織 Amazon Redshift SQL 查詢。所有 SQL 查詢都必須存放在副`.sql`檔名為 的檔案中。

在此模式提供的程式碼範例中，SQL 查詢會以下列資料夾結構組織。您可以修改程式碼 (`sql-queries.tf` 和 `sql-queries.py`)，以使用符合您唯一使用案例的任何結構。

```
/bootstrap
     |- Any # of files
     |- Any # of sub-folders
/nonrepeatable
     |- Any # of files
     |- Any # of sub-folders
/repeatable
     /udf
```

```
          |- Any # of files
          |- Any # of sub-folders
     /table
          |- Any # of files
          |- Any # of sub-folders
     /view
          |- Any # of files
          |- Any # of sub-folders
     /stored-procedure
          |- Any # of files
          |- Any # of sub-folders
/finalize
     |- Any # of files
     |- Any # of sub-folders
```

根據上述資料夾結構，在 Amazon Redshift 叢集部署期間，Terraform 會依下列順序執行查詢：

1. `/bootstrap`
2. `/nonrepeatable`
3. `/repeatable`
4. `/finalize`

`/repeatable` 資料夾包含四個子資料夾：`/udf`、`/view`、 `/table`和 `/stored-procedure`。這些子資料夾指出 Terraform 執行 SQL 查詢的順序。

執行 SQL 查詢的 Python 指令碼為 `sql-queries.py`。首先，指令碼會讀取特定來源目錄的所有檔案和子資料夾，例如 `sql_path_bootstrap` 參數。然後，指令碼會透過呼叫 Amazon Redshift ExecuteStatement API 操作來執行查詢。您可能在檔案中有一或多個 SQL 查詢。下列程式碼片段顯示 Python 函數，該函數會對 Amazon Redshift 叢集執行存放在檔案中的 SQL 陳述式。

```
def execute_sql_statement(filename, cluster_id, db_name, secret_arn, aws_region):
    """Execute SQL statements in a file"""
    redshift_client = boto3.client(
        'redshift-data', region_name=aws_region)
    contents = get_contents_from_file(filename),
    response = redshift_client.execute_statement(
        Sql=contents[0],
        ClusterIdentifier=cluster_id,
        Database=db_name,
        WithEvent=True,
```

```
        StatementName=filename,
        SecretArn=secret_arn
    )
    ...
```

Terraform 指令碼會sql-queries.tf建立叫用sql-queries.py指令碼的 terraform_data 資源。四個資料夾各有一個terraform_data資源：/bootstrap、/repeatable、 /nonrepeatable和 /finalize。下列程式碼片段顯示執行 /bootstrap 資料夾中 SQL 查詢terraform_data的資源。

```
locals {
  program               = "${path.module}/sql-queries.py"
  redshift_cluster_name = try(aws_redshift_cluster.this[0].id, null)
}

resource "terraform_data" "run_bootstrap_queries" {
  count      = var.create && var.run_nonrepeatable_queries && (var.sql_path_bootstrap !
= "") && (var.snapshot_identifier == null) ? 1 : 0
  depends_on = [aws_redshift_cluster.this[0]]

  provisioner "local-exec" {
    command = "python3 ${local.program} ${var.sql_path_bootstrap}
 ${local.redshift_cluster_name} ${var.database_name} ${var.redshift_secret_arn}
 ${local.aws_region}"
  }
}
```

您可以使用下列變數來控制是否執行這些查詢。如果您不想在 sql_path_bootstrap、sql_path_repeatable、 sql_path_nonrepeatable或 中執行查詢sql_path_finalize，請將其值設定為 ""。

```
  run_nonrepeatable_queries = true
  run_repeatable_queries    = true
  sql_path_bootstrap        = "src/redshift/bootstrap"
  sql_path_nonrepeatable    = "src/redshift/nonrepeatable"
  sql_path_repeatable       = "src/redshift/repeatable"
  sql_path_finalize         = "src/redshift/finalize"
```

當您執行 時terraform apply，無論指令碼的結果為何，Terraform 都會考慮在指令碼完成後新增terraform_data的資源。如果某些 SQL 查詢失敗，而且您想要重新執行它們，您可以從 Terraform 狀態手動移除資源，然後terraform apply再次執行。例如，下列命令會從 Terraform 狀態移除run_bootstrap_queries資源。

```
terraform state rm module.redshift.terraform_data.run_bootstrap_queries[0]
```

下列程式碼範例顯示run_repeatable_queries資源如何使用 sha256 雜湊監控repeatable資料夾中的變更。如果資料夾中的任何檔案已更新，Terraform 會標記整個目錄以進行更新。然後，Terraform 會在下一個 期間再次執行目錄中的查詢terraform apply。

```
resource "terraform_data" "run_repeatable_queries" {
  count      = var.create_redshift && var.run_repeatable_queries &&
 (var.sql_path_repeatable != "") ? 1 : 0
  depends_on = [terraform_data.run_nonrepeatable_queries]

  # Continuously monitor and apply changes in the repeatable folder
  triggers_replace = {
    dir_sha256 = sha256(join("", [for f in fileset("${var.sql_path_repeatable}",
 "**") : filesha256("${var.sql_path_repeatable}/${f}")]))
  }

  provisioner "local-exec" {
    command = "python3 ${local.sql_queries} ${var.sql_path_repeatable}
 ${local.redshift_cluster_name} ${var.database_name} ${var.redshift_secret_arn}"
  }
}
```

若要精簡程式碼，您可以實作機制來偵測並僅將變更套用至repeatable資料夾中已更新的檔案，而不是以不區分的方式套用變更至所有檔案。

SQL 陳述式範例

下列輸出顯示失敗的 SQL 執行，以及錯誤訊息。

```
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec): Executing:
 ["/bin/sh" "-c" "python3 modules/redshift/sql-queries.py src/redshift/nonrepeatable
 testcluster-1 db1 arn:aws:secretsmanager:us-east-1:XXXXXXXXXXXX:secret:/redshift/
master_user/password-8RapGH us-east-1"]
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec):
 ----------------------------------------------------------------------
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec): src/redshift/
nonrepeatable/table/admin/admin.application_family.sql
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec):
 ----------------------------------------------------------------------
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec): Status:
 FAILED
```

```
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec): SQL execution
 failed.
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec): Error
 message: ERROR: syntax error at or near ")"
module.redshift.terraform_data.run_nonrepeatable_queries[0] (local-exec):   Position:
 244
module.redshift.terraform_data.run_nonrepeatable_queries[0]: Creation complete after 3s
 [id=ee50ba6c-11ae-5b64-7e2f-86fd8caa8b76]
```

下列輸出顯示成功的 SQL 執行。

```
module.redshift.terraform_data.run_bootstrap_queries[0]: Provisioning with 'local-
exec'...
module.redshift.terraform_data.run_bootstrap_queries[0] (local-exec): Executing:
 ["/bin/sh" "-c" "python3 modules/redshift/sql-queries.py src/redshift/bootstrap
 testcluster-1 db1 arn:aws:secretsmanager:us-east-1:XXXXXXXXXXXX:secret:/redshift/
master_user/password-8RapGH us-east-1"]
module.redshift.terraform_data.run_bootstrap_queries[0] (local-exec):
 ----------------------------------------------------------------
module.redshift.terraform_data.run_bootstrap_queries[0] (local-exec): src/redshift/
bootstrap/db.sql
module.redshift.terraform_data.run_bootstrap_queries[0] (local-exec):
 ----------------------------------------------------------------
module.redshift.terraform_data.run_bootstrap_queries[0] (local-exec): Status: FINISHED
module.redshift.terraform_data.run_bootstrap_queries[0] (local-exec): SQL execution
 successful.
module.redshift.terraform_data.run_bootstrap_queries[0]: Creation complete after 2s
 [id=d565ef6d-be86-8afd-8e90-111e5ea4a1be]
```

從 AWS Organizations 中的整個組織將 AWS Backup 報告匯出為 CSV 檔案 AWS Organizations

Aromal Raj Jayarajan 和 Purushotham G K，Amazon Web Services

總結

此模式說明如何將 AWS Backup 任務報告從 AWS Organizations 中的組織匯出為 CSV 檔案。解決方案使用 AWS Lambda 和 Amazon EventBridge，根據其狀態來分類 AWS Backup 任務報告，這有助於設定以狀態為基礎的自動化。

AWS Backup 可協助組織集中管理和自動化跨 AWS 服務、雲端和內部部署的資料保護。不過，對於在 AWS Organizations 中設定的 AWS Backup 任務，合併報告只能在每個組織的管理帳戶的 AWS 管理主控台中使用。 AWS Organizations 將此報告帶出管理帳戶，可以減少稽核所需的工作量，並提高自動化、通知和提醒的範圍。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- AWS Organizations 中的作用中組織，至少包含管理帳戶和成員帳戶
- 在 AWS Organizations 中的組織層級AWS Organizations Backup （如需詳細資訊，請參閱 AWS 部落格上的使用 AWS Backup 跨 AWS 服務大規模自動化集中備份）
- 在本機電腦上安裝和設定的 Git

限制

此模式中提供的解決方案可識別僅針對 AWS Backup 任務設定的 AWS 資源。報告無法識別未設定為透過 AWS Backup 備份的 AWS Backup 資源。

Architecture

目標技術堆疊

- AWS Backup
- AWS CloudFormation
- Amazon EventBridge
- AWS Lambda

- AWS Security Token Service (AWS STS)
- Amazon Simple Storage Service (Amazon S3)
- AWS Identity and Access Management (IAM)

目標架構

下圖顯示將 AWS Backup 任務報告從 AWS Organizations 中的組織匯出為 CSV 檔案的範例工作流程。

該圖顯示以下工作流程：

1. 排程的 EventBridge 事件規則會在成員（報告）AWS 帳戶中叫用 Lambda 函數。
2. 然後，Lambda 函數會使用 AWS STS 擔任 IAM 角色，該角色具有連線到管理帳戶所需的許可。
3. Lambda 函數接著會執行下列動作：
   - 從 AWS Backup 服務請求合併的 AWS Backup 任務報告
   - 根據 AWS Backup 任務狀態將結果分類
   - 將回應轉換為 CSV 檔案
   - 在根據建立日期標記的資料夾中，將結果上傳至報告帳戶中的 Amazon S3 儲存貯體

工具

工具

- AWS Backup 是一項全受管服務，可協助您集中和自動化跨 AWS 服務、雲端和內部部署的資料保護。
- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶和區域的生命週期中管理這些資源。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如，AWS Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他 AWS 帳戶中的事件匯流排。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而不需要佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。

- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

程式碼

此模式的程式碼可在 GitHub aws-backup-report-generator 儲存庫中使用。

最佳實務

- Amazon S3 的安全最佳實務 (Amazon S3 使用者指南)
- 使用 AWS Lambda 函數的最佳實務 (AWS Lambda 開發人員指南)
- 管理帳戶的最佳實務 (AWS Organizations 使用者指南)

史詩

部署解決方案元件

任務	Description	所需的技能
複製 GitHub 儲存庫。	在終端機視窗中執行下列命令，複製 GitHub aws-backup-report-generator 儲存庫： `git clone https://github.com/aws-samples/aws-backup-report-generator.git` 如需詳細資訊，請參閱 GitHub 文件中的複製儲存庫。	AWS DevOps，DevOps 工程師
在成員（報告）AWS 帳戶中部署解決方案元件。	1. 在成員（報告）帳戶中，登入 AWS 管理主控台，然後開啟 CloudFormation 主控台。 2. 選擇 Create stack (建立堆疊)，然後選擇 With new	DevOps 工程師，AWS DevOps

任務	Description	所需的技能
	resources (standard) (使用新資源 (標準))。 3. 在建立堆疊頁面的指定範本區段中，選擇上傳範本檔案。 4. 選取 Choose file (選擇檔案)。然後，導覽至本機工作站上複製的 GitHub 儲存庫的根資料夾，然後選擇 template-reporting.yaml。 5. 選擇開啟，然後選擇下一步。 6. 在指定堆疊詳細資訊頁面上，針對堆疊名稱輸入 CloudFormation 堆疊的名稱。 7. 在 ManagementAccountID 中，輸入 AWS Organizations 中AWS Organizations 帳戶 ID。 8. 選擇下一步。 9. 在設定堆疊選項頁面上，選擇下一步。 10.在檢閱頁面上，選取核取方塊以確認您已檢閱組態。 11選擇建立堆疊。在成員（報告）帳戶中部署解決方案元件時，堆疊會顯示 CREATE_COMPLETE 狀態。	

測試解決方案

任務	Description	所需的技能
確定 EventBridge 規則在測試之前執行。	透過等待至少 24 小時，或增加 CloudFormation 範本的 template-reporting.yml 檔案中的報告頻率，確保 EventBridge 規則執行。 增加報告頻率 1. 在複製的儲存庫中開啟 template-reporting.yml 檔案。 2. 在邏輯 ID 為「LambdaSchedule」的事件規則中，尋找「ScheduleExpression」。 3. 編輯「ScheduleExpression」索引鍵，使其包含有效的 Cron 表達式。例如，下列 cron 表達式會排程事件規則每五分鐘執行一次：`"cron (* /5 * * * *)"`	AWS DevOps，DevOps 工程師
檢查 Amazon S3 儲存貯體是否有產生的報告。	1. 在成員（報告）帳戶中，登入 AWS 管理主控台，然後開啟 CloudFormation 主控台。 2. 在堆疊窗格中，選取您建立的堆疊名稱。然後，選擇資源索引標籤。 3. 在資源窗格中的邏輯 ID 欄中，尋找 BackupReportS3Bucket。然後，選取	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	邏輯 ID 旁邊的實體 ID 欄中的連結，在新標籤中開啟相關聯的 Amazon S3 儲存貯體。 4. 請確定儲存貯體包含以下列格式產生的報告：BackupReports/<yyyy>/<mm>/<dd>/BackupReport-<BACKUP JOB STATUS>-<dd>-<Mon>-<yyyy>.csv	

清除您的資源

任務	Description	所需的技能
從成員（報告）帳戶刪除解決方案元件。	1. 在成員（報告）帳戶中，開啟解決方案的 Amazon S3 儲存貯體。如需說明，請參閱 檢查 S3 儲存貯體中的步驟 2-4，以取得此模式測試解決方案一節產生的報告案例。 2. 刪除儲存貯體的內容並清空儲存貯體。如需說明，請參閱《Amazon S3 使用者指南》中的清空儲存貯體。 3. 在成員（報告）帳戶中，登入 AWS 管理主控台，然後開啟 CloudFormation 主控台。 4. 在堆疊窗格中，選取您所建立堆疊名稱旁的核取方塊。再選擇 Delete (刪除)。	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
從管理帳戶刪除解決方案元件。	1. 在管理帳戶中，登入 AWS 管理主控台，然後開啟 CloudFormation 主控台。 2. 在堆疊窗格中，選取您所建立堆疊名稱旁的核取方塊。再選擇 Delete (刪除)。	AWS DevOps，DevOps 工程師

相關資源

- 教學課程：搭配排程事件使用 AWS Lambda (AWS Lambda 文件）
- 建立排程事件以執行 AWS Lambda 函數 （適用於 JavaScript 的 AWS 開發套件文件）
- IAM 教學課程：使用 IAM 角色在 AWS 帳戶之間委派存取權 (IAM 文件）
- AWS Organizations 術語和概念 (AWS Organizations 文件）
- 使用 AWS Backup 主控台建立報告計畫 (AWS Backup 文件）
- 建立稽核報告 (AWS Backup 文件）
- 建立隨需報告 (AWS Backup 文件）
- 什麼是 AWS Backup？ (AWS Backup文件）
- 使用 AWS Backup 跨 AWS 服務大規模自動化集中AWS Backup(AWS 部落格文章）

將 Amazon EC2 執行個體清單的標籤匯出至 CSV 檔案

Sida Ju 和 Pac Joonhyun，Amazon Web Services

總結

此模式說明如何以程式設計方式將 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體清單的標籤匯出至 CSV 檔案。

透過使用提供的 Python 指令碼範例，您可以縮短依特定標籤檢閱和分類 Amazon EC2 執行個體所需的時間。例如，您可以使用指令碼快速識別和分類安全團隊為軟體更新標記的執行個體清單。

先決條件和限制

先決條件

- 已安裝並設定 Python 3
- 安裝並設定 AWS Command Line Interface (AWS CLI)

限制

此模式中提供的範例 Python 指令碼只能根據下列屬性搜尋 Amazon EC2 執行個體：

- 執行個體 ID
- 私有 IPv4 地址
- 公有 IPv4 地址

工具

- Python 是一種一般用途的電腦程式設計語言。
- virtualenv 可協助您建立隔離的 Python 環境。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。

程式碼儲存庫

此模式的範例 Python 指令碼可在 GitHub search-ec2-instances-export-tags 儲存庫中使用。

史詩

安裝和設定先決條件

任務	Description	所需的技能
複製 GitHub 儲存庫。	**Note** 如果您在執行 AWS CLI 命令時收到錯誤，請確定您使用的是最新的 AWS CLI 版本。 在終端機視窗中執行下列 Git 命令，複製 GitHub search-ec2-instances-export-tags 儲存庫： `git clone https://github.com/aws-samples/search-ec2-instances-export-tags.git`	DevOps 工程師

安裝並啟用 virtualenv。	1. 執行下列命令來安裝 virtualenv：
	
	`python3 -m pip install virtualenv`
	
	2. 執行下列命令來建立新的虛擬環境：
	
	`python3 -m venv env`
	
	3. 執行下列命令來啟用新的虛擬環境：
		DevOps 工程師

任務	Description	所需的技能
	`source env/bin/activate` 如需詳細資訊，請參閱 Virtualenv 使用者指南。	

安裝依存項目。	1. 在終端機中執行下列命令來開啟程式碼目錄：
	`cd search-ec2-instances-export-tags`
	2. 執行下列 pip 命令來安裝 `requirements.txt` 檔案：
	`pip3 install -r requirements.txt`
		DevOps 工程師

設定名為 的 AWS 設定檔。	如果您尚未設定名為 的 AWS 設定檔，其中包含執行指令碼所需的登入資料。若要建立具名設定檔，請執行 aws configure 命令。 如需詳細資訊，請參閱 AWS CLI 文件中的使用具名設定檔。	DevOps 工程師

設定並執行 Python 指令碼

任務	Description	所需的技能

建立輸入檔案。	建立輸入檔案，其中包含您要指令碼搜尋和匯出標籤的 Amazon EC2 執行個體清單。您可以列出執行個體 IDs、私有 IPv4 地址或公有 IPv4 地址。
	
	Important
	請確定每個 Amazon EC2 執行個體都列在輸入檔案中自己的行上。
	
	輸入檔案範例
	
	`i-0547c351bdfe85b9f`
	`54.157.194.156`
	`172.31.85.33`
	`54.165.198.144`
	`i-0b6223b5914111a4b`
	`172.31.85.44`
	`54.165.198.145`
	`172.31.80.219`
	`172.31.94.199`
		DevOps 工程師

執行 Python 指令碼。	在終端機中執行下列命令來執行指令碼： `python search_instances.py -i INPUTFILE -o OUTPUTFIL`	DevOps 工程師

任務	Description	所需的技能
	E -r REGION [-p PROFILE] **Note** INPUTFILE 將取代為您輸入檔案的名稱。OUTPUTFILE 將取代為您要提供 CSV 輸出檔案的名稱。REGION 將取代為您的 Amazon EC2 資源所在的 AWS 區域。如果您使用的是 AWS 具名設定檔，請將 PROFILE取代為您正在使用的具名設定檔。 若要取得支援的參數清單及其說明，請執行下列命令： python search_instances.py -h 如需詳細資訊並查看輸出檔案範例，請參閱 GitHub search-ec2-instances-export-tags 儲存庫中的 README.md 檔案。	

相關資源

- 設定 AWS CLI (AWS CLI 使用者指南)

使用 Troposphere 產生包含 AWS Config 受管規則的 AWS CloudFormation 範本

Lucas Nation 和 Freddie Wilson，Amazon Web Services

摘要

許多組織使用 AWS Config 受管規則，根據常見最佳實務評估其 Amazon Web Services (AWS) 資源的合規性。不過，這些規則可能會耗時維護，此模式可協助您利用 Python 程式庫 Troposphere 來產生和管理 AWS Config 受管規則。

模式可協助您使用 Python 指令碼，將包含 AWS Config受管規則的 Microsoft Excel 試算表轉換為 AWS CloudFormation 範本，以管理您的 AWS Config 受管規則。Troposphere 做為基礎設施做為程式碼 (IaC)，這表示您可以使用受管規則更新 Excel 試算表，而不是使用 JSON 或 YAML 格式的檔案。然後，您可以使用 範本來啟動 AWS CloudFormation 堆疊，以建立和更新 AWS 帳戶中的受管規則。

AWS CloudFormation 範本使用 Excel 試算表定義每個 AWS Config 受管規則，並協助您避免在 AWS 管理主控台中手動建立個別規則。指令碼會將每個受管規則的參數預設為空白字典，以及來自 的範圍的`ComplianceResourceTypes`預設值`THE_RULE_IDENTIFIER.template file`。 如需規則識別符的詳細資訊，請參閱 AWS Config 文件中的使用 AWS CloudFormation 範本建立 AWS Config 受管規則。 AWS Config

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 熟悉使用 AWS CloudFormation 範本建立 AWS Config 受管規則。如需詳細資訊，請參閱 AWS Config 文件中的使用 AWS CloudFormation 範本建立 AWS Config 受管規則。 AWS Config
- Python 3，已安裝和設定。如需詳細資訊，請參閱 Python 文件。
- 現有的整合開發環境 (IDE)。
- 在範例 `excel_config_rules.xlsx` Excel 試算表 (OUs)。

史詩

自訂和設定 AWS Config 受管規則

任務	Description	所需的技能
更新範例 Excel 試算表。	下載範例 `excel_config_rules.xlsx` Excel 試算表（已連接），並標記為您要使用的 `Implemented` AWS Config 受管規則。 標記為 的規則`Implemented` 將新增至 AWS CloudFormation 範本。	開發人員
（選用）使用 AWS Config 規則參數更新 config_rules_params.json 檔案。	有些 AWS Config 受管規則需要參數，應該使用 `--param-file` 選項以 JSON 檔案的形式傳遞至 Python 指令碼。例如，`access-keys-rotated` 受管規則使用下列`maxAccessKeyAge` 參數： `{ "access-keys-rotated": { "InputParameters": { "maxAccessKeyAge": 90 } } }` 在此範例參數中，`maxAccessKeyAge` 設定為 90 天。指令碼會讀取參數檔案`InputPara`	開發人員

任務	Description	所需的技能
	meters ，並新增它找到的任何 。	
（選用）使用 AWS Config ComplianceResourceTypes 更新 config_rules_params.json 檔案。 ComplianceResourceTypes	根據預設，Python 指令碼ComplianceResourceTypes 會從 AWS 定義的範本擷取 。如果您想要覆寫特定 AWS Config 受管規則的範圍，則需要使用 --param-file 選項將其做為 JSON 檔案傳遞至 Python 指令碼。 例如，下列範例程式碼顯示 ComplianceResourceTypes 的 ec2-volume-inuse-check 如何設定為 ["AWS::EC2::Volume"] 清單： `{ "ec2-volume-inuse-check": { "Scope": { "ComplianceResourceTypes": [ "AWS::EC2::Volume" ] } } }`	開發人員

執行 Python 指令碼

任務	Description	所需的技能
從 requirements.txt 檔案安裝 pip 套件。	下載 `requirements.txt` 檔案 （已連接），並在 IDE 中執行下列命令來安裝 Python 套件： `pip3 install -r requirements.txt`	開發人員
執行 Python 指令碼。	1. 下載 `aws_config_rules.py` 檔案 （已連接）到您的本機電腦。 2. **Note** 執行 `- python3 aws_config_rules.py --ou <OU_NAME>` 命令。：`--ou`定義要在 Excel 試算表中選擇的 OU 資料欄。 您也可以新增下列選用參數： • `--config-rule-option` – 定義從 Excel 試算表中選擇的規則。預設值為 `Implemented` 參數。 • `--excel-file` – Excel 試算表的路徑。預設值為 `aws_config_rules.xlsx` 。	開發人員

任務	Description	所需的技能
	• `--param-file` – 參數 JSON 檔案的路徑。預設值為 `config_rules_params.json` 。 • `--max-execution-frequency` – 定義 AWS Config 受管規則的評估頻率。選項包括 `One_Hour`、`Three_Hours` 、`Twelve_Hours` 、`Six_Hours` 或 `TwentyFour_Hours` 。預設值為 `TwentyFour_Hours` 。	

部署 AWS Config 受管規則

任務	Description	所需的技能
啟動 AWS CloudFormation 堆疊。	1. 登入 AWS 管理主控台，開啟 AWS CloudFormation 主控台，然後選擇建立堆疊。 2. 在指定範本頁面上，選擇上傳範本檔案，然後上傳您的 AWS CloudFormation 範本。 3. 指定堆疊名稱，然後選擇下一步。 4. 指定標籤，然後選擇下一步。 5. 選擇建立堆疊。	開發人員

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

讓 SageMaker 筆記本執行個體暫時存取另一個 AWS 帳戶中的 CodeCommit 儲存庫

Helge Aufderheide，Amazon Web Services

總結

此模式顯示如何授予 Amazon SageMaker 筆記本執行個體和使用者暫時存取另一個 AWS 帳戶中的 AWS CodeCommit 儲存庫。此模式也顯示如何授予每個實體在每個儲存庫上執行之特定動作的精細許可。

組織通常會將 CodeCommit 儲存庫存放在與託管其開發環境的帳戶不同的 AWS 帳戶中。此多帳戶設定有助於控制對儲存庫的存取，並降低意外刪除的風險。若要授予這些跨帳戶許可，最佳實務是使用 AWS Identity and Access Management (IAM) 角色。然後，每個 AWS 帳戶中預先定義的 IAM 身分可以暫時擔任角色，以跨帳戶建立受控制的信任鏈。

> **Note**
>
> 您可以套用類似的程序，將 CodeCommit 儲存庫的跨帳戶存取權授予其他 IAM 身分。如需詳細資訊，請參閱《AWS CodeCommit 使用者指南》中的使用角色設定 AWS CodeCommit 儲存庫的跨帳戶存取權。 AWS CodeCommit

先決條件和限制

先決條件

- 具有 CodeCommit 儲存庫的作用中 AWS 帳戶 (帳戶 A)
- 具有 SageMaker 筆記本執行個體的第二個作用中 AWS 帳戶 (帳戶 B)
- 具備足夠許可的 AWS 使用者，可在帳戶 A 中建立和修改 IAM 角色
- 第二個 AWS 使用者具有足夠的許可，可在帳戶 B 中建立和修改 IAM 角色

Architecture

下圖顯示授予 SageMaker 筆記本執行個體和使用者在一個 AWS 帳戶中跨帳戶存取 CodeCommit 儲存庫的範例工作流程：

該圖顯示以下工作流程：

1. 帳戶 B 中的 AWS 使用者角色和 SageMaker 筆記本執行個體角色擔任具名設定檔。
2. 具名設定檔的許可政策會指定帳戶 A 中設定檔接著擔任的 CodeCommit 存取角色。
3. 帳戶 A 中的 CodeCommit 存取角色信任政策允許帳戶 B 中的具名設定檔擔任 CodeCommit 存取角色。
4. CodeCommit 儲存庫在帳戶 A 中的 IAM 許可政策允許 CodeCommit 存取角色存取 CodeCommit 儲存庫。

技術堆疊

- CodeCommit：
- Git
- IAM
- pip
- SageMaker

工具

- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Git 是一種分散式版本控制系統，可在軟體開發期間追蹤原始程式碼的變更。
- git-remote-codecommit 是一種公用程式，可透過擴展 Git 來協助您從 CodeCommit 儲存庫推送和提取程式碼。
- pip 是 Python 的套件安裝程式。您可以使用 pip 從 Python 套件索引和其他索引安裝套件。

最佳實務

當您使用 IAM 政策設定許可時，請務必僅授予執行任務所需的許可。如需詳細資訊，請參閱 IAM 文件中的套用最低權限許可。

實作此模式時，請務必執行下列動作：

- 確認 IAM 原則只有執行每個儲存庫中特定、必要動作所需的許可。例如，建議允許核准的 IAM 原則推送和合併特定儲存庫分支的變更，但只請求合併到受保護的分支。

- 確認 IAM 原則會根據每個專案各自的角色和責任指派不同的 IAM 角色。例如，開發人員將擁有與發行管理員或 AWS 管理員不同的存取許可。

史詩

設定 IAM 角色

任務	Description	所需的技能
設定 CodeCommit 存取角色和許可政策。	**Note** 若要自動化此史詩中記錄的手動設定程序，您可以使用 AWS CloudFormation 範本。 在包含 CodeCommit 儲存庫 (帳戶 A) 的帳戶中，執行下列動作： 1. 建立可由帳戶 B 中的 SageMaker 筆記本執行個體角色擔任的 IAM 角色。SageMaker 2. 建立授予儲存庫存取權的 IAM 政策，並將政策連接至角色。僅為測試目的，請選擇 AWSCodeCommitPowerUser AWS 受管政策。除了刪除資源的功能之外，此政策會授予所有 CodeCommit 許可。 3. 修改角色的信任政策，讓帳戶 B 列為信任的實體。	一般 AWS、AWS DevOps

任務	Description	所需的技能
	Important 在將此設定移至生產環境之前，最佳實務是撰寫您自己的 IAM 政策，以套用最低權限許可。如需詳細資訊，請參閱此模式的其他資訊一節。	

任務	Description	所需的技能
授予帳戶 B 中的 SageMaker 筆記本執行個體角色許可，以在帳戶 A 中擔任 CodeCommit 存取角色。	在包含 SageMaker 筆記本執行個體 IAM 角色 (帳戶 B) 的帳戶中，執行下列動作： 1. 建立 IAM 政策，允許 IAM 角色或使用者在帳戶 A 中擔任 CodeCommit 存取角色。 允許 IAM 角色或使用者擔任跨帳戶角色的 IAM 許可政策範例 `{ "Version": "2012-10-17", "Statement": [ { "Sid": "VisualEditor0", "Effect": "Allow", "Action": "sts:AssumeRole", "Resource": "arn:aws:iam:::accountA_ID:role/accountArole_ID" } }` 2. 將政策連接至帳戶 B 中 SageMaker 筆記本執行個體的角色。 3. 讓帳戶 B 中的 SageMaker 筆記本執行個體角色擔任帳戶 A 中的 CodeCommit 存取角色。	一般 AWS、AWS DevOps

任務	Description	所需的技能
	Note 若要檢視儲存庫的 Amazon Resource Name (ARN)，請參閱《AWS CodeCommit 使用者指南》中的檢視 CodeCommit 儲存庫詳細資訊。 AWS CodeCommit	

在帳戶 B 中設定 SageMaker 筆記本執行個體

任務	Description	所需的技能
在 AWS SageMaker 筆記本執行個體上設定使用者設定檔，以擔任帳戶 A 中的角色。	**Important** 請確定您已安裝最新版本的 AWS Command Line Interface (AWS CLI)。 在包含 SageMaker 筆記本執行個體 (帳戶 B) 的帳戶中，執行下列動作： 1. 登入 AWS 管理主控台並開啟 SageMaker 主控台。 2. 存取您的 SageMaker 筆記本執行個體。Jupyter 介面隨即開啟。	一般 AWS、AWS DevOps

任務	Description	所需的技能
	3. 選擇新增，然後選擇終端機。新的終端機視窗會在您的 Jupyter 環境中開啟。 4. 導覽至 SageMaker 筆記本執行個體的 ~/.aws/config 檔案。然後，輸入下列陳述式，將使用者設定檔新增至檔案： `------.aws/config-------------[profile remoterepouser]role_arn = arn:aws:iam::<ID of Account A>:role/<rolename>role_session_name = remoteaccesssessionregion = eu-west-1credential_source = Ec2InstanceMetadata----------------------------------`	
安裝 git-remote-codecommit 公用程式。	請遵循 AWS CodeCommit 使用者指南中的步驟 2：安裝 git-remote-codecommit 中的指示。	資料科學家

存取儲存庫

任務	Description	所需的技能
使用 Git 命令或 SageMaker 存取 CodeCommit 儲存庫。SageMaker	使用 Git 帳戶 B 中擔任 SageMaker 筆記本執行個體角色的 IAM 主體現在可以執行 Git 命令來存取帳戶 A 中的 CodeCommit 儲存庫。例如，使用者可以執行 `git clone`、`git pull`和等命令`git push`。 如需說明，請參閱《AWS CodeCommit 使用者指南》中的連線至 AWS CodeCommit 儲存庫。 AWS CodeCommit 如需如何搭配 CodeCommit 使用 Git 的詳細資訊，請參閱 AWS CodeCommit 使用者指南》中的 AWS CodeCommit 入門。 AWS CodeCommit 使用 SageMaker 若要從 SageMaker 主控台使用 Git，您必須允許 Git 從 CodeCommit 儲存庫擷取登入資料。如需說明，請參閱 SageMaker 文件中的將不同 AWS 帳戶中的 CodeCommit 儲存庫與筆記本執行個體建立關聯。	Git、Bash 主控台

相關資源

- 使用角色設定 AWS CodeCommit 儲存庫的跨帳戶存取權 (AWS CodeCommit 文件)
- IAM 教學課程：使用 IAM 角色在 AWS 帳戶之間委派存取權 (IAM 文件)

其他資訊

限制 CodeCommit 對特定動作的許可

若要限制 IAM 主體可以在 CodeCommit 儲存庫中採取的動作，請修改 CodeCommit 存取政策中允許的動作。

如需 CodeCommit API 操作的詳細資訊，請參閱《AWS CodeCommit 使用者指南》中的 CodeCommit 許可參考。 AWS CodeCommit

> **Note**
>
> 您也可以編輯 AWSCodeCommitPowerUser AWS 受管政策，以符合您的使用案例。

限制 CodeCommit 對特定儲存庫的許可

若要建立僅供特定使用者存取多個程式碼儲存庫的多租戶環境，請執行下列動作：

1. 在帳戶 A 中建立多個 CodeCommit 存取角色。然後，設定每個存取角色的信任政策，以允許帳戶 B 中的特定使用者擔任該角色。
2. 將「資源」條件新增至每個 CodeCommit 存取角色的政策，以限制每個角色可以擔任的程式碼儲存庫。

限制 IAM 主體存取特定 CodeCommit 儲存庫的「資源」條件範例

```
"Resource" : [<REPOSITORY_ARN>,<REPOSITORY_ARN> ]
```

> **Note**
>
> 為了協助識別和區分相同 AWS 帳戶中的多個程式碼儲存庫，您可以將不同的字首指派給儲存庫的名稱。例如，您可以使用符合不同開發人員群組的字首來命名程式碼儲存庫，例如 myproject-subproject1-repo1 和 myproject-subproject2-repo1。然後，您可以根據每個

開發人員群組指派的字首來建立 IAM 角色。例如，您可以建立名為 myproject-subproject1-repoaccess 的角色，並授予它存取包含 myproject-subproject1 字首的所有程式碼儲存庫。

參考包含特定字首的程式碼儲存庫 ARN 的「資源」條件範例

```
"Resource" : arn:aws:codecommit:<region>:<account-id>:myproject-subproject1-*
```

為多帳戶 DevOps 環境實作 GitHub 流程分支策略

Mike Stephens 和 Abhilash Vinod，Amazon Web Services

總結

管理原始程式碼儲存庫時，不同的分支策略會影響開發團隊使用的軟體開發和發行程序。常見的分支策略範例包括主體、GitHub Flow 和 Gitflow。這些策略使用不同的分支，而且每個環境中執行的活動都不同。實作 DevOps 程序的組織將受益於視覺化指南，以協助他們了解這些分支策略之間的差異。在您的組織中使用此視覺效果有助於開發團隊協調工作並遵循組織標準。此模式提供此視覺化效果，並說明在您的組織中實作 GitHub Flow 分支策略的程序。

此模式是文件系列的一部分，旨在為具有多個 的組織選擇和實作 DevOps 分支策略 AWS 帳戶。此系列旨在協助您從一開始就套用正確的策略和最佳實務，以簡化雲端體驗。GitHub Flow 只是您的組織可以使用的一個可能分支策略。此文件系列也涵蓋了主體和 Gitflow 分支模型。如果您尚未這麼做，我們建議您在實作此模式中的指引之前，先檢閱為多帳戶 DevOps 環境選擇 Git 分支策略。請使用盡職調查來為您的組織選擇正確的分支策略。

本指南提供圖表，說明組織如何實作 GitHub 流程策略。建議您檢閱 AWS Well-Architected DevOps 指南，以檢閱最佳實務。此模式包含 DevOps 程序中每個步驟的建議任務、步驟和限制。

先決條件和限制

先決條件

- Git，已安裝。這用作原始程式碼儲存庫工具。
- Draw.io，已安裝。此應用程式用於檢視和編輯圖表。

Architecture

目標架構

下圖可以像 Punnett 方形（維基百科）一樣使用。您可以將垂直軸上的分支與水平軸上的 AWS 環境對齊，以決定在每個案例中要執行的動作。這些數字表示工作流程中動作的序列。此範例會帶您從`feature`分支到生產環境中的部署。

如需 GitHub Flow 方法中 AWS 帳戶、 環境和分支的詳細資訊，請參閱為多帳戶 DevOps 環境選擇 Git 分支策略。

自動化和擴展

持續整合和持續交付 (CI/CD) 是自動化軟體版本生命週期的程序。它會自動化傳統上需要的許多或所有手動程序，從初始遞交取得新的程式碼到生產環境。CI/CD 管道包含沙盒、開發、測試、預備和生產環境。在每個環境中，CI/CD 管道會佈建部署或測試程式碼所需的任何基礎設施。透過使用 CI/CD，開發團隊可以對程式碼進行變更，然後自動測試和部署。CI/CD 管道也透過強制執行一致性、標準、最佳實務和最低接受度來為開發團隊提供控管和防護。如需詳細資訊，請參閱實作持續整合和持續交付 AWS。

AWS 提供一套開發人員服務，旨在協助您建置 CI/CD 管道。例如， AWS CodePipeline是一種全受管的持續交付服務，可協助您自動化發行管道，以取得快速且可靠的應用程式和基礎設施更新。 會AWS CodeBuild編譯原始程式碼、執行測試，並產生ready-to-deploy的軟體套件。如需詳細資訊，請參閱 上的開發人員工具 AWS。

工具

AWS 服務和工具

AWS 提供一套開發人員服務，您可以用來實作此模式：

- AWS CodeArtifact 是一種高度可擴展的受管成品儲存庫服務，可協助您存放和共用應用程式開發的軟體套件。
- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeDeploy 會自動部署到 Amazon Elastic Compute Cloud (Amazon EC2) 或內部部署執行個體、 AWS Lambda 函數或 Amazon Elastic Container Service (Amazon ECS) 服務。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。

其他工具

- Draw.io Desktop 是製作流程圖和圖表的應用程式。程式碼儲存庫包含 Draw.io 的 .drawio 格式範本。
- Figma 是一種線上設計工具，專為協同合作而設計。程式碼儲存庫包含 Figma 的 .fig 格式範本。

程式碼儲存庫

此模式中圖表的這個來源檔案可在 GitHub Flow 儲存庫的 GitHub Git 分支策略中使用。它包含 PNG、draw.io 和 Figma 格式的檔案。您可以修改這些圖表以支援組織的程序。

最佳實務

遵循 AWS Well-Architected DevOps 指南中的最佳實務和建議，並為多帳戶 DevOps 環境選擇 Git 分支策略。這些可協助您有效地實作 GitHub Flow 型開發、促進協作、改善程式碼品質，以及簡化開發程序。

史詩

檢閱 GitHub 流程工作流程

任務	Description	所需的技能
檢閱標準 GitHub 流程。	1. 在沙盒環境中，開發人員會從`feature`分支建立`main`分支，並使用命名模式 `feature/<ticket>_<initials>_<short description>` 。 2. 開發人員將一或多個遞交新增至`feature`分支，每個遞交代表不同的變更或改進。 3. 開發人員會開啟合併請求 (MR)，將變更合併到`main`分支。這會啟動檢閱程序。 4. 在檢閱過程中，開發人員會討論程式碼變更並提供意見回饋。目標是確保變更具有高品質，並符合專案的標準。 5. 開發人員建立合併請求後，自動化建置程序會開始並將`feature`分支中的變更部署到開發環境。 6. 自動化測試會驗證合併請求中封裝的變更完整性和品	DevOps 工程師

任務	Description	所需的技能
	質。完成合併請求需要成功建置、成功部署和成功測試。 7. 檢閱程序完成時，變更會合併到main分支中。 8. 核准者手動核准將發行成品部署到測試環境。 9. 核准者手動核准將發行成品部署到預備環境。 10核准者手動核准將發行成品部署到生產環境。	
檢閱錯誤修正 GitHub 流程程序。	1. 開發人員會從bugfix分支建立main分支，並使用命名模式 bugfix/<ticket number>_<developer initials>_<descriptor> 。 2. 開發人員會修正問題、遞交修正，以及建置bugfix分支。 3. 開發人員會開啟合併請求，將bugfix分支合併到main分支。這會啟動檢閱程序。 4. 在檢閱過程中，開發人員會討論程式碼變更並提供意見回饋。 5. 審核完成並核准後，開發人員會完成bugfix分支合併請求至main分支。 6. 核准者會手動核准將發行成品部署到更高的環境。	DevOps 工程師

任務	Description	所需的技能
檢閱 Hotfix GitHub Flow 程序。	GitHub Flow 旨在啟用持續交付，其中程式碼變更經常且可靠地部署到更高的環境。關鍵在於，每個`feature`分支都可以隨時部署。 Hotfix 分支與 `feature`或 `bugfix` 分支類似，可以遵循與這些其他分支相同的程序。不過，考慮到其緊迫性，修正程式通常具有較高的優先順序。根據團隊的政策和情況的緊迫性，程序中的某些步驟可能會加快。例如，可能會快速追蹤 Hotfix 的程式碼檢閱。因此，當 Hotfix 程序平行處理特徵或錯誤修正程序時，圍繞 Hotfix 的緊迫性可能需要修改程序遵循。請務必建立有關管理 Hotfix 的準則，以確保有效且安全地處理它們。	DevOps 工程師

疑難排解

問題	解決方案
分支衝突	GitHub Flow 模型可能發生的常見問題是，Hotfix 需要發生在生產環境中`feature`，但對應的變更需要發生在正在修改相同資源的 `bugfix`、或 `hotfix`分支中。我們建議您經常將變更從 合併`main`到較低的分支，以避免合併到 時發生重大衝突`main`。

問題	解決方案
團隊成熟度	GitHub Flow 鼓勵每日部署到更高的環境，採用真正的持續整合和持續交付 (CI/CD)。團隊必須具備工程成熟度，才能建置功能並為其建立自動化測試。團隊必須先執行詳盡的合併請求檢閱，才能核准變更。這可培養強大的工程文化，在開發過程中提升品質、責任和效率。

相關資源

本指南不包含 Git 的訓練；不過，如果您需要此訓練，網際網路上有許多可用的高品質資源。我們建議您從 Git 文件網站開始。

下列資源可協助您在 中完成 GitHub Flow 分支旅程 AWS 雲端。

AWS DevOps 指引

- AWS DevOps 指引
- AWS 部署管道參考架構
- 什麼是 DevOps？
- DevOps 資源

GitHub 流程指引

- GitHub 流程快速入門教學課程 (GitHub)
- 為什麼選擇 GitHub 流程？

其他資源

- 十二因素應用程式方法 (12factor.net：//)

為多帳戶 DevOps 環境實作 Gitflow 分支策略

Mike Stephens、Stephen DiCato、Abhilash Vinod 和 Tim Wondergem，Amazon Web Services

總結

管理原始程式碼儲存庫時，不同的分支策略會影響開發團隊使用的軟體開發和發行程序。常見的分支策略範例包括主體、Gitflow 和 GitHub Flow。這些策略使用不同的分支，而且每個環境中執行的活動都不同。實作 DevOps 程序的組織將受益於視覺化指南，以協助他們了解這些分支策略之間的差異。在您的組織中使用此視覺效果有助於開發團隊協調工作並遵循組織標準。此模式提供此視覺化效果，並說明在組織中實作 Gitflow 分支策略的程序。

此模式是文件系列的一部分，旨在為具有多個 的組織選擇和實作 DevOps 分支策略 AWS 帳戶。此系列旨在協助您從一開始就套用正確的策略和最佳實務，以簡化雲端體驗。Gitflow 只是您的組織可以使用的一個可能分支策略。此文件系列也涵蓋了主體和 GitHub 流程分支模型。如果您尚未這麼做，我們建議您在實作此模式中的指引之前，先檢閱多帳戶 DevOps 環境的 Git 分支策略。請使用盡職調查來為您的組織選擇正確的分支策略。

本指南提供圖表，顯示組織如何實作 Gitflow 策略。建議您檢閱 AWS Well-Architected DevOps 指南，以檢閱最佳實務。此模式包含 DevOps 程序中每個步驟的建議任務、步驟和限制。

先決條件和限制

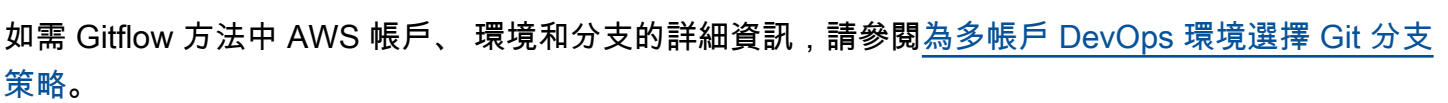
先決條件

- Git，已安裝。這用作原始程式碼儲存庫工具。
- Draw.io，已安裝。此應用程式用於檢視和編輯圖表。
- （選用）Gitflow 外掛程式，已安裝。

Architecture

目標架構

下圖可以像 Punnett 方形（維基百科）一樣使用。您可以將垂直軸上的分支與水平軸上的 AWS 環境對齊，以決定在每個案例中要執行的動作。這些數字表示工作流程中動作的序列。此範例會帶您從功能分支到生產環境中的部署。



如需 Gitflow 方法中 AWS 帳戶、 環境和分支的詳細資訊，請參閱為多帳戶 DevOps 環境選擇 Git 分支策略。

自動化和擴展

持續整合和持續交付 (CI/CD) 是自動化軟體版本生命週期的程序。它會自動化傳統上需要的許多或所有手動程序，以便從初始遞交取得新程式碼到生產環境。CI/CD 管道包含沙盒、開發、測試、預備和生產環境。在每個環境中，CI/CD 管道會佈建部署或測試程式碼所需的任何基礎設施。透過使用 CI/CD，開發團隊可以對程式碼進行變更，然後自動測試和部署。CI/CD 管道也透過強制執行一致性、標準、最佳實務和最低接受水準，為開發團隊提供控管和防護。如需詳細資訊，請參閱實作持續整合和持續交付 AWS。

AWS 提供一套開發人員服務，旨在協助您建置 CI/CD 管道。例如， AWS CodePipeline是一種全受管的持續交付服務，可協助您自動化發行管道，以取得快速可靠的應用程式和基礎設施更新。 會AWS CodeBuild編譯原始程式碼、執行測試，並產生ready-to-deploy的軟體套件。如需詳細資訊，請參閱 上的開發人員工具 AWS。

工具

AWS 服務和工具

AWS 提供一套開發人員服務，您可以用來實作此模式：

- AWS CodeArtifact 是一種高度可擴展的受管成品儲存庫服務，可協助您存放和共用應用程式開發的軟體套件。
- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeDeploy 會自動部署到 Amazon Elastic Compute Cloud (Amazon EC2) 或內部部署執行個體、 AWS Lambda 函數或 Amazon Elastic Container Service (Amazon ECS) 服務。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。

其他工具

- Draw.io Desktop 是製作流程圖和圖表的應用程式。程式碼儲存庫包含 Draw.io 的 .drawio 格式範本。
- Figma 是一種線上設計工具，專為協同合作而設計。程式碼儲存庫包含 Figma 的 .fig 格式範本。
- （選用） Gitflow 外掛程式是一組 Git 延伸模組，可為 Gitflow 分支模型提供高階儲存庫操作。

程式碼儲存庫

此模式中圖表的此來源檔案可在 GitHub Git 分支策略的 GitFlow 儲存庫中使用。它包含 PNG、draw.io 和 Figma 格式的檔案。您可以修改這些圖表以支援組織的程序。

最佳實務

遵循 AWS Well-Architected DevOps 指南中的最佳實務和建議，並為多帳戶 DevOps 環境選擇 Git 分支策略。這些可協助您有效地實作 Gitflow 型開發、促進協作、改善程式碼品質，以及簡化開發程序。

史詩

檢閱 Gitflow 工作流程

任務	Description	所需的技能
檢閱標準 Gitflow 程序。	1. 在沙盒環境中，開發人員會從`feature`分支建立`develop`分支，並使用命名模式 `feature/<ticket>_<initials>_<short description>` 。 2. 開發人員開發程式碼，並以反覆方式將程式碼部署到沙盒環境，以完成票證。 **Note** 開發人員可以選擇性地建立`sandbox`分支，以在沙盒環境中執行自動化建置或部署管道。 3. 開發人員使用 squash 合併，從`feature`分支建立合併請求到`develop`分支。 4. 持續整合和持續交付 (CI/CD) 管道會自動建	DevOps 工程師

任務	Description	所需的技能
	置develop分支並將其部署到開發環境。 5. （選用）在繼續發行活動之前，開發人員會將其他feature分支整合到開發分支中。 6. 當您準備好發行develop分支中的功能時，開發人員release/v<number> 會從release分支建立名為的develop分支。 7. 開發人員會建置發行分支，該分支會發佈成品，以便在其他環境中重複使用。 8. 核准者手動核准將發行成品部署到測試環境。 9. 核准者手動核准將發行成品部署到預備環境。 10.核准者手動核准將發行成品部署到生產環境。 11.開發人員會將release分支合併到main分支中。在理想情況下，開發人員會使用自動化指令碼來執行快速向前合併。請勿使用小隊合併。 12.開發人員會將release分支合併到develop分支中。在理想情況下，開發人員會使用自動化指令碼來執行快速向前合併。請勿使用小隊合併。	

任務	Description	所需的技能
檢閱 Hotfix Gitflow 程序。	1. 開發人員會從hotfix分支建立main分支，並使用命名模式 hotfix/<ticket>_<initials>_<short description> 。 2. 開發人員會從release分支建立main分支，並將其命名為 release/v<number> 。 3. 開發人員會修正問題、遞交修正，以及建置hotfix分支。 4. 開發人員使用 squash 合併，從hotfix分支建立合併請求到release/v<number> 分支。 5. 開發人員會建置release分支，發佈成品，以便在其他環境中重複使用。 6. 核准者手動核准將發行成品部署到測試環境。 7. 核准者手動核准將發行成品部署到預備環境。 8. 核准者手動核准將發行成品部署到生產環境。 9. 開發人員會將release分支合併到main分支中。在理想情況下，開發人員會使用自動化指令碼來執行快速向前合併。請勿使用小隊合併。	DevOps 工程師

任務	Description	所需的技能
	10.開發人員會將`release`分支合併到`develop`分支中。在理想情況下，開發人員會使用自動化指令碼來執行快速向前合併。請勿使用小隊合併。 11.如果偵測到衝突，開發人員會收到提醒，並解決與合併請求的衝突。	

任務	Description	所需的技能
檢閱 bugfix Gitflow 程序。	1. 開發人員會從目前的bugfix分支建立release/v<number> 分支，並使用命名模式 bugfix/<ticket number>_<developer initials>_<descriptor> 。 2. 開發人員會修正問題、遞交修正，以及建置bugfix分支。 3. 開發人員使用 squash 合併，從bugfix分支建立合併請求到release/v<number> 分支。 4. 開發人員會建置release分支，發佈成品以在其他環境中重複使用。 5. 核准者手動核准將發行成品部署到測試環境。 6. 核准者手動核准將發行成品部署到階段環境。 7. 核准者手動核准將發行成品部署到生產環境。 8. 開發人員將release分支合併到main分支中。在理想情況下，開發人員會使用自動化指令碼來執行快速向前合併。請勿使用小隊合併。 9. 開發人員將release分支合併到develop分支中。在理想情況下，開發人員會使用自動化指令碼來執行快速	DevOps 工程師

任務	Description	所需的技能
	向前合併。請勿使用小隊合併。 10.如果偵測到衝突，開發人員會收到提醒並解決與合併請求的衝突。	

疑難排解

問題	解決方案
分支衝突	Gitflow 模型可能發生的常見問題是，Hotfix 需要在生產環境中發生，但對應的變更需要在較低的環境中發生，其中另一個分支正在修改相同的資源。我們建議您一次只啟用一個發行分支。如果您一次有一個以上的作用中，環境中的變更可能會碰撞，而且可能無法將分支向前移至生產環境。
合併	版本應合併回主要分支，並盡快開發，以將工作合併回主要分支。
Squash 合併	只有在從`feature`分支合併到`develop`分支時，才使用小隊合併。在較高分支中使用小隊合併會導致合併變更回到較低分支時遇到困難。

相關資源

本指南不包含 Git 的訓練；不過，如果您需要此訓練，網際網路上有許多可用的高品質資源。我們建議您從 Git 文件網站開始。

下列資源可協助您在 中完成 Gitflow 分支旅程 AWS 雲端。

AWS DevOps 指引

- AWS DevOps 指引

- AWS 部署管道參考架構
- 什麼是 DevOps？
- DevOps 資源

Gitflow 指引

- 原始 Gitflow 部落格 (Vincent Driessen 部落格文章）
- Gitflow 工作流程 (Atlassian)
- GitHub 上的 Gitflow：如何使用 Git Flow 工作流程搭配 GitHub 型儲存庫 (YouTube 影片）
- Git 流程初始化範例 (YouTube 影片）
- 從開始到結束的 Gitflow 發行分支 (YouTube 影片）

其他資源

十二因素應用程式方法 (12factor.net：//)

實作多帳戶 DevOps 環境的主體分支策略

Mike Stephens 和 Rayjan Wilson，Amazon Web Services

總結

管理原始程式碼儲存庫時，不同的分支策略會影響開發團隊使用的軟體開發和發行程序。常見的分支策略範例包括主體、GitHub Flow 和 Gitflow。這些策略使用不同的分支，而且每個環境中執行的活動都不同。實作 DevOps 程序的組織將受益於視覺化指南，以協助他們了解這些分支策略之間的差異。在您的組織中使用此視覺效果有助於開發團隊協調工作並遵循組織標準。此模式提供此視覺效果，並說明在組織中實作中繼線分支策略的程序。

此模式是文件系列的一部分，旨在為具有多個 的組織選擇和實作 DevOps 分支策略 AWS 帳戶。此系列旨在協助您從一開始就套用正確的策略和最佳實務，以簡化雲端體驗。中繼線只是您的組織可以使用的一個可能分支策略。此文件系列也涵蓋 GitHub Flow 和 Gitflow 分支模型。如果您尚未這麼做，建議您在實作此模式中的指引之前，先檢閱多帳戶 DevOps 環境的 Git 分支策略。請使用盡職調查來為您的組織選擇正確的分支策略。

本指南提供圖表，說明組織如何實作主體策略。建議您檢閱正式的 AWS Well-Architected DevOps 指南，以檢閱最佳實務。此模式包含 DevOps 程序中每個步驟的建議任務、步驟和限制。

先決條件和限制

先決條件

- Git，已安裝。這用作原始程式碼儲存庫工具。
- Draw.io，已安裝。此應用程式用於檢視和編輯圖表。

Architecture

目標架構

下圖可以像 Punnett 方形（維基百科）一樣使用。您可以將垂直軸上的分支與水平軸上的 AWS 環境對齊，以決定在每個案例中要執行的動作。這些數字表示工作流程中動作的序列。此範例會帶您從`feature`分支到生產環境中的部署。

如需 Trunk 方法中 AWS 帳戶、 環境和分支的詳細資訊，請參閱為多帳戶 DevOps 環境選擇 Git 分支策略。

自動化和擴展

持續整合和持續交付 (CI/CD) 是自動化軟體版本生命週期的程序。它會自動化傳統上所需的許多或所有手動程序，以便從初始遞交取得新程式碼到生產環境。CI/CD 管道包含沙盒、開發、測試、預備和生產環境。在每個環境中，CI/CD 管道會佈建部署或測試程式碼所需的任何基礎設施。透過使用 CI/CD，開發團隊可以對程式碼進行變更，然後自動測試和部署。CI/CD 管道也透過強制執行一致性、標準、最佳實務和最低接受度來為開發團隊提供控管和防護。如需詳細資訊，請參閱實作持續整合和持續交付 AWS。

AWS 提供一套開發人員服務，旨在協助您建置 CI/CD 管道。例如， AWS CodePipeline是一種全受管的持續交付服務，可協助您自動化發行管道，以取得快速且可靠的應用程式和基礎設施更新。 會AWS CodeBuild編譯原始程式碼、執行測試，並產生ready-to-deploy的軟體套件。如需詳細資訊，請參閱 上的開發人員工具 AWS。

工具

AWS 服務和工具

AWS 提供一套開發人員服務，您可以用來實作此模式：

- AWS CodeArtifact 是一種高度可擴展的受管成品儲存庫服務，可協助您存放和共用應用程式開發的軟體套件。
- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeDeploy 會自動部署到 Amazon Elastic Compute Cloud (Amazon EC2) 或內部部署執行個體、 AWS Lambda 函數或 Amazon Elastic Container Service (Amazon ECS) 服務。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。

其他工具

- Draw.io Desktop – 用於製作流程圖和圖表的應用程式。
- Figma 是一種線上設計工具，專為協同合作而設計。程式碼儲存庫包含 Figma 的 .fig 格式範本。

程式碼儲存庫

此模式中圖表的此來源檔案可在適用於主體儲存庫的 GitHub Git 分支策略中取得。 https://github.com/awslabs/git-branching-strategies-for-multiaccount-devops/tree/main/trunk它包含 PNG、draw.io 和 Figma 格式的檔案。您可以修改這些圖表以支援組織的程序。

最佳實務

遵循 AWS Well-Architected DevOps 指南中的最佳實務和建議，並為多帳戶 DevOps 環境選擇 Git 分支策略。這些可協助您有效地實作以主體為基礎的開發、促進協作、改善程式碼品質，以及簡化開發程序。

史詩

檢閱中繼線工作流程

任務	Description	所需的技能
檢閱標準中繼線程序。	1. 在沙盒環境中，開發人員會從`feature``main`分支建立分支，並使用命名模式 `feature/<ticket>_<initials>_<short description>` 。 2. 開發人員會反覆開發程式碼並將程式碼部署到沙盒環境，以完成票證。 **Note** 開發人員可以選擇性地建立`sandbox`分支，以在沙盒環境中執行自動化建置或部署管道。 3. 開發人員使用 squash 合併，從`feature`分支建立合併請求到`main`分支。 4. 持續整合和持續交付 (CI/CD) 管道會自動建置成品，並將成品從`main`分支發佈到開發環境。	DevOps 工程師

任務	Description	所需的技能
	5. 核准者手動核准將發行成品部署到開發環境。 6. 核准者手動核准將發行成品部署到測試環境。 7. 核准者手動核准將發行成品部署到預備環境。 8. 核准者手動核准將發行成品部署到生產環境。	

疑難排解

問題	解決方案
分支衝突	中繼線模型可能發生的常見問題是在生產環境中需要執行修正，但在正在修改相同資源的`feature`分支中需要發生對應的變更。我們建議您經常將變更從 合併`main`到較低的分支，以避免合併到 時發生重大衝突`main`。

相關資源

本指南不包含 Git 的訓練；不過，如果您需要此訓練，網際網路上有許多可用的高品質資源。我們建議您從 Git 文件網站開始。

下列資源可協助您在 中完成主體分支旅程 AWS 雲端。

AWS DevOps 指引

- AWS DevOps 指引
- AWS 部署管道參考架構
- 什麼是 DevOps？
- DevOps 資源

中繼線指引

- 以中繼線為基礎的開發

其他資源

- 十二因素應用程式方法 (12factor.net：//)

實作集中式自訂 Checkov 掃描，以在部署 AWS 基礎設施之前強制執行政策

Benjamin Morris，Amazon Web Services

總結

此模式提供 GitHub Actions 架構，可在一個儲存庫中撰寫自訂 Checkov 政策，以便在 GitHub 組織中重複使用。透過遵循此模式，資訊安全團隊可以根據公司要求撰寫、新增和維護自訂政策。自訂政策可以自動提取到 GitHub 組織中的所有管道。此方法可用於在資源部署之前強制執行公司資源標準。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 使用 GitHub 動作的 GitHub 組織
- AWS 使用 HashiCorp Terraform 或 部署的 基礎設施 AWS CloudFormation

限制

- 此模式適用於 GitHub 動作。不過，它可以適應類似的持續整合和持續交付 (CI/CD) 架構，例如 GitLab。不需要特定版本的 GitHub。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱 AWS 文件中的服務端點和配額，然後選擇服務的連結。

Architecture

此模式旨在部署為 GitHub 儲存庫，其中包含 GitHub 可重複使用的工作流程和自訂 Checkov 政策。可重複使用的工作流程可以將 Terraform 和 CloudFormation 基礎設施掃描為程式碼 (IaC) 儲存庫。

下圖以個別圖示顯示可重複使用的 GitHub 工作流程儲存庫和自訂 Checkov 政策儲存庫。不過，您可以將這些儲存庫實作為個別儲存庫或單一儲存庫。範例程式碼使用單一儲存庫，其中包含工作流程 (`.github/workflows`) 的檔案，以及相同儲存庫中自訂政策 `.checkov.yml` (`custom_policies` 資料夾和組態檔案） 的檔案。

該圖顯示以下工作流程：

1. 使用者在 GitHub 儲存庫中建立提取請求。

2. 管道工作流程從 GitHub 動作開始，包括對 Checkov 可重複使用工作流程的參考。
3. 管道工作流程會從外部儲存庫下載參考的 Checkov 可重複使用工作流程，並使用 GitHub 動作執行該 Checkov 工作流程。
4. Checkov 可重複使用工作流程會從外部儲存庫下載自訂政策。
5. Checkov 可重複使用工作流程會根據內建和自訂 Checkov 政策來評估 GitHub 儲存庫中的 IaC。Checkov 可重複使用工作流程會根據是否發現安全問題而通過或失敗。

自動化和擴展

此模式允許對 Checkov 組態進行集中管理，以便可以在一個位置套用政策更新。不過，此模式確實要求每個儲存庫使用包含中央可重複使用工作流程參考的工作流程。您可以手動新增此參考，或使用指令碼將檔案推送到每個儲存庫的 `.github/workflows` 資料夾。

工具

AWS 服務

- CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和 區域的生命週期中管理資源。Checkov 可以掃描 CloudFormation。

其他工具

- Checkov 是一種靜態程式碼分析工具，可檢查 IaC 的安全性和合規性設定錯誤。
- GitHub Actions 已整合至 GitHub 平台，協助您在 GitHub 儲存庫中建立、共用和執行工作流程。您可以使用 GitHub 動作來自動化任務，例如建置、測試和部署程式碼。
- Terraform 是 HashiCorp 的 IaC 工具，可協助您建立和管理雲端和內部部署資源。Checkov 可以掃描 Terraform。

程式碼儲存庫

此模式的程式碼可在 GitHub centralized-custom-checkov-sast 儲存庫中使用。

最佳實務

- 若要維持一致的安全狀態，請讓公司的安全政策與 Checkov 政策保持一致。
- 在實作 Checkov 自訂政策的早期階段，您可以使用 Checkov 掃描中的軟失敗選項，以允許合併具有安全問題的 IaC。隨著程序的成熟，從軟失敗選項切換到硬失敗選項。

史詩

建立自訂政策的中央 Checkov 儲存庫

任務	Description	所需的技能
建立中央 Checkov 儲存庫。	建立儲存庫以存放將在組織內使用的自訂 Checkov 政策。 為了快速入門，您可以將此模式的 GitHub centralized-custom-checkov-sast 儲存庫的內容複製到中央 Checkov 儲存庫。	DevOps 工程師
建立可重複使用工作流程的儲存庫。	如果可重複使用工作流程的儲存庫已存在，或者您計劃在與自訂 Checkov 政策相同的儲存庫中包含可重複使用的工作流程檔案，您可以略過此步驟。 建立 GitHub 儲存庫以保留可重複使用的工作流程。其他儲存庫的管道將參考此儲存庫。	DevOps 工程師

建立可重複使用和範例 Checkov 工作流程

任務	Description	所需的技能
新增可重複使用的 Checkov 工作流程。	在可重複使用的工作流程儲存庫中建立可重複使用的 Checkov GitHub 動作工作流程 (YAML 檔案）。您可以從此模式提供的工作流程檔案中調整此可重複使用的工作流程。 您可能想要進行的變更範例是將可重複使用的工作流程變	DevOps 工程師

任務	Description	所需的技能
	更為使用軟失敗選項。`soft-fail` 將 設定為 `true` 可讓任務順利完成，即使 Checkov 掃描失敗也一樣。如需說明，請參閱 Checkov 文件中的硬性與軟性故障。	
新增工作流程範例。	新增參考工作流程的範例 Checkov `reusable`工作流程。這將提供範本，說明如何重複使用`reusable`工作流程。在範例儲存庫中，`checkov-source.yaml` 是可重複使用的工作流程，而 `checkov-scan.yaml` 是使用 的範例`checkov-source` 。 如需撰寫範例 Checkov 工作流程的詳細資訊，請參閱其他資訊。	DevOps 工程師

將公司政策與 Checkov 自訂政策建立關聯

任務	Description	所需的技能
決定可使用 Checkov 強制執行的政策。	1. 檢閱與基礎設施安全相關的公司政策，以及應具備哪些要求。 2. 決定可以使用 Checkov 自訂政策實作哪些需求。 3. 建立將政策控制項映射至 Checkov 自訂政策的命名慣例。一般而言，Checkov 自	安全與合規

任務	Description	所需的技能
	訂政策具有識別符，其中包含 Checkov 名稱、政策來源（自訂）和政策編號（例如 `CKV2_CUSTOM_123`）。 如需建立 Checkov 自訂政策的詳細資訊，請參閱 Checkov 文件中的自訂政策概觀。	
新增 Checkov 自訂政策。	將已識別的公司政策轉換為中央儲存庫中的自訂 Checkov 政策。您可以在 Python 或 YAML 中撰寫簡單的 Checkov 政策。	安全

實作集中式 Checkov 自訂政策

任務	Description	所需的技能
將 Checkov 可重複使用工作流程新增至所有儲存庫。	此時，您應該有一個參考可重複使用工作流程的範例 Checkov 工作流程。將參考可重複使用工作流程的範例 Checkov 工作流程複製到每個需要該工作流程的儲存庫。	DevOps 工程師
建立機制以確保 Checkov 在合併之前執行。	為了確保針對每個提取請求執行 Checkov 工作流程，請建立需要成功 Checkov 工作流程才能合併提取請求的狀態檢查。GitHub 可讓您要求特定工作流程執行，才能合併提取請求。	DevOps 工程師
建立整個組織的 PAT，並將其做為秘密共用。	如果您的 GitHub 組織可公開看見，您可以略過此步驟。	DevOps 工程師

任務	Description	所需的技能
	此模式需要 Checkov 工作流程能夠從 GitHub 組織中的自訂政策儲存庫下載自訂政策。您必須提供許可，讓 Checkov 工作流程可以存取這些儲存庫。 若要這樣做，請建立具有讀取組織儲存庫許可的個人存取字符 (PAT)。將此 PAT 與儲存庫共用，可以是整個組織的秘密（如果在付費計劃中）或每個儲存庫中的秘密（免費版本）。在範例程式碼中，秘密的預設名稱為 `ORG_PAT`。	

任務	Description	所需的技能
（選用）防止 Checkov 工作流程檔案遭到修改。	若要保護 Checkov 工作流程檔案免於不必要的變更，您可以使用 CODEOWNERS 檔案。CODEOWNERS 檔案通常部署在 目錄的根目錄中。 例如，若要在修改checkov-scan.yaml 檔案時要求 GitHub 組織的 secEng 群組核准，請將下列項目附加至儲存庫的 CODEOWNERS 檔案： `[Checkov] .github/workflows /checkov-scan.yaml @myOrg/secEng` CODEOWNERS 檔案專屬於其所在的儲存庫。若要保護儲存庫使用的 Checkov 工作流程，您必須在每個儲存庫中新增（或更新）CODEOWNERS 檔案。 如需保護 Checkov 工作流程檔案的詳細資訊，請參閱其他資訊。如需CODEOWNERS 檔案的詳細資訊，請參閱 CI/CD 提供者的官方文件（例如 GitHub)。	DevOps 工程師

相關資源

- Checkov 自訂政策概觀
- CloudFormation 組態掃描

- GitHub 動作可重複使用的工作流程

其他資訊

撰寫 Checkov 工作流程檔案

寫入 時checkov-scan.yaml，請考慮何時要執行。最上層on金鑰決定工作流程何時執行。在範例儲存庫中，當有以main分支為目標的提取請求時（以及只要修改提取請求的來源分支時），工作流程就會執行。由於 workflow_dispatch金鑰，工作流程也可以視需要執行。

您可以根據您希望工作流程執行的頻率來變更工作流程觸發條件。例如，您可以將 pull_request取代為 push 並移除branches金鑰，將工作流程變更為每次將程式碼推送至任何分支時執行。

您可以修改在個別儲存庫中建立的範例工作流程檔案。例如，production如果儲存庫是圍繞分支建構的，您可以將目標production分支的名稱從 main 調整為 。

保護 Checkov 工作流程檔案

Checkov 掃描提供有關潛在安全錯誤組態的有用資訊。不過，有些開發人員可能會認為這是生產力的障礙，並嘗試移除或停用掃描工作流程。

有幾種方法可以解決此問題，包括更清楚的安全掃描長期價值的訊息，以及更清楚如何部署安全基礎設施的文件。這些是 DevSecOps 協同合作的重要「軟」方法，可視為此問題根本原因的解決方案。不過，您也可以使用像是 CODEOWNERS 檔案之類的技術控制項做為護欄，協助開發人員保持在正確的路徑上。

在沙盒中測試模式

若要在沙盒環境中測試此模式，請遵循下列步驟：

1. 建立新的 GitHub 組織。建立具有組織中所有儲存庫唯讀存取權的字符。由於此字符適用於沙盒環境，而非付費環境，因此您將無法將此字符存放在整個組織的秘密中。
2. 建立儲存checkov庫以保留 Checkov 組態，以及建立儲存github-workflows庫以保留可重複使用的工作流程組態。使用範例儲存庫的內容填入儲存庫。
3. 建立應用程式儲存庫，並將checkov-scan.yaml工作流程複製並貼到其.github/workflows資料夾。將秘密新增至儲存庫，其中包含您為組織唯讀存取建立的 PAT。預設秘密為 ORG_PAT。
4. 建立提取請求，將一些 Terraform 或 CloudFormation 程式碼新增至應用程式儲存庫。Checkov 應掃描並傳回結果。

使用 K8sGPT 和 Amazon Bedrock 整合實作採用 AI 技術的 Kubernetes 診斷和故障診斷

Ishwar Chauthaiwale， Muskan . 和 Amazon Web Services Prafful Gupta

總結

此模式示範如何透過將 K8sGPT 與 Amazon Bedrock 上提供的 Anthropic Claude v2 模型整合，來實作 AI 支援的 Kubernetes 診斷和故障診斷。解決方案透過安全的堡壘主機架構，為 Kubernetes 叢集問題提供自然語言分析和修復步驟。透過將 K8sGPT Kubernetes 專業知識與 Amazon Bedrock 進階語言功能結合，DevOps 團隊可以快速識別和解決叢集問題。透過這些功能，可以將平均解析時間 (MTTR) 減少高達 50%。

這種雲端原生模式利用 Amazon Elastic Kubernetes Service (Amazon EKS) 進行 Kubernetes 管理。模式會透過適當的 AWS Identity and Access Management (IAM) 角色和網路隔離實作安全最佳實務。此解決方案對於想要簡化 Kubernetes 操作並在 AI 協助下增強故障診斷功能的組織特別有用。

先決條件和限制

先決條件

- AWS 帳戶 具有適當許可的作用中
- AWS Command Line Interface (AWS CLI) 已安裝並設定
- Amazon EKS 叢集
- 在 Amazon Bedrock 上存取 Anthropic Claude 2 模型
- 具有必要安全群組設定的堡壘主機
- 已安裝 K8sGPT

限制

- K8sGPT 分析受限於 Claude v2 模型的內容視窗大小。
- Amazon Bedrock API 速率限制會根據您的帳戶配額套用。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

- Amazon EKS 1.31 版或更新版本
- Amazon Bedrock 上的 Claude 2 模型
- K8sGPT 0.4.2 版或更新版本

Architecture

下圖顯示使用與 Amazon Bedrock 整合之 K8sGPT 的 AI 型 Kubernetes 診斷架構 AWS 雲端。

架構顯示下列工作流程：

1. 開發人員透過安全連線至堡壘主機來存取環境。此 Amazon EC2 執行個體做為安全進入點，並包含 K8sGPT 命令列界面 (CLI) 安裝和必要的組態。
2. 使用特定 IAM 角色設定的堡壘主機會建立與 Amazon EKS 叢集和 Amazon Bedrock 端點的安全連線。K8sGPT 已安裝並設定在堡壘主機上，以執行 Kubernetes 叢集分析。
3. Amazon EKS 會管理 Kubernetes 控制平面和工作者節點，提供 K8sGPT 分析的目標環境。此服務會跨虛擬私有雲端 (VPC) 內的多個可用區域執行，這有助於提供高可用性和彈性。Amazon EKS 透過 Kubernetes API 提供操作資料，實現全面的叢集分析。
4. K8sGPT 會將分析資料傳送至 Amazon Bedrock，提供 Claude v2 基礎模型 (FM) 以進行自然語言處理。服務會處理 K8sGPT 分析，以產生人類可讀的說明，並根據已識別的問題提供詳細的修補建議。Amazon Bedrock 以無伺服器 AI 服務的形式運作，具有高可用性和可擴展性。

> **Note**
>
> 在此工作流程中，IAM 透過角色和政策控制元件之間的存取，管理堡壘主機、Amazon EKS 和 Amazon Bedrock 互動的身分驗證。IAM 實作最低權限原則，並在整個架構中啟用安全的跨服務通訊。

自動化和擴展

K8sGPT 操作可以透過各種 和 工具，自動化 AWS 服務 並跨多個 Amazon EKS 叢集進行擴展。此解決方案支援使用 Jenkins、GitHub Actions 或 AWS CodeBuild 進行排程分析的持續整合和持續部署 (CI/CD) 整合。K8sGPT 運算子透過自動化問題偵測和報告功能啟用叢集內持續監控。對於企業規模部署，您可以使用 Amazon EventBridge 來排程掃描，並使用自訂指令碼觸發自動回應。 AWS SDK 整合可在大型叢集機群之間進行程式設計控制。

工具

AWS 服務

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 Shell 中的命令與 互動。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 上執行 Kubernetes， AWS 而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。

其他工具

- K8sGPT 是一種開放原始碼 AI 支援的工具，可轉換 Kubernetes 管理。它充當虛擬站點可靠性工程 (SRE) 專家，自動掃描、診斷和疑難排解 Kubernetes 叢集問題。管理員可以使用自然語言與 K8sGPT 互動，並取得有關叢集狀態、Pod 損毀和服務故障的明確可行洞見。工具的內建分析器可偵測各種問題，從設定錯誤的元件到資源限制，並提供easy-to-understand的說明和解決方案。

最佳實務

- 使用 AWS Systems Manager Session Manager 進行堡壘主機存取，以實作安全存取控制。
- 確定 K8sGPT 身分驗證使用具有最低權限許可的專用 IAM 角色進行 Amazon Bedrock 和 Amazon EKS 互動。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。
- 設定資源標記、啟用稽核線索的 Amazon CloudWatch 記錄，以及實作敏感資訊的資料匿名化。https://docs.aws.amazon.com/awscloudtrail/latest/userguide/monitor-cloudtrail-log-files-with-cloudwatch-logs.html
- 維護 K8sGPT 組態的定期備份，同時在離峰時間設定自動掃描排程，以將操作影響降至最低。

史詩

將 Amazon Bedrock 新增至 AI 後端供應商清單。

任務	Description	所需的技能
將 Amazon Bedrock 設定為 K8sGPT 的 AI 後端提供者。	若要將 Amazon Bedrock 設定為 AI 後端為 K8sGPT 提供 r，	AWS DevOps

任務	Description	所需的技能

	請使用下列 AWS CLI 命令：K8sGPT
	
	`k8sgpt auth add -b amazonbedrock \ -r us-west-2 \ -m anthropic.claude-v2 \ -n endpoint-name`
	
	範例命令會`us-west-2` 針對 使用 AWS 區域。不過，您可以選取另一個區域，前提是 Amazon EKS 叢集和對應的 Amazon Bedrock 模型在該選取的區域中都可用並啟用。
	
	若要檢查 `amazonbedrock` 是否已新增至 AI 後端提供者清單且處於 `Active` 狀態，請執行下列命令：
	
	`k8sgpt auth list`
	
	以下是此命令預期輸出的範例：
	
	`Default: > openai Active: > amazonbedrock Unused: > openai > localai > ollama > azureopenai > cohere > amazonsagemaker`	

任務	Description	所需的技能

	`> google`
	`> noopai`
	`> huggingface`
	`> googlevertexai`
	`> oci`
	`> customrest`
	`> ibmwatsonxai`	

使用篩選條件掃描資源

任務	Description	所需的技能

檢視可用的篩選條件清單。	若要查看所有可用篩選條件的清單，請使用下列 AWS CLI 命令：
	`k8sgpt filters list`
	以下是此命令預期輸出的範例：
	`Active:`
	`> Deployment`
	`> ReplicaSet`
	`> PersistentVolumeCl aim`
	`> Service`
	`> CronJob`
	`> Node`
	`> MutatingWebhookCon figuration`
	`> Pod`
	`> Ingress`
	`> StatefulSet`
	`> ValidatingWebhookC onfiguration`
		AWS DevOps

任務	Description	所需的技能

使用篩選條件掃描特定命名空間中的 Pod。	此命令適用於針對 Kubernetes 叢集內的特定 Pod 問題進行偵錯，使用 Amazon Bedrock AI 功能來分析和解釋其發現的問題。
	
	若要使用篩選條件掃描特定命名空間中的 Pod，請使用下列 AWS CLI 命令：
	
	`k8sgpt analyze --backend amazonbedrock --explain --filter Pod -n default`
	
	以下是此命令預期輸出的範例：
	
	`100% |########## ########## ########## ###| (1/1, 645 it/s)`
	
	`AI Provider: amazonbedrock`
	
	`0: Pod default/crashme()`
	`- Error: the last termination reason is Error container=crashme pod=crashme`
	`Error: The pod named crashme terminated because the container named crashme crashed.`
	`Solution: Check logs for crashme pod to identify reason for`
		AWS DevOps

任務	Description	所需的技能
	`crash. Restart pod or redeploy application to resolve crash.`	

任務	Description	所需的技能

使用篩選條件掃描特定命名空間中的部署。	此命令有助於識別和疑難排解部署特定的問題，特別是當實際狀態不符合所需狀態時。
	
	若要使用篩選條件掃描特定命名空間中的部署，請使用下列 AWS CLI 命令：
	
	`k8sgpt analyze --backend amazonbedrock --explain --filter Deployment -n default`
	
	以下是此命令預期輸出的範例：
	
	`100% |########## ########## ########## #####| (1/1, 10 it/min)`
	
	`AI Provider: amazonbedrock`
	
	`0: Deployment default/nginx()`
	`- Error: Deployment default/nginx has 1 replicas but 2 are available`
	`Error: The Deployment named nginx in the default namespace has 1 replica specified but 2 pod replicas are running.`
	`Solution: Check if any other controllers like ReplicaSet`
		AWS DevOps

任務	Description	所需的技能
	`or StatefulSet have created extra pods. Delete extra pods or adjust replica count to match available pods.`	

使用篩選條件掃描特定命名空間中的節點。	若要使用篩選條件掃描特定命名空間中的節點，請使用下列 AWS CLI 命令：
	`k8sgpt analyze --backend amazonbedrock --explain --filter Node -n default`
	以下是此命令預期輸出的範例：
	`AI Provider: amazonbedrock`
	`No problems detected`
		AWS DevOps

分析詳細輸出

任務	Description	所需的技能
取得詳細輸出。	若要取得詳細輸出，請使用下列 AWS CLI 命令： `k8sgpt analyze --backend amazonbedrock --explain --ouput json` 以下是此命令預期輸出的範例：	AWS DevOps

任務	Description	所需的技能

	{
	"provider": "amazonbedrock",
	"errors": null,
	"status": "ProblemDetected",
	"problems": 1,
	"results": [
	{
	"kind": "Pod",
	"name": "default/crashme",
	"error": [
	{
	"Text": "the last termination reason is Error container=crashme pod=crashme",
	"KubernetesDoc": "",
	"Sensitive": []
	}
	],
	"details": "Error: The pod named crashme terminated because the container named crashme crashed.\nSolution: Check logs for crashme pod to identify reason for crash. Restart pod or redeploy application to resolve crash.",
	"parentObject": ""
	}
	]	

任務	Description	所需的技能
	`}`	

檢查有問題的 Pod。	若要檢查是否有特定有問題的 Pod，請使用下列 AWS CLI 命令：
	`kubectl get pods --all-namespaces | grep -v Running`
	以下是此命令預期輸出的範例：
	`NAMESPACE NAME READY STATUS RESTARTS AGE`
	`default crashme 0/1 CrashLoopBackOff 260(91s ago) 21h`
		AWS DevOps

任務	Description	所需的技能

取得應用程式特定的洞見。	此命令在下列情況下特別有用：
	• 您想要專注於叢集中的特定應用程式。
	• 您可以有效地使用標籤來組織 Kubernetes 資源。
	• 您需要快速檢查特定應用程式元件的運作狀態。
	
	若要取得應用程式特定的洞見，請使用下列命令：
	`k8sgpt analyze --backend amazonbedrock --explain -L app=nginx -n default`
	
	以下是此命令預期輸出的範例：
	`AI Provider: amazonbedrock`
	
	`No problems detected`	

相關資源

AWS 部落格

- 使用 Amazon Bedrock 代理程式工作流程自動化 Amazon EKS 故障診斷
- 使用 K8sGPT 和 Amazon Bedrock 進行簡化的 Kubernetes 叢集維護

AWS 文件

- AWS CLI 命令：create-cluster 和 describe-cluster
- 開始使用 Amazon EKS (Amazon EKS 文件)
- IAM 中的安全最佳實務 (IAM 文件)

其他資源

- K8sGPT

在 CodeCommit 中自動偵測變更並啟動單一儲存庫的不同 CodePipeline 管道 CodeCommit

Helton Ribeiro、Petrus Batalha 和 Amazon Web Services Ricardo Morais

總結

注意： AWS Cloud9 不再提供給新客戶。的現有客戶 AWS Cloud9 可以繼續正常使用服務。進一步了解

此模式可協助您在 中自動偵測單一儲存庫型應用程式的原始碼變更， AWS CodeCommit 然後在 中啟動管道 AWS CodePipeline ，以針對每個微服務執行持續整合和持續交付 (CI/CD) 自動化。此方法表示單一儲存庫型應用程式中的每個微服務都可以有專用的 CI/CD 管道，以確保更佳的可見性、更輕鬆地共用程式碼，並改善協同合作、標準化和可探索性。

此模式中描述的解決方案不會在 monorepo 內的微服務之間執行任何相依性分析。它只會偵測原始程式碼中的變更，並啟動相符的 CI/CD 管道。

模式使用 AWS Cloud9 做為整合式開發環境 (IDE) AWS Cloud Development Kit (AWS CDK) ，並使用兩個 CloudFormation 堆疊定義基礎設施： `MonoRepoStack`和 `PipelinesStack`。`MonoRepoStack` 堆疊會在 中建立單儲存庫， AWS CodeCommit 以及啟動 CI/CD 管道的 AWS Lambda 函數。`PipelinesStack` 堆疊會定義您的管道基礎設施。

> **Important**
>
> 此模式的工作流程是一種概念驗證 (PoC)。建議您只在測試環境中使用它。如果您想要在生產環境中使用此模式的方法，請參閱《 AWS Identity and Access Management (IAM) 文件》中的 IAM 中的安全最佳實務，並對 IAM 角色和 進行必要的變更 AWS 服務。

先決條件和限制

先決條件

- 作用中 AWS 的帳戶。
- AWS Command Line Interface (AWS CLI)，已安裝並設定。如需詳細資訊，請參閱 AWS CLI 文件AWS CLI中的安裝、更新和解除安裝 。
- Python 3 和 `pip`，安裝在本機電腦上。如需詳細資訊，請參閱 Python 文件。

- AWS CDK，已安裝並設定。如需詳細資訊，請參閱 AWS CDK 文件中的開始使用 AWS CDK。
- 安裝和設定的 AWS Cloud9 IDE。如需詳細資訊，請參閱 AWS Cloud9 文件中的設定 AWS Cloud9。
- GitHub AWS CodeCommit monorepo 多管道觸發程式儲存庫，在您的本機電腦上複製。
- 包含您要使用 CodePipeline 建置和部署之應用程式程式碼的現有目錄。
- 具備 DevOps 最佳實務的熟悉度和經驗 AWS 雲端。若要提高您對 DevOps 的熟悉度，您可以使用模式 使用 DevOps 實務和 規範指引網站建置鬆散耦合的架構與微服務 AWS Cloud9。 AWS

Architecture

下圖顯示如何使用 AWS CDK 來定義具有兩個 AWS CloudFormation 堆疊的基礎設施：`MonoRepoStack`和 `PipelinesStack`。

該圖顯示以下工作流程：

1. 引導程序會使用 AWS CDK 來建立 AWS CloudFormation 堆疊 `MonoRepoStack`和 `PipelinesStack`。
2. `MonoRepoStack` 堆疊會為您的應用程式建立 CodeCommit 儲存庫，以及在每次遞交後啟動的 `monorepo-event-handler` Lambda 函數。
3. `PipelinesStack` 堆疊會在 CodePipeline 中建立由 Lambda 函數啟動的管道。每個微服務都必須有定義的基礎設施管道。
4. 的管道由 Lambda 函數`microservice-n`啟動，並啟動以 CodeCommit 中的原始程式碼為基礎的隔離 CI/CD 階段。
5. 的管道由 Lambda 函數`microservice-1`啟動，並啟動以 CodeCommit 中的原始程式碼為基礎的隔離 CI/CD 階段。

下圖顯示 `PipelinesStack`帳戶中 AWS CloudFormation 堆疊 `MonoRepoStack`和 的部署。

1. 使用者變更其中一個應用程式的微服務中的程式碼。
2. 使用者會將變更從本機儲存庫推送到 CodeCommit 儲存庫。
3. 推送活動會啟動接收所有推送至 CodeCommit 儲存庫的 Lambda 函數。
4. Lambda 函數會讀取 參數存放區中的參數，這是 的一項功能 AWS Systems Manager，用於擷取最新的遞交 ID。參數具有命名格式：`/MonoRepoTrigger/{repository}/{branch_name}/`

`LastCommit`。如果找不到 參數，Lambda 函數會從 CodeCommit 儲存庫讀取最後一個遞交 ID，並將傳回的值儲存在參數存放區中。

5. 識別遞交 ID 和變更的檔案之後，Lambda 函數會識別每個微服務目錄的管道，並啟動所需的 CodePipeline 管道。

工具

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，用於在程式碼中定義雲端基礎設施並透過其佈建 CloudFormation。
- Python 是一種程式設計語言，可讓您快速工作並更有效地整合系統。

Code

此模式的原始程式碼和範本可在 GitHub AWS CodeCommit monorepo 多管道觸發程式儲存庫中使用。

最佳實務

- 此範例架構不包含已部署基礎設施的監控解決方案。如果您想要在生產環境中部署此解決方案，建議您啟用監控。如需詳細資訊，請參閱《 AWS Serverless Application Model (AWS SAM) 文件》中的使用 CloudWatch Application Insights 監控無伺服器應用程式。
- 當您編輯此模式提供的範例程式碼時，請遵循 AWS CDK 文件中開發和部署雲端基礎設施的最佳實務。
- 當您定義微服務管道時，請檢閱 AWS CodePipeline 文件中的安全最佳實務。
- 您也可以使用 cdk-nag 公用程式來檢查 AWS CDK 程式碼是否有最佳實務。此工具使用一組依套件分組的規則來評估您的程式碼。可用的套件包括：
  - AWS 解決方案程式庫
  - 健康保險流通與責任法案 (HIPAA) 安全性
  - 國家標準技術研究所 (NIST) 800-53 修訂版 4
  - NIST 800-53 修訂版 5
  - 支付卡產業資料安全標準 (PCI DSS) 3.2.1

史詩

設定環境

任務	Description	所需的技能
建立虛擬 Python 環境。	在您的 AWS Cloud9 IDE 中，透過執行下列命令，建立虛擬 Python 環境並安裝所需的相依性： `make install`	開發人員
AWS 區域 為 引導 AWS 帳戶 和 AWS CDK。	執行下列命令來引導所需的 AWS 帳戶 和 區域： `make bootstrap account-id=<your-AWS-account-ID> region=<required-region>`	開發人員

新增微服務的新管道

任務	Description	所需的技能
將範例程式碼新增至應用程式目錄。	將包含範例應用程式程式碼的目錄新增至複製的 GitHub AWS CodeCommit monorepo 多管道觸發程式儲存庫中的`monorepo-sample` 目錄。	開發人員
編輯 `monorepo-main.json` 檔案。	將應用程式程式碼的目錄名稱和管道名稱新增至複製儲存庫中的 `monorepo-main.json` 檔案。	開發人員

任務	Description	所需的技能
建立管道。	在儲存庫的 `Pipelines` 目錄中，`class`為您的應用程式新增管道。目錄包含兩個範例檔案 `pipeline_hotsite.py` 和 `pipeline_demo.py` 。每個檔案都有三個階段：來源、建置和部署。 您可以複製其中一個檔案，並根據應用程式的需求對其進行變更。	開發人員

任務	Description	所需的技能

編輯 monorepo_config.py 檔案。	在 中service_map ，為您的應用程式新增目錄名稱，以及您為管道建立的類別。
	
	例如，下列程式碼顯示 Pipelines 目錄中的管道定義，該定義使用名為 的檔案pipeline_mysample.py 搭配 MySamplePipeline 類別：
	
	`...`
	`# Pipeline definition imports`
	`from pipelines.pipeline_demo import DemoPipeline`
	`from pipelines.pipeline_hotsite import HotsitePipeline`
	`from pipelines.pipeline_mysample import MySamplePipeline`
	
	`### Add your pipeline configuration here`
	`service_map: Dict[str, ServicePipeline] = {`
	`# folder-name -> pipeline-class`
	`'demo': DemoPipeline(),`
	`'hotsite': HotsitePipeline(),`
	`'mysample': MySamplePipeline()`
	`}`
		開發人員

部署 MonoRepoStack 堆疊

任務	Description	所需的技能
部署 AWS CloudFormation 堆疊。	執行 AWS CloudFormation `MonoRepoStack make deploy-core` 命令，在複製儲存庫的根目錄中部署具有預設參數值的堆疊。 您可以執行 `make deploy-core monorepo-name=<repo_name>` 命令來變更儲存庫的名稱。 **Note** 您可以使用 `make deploy monorepo-name=<repo_name>` 命令同時部署這兩個管道。	開發人員
驗證 CodeCommit 儲存庫。	透過執行 `aws codecommit get-repository --repository-name <repo_name>` 命令來驗證您的資源是否已建立。 **Important** 由於 CloudFormation 堆疊會建立儲存 monorepo 的 CodeCommit 儲存庫，因此如果您已開始將修改推送至其中，請	開發人員

任務	Description	所需的技能
	勿執行 `cdk destroy MonoRepoStack` 命令。	
驗證 CloudFormation 堆疊結果。	執行下列命令，CloudFormation `MonoRepoStack` 驗證堆疊是否已正確建立和設定： `aws cloudformation list-stacks --stack-status-filter CREATE_COMPLETE --query 'StackSummaries[?StackName == 'MonoRepoStack']'`	開發人員

部署 PipelinesStack 堆疊

任務	Description	所需的技能
部署 CloudFormation 堆疊。	部署 AWS CloudFormation `PipelinesStack` 堆疊之後，必須部署`MonoRepoStack` 堆疊。將新的微服務新增至 monorepo 的程式碼基底時，堆疊的大小會增加，並在加入新的微服務時重新部署。 執行 `make deploy-pipelines` 命令來部署 PipelinesStack 堆疊。	開發人員

任務	Description	所需的技能

	Note
	您也可以執行 `make deploy monorepo-name=<repo_name>` 命令，同時部署這兩個管道。
	
	下列範例輸出顯示`Pipelines Stacks` 部署如何在實作結束時列印微服務 URLs：
	
	`Outputs:`
	`PipelinesStack.dem ourl = .cloudfront.net`
	`PipelinesStack.hotsi teurl = .cloudfro nt.net`	

驗證 AWS CloudFormation 堆疊結果。	執行下列命令， AWS CloudFormation `Pipelines Stacks` 驗證堆疊是否已正確建立和設定： `aws cloudformation list-stacks --stack-s tatus-filter CREATE_CO MPLETE UPDATE_COMPLETE --query 'StackSum maries[?StackName == 'PipelinesStack']'`	開發人員

清除資源

任務	Description	所需的技能
刪除您的 AWS CloudFormation 堆疊。	執行 `make destroy` 命令。	開發人員
刪除管道的 S3 儲存貯體。	1. 登入 AWS 管理主控台並開啟 Amazon Simple Storage Service (Amazon S3) 主控台。 2. 刪除與您的管道相關聯的 S3 儲存貯體，並使用下列名稱： `pipelinesstack-codepipeline*`	開發人員

疑難排解

問題	解決方案
我遇到 AWS CDK 問題。	請參閱 AWS CDK 文件中的疑難排解 AWS CDK 常見問題。
我推送了微服務程式碼，但微服務管道未執行。	設定驗證 驗證分支組態： • 請確定您正在將程式碼推送至正確的分支。此管道設定為僅在對`main`分支進行變更時執行。除非特別設定，否則推送至其他分支不會啟動管道。 • 推送程式碼後，請檢查遞交是否在 中可見，AWS CodeCommit 以確保推送成功，以及本機環境與儲存庫之間的連線是否完好無損。如果推送程式碼時發生問題，請重新整理您的登入資料。

問題	解決方案
	驗證組態檔案： • 確認 中的`service_map` 變數`monorepo_config.py` 準確反映微服務目前的目錄結構。此變數在將程式碼推送映射至個別管道時扮演重要角色。 • 請確定 `monorepo-main.json` 已更新，以包含微服務的新映射。此檔案對於管道識別和正確處理微服務的變更至關重要。 在主控台進行故障診斷 AWS CodePipeline 檢查： • 在 上AWS 管理主控台，確認您位於託管管道 AWS 區域 的 中。開啟 CodePipeline 主控台，並檢查與微服務對應的管道是否已啟動。 錯誤分析：如果管道已啟動但失敗，請檢閱 CodePipeline 提供的任何錯誤訊息或日誌，以了解發生了什麼問題。 AWS Lambda 故障診斷： • 在 AWS Lambda 主控台上，開啟 `monorepo-event-handler` Lambda 函數。確認函數已啟動以回應程式碼推送。 日誌分析：檢查 Lambda 函數的日誌是否有任何問題。日誌可以提供函數執行時所發生情況的詳細洞見，並協助識別函數是否如預期處理事件。

問題	解決方案
我需要重新部署所有微服務。	強制重新部署所有微服務的方法有兩種。選擇符合您需求的選項。 方法 1：刪除參數存放區中的參數 此方法涉及在 Systems Manager 參數存放區中刪除特定參數，以追蹤用於部署的最後一個遞交 ID。當您移除此參數時，系統會強制在下一次觸發時重新部署所有微服務，因為它會將其視為全新狀態。 步驟： 1. 找到特定參數存放區項目，該項目會保留單儲存庫的遞交 ID 或相關部署標記。參數名稱的格式如下：`"/MonoRepoTrigger/{repository}/{branch_name}/LastCommit"` 2. 如果參數值很重要，或者您想要在重設之前保留部署狀態的記錄，請考慮備份參數值。 3. 使用 AWS 管理主控台 AWS CLI、 或 SDKs 來刪除已識別的參數。此動作會重設部署標記。 4. 刪除之後，下一個推送到儲存庫應該會導致系統部署所有微服務，因為它會尋找要考慮部署的最新遞交。 優點： • 以最少的步驟輕鬆快速地實作。 • 不需要任意變更程式碼來啟動部署。 缺點： • 減少對部署程序的精細控制。

問題	解決方案
	• 如果參數存放區用於管理其他關鍵組態，則可能存在風險。 方法 2：在每個 monorepo 子資料夾中推送遞交 此方法涉及進行次要變更，並在單一儲存庫中的每個微服務子資料夾中推送，以啟動其個別管道。 步驟： 1. 列出 monorepo 內需要重新部署的所有微服務。 2. 對於每個微服務，在其子資料夾中進行最少、無影響的變更。這可能是更新README檔案、在組態檔案中新增註解，或不會影響服務功能的任何變更。 3. 使用清晰的訊息遞交這些變更（例如「啟動微服務重新部署」），並將其推送至儲存庫。請務必將變更推送至啟動部署的分支。 4. 監控每個微服務的管道，以確認它們已啟動並成功完成。 優點： • 提供對哪些微服務重新部署的精細控制。 • 更安全，因為它不涉及刪除可能用於其他用途的組態參數。 缺點： • 更耗時，尤其是使用大量的微服務。 • 需要進行不必要的程式碼變更，以免混淆遞交歷史記錄。

相關資源

- 使用 CDK 管道的持續整合和交付 (CI/CD) (AWS CDK 文件）
- aws-cdk/pipelines 模組 (AWS CDK API 參考）

使用 AWS CloudFormation 將 Bitbucket 儲存庫與 AWS Amplify 整合

Alwin Abraham，Amazon Web Services

總結

AWS Amplify 可協助您快速部署和測試靜態網站，而無需設定通常需要的基礎設施。如果您的組織想要使用 Bitbucket 進行來源控制，無論是遷移現有的應用程式程式碼還是建立新的應用程式，都可以部署此模式的方法。透過使用 AWS CloudFormation 自動設定 Amplify，您可以查看所使用的組態。

此模式說明如何使用 AWS CloudFormation 將 Bitbucket 儲存庫與 AWS Amplify 整合，以建立前端持續整合和持續部署 (CI/CD) 管道和部署環境。模式的方法表示您可以為可重複的部署建置 Amplify 前端管道。

先決條件和限制

先決條件

- 作用中的 Amazon Web Services (AWS) 帳戶
- 具有管理員存取權的作用中 Bitbucket 帳戶
- 存取使用 cURL 或 Postman 應用程式的終端機
- 熟悉 Amplify
- 熟悉 AWS CloudFormation
- 熟悉 YAML 格式的檔案

Architecture

技術堆疊

- Amplify
- AWS CloudFormation
- Bitbucket

工具

- AWS Amplify – Amplify 可協助開發人員開發和部署雲端驅動的行動和 Web 應用程式。

- AWS CloudFormation – AWS CloudFormation 是一項服務，可協助您建立和設定 AWS 資源的模型，以減少管理這些資源的時間，並有更多時間專注於在 AWS 中執行的應用程式。
- Bitbucket – Bitbucket 是專為專業團隊設計的 Git 儲存庫管理解決方案。它可讓您集中管理 Git 儲存庫、協作您的原始程式碼，並引導您完成開發流程。

Code

`bitbucket-amplify.yml` 檔案（已連接）包含此模式的 AWS CloudFormation 範本。

史詩

設定 Bitbucket 儲存庫

任務	Description	所需的技能
（選用）建立 Bitbucket 儲存庫。	1. 登入您的 Bitbucket 帳戶並建立新的儲存庫。如需詳細資訊，請參閱 Bitbucket 文件中的建立 Git 儲存庫。 2. 記錄工作區的名稱。 **Note** 您也可以使用現有的 Bitbucket 儲存庫。	DevOps 工程師
開啟工作區設定。	1. 開啟工作區，然後選擇儲存庫索引標籤。 2. 選擇您要與 Amplify 整合的儲存庫。 3. 選擇儲存庫名稱上方的工作區名稱。 4. 在側邊列上，選擇設定。	DevOps 工程師

任務	Description	所需的技能
建立 OAuth 取用者。	1. 在應用程式和功能區段中，選擇 OAuth 取用者，然後選擇新增取用者。 2. 輸入消費者的名稱，例如 `Amplify Integration` 。 3. 輸入回呼 URL。雖然此欄位是必要的輸入，但不會用來完成整合，因此值可能是 `http://localhost:3000` 4. 勾選這是私有取用者的方塊。 5. 選擇下列許可： • 專案 – `Read` • 儲存庫 – `Admin` • 提取請求 – `Read` • Webhook - `Read`和 `Write` 6. 保留所有其他欄位的預設選項，然後選擇提交。 7. 記錄產生的金鑰和秘密。	DevOps 工程師

任務	Description	所需的技能
取得 OAuth 存取權杖。	1. 開啟終端機視窗並執行下列命令： `curl -X POST -u "KEY:SECRET" https://bitbucket.org/site/oauth2/access_token -d grant_type=client_credentials` **Important** SECRET 將 KEY和 取代為您先前記錄的金鑰和秘密。 2. 不使用引號記錄存取字符。字符僅在有限時間內有效，預設時間為兩小時。您必須在此時間範圍內執行 AWS CloudFormation 範本。	DevOps 工程師

建立和部署 AWS CloudFormation 堆疊

任務	Description	所需的技能
下載 AWS CloudFormation 範本。	下載 `bitbucket-amplify.yml` AWS CloudFormation 範本（已連接）。除了 Amplify 專案和分支之外，此範本還會在 Amplify 中建立 CI/CD 管道。	

任務	Description	所需的技能
建立和部署 AWS CloudFormation 堆疊。	1. 在您要部署的 AWS 區域中登入 AWS 管理主控台，然後開啟 AWS CloudFormation 主控台。 2. 選擇建立堆疊（使用新資源），然後選擇上傳範本檔案。 3. 上傳 `bitbucket-amplify.yml` 檔案。 4. 選擇下一步，輸入堆疊名稱，然後輸入下列參數： • 存取權杖：貼上您先前建立的 OAuth 存取權杖。 • 儲存庫 URL：新增 Bitbucket 專案儲存庫的 URL。URL 通常採用下列格式： `https://bitbucket.org/<WORKSPACE_NAME>/<REPO_NAME>` • 分支名稱：這必須符合 Bitbucket 儲存庫中的分支名稱。當您執行 AWS CloudFormation 堆疊時，此分支不需要存在，但需要它才能將程式碼部署到環境。 • 專案名稱：這是要與 Amplify 專案建立關聯的名稱。	DevOps 工程師

任務	Description	所需的技能
	5. 選擇下一步，然後選擇建立堆疊。	

測試 CI/CD 管道

任務	Description	所需的技能
將程式碼部署到儲存庫中的分支。	1. 執行下列命令來複製 Bitbucket 儲存庫： `git clone https://bitbucket.org/<WORKSPACE_NAME>/<REPO_NAME>` 2. 查看執行 AWS CloudFormation 指令碼時所使用的分支名稱。若要建立和查看新的分支，請執行 `git checkout -b <BRANCH_NAME>` 命令。若要查看現有的分支，請執行 `git checkout <BRANCH_NAME>` 命令 3. 將程式碼遞交至分支，並透過執行 `git commit`和 `git push`命令將其推送至遠端分支。 4. Amplify 接著會建置和部署應用程式。 如需詳細資訊，請參閱 Bitbucket 文件中的基本 Git 命令。	應用程式開發人員

相關資源

身分驗證方法 (Atlassian 文件)

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Step Functions 和 Lambda 代理函數跨 AWS 帳戶啟動 CodeBuild 專案

Richard Milner-Watts 和 Amit Anjarlekar，Amazon Web Services

總結

此模式示範如何使用 AWS Step Functions 和 AWS Lambda 代理函數，在多個 AWS 帳戶中非同步啟動 AWS CodeBuild 專案。 AWS Step Functions AWS Lambda 您可以使用模式的範例 Step Functions 狀態機器來測試 CodeBuild 專案的成功。

CodeBuild 可協助您從全受管執行期環境使用 AWS Command Line Interface (AWS CLI) 啟動操作任務。您可以透過覆寫環境變數，在執行時間變更 CodeBuild 專案的行為。此外，您可以使用 CodeBuild 來管理工作流程。如需詳細資訊，請參閱 AWS 研討會網站上的 Service Catalog 工具，以及 AWS 資料庫部落格中的使用 AWS CodeBuild 和 Amazon EventBridge 在 Amazon RDS for PostgreSQL 中排程任務。

先決條件和限制

先決條件

- 兩個作用中的 AWS 帳戶：使用 Step Functions 叫用 Lambda 代理函數的來源帳戶，以及建置遠端 CodeBuild 範例專案的目標帳戶

限制

- 此模式無法用於在帳戶之間複製成品。

Architecture

下圖顯示此模式建置的架構。

該圖顯示以下工作流程：

1. Step Functions 狀態機器會剖析提供的輸入映射，並針對您定義的每個帳戶、區域和專案叫用 Lambda 代理函數 (`codebuild-proxy-lambda`)。
2. Lambda 代理函數使用 AWS Security Token Service (AWS STS) 來擔任 IAM 代理角色 (`codebuild-proxy-role`)，此角色與目標帳戶中的 IAM 政策 (`codebuild-proxy-policy`) 相關聯。

3. 使用擔任的角色，Lambda 函數會啟動 CodeBuild 專案並傳回 CodeBuild 任務 ID。Step Functions 狀態機器會循環和輪詢 CodeBuild 任務，直到收到成功或失敗狀態為止。

狀態機器邏輯會顯示在下圖中。

技術堆疊

- AWS CloudFormation
- CodeBuild：
- IAM
- Lambda
- 步驟函數
- X-Ray

工具

- AWS CloudFormation 可協助您設定 AWS 資源、快速一致地佈建資源，以及在整個 AWS 帳戶和區域的生命週期進行管理。
- AWS CloudFormation 設計工具提供整合式 JSON 和 YAML 編輯器，可協助您檢視和編輯 CloudFormation 範本。
- AWS CodeBuild 是一項全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而不需要佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務來建置業務關鍵應用程式。
- AWS X-Ray 可協助您收集應用程式提供的請求相關資料，並提供可用來檢視、篩選和深入了解該資料的工具，以識別問題和最佳化的機會。

Code

此模式的範例程式碼可在 GitHubCross Account CodeBuild Proxy 儲存庫中使用。此模式使用適用於 Python 的 AWS Lambda Powertools 程式庫來提供記錄和追蹤功能。如需此程式庫及其公用程式的詳細資訊，請參閱 Powertools for AWS Lambda (Python)。

最佳實務

1. 調整 Step Function 狀態機器中的等待時間值，將輪詢任務狀態的請求降至最低。使用 CodeBuild 專案的預期執行時間。
2. 在 Step Functions 中調整映射的`MaxConcurrency`屬性，以控制可以平行執行的 CodeBuild 專案數量。
3. 如有必要，請檢閱生產準備的範例程式碼。考慮解決方案可能會記錄哪些資料，以及預設的 Amazon CloudWatch 加密是否足夠。

史詩

在來源帳戶中建立 Lambda 代理函數和相關聯的 IAM 角色

任務	Description	所需的技能
記錄 AWS IDs。	需要 AWS 帳戶 IDs才能跨帳戶設定存取權。 記錄來源和目標帳戶的 AWS 帳戶 ID。如需詳細資訊，請參閱 IAM 文件中的尋找您的 AWS 帳戶 ID。	AWS DevOps
下載 AWS CloudFormation 範本。	1. 從 GitHub 儲存庫下載此模式的 `sample_target_codebuild_template.yaml` AWS CloudFormation 範本。 2. 從 GitHub 儲存庫下載`codebuild_lambda_proxy_template.yaml` 此模式的 AWS CloudFormation 範本。	AWS DevOps

任務	Description	所需的技能
	Note 在 AWS CloudFormation 範本中，<SourceAccountId> 是來源帳戶的 AWS 帳戶 ID，而 <TargetAccountId> 是目標帳戶的 AWS 帳戶 ID。	

任務	Description	所需的技能
建立和部署 AWS CloudFormation 堆疊。	1. 登入來源帳戶的 AWS 管理主控台，開啟 AWS CloudFormation 主控台，然後選擇 Stacks。 2. 選擇 Create stack (建立堆疊)，然後選擇 With new resources (standard) (使用新資源 (標準))。 3. 針對 Template source (範本來源)，選擇 Upload a template file (上傳範本檔案)。 4. 針對上傳範本檔案，選擇檔案，然後選擇下載`codebuild_lambda_proxy_template.yaml` 的檔案。選擇下一步。 5. 針對堆疊名稱，輸入堆疊的名稱 （例如，`codebuild-lambda-proxy` ）。 6. **Note** 將 `crossAccountTargetRoleArn` 參數取代為您的 `<TargetAccountId>` （例如 `<arn:aws:iam::123456789012:role/proxy-lambda-`	AWS DevOps

任務	Description	所需的技能
	codebuild-role>)。：您不需要更新 參數的targetCodeBuildProject 預設值。 7. 選擇下一步，接受預設堆疊建立選項，然後選擇下一步。 8. 選擇我確認 AWS CloudFormation 可能會建立具有自訂名稱的 IAM 資源核取方塊，然後選擇建立堆疊。 **Note** 您必須先為代理 Lambda 函數建立 AWS CloudFormation 堆疊，才能在目標帳戶中建立任何資源。當您在目標帳戶中建立信任政策時，IAM 角色會從角色名稱轉譯為內部識別符。這就是 IAM 角色必須已存在的原因。	

任務	Description	所需的技能
確認建立代理函數和狀態機器。	1. 等待 AWS CloudFormation 堆疊達到 CREATE_COMPLETE 狀態。這應該需要不到一分鐘的時間。 2. 開啟 AWS Lambda 主控台，選擇函數，然後尋找`lambda-proxy-ProxyLambda-<GUID>` 函數。 3. 開啟 AWS Step Functions 主控台，選擇狀態機器，然後尋找`sample-crossaccount-codebuild-state-machine` 狀態機器。	AWS DevOps

在目標帳戶中建立 IAM 角色，並啟動範例 CodeBuild 專案

任務	Description	所需的技能
建立和部署 AWS CloudFormation 堆疊。	1. 登入目標帳戶的 AWS 管理主控台，開啟 AWS CloudFormation 主控台，然後選擇 Stacks。 2. 選擇建立堆疊，然後選擇使用新資源 （標準）。 3. 針對 Template source (範本來源)，選擇 Upload a template file (上傳範本檔案)。 4. 針對上傳範本檔案，選擇選擇檔案，然後選擇`sample_target_code`	AWS DevOps

任務	Description	所需的技能
	build_template.yaml 檔案。選擇下一步。 5. 針對堆疊名稱，輸入堆疊的名稱（例如：sample-codebuild-stack）。 6. 將 crossAccountSourceRoleArn 參數取代為您的 <SourceAccountId>（例如 <arn:aws:iam::123456789012:role/codebuild-proxy-lambda-role>）。 7. 選擇下一步，接受預設堆疊建立選項，然後選擇下一步。 8. 選擇我確認 AWS CloudFormation 可能會建立具有自訂名稱的 IAM 資源核取方塊，然後選擇建立堆疊。	
確認已建立範例 CodeBuild 專案。	1. 等待 AWS CloudFormation 堆疊達到 CREATE_COMPLETE 狀態。這應該需要不到一分鐘的時間。 2. 開啟 AWS CodeBuild 主控台，然後尋找sample-codebuild-project 專案。	AWS DevOps

測試跨帳戶 Lambda 代理函數

任務	Description	所需的技能
啟動狀態機器。	1. 登入來源帳戶的 AWS 管理主控台，開啟 AWS Step Functions 主控台，然後選擇狀態機器。 2. 選擇`sample-crossaccount-codebuild-state-machine` 狀態機器，然後選擇開始執行。 3. 在輸入編輯器中，輸入下列 JSON，並以包含 CodeBuild 專案之帳戶的 `<TargetAccountID>` AWS 帳戶 ID 取代 。 `{ "crossAccountTargetRoleArns": [ { "arn": "arn:aws:iam::<TargetAccountID>:role/proxy-lambda-codebuild-role", "region": "eu-west-1", "codeBuildProject": "sample-codebuild-project", "SampleValue1": "Value1", "SampleValue2": "Value2" } ]`	AWS DevOps

任務	Description	所需的技能
	} **Note** 鍵/值對會以環境變數的形式，從來源帳戶中的 函數傳遞至目標帳戶中的 CodeBuild 專案。 4. 選擇 Start execution (開始執行)。 5. 在狀態機器頁面的詳細資訊索引標籤上，檢查 ifExecution Status 設定為成功。這會確認您的狀態機器正在執行。注意：狀態機器可能需要大約 30 秒才能達到成功狀態。 6. 若要查看狀態機器中步驟的輸出和輸入，請在執行事件歷史記錄區段中展開該步驟。例如，展開 Lambda - CodeBuild Proxy – 開始步驟。輸出包含覆寫的環境變數、原始承載和 CodeBuild 任務 ID 的詳細資訊。	

任務	Description	所需的技能
驗證環境變數。	1. 登入目標帳戶的 AWS 管理主控台。 2. 開啟 AWS CodeBuild 主控台，展開組建，然後選擇組建專案。 3. 選擇`sample-codebuild-project` 專案，然後選擇檢視詳細資訊。 4. 在建置歷史記錄索引標籤上，選擇專案的最新建置，然後選擇檢視日誌。 5. 在日誌輸出中，確認列印到 STDOUT 的環境變數符合 Step Functions 範例狀態機器的環境變數。	AWS DevOps

疑難排解

問題	解決方案
Step Functions 執行所花費的時間超過預期。	在 Step Function 狀態機器中調整映射的`MaxConcurrency` 屬性，以控制可以平行執行的 CodeBuild 專案數量。
CodeBuild 任務的執行時間超過預期。	1. 調整 Step Functions 狀態機器中的等待時間值，將輪詢任務狀態的請求降至最低。使用 CodeBuild 專案的預期執行時間。 2. 考慮 CodeBuild 是否為要使用的適當工具。例如，初始化 CodeBuild 任務所需的時間可能遠比 AWS Lambda 長。如果需要高輸送量和快速完成時間，請考慮將商業邏輯遷移至 AWS Lambda 並使用廣發架構。

使用應用程式復原控制器管理 EMR 叢集的多可用區域容錯移轉

Aarti Rajput、Ashish Bhatt、Neeti Mishra 和 Nidhi Sharma，Amazon Web Services

摘要

此模式為 Amazon EMR 工作負載提供有效的災難復原策略，以協助確保單一區域內多個可用區域的高可用性和資料一致性 AWS 區域。此設計使用 Amazon Application Recovery Controller 和 Application Load Balancer 來管理 Apache Spark 型 EMR 叢集的容錯移轉操作和流量分佈。

在標準條件下，主要可用區域會託管具有完整讀取/寫入功能的作用中 EMR 叢集和應用程式。如果可用區域意外故障，流量會自動重新導向至次要可用區域，其中會啟動新的 EMR 叢集。兩個可用區域都透過專用閘道端點存取共用的 Amazon Simple Storage Service (Amazon S3) 儲存貯體，以確保一致的資料管理。此方法可將停機時間降至最低，並在可用區域故障期間快速復原關鍵大數據工作負載。此解決方案適用於金融或零售等產業，其中即時分析至關重要。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- Amazon Elastic Compute Cloud (Amazon EC2) 上的 Amazon EMR Amazon EC2
- 從 EMR 叢集的主節點存取 Amazon S3。
- AWS 多可用區域基礎設施

限制

- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

產品版本

- Amazon EMR 6.x 和更新版本

Architecture

目標技術堆疊

- Amazon EMR 叢集
- Amazon 應用程式復原控制器說明
- Application Load Balancer
- Amazon S3 儲存貯體
- Amazon S3 閘道端點

目標架構

此架構使用多個可用區域並透過應用程式復原控制器實作自動化復原機制，以提供應用程式彈性。

1. Application Load Balancer 會將流量路由到作用中的 Amazon EMR 環境，通常是主要可用區域中的主要 EMR 叢集。
2. 作用中的 EMR 叢集會處理應用程式請求，並透過專用 Amazon S3 閘道端點連線至 Amazon S3，以進行讀取和寫入操作。
3. Amazon S3 做為中央資料儲存庫，可能用作檢查點或 EMR 叢集之間的共用儲存。當 EMR 叢集透過`s3://`通訊協定和 EMR 檔案系統 (EMRFS) 直接寫入 Amazon S3 時，會維持資料一致性。
4. 應用程式復原控制器會持續監控主要可用區域的運作狀態，並視需要自動管理容錯移轉操作。
5. 如果應用程式復原控制器偵測到主要 EMR 叢集失敗，則會採取下列動作：
   - 在可用區域 2 中啟動次要 EMR 叢集的容錯移轉程序。
   - 更新路由組態，將流量導向次要叢集。

工具

AWS 服務

- Amazon Application Recovery Controller 可協助您管理和協調跨 和 AWS 區域 可用區域的應用程式復原。此服務透過減少傳統工具和程序所需的手動步驟，簡化程序並改善應用程式復原的可靠性。
- Application Load Balancer 在應用程式層操作，這是開放系統互連 (OSI) 模型的第七層。它將傳入的應用程式流量分散到多個可用區域中的多個目標，例如 EC2 執行個體。這會提高您應用程式的可用性。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- Amazon EMR 是一種大數據平台，可為 Apache Spark、Apache Hive 和 Presto 等開放原始碼架構提供資料處理、互動式分析和機器學習。

- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon S3 提供簡單的 Web 服務介面，可讓您隨時從任何地方存放和擷取任意數量的資料。使用此服務，您可以輕鬆建置利用雲端原生儲存的應用程式。
- Amazon S3 的閘道端點是您在路由表中指定的閘道，可透過 AWS 網路從虛擬私有雲端 (VPC) 存取 Amazon S3。

最佳實務

- 遵循AWS 安全性、身分和合規性的最佳實務，以確保強大且安全的架構。
- 將架構與 AWS Well-Architected Framework 對齊。
- 使用 Amazon S3 Access Grants 來管理從 Spark 型 EMR 叢集到 Amazon S3 的存取權。如需詳細資訊，請參閱部落格文章使用 Amazon EMR 搭配 S3 存取授權擴展 Amazon S3 的 Spark 存取。
- 使用 Amazon S3 改善 Spark 效能。

史詩

設定您的環境

任務	Description	所需的技能
登入 AWS 管理主控台。	以 IAM 使用者AWS 管理主控台身分登入 。如需說明，請參閱 AWS 文件。	AWS DevOps
設定 AWS CLI。	安裝 AWS CLI 或將其更新至最新版本，以便在 AWS 服務 中與 互動 AWS 管理主控台。如需說明，請參閱 AWS CLI 文件。	AWS DevOps

在 EMR 叢集上部署 Spark 應用程式

任務	Description	所需的技能
建立 S3 儲存貯體。	1. 建立 S3 儲存貯體以存放輸入資料集、日誌、應用程式和輸出資料。如需說明，請參閱 Amazon S3 文件。 2. 將儲存貯體組織到用於輸入資料 (dataset)、日誌 (logs)、Spark 應用程式 (spark-app) 和輸出資料 () 的個別資料夾中output。	AWS DevOps
建立 EMR 叢集。	1. 使用下列 AWS CLI 命令來建立 EMR 叢集（例如 6.12 版或更新版本），其執行個體跨越兩個可用區域（例如 us-east-1a 和 us-east-1b ）以實現高可用性。命令會將m4.large執行個體類型指定為範例。 `aws emr create-cluster \ --ec2-attributes AvailabilityZone=<AZ-name-1> \ --release-label emr-6.12.0 \ --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large`	AWS DevOps

任務	Description	所需的技能

	<pre>aws emr create-cluster \
	    --ec2-attributes AvailabilityZone=<AZ-name-2> \
	    --release-label emr-6.12.0 \
	    --instance-groups InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m4.large InstanceGroupType=CORE,InstanceCount=2,InstanceType=m4.large</pre>
	如需詳細資訊，請參閱 create-cluster 命令和 Amazon EMR 文件。
	2. 視需要提供金鑰對、服務角色和執行個體描述檔所需的許可。	

任務	Description	所需的技能
設定 EMR 叢集的安全設定。	1. 使用 AWS CLI describe-cluster 命令識別與 EMR 叢集主節點相關聯的安全群組： `aws emr describe-cluster --cluster-id j-XXXXXXXX` 2. 若要增強安全性，請修改安全群組設定以允許 SSH 存取 (TCP 連接埠 22) 主節點，但將其限制為您的特定 IP 地址。 如需詳細資訊，請參閱 Amazon EMR 文件。	AWS DevOps

連線至 EMR 叢集。	使用提供的金鑰對，透過 SSH 連線至 EMR 叢集的主節點。
	請確定金鑰對檔案與您的應用程式位於相同的目錄中。
	執行下列命令來設定金鑰對的正確許可，並建立 SSH 連線：
	`chmod 400 <key-pair-name>`
	`ssh -i ./<key-pair-name> hadoop@<master-node-public-dns>`
		AWS DevOps

任務	Description	所需的技能

部署 Spark 應用程式。	建立 SSH 連線後，您會在 Hadoop 主控台中。
	
	1. 使用 vim 等文字編輯器建立或編輯 Spark 應用程式檔案 (`main.py`)：
	
	`vim main.py`
	
	如需建立和修改 Spark 應用程式的詳細資訊，請參閱 Amazon EMR 文件。
	2. 將 Spark 應用程式提交至 EMR 叢集，在 S3 儲存貯體中指定輸入資料和輸出資料位置：
	
	`spark-submit main.py –data_source <input-data-folder-in-s3> –output_uri <output-folder-in-s3>`
	
	以下是根據您先前設定的資料夾的範例：
	
	`spark-submit main.py –data_source dataset –output_uri output`
	3. 檢查應用程式日誌以監控應用程式的進度：
	
	`yarn logs -applicationId <application-id>`
		AWS DevOps

任務	Description	所需的技能
監控 Spark 應用程式。	1. 開啟另一個終端機視窗，並建立 EMR 叢集資源管理員 Web UI 的 SSH 通道： `ssh -i <key-pair-name> -N -L 8157:<resource-manager-public-dns>:8088 hadoop@<resource-manager-public-dns>` 2. 若要監控應用程式，請在 Web 瀏覽器`http://localhost:8157` 中導覽至 以存取資源管理員 Web UI。	AWS DevOps

將流量轉移到另一個可用區域

任務	Description	所需的技能
建立 Application Load Balancer。	設定目標群組，在 內跨兩個可用區域部署的 Amazon EMR 主節點之間路由流量 AWS 區域。 如需說明，請參閱 Elastic Load Balancing Load Balancing 文件中的為 Application Load Balancer 建立目標群組。	AWS DevOps
在應用程式復原控制器中設定區域轉移。	在此步驟中，您將使用應用程式復原控制器中的區域轉移功能，將流量轉移到另一個可用區域。	AWS DevOps

任務	Description	所需的技能
	1. 開啟應用程式復原控制器主控台。 2. 在入門下，選擇區域轉移、啟動區域轉移。 3. 選取您要從中轉移流量的可用區域。 4. 從資源資料表中選取區域轉移的支援資源 （ 例如 Application Load Balancer) 。 5. 針對設定區域轉移過期，選擇或輸入區域轉移的過期。您可以設定介於 1 分鐘到三天 (72 小時 ） 之間的持續時間。 所有區域轉移都是暫時的。您必須設定過期，但稍後可以更新作用中的輪班，將新的過期期間設定為最多三天。 6. 輸入有關此區域轉移的註解。 7. 選取核取方塊，確認開始區域轉移將減少應用程式的可用容量，方法是將流量移離可用區域。 8. 選擇 開始使用。 若要使用 AWS CLI，請參閱應用程式復原控制器文件中使用 AWS CLI 搭配區域轉移的範例 。	

任務	Description	所需的技能

驗證區域轉移組態和進度。	1. 驗證向區域轉移註冊的資源：
	
	`aws arc-zonal-shift list-managed-resources --region <AWS-region-name>`
	
	例如，下列輸出會確認兩個可用區域中的資源都已啟動並執行。
	
	`"appliedWeights": { "use1-az1": 1.0, "use1-az2": 1.0 },`
	
	2. 若要視覺化區域轉移，請使用下列 AWS CLI 命令來啟動區域轉移：
	
	`aws arc-zonal-shift start-zonal-shift \ --resource-identifier <application-load-balancer-arn> \ --away-from <source-AZ> \ --expires-in 10m --comment "testing" \ --region <AWS-region-name>`
	
	其中 <source-AZ> 是您要從中轉移流量的可用區域識別符，而 <applicat
		AWS DevOps

任務	Description	所需的技能

	ion-load-balancer-arn> 是 Application Load Balancer 的 Amazon Resource Name (ARN)。
	3. 確認流量已轉移到另一個可用區域。
	`aws arc-zonal-shift get-managed-resource \ --resource-identifier <application-load-balancer-arn> \ --region <AWS-region-name>`
	您可以看到這些權重確認的區域轉移：
	`"appliedWeights": { "use1-az1": 0.0, "use1-az2": 1.0 },`	

相關資源

- AWS CLI 命令：
  - create-cluster
  - describe-cluster
  - arc-zonal-shift
- 設定 Spot 執行個體的 Amazon EMR 叢集執行個體類型和最佳實務 (Amazon EMR 文件)
- IAM 中的安全最佳實務 (IAM 文件)
- 使用執行個體描述檔 (IAM 文件)

- 在 ARC 中使用區域轉移和區域自動轉移來復原應用程式（應用程式復原控制器文件）

使用 AWS 程式碼服務和 AWS KMS 多區域金鑰，管理將微服務部署至多個帳戶和區域的藍/綠部署

Balaji Vedagiri、Vanitha Dontireddy、Ashish Kumar、Faisal Shahdad、Vivek Thangamuthu 和 Anand Krishna Varanasi，Amazon Web Services

摘要

此模式說明如何根據藍/綠部署策略，將全域微服務應用程式從中央 AWS 帳戶部署到多個工作負載帳戶和區域。模式支援下列項目：

- 軟體是在中央帳戶中開發，而工作負載和應用程式則分散在多個帳戶和 AWS 區域。
- 單一 AWS Key Management System (AWS KMS) 多區域金鑰用於加密和解密，以涵蓋災難復原。
- KMS 金鑰是區域特定的，必須在管道成品的三個不同區域中維護或建立。KMS 多區域金鑰有助於跨區域保留相同的金鑰 ID。
- Git 工作流程分支模型使用兩個分支（開發和主要）實作，並使用提取請求 (PRs) 合併程式碼。從此堆疊部署的 AWS Lambda 函數會建立從開發分支到主分支的 PR。合併到主分支的 PR 會啟動 AWS CodePipeline 管道，協調持續整合和持續交付 (CI/CD) 流程，並在帳戶之間部署堆疊。

此模式 透過 AWS CloudFormation 堆疊提供做為程式碼 (IaC) 設定的範例基礎設施，以示範此使用案例。使用 AWS CodeDeploy 實作微服務的藍/綠部署。

先決條件和限制

先決條件

- 四個作用中的 AWS 帳戶：
  - 用於管理程式碼管道和維護 AWS CodeCommit 儲存庫的工具帳戶。
  - 部署微服務工作負載的三個工作負載（測試）帳戶。
- 此模式使用下列區域。 如果您想要使用其他區域，您必須對 AWS CodeDeploy 和 AWS KMS 多區域堆疊進行適當的修改。
  - 工具 (AWS CodeCommit) 帳戶： `ap-south-1`
  - 工作負載（測試）帳戶 1： `ap-south-1`
  - 工作負載（測試）帳戶 2： `eu-central-1`
  - 工作負載（測試）帳戶 3： `us-east-1`

- 每個工作負載帳戶中部署區域的三個 Amazon Simple Storage Service (Amazon S3) 儲存貯體。（這些在此模式中稱為 S3BUCKETNAMETESTACCOUNT1S3BUCKETNAMETESTACCOUNT2 和S3BUCKETNAMETESTACCOUNT3 更新版本。）

  例如，您可以在具有唯一儲存貯體名稱的特定帳戶和區域中建立這些儲存貯體，如下所示（以隨機數字取代 xxxx）：

```
##In Test Account 1
aws s3 mb s3://ecs-codepipeline-xxxx-ap-south-1 --region ap-south-1
##In Test Account 2
aws s3 mb s3://ecs-codepipeline-xxxx-eu-central-1 --region eu-central-1
##In Test Account 3
aws s3 mb s3://ecs-codepipeline-xxxx-us-east-1 --region us-east-1

#Example
##In Test Account 1
aws s3 mb s3://ecs-codepipeline-18903-ap-south-1 --region ap-south-1
##In Test Account 2
aws s3 mb s3://ecs-codepipeline-18903-eu-central-1 --region eu-central-1
##In Test Account 3
aws s3 mb s3://ecs-codepipeline-18903-us-east-1 --region us-east-1
```

限制

模式使用 AWS CodeBuild 和其他組態檔案來部署範例微服務。如果您有不同的工作負載類型（例如無伺服器），您必須更新所有相關組態。

Architecture

目標技術堆疊

- AWS CloudFormation
- AWS CodeCommit
- AWS CodeBuild
- AWS CodeDeploy
- AWS CodePipeline

目標架構

自動化和擴展

使用 AWS CloudFormation 堆疊範本 (IaC) 來自動化設定。它可以針對多個環境和帳戶輕鬆擴展。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速一致地佈建資源，以及在整個 AWS 帳戶和區域的生命週期進行管理。
- AWS CodeBuild 是一項全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。
- AWS CodeDeploy 會自動部署到 Amazon Elastic Compute Cloud (Amazon EC2) 或內部部署執行個體、AWS Lambda 函數或 Amazon Elastic Container Service (Amazon ECS) 服務。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- Amazon Elastic Container Service (Amazon ECS) 是快速、可擴展的容器管理服務，可協助您執行、停止和管理叢集上的容器。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他工具

- Git 是一種開放原始碼的分散式版本控制系統，可與 AWS CodeCommit 儲存庫搭配使用。
- Docker 是一組平台即服務 (PaaS) 產品，可在作業系統層級使用虛擬化在容器中交付軟體。此模式使用 Docker 在本機建置和測試容器映像。
- cfn-lint 和 cfn-nag 是開放原始碼工具，可協助您檢閱 CloudFormation 堆疊是否有任何錯誤和安全問題。

程式碼儲存庫

此模式的程式碼可在多個區域和帳戶儲存庫的 GitHub 全域藍/綠部署中使用。 https://github.com/aws-samples/ecs-blue-green-global-deployment-with-multiregion-cmk-codepipeline

史詩

設定環境變數

任務	Description	所需的技能

匯出 CloudFormation 堆疊部署的環境變數。	定義稍後在此模式中將用作 CloudFormation 堆疊輸入的環境變數。
	
	1. 更新您在三個帳戶和區域中建立的儲存貯體名稱，如先決條件一節先前所述：
	
	`export S3BUCKETNAMETESTACCOUNT1=<S3BUCKETACCOUNT1>`
	`export S3BUCKETNAMETESTACCOUNT2=<S3BUCKETACCOUNT2>`
	`export S3BUCKETNAMETESTACCOUNT3=<S3BUCKETACCOUNT3>`
	
	2. 定義隨機字串以建立成品儲存貯體，因為儲存貯體名稱全域必須是唯一的：
	
	`export BUCKETSTARTNAME=ecs-codepipeline-artifacts-19992`
	
	3. 定義和匯出帳戶 IDs和區域：
	
	`export TOOLSACCOUNT=<TOOLSACCOUNT>`
		AWS DevOps

任務	Description	所需的技能

	`export CODECOMMITACCOUNT=<CODECOMMITACCOUNT>`
	`export CODECOMMITREGION=ap-south-1`
	`export CODECOMMITREPONAME=Poc`
	`export TESTACCOUNT1=<TESTACCOUNT1>`
	`export TESTACCOUNT2=<TESTACCOUNT2>`
	`export TESTACCOUNT3=<TESTACCOUNT3>`
	`export TESTACCOUNT1REGION=ap-south-1`
	`export TESTACCOUNT2REGION=eu-central-1`
	`export TESTACCOUNT3REGION=us-east-1`
	`export TOOLSACCOUNTREGION=ap-south-1`
	`export ECRREPOSITORYNAME=web`	

封裝和部署基礎設施的 CloudFormation 堆疊

任務	Description	所需的技能

複製儲存庫。	將範例儲存庫複製到您工作位置的新儲存庫中：
	`##In work location`
	`git clone https://github.com/aws-samples/ecs-blue-green-global-deployment-`
		AWS DevOps

任務	Description	所需的技能
	`with-multiregion-cmk-codepipeline.git`	

任務	Description	所需的技能

封裝 Cloudformation 資源。	在此步驟中，您會封裝 CloudFormation 範本參考的本機成品，以建立 Amazon Virtual Private Cloud (Amazon VPC) 和 Application Load Balancer 等服務所需的基礎設施資源。
	
	範本可在程式碼儲存庫的 `Infra`資料夾中使用。
	
	`##In TestAccount1##`
	`aws cloudformation package \`
	`--template-file mainInfraStack.yaml \`
	`--s3-bucket $S3BUCKETNAMETESTACCOUNT1 \`
	`--s3-prefix infraStack \`
	`--region $TESTACCOUNT1REGION \`
	`--output-template-file infrastructure_${TESTACCOUNT1}.template`
	
	`##In TestAccount2##`
	`aws cloudformation package \`
	`--template-file mainInfraStack.yaml \`
	`--s3-bucket $S3BUCKETNAMETESTACCOUNT2 \`
	`--s3-prefix infraStack \`
		AWS DevOps

任務	Description	所需的技能
	--region $TESTACCO UNT2REGION \ --output-template- file infrastructure_ ${TESTACCOUNT2}.templ ate ##In TestAccount3## aws cloudformation package \ --template-file mainInfraStack.yaml \ --s3-bucket $S3BUCKETNAMETESTA CCOUNT3 \ --s3-prefix infraStack \ --region $TESTACCO UNT3REGION \ --output-template- file infrastructure_ ${TESTACCOUNT3}.templ ate	

任務	Description	所需的技能

驗證套件範本。	驗證套件範本：
	
	`aws cloudformation validate-template \ --template-body file://infrastructure_${TESTACCOUNT1}.template`
	
	`aws cloudformation validate-template \ --template-body file://infrastructure_${TESTACCOUNT2}.template`
	
	`aws cloudformation validate-template \ --template-body file://infrastructure_${TESTACCOUNT3}.template`
		AWS DevOps

任務	Description	所需的技能

將套件檔案部署到工作負載帳戶，	1. 根據您的設定更新 infraParameters.json 指令碼中的預留位置值和帳戶名稱。
	2. 將套件範本部署到您的三個工作負載帳戶。
	
	`##In TestAccount1##`
	`aws cloudformation deploy \`
	`--template-file infrastructure_${TESTACCOUNT1}.template \`
	`--stack-name mainInfrastack \`
	`--parameter-overrides file://infraParameters.json \`
	`--region $TESTACCOUNT1REGION \`
	`--capabilities CAPABILITY_IAM CAPABILITY_NAMED_IAM`
	
	`##In TestAccount2##`
	`aws cloudformation deploy \`
	`--template-file infrastructure_${TESTACCOUNT2}.template \`
	`--stack-name mainInfrastack \`
	`--parameter-overrides file://infraParameters.json \`
	`--region $TESTACCOUNT2REGION \`
		AWS DevOps

任務	Description	所需的技能
	--capabilities CAPABILITY_IAM CAPABILITY_NAMED_IAM ##In TestAccount3## aws cloudformation deploy \ --template-file infrastructure_${TESTACCOUNT3}.template \ --stack-name mainInfrastack \ --parameter-overrides file://infraParameters.json \ --region $TESTACCOUNT3REGION \ --capabilities CAPABILITY_IAM CAPABILITY_NAMED_IAM	

推送範例映像並擴展 Amazon ECS

任務	Description	所需的技能
將範例映像推送至 Amazon ECR 儲存庫。	將範例 (NGINX) 映像推送至名為 web（如參數中設定）的 Amazon Elastic Container Registry (Amazon ECR) 儲存庫。您可以視需要自訂映像。 若要登入並設定將映像推送到 Amazon ECR 的登入資料，請遵循 Amazon ECR 文件中的指示。	AWS DevOps

任務	Description	所需的技能

	命令包括：
	
	`docker pull nginx`
	`docker images`
	`docker tag <imageid> aws_account_id.dkr.ecr.region.amazonaws.com/<web>:latest`
	`docker push <aws_account_id>.dkr.ecr.<region>.amazonaws.com/<web>:tag`	

擴展 Amazon ECS 並驗證存取。	1. 擴展 Amazon ECS 以建立兩個複本： `aws ecs update-service --cluster QA-Cluster --service Poc-Service --desired-count 2` 其中 `Poc-Service` 是指您的範例應用程式。 2. 使用瀏覽器的完整網域名稱 (FQDN) 或 DNS，或使用 curl 命令，確認可從 Application Load Balancer 存取服務。	AWS DevOps

設定程式碼服務和資源

任務	Description	所需的技能
在工具帳戶中建立 CodeCommit 儲存庫。	使用位於 GitHub 儲存庫`code`資料夾的 `codecommi`	AWS DevOps

任務	Description	所需的技能
	t.yaml 範本，在工具帳戶中建立 CodeCommit 儲存庫。您只能在計劃開發程式碼的單一區域中建立此儲存庫。 `aws cloudformation deploy --stack-name codecommitrepoStack --parameter-overrides CodeCommitReponame=$CODECOMMITREPONAME \ ToolsAccount=$TOOLSACCOUNT --template-file codecommit.yaml --region $TOOLSACCOUNTREGION \ --capabilities CAPABILITY_NAMED_IAM`	

任務	Description	所需的技能
建立 S3 儲存貯體以管理 CodePipeline 產生的成品。	使用位於 GitHub 儲存庫 `code` 資料夾中的 `pre-reqs-bucket.yaml` 範本，建立 S3 儲存貯體來管理 CodePipeline 產生的成品。堆疊必須部署在所有三個工作負載（測試）和工具帳戶和區域中。 `aws cloudformation deploy --stack-name pre-reqs-artifacts-bucket --parameter-overrides BucketStartName=$BUCKETSTARTNAME \ TestAccount1=$TESTACCOUNT1 TestAccount2=$TESTACCOUNT2 \ TestAccount3=$TESTACCOUNT3 CodeCommitAccount=$CODECOMMITACCOUNT ToolsAccount=$TOOLSACCOUNT \ --template-file pre-reqs_bucket.yaml --region $TESTACCOUNT1REGION --capabilities CAPABILITY_NAMED_IAM`  `aws cloudformation deploy --stack-name pre-reqs-artifacts-bucket --parameter-overrides BucketStartName=$BUCKETSTARTNAME \ TestAccount1=$TESTACCOUNT1 TestAccount2=$TESTACCOUNT2 \`	AWS DevOps

任務	Description	所需的技能
	TestAccount3=$TE STACCOUNT3 CodeCommi tAccount=$CODECOMM ITACCOUNT ToolsAcco unt=$TOOLSACCOUNT \ --template-file pre- reqs_bucket.yaml --region $TESTACCO UNT2REGION --capabil ities CAPABILIT Y_NAMED_IAM aws cloudformation deploy --stack-name pre-reqs-artifacts -bucket --parameter- overrides BucketSta rtName=$BUCKETSTAR TNAME \ TestAccount1=$TE STACCOUNT1 TestAccou nt2=$TESTACCOUNT2 \ TestAccount3=$TE STACCOUNT3 CodeCommi tAccount=$CODECOMM ITACCOUNT ToolsAcco unt=$TOOLSACCOUNT \ --template-file pre- reqs_bucket.yaml --region $TESTACCO UNT3REGION --capabil ities CAPABILIT Y_NAMED_IAM aws cloudformation deploy --stack-name pre-reqs-artifacts -bucket --parameter- overrides BucketSta rtName=$BUCKETSTAR TNAME \	

任務	Description	所需的技能
	TestAccount1=$TESTACCOUNT1 TestAccount2=$TESTACCOUNT2 \ TestAccount3=$TESTACCOUNT3 CodeCommitAccount=$CODECOMMITACCOUNT ToolsAccount=$TOOLSACCOUNT \ --template-file pre-reqs_bucket.yaml --region $TOOLSACCOUNTREGION --capabilities CAPABILITY_NAMED_IAM	

任務	Description	所需的技能
設定多區域 KMS 金鑰。	1. 使用 CodePipeline 將使用的主要金鑰和複本金鑰來建立多區域 KMS 金鑰。在我們的範例中，`ToolsAccount1region - ap-south-1` 將是主要區域。 `aws cloudformation deploy --stack-name ecs-codepipeline-pre-reqs-KMS \ --template-file pre-reqs_KMS.yaml --parameter-overrides \ TestAccount1=$TESTACCOUNT1 TestAccount2=$TESTACCOUNT2 \ TestAccount3=$TESTACCOUNT3 CodeCommitAccount=$CODECOMMITACCOUNT ToolsAccount=$TOOLSACCOUNT --region $TOOLSACCOUNTREGION` 2. 設定要傳遞至 CodeBuild 專案的 CMKARN 變數。這些值可在 ecs-codepipeline-pre-reqs-KMS 範本堆疊的輸出中使用（金鑰 ID 在所有區域中都會相同，並以開頭`mrk-`)。或者，您可以從工具帳戶取得 CMKARN 值。在所有帳戶工作階段中匯出它們：	AWS DevOps

任務	Description	所需的技能

	`export CMKARN1=arn:aws:kms:ap-south-1:<TOOLSACCOUNTID>:key/mrk-xxx`
	`export CMKARN2=arn:aws:kms:eu-central-1:<TOOLSACCOUNTID>:key/mrk-xxx`
	`export CMKARN3=arn:aws:kms:us-east-1:<TOOLSACCOUNTID>:key/mrk-xxx`
	`export CMARNTOOLS=arn:aws:kms:ap-south-1:<TOOLSACCOUNTID>:key/mrk-xxx`	

任務	Description	所需的技能

在工具帳戶中設定 CodeBuild 專案。	1. 使用 GitHub 儲存庫code資料夾中的 codebuild_IAM.yaml 範本，在工具帳戶中的單一區域中設定 AWS CodeBuild 的 AWS Identity and Access Management (IAM)： AWS CodeBuild
	`#In ToolsAccount`
	`aws cloudformation deploy --stack-name ecs-codebuild-iam \`
	`--template-file codebuild_IAM.yaml --region $TOOLSACCOUNTREGION \`
	`--capabilities CAPABILITY_NAMED_IAM`
	2. 使用 codebuild.yaml 範本為您的建置專案設定 CodeBuild。在全部三個區域中部署此範本，如下所示：
	`aws cloudformation deploy --stack-name ecscodebuildstack --parameter-overrides ToolsAccount=$TOOLSACCOUNT \`
	`CodeCommitRepoName=$CODECOMMITREPONAME ECRRepositoryName=$ECRREPOSITORYNAME APPACCOUNTID=$TESTACCOUNT1 \`
		AWS DevOps

任務	Description	所需的技能
	TestAccount3=$TE STACCOUNT3 CodeCommi tRegion=$CODECOMMI TREGION CMKARN=$C MKARN1 \ --template-file codebuild.yaml --region $TESTACCO UNT1REGION --capabil ities CAPABILIT Y_NAMED_IAM aws cloudformation deploy --stack-name ecscodebuildstack -- parameter-overrides ToolsAccount=$TOOL SACCOUNT \ CodeCommitRepoName= $CODECOMMITREPONAME ECRRepositoryName= $ECRREPOSITORYNAME APPACCOUNTID=$TEST ACCOUNT2 \ TestAccount3=$TE STACCOUNT3 CodeCommi tRegion=$CODECOMMI TREGION CMKARN=$C MKARN2 \ --template-file codebuild.yaml --region $TESTACCO UNT2REGION --capabil ities CAPABILIT Y_NAMED_IAM aws cloudformation deploy --stack-name ecscodebuildstack -- parameter-overrides ToolsAccount=$TOOL SACCOUNT \	

任務	Description	所需的技能
	CodeCommitRepoName=$CODECOMMITREPONAME ECRRepositoryName=$ECRREPOSITORYNAME APPACCOUNTID=$TESTACCOUNT3 \ CodeCommitRegion=$CODECOMMITREGION CMKARN=$CMKARN3 \ --template-file codebuild.yaml --region $TESTACCOUNT3REGION --capabilities CAPABILITY_NAMED_IAM	

任務	Description	所需的技能

在工作負載帳戶中設定 CodeDeploy。	使用 GitHub 儲存庫 `code` 資料夾中的 `codedeploy.yaml` 範本，在所有三個工作負載帳戶中設定 CodeDeploy。的輸出`mainInfraStack` 包含 Amazon ECS 叢集的 Amazon Resource Name (ARNs) 和 Application Load Balancer 接聽程式。
	
	Note
	基礎設施堆疊的值已匯出，因此它們是由 CodeDeploy 堆疊範本匯入。
	
	`##WorkloadAccount1##`
	`aws cloudformation deploy --stack-name ecscodedeploystack \`
	`--parameter-overrides ToolsAccount=$TOOLSACCOUNT mainInfrastackname=mainInfrastack \`
	`--template-file codedeploy.yaml --region $TESTACCOUNT1REGION --capabilities CAPABILITY_NAMED_IAM`
	
	`##WorkloadAccount2##`
	`aws cloudformation deploy --stack-name ecscodedeploystack \`
		AWS DevOps

任務	Description	所需的技能
	--parameter-overrides ToolsAccount=$TOOLSACCOUNT mainInfrastackname=mainInfrastack \ --template-file codedeploy.yaml --region $TESTACCOUNT2REGION --capabilities CAPABILITY_NAMED_IAM ##WorkloadAccount3## aws cloudformation deploy --stack-name ecscodedeploystack \ --parameter-overrides ToolsAccount=$TOOLSACCOUNT mainInfrastackname=mainInfrastack \ --template-file codedeploy.yaml --region $TESTACCOUNT3REGION --capabilities CAPABILITY_NAMED_IAM	

在工具帳戶中設定 CodePipeline

任務	Description	所需的技能
在工具帳戶中建立程式碼管道。	在工具帳戶中，執行 命令： aws cloudformation deploy --stack-name ecscodepipelinestack --parameter-overrides \	AWS DevOps

任務	Description	所需的技能
	TestAccount1=$TE STACCOUNT1 TestAccou nt1Region=$TESTACC OUNT1REGION \ TestAccount2=$TE STACCOUNT2 TestAccou nt2Region=$TESTACC OUNT2REGION \ TestAccount3=$TE STACCOUNT3 TestAccou nt3Region=$TESTACC OUNT3REGION \ CMKARNTools=$CMK TROOLSARN CMKARN1= $CMKARN1 CMKARN2=$ CMKARN2 CMKARN3=$ CMKARN3 \ CodeCommitRepoName= $CODECOMMITREPONAME BucketStartName=$B UCKETSTARTNAME \ --template-file codepipeline.yaml -- capabilities CAPABILIT Y_NAMED_IAM	

任務	Description	所需的技能
在 AWS KMS 金鑰政策和 S3 儲存貯體政策中提供 CodePipeline 和 CodeBuild 角色的存取權。	1. 在 AWS KMS 金鑰政策中提供 CodePipeline 和 CodeBuild 角色的存取權： `aws cloudformation deploy --stack-name ecs-codepipeline-pre-reqs-KMS \ --template-file pre-reqs_KMS.yaml --parameter-overrides \ CodeBuildCondition=true TestAccount1=$TESTACCOUNT1 TestAccount2=$TESTACCOUNT2 \ TestAccount3=$TESTACCOUNT3 CodeCommitAccount=$CODECOMMITACCOUNT ToolsAccount=$TOOLSACCOUNT --region $TOOLSACCOUNTREGION` 2. 更新 S3 儲存貯體政策以允許 CodePipeline 和 CodeDeploy 角色的存取： `aws cloudformation deploy --stack-name pre-reqs-artifacts-bucket --parameter-overrides BucketStartName=$BUCKETSTARTNAME \ PutS3BucketPolicy=true TestAccount1=$TESTACCOUNT1`	AWS DevOps

任務	Description	所需的技能
	TestAccount2=$TEST ACCOUNT2 \ TestAccount3=$TE STACCOUNT3 CodeCommi tAccount=$CODECOMM ITACCOUNT ToolsAcco unt=$TOOLSACCOUNT \ --template-file pre- reqs_bucket.yaml --region $TESTACCO UNT1REGION --capabil ities CAPABILIT Y_NAMED_IAM aws cloudformation deploy --stack-name pre-reqs-artifacts -bucket --parameter- overrides BucketSta rtName=$BUCKETSTAR TNAME \ PutS3BucketPolic y=true TestAccou nt1=$TESTACCOUNT1 TestAccount2=$TEST ACCOUNT2 \ TestAccount3=$TE STACCOUNT3 CodeCommi tAccount=$CODECOMM ITACCOUNT ToolsAcco unt=$TOOLSACCOUNT \ --template-file pre- reqs_bucket.yaml --region $TESTACCO UNT2REGION --capabil ities CAPABILIT Y_NAMED_IAM aws cloudformation deploy --stack-name pre-reqs-artifacts -bucket --parameter-	

任務	Description	所需的技能
	overrides BucketStartName=$BUCKETSTARTNAME \ PutS3BucketPolicy=true TestAccount1=$TESTACCOUNT1 TestAccount2=$TESTACCOUNT2 \ TestAccount3=$TESTACCOUNT3 CodeCommitAccount=$CODECOMMITACCOUNT ToolsAccount=$TOOLSACCOUNT \ --template-file pre-reqs_bucket.yaml --region $TESTACCOUNT3REGION --capabilities CAPABILITY_NAMED_IAM aws cloudformation deploy --stack-name pre-reqs-artifacts-bucket --parameter-overrides BucketStartName=$BUCKETSTARTNAME \ PutS3BucketPolicy=true TestAccount1=$TESTACCOUNT1 TestAccount2=$TESTACCOUNT2 \ TestAccount3=$TESTACCOUNT3 CodeCommitAccount=$CODECOMMITACCOUNT ToolsAccount=$TOOLSACCOUNT \ --template-file pre-reqs_bucket.yaml --region $TOOLSACCOUNTREGION --capabil	

任務	Description	所需的技能
	`ities CAPABILIT Y_NAMED_IAM`	

呼叫並測試管道

任務	Description	所需的技能

將變更推送至 CodeCommit 儲存庫。	1. `codecommitrepoStac k` 使用 `git clone`命令複製在 中建立的 CodeCommit 儲存庫，如 AWS CodeCommit 文件所述。
	2. 使用必要的詳細資訊更新輸入成品：
	• JSON 檔案：在此檔案的三個位置更新 檔案`AccountID` 中的 。重新命名三個檔案以包含帳戶 IDs。
	• YAML 檔案：更新任務定義 ARN 和版本。重新命名三個檔案以包含帳戶 IDs。
	3. 修改 `index.html` 檔案，對首頁進行一些次要變更。
	4. 將下列檔案複製到儲存庫並遞交：
	`index.html`
	`Dockerfile`
	`buildspec.yaml`	

任務	Description	所需的技能
	appspec_<accountid>.yaml (3 files - one per account) taskdef<accountid>.json (3 files - one per account) 5. 啟動或重新啟動管道並驗證結果。 6. 使用 FQDN 或 DNS 從 Application Load Balancer 存取服務，並確認已部署更新。	

清除

任務	Description	所需的技能

清除所有已部署的資源。	1. 向下擴展 Amazon ECS 到零執行個體：
	`aws ecs update-service --cluster QA-Cluster --service Poc-Service --desired-count 0`
	2. 刪除每個帳戶和區域中的 CloudFormation 堆疊：
	`##In Tools Account## aws cloudformation delete-stack --stack-name ecscodepipelinestack --region $TOOLSACCOUNTREGION aws cloudformation delete-stack --`	

任務	Description	所需的技能
	stack-name ecscodebuildstack --region $TESTACCOUNT1REGION aws cloudformation delete-stack --stack-name ecscodebuildstack --region $TESTACCOUNT2REGION aws cloudformation delete-stack --stack-name ecscodebuildstack --region $TESTACCOUNT3REGION aws cloudformation delete-stack --stack-name ecs-codepipeline-pre-reqs-KMS --region $TOOLSACCOUNTREGION aws cloudformation delete-stack --stack-name codecommitrepoStack --region $TOOLSACCOUNTREGION aws cloudformation delete-stack --stack-name pre-reqs-artifacts-bucket --region $TESTACCOUNT1REGION aws cloudformation delete-stack --stack-name pre-reqs-artifacts-bucket --region $TESTACCOUNT2REGION aws cloudformation delete-stack --stack-name pre-reqs-artifacts-bucket --region $TESTACCOUNT3REGION	

任務	Description	所需的技能
	aws cloudformation delete-stack --stack-name pre-reqs-artifacts-bucket --region $TOOLSACCOUNTREGION aws cloudformation delete-stack --stack-name ecs-codebuild-iam --region $TOOLSACCOUNTREGION ##NOTE: Artifact buckets will not get deleted if there are artifacts so it has to be emptied manually before deleting.##	
	##In Workload / Test Accounts## ##Account:1## aws cloudformation delete-stack --stack-name ecscodedeployStack --region $TESTACCOUNT1REGION aws cloudformation delete-stack --stack-name mainInfraStack --region $TESTACCOUNT1REGION ##Account:2## aws cloudformation delete-stack --stack-name ecscodedeployStack --region $TESTACCOUNT2REGION aws cloudformation delete-stack --	

任務	Description	所需的技能
	`stack-name mainInfra stack --region $TESTACCOUNT2REGION ##Account:3## aws cloudformation delete-stack --stack-name ecscodedeployst ack --region $TESTACCOUNT3REGION aws cloudformation delete-stack --stack-name mainInfra stack --region $TESTACCOUNT3REGION ##NOTE: Amazon ECR (web) will not get deleted if the registry still includes images. It can be manually cleaned up if not required.`	

疑難排解

問題	解決方案
您遞交給儲存庫的變更並未部署。	• 檢查 CodeBuild 日誌是否有 Docker 建置動作中的錯誤。如需詳細資訊，請參閱 CodeBuild 文件。 • 檢查 CodeDeploy 部署是否有任何 Amazon ECS 部署問題。

相關資源

- 推送 Docker 映像 (Amazon ECR 文件)

- 連線至 AWS CodeCommit 儲存庫 (AWS CodeCommit 文件)
- 故障診斷 AWS CodeBuild (AWS CodeBuild 文件)

使用 AWS CloudFormation 和 AWS Config 監控 Amazon ECR 儲存庫是否有萬用字元許可

Vikrant Telkar、Wassim Benhallam 和 Sajid Momin，Amazon Web Services

總結

在 Amazon Web Services (AWS) 雲端上，Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，使用 AWS Identity and Access Management (IAM) 支援具有資源型許可的私有儲存庫。

IAM 在資源和動作屬性中都支援「*」萬用字元，這可讓您更輕鬆地自動選擇多個相符項目。在您的測試環境中，您可以在儲存庫政策陳述式的主體元素中使用`ecr:*`萬用字元許可，允許所有已驗證的 AWS 使用者存取 Amazon ECR 儲存庫。 https://docs.aws.amazon.com/AmazonECR/latest/userguide/set-repository-policy.html萬`ecr:*`用字元許可在無法存取生產資料的開發帳戶中開發和測試時非常有用。

不過，您必須確定您的生產環境中未使用`ecr:*`萬用字元許可，因為它可能會導致嚴重的安全漏洞。此模式的方法可協助您識別儲存庫政策陳述式中包含`ecr:*`萬用字元許可的 Amazon ECR 儲存庫。 模式提供步驟和 AWS CloudFormation 範本，以在 AWS Config 中建立自訂規則。然後AWS Lambda 函數會監控您的 Amazon ECR 儲存庫政策陳述式是否有`ecr:*`萬用字元許可。如果找到不合規的儲存庫政策陳述式，Lambda 會通知 AWS Config 將事件傳送至 Amazon EventBridge，然後 EventBridge 會啟動 Amazon Simple Notification Service (Amazon SNS) 主題。SNS 主題會透過電子郵件通知您不合規的儲存庫政策陳述式。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 安裝並設定 AWS Command Line Interface (AWS CLI)。如需詳細資訊，請參閱 AWS CLI 文件中的安裝、更新和解除安裝 AWS CLI。
- 具有附加政策陳述式的現有 Amazon ECR 儲存庫，在您的測試環境中安裝和設定。如需詳細資訊，請參閱 Amazon ECR 文件中的建立私有儲存庫和設定儲存庫政策陳述式。
- AWS Config，在您偏好的 AWS 區域中設定。如需詳細資訊，請參閱 AWS Config 文件中的 AWS Config 入門。 AWS Config
- `aws-config-cloudformation.template` 檔案 （ 已連接），下載到您的本機電腦。

限制

- 此模式的解決方案為區域性，您的資源必須在相同的區域中建立。

Architecture

下圖顯示 AWS Config 如何評估 Amazon ECR 儲存庫政策陳述式。

該圖顯示以下工作流程：

1. AWS Config 會啟動自訂規則。
2. 自訂規則會叫用 Lambda 函數，以評估 Amazon ECR 儲存庫政策陳述式的合規性。然後，Lambda 函數會識別不合規的儲存庫政策陳述式。
3. Lambda 函數會將不合規狀態傳送至 AWS Config。
4. AWS Config 會將事件傳送至 EventBridge。
5. EventBridge 會將不合規通知發佈至 SNS 主題。
6. Amazon SNS 會傳送電子郵件提醒給您或授權的使用者。

自動化和擴展

此模式的解決方案可以監控任意數量的 Amazon ECR 儲存庫政策陳述式，但您想要評估的所有資源都必須在相同區域中建立。

工具

- AWS CloudFormation – AWS CloudFormation 可協助您建立模型並設定 AWS 資源、快速一致地佈建資源，以及在整個生命週期中管理資源。您可以使用範本來描述資源及其相依性，並將它們一起啟動和設定為堆疊，而不是個別管理資源。您可以管理和佈建跨多個 AWS 帳戶和 AWS 區域的堆疊。
- AWS Config – AWS Config 提供 AWS 帳戶中 AWS 資源組態的詳細檢視。這包含資源彼此之間的關係和之前的組態方式，所以您可以看到一段時間中組態和關係的變化。
- Amazon ECR–Amazon Elastic Container Registry (Amazon ECR) 是一種 AWS 受管容器映像登錄服務，安全、可擴展且可靠。Amazon ECR 支援私有儲存庫，其具有使用 IAM 的資源型許可。

- Amazon EventBridge – Amazon EventBridge 是一種無伺服器事件匯流排服務，可用來將應用程式與來自各種來源的資料連線。EventBridge 會將即時資料從您的應用程式、軟體即服務 (SaaS) 應用程式和 AWS 服務串流傳送至目標，例如 AWS Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他帳戶中的事件匯流排。
- AWS Lambda – AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。只需為使用的運算時間支付費用，一旦未執行程式碼，就會停止計費。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 會協調和管理發佈者和用戶端之間的訊息傳遞或傳送，包括 Web 伺服器和電子郵件地址。訂閱者會收到發佈到所訂閱主題的所有訊息，且某一主題的所有訂閱者均會收到相同訊息。

Code

此模式的程式碼可在 `aws-config-cloudformation.template` 檔案中取得 （已連接）。

史詩

建立 AWS CloudFormation 堆疊

任務	Description	所需的技能
建立 AWS CloudFormation 堆疊。	在 AWS CLI 中執行下列命令來建立 AWS CloudFormation 堆疊： `$ aws cloudformation create-stack --stack-name=AWSConfigECR \ --template-body file://aws-config-cloudformation.template \ --parameters ParameterKey=<email>,ParameterValue=<myemail@example.com> \ --capabilities CAPABILITY_NAMED_IAM`	AWS DevOps

測試 AWS Config 自訂規則

任務	Description	所需的技能
測試 AWS Config 自訂規則。	1. 登入 AWS 管理主控台，開啟 AWS Config 主控台，然後選擇資源。 2. 在資源庫存頁面上，您可以依資源類別、資源類型和合規狀態進行篩選。 3. 包含 的 Amazon ECR 儲存庫`ecr:*`是 `NON-COMPLIANT?` ，而不包含 的 Amazon ECR 儲存庫`ecr:*`是 `COMPLIANT` 。 4. 如果 Amazon ECR 儲存庫包含不合規的政策陳述式，訂閱 SNS 主題的電子郵件地址會收到通知。	AWS DevOps

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS CDK 和 GitHub Actions 工作流程最佳化多帳戶無伺服器部署

Sarat Chandra Pothula 和 VAMSI KRISHNA SUNKAVALLI，Amazon Web Services

總結

跨多個 AWS 帳戶 和環境部署無伺服器基礎設施的組織，通常會遇到重複程式碼、手動程序和不一致實務等挑戰。此模式的解決方案說明如何使用 Go 和 GitHub Actions 可重複使用工作流程 AWS Cloud Development Kit (AWS CDK) 中的 ，以簡化多帳戶無伺服器基礎設施管理。此解決方案示範如何將雲端資源定義為程式碼、實作標準化持續整合/持續部署 (CI/CD) 程序，以及建立模組化、可重複使用的元件。

透過使用這些工具，組織可以有效率地管理跨帳戶資源、實作一致的部署管道，以及簡化複雜的無伺服器架構。該方法還透過強制執行標準化實務來與 搭配使用來增強安全性和合規性 AWS 帳戶，最終提高生產力並減少無伺服器應用程式開發和部署中的錯誤。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- AWS Identity and Access Management (IAM) 角色和許可適用於部署程序。這包括存取 Amazon Elastic Container Registry (Amazon ECR) 儲存庫、建立 AWS Lambda 函數和目標中任何其他必要資源的許可 AWS 帳戶。
- AWS Command Line Interface (AWS CLI) 2.9.11 版或更新版本，已安裝並設定。
- AWS Cloud Development Kit (AWS CDK) 2.114.1 版或更新版本，已安裝並已引導。
- Go 1.22 或更新版本，已安裝。
- 已安裝 Docker 24.0.6 或更新版本。 https://docs.docker.com/engine/install/

限制

- 語言相容性 – Go 是無伺服器應用程式的熱門語言。不過，除了 Go 之外， 還 AWS CDK 支援其他程式設計語言，包括 C#、Java、Python 和 TypeScript。如果您的組織有現有的程式碼庫或其他語言的專業知識，您可能需要調整或學習 Go 以充分利用模式中描述的解決方案。
- 學習曲線 – 採用 AWS CDK、Go （ 如果是組織的新手 ） 和 GitHub 可重複使用的工作流程，可能涉及開發人員和 DevOps 團隊的學習曲線。可能需要訓練和文件，以確保這些技術的順利採用和有效使用。

Architecture

下圖顯示此模式的工作流程和架構元件。

此解決方案會執行下列步驟：

1. 開發人員會複製儲存庫、建立新的分支，並在其本機環境中變更應用程式碼。
2. 開發人員遞交這些變更，並將新的分支推送至 GitHub 儲存庫。
3. 開發人員在 GitHub 儲存庫中建立提取請求，提議將其功能或新功能分支合併到主分支。
4. 此提取請求會觸發持續整合 (CI) GitHub 動作工作流程。此模式中的 CI 和持續部署 (CD) 工作流程使用可重複使用的工作流程，這是預先定義的模組化範本，可在不同的專案或儲存庫之間共用和執行。可重複使用的工作流程可提升 CI/CD 程序的標準化和效率。
5. CI 工作流程會設定必要的環境、產生映像的 Docker 標籤，並使用應用程式碼建置 Docker 映像。
6. CI 工作流程使用 central AWS 帳戶 GitHub OIDC 角色 AWS 向 進行驗證。針對 CI 工作流程，Central AWS 帳戶 GitHub OIDC 角色會使用 AWS Security Token Service (AWS STS) 來取得臨時登入資料。這些登入資料可讓角色建置 Docker 映像並將其推送至中央的 Amazon ECR 儲存庫 AWS 帳戶。
7. CI 工作流程會將建置的 Docker 映像推送至 Amazon ECR。
8. CI 工作流程會將映像標籤儲存到 Systems Manager 參數存放區。
9. CI 工作流程成功完成後，系統會輸出 Docker 映像標籤。
10. 觸發 CD 工作流程時，開發人員會手動輸入要部署之 Docker 映像的映像標籤。此映像標籤對應於 CI 工作流程期間產生並推送至 Amazon ECR 的標籤。
11. 開發人員會手動觸發使用 CD 可重複使用工作流程的 CD 工作流程。
12. CD 工作流程 AWS 使用 central AWS 帳戶 GitHub OIDC 角色向 進行驗證。對於 CD 工作流程，AWS STS 會先用來擔任 central AWS 帳戶 GitHub OIDC 角色。然後，此角色會擔任目標帳戶部署的 CDK 引導角色。
13. CD 工作流程使用 AWS CDK 來合成 AWS CloudFormation 範本。
14. CD 工作流程 AWS 帳戶 會使用 CDK 部署，使用 Lambda 函數的手動指定映像標籤，將應用程式部署到目標。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端 基礎設施。
- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。CloudFormation 是 AWS CDK 部署程序不可或缺的一部分。CDK 會合成 CloudFormation 範本，然後使用 CloudFormation 在 AWS 環境中建立或更新資源。
- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Systems Manager 參數存放區為組態資料管理和秘密管理提供安全的階層式儲存。

其他工具

- Docker 是一組平台即服務 (PaaS) 產品，可在作業系統層級使用虛擬化在容器中交付軟體。
- GitHub Actions 是與 GitHub 儲存庫緊密整合的持續整合和持續交付 (CI/CD) 平台。您可以使用 GitHub 動作來自動化建置、測試和部署管道。
- Go 是 Google 支援的開放原始碼程式設計語言。

程式碼儲存庫

此模式的程式碼可在 GitHub aws-cdk-golang-serverless-cicd-github-actions 儲存庫中使用。

最佳實務

- 模組化設計 – 將您的 AWS CDK 程式碼組織成模組化且可重複使用的建構或堆疊，提升跨多個帳戶和專案的程式碼重複使用和維護能力。
- 分離問題 – 將基礎設施程式碼與應用程式程式碼分開，允許獨立部署和管理每個元件。
- 版本控制和不可變性 – 將您的基礎設施視為程式碼 (IaC)，並使用 Git 進行版本控制。透過建立新的資源，而不是修改現有的資源，來接受不可變的基礎設施原則。
- 測試和驗證 – 實作全面的測試策略，包括單元測試、整合測試和end-to-end測試，以協助支援 AWS CDK 程式碼和部署的正確性和可靠性。

- 安全與合規 – 遵循 AWS 安全最佳實務，例如最低權限存取、安全通訊和資料加密。實作合規檢查和稽核機制，以確保遵守組織政策和法規要求。實作容器映像的安全最佳實務，例如掃描漏洞、強制執行映像簽署，以及遵守組織的合規要求。
- 監控和記錄 – 設定監控和記錄機制，以追蹤無伺服器應用程式和基礎設施的運作狀態和效能。使用 AWS 服務 如 Amazon CloudWatch AWS CloudTrail，以及 AWS X-Ray 進行監控和稽核。
- 自動化和 CI/CD – 使用 GitHub 可重複使用的工作流程和其他 CI/CD 工具來自動化建置、測試和部署程序，這有助於支援跨多個帳戶進行一致且可重複的部署。
- 環境管理 – 維護個別的環境 （例如，開發、預備和生產）。實作促進環境之間變更的策略，確保在生產部署之前進行適當的測試和驗證。
- 文件和協作 – 記錄您的基礎設施程式碼、部署程序和最佳實務，以促進團隊內的知識分享和協作。
- 成本最佳化 – 實作成本監控和最佳化策略，例如對資源進行授權、利用自動擴展，以及利用 AWS Budgets 和 等 AWS 成本最佳化服務 AWS Cost Explorer。
- 災難復原和備份 – 透過實作無伺服器應用程式和基礎設施資源的備份和還原機制，規劃災難復原案例。
- 持續改進 – 定期檢閱並更新您的實務、工具和程序，以符合無伺服器生態系統的最新最佳實務、安全建議和技術進展。
- 改善安全狀態 – 透過為 Amazon ECR 和 AWS Systems Manager 參數存放區設定介面 VPC 端點 AWS Lambda，使用 AWS PrivateLink來改善虛擬私有雲端 (VPC) 的安全狀態。

史詩

設定環境

任務	Description	所需的技能
在中央建立 Amazon ECR 儲存庫 AWS 帳戶。	若要跨多個 共用容器映像 AWS 帳戶，您必須設定 Amazon ECR 的跨帳戶存取。首先，在中央建立 Amazon ECR 儲存庫 AWS 帳戶。 若要建立 Amazon ECR 儲存庫，請執行下列命令：	AWS DevOps

任務	Description	所需的技能
	`aws ecr create-repository --repository-name sample-repo` 在稍後的任務中，將提取存取權授予需要使用容器映像的另一個 AWS 帳戶 。	

任務	Description	所需的技能
將跨帳戶許可新增至 Amazon ECR 儲存庫。	若要將跨帳戶許可新增至中央的 Amazon ECR 儲存庫 AWS 帳戶，請執行下列程式碼： `{ "Version": "2008-10-17", "Statement": [{ "Sid": "LambdaECRImageRetrievalPolicy", "Effect": "Allow", "Principal": { "Service": "lambda.amazonaws.com" }, "Action": ["ecr:BatchGetImage", "ecr:GetDownloadUrlForLayer",], "Condition": { "StringLike": { "aws:sourceArn": "arn:aws:lambda:<Target_Region>:<Target_Account_ID>:function:*" } } }, { "Sid": "new statement", "Effect": "Allow", "Principal": {`	AWS DevOps

任務	Description	所需的技能

	<pre>"AWS": "arn:aws:iam::<Target_Account_ID>:root"
	},
	"Action": [
	"ecr:BatchGetImage",
	"ecr:GetDownloadUrlForLayer",
	],
	}
	]
	}</pre>	

在中央設定 GitHub OIDC 角色的角色 AWS 帳戶。	1. 將 AWS 設定為信任 GitHub 的 OIDC 作為聯合身分，其中包括將 GitHub OIDC 提供者新增至 AWS ，以及在 IAM 中設定角色和信任政策。若要這樣做，請遵循 GitHub 文件中在 Amazon Web Services 中設定 OpenID Connect 的指示。 2. 建立角色之後，請將必要的許可新增至角色。例如，新增 Amazon ECR 和 AWS Systems Manager 參數存放區的許可。如需詳細資訊，請參閱 IAM 文件中的設定 GitHub OIDC 身分提供者的角色。	AWS DevOps

任務	Description	所需的技能
引導目標中的 AWS 環境 AWS 帳戶。	在特定 中設定 CDK 環境 AWS 區域 ， AWS 帳戶 該環境可從中央帳戶啟用跨帳戶部署，並將最低權限原則套用至 CloudFormation 執行角色。 若要引導 AWS 環境，請執行下列命令： `cdk bootstrap aws://<Target_Account_ID>/<Target_Region> --trust <Central_Account_ID> --cloudformation-execution-policies arn:aws:iam::aws:policy/<Least_Privilege_Policy>`	AWS DevOps

任務	Description	所需的技能

授予中央 AWS 帳戶 OIDC 角色對目標 AWS 帳戶 引導角色的存取權。	CDK 引導會建立下列 IAM 角色，這些角色旨在由中央在 CDK 部署程序的各個階段 AWS 帳戶 中擔任：
	• 檔案發佈角色
	• 影像發佈角色
	• 查詢角色
	• 部署角色
	
	每個角色都有專為其用途量身打造的特定許可，並遵循最低權限原則。每個角色名稱`Target_Region` 中的 `Target_Account_ID` 和 都有助於指出這些角色在不同 AWS 帳戶 和 區域中是唯一的。此方法支援多帳戶、多區域設定中的清晰識別和管理。
	
	`Target Account CDK Bootstrap Roles`
	`arn:aws:iam::<Target_Account_ID>:role/cdk-deploy-role-<Target_Account_ID>-<Target_Region>`
	`arn:aws:iam::<Target_Account_ID>:role/cdk-file-publishing-role-<Target_Account_ID>-<Target_Region>`
	`arn:aws:iam::<Target_Account_ID>:role/cdk-image-publishing-role-<Target_Account_ID>-<Target_Region>`
		AWS DevOps

任務	Description	所需的技能

	<pre>arn:aws:iam::<Target
	_Account_ID>:role/
	cdk-lookup-role-<T
	arget_Account_ID>-
	<Target_Region></pre>
	• 更新中央帳戶中 OIDC 角色的許可政策，以授予它在目標帳戶中擔任角色的能力。此組態可跨不同 來部署 CDK 堆疊 AWS 帳戶。透過允許中央帳戶的 OIDC 角色採用目標帳戶的必要許可，您可以為跨帳戶 CDK 部署建立安全橋接。此方法可維持適當的存取控制，同時促進無縫的多帳戶基礎設施管理。
	
	若要更新中央 OIDC 角色的許可政策 AWS 帳戶，請使用下列程式碼：
	
	<pre>{
	    "Version":
	"2012-10-17",
	    "Statement": [
	        {
	            "Effect":
	"Allow",
	            "Action":
	"sts:AssumeRole",
	            "Resource":
	[
	
	"arn:aws:iam::<Tar
	get_Account_ID>:ro
	le/cdk-deploy-role-</pre>	

任務	Description	所需的技能
	<Target_Account_ID>- <Target_Region>", "arn:aws:iam::<Tar get_Account_ID>:role/ cdk-file-publishing- role-<Target_Account_I D>-<Target_Region>", "arn:aws:iam::<Tar get_Account_ID>:role/ cdk-image-publishing- role-<Target_Account_ ID>-<Target_Region>", "arn:aws:iam::<Tar get_Account_ID>:ro le/cdk-lookup-role- <Target_Account_ID>- <Target_Region>"] }] }	

建置 Docker 映像

任務	Description	所需的技能
複製專案儲存庫。	若要複製此模式的 GitHub 儲存庫，請執行下列命令： `git clone https://github.com/aws-samples/aws-cdk-golang-serverless-cicd-github-actions.git`	AWS DevOps

任務	Description	所需的技能
前往 Dockerfile 路徑。	若要導覽至 Dockerfile 路徑，請執行下列命令： `cd lambda`	AWS DevOps
使用 Amazon ECR 驗證 Docker。	Amazon ECR 需要安全存取您的私有容器儲存庫。以這種方式登入，即表示您允許本機機器或 CI/CD 環境上的 Docker 安全地與 Amazon ECR 互動。 若要使用 Amazon ECR 驗證 Docker，請執行下列命令： `aws ecr get-login -password --region <AWS_REGION> | docker login --username AWS --password-stdin <AWS_ACCOUNT_ID>.dkr.ecr.<AWS_REGION>.amazonaws.com` `AWS_Account_ID` 使用您的資訊修訂預留位置 `AWS_REGION` 和 。	AWS DevOps
建置 Docker 影像。	若要建置 Docker 映像，請執行下列命令： `docker build --platform linux/arm64 -t sample-app .`	AWS DevOps

任務	Description	所需的技能

標記並推送 Docker Image。	若要標記 Docker 映像並將其推送至 Amazon ECR 儲存庫，請執行下列命令：
	`docker tag sample-app:latest <AWS_ACCOUNT_ID>.dkr.ecr.<AWS_REGION>.amazonaws.com/<ECR_REPOSITORY>:<DOCKER_TAG>`
	`docker push <AWS_ACCOUNT_ID>.dkr.ecr.<AWS_REGION>.amazonaws.com/<ECR_REPOSITORY>:<DOCKER_TAG>`
	使用DOCKER_TAG 您的資訊修訂預留位置 AWS_Account_ID 、ECR_REPOSITORY 、 AWS_REGION 和 。
		AWS DevOps

部署 AWS CDK 應用程式

任務	Description	所需的技能
使用環境特定變數來合成 CDK 堆疊。	若要產生 CDK 程式碼中定義的基礎設施 CloudFormation 範本，請執行下列命令： `ENV=<environment> IMAGETAG=<image_tag> ECR_ARN=<ecr_repo_arn> cdk synth`	AWS DevOps

任務	Description	所需的技能
	使用您的資訊修訂下列預留位置： • `environment` – 以特定環境名稱取代，例如 `dev`、`staging`或 `prod`。 • `image_tag` – 將 取代為 Docker 影像的特定標籤，例如 `v1.0.0`或 `latest`。 • `ecr_repo_arn` – 將 取代為 Amazon ECR 儲存庫的 Amazon Resource Name (ARN)。	
部署 CDK 堆疊。	若要將 CDK 堆疊部署到您的 AWS 帳戶，請執行下列命令。`--require-approval never` 旗標表示 CDK 將自動核准並執行所有變更。這包括 CDK 通常會標記為需要手動檢閱的變更（例如 IAM 政策變更或移除資源）。在生產環境中使用 `--require-approval never`旗標之前，請確定您的 CDK 程式碼和 CI/CD 管道已經過良好測試且安全。 `ENV=<environment> IMAGETAG=<image_tag> ECR_ARN=<ecr_repo_arn> cdk deploy --require-approval never`	AWS DevOps

使用 GitHub 動作工作流程自動化 CI/CD

任務	Description	所需的技能

建立功能分支，並新增您的變更。	使用您先前建立的複製儲存庫、建立功能分支，然後將您的變更新增至應用程式程式碼。使用下列命令：
	`git checkout -b <feature_branch>`
	`git add .`
	`git commit -m "add your changes"`
	`git push origin <feature_branch>`
	以下是變更的範例：
	• Lambda 函數邏輯的變更
	• 將新功能新增至 Lambda 程式碼
	• 修正錯誤或最佳化 Lambda 函數中的現有程式碼
	GitHub 動作將使用可重複使用的工作流程並觸發 CI/CD 管道。
		AWS DevOps

合併您的變更。	建立提取請求，並將您的變更合併至主要。	AWS DevOps

疑難排解

問題	解決方案
`AccessDenied` 在 之間部署資源時發生錯誤 AWS 帳戶，例如 `AccessDenied: User not authorized to perform: "sts:AssumeRole"` 。	為了協助解決此問題，請執行下列動作來驗證跨帳戶許可： • 確定跨帳戶部署具有必要的 IAM 角色和政策。 • 檢查assume角色許可是否已正確設定。
相容性問題，因為版本不相符，例如，CDK 版本過期的`undefined: awscdkStack` 錯誤。	為了協助解決此問題，請執行下列動作，以確認您使用的是必要的 AWS CDK 和 Go 版本： • 請確定您使用的是 AWS CDK 和 Go 的相容版本。 • 檢查是否有已知問題或最近版本是否有重大變更。
例如，`Error: No such file or directory` 由於不正確的 YAML 組態或受保護`Permission denied` 的分支，CI/CD 管道失敗。	為了協助解決 GitHub 動作組態的問題，請確認可重複使用的工作流程已正確參考和設定。

相關資源

AWS 資源

- AWS 安全性、身分與合規的最佳實務
- AWS CDK 研討會
- AWS 雲端開發套件程式庫
- 使用容器映像建立 Lambda 函數
- Amazon Elastic Container Registry 的 Identity and Access Management
- 在 Go 中使用 AWS CDK

其他資源

- 在 Amazon Web Services 中設定 OpenID Connect (GitHub 文件)
- Golang 文件
- GitHub 動作的快速入門 (GitHub 文件)
- 重複使用工作流程 (GitHub 文件)

使用 GitHub 動作根據 AWS CloudFormation 範本佈建 AWS Service Catalog 產品

Ashish Bhatt 和 Ruchika Modi，Amazon Web Services

摘要

此模式為組織提供了一種簡化的方法，使用AWS Service Catalog產品和產品組合跨 AWS 服務 團隊佈建標準化和合規。 AWS CloudFormation有助於結合 Service Catalog 產品和產品組合中的基本元件，以佈建基礎網路基礎設施 AWS 雲端。此模式也會使用 GitHub 動作，將基礎設施即程式碼 (IaC) 整合到自動化開發工作流程中，藉此提升 DevOps 實務。

AWS Service Catalog 可讓組織在 上建立和管理核准的 IT 服務 AWS，提供標準化、集中式控制、自助式佈建和成本管理等優勢。透過 GitHub Actions 自動化 Service Catalog 產品組合和產品的部署，公司可以執行下列動作：

- 實現一致且可重複的部署。
- 使用 IaC 的版本控制。
- 整合雲端資源管理與現有的開發工作流程。

此組合可簡化雲端操作、強制執行合規，並加速交付核准的服務，同時減少手動錯誤並改善整體效率。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 存取 GitHub 儲存庫
- 對 AWS CloudFormation 和 的基本了解 AWS Service Catalog
- 用於託管 CloudFormation 範本的 Amazon Simple Storage Service (Amazon S3) 儲存貯體
- 名為 `github-actions`的 AWS Identity and Access Management (IAM) 角色，用於 GitHub 和 之間的連線 AWS

限制

- 此模式的可重複使用程式碼已僅使用 GitHub 動作進行測試。

- 有些 AWS 服務 無法全部使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

此模式的解決方案是透過使用下列 GitHub Marketplace 動作及其個別版本所建立：

- `actions/checkout@v4`
- `aws-actions/configure-aws-credentials@v2`
- `aws-actions/aws-cloudformation-github-deploy@v1.2.0`

Architecture

下圖顯示此解決方案的架構。

1. 管理員或平台工程師會將標準化 CloudFormation 範本推送至 GitHub 儲存庫，其中會維護範本。GitHub 儲存庫也包含工作流程，可自動佈建 AWS Service Catalog 使用 GitHub 動作的 。
2. GitHub 動作會使用 OpenID Connect (OIDC) 供應商來佈建 Service Catalog 來觸發連線至 AWS 雲端 的工作流程。
3. Service Catalog 包含開發人員可以直接用來佈建標準化 AWS 資源的產品組合和產品。此模式會綁定 AWS 資源，例如虛擬私有雲端 (VPCs)、子網路、NAT 和網際網路閘道，以及路由表。
4. 開發人員建立 Service Catalog 產品後，Service Catalog 會將其轉換為預先設定且標準化 AWS 的資源。因此，開發人員可以節省時間，因為他們不需要佈建個別資源並手動設定。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。它是基礎設施即程式碼 (IaC) 服務，可輕鬆用作其中一種產品類型 AWS Service Catalog。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Service Catalog 可協助您集中管理已核准的 IT 服務目錄 AWS。最終使用者可在機構所設的限制範圍內，迅速地只部署自己需要且經核准的 IT 服務。

- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他

- GitHub Actions 是與 GitHub 儲存庫緊密整合的持續整合和持續交付 (CI/CD) 平台。您可以使用 GitHub 動作來自動化建置、測試和部署管道。

程式碼儲存庫

此模式的程式碼可在 GitHub service-catalog-with-github-actions 儲存庫中使用。儲存庫包含下列感興趣的檔案：

- `github/workflows`:
  - `e2e-test.yaml` – 此檔案會呼叫 `workflow.yaml`，這是可重複使用的工作流程。一旦有遞交並推送至分支，就會觸發此工作流程。
  - `workflow.yaml` – 此檔案包含此解決方案的可重複使用工作流程，並使用 設定為`workflow_call`其觸發條件。作為可重複使用的工作流程， `workflow.yaml` 可以從任何其他工作流程呼叫。
- `templates`:
  - `servicecatalog-portfolio.yaml` – 此 CloudFormation 範本包含佈建 Service Catalog 產品組合和 Service Catalog 產品的資源。範本包含一組參數，用於佈建 Service Catalog 產品組合和產品。一個參數接受`vpc.yaml`範本上傳所在的 Amazon S3 檔案 URL。雖然此模式包含用來佈建 AWS 資源`vpc.yaml`的檔案，但您也可以使用參數 S3 檔案 URL 進行組態。
  - `vpc.yaml` – 此 CloudFormation 範本包含要在 Service Catalog product. AWS resources 中新增 AWS 的資源，包括 VPCs、子網路、網際網路閘道、NAT 閘道和路由表。`vpc.yaml` 範本是如何使用任何 CloudFormation 範本搭配 Service Catalog 產品和產品組合範本的範例。

最佳實務

- 請參閱 AWS Service Catalog 文件中的 的安全最佳實務 AWS Service Catalog。
- 請參閱 GitHub 文件中的 GitHub 動作安全強化。 GitHub

史詩

設定本機工作站

任務	Description	所需的技能
在您的本機工作站設定 Git。	若要在本機工作站上安裝和設定 Git，請使用 Git 文件中的入門 – 安裝 Git 說明。	應用程式開發人員
複製 GitHub 專案儲存庫。	若要複製 GitHub 專案儲存庫，請執行下列動作： 1. 開啟此模式的 GitHub 儲存庫 2. 選擇程式碼以查看複製選項，然後複製 HTTPS 索引標籤中提供的 URL。 3. 在工作站上為您的專案建立資料夾。 4. 開啟終端機，然後導覽至此資料夾 5. 若要複製 GitHub 儲存庫，請使用您在步驟 2 中複製的 URL 執行下列命令： `git clone https://github.com/aws-samples/service-catalog-with-github-actions.git` 6. 複製完成後，若要變更為專案資料夾中的複製儲存庫，請執行下列命令：	DevOps 工程師

任務	Description	所需的技能
	`cd <folder-name>/service-catalog-with-github-actions` 7. 在您選擇的整合式開發環境 (IDE) 中開啟專案。	

設定 OIDC 提供者

任務	Description	所需的技能
設定 OIDC 提供者。	建立 OpenID Connect (OIDC) 提供者，允許 GitHub 動作工作流程存取其中的資源 AWS，而不需要將 AWS 登入資料存放為長期的 GitHub 秘密。如需說明，請參閱 GitHub 文件中的在 Amazon Web Services 中設定 OpenID Connect。 設定 OIDC 供應商後，先前在先決條件中`github-actions` 提到的 IAM 角色 的信任政策將會更新。	AWS 管理員、AWS DevOps、一般 AWS

觸發 GitHub 動作管道以部署 Service Catalog 產品組合和產品

任務	Description	所需的技能
更新 `e2e-test.yaml` 。	`e2e-test.yaml` 檔案會在 觸發可重複使用的工作流程`workflow.yaml` 。在 中更新並驗證下列輸入參數的值`e2e-test.yaml` ：	DevOps 工程師

任務	Description	所需的技能

	• `aws_account_id` – 指定正確的 AWS 帳戶。
	• `aws_region` – 指定正確的 AWS 區域。
	• `s3BucketName` – 指定 Amazon S3 儲存貯體以保留 CloudFormation 範本。
	• 工作流程檔案需要兩個 IAM 角色做為輸入：
	  • `LaunchConstraintRole` - 最終使用者啟動、更新或終止產品時 AWS Service Catalog 擔任的 IAM 角色。
	  • `PrincipalArn` - 將與 Service Catalog 產品組合建立關聯的委託人 (IAM 使用者、角色或群組） 的 Amazon Resource Name (ARN)。如果 `PrincipalType` 是 `IAM`，則支援的值是完整定義的 IAM Amazon Resource Name (ARN)。如果 `PrincipalType` 是 `IAM_PATTERN` ，則支援的值是沒有 `AccountID` 的 IAM ARN，格式如下：`arn:partition:iam::resource-type/resource-id`	

驗證部署

任務	Description	所需的技能
驗證 Service Catalog 資源。	若要驗證 Service Catalog 資源，請執行下列動作： 1. 登入 AWS 管理主控台 的 AWS 帳戶，並驗證 AWS 區域 是否正確。 2. 導覽至管理、產品組合，AWS Service Catalog並驗證產品組合是否存在。 3. 選擇產品組合，並驗證產品、限制條件和存取標籤上的資訊。	AWS DevOps

清除資源

任務	Description	所需的技能
刪除 CloudFormation 堆疊。	若要刪除 CloudFormation 堆疊，請執行下列動作： 1. 在 https：//https://console.aws.amazon.com/cloudformation 開啟 AWS CloudFormation 主控台。 2. 在畫面頂端的導覽列上，選擇 AWS 區域 堆疊所在的 。 3. 在堆疊頁面中，選取要刪除的堆疊。此堆疊目前必須正在執行。	DevOps 工程師、AWS 管理員

任務	Description	所需的技能
	4. 在 stack details (堆疊詳細資訊) 窗格中，選擇 Delete (刪除)。 5. 當系統提示時，選取 Delete stack (刪除堆疊)。 如需詳細資訊，請參閱 CloudFormation 文件中的從 CloudFormation 主控台刪除堆疊 CloudFormation	

疑難排解

問題	解決方案

`e2e-test`
 	
`Can't find 'action.yml', 'action.yaml' or 'Dockerfile' under '*/home/runner/work/service-catalog-with-github-actions/service-catalog-with-github-actions`
 	
`Did you forget to run actions/checkout before running your local action?`
	若要確保您已啟用正確的儲存庫設定，請執行下列動作：
	
	1. 導覽至 Github 儲存庫的設定索引標籤。
	2. 從左側的功能表中選擇動作、一般
	3. 前往存取區段，然後選取可從 'XXX' 組織中的儲存庫存取的選項。

相關資源

AWS 文件

- Service Catalog 概觀

其他資源

- 關於觸發工作流程的事件 (GitHub 文件)
- 重複使用工作流程 (GitHub 文件)

其他資訊

若要查看與 Epics 相關的螢幕擷取畫面，請前往此模式 GitHub 儲存庫中的映像資料夾。可用的螢幕擷取畫面如下：

- AWS Service Catalog 產品組合，管理區段
- AWS Service Catalog product， 管理區段
- AWS Service Catalog product、User/Provisioning 區段

透過部署角色販賣機解決方案來佈建最低權限的 IAM 角色

Benjamin Morris、Nima Fotouhi、Aman Kaur Gandhi 和 Chad Moon，Amazon Web Services

總結

管道的超出範圍 AWS Identity and Access Management (IAM) 角色許可可能會對組織帶來不必要的風險。開發人員有時會在開發期間授予廣泛的許可，但在對程式碼進行故障診斷後忽略縮小許可範圍。這會導致一個問題，其中強大的角色在沒有業務需求的情況下存在，並且可能從未經過安全工程師的審核。

此模式提供此問題的解決方案：角色販賣機 (RVM)。使用安全且集中的部署模型，RVM 示範如何為個別 GitHub 儲存庫的管道佈建最低權限的 IAM 角色，而開發人員只需最少的努力。由於 RVM 是集中式解決方案，因此您可以將安全團隊設定為必要的檢閱者，以核准變更。此方法可讓安全性拒絕過度許可的管道角色請求。

RVM 接受 Terraform 程式碼做為輸入，並產生管道就緒的 IAM 角色做為輸出。所需的輸入是 AWS 帳戶 ID、GitHub 儲存庫名稱和許可政策。RVM 使用這些輸入來建立角色的信任政策和許可政策。產生的信任政策允許指定的 GitHub 儲存庫擔任角色，並將其用於管道操作。

RVM 使用 IAM 角色 （在引導期間設定）。此角色具有在組織中每個帳戶中擔任role-provisioning-role的許可。角色是透過 AWS Control Tower Account Factory for Terraform (AFT) 或 AWS CloudFormation StackSets 設定。role-provisioning-roles是實際為開發人員建立管道角色的角色。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- GitHub 組織，用於透過 GitHub 動作將基礎設施部署為程式碼 (IaC)。(不需要 GitHub Enterprise/Premium/Ultimate。)
- 多帳戶 AWS 環境。此環境不需要是 的一部分 AWS Organizations。
- 在 all 中部署 IAM 角色的機制 AWS 帳戶 （例如，AFT 或 CloudFormation StackSets)。
- 已安裝並設定 Terraform 1.3 版或更新版本。
- 已安裝並設定 Terraform AWS Provider 第 4 版或更新版本。

限制

- 此模式的程式碼專屬於 GitHub Actions 和 Terraform。不過，模式的一般概念可以在其他持續整合和交付 (CI/CD) 架構中重複使用。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

下圖說明此模式的工作流程。

角色自動販賣機的典型使用工作流程包含下列步驟：

1. 開發人員將包含新請求 IAM 角色 Terraform 程式碼的程式碼推送至 RVM GitHub 儲存庫。此動作會觸發 RVM GitHub 動作管道。
2. 管道使用 OpenID Connect (OIDC) 信任政策來擔任 RVM 角色擔任角色。
3. 當 RVM 管道執行時，它會在佈建開發人員新 IAM 角色的帳戶中擔任 RVM 工作流程角色。（使用 AFT 或 CloudFormation StackSets.）
4. RVM 會建立具有適當許可和信任的開發人員 IAM 角色，讓其他應用程式管道可以擔任該角色。
5. 應用程式開發人員可以設定其應用程式管道，以擔任此 RVM 佈建的角色。

建立的角色包含開發人員請求的許可和`ReadOnlyAccess`政策。該角色只能由針對開發人員指定儲存庫`main`分支執行的管道擔任。此方法有助於確保可能需要分支保護和檢閱才能使用角色。

自動化和擴展

最低權限許可需要注意佈建的每個角色的詳細資訊。此模型可降低建立這些角色所需的複雜性，讓開發人員無需額外的學習或精力即可建立所需的角色。

工具

AWS 服務

- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。

其他工具

- Git 是一種開放原始碼的分散式版本控制系統。它包含建立組織帳戶的能力。
- GitHub Actions 是與 GitHub 儲存庫緊密整合的持續整合和持續交付 (CI/CD) 平台。您可以使用 GitHub 動作來自動化建置、測試和部署管道。
- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。

程式碼儲存庫

此模式的程式碼可在 GitHub role-vending-machine 儲存庫中使用。

最佳實務

- 以正確的方式簡單又困難 – 輕鬆執行正確的動作。如果開發人員在 RVM 佈建程序中遇到困難，他們可能會嘗試透過其他方式建立角色，這會破壞 RVM 的中心性質。確保您的安全團隊提供有關如何安全有效地使用 RVM 的明確指導。

  您也應該讓開發人員難以做錯事。使用服務控制政策 SCPs) 或許可界限來限制哪些角色可以建立其他角色。這種方法有助於將角色建立限制為僅 RVM 和其他信任的來源。
- 提供良好的範例 – 不可避免地，某些開發人員會將 RVM 儲存庫中的現有角色調整為非正式範本，以授予其新角色的許可。如果您有可從中複製的最低許可範例，這可以降低開發人員請求廣泛、萬用字元密集許可的風險。如果您從具有大量萬用字元的高度許可角色開始，該問題可能會隨著時間的推移而增加。
- 使用命名慣例和條件 – 即使開發人員不知道其應用程式將建立的所有資源名稱，他們仍應該使用命名慣例來限制角色許可。例如，如果他們正在建立 Amazon S3 儲存貯體，其資源金鑰的值可能看起來像這樣，`arn:aws:s3:::myorg-myapp-dev-*`因此其角色在儲存貯體之外沒有符合該名稱的許可。透過 IAM 政策強制執行命名慣例可提高符合命名慣例的額外好處。由於不允許建立不相符的資源，因此會發生此改善。
- 需要提取請求 (PR) 檢閱 – RVM 解決方案的值是建立一個中央位置，其中可以檢閱新的管道角色。不過，此設計只有在有護欄有助於確保將安全、高品質的程式碼遞交至 RVM 時才有用。保護用於部署程式碼的分支 （ 例如 `main`) 免於直接推送，並需要核准以它們為目標的任何合併請求。
- 設定唯讀角色 – 根據預設，RVM 會為每個請求的角色佈建一個`readonly`版本。此角色可用於不會寫入資料的 CI/CD 管道，例如`terraform plan`管道工作流程。如果唯讀工作流程行為錯誤，此方法有助於防止不必要的變更。

根據預設， AWS 受管ReadOnlyAccess政策會同時連接至唯讀角色和讀寫角色。此政策可減少在決定所需許可時需要反覆運算的需求，但對某些組織而言可能過於寬鬆。如果需要，您可以從 Terraform 程式碼中移除政策。

- 授予最低許可 – 遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。

史詩

準備環境

任務	Description	所需的技能

將範例儲存庫複製到您的 GitHub 組織。	將此模式的儲存庫複製或將此儲存庫分支到您的 GitHub 組織，以便您可以根據需求進行調整。
	• 如果您選擇複製此儲存庫，您可以使用下列命令：
	`git clone https://github.com/aws-samples/role-vending-machine`
	• 如果您選擇將儲存庫授與 GitHub 組織，您可以使用下列命令：
	`gh repo fork aws-samples/role-vending-machine --org YOUR_ORGANIZATION_NAME`
		DevOps 工程師

判斷 RVM AWS 帳戶 的 。	決定 AWS 帳戶 要用於 RVM 的基礎設施部署。請勿使用 管理或根帳戶。	雲端架構師

任務	Description	所需的技能
（選用）允許組織的管道建立 PRs。	**Note** 只有在您想要允許`generate_providers_and_account_vars` 工作流程建立 PRs時，才需要此步驟。 若要允許組織的管道建立 PRs，請使用下列步驟： 1. 前往 `https://github.com/organizations/YOUR_ORG/settings/actions` 。 2. 選取允許 GitHub 動作以建立和核准提取請求。 如需詳細資訊，請參閱 GitHub 文件中的管理儲存庫的 GitHub 動作設定。 GitHub	DevOps 工程師

任務	Description	所需的技能
將唯讀許可授予 RVM 帳戶。	在管理帳戶中建立授予 RVM 帳戶唯讀許可的委派政策。這可讓您的 RVM GitHub 工作流程在`generate_providers_and_account_vars.py` 指令碼執行時動態提取 AWS 組織的帳戶清單。 使用下列程式碼，並以您在步驟 2 中選取`<YOUR RVM Account ID>`的 AWS 帳戶 ID 取代 ： `{ "Version": "2012-10-17", "Statement": [{ "Sid": "Statement", "Effect": "Allow", "Principal": { "AWS": "arn:aws:iam::<YOUR RVM Account ID>:root" }, "Action": ["organizations:ListAccounts", "organizations:DescribeOrganization", "organizations:DescribeOrganizationalUnit", "organizations:ListRoots", "organizations:ListAWSServi`	雲端管理員

任務	Description	所需的技能
	ceAccessForOrganiz ation", "organiza tions:ListDelegate dAdministrators"], "Resource": "*" }] }	
從範例儲存庫更新預設值。	若要將 RVM 設定為在特定環境中操作 AWS 區域，請執行下列動作： 1. 使用您操作所在的適當區域更新 `scripts/generate_providers_and_account_vars.py` 和其他檔案（例如 `bootstrap` 資料夾）。 2. 使用您要指定的 AWS 帳戶 ID、區域和任何其他變數來更新`.github/workflows/.env` 檔案。	DevOps 工程師

初始化基礎設施

任務	Description	所需的技能
引導 RVM 儲存庫。	此步驟是建立 RVM 管道本身使用的 OIDC 信任和 IAM 角色的必要步驟，因此可以開始操作和販賣其他角色。	DevOps 工程師

任務	Description	所需的技能
	在您的 RVM 帳戶內容中，從 `scripts/bootstrap` 目錄手動執行`terraform apply`命令。根據變數文件提供任何必要的值。	

設定操作

任務	Description	所需的技能
將 `github-workflow-rvm` 和 `github-workflow-rvm-readonly` 角色部署到所有 帳戶。	選擇符合您組織實務的部署方法，例如 AFT 或 StackSets。使用該方法將`scripts/assumed_role/main.tf` 檔案中的兩個 IAM 角色（預設名稱`github-workflow-rvm` 和 `github-workflow-rvm-readonly`）部署到您希望 RVM 能夠建立管道角色的每個帳戶。 這些 IAM 角色具有信任政策，允許 RVM 帳戶的角色擔任角色（或其`readonly`同等角色）擔任該角色。這些角色也有 IAM 許可政策，允許他們讀取和寫入（除非使用`readonly`角色）符合 的角色`github-workflow-role-*` 。	AWS 管理員
執行`generate_providers_and_account_vars` 工作流程。	若要設定 RVM 以準備好建立管道角色，請執行下列動作：	DevOps 工程師

任務	Description	所需的技能
	1. 使用 workflow_dispatch 執行`generate_providers_and_account_vars` 工作流程。 2. 合併工作流程建立的 PR。 工作流程完成後，RVM 已準備好： • 接受新管道角色的請求。 • 依要求建立唯讀或讀寫角色。 • 將角色部署到指定的 AWS 帳戶。	

疑難排解

問題	解決方案
我使用 RVM 建立角色，但 GitHub 無法擔任該角色。	確認 GitHub 儲存庫的名稱符合提供給`github_workflow_roles` 模組的名稱。角色的範圍是讓只有一個儲存庫可以擔任這些角色。 同樣地，請確認 GitHub 管道中使用的分支符合提供給`github_workflow_roles` 模組的分支名稱。一般而言，具有寫入許可的 RVM 建立角色只能由範圍為`main`分支的工作流程使用（也就是從 取得的部署`main`)。
我的唯讀角色無法執行其管道，因為它缺少讀取特定資源的許可。	雖然`ReadOnlyAccess` 政策提供廣泛的唯讀許可，但政策沒有某些讀取動作（例如，某些 AWS Security Hub CSPM 動作）。

問題	解決方案
	您可以使用 `github-workflow-roles` 模組的 `inline_policy_readonly` 參數來新增特定動作許可。

相關資源

- 使用 AWS CloudFormation StackSets 的最佳實務
- 使用多個帳戶組織您的 AWS 環境
- 適用於 Terraform (AFT) AWS Control Tower 的帳戶工廠概觀
- 政策最佳實務

其他資訊

使用 GitHub 環境

GitHub 環境是分支型角色存取限制的替代方法。如果您偏好使用 GitHub 環境，下列是 IAM 信任政策中其他條件的語法範例。此語法指定只有在 GitHub 動作在`Production`環境中執行時，才能使用角色。

```
"StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:octo-org/octo-
repo:environment:Production"
}
```

範例語法使用下列預留位置值：

- `octo-org` 是 GitHub 組織名稱。
- `octo-repo` 是儲存庫名稱。
- `Production` 是特定的 GitHub 環境名稱。

將 Amazon CloudWatch 指標發佈至 CSV 檔案

Abdullahi Olaoye，Amazon Web Services

總結

此模式使用 Python 指令碼擷取 Amazon CloudWatch 指標，並將指標資訊轉換為逗號分隔值 (CSV) 檔案，以提高可讀性。指令碼採用 AWS 服務，其指標應擷取為必要的引數。您可以將 AWS 區域和 AWS 登入資料設定檔指定為選用引數。如果您未指定這些引數，則指令碼會使用為執行指令碼的工作站設定的預設區域和設定檔。指令碼執行後，它會在相同的目錄中產生並儲存 CSV 檔案。

如需此模式提供的指令碼和相關檔案，請參閱附件一節。

先決條件和限制

先決條件

- Python 3.x
- AWS 命令列界面 (AWS CLI)

限制

指令碼目前支援下列 AWS 服務：

- AWS Lambda
- Amazon Elastic Compute Cloud (Amazon EC2)
  - 根據預設，指令碼不會收集 Amazon Elastic Block Store (Amazon EBS) 磁碟區指標。若要收集 Amazon EBS 指標，您必須修改連接的`metrics.yaml`檔案。
- Amazon Relational Database Service (Amazon RDS)
  - 不過，指令碼不支援 Amazon Aurora。
- Application Load Balancer
- Network Load Balancer
- Amazon API Gateway

工具

- Amazon CloudWatch 是一種監控服務，專為 DevOps 工程師、開發人員、網站可靠性工程師 (SREs) 和 IT 管理員而打造。CloudWatch 提供資料和可行的洞見，協助您監控應用程式、回應全系

統效能變更、最佳化資源使用率，以及取得營運運作狀態的統一檢視。CloudWatch 會以日誌、指標和事件的形式收集監控和操作資料，並提供在 AWS 和內部部署伺服器上執行的 AWS 資源、應用程式和服務統一檢視。

史詩

安裝和設定先決條件

任務	Description	所需的技能
安裝先決條件。	執行以下命令： `$ pip3 install -r requirements.txt`	開發人員
設定 AWS CLI。	執行以下命令： `$ aws configure`	開發人員

設定 Python 指令碼

任務	Description	所需的技能
開啟指令碼。	若要變更指令碼的預設組態，請開啟 `metrics.yaml` 。	開發人員
設定指令碼的期間。	這是要擷取的期間。預設期間為 5 分鐘 (300 秒）。您可以變更期間，但請注意下列限制： • 如果您指定的小時值介於 3 小時到 15 天前，請在此期間使用 60 秒 (1 分鐘） 的倍數。 • 如果您指定的小時值介於 15 小時到 63 天之間，請在此期	開發人員

任務	Description	所需的技能
	間使用 300 秒 (5 分鐘） 的倍數。 • 如果您指定的小時值大於 63 天，請在此期間使用 3，600 秒 (1 小時） 的倍數。 否則，API 操作不會傳回任何資料點。	
設定指令碼的時數。	此值指定您要擷取多少小時的指標。預設值為 1 小時。若要擷取多天的指標，請以小時為單位提供 值。例如，在 2 天內指定 48。	開發人員
變更指令碼的統計資料值。	（選用） 全域統計資料值為 `Average`，用於擷取未指派特定統計資料值的指標。指令碼支援統計資料值 `Maximum`、`SampleCount` 和 `Sum`。	開發人員

執行 Python 指令碼

任務	Description	所需的技能
執行指令碼。	使用下列命令： `$ python3 cwreport.py <service>` 若要查看服務值和選用 `region` 和 `profile` 參數的清單，請執行下列命令：	開發人員

任務	Description	所需的技能
	`$ python3 cwreport.py -h` 如需選用參數的詳細資訊，請參閱其他資訊一節。	

相關資源

- 設定 AWS CLI
- 使用 Amazon CloudWatch 指標
- Amazon CloudWatch 文件
- EC2 CloudWatch 指標
- AWS Lambda 指標
- Amazon RDS 指標
- Application Load Balancer 指標
- Network Load Balancer 指標
- Amazon API Gateway 指標

其他資訊

指令碼用量

```
$ python3 cwreport.py -h
```

語法範例

```
python3 cwreport.py <service> <--region=Optional Region> <--profile=Optional credential
 profile>
```

參數

- 服務（必要）– 您要執行指令碼的服務。指令碼目前支援這些服務：AWS Lambda、Amazon EC2、Amazon RDS、Application Load Balancer、Network Load Balancer 和 API Gateway。

- region （選用） – 要從中擷取指標的 AWS 區域。預設區域為 ap-southeast-1。
- 描述檔 （選用） – 要使用的 AWS CLI 命名描述檔。如果未指定此參數，則會使用預設設定的登入資料設定檔。

範例

- 若要使用預設區域ap-southeast-1和預設設定的登入資料來擷取 Amazon EC2 指標： `$ python3 cwreport.py ec2`
- 若要指定區域並擷取 API Gateway 指標： `$ python3 cwreport.py apigateway --region us-east-1`
- 若要指定 AWS 設定檔並擷取 Amazon EC2 指標： `$ python3 cwreport.py ec2 --profile testprofile`
- 若要同時指定區域和設定檔來擷取 Amazon EC2 指標： `$ python3 cwreport.py ec2 --region us-east-1 --profile testprofile`

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS Lambda 自動化 AWS 帳戶 AWS Managed Microsoft AD 從 移除 Amazon EC2 項目

Rahul Sharad Gaikwad 醫生和 Tamilselvan P，Amazon Web Services

總結

Active Directory (AD) 是一種 Microsoft 指令碼工具，可管理網域資訊和使用者與網路服務的互動。它在受管服務供應商 (MSPs中廣泛用於管理員工登入資料和存取許可。由於 AD 攻擊者可以使用非作用中帳戶來嘗試並入侵組織，因此請務必尋找非作用中的帳戶，並依照例行維護排程停用這些帳戶。使用 AWS Directory Service for Microsoft Active Directory，您可以執行 Microsoft Active Directory 做為受管服務。此模式可協助您設定 AWS Lambda 自動化，以快速尋找和移除非作用中的帳戶。

如果下列案例適用於您的組織，此模式可協助您：

- 集中式 AD 管理 – 如果您的組織有多個 AWS 帳戶，每個組織都有自己的 AD 部署，在所有帳戶中一致地管理使用者帳戶和存取許可可能具有挑戰性。使用跨帳戶 AD 清除解決方案，您可以集中方式停用或移除所有 AD 執行個體的非作用中帳戶。
- AD 重組或遷移 – 如果您的組織計劃重組或遷移其 AD 部署，跨帳戶 AD 清除解決方案可協助您準備環境。解決方案可協助您移除不必要或非作用中的帳戶、簡化遷移程序，並減少潛在的衝突或問題。

當您使用此模式時，可以獲得下列優點：

- 改善資料庫和伺服器效能，並修正非作用中帳戶的安全性漏洞。
- 如果您的 AD 伺服器託管在雲端，移除非作用中的帳戶也可以降低儲存成本，同時改善效能。您的每月帳單可能會減少，因為頻寬費用和運算資源可能會同時降低。
- 使用乾淨的 Active Directory 讓潛在的攻擊者屹立不搖。

先決條件和限制

先決條件

- 作用中的父帳戶 AWS 帳戶 和一或多個子帳戶。在此模式中，父帳戶是建立 Active Directory 的位置。子帳戶託管 Windows 伺服器，並透過父帳戶 Active Directory 加入。
- 在本機工作站上安裝https://git-scm.com/book/en/v2/Getting-Started-Installing-Git和設定 Git。
- 在本機工作站上安裝https://learn.hashicorp.com/tutorials/terraform/install-cli和設定 Terraform。

- AWS Managed Microsoft AD 在父帳戶中設定並共用給所有子帳戶的 目錄。如需詳細資訊，請參閱《 AWS Directory Service 管理指南》中的教學課程：共用您的 AWS Managed Microsoft AD 目錄以實現無縫 EC2 網域加入。
- 虛擬私有雲端 (VPC) 對等互連或 AWS Directory Service （ 父帳戶 ） 的 VPC 與 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體 （ 子帳戶 ） 的 VPC 之間可用的 AWS Transit Gateway 連線。如需詳細資訊，請參閱《 AWS Directory Service 管理指南》中的設定目錄擁有者與目錄消費者帳戶之間的 VPC 對等互連。
- 在所有父帳戶和子帳戶上設定`EC2WindowsUserdata`指令碼的 Windows 機器。指令碼檔案可在此模式程式碼儲存庫的根目錄中使用。
- 每個子帳戶上可用的跨帳戶 AWS Identity and Access Management (IAM) 角色，這些子帳戶已設定信任政策，以允許使用父帳戶中的 AWS Lambda 函數。如需詳細資訊，請參閱《Amazon EventBridge 使用者指南 AWS 帳戶 》中的在 Amazon EventBridge 中於 之間傳送和接收事件。EventBridge
- 下列秘密值可在父帳戶的 AWS Systems Manager 參數存放區中使用：
  - `domainJoinUser` – 目錄服務的使用者名稱
  - `domainJoinPassword` – 目錄服務的密碼

  如需秘密的詳細資訊，請參閱AWS Secrets Manager 《 使用者指南》中的建立 AWS Secrets Manager 秘密。

限制

- 在子帳戶中建立資源不會使用 Terraform 自動化。您必須使用 手動建立下列資源 AWS 管理主控台：
  - 將 Amazon EC2 終止事件傳送至父帳戶的 Amazon EventBridge 規則
  - 使用信任政策在子帳戶中建立 Amazon EC2 跨帳戶角色
  - VPC 對等互連或 Transit Gateway 連線
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

- Terraform 1.1.9 版或更新版本
- Terraform AWS Provider 3.0 版或更新版本

Architecture

下圖顯示解決方案的高階架構。

架構圖說明下列程序：

1. 在子帳戶中，EventBridge 規則會收集所有 Amazon EC2 終止事件。規則會將這些事件傳送至父帳戶中存在的 EventBridge。
2. EventBridge 會從父帳戶收集所有事件，並包含觸發 Lambda 函數 的規則`ADcleanup-Lambda`。
3. 父帳戶會從父帳戶或子帳戶接收任何終止事件，並觸發 Lambda 函數。
4. Lambda 函數會使用 Python boto 模組呼叫 Amazon EC2 Auto Scaling 群組，並取得隨機執行個體 ID。執行個體 ID 用於執行 Systems Manager 命令。
5. Lambda 函數會使用 boto 模組對 Amazon EC2 進行另一個呼叫。Lambda 函數會取得執行中 Windows 伺服器的私有 IP 地址，並將地址存放在暫時變數中。在步驟 5.1 和 5.2 中，從子帳戶收集執行中的 Windows EC2 執行個體。
6. Lambda 函數會呼叫 Systems Manager 以取得連線的電腦資訊 AWS Directory Service。
7. AWS Systems Manager 文件有助於在 Amazon EC2 Windows 伺服器上執行 PowerShell 命令，以取得連接到 AD 之電腦的私有 IP 地址。(Systems Manager 文件使用步驟 4 中取得的執行個體 ID。)
8. AD 網域使用者名稱和密碼存放在 AWS Systems Manager Parameter Store。 AWS Lambda Systems Manager 會呼叫 Parameter Store，並取得用來連線至 AD 的使用者名稱和密碼值。
9. 使用 Systems Manager 文件，PowerShell 指令碼會使用步驟 4 稍早取得的執行個體 ID 在 Amazon EC2 Windows 伺服器上執行。
10. Amazon EC2 AWS Directory Service 會使用 PowerShell 命令連線至 ，並移除未使用或非作用中的電腦。

工具

AWS 服務

- AWS Directory Service 提供多種搭配其他 使用 Microsoft Active Directory (AD) 的方式， AWS 服務 例如 Amazon Elastic Compute Cloud (Amazon EC2)、適用於 SQL Server 的 Amazon Relational Database Service (Amazon RDS) 和適用於 Windows File Server 的 Amazon FSx。
- AWS Directory Service for Microsoft Active Directory 可讓您的目錄感知工作負載 AWS 和資源在 中使用 Microsoft Active Directory AWS 雲端。

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如， AWS Lambda 函數、使用 API 目的地的 HTTP 呼叫端點，或其他事件匯流排 AWS 帳戶。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。使用 IAM，您可以指定誰或什麼可以存取其中的服務和資源 AWS、集中管理精細的許可，以及分析存取以縮小許可範圍 AWS。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Systems Manager 可協助您管理在 中執行的應用程式和基礎設施 AWS 雲端。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。
- AWS Systems Manager 文件定義 Systems Manager 在受管執行個體上執行的動作。Systems Manager 包含 100 多個預先設定的文件，可讓您用來在執行時間時指定參數。
- AWS Systems Manager 參數存放區是 的功能，可為組態資料管理和秘密管理 AWS Systems Manager 提供安全的階層式儲存。

其他工具

- HashiCorp Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。
- PowerShell 是在 Windows、Linux 和 macOS 上執行的 Microsoft 自動化和組態管理程式。
- Python 是一種一般用途的電腦程式設計語言。

程式碼儲存庫

此模式的程式碼可在 GitHub aws-lambda-ad-cleanup-terraform-samples 儲存庫中使用。

最佳實務

- 自動加入網域。 當您啟動屬於 Directory Service 網域的 Windows 執行個體時，請在執行個體建立程序期間加入網域，而不是稍後手動新增執行個體。若要自動加入網域，請在啟動新執行個體時，從網域加入目錄下拉式清單中選取正確的目錄。如需詳細資訊，請參閱《 Directory Service 管理指南》中的將 Amazon EC2 Windows 執行個體無縫加入 AWS Managed Microsoft AD Active Directory。

- 刪除未使用的帳戶。 在 AD 中尋找從未使用過的帳戶很常見。與保留在系統中的已停用或非作用中帳戶一樣，忽略未使用的帳戶可能會降低 AD 系統的速度，或讓您的組織容易受到資料外洩的影響。
- 自動化 Active Directory 清除。 為了協助降低安全風險並防止過時的帳戶影響 AD 效能，請定期執行 AD 清理。您可以撰寫指令碼來完成大多數 AD 管理和清除任務。範例任務包括移除已停用和非作用中的帳戶、刪除空和非作用中的群組，以及尋找過期的使用者帳戶和密碼。

史詩

設定子帳戶

任務	Description	所需的技能
在子帳戶中建立跨帳戶角色。	若要在子帳戶中建立跨帳戶角色，請執行下列動作： 1. 針對每個子帳戶，`ec2crossaccountrole` 使用名為的受管政策建立名為 的角色`AmazonEC2ReadOnlyAccess` 。（如需詳細資訊，請參閱《IAM 文件》中的使用自訂信任政策建立角色。） 2. 在自訂信任政策區段中，新增下列程式碼： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": {`	DevOps 工程師

任務	Description	所需的技能

	"Service": "ec2.amazonaws.com"
	},
	"Action": "sts:AssumeRole"
	},
	{
	"Effect": "Allow",
	"Principal": {
	"AWS": "arn:aws:iam::${Parentaccountid}:role/ADcleanuprole"
	},
	"Action": "sts:AssumeRole"
	}
	]
	}	

任務	Description	所需的技能
在子帳戶中建立事件規則。	若要為每個子帳戶建立 EventBridge 規則，請執行下列動作： 1. 登入子系 AWS 帳戶，然後開啟位於 https：//https://console.aws.amazon.com/events/ 的 Amazon EventBridge 主控台。 2. 在導覽窗格中，選擇規則。 3. 選擇建立規則。 4. 輸入名稱，並選擇性地輸入規則的描述。 5. 針對事件匯流排，選取 AWS 預設事件匯流排。 6. 針對規則類型， 選擇具有事件模式的規則。 7. 選擇下一步。 8. 針對事件模式，貼上下列程式碼： `{ "source": ["aws.ec2"], "detail-type": ["EC2 Instance State-change Notification"], "detail": { "state": ["terminated"] } }` 9. 選擇下一步。	DevOps 工程師

任務	Description	所需的技能
	10針對目標類型，選擇不同帳戶或區域中的事件匯流排。針對做為目標的事件匯流排，輸入父帳戶的事件匯流排 Amazon Resource Name (ARN)。 11針對執行角色，選擇為此特定資源建立新角色。 12選擇下一步以檢閱新規則的詳細資訊，然後選擇建立。 如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的在 Amazon EventBridge 中建立對事件做出反應的規則。 EventBridge	
建立 EC2 執行個體並加入 AD。	若要為 Windows 建立 EC2 執行個體，請執行下列動作： 1. 使用此模式程式碼儲存庫中可用的EC2WindowsUserdata 指令碼。 2. 在使用者資料指令碼中，修改下列程式碼以使用父帳戶中Directory service addresses 的值： `set-DnsClientServerAddress -InterfaceIndex 6 -ServerAddresses $(Directory service addresses)`	DevOps 工程師

設定本機工作站

任務	Description	所需的技能

建立專案資料夾並新增檔案。	若要複製儲存庫並建立專案資料夾，請執行下列動作：
	1. 開啟此模式的 GitHub 儲存庫。
	2. 選擇程式碼按鈕，以查看複製下拉式清單中要複製的選項。
	3. 在 HTTPS 索引標籤上，使用 Web URL 複製複製中提供的 URL。
	4. 在機器上建立資料夾，並使用專案名稱命名。
	5. 在本機電腦中開啟終端機，然後導覽至此資料夾。
	6. 若要複製 git 儲存庫，請使用下列命令。
	`git clone <repository-URL>.git`
	7. 複製儲存庫之後，請使用下列命令前往複製的目錄。
	`cd <directory name>/terraform-aws-lambda-ad-cleanup/multiple-account-cleanup`
	8. 在複製的儲存庫中，在您選擇的整合開發環境 (IDE) 中開啟此專案。
		DevOps 工程師

任務	Description	所需的技能
建置 adcleanup.zip 檔案。	若要壓縮lambda_function.py 檔案，請執行下列命令： zip -r adcleanup.zip lambda_function.py	DevOps 工程師

使用 Terraform 組態佈建目標架構

任務	Description	所需的技能
提供 Terraform 變數的值。	對於子帳戶，請在 terraform.tfvars 檔案中以字串類型提供下列arn變數的值： • lambda_env_cross_role_arn • child_account_cross_role_arn	DevOps 工程師
初始化 Terraform 組態。	若要初始化包含 Terraform 檔案的工作目錄，請執行下列命令： terraform init	DevOps 工程師
預覽變更。	您可以預覽 Terraform 在部署基礎設施之前對基礎設施所做的變更。若要驗證 Terraform 是否將視需要進行變更，請執行下列命令： terraform plan –-var-file=examples/terraform.tfvars	DevOps 工程師

任務	Description	所需的技能
執行提議的動作。	若要驗證`terraform plan`命令的結果是否如預期，請執行下列動作： 1. 執行下列命令：`terraform apply` 2. 登入 AWS 管理主控台，並驗證資源是否存在。	DevOps 工程師

驗證部署

任務	Description	所需的技能
執行並測試 Lambda 函數。	若要驗證部署是否成功執行，請執行下列動作： 1. 登入 AWS 管理主控台，然後開啟 Lambda 主控台。開啟函數頁面，然後選取以 ADcleanup-Lambda-* 開頭的函數名稱。 2. 在函數概觀頁面上，在程式碼來源區段的程式碼索引標籤上選擇測試。 3. 若要儲存測試事件，請提供事件的名稱，然後選擇儲存。若要測試事件，請再次選擇測試。 執行結果會顯示函數的輸出。	DevOps 工程師
檢視父帳戶 EventBridge 規則執行的結果。	若要檢視以父帳戶 Amazon EC2 終止事件為基礎的	DevOps 工程師

任務	Description	所需的技能
	EventBridge 規則結果，請執行下列動作： 1. 從父帳戶終止 EC2 執行個體。 2. 開啟父帳戶的 Lambda 主控台。開啟函數頁面，然後選取開頭為 ADcleanup-Lambda-* 的函數名稱。 3. 選擇監控索引標籤，然後選擇檢視 CloudWatch 日誌。 在 CloudWatch 主控台中，日誌群組頁面會顯示 Lambda 函數的結果。	
從子帳戶檢視 EventBridge 規則執行的結果。	若要檢視以子帳戶 Amazon EC2 終止事件為基礎的 EventBridge 規則結果，請執行下列動作： 1. 從子帳戶終止 EC2 執行個體。 2. 開啟父帳戶的 Lambda 主控台。開啟函數頁面，然後選取以 ADcleanup-Lambda-* 開頭的函數名稱。 3. 選擇監控索引標籤，然後選擇檢視 CloudWatch 日誌。 在 CloudWatch 主控台中，日誌群組頁面會顯示 Lambda 函數的結果。	DevOps 工程師

使用後清除基礎設施

任務	Description	所需的技能
清除基礎設施。	若要清除您建立的基礎設施，請使用下列命令： `terraform destroy` 若要確認`destroy`命令，請輸入 `yes`。	DevOps 工程師
清除後驗證 。	確認資源已成功移除。	DevOps 工程師

疑難排解

問題	解決方案
AWS Directory Service （父帳戶）與 Amazon EC2 執行個體（子帳戶）之間的連線問題 – 即使 VPC 互連可用，您也無法將子帳戶的電腦加入 AD。	在 VPCs中新增路由。如需說明，請參閱 AWS Directory Service 文件中的設定目錄擁有者與目錄消費者帳戶之間的 VPC 對等互連。

相關資源

AWS 文件

- Amazon EventBridge 和 AWS Identity and Access Management
- 設定 Systems Manager 所需的執行個體許可
- 的身分和存取管理 Directory Service
- Lambda 的身分型 IAM 政策
- 手動將 Amazon EC2 Windows 執行個體加入您的 AWS Managed Microsoft AD Active Directory
- 使用 AWS Lambda 自動化 AWS 帳戶 從 移除相同 中的 Amazon EC2 AWS Managed Microsoft AD 項目

其他資源

- AWS 提供者 (Terraform 文件）
- 後端組態 (Terraform 文件）
- 安裝 Terraform (Terraform 文件）
- Python boto 模組 (Python 套件索引儲存庫）
- Terraform 二進位下載 (Terraform 文件）

使用 AWS Lambda 自動化 AWS 帳戶 從 移除相同 中的 Amazon EC2 AWS Managed Microsoft AD 項目

Rahul Sharad Gaikwad 醫生和 Tamilselvan P，Amazon Web Services

總結

Active Directory (AD) 是一種 Microsoft 指令碼工具，可管理網域資訊和使用者與網路服務的互動。它在受管服務供應商 (MSPs中廣泛用於管理員工登入資料和存取許可。由於 AD 攻擊者可以使用非作用中帳戶來嘗試並入侵組織，因此請務必尋找非作用中的帳戶，並依照例行維護排程停用這些帳戶。使用 AWS Directory Service for Microsoft Active Directory，您可以執行 Microsoft Active Directory 做為受管服務。

此模式可協助您設定 AWS Lambda 自動化，以快速尋找和移除非作用中的帳戶。當您使用此模式時，可以獲得下列優點：

- 改善資料庫和伺服器效能，並修正非作用中帳戶的安全性漏洞。
- 如果您的 AD 伺服器託管在雲端，移除非作用中帳戶也可以降低儲存成本，同時改善效能。您的每月帳單可能會減少，因為頻寬費用和運算資源可能會同時降低。
- 使用乾淨的 Active Directory 讓潛在的攻擊者屹立不搖。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 在本機工作站上安裝https://git-scm.com/book/en/v2/Getting-Started-Installing-Git和設定 Git。
- 在本機工作站上安裝https://learn.hashicorp.com/tutorials/terraform/install-cli和設定 Terraform。
- 具有 Active Directory 模組的 Windows 電腦 (`ActiveDirectory`)。
- 中的目錄 AWS Managed Microsoft AD 和儲存在參數存放區中參數中的登入 AWS Systems Manager 資料。
- AWS Identity and Access Management 具有工具中 AWS 服務 所列 許可的 (IAM) 角色。如需 IAM 的詳細資訊，請參閱相關資源。

限制

- 此模式不支援跨帳戶設定。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

- Terraform 1.1.9 版或更新版本
- Terraform AWS 提供者 3.0 版或更新版本

Architecture

下圖顯示此模式的工作流程和架構元件。

該圖顯示以下工作流程：

1. Amazon EventBridge 會根據 Cron 表達式觸發 AWS Lambda 函數。（對於此模式，cron 表達式排程是每天一次。）
2. 必要的 IAM 角色和政策會透過 AWS Lambda Terraform 建立並連接至 。
3. AWS Lambda 函數會執行，並使用 Python boto 模組呼叫 Amazon Elastic Compute Cloud (Amazon EC2) Auto Scaling 群組。Lambda 函數會取得隨機執行個體 ID。執行個體 ID 用於執行 AWS Systems Manager 命令。
4. AWS Lambda 使用 boto 模組對 Amazon EC2 進行另一個呼叫，並取得執行中 Windows 伺服器的私有 IP 地址，並將地址存放在暫時變數中。
5. AWS Lambda 會再次呼叫 Systems Manager，以取得連線的電腦資訊 Directory Service。
6. AWS Systems Manager 文件有助於在 Amazon EC2 Windows 伺服器上執行 PowerShell 指令碼，以取得與 AD 連線之電腦的私有 IP 地址。
7. AD 網域使用者名稱和密碼存放在 AWS Systems Manager Parameter Store。 AWS Lambda Systems Manager 會呼叫 Parameter Store，並取得用於連接 AD 的使用者名稱和密碼值。
8. 使用 Systems Manager 文件，PowerShell 指令碼會使用步驟 3 稍早取得的執行個體 ID 在 Amazon EC2 Windows 伺服器上執行。
9. Amazon EC2 Directory Service 會使用 PowerShell 命令進行連線，並移除未使用或非作用中的電腦。

工具

AWS 服務

- AWS Directory Service 提供多種搭配其他 使用 Microsoft Active Directory (AD) 的方式， AWS 服務 例如 Amazon Elastic Compute Cloud (Amazon EC2)、適用於 SQL Server 的 Amazon Relational Database Service (Amazon RDS) 和適用於 Windows File Server 的 Amazon FSx。
- AWS Directory Service for Microsoft Active Directory 可讓您的目錄感知工作負載 AWS 和資源在 中使用 Microsoft Active Directory AWS 雲端。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如， AWS Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他事件匯流排 AWS 帳戶。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS資源的存取。使用 IAM，您可以指定誰或什麼可以存取其中的 服務和資源 AWS、集中管理精細許可，以及分析存取權以縮小許可範圍 AWS。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- AWS Systems Manager 可協助您管理在 中執行的應用程式和基礎設施 AWS 雲端。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。
- AWS Systems Manager 文件定義 Systems Manager 在受管執行個體上執行的動作。Systems Manager 包含 100 多個預先設定的文件，可讓您用來在執行時間時指定參數。
- AWS Systems Manager 參數存放區是 的功能， AWS Systems Manager 並提供安全的階層式儲存，用於組態資料管理和秘密管理。

其他工具

- HashiCorp Terraform 是一種開放原始碼基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。
- PowerShell 是在 Windows、Linux 和 macOS 上執行的 Microsoft 自動化和組態管理程式。
- Python 是一種一般用途的電腦程式設計語言。

Code 儲存庫

此模式的程式碼可在 GitHub Custom AD Cleanup Automation 解決方案儲存庫中找到。

最佳實務

- 自動加入網域。 當您啟動屬於 Directory Service 網域的 Windows 執行個體時，請在執行個體建立程序期間加入網域，而不是稍後手動新增執行個體。若要自動加入網域，請在啟動新執行個體時，從網域加入目錄下拉式清單中選取正確的目錄。如需詳細資訊，請參閱《 Directory Service 管理指南》中的將 Amazon EC2 Windows 執行個體無縫加入 AWS Managed Microsoft AD Active Directory。
- 刪除未使用的帳戶。 在 AD 中尋找從未使用過的帳戶很常見。與保留在系統中的已停用或非作用中帳戶一樣，忽略未使用的帳戶可能會降低 AD 系統的速度，或使您的組織容易受到資料外洩的影響。
- 自動化 Active Directory 清除。 為了協助降低安全風險並防止過時的帳戶影響 AD 效能，請定期執行 AD 清理。您可以撰寫指令碼來完成大多數 AD 管理和清除任務。範例任務包括移除已停用和非作用中的帳戶、刪除空白和非作用中的群組，以及尋找過期的使用者帳戶和密碼。

史詩

設定您的環境

任務	Description	所需的技能
建立專案資料夾，並新增檔案。	若要複製儲存庫並建立專案資料夾，請執行下列動作： 1. 開啟此模式的 GitHub 儲存庫。 2. 選擇程式碼按鈕，以查看複製下拉式清單中要複製的選項。 3. 在 HTTPS 索引標籤上，使用 Web URL 複製複製中提供的 URL。 4. 在機器上建立資料夾，並使用專案名稱命名。 5. 在本機電腦中開啟終端機，然後導覽至此資料夾。	DevOps 工程師

任務	Description	所需的技能

	6. 若要複製 git 儲存庫，請使用下列命令。
	
	`git clone <repository-URL>.git`
	7. 複製儲存庫之後，請使用下列命令前往複製的目錄。
	
	`cd <directory name>`
	8. 在複製的儲存庫中，在您選擇的整合開發環境 (IDE) 中開啟此專案。	

使用 Terraform 組態佈建目標架構

任務	Description	所需的技能
初始化 Terraform 組態。	若要初始化包含 Terraform 檔案的工作目錄，請執行下列命令。 `terraform init`	DevOps 工程師
預覽變更。	您可以預覽 Terraform 在部署基礎設施之前對基礎設施所做的變更。若要驗證 Terraform 是否將視需要進行變更，請執行下列命令。 `terraform plan`	DevOps 工程師
執行提議的動作。	若要驗證`terraform plan`命令的結果是否如預期，請執行下列動作： 1. 執行下列命令。	DevOps 工程師

任務	Description	所需的技能
	`terraform apply` 2. 登入 AWS 管理主控台，並確認資源是否存在。	
清除基礎設施。	若要清除您建立的基礎設施，請使用下列命令。 `terraform destroy` 若要確認銷毀命令，請輸入 `yes`。	DevOps 工程師

驗證部署

任務	Description	所需的技能
執行和測試 Lambda 函數。	若要驗證部署是否成功執行，請執行下列動作： 1. 登入 AWS 管理主控台 並開啟 主控台。開啟函數頁面，然後選取以 ADcleanup-Lambda-* 開頭的函數名稱。 2. 在函數概觀頁面上，在程式碼來源區段的程式碼索引標籤上選擇測試。 3. 若要儲存測試事件，請提供事件的名稱，然後選擇儲存。然後，若要測試事件，請再次選擇測試。 執行結果會顯示函數的輸出。	DevOps 工程師

任務	Description	所需的技能
檢視 Lambda 函數的結果。	在此模式中，EventBridge 規則每天執行一次 Lambda 函數。若要檢視 Lambda 函數的結果，請執行下列動作： 1. 登入 AWS 管理主控台 並開啟 AWS Lambda 主控台。開啟函數頁面，然後選取開頭為 ADcleanup-Lambda-* 的函數名稱。 2. 選擇監控索引標籤，然後選擇檢視 CloudWatch 日誌。 在 CloudWatch 主控台中，日誌群組頁面會顯示 Lambda 函數的結果。	DevOps 工程師

使用後清除基礎設施

任務	Description	所需的技能
清除基礎設施。	若要清除您建立的基礎設施，請使用下列命令。 `terraform destroy` 若要確認銷毀命令，請輸入 `yes`。	DevOps 工程師
清除後驗證。	確認資源已成功移除。	DevOps 工程師

疑難排解

問題	解決方案
如果您嘗試移除 AD 電腦，您會收到「拒絕存取」訊息。無法移除 AD 電腦，因為依預設，動作會嘗試移除做為 AD 服務一部分連線的兩個私有 IP 地址。	為避免此錯誤，當您列出 AD 電腦輸出與執行 Windows 之機器輸出之間的差異時，請使用下列 Python 操作忽略前兩部電腦。 `Difference = Difference[2:]`
當 Lambda 在 Windows 伺服器上執行 PowerShell 指令碼時，預期 Active Directory 模組預設為可用。如果模組無法使用，Lambda 函數會建立錯誤，指出「執行個體上未安裝 Get-AdComputer」。	若要避免此錯誤，請使用 EC2 執行個體的使用者資料安裝所需的模組。使用此模式 GitHub 儲存庫中的 EC2WindowsUserdata 指令碼。

相關資源

AWS 文件

- Amazon EventBridge 和 AWS Identity and Access Management
- 設定 Systems Manager 所需的執行個體許可
- 的身分和存取管理 Directory Service
- 手動將 Amazon EC2 Windows 執行個體加入您的 AWS Managed Microsoft AD Active Directory
- 在 中使用身分型 IAM 政策 AWS Lambda

其他資源

- AWS 供應商 (Terraform 文件)
- 後端組態 (Terraform 文件)
- 安裝 Terraform (Terraform 文件)
- Python boto 模組 (Python 套件索引儲存庫)
- Terraform 二進位下載 (Terraform 文件)

AWS Glue 使用 pytest 架構在 中執行 Python ETL 任務的單元測試

Praveen Kumar Jeyarajan 和 Vaidy Sankaran，Amazon Web Services

總結

您可以在 AWS Glue 本機開發環境中為 執行 Python 擷取、轉換和載入 (ETL) 任務的單元測試，但在 DevOps 管道中複寫這些測試可能既困難又耗時。當您在 AWS 技術堆疊上現代化大型主機 ETL 程序時，單元測試特別具挑戰性。此模式說明如何簡化單元測試，同時保持現有功能完整，避免在您發佈新功能時中斷關鍵應用程式功能，並維護高品質的軟體。您可以使用此模式中的步驟和程式碼範例，AWS Glue 在 中使用 pytest 架構來執行 Python ETL 任務的單元測試 AWS CodePipeline。您也可以使用此模式來測試和部署多個 AWS Glue 任務。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 您 AWS Glue 程式庫的 Amazon Elastic Container Registry (Amazon ECR) 映像 URI，從 Amazon ECR Public Gallery 下載
- 具有目標 AWS 帳戶 和 設定檔的 Bash 終端機 （ 在任何作業系統上 ） AWS 區域
- Python 3.10 或更新版本
- Pytest
- 用於測試的 Moto Python 程式庫 AWS 服務

Architecture

下圖說明如何將以 Python 為基礎的 AWS Glue ETL 程序單元測試納入典型的企業規模 AWS DevOps 管道。

該圖顯示以下工作流程：

1. 在來源階段中， AWS CodePipeline 會使用版本控制的 Amazon Simple Storage Service (Amazon S3) 儲存貯體來存放和管理原始程式碼資產。這些資產包括範例 Python ETL 任務 (`sample.py`)、單元測試檔案 (`test_sample.py`) 和 AWS CloudFormation 範本。然後，CodePipeline 會將最新的程式碼從主要分支傳輸到 AWS CodeBuild 專案，以供進一步處理。

2. 在建置和發佈階段，上一個來源階段的最新程式碼會在 AWS Glue 公有 Amazon ECR 映像的協助下進行單元測試。然後，測試報告會發佈至 CodeBuild 報告群組。適用於 AWS Glue 程式庫的公有 Amazon ECR 儲存庫中的容器映像包含在 AWS Glue 本機執行和單位測試 PySpark 型 ETL 任務所需的所有二進位檔。公有容器儲存庫有三個映像標籤，每個 支援的版本各一個 AWS Glue。基於示範目的，此模式會使用`glue_libs_4.0.0_image_01`映像標籤。若要在 CodeBuild 中使用此容器映像做為執行期映像，請複製對應至您要使用之映像標籤的映像 URI，然後更新`TestBuild`資源的 GitHub 儲存庫中的`pipeline.yml`檔案。
3. 在部署階段，會啟動 CodeBuild 專案，並在所有測試通過時，將程式碼發佈至 Amazon S3 儲存貯體。
4. 使用者在 `deploy`資料夾中使用 CloudFormation 範本部署 AWS Glue 任務。

工具

AWS 服務

- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- AWS Glue 是全受管 ETL 服務。它可協助您可靠地分類、清理、擴充和移動資料存放區和資料串流之間的資料。
- Amazon Simple Storage Service (Amazon S3) 是一種物件儲存服務，提供業界領先的可擴展性、資料可用性、安全性和效能。

其他工具

- Python 是一種高階、解譯的一般用途程式設計語言。
- Moto 是用於測試的 Python 程式庫 AWS 服務。
- Pytest 是一種用於撰寫小型單元測試的架構，可擴展以支援應用程式和程式庫的複雜功能測試。
- Python ETL 程式庫 AWS Glue 是 Python 程式庫的儲存庫，用於 PySpark 批次任務的本機開發 AWS Glue。

程式碼儲存庫

此模式的程式碼可在 GitHubaws-glue-jobs-unit-testing 儲存庫中使用。儲存庫包含下列資源：

- `src` 資料夾中以 Python 為基礎的 AWS Glue 任務範例
- `tests` 資料夾中相關聯的單元測試案例 （使用 pytest 架構建置）
- `deploy` 資料夾中的 CloudFormation 範本 （以 YAML 撰寫）

最佳實務

CodePipeline 資源的安全性

最佳實務是對連線至 CodePipeline 中管道的來源儲存庫使用加密和身分驗證。如需詳細資訊，請參閱 CodePipeline 文件中的安全最佳實務。

CodePipeline 資源的監控和記錄

最佳實務是使用 AWS 記錄功能來判斷使用者在帳戶中採取的動作，以及他們使用的資源。日誌檔案會顯示下列項目：

- 動作的時間和日期
- 動作的來源 IP 地址
- 哪些動作因許可不足而失敗

記錄功能可在 AWS CloudTrail 和 Amazon CloudWatch Events 中使用。您可以使用 CloudTrail 記錄 AWS API 呼叫，以及由 發出或代表您的 發出的相關事件 AWS 帳戶。如需詳細資訊，請參閱 CodePipeline 文件中的使用 記錄 CodePipeline API 呼叫 AWS CloudTrail。 CodePipeline

您可以使用 CloudWatch Events 來監控在 上執行 AWS 雲端 的資源和應用程式 AWS。您也可以在 CloudWatch Events 中建立提醒。如需詳細資訊，請參閱 CodePipeline 文件中的監控 CodePipeline 事件。 CodePipeline

史詩

部署原始程式碼

任務	Description	所需的技能
準備程式碼封存以進行部署。	1. `code.zip` 從 GitHubaws-glue-jobs-unit-testing 下	DevOps 工程師

任務	Description	所需的技能

	載，或使用命令列工具自行建立 .zip 檔案。例如，您可以在 Linux 或 Mac 上建立 .zip 檔案，方法是在終端機中執行下列命令：
	
	`git clone https://github.com/aws-samples/aws-glue-jobs-unit-testing.git`
	`cd aws-glue-jobs-unit-testing`
	`git checkout master`
	`zip -r code.zip src/ tests/ deploy/`
	
	2. 登入 AWS 管理主控台並選擇 AWS 區域 您所選的 。
	3. 建立 Amazon S3 儲存貯體，然後將 .zip 套件和`code.zip`檔案（先前已下載）上傳至您建立的 Amazon S3 儲存貯體。	

任務	Description	所需的技能
建立 CloudFormation 堆疊。	1. 登入 ， AWS 管理主控台 然後開啟 CloudFormation 主控台。 2. ChooseCreate 堆疊，然後選擇使用現有資源 （匯入資源）。 3. 在建立堆疊頁面的指定範本區段中，選擇上傳範本檔案，然後選擇pipeline.yml 範本 （從 GitHub 儲存庫下載）。然後選擇下一步。 4. 針對堆疊名稱，輸入 glue-unit-testing-pipeline，或選擇您選擇的堆疊名稱。 5. 對於 ApplicationStackName，請使用預先填入的 glue-codepipeline-app 名稱。這是管道建立的 CloudFormation 堆疊名稱。 6. 對於 BucketName，請使用預先填入的 aws-glue-artifacts-us-east-1 儲存貯體名稱。這是 Amazon S3 儲存貯體的名稱，其中包含 .zip 檔案，供管道用來存放程式碼成品。 7. 對於 CodeZipFile，請使用預先填入的 code.zip 值。這是範例程式碼 Amazon S3 物件的金鑰名稱。物件應為 .zip 檔案。 8. 對於 TestReportGroupName，請使用預先填入的 glue-	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	unittest-report 名稱。這是為存放單元測試報告而建立的 CodeBuild 測試報告群組名稱。 9. ChooseNext，然後在設定堆疊選項頁面上再次選擇下一步。 10.在檢閱頁面的功能下，選擇我確認 CloudFormation 可能會使用自訂名稱選項建立 IAM 資源。 11選擇提交。堆疊建立完成後，您可以在資源索引標籤上看到建立的資源。堆疊建立大約需要 5-7 分鐘。 堆疊會使用 Amazon S3 作為來源來建立 CodePipeline 檢視。在上述步驟中，管道為 aws-glue-unit-test-pipeline。	

執行單元測試

任務	Description	所需的技能
在管道中執行單元測試。	1. 若要測試部署的管道，請登入 AWS 管理主控台，然後開啟CodePipeline 主控台。 2. 選取 CloudFormation 堆疊建立的管道，然後選擇釋放變更。管道開始執行（使用 Amazon S3 儲存貯體中最新的程式碼）。	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	3. 在Test_and_Build 階段完成後，選擇Detailstab，然後檢查日誌。 4. 選擇報告索引標籤，然後從報告歷史記錄中選擇測試報告，以檢視單位測試結果。 5. 部署階段完成後，請在 AWS Glue 主控台上執行和監控部署 AWS Glue 的任務。如需詳細資訊，請參閱 AWS Glue 文件中的監控 AWS Glue。	

清除所有 AWS 資源

任務	Description	所需的技能
清除環境中的資源。	為了避免額外的基礎設施成本，請務必在實驗此模式中提供的範例之後刪除堆疊。 1. 開啟 CloudFormation 主控台，然後選取您建立的堆疊。 2. 選擇 刪除。這會刪除堆疊建立的所有資源，包括 AWS Identity and Access Management (IAM) 角色、IAM 政策和 CodeBuild 專案。	AWS DevOps，DevOps 工程師

疑難排解

問題	解決方案
CodePipeline 服務角色無法存取 Amazon S3 儲存貯體。	• 對於連接至 CodePipeline 服務角色的政策，請將 `s3:ListBucket` 新增至政策中的動作清單。如需檢視服務角色政策的指示，請參閱檢視管道 ARN 和服務角色 ARN（主控台）。編輯您服務角色的政策陳述式，如將許可新增至 CodePipeline 服務角色中詳述。 • 對於連接至管道 Amazon S3 成品儲存貯體的資源型政策，也稱為成品儲存貯體政策，請新增允許 CodePipeline 服務角色使用 `s3:ListBucket` 許可的陳述式。
CodePipeline 傳回 Amazon S3 儲存貯體未進行版本控制的錯誤。	CodePipeline 要求來源 Amazon S3 儲存貯體進行版本控制。在 Amazon S3 儲存貯體上啟用版本控制。如需說明，請參閱在儲存貯體上啟用版本控制。

相關資源

- AWS Glue
- 在本機開發和測試 AWS Glue 任務
- 的 AWS CloudFormation AWS Glue

其他資訊

此外，您可以使用 AWS Command Line Interface () 部署 AWS CloudFormation 範本AWS CLI。如需詳細資訊，請參閱 CloudFormation 文件中的使用轉換快速部署範本。

使用 AWS CodePipeline 和 AWS CDK 設定 CI/CD 管道

Konstantin Zarudaev、Yasa Dabas、Lars Kinder 和 Cizer Pereira，Amazon Web Services

首頁

透過持續整合和持續交付 (CI/CD) 自動化您的軟體建置和發行程序，支援可重複的建置並快速交付新功能給您的使用者。您可以快速輕鬆地測試每個程式碼變更，而且可以在發行軟體之前攔截和修正錯誤。透過預備和發行程序執行每項變更，您可以驗證應用程式或基礎設施程式碼的品質。CI/CD 體現了一種文化、一組操作原則和一系列實務，可協助應用程式開發團隊更頻繁、更可靠地交付程式碼變更。實作也稱為 CI/CD 管道。

此模式定義了 Amazon Web Services (AWS) 上與 AWS CodeCommit 儲存庫的可重複使用持續整合和持續交付 (CI/CD) 管道。AWS CodePipeline 管道使用 AWS 雲端開發套件 (AWS CDK) v2 撰寫。

使用 CodePipeline，您可以透過 AWS 管理主控台界面、AWS 命令列界面 (AWS CLI)、AWS CloudFormation 或 AWS SDKs，建立軟體版本程序的不同階段模型。此模式示範使用 AWS CDK 實作 CodePipeline 及其元件。除了建構程式庫之外，AWS CDK 還包含工具組 (CLI 命令 `cdk`)，這是與您的 AWS CDK 應用程式互動的主要工具。在其他函數中，工具組可讓您將一或多個堆疊轉換為 CloudFormation 範本，並將其部署至 AWS 帳戶。

管道包含驗證第三方程式庫安全性的測試，有助於確保指定環境中的快速自動發行。您可以透過進行驗證程序來提高應用程式的整體安全性。

此模式的目的是加速您使用 CI/CD 管道來部署程式碼，同時確保您部署的資源遵守 DevOps 最佳實務。實作範例程式碼之後，您會擁有 AWS CodePipeline，其中包含固定、測試、安全檢查、部署和部署後程序。此模式也包含 Makefile 的步驟。使用 Makefile，開發人員可以在本機重現 CI/CD 步驟，並提高開發程序的速度。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 對以下內容的基本了解：
  - AWS CDK
  - AWS CloudFormation
  - AWS CodePipeline

- TypeScript

限制

此模式僅針對 TypeScript 使用 AWS CDK。它不包含 AWS CDK 支援的其他語言。

產品版本

使用下列工具的最新版本：

- AWS 命令列界面 (AWS CLI)
- cfn_nag
- git-remote-codecommit
- Node.js

Architecture

目標技術堆疊

- AWS CDK
- AWS CloudFormation
- AWS CodeCommit
- AWS CodePipeline

目標架構

管道是由 AWS CodeCommit 儲存庫 () 中的變更觸發`SampleRepository`。在一開始，CodePipeline 會建置成品、自行更新，並啟動部署程序。產生的管道會將解決方案部署到三個獨立環境：

- 開發 – 作用中開發環境中的三步驟程式碼檢查
- 測試 – 整合和迴歸測試環境
- 產品 – 生產環境

開發階段中包含的三個步驟是內嵌、安全性和單元測試。這些步驟會平行執行以加速程序。為了確保管道僅提供有效的成品，每當程序中的步驟失敗時，就會停止執行。開發階段部署之後，管道會執行驗證

測試來驗證結果。如果成功，管道接著會將成品部署到測試環境，其中包含部署後驗證。最後一個步驟是將成品部署至 Prod 環境。

下圖顯示從 CodeCommit 儲存庫到 CodePipeline 執行的建置和更新程序的工作流程、三個開發環境步驟，以及三個環境中每個環境的後續部署和驗證。

工具

AWS 服務

- AWS 雲端開發套件 (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。
- AWS CloudFormation 可協助您設定 AWS 資源、快速一致地佈建資源，以及在整個 AWS 帳戶和區域的生命週期進行管理。在此模式中CloudFormation 範本可用來建立 CodeCommit 儲存庫和 CodePipeline CI/CD 管道。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。
- AWS CodePipeline 是一種 CI/CD 服務，可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。

其他工具

- cfn_nag 是一種開放原始碼工具，可在 CloudFormation 範本中尋找模式，以識別潛在的安全問題。
- git-remote-codecommit 是一種公用程式，可透過擴展 Git 從 CodeCommit 儲存庫推送和提取程式碼。
- Node.js 是一種事件驅動的 JavaScript 執行期環境，旨在建置可擴展的網路應用程式。

Code

此模式的程式碼可在 GitHub AWS CodePipeline 搭配 CI/CD 實務儲存庫中使用。

最佳實務

檢閱資源，例如 AWS Identity and Access Management (IAM) 政策，以確認它們符合您的組織最佳實務。

史詩

安裝工具

任務	Description	所需的技能

在 macOS 或 Linux 上安裝工具。	如果您使用的是 MacOS 或 Linux，您可以在偏好的終端機中執行下列命令或使用 Homebrew for Linux 來安裝工具。
	
	`brew install`
	`brew install git-remote-codecommit`
	`brew install ruby brew-gem`
	`brew-gem install cfn-nag`
		DevOps 工程師

設定 AWS CLI。	若要設定 AWS CLI，請使用作業系統的指示： • Windows：使用 AWS CLI 登入資料協助程式在 Windows 上設定 HTTPS 連線至 AWS CodeCommit 儲存庫的步驟 • Linux、macOS、Unix：使用 AWS CLI 憑證協助程式，在 Linux、macOS 或 Unix 上設定 HTTPS 連線至 AWS CodeCommit 儲存庫的步驟 macOS	DevOps 工程師

設定初始部署

任務	Description	所需的技能

下載或複製程式碼。	若要取得此模式使用的程式碼，請執行下列其中一項操作：
	• 從 GitHub 儲存庫的版本下載最新的原始程式碼，並將下載的檔案解壓縮到資料夾中。
	• 執行下列命令來複製專案。
	`git clone --depth 1 https://github.com/aws-samples/aws-codepipeline-cicd.git`
	從複製的儲存庫移除`.git`目錄。
	`cd ./aws-codepipeline-cicd`
	`rm -rf ./.git`
	稍後，您將使用新建立的 AWS CodeCommit 儲存庫做為遠端原始伺服器。
		DevOps 工程師

連線至 AWS 帳戶。	您可以使用臨時安全字符或登陸區域身分驗證來連線。若要確認您使用的是正確的帳戶和 AWS 區域，請執行下列命令。
	`AWS_REGION="eu-west-1"`
	`ACCOUNT_NUMBER=$(aws sts get-caller-identit`
		DevOps 工程師

任務	Description	所需的技能
	`y --query Account --output text)` `echo "${ACCOUNT_NUMBER}"`	

引導環境。	若要引導 AWS CDK 環境，請執行下列命令。
	`npm install`
	`npm run cdk bootstrap "aws://${ACCOUNT_NUMBER}/${AWS_REGION}"`
	成功引導環境後，應該會顯示下列輸出。
	`# Bootstrapping environment aws://{account}/{region}...`
	`# Environment aws://{account}/{region} bootstrapped`
	如需 AWS CDK 引導的詳細資訊，請參閱 AWS CDK 文件。
		DevOps 工程師

任務	Description	所需的技能

合成範本。	若要合成 AWS CDK 應用程式，請使用 `cdk synth`命令。
	`npm run cdk synth`
	您應該會看到下列輸出。
	`Successfully synthesized to <path-to-directory>/aws-codepipeline-cicd/cdk.out`
	`Supply a stack id (CodePipeline, Dev-MainStack) to display its template.`
		DevOps 工程師

任務	Description	所需的技能

部署 CodePipeline 堆疊。	現在您已啟動並合成 CloudFormation 範本，您可以進行部署。部署將建立 CodePipeline 管道和 CodeCommit 儲存庫，這會是管道的來源和觸發條件。
	`npm run cdk -- deploy CodePipeline --require-approval never`
	執行 命令後，您應該會看到 CodePipeline 堆疊和輸出資訊的成功部署。`CodePipeline.RepositoryName` 為您提供 AWS 帳戶中 CodeCommit 儲存庫的名稱。
	`CodePipeline: deploying...`
	`CodePipeline: creating CloudFormation changeset...`
	`# CodePipeline Outputs:`
	`CodePipeline.RepositoryName = SampleRepository`
	`Stack ARN: arn:aws:cloudformation:REGION:ACCOUNT-ID:stack/CodePipeline/STACK-ID`
		DevOps 工程師

任務	Description	所需的技能

設定遠端 CodeCommit 儲存庫和分支。	成功部署後，CodePipeline 會啟動管道的第一次執行，您可以在 AWS CodePipeline 主控台中找到。由於 AWS CDK 和 CodeCommit 不會啟動預設分支，因此此初始管道執行會失敗並傳回下列錯誤訊息。
	`The action failed because no branch named main was found in the selected AWS CodeCommit repository SampleRepository. Make sure you are using the correct branch name, and then try again. Error: null`
	若要修正此錯誤，請將遠端原始伺服器設定為 `SampleRepository` ，並建立所需的`main`分支。
	`RepoName=$(aws cloudformation describe-stacks --stack-name CodePipeline --query "Stacks[0].Outputs[?OutputKey=='RepositoryName'].OutputValue" --output text)`
	`echo "${RepoName}"`
	`#`
	`git init`
	`git branch -m master main`
		DevOps 工程師

任務	Description	所需的技能

	`git remote add origin codecommit://${RepoName}`
	`git add .`
	`git commit -m "Initial commit"`
	`git push -u origin main`	

測試部署的 CodePipeline 管道

任務	Description	所需的技能
遞交變更以啟用管道。	成功初始部署後，您應該擁有完整的 CI/CD 管道，並將 `main`的分支`SampleRepository` 做為來源分支。一旦您將變更遞交至`main`分支，管道就會啟動並執行下列動作序列： 1. 從 CodeCommit 儲存庫取得您的程式碼。 2. 建置您的程式碼。 3. 更新管道本身 (`UpdatePipeline`)。 4. 執行三個平行任務以進行固定、安全和單元測試檢查。 5. 如果成功，管道將從 部署`Main`堆疊`./lib/main-stack.ts` 到開發環境。 6. 執行已部署資源的部署後檢查。您可以在 CodePipel	DevOps 工程師

任務	Description	所需的技能
	ine 主控台中遵循所有 CodePipeline 步驟和結果。 7. 如果成功，管道將重複測試和生產環境的部署和驗證。	

使用 Makefile 在本機進行測試

任務	Description	所需的技能
使用 Makefile 執行開發程序。	您可以使用 `make`命令在本機執行整個管道，也可以執行個別步驟 （例如 `make linting`)。 若要使用 進行測試`make`，請執行下列動作： • 實作本機管道： `make` • 僅執行單元測試： `make unittest` • 部署至目前的 帳戶： `make deploy` • 清除環境： `make clean`	應用程式開發人員、DevOps 工程師

清除資源

任務	Description	所需的技能
刪除 AWS CDK 應用程式資源。	若要清除您的 AWS CDK 應用程式，請執行下列命令。 `cdk destroy --all`	DevOps 工程師

任務	Description	所需的技能
	請注意，在引導期間建立的 Amazon Simple Storage Service (Amazon S3) 儲存貯體不會自動刪除。他們需要允許刪除的保留政策，或者您需要在 AWS 帳戶中手動刪除。	

疑難排解

問題	解決方案
範本未如預期般運作。	如果發生問題且範本無法運作，請確定您有下列項目： • 工具的適當版本。 • 存取目標 AWS 帳戶 （網路連線）。 • 目標 AWS 帳戶的足夠許可。

相關資源

- 在 IAM Identity Center 中開始進行常見任務
- AWS CodePipeline 文件
- AWS CDK

使用 Terraform 設定企業規模的集中式記錄

Aarti Rajput、Yashwant Patel 和 Nishtha Yadav，Amazon Web Services

總結

集中式記錄對於組織的雲端基礎設施至關重要，因為它提供對其操作、安全性和合規性的可見性。隨著您的組織將其 AWS 環境擴展到多個帳戶，結構化日誌管理策略成為執行安全操作、滿足稽核要求和實現卓越營運的基礎。

此模式提供可擴展且安全的架構，用於集中來自多個 AWS 帳戶 和 服務的日誌，以跨複雜 AWS 部署啟用企業規模的記錄管理。該解決方案使用 Terraform 自動化，Terraform 是一種來自 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可確保一致且可重複的部署，並將手動組態降至最低。透過結合 Amazon CloudWatch Logs、Amazon Data Firehose 和 Amazon Simple Storage Service (Amazon S3)，您可以實作強大的日誌彙總和分析管道，提供：

- 中整個組織的集中式日誌管理 AWS Organizations
- 使用內建安全控制自動收集日誌
- 可擴展的日誌處理和耐用的儲存
- 簡化的合規報告和稽核追蹤
- 即時營運洞察和監控

解決方案會透過 CloudWatch Logs 從 Amazon Elastic Kubernetes Service (Amazon EKS) 容器、AWS Lambda 函數和 Amazon Relational Database Service (Amazon RDS) 資料庫執行個體收集日誌。它會使用 CloudWatch 訂閱篩選條件，自動將這些日誌轉送至專用記錄帳戶。Firehose 會管理 Amazon S3 的高輸送量日誌串流管道，以進行長期儲存。Amazon Simple Queue Service (Amazon SQS) 設定為在物件建立時接收 Amazon S3 事件通知。這可讓 與 分析服務整合，包括：

- Amazon OpenSearch Service 用於日誌搜尋、視覺化和即時分析
- 適用於 SQL 型查詢的 Amazon Athena
- 適用於大規模處理的 Amazon EMR
- 用於自訂轉換的 Lambda
- Amazon Quick Sight for 儀表板

所有資料都是使用 AWS Key Management Service (AWS KMS) 加密，而整個基礎設施則是使用 Terraform 跨環境進行一致組態來部署。

這種集中式記錄方法可讓組織改善其安全狀態、維持合規要求，以及最佳化 AWS 基礎設施的操作效率。

先決條件和限制

先決條件

- 使用 建置的組織登陸區域 AWS Control Tower
- 適用於 Terraform 的 Account Factory (AFT)，使用必要的帳戶部署和設定
- 用於佈建基礎設施的 Terraform
- 跨帳戶存取的 AWS Identity and Access Management (IAM) 角色和政策

如需設定 AWS Control Tower、AFT 和應用程式帳戶的指示，請參閱 Epics 一節。

必要帳戶

您在 中的組織 AWS Organizations 應該包含這些帳戶：

- 應用程式帳戶 – 一或多個來源帳戶，其中 AWS 服務 (Amazon EKS、Lambda 和 Amazon RDS) 執行和產生日誌
- Log Archive 帳戶 – 用於集中式日誌儲存和管理的專用帳戶

產品版本

- AWS Control Tower 3.1 版或更新版本
- Terraform 0.15.0 版或更新版本

Architecture

下圖說明 AWS 集中式記錄架構，提供可擴展的解決方案，用於從多個應用程式帳戶收集、處理和儲存日誌至專用 Log Archive 帳戶。此架構可有效率地處理來自 的日誌 AWS 服務，包括 Amazon RDS、Amazon EKS 和 Lambda，並透過簡化程序將日誌路由至 Log Archive 帳戶中的區域 S3 儲存貯體。

工作流程包含五個程序：

1. 日誌流程程序

- 日誌流程會在應用程式帳戶中開始，其中 AWS 服務 會產生各種類型的日誌，例如一般、錯誤、稽核、來自 Amazon RDS 的慢查詢日誌、來自 Amazon EKS 的控制平面日誌，以及來自 Lambda 的函數執行和錯誤日誌。
- CloudWatch 做為初始收集點。它會在每個應用程式帳戶中的日誌群組層級收集這些日誌。
- 在 CloudWatch 中，訂閱篩選條件會決定哪些日誌應轉送至中央帳戶。這些篩選條件可讓您精細控制日誌轉送，因此您可以指定確切的日誌模式或完整的日誌串流以進行集中化。

2. 跨帳戶日誌傳輸
   - 日誌會移至 Log Archive 帳戶。CloudWatch 訂閱篩選條件可促進跨帳戶轉移並保留區域內容。
   - 架構會建立多個平行串流，以有效率地處理不同的日誌來源，以確保最佳效能和可擴展性。
3. Log Archive 帳戶中的日誌處理
   - 在日誌封存帳戶中，Firehose 會處理傳入的日誌串流。
   - 每個區域都會維護專用的 Firehose 交付串流，可視需要轉換、轉換或擴充日誌。
   - 這些 Firehose 串流會將處理過的日誌交付至 Log Archive 帳戶中的 S3 儲存貯體，該帳戶與來源應用程式帳戶 （ 圖表中的區域 A) 位於相同的區域，以維護資料主權需求。
4. 通知和其他工作流程
   - 當日誌到達其目的地 S3 儲存貯體時，架構會使用 Amazon SQS 實作通知系統。
   - 區域 SQS 佇列可啟用非同步處理，並根據儲存的日誌觸發其他工作流程、分析或提醒系統。
5. AWS KMS 確保安全

   架構整合 AWS KMS 用於 security。 AWS KMS 提供 S3 儲存貯體的加密金鑰。這可確保所有儲存的日誌保持靜態加密，同時保持區域加密以滿足資料駐留要求。

工具

AWS 服務

- Amazon CloudWatch 是一種監控和可觀測性服務，以日誌、指標和事件的形式收集監控和操作資料。它提供在 AWS 和內部部署伺服器上執行 AWS 的資源、應用程式和服務統一檢視。
- CloudWatch Logs 訂閱篩選條件是符合傳入日誌事件中模式的表達式，並將相符的日誌事件交付給指定的 AWS 資源，以進行進一步的處理或分析。
- AWS Control Tower Account Factory for Terraform (AFT) 會設定 Terraform 管道，協助您在其中佈建和自訂帳戶 AWS Control Tower。AFT 提供以 Terraform 為基礎的帳戶佈建，同時允許您使用 管理帳戶 AWS Control Tower。

- Amazon Data Firehose 會將即時串流資料交付至目的地，例如 Amazon S3、Amazon Redshift 和 Amazon OpenSearch Service。它會自動擴展以符合資料的輸送量，而且不需要持續管理。
- Amazon Elastic Kubernetes Service (Amazon EKS) 是一種受管容器協同運作服務，可讓您使用 Kubernetes 輕鬆部署、管理和擴展容器化應用程式。它會自動管理 Kubernetes 控制平面節點的可用性和可擴展性。
- AWS Key Management Service (AWS KMS) 會建立和控制加密資料的加密金鑰。 會與其他 AWS KMS 整合 AWS 服務 ，以協助您保護使用這些服務存放的資料。
- AWS Lambda 是一種無伺服器運算服務，可讓您執行程式碼，而無需佈建或管理伺服器。它會執行程式碼以回應每個觸發條件，並僅針對您使用的運算時間來自動擴展您的應用程式。
- Amazon Relational Database Service (Amazon RDS) 是一種受管關聯式資料庫服務，可讓您輕鬆地在雲端中設定、操作和擴展關聯式資料庫。它提供經濟實惠且可擴展的容量，同時自動化耗時的管理任務。
- Amazon Simple Queue Service (Amazon SQS) 是一種訊息佇列服務，可讓您解耦和擴展微服務、分散式系統和無伺服器應用程式。它消除了管理和操作訊息導向中介軟體的複雜性。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可提供可擴展性、資料可用性、安全性和效能。它可以從 Web 上的任何位置存放和擷取任意數量的資料。

其他工具

- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。

Code

此模式的程式碼可在 GitHub 集中式日誌儲存庫中使用。

最佳實務

- 在AWS 帳戶 單一組織中使用多個 AWS Organizations 。此實務可啟用跨帳戶的集中式管理和標準化記錄。
- 使用版本控制、生命週期政策和跨區域複寫來設定 S3 儲存貯體。實作加密和存取記錄，以確保安全性和合規性。
- 使用 JSON 格式搭配標準時間戳記和欄位，實作常見的記錄標準。使用一致的字首結構和相互關聯 IDs以便於追蹤和分析。

- 啟用具有 AWS KMS 加密和最低權限存取的安全控制。維持 AWS CloudTrail 監控和定期金鑰輪換，以增強安全性。
- 設定 CloudWatch 指標和提醒以進行交付追蹤。透過自動通知監控成本和效能。
- 設定 Amazon S3 保留政策以符合合規要求，並啟用 Amazon S3 伺服器存取記錄來追蹤對 S3 儲存貯體提出的所有請求。維護 S3 儲存貯體政策和生命週期規則的文件。定期審查存取日誌、儲存貯體許可和儲存組態，以協助確保合規性和安全性最佳實務。

史詩

設定 AWS Control Tower、AFT 和應用程式帳戶

任務	Description	所需的技能
使用 AFT 設定 AWS Control Tower 環境。	1. 請依照 AWS Control Tower 文件中的指示 AWS Control Tower 進行部署。 2. 遵循 AWS Control Tower 文件中的指示部署 AFT。	AWS 管理員
為組織啟用資源共用。	1. Configure AWS Command Line Interface (AWS CLI) 具有管理帳戶登入資料，可提供管理許可。 AWS Control Tower 2. 在任何 中執行下列 AWS CLI 命令 AWS 區域： `aws ram enable-sharing-with-aws-organization` 這可讓您在 中 AWS Organizations 跨所有支援 AWS Resource Access Manager () 的區域進行組織內的資源共用AWS RAM。	AWS 管理員

任務	Description	所需的技能
驗證或佈建應用程式帳戶。	若要為您的使用案例佈建新的應用程式帳戶，請透過 AFT 建立它們。如需詳細資訊，請參閱 AWS Control Tower 文件中的使用 AFT 佈建新帳戶。	AWS 管理員

設定應用程式帳戶的組態檔案

任務	Description	所需的技能
將Application_account 資料夾內容複製到aft-account-customizations 儲存庫。	1. 在aft-account-customizations 儲存庫的根路徑Application_account 中建立名為 的資料夾。當您設定 AFT 時，會自動建立此儲存庫（請參閱上一個範例）。 2. 導覽至 centralised-logging-at-enterprise-scale-using-terraform 儲存庫的根目錄，複製aft/account 目錄的內容，然後將其貼到您在aft-account-customizations 儲存庫的步驟 1 中建立的Application_account 目錄中。 3. 從centralised-logging-at-enterprise-scale-using-terraform 儲存庫的根目錄中，將Application_account 目錄的內容複製到aft-account-custom	DevOps 工程師

任務	Description	所需的技能
	izations 儲存庫中的 Application_accoun t/terraform 目錄。 4. 在 aft-account-customizations/ Application_accoun t/terraform.tfvars 檔案中，確認所有參數都會做為對應 Terraform 組態檔案中的引數傳遞。	

任務	Description	所需的技能
檢閱和編輯用於設定應用程式帳戶的輸入參數。	在此步驟中，您會設定組態檔案以在應用程式帳戶中建立資源，包括 CloudWatch 日誌群組、CloudWatch 訂閱篩選條件、IAM 角色和政策，以及 Amazon RDS、Amazon EKS 和 Lambda 函數的組態詳細資訊。 在`aft-account-customizations` 儲存庫的 `Application_account` 資料夾中，根據您的組織需求設定 `terraform.tfvars` 檔案中的輸入參數： • `environment` ：將部署資源的環境名稱（例如 `prod`、`dev`、`staging`)。 • `account_name` ：要建立資源的 AWS 帳戶 名稱。 • `log_archive_account_id` ：將封存日誌的 AWS 帳戶 ID。 • `admin_role_name` ：將用於管理資源的管理角色名稱。 • `tags`：代表要套用至所有資源之常見標籤的鍵/值對映射。 • `rds_config` ：包含 Amazon RDS 執行個體組態詳細資訊的物件。	DevOps 工程師

任務	Description	所需的技能
	• `allowed_cidr_blocks` ：允許存取資源的 CIDR 區塊清單。 • `destination_name:` 用來建立 CloudWatch 目的地 Amazon Resource Name (ARN) 的變數，其中將串流日誌。 • `rds_parameters:` 包含 Amazon RDS 參數群組設定的物件。 • `vpc_config` ：包含 VPC 組態詳細資訊的物件。 • `eks_config` ：包含 Amazon EKS 叢集組態詳細資訊的物件。 • `lambda_config` ：包含 Lambda 函數組態詳細資訊的物件。 • `restrictive_cidr_range` ：安全群組規則的限制 CIDR 範圍清單。 • `target_account_id` ：將部署資源的目標 Log Archive 帳戶 AWS 帳戶 ID。	

設定 Log Archive 帳戶的組態檔案

任務	Description	所需的技能
將`Log_archive_account` 資料夾內容複製到`aft-`	1. 在`aft-account-customizations` 儲存庫的根路	DevOps 工程師

任務	Description	所需的技能
account-customizations 儲存庫。	徑Log_archive_account 中建立名為 的資料夾，此儲存庫會在您設定 AFT 時自動建立。 2. 導覽至centralised-logging-at-enterprise-scale-using-terraform 儲存庫的根目錄，複製aft/account 目錄的內容，然後將它們貼到您在aft-account-customizations 儲存庫的上一個步驟中建立的Log_archive_account 目錄中。 3. 從centralised-logging-at-enterprise-scale-using-terraform 儲存庫的根目錄，將Log_archive_account 目錄的內容複製到aft-account-customizations 儲存庫中的Log_archive_account/terraform 目錄。 4. 在 aft-account-customizations/Log_archive_account/terraform.tfvars 檔案中，確認所有參數都會做為對應 Terraform 組態檔案中的引數傳遞。	

任務	Description	所需的技能
檢閱和編輯用於設定 Log Archive 帳戶的輸入參數。	在此步驟中，您會設定組態檔案以在 Log Archive 帳戶中建立資源，包括 Firehose 交付串流、S3 儲存貯體、SQS 佇列，以及 IAM 角色和政策。 在`aft-account-customizations` 儲存庫的 `Log_archive_account` 資料夾中，根據您的組織需求設定 `terraform.tfvars` 檔案中的輸入參數： • `environment` ：將部署資源的環境名稱（例如 `prod`、`dev`、`staging`)。 • `destination_name` ：用來建立 CloudWatch 目的地 ARN 的變數，其中將串流日誌。 • `source_account_ids` ：允許在日誌目的地放置訂閱篩選條件的 AWS 帳戶 IDs 清單。您可以輸入任意數量的帳戶 IDs，以啟用集中式記錄。	DevOps 工程師

執行 Terraform 命令來佈建資源

任務	Description	所需的技能
選項 1 - 從 AFT 部署 Terraform 組態檔案。	在 AFT 中，當您將具有組態變更的程式碼推送至 GitHub `aft-account-custom`	DevOps 工程師

任務	Description	所需的技能

	izations 儲存庫後，即會觸發 AFT 管道。AFT 會自動偵測變更並啟動帳戶自訂程序。
	
	對 Terraform (terraform .tfvars) 檔案進行變更後，請遞交變更並推送至儲存aft-account-customizations 庫：
	
	`$ git add *`
	`$ git commit -m "update message"`
	`$ git push origin main`
	
	Note
	如果您使用的是不同的分支 （ 例如 dev)，請將 取代main為您的分支名稱。	

任務	Description	所需的技能

選項 2 - 手動部署 Terraform 組態檔案。	如果您未使用 AFT 或想要手動部署解決方案，您可以從 `Application_account` 和 `Log_archive_account` 資料夾使用以下 Terraform 命令：
	1. 複製 GitHub 儲存庫並設定 `terraform.tfvars` 檔案中的輸入參數。
	2. 執行以下命令：
	`$ terraform init`
	3. 預覽變更：
	`$ terraform plan`
	此命令會評估 Terraform 組態，以判斷資源的所需狀態，並將其與基礎設施的目前狀態進行比較。
	4. 套用變更：
	`$ terraform apply`
	5. 檢閱計劃變更，並在提示中輸入 yes 以繼續應用程式。
		DevOps 工程師

驗證資源

任務	Description	所需的技能
驗證訂閱篩選條件。	若要驗證訂閱篩選條件是否正確地將日誌從應用程式帳戶日誌群組轉送至日誌封存帳戶： 1. 在應用程式帳戶中，開啟 CloudWatch 主控台。 2. 在左側導覽窗格中，選擇 Log groups (日誌群組)。 3. 選取每個日誌群組 (`/aws/rds`、`/aws/eks`、`/aws/lambda`)，然後選擇訂閱篩選條件索引標籤。 根據您在 Terraform 組態檔案中指定的名稱，您應該會看到指向目的地 ARN 的作用中訂閱篩選條件。 4. 選擇任何訂閱篩選條件來驗證其組態和狀態。	DevOps 工程師
驗證 Firehose 串流。	若要驗證日誌封存帳戶程序應用程式日誌中的 Firehose 串流是否成功： 1. 在 Log Archive 帳戶中，開啟 Firehose 主控台。 2. 在左側導覽窗格中，選擇 Firehose 串流。 3. 選擇任何 Firehose 串流並驗證下列項目： • 目的地會顯示正確的 S3 儲存貯體。	DevOps 工程師

任務	Description	所需的技能
	• 監控索引標籤會顯示成功的交付指標。 • 最近的交付時間戳記是最新的。	
驗證集中式 S3 儲存貯體。	若要驗證集中式 S3 儲存貯體是否正確接收和組織日誌： 1. 在 Log Archive 帳戶中，開啟 Amazon S3 主控台。 2. 選取每個中央記錄儲存貯體。 3. 瀏覽資料夾結構：AWSLogs/AccountID/Region/Service。 您應該會看到依時間戳記 (YYYY/MM/DD/HH) 整理的日誌檔案。 4. 選擇任何最近的日誌檔案，並驗證其格式和資料完整性。	DevOps 工程師

任務	Description	所需的技能
驗證 SQS 佇列。	若要驗證 SQS 佇列是否收到新日誌檔案的通知： 1. 在 Log Archive 帳戶中，開啟 Amazon SQS 主控台。 2. 在左側導覽窗格中，選擇 Queues (佇列)。 3. 選取每個設定的佇列，然後選擇傳送和接收訊息。 您應該會看到包含新日誌檔案 S3 事件通知的訊息。 4. 選擇任何訊息，以確認其中包含正確的 S3 物件資訊。	DevOps 工程師

清除資源

任務	Description	所需的技能
選項 1 - 從 AFT 停用 Terraform 組態檔案。	當您移除 Terraform 組態檔案並推送變更時，AFT 會自動啟動資源移除程序。 1. 導覽至 `aft-account-customizations` 儲存庫。 2. 前往 `terraform` 目錄。 3. 刪除下列檔案： • `modules` 目錄 • `iam.tf` • `versions.tf` • `variables.tf` • `outputs.tf`	DevOps 工程師

任務	Description	所需的技能

	• terraform.tfvars
	4. 清除 main.tf 檔案的內容。
	5. 將您的變更推送至儲存庫：
	`# Stage all changes`
	`$ git add *`
	`# Commit cleanup changes`
	`$ git commit -m "Remove AFT customizations"`
	`# Push to repository`
	`$ git push origin main`
	Note
	如果您使用的是不同的分支（例如 dev），請將 取代main為您的分支名稱。	

任務	Description	所需的技能

選項 2 – 手動清除 Terraform 資源。	如果您未使用 AFT 或想要手動清除資源，請使用 `Application_account` 和 `Log_archive_account` 資料夾的下列 Terraform 命令：
	1. 初始化 Terraform 組態：
	`$ terraform init`
	此命令會初始化 Terraform，並確保存取目前狀態。
	2. 預覽清除變更：
	`$ terraform destroy`
	此命令會評估要銷毀哪些資源，並將所需的狀態與基礎設施的目前狀態進行比較。
	3. 執行清除。出現提示時，請輸入 yes 以確認並執行銷毀計畫。
		DevOps 工程師

疑難排解

問題	解決方案
CloudWatch Logs 目的地未建立或處於非作用中狀態。	驗證下列項目： 1. 在 Log Archive 帳戶中，確認目的地政策包含： • 正確的來源帳戶主體。

問題	解決方案
	• 正確的動作 (`logs:PutSubscriptionFilter`)。 • 有效的目的地 ARN。 2. 確認 Firehose 串流存在且處於作用中狀態。 3. 確認連接至目的地的 IAM 角色具有 Firehose 的許可。
訂閱篩選條件失敗或停滯在待定狀態。	請檢查以下內容： 1. 在應用程式帳戶中，確認 IAM 角色具有： • 呼叫 的許可`PutSubscriptionFilter` 。 • 與 CloudWatch Logs 的信任關係。 2. 確認目的地 ARN 正確。 3. 檢查 CloudWatch Logs 是否有特定錯誤訊息。
Firehose 交付串流不會顯示傳入記錄。	請確認下列內容： 1. 確認 Firehose IAM 角色具有： • 寫入 Amazon S3 的許可。 • 如果啟用加密，則存取 AWS KMS 金鑰。 2. 檢閱 CloudWatch 指標： • `IncomingRecords` • `DeliveryToS3.Records` 3. 驗證緩衝區設定和交付組態。

相關資源

- Terraform 基礎設施設定 (Terraform 文件)
- 部署適用於 Terraform (AFT) AWS Control Tower 的帳戶工廠 (AWS Control Tower 文件)
- IAM 教學課程： AWS 帳戶 使用 IAM 角色將存取權委派給 (IAMdocumentation)

使用 cert-manager 和 Let's Encrypt 為 Amazon EKS 上的應用程式設定end-to-end加密

Mahendra Revanasiddappa 和 Vasanth Jeyaraj，Amazon Web Services

總結

實作end-to-end加密可能很複雜，您需要管理微服務架構中每個資產的憑證。雖然您可以使用 Network Load Balancer 或 Amazon API Gateway 在 Amazon Web Services (AWS) 網路的邊緣終止 Transport Layer Security (TLS) 連線，但某些組織需要end-to-end加密。

此模式使用 NGINX 傳入控制器進行傳入。這是因為當您建立 Kubernetes 輸入時，輸入資源會使用 Network Load Balancer。Network Load Balancer 不允許上傳用戶端憑證。因此，您無法透過 Kubernetes 輸入實現交互 TLS。

此模式適用於在其應用程式中的所有微服務之間需要交互身分驗證的組織。相互 TLS 可減少維護使用者名稱或密碼的負擔，也可以使用統包安全架構。如果您的組織有大量連線裝置，或必須符合嚴格的安全準則，則此模式的方法是相容的。

此模式透過為在 Amazon Elastic Kubernetes Service (Amazon EKS) 上執行的應用程式實作end-to-end加密，協助提高組織的安全狀態。此模式在 Amazon EKS 儲存庫上的 GitHub 端對端加密中提供範例應用程式和程式碼，以顯示微服務如何在 Amazon EKS 上使用end-to-end加密來執行。 End-to-end 模式的方法使用 cert-manager，這是 Kubernetes 的附加元件，並以 Let's Encrypt 做為憑證授權單位 (CA)。Let's Encrypt 是一種經濟實惠的解決方案，可用來管理憑證，並提供 90 天內有效的免費憑證。在 Amazon EKS 上部署新的微服務時，Cert-manager 會自動化隨需佈建和輪換憑證。

目標對象

對於具有 Kubernetes、TLS、Amazon Route 53 和網域名稱系統 (DNS) 經驗的使用者，建議使用此模式。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 現有 Amazon EKS 叢集。
- 在 macOS、Linux 或 Windows 上安裝和設定 AWS Command Line Interface (AWS CLI) 1.7 版或更新版本。

- kubectl 命令列公用程式，已安裝並設定為存取 Amazon EKS 叢集。如需詳細資訊，請參閱 Amazon EKS 文件中的安裝 kubectl。
- 用來測試應用程式的現有 DNS 名稱。如需詳細資訊，請參閱《Amazon Route 53 文件》中的使用 Amazon Route 53 註冊網域名稱。 Amazon Route 53
- 安裝在本機電腦上的最新 Helm 版本。如需詳細資訊，請參閱 Amazon EKS 文件和 GitHub Helm 儲存庫中的搭配使用 Helm 與 Amazon EKS。 GitHub
- Amazon EKS 儲存庫上的 GitHub 端對端加密，複製到您的本機電腦。 End-to-end
- 從 Amazon EKS 儲存庫上複製的 GitHub 端對端加密取代 policy.json和 trustpolicy.json 檔案中的下列值： End-to-end
  - <account number> – 將 取代為您要部署解決方案之帳戶的 AWS 帳戶 ID。
  - <zone id> – 將 取代為網域名稱的 Route 53 區域 ID。
  - <node_group_role> – 將 取代為與 Amazon EKS 節點相關聯的 AWS Identity and Access Management (IAM) 角色名稱。
  - <namespace> – 將 取代為您部署 NGINX 傳入控制器和範例應用程式的 Kubernetes 命名空間。
  - <application-domain-name> – 將 取代為來自 Route 53 的 DNS 網域名稱。

限制

- 此模式不會描述如何輪換憑證，只會示範如何在 Amazon EKS 上使用微服務憑證。

Architecture

下圖顯示此模式的工作流程和架構元件。

該圖顯示以下工作流程：

1. 用戶端傳送存取應用程式至 DNS 名稱的請求。
2. Route 53 記錄是 Network Load Balancer 的 CNAME。
3. Network Load Balancer 會將請求轉送至使用 TLS 接聽程式設定的 NGINX 傳入控制器。NGINX 傳入控制器與 Network Load Balancer 之間的通訊遵循 HTTPS 通訊協定。
4. NGINX 傳入控制器會根據用戶端對應用程式服務的請求，執行以路徑為基礎的路由。
5. 應用程式服務會將請求轉送至應用程式 Pod。應用程式旨在透過呼叫秘密來使用相同的憑證。

6. Pod 會使用 cert-manager 憑證執行範例應用程式。NGINX 傳入控制器與 Pod 之間的通訊使用 HTTPS。

注意：Cert-manager 會在自己的命名空間中執行。它使用 Kubernetes 叢集角色將憑證佈建為特定命名空間中的秘密。您可以將這些命名空間連接至應用程式 Pod 和 NGINX 傳入控制器。

工具

AWS 服務

- Amazon Elastic Kubernetes Service (Amazon EKS) 是一種受管服務，可用來在 AWS 上執行 Kubernetes，而不需要安裝、操作和維護您自己的 Kubernetes 控制平面或節點。
- Elastic Load Balancing 會自動將您的傳入流量分配到多個目標、容器和 IP 地址。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon Route 53 是一種可用性高、可擴展性強的 DNS Web 服務。

其他工具

- cert-manager 是 Kubernetes 的附加元件，可請求憑證、將憑證分發至 Kubernetes 容器，以及自動化憑證續約。
- NGINX 輸入控制器是 Kubernetes 和容器化環境中雲端原生應用程式的流量管理解決方案。

史詩

使用 Route 53 建立和設定公有託管區域

任務	Description	所需的技能
在 Route 53 中建立公有託管區域。	登入 AWS 管理主控台，開啟 Amazon Route 53 主控台，選擇託管區域，然後選擇建立託管區域。建立公有託管區域並記錄區域 ID。如需詳細資訊，	AWS DevOps

任務	Description	所需的技能
	請參閱 Amazon Route 53 文件中的建立公有託管區域。 **Note** ACME DNS01 使用 DNS 提供者發佈挑戰，讓 cert-manager 發行憑證。此挑戰要求您證明控制網域名稱的 DNS，方法是將特定值放在該網域名稱下的 TXT 記錄中。在 Let's Encrypt 為您的 ACME 用戶端提供字符之後，您的用戶端會建立衍生自該字符和您帳戶金鑰的 TXT 記錄，並將該記錄放在 `_acme-challenge.<YOURDOMAIN>` 。然後，讓我們加密查詢該記錄的 DNS。如果找到相符項目，您可以繼續發出憑證。	

設定 IAM 角色以允許 cert-manager 存取公有託管區域

任務	Description	所需的技能
建立 cert-manager 的 IAM 政策。	需要 IAM 政策才能為 cert-manager 提供許可，以驗證您擁有 Route 53 網域。`policy.json` 範例 IAM	AWS DevOps

任務	Description	所需的技能
	政策會在 Amazon EKS 儲存庫上複製的 GitHub 端對端加密的 `1-IAMRole` 目錄中提供。End-to-end 在 AWS CLI 中輸入下列命令來建立 IAM 政策。 `aws iam create-policy \ --policy-name PolicyForCertManager \ --policy-document file://policy.json`	
建立 cert-manager 的 IAM 角色。	建立 IAM 政策後，您必須建立 IAM 角色。`trustpolicy.json` 範例 IAM 角色在 `1-IAMRole` 目錄中提供。 在 AWS CLI 中輸入下列命令來建立 IAM 角色。 `aws iam create-role \ --role-name RoleForCertManager \ --assume-role-policy-document file://trustpolicy.json`	AWS DevOps

任務	Description	所需的技能
將政策連接到角色。	在 AWS CLI 中輸入下列命令，將 IAM 政策連接至 IAM 角色。`AWS_ACCOUNT_ID` 將取代為您 AWS 帳戶的 ID。 `aws iam attach-role-policy \ --policy-arn arn:aws:iam::AWS_ACCOUNT_ID:policy/PolicyForCertManager \ --role-name RoleForCertManager`	AWS DevOps

在 Amazon EKS 中設定 NGINX 傳入控制器

任務	Description	所需的技能
部署 NGINX 傳入控制器。	`nginx-ingress` 使用 Helm 安裝最新版本的 。您可以在部署組態之前，根據您的需求修改`nginx-ingress` 組態。此模式使用註釋且面向內部的 Network Load Balancer，該模式可在 `5-Nginx-Ingress-Controller` 目錄中使用。 從 `5-Nginx-Ingress-Controller` 目錄執行下列 Helm 命令來安裝 NGINX 傳入控制器。 `helm install test-nginx nginx-stable/nginx-ingress -f`	AWS DevOps

任務	Description	所需的技能
	`5-Nginx-Ingress-Controller/values_internal_nlb.yaml`	
確認已安裝 NGINX 傳入控制器。	輸入 `helm list` 命令。輸出應會顯示已安裝 NGINX 傳入控制器。	AWS DevOps

任務	Description	所需的技能
建立 Route 53 A 記錄。	A 記錄指向 NGINX 傳入控制器建立的 Network Load Balancer。 1. 取得 Network Load Balancer 的 DNS 名稱。如需說明，請參閱取得 ELB 負載平衡器的 DNS 名稱。 2. 在 Amazon Route 53 主控台上，選擇託管區域。 3. 選取您要在其中建立記錄的公有託管區域，然後選擇建立記錄。 4. 輸入記錄的名稱。 5. 在記錄類型中，選擇 A - 將流量路由到 IPv4 和一些 AWS 資源。 6. 啟用別名。 7. 在路由流量到 中，執行下列動作： a. 選擇別名至 Network Load Balancer。 b. 選擇部署 Network Load Balancer 的 AWS 區域。 c. 輸入 Network Load Balancer 的 DNS 名稱。 8. 選擇建立記錄。	AWS DevOps

在 Amazon EKS 上設定 NGINX VirtualServer

任務	Description	所需的技能
部署 NGINX VirtualServer。	NGINX VirtualServer 資源是一種負載平衡組態，是輸入資源的替代方案。建立 NGINX VirtualServer 資源的組態可在 `6-Nginx-Virtual-Server` 目錄中的 `nginx_virtualserver.yaml` 檔案中取得。在 中輸入下列命令`kubectl`以建立 NGINX VirtualServer 資源。 `kubectl apply -f nginx_virtualserver.yaml` **Important** 請務必更新 `nginx_virtualserver.yaml` 檔案中的應用程式網域名稱、憑證秘密和應用程式服務名稱。	AWS DevOps
確認已建立 NGINX VirtualServer。	在 中輸入下列命令`kubectl`，以確認已成功建立 NGINX VirtualServer 資源。 `kubectl get virtualserver`	AWS DevOps

任務	Description	所需的技能
	Note 確認資料Host欄符合您應用程式的網域名稱。	
部署已啟用 TLS 的 NGINX Web 伺服器。	此模式使用已啟用 TLS 的 NGINX Web 伺服器做為應用程式，以測試end-to-end加密。部署測試應用程式所需的組態檔案可在 `demo-webserver` 目錄中取得。 在 中輸入下列命令kubectl以部署測試應用程式。 `kubectl apply -f nginx-tls-ap.yaml`	AWS DevOps

任務	Description	所需的技能
確認測試應用程式資源已建立。	在 中輸入下列命令kubectl，以確認已建立測試應用程式所需的資源： • `kubectl get deployments` **Note** 驗證資料Ready欄和資料Available欄。 • `kubectl get pods | grep -i example-deploy` **Note** Pod 應該處於 running 狀態。 • `kubectl get configmap` • `kubectl get svc`	AWS DevOps
驗證應用程式。	1. 將 取代<application-domain-name> 為您先前建立的 Route53 DNS 名稱，以輸入下列命令。 `curl --verbose https://<application-domain-name>` 2. 確認您可以存取應用程式。	AWS DevOps

相關資源

AWS 資源

- 使用 Amazon Route 53 主控台建立記錄 (Amazon Route 53 文件）
- 在 Amazon EKS 上使用 Network Load Balancer 搭配 NGINX 輸入控制器 (AWS 部落格文章）

其他資源

- Route 53 (cert-manager 文件）
- 設定 DNS01 挑戰提供者 （憑證管理員文件）
- 讓我們加密 DNS 挑戰 （讓我們的加密文件）

使用 Flux 簡化 Amazon EKS 多租戶應用程式部署

Nadeem Rahaman、Aditya Ambati、Aniket Dekate 和 Shrikant Patil，Amazon Web Services

總結

許多提供產品和服務的公司都是資料受管產業，其內部業務職能之間需要維持資料障礙。此模式說明如何使用 Amazon Elastic Kubernetes Service (Amazon EKS) 中的多租用戶功能來建置資料平台，以在共用單一 Amazon EKS 叢集的租用戶或使用者之間實現邏輯和實體隔離。模式透過下列方法提供隔離：

- Kubernetes 命名空間隔離
- 角色型存取控制 (RBAC)
- 網路政策
- 資源配額
- AWS Identity and Access Management 服務帳戶 (IRSA) 的 (IAM) 角色

此外，在您部署應用程式時，此解決方案會使用 Flux 來保持租戶組態不變。您可以在組態中指定包含 Flux `kustomization.yaml` 檔案的租用戶儲存庫，以部署租用戶應用程式。

此模式實作下列項目：

- 透過手動部署 Terraform 指令碼建立的 AWS CodeCommit 儲存庫、 AWS CodeBuild 專案和 AWS CodePipeline 管道。
- 託管租用戶所需的網路和運算元件。這些是使用 Terraform 透過 CodePipeline 和 CodeBuild 建立。
- 透過 Helm Chart 設定的租戶命名空間、網路政策和資源配額。
- 屬於不同租用戶的應用程式，使用 Flux 部署。

我們建議您根據您的獨特需求和安全考量，仔細規劃和建置自己的多租戶架構。此模式為您的實作提供起點。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- AWS Command Line Interface (AWS CLI) 2.11.4 版或更新版本，已安裝並設定

- 安裝在本機電腦上的 Terraform 0.12 版或更新版本
- Terraform AWS 提供者 3.0.0 版或更新版本
- Kubernetes Provider 2.10 版或更新版本
- Helm Provider 2.8.0 版或更新版本
- Kubectl 提供者 1.14 版或更新版本

限制

- 相依於 Terraform 手動部署：工作流程的初始設定，包括建立 CodeCommit 儲存庫、CodeBuild 專案和 CodePipeline 管道，依賴手動 Terraform 部署。這在自動化和可擴展性方面引入了潛在的限制，因為它需要手動介入基礎設施變更。
- CodeCommit 儲存庫相依性：工作流程依賴 CodeCommit 儲存庫做為原始碼管理解決方案，並與 緊密結合 AWS 服務。

Architecture

目標架構

此模式部署三個模組來建置資料平台的管道、網路和運算基礎設施，如下圖所示。

管道架構：

網路架構：

運算架構：

工具

AWS 服務

- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。

- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 上執行 Kubernetes， AWS 而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- AWS Transit Gateway 是連接虛擬私有雲端 (VPC) 和內部部署網路的中央中樞。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。

其他工具

- Cilium 網路政策支援 Kubernetes L3 和 L4 網路政策。它們可以透過 L7 政策擴充，為 HTTP、Kafka 和 gRPC 以及其他類似的通訊協定提供 API 層級安全性。
- Flux 是一種 Git 型持續交付 (CD) 工具，可在 Kubernetes 上自動化應用程式部署。
- Helm 是 Kubernetes 的開放原始碼套件管理員，可協助您在 Kubernetes 叢集上安裝和管理應用程式。
- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。

程式碼儲存庫

此模式的程式碼可在 GitHub EKS 多租用戶 Terraform 解決方案儲存庫中使用。

最佳實務

如需使用此實作的指導方針和最佳實務，請參閱下列內容：

- Amazon EKS 多租戶最佳實務
- Flux 文件

史詩

建立 Terraform 建置、測試和部署階段的管道

任務	Description	所需的技能
複製專案儲存庫。	在終端機視窗中執行下列命令，複製 GitHub EKS 多租用戶 Terraform 解決方案儲存庫： `git clone https://github.com/aws-samples/aws-eks-multitenancy-deployment.git`	AWS DevOps

引導 Terraform S3 儲存貯體和 Amazon DynamoDB。	1. 在 `bootstrap` 資料夾中，開啟 `bootstrap.sh` 檔案並更新 S3 儲存貯體名稱、DynamoDB 資料表名稱和 的變數值 AWS 區域：
	`S3_BUCKET_NAME="<S3_BUCKET_NAME>" DYNAMODB_TABLE_NAME="<DYNAMODB_NAME>" REGION="<AWS_REGION>"`
	2. 執行 `bootstrap.sh` 指令碼。指令碼需要 AWS CLI 您安裝做為先決條件一部分的 。
	`cd bootstrap ./bootstrap.sh`
		AWS DevOps

任務	Description	所需的技能

更新 `run.sh`和 `locals.tf` 檔案。	1. 引導程序成功完成後，請從`bootstrap.sh` 指令碼的 `variables` 區段複製 S3 儲存貯體和 DynamoDB 資料表名稱：
	`# Variables S3_BUCKET_NAME="<S3_BUCKET_NAME>" DYNAMODB_TABLE_NAME="<DYNAMODB_NAME"`
	2. 將這些值貼到指令碼，該`run.sh`指令碼位於專案的根目錄中：
	`BACKEND_BUCKET_ID="<SAME_NAME_AS_S3_BUCKET_NAME>" DYNAMODB_ID="<SAME_NAME_AS_DYNAMODB_NAME>"`
	3. 將專案程式碼上傳至 CodeCommit 儲存庫。您可以透過 Terraform 自動建立此儲存庫，方法是`true`在 `demo/pipeline/locals.tf` 檔案中將下列變數設定為 ：
	`create_new_repo = true`
	4. 根據您的需求更新 `locals.tf` 檔案，以建立管道資源。
		AWS DevOps

任務	Description	所需的技能

部署管道模組。	若要建立管道資源，請手動執行下列 Terraform 命令。自動執行這些命令沒有協同運作。
	
	`./run.sh -m pipeline -e demo -r <AWS_REGION> -t init`
	`./run.sh -m pipeline -e demo -r <AWS_REGION> -t plan`
	`./run.sh -m pipeline -e demo -r <AWS_REGION> -t apply`
		AWS DevOps

建立網路基礎設施

任務	Description	所需的技能
啟動管道。	1. 在 `templates` 資料夾中，請確定`buildspec` 檔案的下列變數設定為 `network`： `TF_MODULE_TO_BUILD: "network"` 2. 在 CodePipeline 主控台的管道詳細資訊頁面上，選擇發行變更來啟動管道。 第一次執行後，每當您將變更遞交至 CodeCommit 儲存庫主分支時，管道會自動啟動。 管道包含下列階段：	AWS DevOps

任務	Description	所需的技能
	• `validate` 會初始化 Terraform、使用 checkov 和 tfsec 工具執行 Terraform 安全性掃描，並將掃描報告上傳至 S3 儲存貯體。 • `plan` 顯示 Terraform 計劃，並將計劃上傳至 S3 儲存貯體。 • `apply` 會套用 S3 儲存貯體的 Terraform 計劃輸出，並建立 AWS 資源。 • `destroy` 會移除在`apply`階段期間建立 AWS 的資源。若要啟用此選用階段，`true`請在 `demo/pipeline/locals.tf` 檔案中將下列變數設定為 ： `enable_destroy_stage = true`	

任務	Description	所需的技能
驗證透過網路模組建立的資源。	確認已在管道成功部署後建立下列 AWS 資源： • 具有三個公有和三個私有子網路、網際網路閘道和 NAT 閘道的輸出 VPC。 • 具有三個私有子網路的 Amazon EKS VPC。 • 租用戶 1 和租用戶 2 個 VPCs每個 VPC 具有三個私有子網路。 • 具有所有 VPC 連接和路由到每個私有子網路的傳輸閘道。 • 目的地 CIDR 區塊為的 Amazon EKS 輸出 VPC 靜態傳輸閘道路由`0.0.0.0/0` 。這是啟用所有 VPCsAmazon EKS 輸出 VPC 進行傳出網際網路存取的必要條件。	AWS DevOps

建立運算基礎設施

任務	Description	所需的技能
更新`locals.tf` 以啟用 CodeBuild 專案對 VPC 的存取。	若要部署 Amazon EKS 私有叢集的附加元件，CodeBuild 專案必須連接至 Amazon EKS VPC。 1. 在 `demo/pipeline` 資料夾中，開啟 `locals.tf`	AWS DevOps

任務	Description	所需的技能

	檔案，並將 vpc_enabled 變數設定為 true。
	2. 執行run.sh指令碼，將變更套用至管道模組：
	`demo/pipeline/locals.tf`
	`./run.sh -m pipeline -env demo -region <AWS_REGION> -tfcmd init`
	`./run.sh -m pipeline -env demo -region <AWS_REGION> -tfcmd plan`
	`./run.sh -m pipeline -env demo -region <AWS_REGION> -tfcmd apply`	

更新buildspec 檔案以建置運算模組。	在 templates 資料夾的所有 buildspec YAML 檔案中，將 TF_MODULE_TO_BUILD 變數的值從 設定為 network compute： `TF_MODULE_TO_BUILD: "compute"`	AWS DevOps

任務	Description	所需的技能

更新租戶管理 Helm Chart `values`的檔案。	1. 在下列位置開啟 `values.yaml` 檔案：
	`cd cfg-terraform/demo/compute/cfg-tenant-mgmt`
	檔案如下所示：
	`---`
	`global:`
	`  clusterRoles:`
	`  operator: platform-tenant`
	`  flux: flux-tenant-applier`
	`  flux:`
	`  tenantCloneBaseUrl: ${TEANT_BASE_URL}`
	`  repoSecret: ${TENANT_REPO_SECRET}`
	`tenants:`
	`  tenant-1:`
	`  quotas:`
	`  limits:`
	`  cpu: 1`
	`  memory: 1Gi`
	`  flux:`
	`  path: overlays/tenant-1`
	`  tenant-2:`
	`  quotas:`
	`  limits:`
	`  cpu: 1`
	`  memory: 2Gi`
	`  flux:`
		AWS DevOps

任務	Description	所需的技能

	<pre>path: overlays/
	tenant-2</pre>
	2. 在 global和 tenants區段中，根據您的需求更新組態：
	• tenantCloneBaseUrl – 為所有租用戶託管程式碼的儲存庫路徑（我們為所有租用戶使用相同的 Git 儲存庫）
	• repoSecret – 保存 SSH 金鑰和已知主機以向全域租用戶 Git 儲存庫進行身分驗證的 Kubernetes 秘密
	• quotas – 您要套用至每個租用戶的 Kubernetes 資源配額
	• flux path – 全域租用戶儲存庫中租用戶應用程式 YAML 檔案的路徑	

任務	Description	所需的技能
驗證運算資源。	在您更新先前步驟中的檔案後，CodePipeline 會自動啟動。確認它為運算基礎設施建立了下列 AWS 資源： • 具有私有端點的 Amazon EKS 叢集 • Amazon EKS 工作者節點 • Amazon EKS 附加元件：外部秘密`aws-loadbalancer-controller`、和 `metrics-server` • GitOps 模組、Flux Helm Chart、Cilium Helm Chart 和租用戶管理 Helm Chart	AWS DevOps

檢查租戶管理和其他資源

任務	Description	所需的技能
驗證 Kubernetes 中的租戶管理資源。	執行下列命令，以檢查在 Helm 的協助下已成功建立租用戶管理資源。 1. 租用戶命名空間已建立，如中所指定`values.yaml`： `kubectl get ns -A` 2. 配額會指派給每個租用戶命名空間，如 中所指定`values.yaml`：	AWS DevOps

任務	Description	所需的技能

	`kubectl get quota --namespace=<tenant_namespace>`
	3. 每個租用戶命名空間的配額詳細資訊都是正確的：
	`kubectl describe quota cpu-memory-resource-quota-limit -n <tenant_namespace>`
	4. Cilium 網路政策已套用至每個租用戶命名空間：
	`kubectl get CiliumNetworkPolicy -A`	

任務	Description	所需的技能

驗證租戶應用程式部署。	執行下列命令來驗證租用戶應用程式是否已部署。
	1. Flux 能夠連線到 GitOps 模組中指定的 CodeCommit 儲存庫：
	`kubectl get gitrepositories -A`
	2. Flux kustomization 控制器已在 CodeCommit 儲存庫中部署 YAML 檔案：
	`kubectl get kustomizations -A`
	3. 所有應用程式資源都會部署在其租戶命名空間中：
	`kubectl get all -n <tenant_namespace>`
	4. 已為每個租用戶建立輸入：
	`kubectl get ingress -n <tenant_namespace>`	

疑難排解

問題	解決方案
您遇到類似以下的錯誤訊息： `Failed to checkout and determine revision: unable to clone unknown`	請依照下列步驟對問題進行疑難排解：

問題	解決方案

`error: You have successfully authenticated over SSH. You can use Git to interact with AWS CodeCommit.`	1. 驗證租戶應用程式儲存庫：空的儲存庫或設定錯誤的儲存庫可能會導致錯誤。請確定租戶應用程式儲存庫包含必要的程式碼。
	2. 重新部署`tenant_mgmt` 模組：在`tenant_mgmt` 模組組態檔案中，找到`app`區塊，然後將 `deploy` 參數設定為 `0`：
	`deploy = 0`
	執行 Terraform `apply`命令之後，請將`deploy`參數值變更回 `1`：
	`deploy = 1`
	3. 重新檢查狀態：執行先前步驟後，請使用下列命令來檢查問題是否仍然存在：
	`kubectl get gitrepositories -A`
	如果持續存在，請考慮深入了解 Flux 日誌以取得更多詳細資訊，或參閱 Flux 一般故障診斷指南。

相關資源

- Terraform 的 Amazon EKS 藍圖
- Amazon EKS 最佳實務指南，多租戶區段
- Flux 網站
- Helm 網站

其他資訊

以下是部署租戶應用程式的範例儲存庫結構：

```
applications
sample_tenant_app
### README.md
### base
#   ### configmap.yaml
#   ### deployment.yaml
#   ### ingress.yaml
#   ### kustomization.yaml
#   ### service.yaml
### overlays
    ### tenant-1
    #   ### configmap.yaml
    #   ### deployment.yaml
    #   ### kustomization.yaml
    ### tenant-2
        ### configmap.yaml
        ### kustomization.yaml
```

使用自動化工作流程簡化 Amazon Lex 機器人開發和部署

Balaji Panneerselvam、Attila Dancso、Pavan Dusanapudi、Anand Jumnani 和 Amazon Web Services 的 James O'Hara

總結

當您嘗試管理多個功能、開發人員和環境時，開發和部署 Amazon Lex 對話式機器人可能具有挑戰性。使用基礎設施即程式碼 (IaC) 原則的自動化工作流程有助於簡化程序。此模式有助於提高 Amazon Lex 開發人員的生產力，並透過下列方式實現高效率的機器人生命週期管理：

- 啟用並行開發多個功能 - 透過自動化工作流程，開發人員可以在不同的分支中平行處理不同的功能。然後，可以合併和部署變更，而不會封鎖其他工作。
- 使用 Amazon Lex 主控台 UI - 開發人員可以使用易於使用的 Amazon Lex 主控台來建置和測試機器人。然後，機器人會在用於部署的基礎設施程式碼中描述。
- 跨環境提升機器人 - 工作流程會自動從開發和測試等較低環境提升機器人版本，直到生產。此方法可降低手動提升的風險和額外負荷。
- 維護版本控制 - 在 Git 中管理機器人定義，而不是僅透過 Amazon Lex 服務為您提供版本控制和稽核線索。與僅使用 AWS 管理主控台 或 APIs 修改存放在其中的機器人不同，變更會追蹤到個別開發人員 AWS。

透過自動化 Amazon Lex 機器人發行程序，團隊可以更快地提供功能，同時降低風險和精力。機器人保持在版本控制下，而不是在 Amazon Lex 主控台中隔離。

先決條件和限制

先決條件

- 工作流程 AWS 帳戶 針對不同的環境 （ 開發、生產和 DevOps) 涉及多個 ，這需要帳戶管理和跨帳戶存取組態。
- Python 3.9 可在您的部署環境或管道中使用。
- 在本機工作站上安裝https://git-scm.com/book/en/v2/Getting-Started-Installing-Git和設定 Git 以進行來源控制。
- AWS Command Line Interface (AWS CLI) 已安裝並設定為使用命令列或 Python 進行身分驗證。

限制

- 儲存庫存取 – 工作流程假設持續整合和持續交付 (CI/CD) 管道具有將變更遞交至原始程式碼儲存庫的必要許可。
- 初始機器人版本 – 工具要求使用 AWS CloudFormation 範本部署機器人的初始版本。您必須建立機器人的第一次反覆運算，並將其遞交至儲存庫，自動化工作流程才能接管。
- 合併衝突 – 雖然工作流程旨在啟用並行開發，但在整合來自不同分支的變更時，仍有可能發生合併衝突。解決機器人組態中的衝突可能需要手動介入。

產品版本

- Python 3.9 或更新版本
- AWS CDK v2 2.124.0 或更新版本
- 適用於 Python (Boto3) 的 AWS SDK1.28 或更新版本

Architecture

下圖顯示解決方案的高階架構和關鍵元件。

主要元件包括下列項目：

- Lex 機器人儲存庫 – 存放 Amazon Lex 機器人 IaC 定義的 Git 儲存庫。
- DevOps – AWS 帳戶 專用於容納 CI/CD 管道和開發和部署程序的相關資源。
- 管道 – 自動化機器人開發和部署生命週期各種階段的 AWS CodePipeline 執行個體，例如建立新的機器人、匯出機器人的定義、匯入機器人定義，以及刪除機器人。
- 票證機器人和主要機器人 – Amazon Lex 機器人資源，其中票證機器人是由個別團隊或開發人員開發的特定功能機器人，而主要機器人是整合所有功能的基準機器人。

架構圖說明下列工作流程：

1. 基準主要機器人 – 工作流程的起點是在開發 (Dev) 環境中基準化主要機器人。主要機器人是未來開發和功能新增的基礎。
2. 建立票證機器人 – 需要新功能或變更時，會建立票證機器人。票證機器人基本上是開發人員可以處理的主要機器人複本或分支，而不會影響主要版本。

3. 匯出票證機器人 - 處理票證機器人完成後，它會從 Amazon Lex 服務匯出。然後，包含票證機器人的分支會從主分支重新建立基礎。此步驟可確保在票證機器人開發期間對主要機器人所做的任何變更都已納入，以減少潛在的衝突。
4. 匯入以重新為基礎的票證機器人並進行驗證 – 以重新為基礎的票證機器人會匯入到開發環境並進行驗證，以確保其與主分支的最新變更一起正常運作。如果驗證成功，則會建立提取請求 (PR)，將票證機器人變更合併到主分支。
5. 刪除票證機器人 – 變更成功合併到主分支後，不再需要票證機器人。您可以刪除票證機器人，以保持環境乾淨且可管理。
6. 將主要機器人部署到開發環境並測試 – 更新的主要機器人現在包括新功能或變更，已部署到開發環境。在這裡，它會進行徹底的測試，以確保所有功能都能如預期般運作。
7. 將主要機器人部署到生產環境中 – 在開發環境中測試完成且成功之後，主要機器人會部署到生產環境。此步驟是工作流程的最後階段，其中新功能可供最終使用者使用。

自動化和擴展

自動化工作流程可讓開發人員平行處理不同的功能，每個功能都位於不同的分支中。這有助於並行開發，使團隊能夠有效地協作並更快地交付功能。在分支彼此隔離的情況下，可以合併和部署變更，而不會封鎖或干擾其他進行中的工作。

工作流程可自動化機器人版本在不同環境中的部署和提升，例如開發、測試和生產。

在 Git 等版本控制系統中存放機器人定義可提供全面的稽核線索，並實現高效的協同合作。系統會追蹤個別開發人員的變更，確保整個開發生命週期的透明度和責任。這種方法也有助於程式碼檢閱，讓團隊在部署到生產環境之前識別和解決問題。

透過使用 AWS CodePipeline 和其他 AWS 服務，自動化工作流程可以擴展以適應不斷增加的工作負載和團隊規模。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) 是一種開放原始碼軟體開發架構，可透過使用熟悉的程式設計語言並佈建來定義程式碼中的 AWS 雲端 基礎設施 CloudFormation。此模式中的範例實作使用 Python。
- AWS CDK 命令列界面 (AWS CDK CLI) - Toolkit AWS CDK 是與您的 AWS CDK 應用程式互動的主要工具。它會執行您的應用程式、查詢您定義的應用程式模型，以及產生和部署 CDK 產生的 CloudFormation 範本。

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶和 生命週期中管理資源 AWS 區域。此模式使用 CloudFormation，使用基礎設施做為程式碼來部署 Amazon Lex 機器人組態和相關資源。
- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。此模式使用 CodeBuild 來建置和封裝部署成品。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。此模式使用 CodePipeline 來協調持續交付管道。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您AWS 服務 透過命令列 shell 中的命令與 互動。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- Amazon Lex V2 是 AWS 服務 ，用於使用語音和文字為應用程式建置對話介面 （機器人）。
- 適用於 Python (Boto3) 的 AWS SDK 是一種軟體開發套件，可協助您整合 Python 應用程式、程式庫或指令碼 AWS 服務。

其他工具

- Git 是一種開放原始碼分散式版本控制系統。

程式碼儲存庫

此模式的程式碼可在 GitHub management-framework-sample-for-amazon-lex 儲存庫中使用。程式碼儲存庫包含下列資料夾和檔案：

- `prerequisite` 資料夾 – 包含設定所需資源和環境的 CloudFormation 堆疊定義 （使用 AWS CDK)。
- `prerequisite/lexmgmtworkflow` 資料夾 – Lex 管理工作流程專案的主要目錄，包括堆疊定義和 Python 程式碼。
- `prerequisite/tests` – 包含單元測試。
- `src` 資料夾 – 原始程式碼目錄，包括 Amazon Lex 機器人管理包裝函式和公用程式。
- `src/dialogue_lambda` – 對話掛鉤 Lambda 函數的原始程式碼目錄，可在與 Amazon Lex 機器人的對話期間攔截和處理使用者輸入。

最佳實務

- 分離問題
  - 在 DevOps、開發和生產環境之間保持明確的責任分離。
  - 針對 AWS 帳戶 每個環境使用單獨的 ，以強制執行適當的隔離和安全性界限。
  - 使用跨帳戶角色和最低權限存取原則，以確保環境之間的受控制存取。
- 基礎設施即程式碼
  - 定期檢閱和更新基礎設施程式碼，以符合最佳實務和不斷變化的需求。
  - 為原始程式碼儲存庫建立明確的分支和合併策略
- 測試和驗證
  - 在管道的各個階段實作自動化測試，以在開發週期的早期發現問題。
  - 使用 Amazon Lex 主控台或自動測試架構來驗證機器人組態和功能，然後再提升到更高的環境。
  - 考慮為部署到生產環境或關鍵環境實作手動核准閘道。
- 監控和記錄
  - 設定管道、部署和機器人互動的監控和記錄機制。
  - 監控管道事件、部署狀態和機器人效能指標，以快速識別和解決問題。
  - 使用 Amazon CloudWatch 等 AWS 服務 AWS CloudTrail，以及 AWS X-Ray 進行集中式記錄和監控。
  - 定期檢閱和分析自動化工作流程的效能、效率和有效性。
- 安全性與合規
  - 實作安全編碼實務，並遵循 Amazon Lex 機器人開發和部署 AWS 的安全最佳實務。
  - 定期檢閱和更新 IAM 角色、政策和許可，以符合最低權限原則。
  - 考慮將安全掃描和合規檢查整合到管道中。

史詩

設定 Amazon Lex 機器人管理的 IaC

任務	Description	所需的技能
設定本機 CDK 環境。	1. 若要複製此模式的儲存庫並導覽至 `prerequisite` 目錄，請執行下列命令：	AWS DevOps

任務	Description	所需的技能

	`git clone https://github.com/aws-samples/management-framework-sample-for-amazon-lex.git`
	
	`cd management-framework-sample-for-amazon-lex`
	
	2. 若要安裝和啟用 Python 虛擬環境，請執行下列命令，在專案資料夾本機安裝 CDK 的相依性，而非全域安裝：
	
	`pip install virtualenv`
	`python<version> -m venv .venv`
	`source .venv/bin/activate`
	`python -m pip install -r requirements.txt`	

任務	Description	所需的技能

在 devops 環境中建立跨帳戶角色。	devops 帳戶負責託管和管理 CI/CD 管道。若要讓 CI/CD 管道與 dev和 prod環境互動，請執行下列命令以在devops帳戶中建立跨帳戶角色。
	`cdk bootstrap --profile =devops`
	`cdk deploy LexMgmtDe vopsRoleStack -c dev-account-id=222 2222222222 -c prod- account-id=33333333333 3 --profile=devops`
		AWS DevOps

在 dev 環境中建立跨帳戶角色。	在dev帳戶中建立具有必要許可的 IAM 角色，以允許devops帳戶擔任此角色。CI/CD 管道會使用此角色在dev帳戶中執行動作，例如部署和管理 Amazon Lex 機器人資源。
	若要建立 IAM 角色，請執行下列命令：
	`cdk bootstrap --profile =dev`
	`cdk deploy LexMgmtCr ossaccountRoleStac k -c devops-account- id=1111111111111 -- profile=dev`
		AWS DevOps

任務	Description	所需的技能

在 prod 環境中建立跨帳戶角色。	在prod帳戶中建立具有必要許可的 IAM 角色，以允許devops帳戶擔任此角色。CI/CD 管道會使用此角色在prod帳戶中執行動作，例如部署和管理 Amazon Lex 機器人資源。
	
	`cdk bootstrap --profile =prod`
	
	`cdk deploy LexMgmtCr ossaccountRoleStac k -c devops-account- id=1111111111111 -- profile=prod`
		AWS DevOps

在 devops 環境中建立管道。	若要管理 Amazon Lex 機器人的開發工作流程，請執行下列命令以在devops環境中設定管道 。 `cdk deploy LexMgmtWo rkflowStack -c devops- account-id=111111111 1111 -c dev-account- id=2222222222222 -c prod-account-id=33 3333333333 --profile =devops`	AWS DevOps

建立主要機器人的基準

任務	Description	所需的技能
定義主要機器人的初始版本。	若要定義主要機器人的初始版本，請觸發`BaselineBotPipeline` 管道。 管道會部署 CloudFormation 範本中定義的基本機器人定義，將主要機器人定義匯出為 .json 檔案。 並將主要機器人程式碼存放在版本控制系統中。	AWS DevOps

實作功能開發工作流程

任務	Description	所需的技能
建立票證機器人以開發和測試功能。	`TicketBot` 是從功能分支中現有主要機器人定義匯入的新機器人執行個體。此方法可確保新機器人具有主機器人的所有目前功能和組態。 若要定義票證機器人的初始版本，請觸發`CreateTicketBotPipeline` 管道。 管道會在版本控制系統中建立新的功能分支，並根據主要機器人建立新的票證機器人執行個體。	Lex Bot 開發人員
開發和測試票證機器人功能。	若要開發和測試此功能，請登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/lex/	Lex Bot 開發人員

任務	Description	所需的技能
	的 Amazon Lex 主控台。如需詳細資訊，請參閱《Amazon Lex 文件》中的使用主控台測試機器人。 使用`TicketBot` 執行個體，您現在可以新增、修改或擴展機器人的功能，以實作新功能。例如，您可以建立或修改意圖、表達用語、槽和對話方塊流程。如需詳細資訊，請參閱 Amazon Lex 文件中的新增意圖。	
匯出票證機器人定義。	匯出的機器人定義基本上是以 JSON 格式呈現機器人的組態和功能。 若要匯出票證機器人定義，請觸發`ExportTicketBotPipeline` 管道。 管道會將票證機器人定義匯出為 .json 檔案，並將票證機器人程式碼存放在版本控制系統中的功能分支中。	Lex Bot 開發人員

任務	Description	所需的技能
從最新的主分支重新建立特徵分支的基礎。	在開發新功能期間，主要分支可能已收到來自不同開發人員或團隊的其他變更。 若要將這些變更納入功能分支，請執行 Git `rebase`操作。此操作基本上會在來自主分支的最新遞交之上，從特徵分支重播遞交，以確保特徵分支包含所有最新的變更	Lex Bot 開發人員
匯入和驗證以重新為基礎的票證機器人。	重新建立特徵分支的基礎之後，您必須將其匯入票證機器人執行個體。此匯入會使用重新為基礎的分支的最新變更來更新現有的票證機器人。 若要匯入以重新為基礎的票證機器人，請觸發`ImportTicketBotPipeline` 管道。 管道會將版本控制系統中特徵分支中的票證機器人定義 .json 檔案匯入`TicketBot` 執行個體。	Lex Bot 開發人員

任務	Description	所需的技能
驗證重新型機器人定義。	在您匯入重新型機器人定義之後，驗證其功能至關重要。您想要確保新功能如預期般運作，且不會與現有功能衝突。 此驗證通常涉及使用各種輸入案例測試機器人、檢查回應，以及驗證機器人的行為是否如預期。您可以透過下列其中一種方式執行驗證： • 使用 Amazon Lex 主控台手動測試機器人。 • 使用可模擬使用者互動並宣告預期回應的測試架構和工具，來使用自動化方法。	Lex Bot 開發人員
將功能分支合併到主分支。	在隔離的`TicketBot` 執行個體中開發和測試新功能之後，請執行下列動作： 1. 將變更遞交至版本控制系統中對應的功能分支。 2. 若要將功能分支合併到主分支，請建立提取請求 (PR)。此 PR 可做為檢閱並將變更納入主要程式碼庫的請求。	Lex Bot Developer，儲存庫管理員

任務	Description	所需的技能
刪除功能分支和票證機器人。	將功能分支成功合併至主分支後，請從原始碼儲存庫刪除功能分支和票證機器人。 若要刪除功能分支和票證機器人，請觸發`DeleteTicketBotPipeline` 管道。 管道會移除開發過程中建立的臨時機器人資源（例如票證機器人）。此動作有助於維護乾淨的儲存庫，並防止與未來的特徵分支混淆或衝突。	Lex Bot 開發人員

維護主要機器人

任務	Description	所需的技能
將最新的主要機器人定義匯入`dev`環境。	若要將主分支中最新的主機器人定義匯入`dev`環境，請觸發`DeployBotDevPipeline` 管道。 管道也會在核准時建立 git 標籤。	AWS DevOps
將最新的主要機器人定義匯入`prod`環境。	若要將主分支中最新的機器人定義匯入`prod`環境，請提供先前任務的標籤參考做為參數，並觸發`DeployBotProdPipeline` 管道。 管道會將最新的機器人定義從特定標籤匯入`prod`環境。	AWS DevOps

疑難排解

問題	解決方案
當您將 Amazon Lex 機器人部署到不同的 時 AWS 帳戶，工具服務必須具有存取這些帳戶中資源的必要許可。	若要授予跨帳戶存取權，請使用 IAM 角色和政策。在目標帳戶中建立 IAM 角色，並將政策連接到授予必要許可的角色。然後，從部署 Amazon Lex 機器人的帳戶擔任這些角色。 如需詳細資訊，請參閱 Amazon Lex 文件中的匯入所需的 IAM 許可和在 Lex V2 中匯出機器人所需的 IAM 許可。

相關資源

- 在 Amazon Lex V2 中匯入機器人
- 在 CodePipeline 中啟動管道
- 使用 Amazon Lex V2 機器人

使用 AWS Fargate WaitCondition 勾點建構來協調資源相依性和任務執行

Stan Fan，Amazon Web Services

摘要

此模式描述 WaitCondition hook (`waitcondition-hook-for-aws-fargate-task`) npm 套件，這是雲端原生解決方案，專為在 Amazon Elastic Container Service (Amazon ECS) 叢集中協調AWS Fargate任務而設計。

WaitCondition 掛鉤是一種專門為 整合量身打造的 AWS Cloud Development Kit (AWS CDK) 建構 AWS CloudFormation。WaitCondition 掛鉤提供下列關鍵功能：

- 做為等待條件機制，暫停 CloudFormation 堆疊執行，直到指定的 Fargate 任務完成，這有助於有序的部署和資源佈建。
- 支援 TypeScript 和 Python，使其非常適合 AWS CDK 專案。
- 允許開發人員和架構師協調容器化應用程式的任務完成和資源管理，以協調部署 AWS。
- 使用內嵌在 CloudFormation 生命週期中的一或多個容器來執行 Fargate 任務。 和 可以在任務失敗後處理任務失敗並復原 CloudFormation 堆疊。
- 提供彈性，在資源與 Fargate 任務執行結果之間新增相依性，啟用自訂任務或叫用其他端點。例如，您可以暫停 CloudFormation 堆疊並等待資料庫遷移 （ 由 Fargate 任務完成 ），並佈建可能取決於資料庫遷移成功的其他資源。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- AWS Cloud Development Kit (AWS CDK) 安裝在本機工作站上的命令列界面 (CLI)。如需詳細資訊，請參閱 AWS CDK 文件中的 AWS CDK CLI 參考。
- 節點套件管理員 (npm)，安裝在本機工作站上，並在 AWS CDK TypeScript 中為 設定。如需詳細資訊，請參閱 npm 文件中的下載並安裝 Node.js 和 npm。
- 安裝在本機工作站上的 Yarn。如需詳細資訊，請參閱 Yarn 文件中的 安裝。

限制

- 此解決方案會部署到單一 AWS 帳戶。

- 容器的預期傳回碼0是為了成功。任何其他傳回碼表示失敗，CloudFormation 堆疊將復原。
- 有些 AWS 服務 無法全部使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

下圖顯示建構架構。

圖表顯示 的工作流程waitcondition-hook-for-aws-fargate-task：

1. WaitCondition 和 WaitConditionHandler 會佈建為接聽 AWS Lambda 函數的回應。
2. 根據任務的結果， CallbackFunction或 ErrorHandlerFunction是由 Fargate 任務的完成所觸發。
3. Lambda 函數會將 SUCCEED 或 FAILURE 訊號傳送至 WaitConditionHandler。
4. WaitConditionHandler 如果 Fargate 任務的執行結果成功， 會繼續佈建資源，或在任務失敗時轉返堆疊。

下圖顯示執行資料庫遷移的工作流程範例。

範例工作流程使用 waitcondition-hook-for-aws-fargate-task 建構來執行資料庫遷移，如下所示：

1. 已佈建 Amazon Relational Database Service (Amazon RDS) 執行個體。
2. waitcondition-hook-for-aws-fargate-task 建構模組會執行資料庫遷移任務，並將堆疊暫停為 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。
3. 如果遷移任務成功完成，它會向 CloudFormation 傳送成功訊號。否則，它會將失敗訊號傳送至 CloudFormation 並轉返堆疊。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義雲端基礎設施並進行佈建 CloudFormation。

- CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和 生命週期中管理資源 AWS 區域。
- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您在 上執行的應用程式。
- Amazon Elastic Container Service (Amazon ECS) 是快速、可擴展的容器管理服務，可協助您執行、停止和管理叢集上的容器。
- AWS Fargate 可協助您執行容器，而不需要管理伺服器或 Amazon EC2 執行個體。它與 Amazon ECS 搭配使用。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務 來建置業務關鍵型應用程式。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路類似於傳統網路，您可以在自己的資料中心中操作，並享有使用 可擴展基礎設施的優勢 AWS。

其他工具

- npm 是在 Node.js 環境中執行的軟體登錄檔，用於共用或借用套件和管理私有套件的部署。
- Yarn 是開放原始碼套件管理員，可用來管理 JavaScript 專案中的相依性。Yarn 可協助您安裝、更新、設定和移除套件相依性。

程式碼儲存庫

此模式的程式碼可在 GitHub waitcondition-hook-for-aws-fargate-task 儲存庫中使用。

最佳實務

- 建置 AWS CDK 應用程式時，請遵循 AWS CDK v2 文件中的使用 開發和部署雲端基礎設施的最佳實務 AWS CDK。
- 針對 AWS Fargate 任務，請遵循 Amazon ECS 文件中的 Amazon ECS 容器映像最佳實務。

史詩

設定 AWS CDK

任務	Description	所需的技能
安裝 AWS CDK。	若要 AWS CDK 在本機電腦或其他環境上安裝 ，請執行下列命令： `npm install -g aws-cdk@latest`	雲端架構師、應用程式開發人員
引導 AWS CDK。	引導是準備環境以進行部署的程序。若要為目標引導 AWS CDK 您的工具組 AWS 區域，AWS 帳戶 並執行下列命令： `cdk bootstrap aws://ACCOUNT-NUMBER-1/REGION-1` 此命令會建立名為 的 CloudFormation 堆疊`CDKToolkit` 。	雲端架構師

執行 AWS Fargate 任務建構的 WaitCondition 掛鉤

任務	Description	所需的技能
建立 CDK 專案。	使用您偏好的語言建立 CDK 專案。此模式使用 TypeScript。若要使用 TypeScript 建立 CDK 專案，請執行下列命令： `cdk init app –language typescript`	雲端架構師

任務	Description	所需的技能
安裝套件。	在 CDK 專案的根路徑npm install上執行 。安裝 CDK 程式庫之後，請執行下列命令來安裝 `waitcondition-hook-for-aws-fargate-task` ： `yarn add waitcondition-hook-for-aws-fargate-task`	雲端架構師

任務	Description	所需的技能

建置您的 CDK 應用程式和 Amazon ECS 元件。	建置您的 CDK 專案。需要 Amazon ECS 任務定義資源。如需建立任務定義的資訊，請參閱 Amazon ECS 文件中的 Amazon ECS 任務定義。
	
	下列範例使用此建構：
	
	`import * as cdk from 'aws-cdk-lib';`
	`import { Vpc } from 'aws-cdk-lib/aws-ec2';`
	`import * as ecr from 'aws-cdk-lib/aws-ecr';`
	`import * as ecs from 'aws-cdk-lib/aws-ecs';`
	`import { Construct } from 'constructs';`
	`import { FargateRunner } from 'waitcondition-hook-for-aws-fargate-task';`
	`import { Queue } from 'aws-cdk-lib/aws-sqs';`
	
	`export class FargateRunnerStack extends cdk.Stack {`
	`constructor(scope: Construct, id: string, props?: cdk.StackProps) {`
	`super(scope, id, props);`
	`// Define the VPC`
		雲端架構師

任務	Description	所需的技能

	`const vpc = new Vpc(this, 'MyVpc')`
	`// Define the Fargate Task`
	`const taskDefinition = new ecs.FargateTaskDefinition(this, 'MyTask', {});`
	`// Import exiting ecr repo`
	`const repo = ecr.Repository.fromRepositoryName(this, 'MyRepo', 'RepoName');`
	`// Add a container to the task`
	`taskDefinition.addContainer('MyContainer', {`
	`image: ecs.ContainerImage.fromEcrRepository(repo),`
	`});`
	`// Create the Fargate runner`
	`const myFargateRunner = new FargateRunner(this, 'MyRunner', {`
	`fargateTaskDef: taskDefinition,`
	`timeout: ` ${60 * 5}`,`
	`vpc: vpc,`
	`});`
	`// Create the SQS queue`
	`const myQueue = new Queue(this, 'MyQueue', {});`	

任務	Description	所需的技能
	`// Add dependency myQueue.node.addDependency(myFargateRunner); } }`	

合成並啟動 CDK 應用程式。	1. 若要產生資產和 CloudFormation 範本，請在 CDK 根路徑中執行下列命令：
	`cdk synth`
	2. `synth` 命令成功後，請執行下列命令來部署資源：
	`cdk deploy`
	建構執行 Fargate `waitcondition-hook-for-aws-fargate-task` 任務。
		雲端架構師

清除

任務	Description	所需的技能
清除資源。	若要清除上一個步驟佈建的資源，請執行下列命令： `cdk destroy`	雲端架構師

疑難排解

問題	解決方案
一般 CloudFormation 堆疊失敗	若要協助疑難排解一般 CloudFormation 堆疊失敗，請新增 `--no-rollback` 旗標，如下列範例所示： `cdk deploy --no-rollback` 此命令會暫停 CloudFormation 堆疊的復原，讓您進行疑難排解。如需詳細資訊，請參閱在 CloudFormation 文件中佈建資源時選擇如何處理失敗。
AWS Step Functions 失敗	AWS Step Functions 狀態機器可能會因不同原因而無法執行。`–disable-rollback` 設定後，請使用下列步驟進行故障診斷： 1. 登入 AWS 管理主控台，在搜尋欄位中輸入 Step Functions，然後選擇 Step Functions 服務。 2. 在左側導覽窗格中，選擇狀態機器，然後選取 CloudFormation 堆疊佈建的狀態機器。 3. 在執行中，選擇意外失敗的執行名稱。 4. 在事件檢視中，選擇失敗的步驟。 如需詳細資訊，請參閱 AWS Step Functions 文件中的 Step Functions 中的疑難排解問題和 Step Functions 主控台中的檢視執行詳細資訊。
AWS Lambda 函數失敗	此建構模組會佈建兩個 Lambda 函數：`CallbackFunction` 和 `ErrorhandlerFunction`。它們可能會因各種原因而失敗，例如未處理的例外狀況。使用下列步驟進行疑難排解：

問題	解決方案
	1. 登入 AWS 管理主控台，在搜尋欄位中輸入 CloudWatch，然後選擇 CloudWatch 服務。 2. 在左側導覽窗格中，選擇 Log groups (日誌群組)。 3. 在搜尋欄位中，輸入 Lambda 函數的名稱。 4. 選擇與 Lambda 函數相關聯的日誌群組名稱。 5. 若要前往 Lambda 函數執行結果，請選擇最新的日誌串流。 如需詳細資訊，請參閱 AWS Lambda 文件中的對 Lambda 中的問題進行故障診斷。

相關資源

AWS 文件

- AWS CDK 建構 API 參考
- 開始使用 AWS CDK
- 了解如何建立和使用 Amazon ECS 資源
- 了解如何開始使用 Step Functions
- 什麼是 AWS CDK？

其他資源

- AWS Fargate 任務的等待條件掛接 (npm)
- waitcondition-hook-for-aws-fargate-task 1.0.6 (pypi.org：//)

在 AWS CodePipeline 中使用第三方 Git 來源儲存庫

Kirankumar Chandrashekar，Amazon Web Services

總結

此模式說明如何搭配第三方 Git 來源儲存庫使用 AWS CodePipeline。

AWS CodePipeline 是一種持續交付服務，可自動化建置、測試和部署軟體的任務。此服務目前支援 GitHub 儲存庫。 AWS CodeCommit 不過，有些企業使用與單一登入 (SSO) 服務和 Microsoft Active Directory 整合的第三方 Git 儲存庫進行身分驗證。您可以透過建立自訂動作和 Webhook，使用這些第三方 Git 儲存庫做為 CodePipeline 的來源。

Webhook 是一種 HTTP 通知，能在另一個工具中 (例如 GitHub 儲存庫) 偵測事件，並連接這些外部事件至管道。當您在 CodePipeline 中建立 Webhook 時，服務會傳回您可以在 Git 儲存庫 Webhook 中使用的 URL。如果您將程式碼推送至 Git 儲存庫的特定分支，Git Webhook 會透過此 URL 啟動 CodePipeline Webhook，並將管道的來源階段設定為進行中。當管道處於此狀態時，任務工作者會輪詢自訂任務的 CodePipeline、執行任務，並將成功或失敗狀態傳送至 CodePipeline。在此情況下，由於管道位於來源階段，任務工作者會使用輪詢任務提供的物件金鑰，取得 Git 儲存庫的內容、壓縮內容，並將其上傳至存放管道成品的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。您也可以將自訂動作的轉換與 Amazon CloudWatch 中的事件建立關聯，並根據事件啟動任務工作者。此設定可讓您使用服務原生不支援作為 CodePipeline 來源的第三方 Git 儲存庫。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 支援 Webhook 並可透過網際網路連線至 CodePipeline Webhook URL 的 Git 儲存庫
- 已安裝並設定 AWS Command Line Interface (AWS CLI) 以使用 AWS 帳戶

Architecture

模式涉及以下步驟：

1. 使用者將程式碼遞交至 Git 儲存庫。
2. Git Webhook 稱為 。
3. CodePipeline Webhook 稱為 。
4. 管道設定為進行中，而來源階段設定為進行中狀態。

5. 來源階段動作會啟動 CloudWatch Events 規則，指出已啟動。
6. CloudWatch 事件會啟動 Lambda 函數。
7. Lambda 函數會取得自訂動作任務的詳細資訊。
8. Lambda 函數會啟動 AWS CodeBuild 並傳遞所有與任務相關的資訊。
9. CodeBuild 會從 Secrets Manager 取得 HTTPS Git 存取的公有 SSH 金鑰或使用者憑證。
10. CodeBuild 會複製特定分支的 Git 儲存庫。
11. CodeBuild 會壓縮封存檔，並將其上傳至做為 CodePipeline 成品存放區的 S3 儲存貯體。

工具

- AWS CodePipeline – AWS CodePipeline 是一項全受管的持續交付服務，可協助您自動化發行管道，以實現快速可靠的應用程式和基礎設施更新。CodePipeline 會根據您定義的發行模型，將每個程式碼變更的發行程序建置、測試和部署階段自動化。這可讓您快速且可靠地交付功能和更新。您可以將 AWS CodePipeline 與 GitHub 等第三方服務或您自己的自訂外掛程式整合。
- AWS Lambda – AWS Lambda 可讓您執行程式碼，而無需佈建或管理伺服器。使用 Lambda，您可以為幾乎任何類型的應用程式或後端服務執行程式碼，而不需要管理。您可以上傳程式碼，Lambda 會處理執行和擴展程式碼所需的一切，並提供高可用性。您可以設定程式碼以自動從其他 AWS 服務啟動，或直接從任何 Web 或行動應用程式呼叫。
- AWS CodeBuild – AWS CodeBuild 是全受管的持續整合服務，可編譯原始程式碼、執行測試，並產生可立即部署的軟體套件。使用 CodeBuild，您不需要佈建、管理和擴展自己的建置伺服器。CodeBuild 會持續擴展並同時處理多個組建，所以您的組建不必排入佇列中等候。您可以利用預先封裝好的組建環境立即開始使用，或是建立自訂的組建環境來使用您自己的組建工具。
- AWS Secrets Manager – AWS Secrets Manager 可協助您保護存取應用程式、服務和 IT 資源所需的秘密。此服務可讓您在整個生命週期輪換、管理和擷取資料庫登入資料、API 金鑰和其他秘密。使用者和應用程式透過呼叫 Secrets Manager APIs 來擷取秘密，而無需以純文字硬式編碼敏感資訊。Secrets Manager 提供秘密輪換與 Amazon Relational Database Service (Amazon RDS)、Amazon Redshift 和 Amazon DocumentDB 的內建整合。服務可以擴展以支援其他類型的秘密，包括 API 金鑰和 OAuth 權杖。此外，Secrets Manager 可讓您使用精細的許可來控制對秘密的存取，並針對 AWS 雲端、第三方服務和內部部署環境中的資源集中稽核秘密輪換。
- Amazon CloudWatch – Amazon CloudWatch 是一種監控和觀察服務，專為 DevOps 工程師、開發人員、網站可靠性工程師 (SREs) 和 IT 管理員而打造。CloudWatch 為您提供資料和可行的洞見，以監控您的應用程式、回應整個系統的效能變更、最佳化資源使用率，以及取得營運運作狀態的統一

檢視。CloudWatch 會以日誌、指標和事件的形式收集監控和操作資料，讓您統一檢視在 AWS 和內部部署伺服器上執行的 AWS 資源、應用程式和服務。您可以使用 CloudWatch 偵測環境中的異常行為、設定警示、並排視覺化日誌和指標、採取自動化動作、疑難排解問題，以及探索洞見，讓您的應用程式順暢運作。

- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種物件儲存服務，可讓您針對各種使用案例存放和保護任意數量的資料，例如網站、行動應用程式、備份和還原、封存、企業應用程式、IoT 裝置和大數據分析。Amazon S3 easy-to-use管理功能，可協助您整理資料，並設定微調後的存取控制，以符合您的特定業務、組織和合規需求。

史詩

在 CodePipeline 中建立自訂動作

任務	Description	所需的技能
使用 AWS CLI 或 AWS CloudFormation 建立自訂動作。	此步驟涉及建立自訂來源動作，可用於特定區域中 AWS 帳戶中管道的來源階段。您必須使用 AWS CLI 或 AWS CloudFormation （而非主控台）來建立自訂來源動作。如需此和其他 epics 中所述命令和步驟的詳細資訊，請參閱此模式結尾的「相關資源」一節。在 AWS CLI 中，使用 create-custom-action-type 命令。使用 --configuration-properties 提供任務工作者在輪詢任務的 CodePipeline 時需要處理的所有參數。請務必記下提供給 --provider 和 --action-version 選項的值，以便在使用此自訂來源階段建立管道時使用相同的值。您也可以使用資源類型 AWS：：CodePipeline：：	一般 AWS

任務	Description	所需的技能
	CustomActionType，在 AWS CloudFormation 中建立自訂來源動作。 AWS::CodePipeline::CustomActionType	

設定身分驗證

任務	Description	所需的技能
建立 SSH 金鑰對。	建立 Secure Shell (SSH) 金鑰對。如需說明，請參閱 GitHub 文件。	系統/DevOps 工程師
在 AWS Secrets Manager 中建立秘密。	從 SSH 金鑰對複製私有金鑰的內容，並在 AWS Secrets Manager 中建立秘密。存取 Git 儲存庫時，此秘密會用於身分驗證。	一般 AWS
將公有金鑰新增至 Git 儲存庫。	將公有金鑰從 SSH 金鑰對新增至 Git 儲存庫帳戶設定，以針對私有金鑰進行身分驗證。	系統/DevOps 工程師

建立管道和 Webhook

任務	Description	所需的技能
建立包含自訂來源動作的管道。	在 CodePipeline 中建立管道。當您設定來源階段時，請選擇您先前建立的自訂來源動作。您可以在 AWS CodePipeline 主控台或 AWS CLI 中執行此操作。CodePipeline 會提示您輸入您在自訂動作上設定的組態	一般 AWS

任務	Description	所需的技能
	屬性。需要此資訊，任務工作者才能處理自訂動作的任務。遵循精靈並建立管道的下一個階段。	
建立 CodePipeline Webhook。	為您使用自訂來源動作建立的管道建立 Webhook。您必須使用 AWS CLI 或 AWS CloudFormation（而非主控台）來建立 Webhook。在 AWS CLI 中，執行 put-webhook 命令，並提供 Webhook 選項的適當值。請記下 命令傳回的 Webhook URL。如果您使用 AWS CloudFormation 建立 Webhook，請使用資源類型 AWS::CodePipeline::Webhook。請務必從建立的資源輸出 Webhook URL，並將其記下。	一般 AWS

任務	Description	所需的技能
建立 Lambda 函數和 CodeBuild 專案。	在此步驟中，您可以使用 Lambda 和 CodeBuild 建立任務工作者，以輪詢 CodePipeline 以取得自訂動作的任務請求、執行任務，並將狀態結果傳回 CodePipeline。當管道的自訂來源動作階段轉換為「進行中」時，建立由 Amazon CloudWatch Events 規則啟動的 Lambda 函數。啟動 Lambda 函數時，它應該透過輪詢任務來取得自訂動作任務詳細資訊。您可以使用 PollForJobs API 傳回此資訊。取得輪詢的任務資訊後，Lambda 函數應傳回確認，然後使用其從自訂動作的組態屬性取得的資料來處理資訊。當工作者準備好與 Git 儲存庫交談時，您可以啟動 CodeBuild 專案，因為使用 SSH 用戶端處理 Git 任務非常方便。	General AWS，程式碼開發人員

在 CloudWatch 中建立事件

任務	Description	所需的技能
建立 CloudWatch Events 規則。	建立 CloudWatch Events 規則，每當管道的自訂動作階段轉換為「進行中」時，就會啟動 Lambda 函數做為目標。	一般 AWS

相關資源

在 CodePipeline 中建立自訂動作

- 在 CodePipeline 中建立和新增自訂動作
- AWS::CodePipeline::CustomActionType 資源

設定身分驗證

- 使用 AWS Secrets Manager 建立和管理秘密

建立管道和 Webhook

- 在 CodePipeline 中建立管道
- put-webhook 命令參考
- AWS::CodePipeline::Webhook 資源
- PollForJobs API 參考
- 在 CodePipeline 中建立和新增自訂動作
- 在 AWS CodeBuild 中建立組建專案

建立事件

- 使用 Amazon CloudWatch Events 偵測管道狀態的變更並做出反應

其他參考

- 在 CodePipeline 中使用管道
- AWS Lambda 開發人員指南

使用 AWS CodePipeline 建立 CI/CD 管道來驗證 Terraform 組態

Aromal Raj Jayarajan 和 Vijesh Vijayakumaran Nair，Amazon Web Services

總結

此模式示範如何使用 AWS CodePipeline 部署的持續整合和持續交付 (CI/CD) 管道來測試 HashiCorp Terraform 組態。

Terraform 是一種命令列界面應用程式，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。此模式中提供的解決方案會建立 CI/CD 管道，協助您執行五個 CodePipeline 階段來驗證 Terraform 組態的完整性：

1. "checkout" 會從 AWS CodeCommit 儲存庫提取您正在測試的 Terraform 組態。
2. "validate" 會以程式碼 (IaC) 驗證工具的形式執行基礎設施，包括 tfsec、TFLint 和 checkov。階段也會執行下列 Terraform IaC 驗證命令： `terraform validate`和 `terraform fmt`。
3. "plan" 顯示如果套用 Terraform 組態，將套用哪些變更到基礎設施。
4. "apply" 使用產生的計劃在測試環境中佈建所需的基礎設施。
5. "destroy" 會移除在"apply"階段期間建立的測試基礎設施。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- AWS Command Line Interface (AWS CLI)，已安裝並設定
- 在本機電腦上安裝和設定的 Git
- 本機電腦上安裝和設定的 Terraform

限制

- 此模式的方法只會將 AWS CodePipeline 部署到一個 AWS 帳戶和 AWS 區域。多帳戶和多區域部署需要變更組態。
- 此模式佈建的 AWS Identity and Access Management (IAM) 角色 (codepipeline_iam_role) 遵循最低權限原則。此 IAM 角色的許可必須根據管道需要建立的特定資源進行更新。

產品版本

- AWS CLI 2.9.15 版或更新版本
- Terraform 1.3.7 版或更新版本

Architecture

目標技術堆疊

- AWS CodePipeline
- AWS CodeBuild
- AWS CodeCommit
- AWS IAM
- Amazon Simple Storage Service (Amazon S3)
- AWS Key Management Service (AWS KMS)
- Terraform

目標架構

下圖顯示在 CodePipeline 中測試 Terraform 組態的範例 CI/CD 管道工作流程。

該圖顯示以下工作流程：

1. 在 CodePipeline 中，AWS 使用者透過在 AWS CLI 中執行 `terraform apply`命令，啟動 Terraform 計劃中提議的動作。
2. AWS CodePipeline 擔任 IAM 服務角色，其中包含存取 CodeCommit、CodeBuild、AWS KMS 和 Amazon S3 所需的政策。
3. CodePipeline 會執行`"checkout"`管道階段，從 AWS CodeCommit 儲存庫提取 Terraform 組態以供測試。
4. CodePipeline 會在 CodeBuild 專案中執行 IaC 驗證工具和執行 Terraform IaC 驗證命令，以執行`"validate"`階段來測試 Terraform 組態。
5. CodePipeline 會執行 `"plan"`階段，根據 Terraform 組態在 CodeBuild 專案中建立計畫。AWS 使用者可以在將變更套用至測試環境之前檢閱此計畫。
6. Code Pipeline 會執行`"apply"`階段來實作計畫，方法是使用 CodeBuild 專案在測試環境中佈建所需的基礎設施。

7. CodePipeline 會執行"destroy"階段，使用 CodeBuild 移除"apply"階段期間建立的測試基礎設施。
8. Amazon S3 儲存貯體會存放管道成品，這些成品會使用 AWS KMS 客戶受管金鑰進行加密和解密。

工具

工具

AWS 服務

- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- AWS CodeBuild 是一種全受管的建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他服務

- HashiCorp Terraform 是一種命令列界面應用程式，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。

Code

此模式的程式碼可在 GitHub aws-codepipeline-terraform-cicdsamples 儲存庫中使用。儲存庫包含建立此模式中概述之目標架構所需的 Terraform 組態。

史詩

佈建解決方案元件

任務	Description	所需的技能
複製 GitHub 儲存庫。	在終端機視窗中執行下列命令，複製 GitHubaws-codepipeline-terraform-cicdsamples： `git clone https://github.com/aws-samples/aws-codepipeline-terraform-cicd-samples.git` 如需詳細資訊，請參閱 GitHub 文件中的複製儲存庫。	DevOps 工程師
建立 Terraform 變數定義檔案。	根據您的使用案例需求建立`terraform.tfvars` 檔案。您可以更新複製儲存庫中`examples/terraform.tfvars` 檔案中的變數。 如需詳細資訊，請參閱 Terraform 文件中的將值指派給根模組變數。 **Note** 儲存庫的 `Readme.md` 檔案包含所需變數的詳細資訊。	DevOps 工程師
將 AWS 設定為 Terraform 提供者。	1. 在程式碼編輯器中，開啟複製的儲存庫`main.tf`檔案。	DevOps 工程師

任務	Description	所需的技能
	2. 新增必要的組態，以建立與目標 AWS 帳戶的連線。 如需詳細資訊，請參閱 Terraform 文件中的 AWS 提供者。	
更新用於建立 Amazon S3 複寫儲存貯體的 Terraform 提供者組態。	1. 執行下列命令以開啟儲存庫的S3目錄： `cd ./modules/s3` 2. 透過更新 `tf` 檔案中`region`的值，更新用於建立 Amazon S3 複寫儲存貯體的 Terraform 提供者組態。請務必輸入您希望 Amazon S3 複寫物件的區域。 3. （選用）根據預設，Terraform 會使用本機狀態檔案進行狀態管理。如果您想要新增 Amazon S3 做為遠端後端，則必須更新 Terraform 組態。如需詳細資訊，請參閱 Terraform 文件中的後端組態。 **Note** 複寫會在 Amazon S3 儲存貯體中啟用物件的自動非同步複製。	DevOps 工程師

任務	Description	所需的技能
初始化 Terraform 組態。	若要初始化包含 Terraform 組態檔案的工作目錄，請在複製的儲存庫根資料夾中執行下列命令： `terraform init`	DevOps 工程師
建立 Terraform 計劃。	若要建立 Terraform 計劃，請在複製的儲存庫根資料夾中執行下列命令： `terraform plan --var-file=terraform.tfvars -out=tfplan` **Note** Terraform 會評估組態檔案，以判斷宣告資源的目標狀態。然後，它會比較目標狀態與目前狀態，並建立計劃。	DevOps 工程師
驗證 Terraform 計劃。	檢閱 Terraform 計劃，並確認它在您的目標 AWS 帳戶中設定所需的架構。	DevOps 工程師

任務	Description	所需的技能
部署解決方案。	1. 若要套用 Terraform 計劃，請在複製的儲存庫根資料夾中執行下列命令： `terraform apply "tfplan"` 2. 輸入 yes 以確認您想要部署資源。 **Note** Terraform 會建立、更新或銷毀基礎設施，以達到組態檔案中宣告的目標狀態。	DevOps 工程師

執行管道來驗證 Terraform 組態

任務	Description	所需的技能
設定原始程式碼儲存庫。	1. 從 Terraform 輸出中，取得包含您要驗證之 Terraform 組態之儲存庫的來源儲存庫詳細資訊。 2. 登入 AWS 管理主控台。然後，開啟 CodeCommit 主控台。 3. 在名為 的來源儲存庫中建立新的分支`main`。如需說明，請參閱 CodeCommit 文件中的在 AWS CodeCommit 中建立分支。 CodeCommit	DevOps 工程師

任務	Description	所需的技能
	4. 將來源儲存庫的`main`分支複製到本機工作站。如需說明，請參閱 AWS CodeCommit CLI 登入資料協助程式在 Windows 上設定 HTTPS 連線至 AWS CodeCommit 儲存庫的步驟。 CodeCommit 5. 執行下列命令，從 GitHubaws-codepipeline-terraform-cicdsamples 複製 `templates` 資料夾： `cp -r templates $YOUR_CODECOMMIT_REPO_ROOT` **Note** `templates` 資料夾包含建置規格檔案，以及來源儲存庫根目錄的驗證指令碼。 6. 將所需的 Terraform IaC 組態新增至來源儲存庫的根資料夾。 7. 在專案的 Terraform 組態中新增遠端後端的詳細資訊。如需詳細資訊，請參閱 Terraform 文件中的 S3。 8. （選用）更新 `templates` 資料夾中的變數，以啟用或停用預先設定的掃描、	

任務	Description	所需的技能
	工具變更版本，並在自訂指令碼檔案中指定您的目錄。如需詳細資訊，請參閱此模式的其他資訊一節。 9. 將變更推送至來源儲存庫的`main`分支。	
驗證管道階段。	1. 登入 AWS 管理主控台並開啟 CodePipeline 主控台。 2. 在上一個 Epic 區段中從`terraform apply "tfplan"`命令產生的輸出中，尋找產生的 CodePipeline 名稱。 3. 在 CodePipeline 主控台中開啟管道，然後選擇發行變更。 4. 檢閱每個管道階段，並確認其正常運作。 如需詳細資訊，請參閱《AWS CodePipeline 使用者指南》中的檢視管道詳細資訊和歷史記錄（主控台）。 ⚠ Important 將變更遞交至來源儲存庫的主分支時，會自動啟用測試管道。	DevOps 工程師

任務	Description	所需的技能
驗證報告輸出。	1. 在 CodePipeline 主控台的左側導覽窗格中，選擇建置。然後，選擇報告歷史記錄。 2. 檢閱管道產生的 tfsec 和 checkov 掃描報告。這些報告可協助您透過視覺化和圖形呈現來識別問題。 **Note** `<project_name>-validate` CodeBuild 專案會在`"validate"` 階段期間為您的程式碼產生漏洞報告。	DevOps 工程師

清除您的資源

任務	Description	所需的技能
清除管道和相關聯的資源。	若要從您的 AWS 帳戶刪除測試資源，請在複製的儲存庫根資料夾中執行下列命令： `terraform destroy --var-file=terraform.tfvars`	DevOps 工程師

疑難排解

問題	解決方案
您在"apply"階段期間收到 AccessDenied 錯誤。	1. 檢閱與"apply"階段相關聯的 CodeBuild 專案執行日誌，以識別任何缺少的 IAM 許可。如需詳細資訊，請參閱《AWS CodeBuild 使用者指南》中的在 AWS CodeBuild 中檢視建置詳細資訊。 AWS CodeBuild 2. 在程式碼編輯器中，開啟複製的儲存庫資料夾modules 。然後，導覽至 iam-role 資料夾，並開啟該資料夾中main.tf的檔案。 3. 在 codepipeline_policy 陳述式中，新增在 AWS 帳戶中佈建資源所需的 IAM 政策。

相關資源

- 模組區塊 (Terraform 文件）
- 如何使用 CI/CD 透過 Terraform 部署和設定 AWS 安全服務 (AWS 部落格文章）
- 使用服務連結角色 (IAM 文件）
- create-pipeline (AWS CLI 文件）
- 針對存放在 Amazon S3 for CodePipeline 中的成品設定伺服器端加密 (AWS CodePipeline 文件）
- AWS CodeBuild 配額 (AWS CodeBuild 文件）
- AWS CodePipeline 中的資料保護 (AWS CodePipeline 文件）

其他資訊

自訂 Terraform 模組

以下是在此模式中使用的自訂 Terraform 模組清單：

- codebuild_terraform 會建立構成管道每個階段的 CodeBuild 專案。
- codecommit_infrastructure_source_repo 會擷取並建立來源 CodeCommit 儲存庫。

- codepipeline_iam_role 會為管道建立所需的 IAM 角色。
- codepipeline_kms 會建立 Amazon S3 物件加密和解密所需的 AWS KMS 金鑰。
- codepipeline_terraform 會為來源 CodeCommit 儲存庫建立測試管道。
- s3_artifacts_bucket 會建立 Amazon S3 儲存貯體來管理管道成品。

建置規格檔案

以下是組建規格 (buildspec) 檔案的清單，此模式用於執行每個管道階段：

- buildspec_validate.yml 會執行"validate"階段。
- buildspec_plan.yml 會執行"plan"階段。
- buildspec_apply.yml 會執行"apply"階段。
- buildspec_destroy.yml 會執行"destroy"階段。

建置規格檔案變數

每個 buildspec 檔案使用以下變數來啟用不同的組建特定設定：

變數	預設值	Description
CODE_SRC_DIR	"."	定義來源 CodeCommit 目錄
TF_VERSION	"1.3.7"	定義建置環境的 Terraform 版本

buildspec_validate.yml 檔案也支援下列變數，以啟用不同的建置特定設定：

變數	預設值	Description
SCRIPT_DIR	"./templates/scripts"	定義指令碼目錄
ENVIRONMENT	「開發」	定義環境名稱
SKIPVALIDATIONFAILURE	「Y」	略過失敗的驗證

ENABLE_TFVALIDATE	「Y」	啟用 Terraform 驗證
ENABLE_TFFORMAT	「Y」	啟用 Terraform 格式
ENABLE_TFCHECKOV	「Y」	啟用檢查掃描
ENABLE_TFSEC	「Y」	啟用 tfsec 掃描
TFSEC_VERSION	「v1.28.1」	定義 tfsec 版本

更多模式

- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon EKS 上私下存取容器應用程式
- 將儲存 AWS CodeCommit 庫與另一個帳戶中 AWS 帳戶 的 Amazon SageMaker AI Studio Classic 建立關聯
- 在 上使用登陸區域加速器自動建立帳戶 AWS
- 使用 AWS Systems Manager 自動化新增或更新 Windows 登錄項目
- 使用 AWS Batch 自動化 Amazon RDS for PostgreSQL 資料庫執行個體的備份
- 使用 AWS SAM 自動化巢狀應用程式的部署
- 使用 CI/CD 管道在 Amazon EKS 中自動化節點終止處理常式的部署
- 在 Amazon MQ 中自動化 RabbitMQ 組態 Amazon MQ
- 自動化跨 的 Amazon RDS 執行個體複寫 AWS 帳戶
- 使用 CI/CD 管道自動建置 Java 應用程式並將其部署到 Amazon EKS
- 使用 Python 應用程式自動產生 Amazon DynamoDB 的 PynamoDB 模型和 CRUD 函數 DynamoDB
- 使用 CodePipeline、IAM Access Analyzer 和 AWS CloudFormation 巨集自動驗證和部署 IAM 政策和角色
- 在 上的 Stromasys Charon-SSP 模擬器中備份 Sun SPARC 伺服器 AWS 雲端
- 建置資料管道，以使用 AWS DataOps 開發套件擷取、轉換和分析 Google Analytics 資料
- 使用 Amazon EC2 Auto Scaling 和 Systems Manager 建置 Micro Focus Enterprise Server PAC
- 使用 EC2 Image Builder 和 Terraform 建置強化容器映像的管道
- 使用 Amazon SageMaker AI 和 Azure DevOps 建置 MLOps 工作流程
- 使用 AWS Managed Microsoft AD 和內部部署 Microsoft Active Directory 集中 DNS 解析
- 在狀態檔案遺失後，安全地清除 AWS Account Factory for Terraform (AFT) 資源
- 使用 NLog 在 Amazon CloudWatch Logs 中設定 .NET 應用程式的記錄
- 跨 AWS 帳戶 和 複製 Amazon ECR 容器映像 AWS 區域
- 為 SageMaker 建立自訂 Docker 容器映像，並將其用於 AWS Step Functions 中的模型訓練
- 在不支援 AWS CodePipeline 的 AWS 區域中建立管道
- 使用 Amazon CloudWatch 異常偵測為自訂指標建立警示
- 使用 AWS CDK 層面和逃生艙自訂預設角色名稱
- 部署可同時偵測多個程式碼交付項目中安全問題的管道
- 使用基礎設施做為程式碼，在 AWS 雲端上部署和管理無伺服器資料湖

- 在做為 Docker 容器 AWS IoT Greengrass V2 執行時部署容器化應用程式
- 在 Amazon S3 中使用 Amazon EKS 和 Helm Chart 儲存庫部署 Kubernetes 資源和套件
- 使用 AWS CDK 搭配 TypeScript 部署多堆疊應用程式
- 搭配 Kiro 和其他編碼助理使用 MCP 伺服器部署即時編碼安全驗證
- 使用 Terraform 在 Amazon EC2 和 Amazon FSx 上部署 SQL Server 容錯移轉叢集執行個體
- 使用 Terraform 部署 AWS WAF 解決方案的安全自動化
- 使用 RAG 和 ReAct 提示，開發進階生成式 AI 聊天式助理
- 使用 AWS CloudFormation 範本有條件地啟用 Amazon GuardDuty
- 使用 Amazon EKS Pod Identity 和 KEDA 在 Amazon EKS 中設定事件驅動的自動擴展
- 使用 Amazon Personalize 產生個人化和重新排名的建議
- 當 AWS KMS 金鑰的金鑰狀態變更時，取得 Amazon SNS 通知
- 使用 Account Factory for Terraform 管理多個帳戶的許可集
- 遷移至 Amazon ECR 儲存庫時，自動識別重複的容器映像
- 在 Amazon API Gateway 中使用自訂網域實作路徑型 API 版本控制
- 使用 AWS CDK 跨多個 AWS 區域、帳戶和 OUs 啟用 Amazon DevOps Guru，以改善營運效能
- 使用 Kubernetes DaemonSet 在 Amazon EKS 工作者節點上安裝 SSM Agent
- 將stonebranch 通用控制器與 AWS Mainframe Modernization 整合
- 大型主機現代化：DevOps on AWS with Rocket Software Enterprise Suite
- 使用 以程式碼形式管理 AWS IAM Identity Center 許可集 AWS CodePipeline
- 使用 Terraform 動態管理 AWS 許可集
- 使用 AWS CDK 設定 Amazon ECS Anywhere 來管理內部部署容器應用程式
- 使用 AWS CodePipeline 和 Amazon Bedrock 以程式碼形式管理 AWS Organizations 政策
- 將大量 DNS 記錄遷移至 Amazon Route 53 私有託管區域
- 使用 appcmd.exe 將 IIS 託管應用程式遷移至 Amazon EC2
- 跨多個 監控共用 Amazon Machine Image 的使用 AWS 帳戶
- 使用 透過驗證、轉換和分割來協調 ETL 管道 AWS Step Functions
- 使用 IaC 原則自動化 Amazon Aurora 全域資料庫的藍/綠部署
- 在非工作負載子網路的多帳戶 VPC 設計中保留可路由 IP 空間
- AWS Service Catalog 使用程式碼儲存庫在 中佈建 Terraform 產品
- 從 AWS Step Functions 同步執行 AWS Systems Manager 自動化任務 AWS Step Functions

- 使用 AWS CDK 和 GitLab 在 Amazon ECS Anywhere 上設定混合工作負載的 CI/CD 管道
- 使用 Terraform 設定資料庫遷移的 CI/CD 管道
- 使用 Amazon FSx 設定 SQL Server Always On FCI 的異地同步備份基礎設施
- 使用 AWS CloudFormation 在 Amazon EC2 上自動設定 UiPath RPA 機器人
- 使用 Application Load Balancer 在 Amazon ECS 中使用交互 TLS 簡化應用程式身分驗證
- 使用 C# 和 AWS CDK 在孤立模型的 SaaS 架構中加入租用戶
- 使用 Terraform 自動為組織啟用 Amazon GuardDuty
- 使用 Amazon Bedrock 代理程式，透過文字型提示在 Amazon EKS 中自動建立存取項目控制項
- 在本機驗證帳戶工廠的 Terraform (AFT) 程式碼
- 使用 Flask 和 AWS Elastic Beanstalk 視覺化 AI/ML 模型結果

基礎設施

主題

- 使用 Session Manager 和 Amazon EC2 Instance Connect 存取堡壘主機
- 使用 AWS Managed Microsoft AD 和內部部署 Microsoft Active Directory 集中 DNS 解析
- 使用 Amazon CloudWatch Observability Access Manager 集中監控
- 在啟動時檢查 EC2 執行個體是否有強制性標籤
- 在狀態檔案遺失後，安全地清除 AWS Account Factory for Terraform (AFT) 資源
- 在不支援 AWS CodePipeline 的 AWS 區域中建立管道
- 使用 AWS CDK 層面和逃生艙自訂預設角色名稱
- 使用私有靜態 IPs 在 Amazon EC2 上部署 Cassandra 叢集，以避免重新平衡
- 使用 AWS Transit Gateway Connect 將 VRFs 擴展至 AWS AWS Transit Gateway
- 當 AWS KMS 金鑰的金鑰狀態變更時，取得 Amazon SNS 通知
- 在非工作負載子網路的多帳戶 VPC 設計中保留可路由 IP 空間
- AWS Service Catalog 使用程式碼儲存庫在 中佈建 Terraform 產品
- 使用 Amazon SES AWS 帳戶 向單一電子郵件地址註冊多個
- 在單一帳戶 AWS 環境中設定混合網路的 DNS 解析
- 使用 AWS CloudFormation 在 Amazon EC2 上自動設定 UiPath RPA 機器人
- 在 AWS 上設定高度可用的 PeopleSoft 架構
- 使用 AWS Elastic Disaster Recovery 為 Oracle JD Edwards EnterpriseOne 設定災難復原
- 在多區域、多帳戶組織中設定 CloudFormation 偏離偵測
- 成功將 S3 儲存貯體匯入為 AWS CloudFormation 堆疊
- 使用 AWS DataSync 同步不同 AWS 區域中 Amazon EFS 檔案系統之間的資料 DataSync
- 使用 LocalStack 和 Terraform Tests 測試 AWS 基礎設施
- 從 ENSA1 升級 SAP Pacemaker 叢集到 ENSA2
- 在不同 AWS 帳戶中使用 VPCs 中的一致可用區域
- 在 IAM 政策中使用使用者 IDs 進行存取控制和自動化
- 在本機驗證帳戶工廠的 Terraform (AFT) 程式碼
- 更多模式

使用 Session Manager 和 Amazon EC2 Instance Connect 存取堡壘主機

Piotr Chotkowski 和 Witold Kowalik，Amazon Web Services

總結

堡壘主機有時稱為跳轉方塊，是一種伺服器，可提供從外部網路到私有網路中資源的單一存取點。向網際網路等外部公有網路公開的伺服器，會對未經授權的存取造成潛在安全風險。請務必保護和控制對這些伺服器的存取。

此模式說明如何使用 Session Manager 和 Amazon EC2 Instance Connect 安全地連線至部署在 中的 Amazon Elastic Compute Cloud (Amazon EC2) 堡壘主機 AWS 帳戶。Session Manager 是 的功能 AWS Systems Manager。此模式的優點包括：

- 部署的堡壘主機沒有任何開放的傳入連接埠會公開至公有網際網路。這可減少潛在的攻擊面。
- 您不需要在 中存放和維護長期 Secure Shell (SSH) 金鑰 AWS 帳戶。反之，每個使用者每次連線到堡壘主機時都會產生新的 SSH 金鑰對。 AWS Identity and Access Management (連接至使用者 AWS 登入資料的 IAM) 政策會控制對堡壘主機的存取。

目標對象

此模式適用於對 Amazon EC2、Amazon Virtual Private Cloud (Amazon VPC) 和 Hashicorp Terraform 有基本了解經驗的讀者。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- AWS Command Line Interface (AWS CLI) 第 2 版，已安裝並設定
- 的 Session Manager 外掛程式 AWS CLI，已安裝
- Terraform CLI，已安裝
- Terraform 狀態的儲存體，例如 Amazon Simple Storage Service (Amazon S3) 儲存貯體和 Amazon DynamoDB 資料表，可做為遠端後端來存放 Terraform 狀態。如需針對 Terraform 狀態使用遠端後端的詳細資訊，請參閱 Amazon S3 後端 (Terraform 文件）。如需使用 Amazon S3 後端設定遠端狀態管理的程式碼範例，請參閱 remote-state-s3-backend (Terraform Registry)。請注意以下要求：
  - Amazon S3 儲存貯體和 DynamoDB 資料表必須位於相同的 中 AWS 區域。

  - 建立 DynamoDB 資料表時，分割區索引鍵必須是 `LockID`（區分大小寫），而分割區索引鍵類型必須是 `String`。所有其他資料表設定必須處於其預設值。如需詳細資訊，請參閱 DynamoDB 文件中的關於主索引鍵和建立資料表。
- SSH 用戶端，已安裝

限制

- 此模式旨在做為概念驗證 (PoC) 或做為進一步開發的基礎。它不應在生產環境中以目前的形式使用。部署之前，請調整儲存庫中的範本程式碼，以符合您的需求和使用案例。
- 此模式假設目標堡壘主機使用 Amazon Linux 2 做為其作業系統。雖然可以使用其他 Amazon Machine Image (AMIs)，但其他作業系統超出此模式的範圍。

> **Note**
>
> Amazon Linux 2 即將終止支援。如需詳細資訊，請參閱 Amazon Linux 2 FAQs。

- 在此模式中，堡壘主機位於沒有 NAT 閘道和網際網路閘道的私有子網路中。此設計會將 Amazon EC2 執行個體與公有網際網路隔離。您可以新增特定的網路組態，以允許其與網際網路通訊。如需詳細資訊，請參閱 Amazon VPC 文件中的將您的虛擬私有雲端 (VPC) 連線到其他網路。同樣地，遵循最低權限原則，除非您明確授予許可， AWS 帳戶 否則堡壘主機無法存取 中的任何其他資源。如需詳細資訊，請參閱 IAM 文件中的資源型政策。

產品版本

- AWS CLI 第 2 版
- Terraform 1.3.9 版

Architecture

目標技術堆疊

- 具有單一私有子網路的 VPC
- 下列界面 VPC 端點：
  - `amazonaws.<region>.ssm` – AWS Systems Manager 服務的端點。
  - `amazonaws.<region>.ec2messages` – Systems Manager 使用此端點從 SSM Agent 呼叫 Systems Manager 服務。

  - `amazonaws.<region>.ssmmessages` – Session Manager 使用此端點透過安全的資料通道連線至 Amazon EC2 執行個體。
- 執行 `t3.nano` Amazon Linux 2 的 Amazon EC2 執行個體
- IAM 角色和執行個體描述檔
- 端點和 Amazon EC2 執行個體的 Amazon VPC 安全群組和安全群組規則

目標架構

圖表顯示下列程序：

1. 使用者擔任的 IAM 角色具有執行下列動作的許可：
   - 驗證、授權和連線至 Amazon EC2 執行個體
   - 使用 Session Manager 啟動工作階段
2. 使用者透過 Session Manager 啟動 SSH 工作階段。
3. Session Manager 會驗證使用者、驗證相關聯 IAM 政策中的許可、檢查組態設定，以及傳送訊息給 SSM Agent 以開啟雙向連線。
4. 使用者透過 Amazon EC2 中繼資料將 SSH 公有金鑰推送至堡壘主機。這必須在每個連線之前完成。SSH 公有金鑰保持可用 60 秒。
5. 堡壘主機會與 Systems Manager 和 Amazon EC2 的介面 VPC 端點通訊。
6. 使用者使用 TLS 1.2 加密的雙向通訊管道，透過 Session Manager 存取堡壘主機。

自動化和擴展

下列選項可用於自動化部署或擴展此架構：

- 您可以透過持續整合和持續交付 (CI/CD) 管道部署架構。
- 您可以修改程式碼來變更堡壘主機的執行個體類型。
- 您可以修改程式碼以部署多個堡壘主機。在 `bastion-host/main.tf` 檔案`aws_instance`的資源區塊中，新增`count`中繼引數。如需詳細資訊，請參閱 Terraform 文件。

工具

AWS 服務

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Systems Manager 可協助您管理在 中執行的應用程式和基礎設施 AWS 雲端。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。此模式使用 Systems Manager 的功能 Session Manager。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。

其他工具

- HashiCorp Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。此模式使用 Terraform CLI。

程式碼儲存庫

此模式的程式碼可在 GitHub 中使用 Session Manager 和 Amazon EC2 Instance Connect 儲存庫存取堡壘主機中取得。

最佳實務

- 建議使用自動程式碼掃描工具來改善程式碼的安全性和品質。此模式是透過使用 Checkov 進行掃描，這是 IaC 的靜態程式碼分析工具。我們建議您至少使用 和 `terraform fmt -check -recursive` Terraform 命令來執行基本驗證`terraform validate`和格式化檢查。
- 為 IaC 新增自動化測試是很好的做法。如需測試 Terraform 程式碼之不同方法的詳細資訊，請參閱測試 HashiCorp Terraform (Terraform 部落格文章）。
- 在部署期間，每次偵測到新版本的 Amazon EC22 執行個體。這會部署新版本的作業系統，包括修補程式和升級。如果部署排程不常發生，這可能會帶來安全風險，因為執行個體沒有最新的修補程式。請務必經常更新並套用安全性修補程式至部署的 Amazon EC2 執行個體。如需詳細資訊，請參閱在 Amazon EC2 中更新管理。
- 由於此模式是一種概念驗證，因此使用 AWS 受管政策，例如 `AmazonSSMManagedInstanceCore`. AWS managed 政策涵蓋常見的使用案例，但不授予最低權

限許可。根據您的使用案例，我們建議您建立自訂政策，以授予此架構中所部署資源的最低權限許可。如需詳細資訊，請參閱開始使用 AWS 受管政策並移至最低權限許可。

- 使用密碼來保護對 SSH 金鑰的存取，並將金鑰存放在安全的位置。
- 設定堡壘主機的記錄和監控。從操作和安全性的角度來看，記錄和監控是維護系統的重要部分。有多種方式可以監控堡壘主機中的連線和活動。如需詳細資訊，請參閱 Systems Manager 文件中的下列主題：
  - 監控 AWS Systems Manager
  - 在 中記錄和監控 AWS Systems Manager
  - 稽核工作階段活動
  - 記錄工作階段活動

史詩

部署 資源

任務	Description	所需的技能
複製程式碼儲存庫。	1. 在命令列界面中，將工作目錄變更為您要存放範例檔案的位置。 2. 輸入以下命令。 `git clone https://github.com/aws-samples/secured-bastion-host-terraform.git`	DevOps 工程師、開發人員
初始化 Terraform 工作目錄。	只有第一個部署需要此步驟。如果您要重新部署模式，請跳至下一個步驟。 在複製儲存庫的根目錄中，輸入下列命令，其中：	DevOps 工程師、開發人員、Terraform

任務	Description	所需的技能

	• $S3_STATE_BUCKET 是包含 Terraform 狀態的 Amazon S3 儲存貯體名稱
	• $PATH_TO_STATE_FILE 是 Terraform 狀態檔案的金鑰，例如 infra/bastion-host/tetfstate
	• $AWS_REGION 是部署 Amazon S3 儲存貯體的區域
	
	`terraform init \`
	`  -backend-config="bucket=$S3_STATE_BUCKET" \`
	`  -backend-config="key=$PATH_TO_STATE_FILE" \`
	`  -backend-config="region=$AWS_REGION`
	
	Note
	或者，您可以開啟 config.tf 檔案，並在 terraform 區段中手動提供這些值。	

任務	Description	所需的技能
部署 資源。	1. 在複製儲存庫的根目錄中，輸入下列命令。 `terraform apply -var-file="dev.tfvars"` 2. 檢閱將套用至您 的所有變更清單 AWS 帳戶，然後確認部署。 3. 等待所有資源部署完成。	DevOps 工程師、開發人員、Terraform

設定本機環境

任務	Description	所需的技能
設定 SSH 連線。	更新 SSH 組態檔案，以允許透過 Session Manager 進行 SSH 連線。如需說明，請參閱允許 Session Manager 的 SSH 連線。這可讓授權使用者輸入代理命令，以啟動 Session Manager 工作階段並透過雙向連線傳輸所有資料。	DevOps 工程師
產生 SSH 金鑰。	輸入下列命令以產生本機私有和公有 SSH 金鑰對。您可以使用此金鑰對來連線至堡壘主機。 `ssh-keygen -t rsa -f my_key`	DevOps 工程師、開發人員

使用 Session Manager 連線至堡壘主機

任務	Description	所需的技能

取得執行個體 ID。	1. 若要連線到部署的堡壘主機，您需要 Amazon EC2 執行個體的 ID。執行下列其中一項來尋找 ID：
	• 開啟 Amazon EC2 主控台。在導覽窗格中，選擇執行個體。找到堡壘主機執行個體。
	• 在 中 AWS CLI，輸入下列命令。
	`aws ec2 describe-instances`
	若要篩選結果，請輸入下列命令，其中 `$BASTION_HOST_TAG` 是您指派給堡壘主機的標籤。此標籤的預設值為 `sandbox-dev-bastion-host` 。
	`aws ec2 describe-instances \ --filters "Name=tag:Name,Values=$BASTION_HOST_TAG" \ --output text \ --query 'Reservations[*].Instances[*].InstanceId' \ --output text`
		一般 AWS

任務	Description	所需的技能
	2. 複製 Amazon EC2 執行個體的 ID。您稍後會使用此 ID。	

任務	Description	所需的技能
傳送 SSH 公有金鑰。	**Note** 在本節中，您將公有金鑰上傳到堡壘主機的執行個體中繼資料。上傳金鑰後，您有 60 秒的時間開始與堡壘主機的連線。60 秒後，會移除公有金鑰。如需詳細資訊，請參閱此模式的故障診斷一節。快速完成後續步驟，以防止在您連線到堡壘主機之前移除金鑰。 1. 使用 Amazon EC2 Instance Connect 將 SSH 金鑰傳送至堡壘主機。輸入下列命令，其中： • `$INSTANCE_ID` 是 Amazon EC2 執行個體的 ID • `$PUBLIC_KEY_FILE` 是您公有金鑰檔案的路徑，例如 `my_key.pub` **Important** 請務必使用公有金鑰，而非私有金鑰。	一般 AWS

任務	Description	所需的技能

	``
	aws ec2-instance-connect send-ssh-public-key \
	    --instance-id $INSTANCE_ID \
	    --instance-os-user ec2-user \
	    --ssh-public-key file://$PUBLIC_KEY_FILE
	``
	2. 請等到您收到訊息，指出金鑰已成功上傳。立即繼續下一個步驟。	

任務	Description	所需的技能
連線至堡壘主機。	1. 輸入下列命令，透過 Session Manager 連線至堡壘主機，其中： • `$PRIVATE_KEY_FILE` 是您私有金鑰的路徑，例如 `my_key` • `$INSTANCE_ID` 是 Amazon EC2 執行個體的 ID `ssh -i $PRIVATE_KEY_FILE ec2-user@$INSTANCE_ID` 2. 輸入 以確認連線`yes`。這會使用 Session Manager 開啟 SSH 連線。 **Note** 有其他選項可用來開啟與堡壘主機的 SSH 連線。如需詳細資訊，請參閱此模式額外資訊區段中與堡壘主機建立 SSH 連線的替代方法。	一般 AWS

(選用) 清除

任務	Description	所需的技能
移除已部署的資源。	1. 若要移除所有部署的資源，請從複製儲存庫的根目錄執行下列命令。 `terraform destroy -var-file="dev.tfvars"` 2. 確認資源的移除。	DevOps 工程師、開發人員、Terraform

疑難排解

問題	解決方案
`TargetNotConnected` 嘗試連線至堡壘主機時發生錯誤	1. 根據 Amazon EC2 文件中重新啟動執行個體中的指示，重新啟動堡壘主機。 2. 執行個體成功重新啟動後，將公有金鑰重新傳送至堡壘主機，然後重新嘗試連線。
`Permission denied` 嘗試連線至堡壘主機時發生錯誤	將公有金鑰上傳到堡壘主機後，您只有 60 秒的時間來啟動連線。60 秒後，金鑰會自動移除，而且您無法使用它連線到執行個體。如果發生這種情況，您可以重複 步驟，將金鑰重新傳送至執行個體。

相關資源

AWS 文件

- AWS Systems Manager Session Manager (Systems Manager 文件)
- 安裝 的 Session Manager 外掛程式 AWS CLI(Systems Manager 文件)
- 允許 Session Manager 的 SSH 連線 (Systems Manager 文件)

- 關於使用 EC2 Instance Connect (Amazon EC2 文件）
- 使用 EC2 Instance Connect 進行連線 (Amazon EC2 文件）
- Amazon EC2 的身分和存取管理 (Amazon EC2 文件）
- 使用 IAM 角色將許可授予在 Amazon EC2 執行個體上執行的應用程式 (IAM 文件）
- IAM 中的安全最佳實務 (IAM 文件）
- 使用安全群組控制資源的流量 (Amazon VPC 文件）

其他資源

- Terraform 開發人員網頁
- 命令：驗證 (Terraform 文件）
- 命令：fmt (Terraform 文件）
- 測試 HashiCorp Terraform (HashiCorp 部落格文章）
- Checkov 網頁

其他資訊

與堡壘主機建立 SSH 連線的替代方法

網路埠轉遞

您可以使用 `-D 8888`選項來開啟具有動態連接埠轉送的 SSH 連線。如需詳細資訊，請參閱 https://explainshell.com 的說明。以下是使用連接埠轉送開啟 SSH 連線的命令範例。

```
ssh -i $PRIVATE_KEY_FILE -D 8888 ec2-user@$INSTANCE_ID
```

這種連線會開啟 SOCKS 代理，透過堡壘主機從本機瀏覽器轉送流量。如果您使用的是 Linux 或 MacOS，若要查看所有選項，請輸入 `man ssh`。這會顯示 SSH 參考手冊。

使用提供的指令碼

您可以使用程式碼儲存庫中包含的 connect.sh 指令碼，而不是使用 Epics 區段中的 Session Manager 手動執行連線至堡壘主機中所述的步驟。此指令碼會產生 SSH 金鑰對、將公有金鑰推送至 Amazon EC2 執行個體，以及啟動與堡壘主機的連線。當您執行指令碼時，您可以將標籤和金鑰名稱做為引數傳遞。以下是執行指令碼的 命令範例。

```
./connect.sh sandbox-dev-bastion-host my_key
```

使用 AWS Managed Microsoft AD 和內部部署 Microsoft Active Directory 集中 DNS 解析

Brian Westmoreland，Amazon Web Services

總結

此模式提供使用 AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD) 和 Amazon Route 53 在 AWS 多帳戶環境中集中 DNS 解析的指引。在此模式中， AWS DNS 命名空間是內部部署 DNS 命名空間的子網域。此模式也提供有關如何設定現場部署 DNS 伺服器，以在現場部署 DNS 解決方案使用 Microsoft Active Directory AWS 時將查詢轉送至 的指引。

先決條件和限制

先決條件

- 使用 設定的 AWS 多帳戶環境 AWS Organizations。
- 之間建立的網路連線 AWS 帳戶。
- 在 AWS 和內部部署環境之間建立的網路連線 （ 使用 AWS Direct Connect 或任何類型的 VPN 連線 ）。
- AWS Command Line Interface 在本機工作站上設定的 (AWS CLI)。
- AWS Resource Access Manager (AWS RAM) 用於在帳戶之間共用 Route 53 規則。因此，共享必須在 AWS Organizations 環境中啟用，如 Epics 一節所述。

限制

- AWS Managed Microsoft AD Standard Edition 有 5 個共用的限制。
- AWS Managed Microsoft AD Enterprise Edition 有 125 個共用的限制。
- 此模式中的解決方案僅限於支援透過 AWS 區域 共用的 AWS RAM。

產品版本

- 在 Windows Server 2008、2012、2012 R2 或 2016 上執行的 Microsoft Active Directory。

Architecture

目標架構

在此設計中， AWS Managed Microsoft AD 安裝在共用服務中 AWS 帳戶。雖然這不是必要項目，但此模式會採用此組態。如果您在不同的 AWS Managed Microsoft AD 中設定 AWS 帳戶，您可能需要相應地修改 Epics 區段中的步驟。

此設計使用 Route 53 解析程式，透過使用 Route 53 規則來支援名稱解析。如果內部部署 DNS 解決方案使用 Microsoft DNS，為 AWS 命名空間 (aws.company.com) 建立條件式轉送規則，這是公司 DNS 命名空間 (company.com) 的子網域，並不直接。如果您嘗試建立傳統條件式轉送器，將導致錯誤。這是因為 Microsoft Active Directory 已被視為 任何子網域的授權company.com。若要解決此錯誤，您必須先為 建立委派，aws.company.com以委派該命名空間的授權。然後，您可以建立條件式轉寄站。

每個發言帳戶的虛擬私有雲端 (VPC) 可以根據根命名空間擁有自己的唯一 DNS AWS 命名空間。在此設計中，每個輪輻帳戶都會將帳戶名稱的縮寫附加到基本 AWS 命名空間。建立發言帳戶中的私有託管區域後，區域會與發言帳戶中的本機 VPC 以及中央 AWS 網路帳戶中的 VPC 建立關聯。這可讓中央 AWS 網路帳戶回答與輪輻帳戶相關的 DNS 查詢。如此一來，Route 53 和 都會 AWS Managed Microsoft AD 共同分擔管理 AWS 命名空間 () 的責任aws.company.com。

自動化和擴展

此設計使用 Route 53 Resolver 端點，在 AWS 和您的內部部署環境之間擴展 DNS 查詢。每個 Route 53 Resolver 端點包含多個彈性網路介面 （分散在多個可用區域），而且每個網路介面每秒最多可以處理 10，000 個查詢。Route 53 Resolver 支援每個端點最多 6 個 IP 地址，因此此設計支援每秒最多 60，000 個 DNS 查詢分散到多個可用區域，以實現高可用性。

此外，此模式會自動考慮內部的未來成長 AWS。在內部部署設定的 DNS 轉送規則不需要修改，即可支援新增的新 VPCs 及其相關聯的私有託管區域 AWS。

工具

AWS 服務

- AWS Directory Service for Microsoft Active Directory 可讓您的目錄感知工作負載 AWS 和資源在 中使用 Microsoft Active Directory AWS 雲端。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。
- AWS Resource Access Manager (AWS RAM) 可協助您安全地跨 共用資源 AWS 帳戶 ，以減少營運開銷並提供可見性和可稽核性。
- Amazon Route 53 是一種可用性高、可擴展性強的 DNS Web 服務。

工具

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。在此模式中， AWS CLI 用於設定 Route 53 授權。

史詩

建立和共用 AWS Managed Microsoft AD 目錄

任務	Description	所需的技能
部署 AWS Managed Microsoft AD。	1. 建立新的目錄。如需詳細步驟，請參閱《 AWS Directory Service 管理指南》中的建立您的 AWS Managed Microsoft AD 。 2. 記錄 AWS Managed Microsoft AD 網域控制站的 IP 地址。 這些將在後續步驟中參考。	AWS 管理員
共用 目錄。	建置目錄之後，請將其與 AWS 帳戶 AWS 組織中的其他 共用。如需說明，請參閱《 AWS Directory Service 管理指南》中的共用您的目錄。 **Note** AWS Managed Microsoft AD Standard Edition 有 5 個共用的限制。Enterprise Edition 有 125 個共用的限制。	AWS 管理員

設定路由 53

任務	Description	所需的技能
建立 Route 53 解析程式。	Route 53 Resolvers 可協助解決 AWS 和內部部署資料中心之間的 DNS 查詢。 1. 遵循 Route 53 開發人員指南中的 instructions 安裝 Route 53 解析程式。 2. 在中央 AWS 網路帳戶 VPC 內至少兩個可用區域中的私有子網路中設定 Route 53 解析程式，以獲得高可用性。 **Note** 雖然使用中央 AWS 網路帳戶 VPC 並非必要，但其餘步驟會採用此組態。	AWS 管理員
建立 Route 53 規則。	您的特定使用案例可能需要大量的 Route 53 規則，但您需要將下列規則設定為基準： • 內部部署命名空間 (company.com) 的傳出規則，方法是使用中央網路帳戶傳出 Route 53 解析程式。目標 IP 地址是內部部署 DNS 伺服器。 • 將此規則與中央網路帳戶 VPC 建立關聯。	AWS 管理員

任務	Description	所需的技能

	• 使用中央網路帳戶傳出 Route 53 解析程式的 AWS 命名空間傳出規則 (`aws.company.com`)。目標 IP 地址是中央網路帳戶傳入 Route 53 Resolver IP 地址。
	  • 請勿將此規則與中央 AWS 網路帳戶 VPC （包含 Route 53 解析程式） 建立關聯。
	• AWS 命名空間 (`aws.company.com`) 的第二個傳出規則，指向 AWS Managed Microsoft AD 網域控制站（使用上一個史詩IPs)。
	  • 將此規則與中央 AWS 網路帳戶 VPC （包含 Route 53 解析程式） 建立關聯。
	  • 請勿與其他 共用或關聯此規則 AWS 帳戶。
	
	如需詳細資訊，請參閱 Route 53 開發人員指南中的管理轉送規則。	

任務	Description	所需的技能
設定 Route 53 設定檔。	Route 53 設定檔用於與發言帳戶共用規則。 1. 遵循 Route 53 開發人員指南中的指示，在中央聯網帳戶中建立新的 Route 53 設定檔。 2. 將內部部署命名空間 (company.com) 的規則新增至設定檔。 3. 將 AWS 命名空間 (aws.company.com) 的第一個規則，以 Route 53 傳入解析程式的 IP 地址為目標，新增至設定檔。 4. 與 AWS 組織共用 Route 53 設定檔。 5. 接受每個發言帳戶中的 Route 53 Profile 資源共用。 6. 將 Route 53 設定檔與每個語音帳戶 VPC 建立關聯。	AWS 管理員

設定內部部署 Active Directory DNS

任務	Description	所需的技能
建立委派。	使用 Microsoft DNS 嵌入 (dnsmgmt.msc) 為 Active Directory 中的company.com 命名空間建立新的委派。委派網域的名稱應為 aws。這會使委派 的完整網域名稱 (FQDN)aws.compa	Active Directory

任務	Description	所需的技能
	ny.com 。使用 AWS Managed Microsoft AD 網域控制站的 IP 地址做為名稱伺服器 IP 值，並使用 server.aws.company.com 做為名稱。（此委派僅用於備援，因為會為此命名空間建立條件式轉送器，優先於委派。）	
建立條件式轉送器。	使用 Microsoft DNS 嵌入 (dnsmgmt.msc) 為 建立新的條件式轉送器aws.company.com 。 針對條件式轉送器 AWS 帳戶 的目標，在中央 DNS 中使用 AWS 傳入 Route 53 解析程式的 IP 地址。	Active Directory

建立發言的 Route 53 私有託管區域 AWS 帳戶

任務	Description	所需的技能
建立 Route 53 私有託管區域。	在每個發言帳戶中建立 Route 53 私有託管區域。將此私有託管區域與發言帳戶 VPC 建立關聯。如需詳細步驟，請參閱 Route 53 開發人員指南中的建立私有託管區域。	AWS 管理員
建立授權。	使用 AWS CLI 建立中央 AWS 網路帳戶 VPC 的授權。從每個語音的內容執行此命令 AWS 帳戶： aws route53 create-vpc-association-auth	AWS 管理員

任務	Description	所需的技能
	orization --hosted-zone-id <hosted-zone-id> \ --vpc VPCRegion=<region>,VPCId=<vpc-id> 其中： • <hosted-zone-id> 是輻條帳戶中的 Route 53 私有託管區域。 • <region> 和 <vpc-id>是中央 AWS 網路帳戶 VPC 的 AWS 區域 和 VPC ID。	

任務	Description	所需的技能
建立關聯。	使用 為中央 AWS 網路帳戶 VPC 建立 Route 53 私有託管區域關聯 AWS CLI。從中央 AWS 網路帳戶的內容執行此命令： `aws route53 associate-vpc-with-hosted-zone --hosted-zone-id <hosted-zone-id> \ --vpc VPCRegion=<region>,VPCId=<vpc-id>` 其中： • `<hosted-zone-id>` 是輻條帳戶中的 Route 53 私有託管區域。 • `<region>` 和 `<vpc-id>`是中央 AWS 網路帳戶的 AWS 區域 和 VPC ID。	AWS 管理員

相關資源

- 使用 Route 53 Resolver 簡化多帳戶環境中的 DNS 管理 (AWS 部落格文章）
- 建立 AWS Managed Microsoft AD(AWS Directory Service 文件）
- 共用 AWS Managed Microsoft AD 目錄 (AWS Directory Service 文件）
- 什麼是 Amazon Route 53 Resolver？ (Amazon Route 53 文件）
- 建立私有託管區域 (Amazon Route 53 文件）
- 什麼是 Amazon Route 53 設定檔？ (Amazon Route 53 文件）

使用 Amazon CloudWatch Observability Access Manager 集中監控

Anand Krishna Varanasi、JAGDISH KOMAKULA、Ashish Kumar、Jimmy Morgan、Sarat Chandra Pothula、Vivek Thangamuthu 和 Balaji Vedagiri、Amazon Web Services

摘要

可觀測性對於監控、了解和疑難排解應用程式至關重要。跨越多個帳戶的應用程式，如同 AWS Control Tower 或登陸區域實作，會產生大量日誌和追蹤資料。若要快速疑難排解問題或了解使用者分析或商業分析，您需要所有帳戶的通用可觀測性平台。Amazon CloudWatch Observability Access Manager 可讓您從中央位置存取和控制多個帳戶日誌。

您可以使用可觀測性存取管理員來檢視和管理來源帳戶產生的可觀測性資料日誌。來源帳戶是為其資源 AWS 帳戶 產生可觀測性資料的個人。可觀測性資料會在來源帳戶和監控帳戶之間共用。共用的可觀測性資料可包含 Amazon CloudWatch 中的指標、Amazon CloudWatch Logs 中的日誌，以及 中的追蹤 AWS X-Ray。如需詳細資訊，請參閱 Observability Access Manager 文件。

此模式適用於應用程式或基礎設施在多個 中執行， AWS 帳戶 且需要常見位置來檢視日誌的使用者。它說明如何使用 Terraform 設定可觀測性存取管理員，以監控這些應用程式或基礎設施的狀態和運作狀態。您可以透過多種方式安裝此解決方案：

- 作為您手動設定的獨立 Terraform 模組
- 透過使用持續整合和持續交付 (CI/CD) 管道
- 透過與其他解決方案整合，例如 AWS Control Tower Account Factory for Terraform (AFT)

Epics 區段中的指示涵蓋了手動實作。如需 AFT 安裝步驟，請參閱 GitHub Observability Access Manager 儲存庫的 README 檔案。

先決條件和限制

先決條件

- 系統或自動化管道中已安裝或參考的 Terraform。（我們建議您使用最新版本。）
- 您可以使用 做為中央監控帳戶的帳戶。其他帳戶會建立中央監控帳戶的連結，以檢視日誌。
- （選用）來源碼儲存庫，例如 GitHub AWS CodeCommit、Atlassian Bitbucket 或類似系統。如果您使用的是自動化 CI/CD 管道，則不需要原始程式碼儲存庫。
- （選用）在 GitHub 中建立提取請求 (PRs) 以進行程式碼檢閱和程式碼協同合作的許可。

限制

Observability Access Manager 具有下列服務配額，無法變更。部署此功能之前，請考慮這些配額。如需詳細資訊，請參閱 CloudWatch 文件中的 CloudWatch 服務配額。 CloudWatch

- 來源帳戶連結：您可以將每個來源帳戶連結至最多五個監控帳戶。
- 接收器：您可以為帳戶建立多個接收器，但每個 僅允許一個接收器 AWS 區域 。

此外：

- 接收器和連結必須在相同的 中建立 AWS 區域；它們不能是跨區域。

跨區域和跨帳戶監控

對於跨區域、跨帳戶監控，您可以選擇下列其中一個選項：

- 建立警示和指標的跨帳戶和跨區域 CloudWatch 儀表板。此選項不支援日誌和追蹤。
- 使用 Amazon OpenSearch Service 實作集中式記錄。
- 從所有租戶帳戶為每個區域建立一個接收器，將指標推送至集中式監控帳戶 （如此模式所述），然後使用 CloudWatch 指標串流將資料傳送至常見的外部目的地或第三方監控產品，例如 Datadog、Dynatrace、Sumo Logic、Splunk 或 New Relic。

Architecture

元件

CloudWatch Observability Access Manager 包含兩個主要元件，可啟用跨帳戶可觀測性：

- 接收器可讓來源帳戶將可觀測性資料傳送至中央監控帳戶。目的地基本上提供閘道連接，供來源帳戶連線。只能有一個接收器閘道或連線，而且多個帳戶可以與其連線。
- 每個來源帳戶都有目的地閘道連接的連結，而可觀測性資料會透過此連結傳送。您必須建立接收器，才能從每個來源帳戶建立連結。

架構

下圖說明 Observability Access Manager 及其元件。

工具

AWS 服務

- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您執行的應用程式。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。

工具

- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。
- AWS Control Tower Account Factory for Terraform (AFT) 會設定 Terraform 管道，協助您在其中佈建和自訂帳戶 AWS Control Tower。您可以選擇性地使用 AFT，跨多個帳戶大規模設定可觀測性存取管理員。

程式碼儲存庫

此模式的程式碼可在 GitHub Observability Access Manager 儲存庫中使用。

最佳實務

- 在 AWS Control Tower 環境中，將記錄帳戶標記為中央監控帳戶 （接收器）。
- 如果您在 中有多個組織擁有多個帳戶 AWS Organizations，建議您在組態政策中包含組織，而非個別帳戶。如果您有少量帳戶，或帳戶不在目的地組態政策中，您可以決定改為包含個別帳戶。

史詩

設定接收器模組

任務	Description	所需的技能
複製儲存庫。	複製 GitHub 可觀測性存取管理員儲存庫：	AWS DevOps、雲端管理員、AWS 管理員

任務	Description	所需的技能
	`git clone https://github.com/aws-samples/cloudwatch-observability-access-manager-terraform`	
指定目的地模組的屬性值。	在 `main.tf` 檔案中（在儲存庫的`deployments/aft-account-customizations/LOGGING/terraform/` 資料夾中），指定下列屬性的值： • `sink_name` ：CloudWatch 接收器的名稱。 • `allowed_oam_resource_types` ：可觀測性 Access Manager 目前支援 CloudWatch 指標、日誌群組和 AWS X-Ray 追蹤。 • `allowed_source_accounts` ：允許將日誌傳送至中央 CloudWatch 接收器帳戶的來源帳戶。 • `allowed_source_organizations` ：允許將日誌傳送至中央 CloudWatch 接收器帳戶的來源 AWS Control Tower 組織。 如需詳細資訊，請參閱 CloudFormation 文件中的 AWS::Oam::Sink。	AWS DevOps、雲端管理員、AWS 管理員

任務	Description	所需的技能

安裝接收器模組。	匯出 AWS 帳戶 您選取做為監控帳戶的 登入資料，並安裝可觀測性 Access Manager 接收器模組：
	`Terraform Init`
	`Terrafom Plan`
	`Terraform Apply`
		AWS DevOps、雲端管理員、AWS 管理員

設定連結模組

任務	Description	所需的技能

指定連結模組的屬性值。	在 `main.tf` 檔案中（在儲存庫的`deployments/aft-account-customizations/LOGGING/terraform/` 資料夾中），指定下列屬性的值：
	• `account_label` ：使用下列其中一個值：
	  • `$AccountName` ：帳戶的名稱。
	  • `$AccountEmail` ：全域唯一電子郵件地址，其中包含電子郵件網域（例如 `hello@example.com` ）
	  • `$AccountEmailNoDomain` ：沒有網域名稱的電子郵件地址。
	• `allowed_oam_resource_types` ：可觀測性
		AWS DevOps、雲端管理員、雲端架構師

任務	Description	所需的技能
	Access Manager 目前支援 CloudWatch 指標、日誌群組和 AWS X-Ray 追蹤。 如需詳細資訊，請參閱 CloudFormation 文件中的 AWS::Oam::Link。	

安裝個別帳戶的連結模組。	匯出個別帳戶的登入資料，並安裝 Observability Access Manager 連結模組：
	
	`Terraform Plan`
	`Terraform Apply`
	
	您可以為每個帳戶個別設定連結模組，或使用 AFT 在大量帳戶中自動安裝此模組。
		AWS DevOps、雲端管理員、雲端架構師

核准sink-to-link連線

任務	Description	所需的技能
檢查狀態訊息。	1. 登入至監控帳戶。 2. 開啟 CloudWatch 主控台。 3. 在左側的導覽窗格中，選擇設定。 在右側，您應該會看到狀態訊息 監控帳戶已啟用綠色核取記號。這表示監控帳戶具有可觀測性 Access Manager 接收	

任務	Description	所需的技能
	器，而其他帳戶的連結將連接到該接收器。	
核准link-to-sink連線。	1. 選擇狀態訊息下方的資源以連結帳戶選項。資訊確認這是監控帳戶，列出從租戶來源帳戶 (日誌、指標、追蹤) 共用的資料，並將帳戶標籤顯示為 $AccountName。 此畫面提供將租戶帳戶連結至監控帳戶的兩個選項：組織層級核准或帳戶層級核准。對於每個選項，您可以選擇下載 CloudFormation 範本以進行核准，或個別核准每個帳戶。 2. 為了簡化，請選擇每個帳戶層級要核准的任何帳戶。此選項提供帳戶的核准連結。 3. 選擇複製 URL 以複製連結。 4. 登入每個來源帳戶。 5. 在瀏覽器視窗中貼上連結，然後選擇核准連結連線至目的地。 6. 針對其他來源帳戶重複此步驟。 如需詳細資訊，請參閱 CloudWatch 文件中的將監控帳戶與來源帳戶連結。	AWS DevOps、雲端管理員、雲端架構師

驗證跨帳戶可觀測性資料

任務	Description	所需的技能
檢視跨帳戶資料。	1. 登入中央監控帳戶。 2. 開啟 CloudWatch 主控台。 3. 在左側導覽窗格中，選擇選項以檢視跨帳戶日誌、指標和追蹤。	AWS DevOps、雲端管理員、雲端架構師

（選用）啟用來源帳戶以信任監控帳戶

任務	Description	所需的技能
檢視來自其他帳戶的指標、儀表板、日誌、小工具和警示。	作為其他功能， 您可以與其他帳戶共用 CloudWatch 指標、儀表板、日誌、小工具和警示。每個帳戶使用稱為 CloudWatch-CrossAccountSharingRole 的 IAM 角色來存取此資料。 與中央監控帳戶具有信任關係的來源帳戶可以擔任此角色，並檢視來自監控帳戶的資料。 CloudWatch 提供範例 CloudFormation 指令碼來建立角色。選擇在 IAM 中管理角色d 會在您要檢視資料的帳戶中執行此指令碼。 `{ "Version": "2012-10-17", "Statement": [ {`	AWS DevOps、雲端管理員、雲端架構師

任務	Description	所需的技能
	"Effect": "Allow", "Principal": { "AWS": ["arn:aws:iam::XXXXXXXXXX:root", "arn:aws:iam::XXXXXXXXXX:root", "arn:aws:iam::XXXXXXXXXX:root", "arn:aws:iam::XXXXXXXXXX:root"] }, "Action": "sts:AssumeRole" }] } 如需詳細資訊，請參閱 CloudWatch 文件中的在 CloudWatch 中啟用跨帳戶功能。 CloudWatch	

（選用）從監控帳戶檢視跨帳戶跨區域

任務	Description	所需的技能
設定跨帳戶、跨區域存取。	在中央監控帳戶中，您可以選擇新增帳戶選擇器，以便在帳戶之間輕鬆切換並檢視其資料，而無需進行身分驗證。	AWS DevOps、雲端管理員、雲端架構師

任務	Description	所需的技能
	1. 登入中央監控帳戶。 2. 開啟 CloudWatch 主控台。 3. 在左側的導覽窗格中，選擇設定。 4. 在檢視跨帳戶跨區域區段中，選擇設定。 5. 選擇啟用，然後選取主控台中的顯示選取器核取方塊。 6. 選擇這些選項的其中之一： • 帳戶 ID 輸入：當您想要變更帳戶以檢視跨帳戶資料時，此選項會提示您手動輸入帳戶 ID。 • AWS Organization 帳戶選擇器：如果您已將 CloudWatch 與 整合 AWS Organizations，此選項會提供下拉式清單，其中包含組織中帳戶的完整清單。 • 自訂帳戶選取器：此選項可讓您手動輸入帳戶 IDs 清單以填入選取器。 7. 選擇儲存變更。 如需詳細資訊，請參閱 CloudWatch 文件中的跨帳戶跨區域 CloudWatch 主控台。 CloudWatch	

相關資源

- CloudWatch 跨帳戶可觀測性 (Amazon CloudWatch 文件）
- Amazon CloudWatch Observability Access Manager API 參考 (Amazon CloudWatch 文件）
- 資源：aws_oam_sink (Terraform 文件）
- 資料來源：aws_oam_link (Terraform 文件）
- CloudWatchObservabilityAccessManager (AWS Boto3 文件）

在啟動時檢查 EC2 執行個體是否有強制性標籤

Susanne Kangnoh 和 Archit Mathur，Amazon Web Services

總結

Amazon Elastic Compute Cloud (Amazon EC2) 在 Amazon Web Services (AWS) Cloud 提供可擴展的運算容量。使用 Amazon EC2 可減少前期所需的硬體投資，讓您更快速開發並部署應用程式。

您可以使用標記，以不同的方式分類您的 AWS 資源。當您帳戶中有許多資源，而且您想要根據標籤快速識別特定資源時，EC2 執行個體標記非常有用。您可以使用標籤將自訂中繼資料指派給 EC2 執行個體。標籤由使用者定義的索引鍵和值組成。我們建議您建立一組一致的標籤，以符合組織的需求。

此模式提供 AWS CloudFormation 範本，協助您監控特定標籤的 EC2 執行個體。範本會建立監控 AWS CloudTrail TagResource 或 UntagResource 事件的 Amazon CloudWatch Events 事件，以偵測新的 EC2 執行個體標記或標籤移除。 CloudTrail TagResource UntagResource 如果缺少預先定義的標籤，它會呼叫 AWS Lambda 函數，該函數會使用 Amazon Simple Notification Service (Amazon SNS) 將違規訊息傳送至您提供的電子郵件地址。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 用來上傳所提供 Lambda 程式碼的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。
- 您想要接收違規通知的電子郵件地址。

限制

- 此解決方案支援 CloudTrail TagResource 或 UntagResource 事件。它不會為任何其他事件建立通知。
- 此解決方案只會檢查標籤索引鍵。它不會監控索引鍵值。

Architecture

工作流程架構

自動化和擴展

- 您可以針對不同的 AWS 區域和帳戶多次使用 AWS CloudFormation 範本。您只需要在每個區域或帳戶中執行範本一次。

工具

AWS 服務

- Amazon EC2 – Amazon Elastic Compute Cloud (Amazon EC2) 是一種 Web 服務，可在雲端中提供安全、可調整大小的運算容量。它旨在讓開發人員更輕鬆地進行 Web 規模雲端運算。
- AWS CloudTrail – CloudTrail 是一種 AWS 服務，可協助您進行 AWS 帳戶的控管、合規以及操作和風險稽核。使用者、角色或 AWS 服務採取的動作會在 CloudTrail 中記錄為事件。
- Amazon CloudWatch Events – Amazon CloudWatch Events 提供近乎即時的系統事件串流，說明 AWS 資源的變更。CloudWatch Events 會在操作變更發生時得知並在必要時採取修正動作，方法是傳送訊息以回應環境、啟用 函數、進行變更，以及擷取狀態資訊。
- AWS Lambda – Lambda 是一種運算服務，支援執行程式碼，而不需要佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種高度可擴展的物件儲存服務，可用於各種儲存解決方案，包括網站、行動應用程式、備份和資料湖。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一種 Web 服務，可讓應用程式、最終使用者和裝置立即從雲端傳送和接收通知。

Code

此模式包含兩個檔案的附件：

- `index.zip` 是一種壓縮檔案，其中包含此模式的 Lambda 程式碼。
- `ec2-require-tags.yaml` 是部署 Lambda 程式碼的 CloudFormation 範本。

如需如何使用這些檔案的資訊，請參閱 Epics 一節。

史詩

部署 Lambda 程式碼

任務	Description	所需的技能
將程式碼上傳至 S3 儲存貯體。	建立新的 S3 儲存貯體或使用現有的 S3 儲存貯體上傳連接`index.zip` 的檔案 (Lambda 程式碼）。此儲存貯體必須與您要監控的資源 (EC2 執行個體） 位於相同的 AWS 區域。	雲端架構師
部署 CloudFormation 範本。	在與 S3 儲存貯體相同的 AWS 區域中開啟 Cloudformation 主控台，並部署附件中提供`ec2-require-tags.yaml` 的檔案。在下一個史詩中，提供範本參數的值。	雲端架構師

完成 CloudFormation 範本中的參數

任務	Description	所需的技能
提供 S3 儲存貯體名稱。	輸入您在第一個特徵中建立或選取的 S3 儲存貯體名稱。此 S3 儲存貯體包含 Lambda 程式碼的 .zip 檔案，且必須與 CloudFormation 範本和您要監控的 EC2 執行個體位於相同的 AWS 區域。	雲端架構師
提供 S3 金鑰。	提供 Lambda 程式碼 .zip 檔案在 S3 儲存貯體中的位置，不帶正斜線 （例如 `index.zip`	雲端架構師

任務	Description	所需的技能
	或 `controls/index.zip`）。	
提供電子郵件地址。	提供您要接收違規通知的作用中電子郵件地址。	雲端架構師
定義記錄層級。	指定記錄層級和詳細程度。會`Info`指定應用程式進度的詳細資訊性訊息，且應僅用於偵錯。 會`Error`指定仍然可以允許應用程式繼續執行的錯誤事件。 會`Warning`指定可能有害的情況。	雲端架構師
輸入所需的標籤索引鍵。	輸入您要檢查的標籤索引鍵。如果您想要指定多個金鑰，請以逗號分隔，不含空格。（例如，`ApplicationId,CreatedBy,Environment,Organization` 會搜尋四個金鑰。）CloudWatch Events 事件會搜尋這些標籤索引鍵，並在找不到它們時傳送通知。	雲端架構師

確認訂閱

任務	Description	所需的技能
確認電子郵件訂閱。	當 CloudFormation 範本成功部署時，它會傳送訂閱電子郵件訊息到您提供的電子郵件地址。若要接收通知，您必須確認此電子郵件訂閱。	雲端架構師

相關資源

- 建立儲存貯體 (Amazon S3 文件）
- 上傳物件 (Amazon S3 文件）
- 標記您的 Amazon EC2 資源 (Amazon EC2 文件）
- 使用 AWS CloudTrail 建立在 AWS API 呼叫上觸發的 CloudWatch Events 規則 CloudTrail (Amazon CloudWatch 文件）

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

在狀態檔案遺失後，安全地清除 AWS Account Factory for Terraform (AFT) 資源

Gokendra Malviya，Amazon Web Services

摘要

當您使用 AWS Account Factory for Terraform (AFT) 來管理 AWS Control Tower 環境時，AFT 會產生 Terraform 狀態檔案，以追蹤 Terraform 所建立資源的狀態和組態。遺失 Terraform 狀態檔案可能會對資源管理和清除造成重大挑戰。此模式提供系統性方法，可安全地識別和移除 AFT 相關資源，同時維護您 AWS Control Tower 環境的完整性。

此程序旨在確保適當移除所有 AFT 元件，即使沒有原始狀態檔案參考。此程序提供明確的路徑，可在您的環境中成功重新建立和重新設定 AFT，以協助確保對 AWS Control Tower 操作的干擾降到最低。

如需 AFT 的詳細資訊，請參閱 AWS Control Tower 文件。

先決條件和限制

先決條件

- 徹底了解 AFT 架構。
- 下列帳戶的管理員存取權：
  - AFT 管理帳戶
  - AWS Control Tower 管理帳戶
  - Log Archive 帳戶
  - 稽核帳戶
- 驗證沒有服務控制政策 SCPs) 包含會封鎖刪除 AFT 相關資源的限制。

限制

- 此程序可以有效地清除資源，但無法復原遺失的狀態檔案，有些資源可能需要手動識別。
- 清除程序的持續時間取決於您環境的複雜性，可能需要數小時的時間。
- 此模式已使用 AFT 1.12.2 版進行測試，並刪除下列資源。如果您使用的是不同版本的 AFT，您可能需要刪除其他資源。

服務名稱	資源數量

AWS CodeBuild	6
AWS CodeCommit	4
AWS CodePipeline	4
Amazon DynamoDB	5
Amazon Elastic Compute Cloud (Amazon EC2)	16
Amazon EventBridge	4
AWS Identity and Access Management (IAM) 角色	40
AWS Key Management Service (AWS KMS)	2
AWS Lambda	17
Amazon Simple Storage Service (Amazon S3)	2
Amazon Simple Notification Service (Amazon SNS)	2
Amazon Simple Queue Service (Amazon SQS)	2
AWS Systems Manager	62
AWS Step Functions	4

Important

無法復原此模式中步驟刪除的資源。遵循這些步驟之前，請仔細驗證資源名稱，並確認它們是由 AFT 建立。

Architecture

下圖顯示 AFT 元件和高階工作流程。AFT 會設定 Terraform 管道，協助您在其中佈建和自訂帳戶 AWS Control Tower。AFT 遵循 GitOps 模型，以自動化其中帳戶佈建的程序 AWS Control Tower。您可以為帳戶請求建立 Terraform 檔案並將其遞交至儲存庫，該儲存庫提供觸發帳戶佈建之 AFT 工作流程的輸入。帳戶佈建完成後，AFT 可以自動執行其他自訂步驟。

在此架構中：

- AWS Control Tower 管理帳戶是 服務 AWS 帳戶 專用的 AWS Control Tower 。這通常也稱為AWS 付款人帳戶或AWS Organizations 管理帳戶。
- AFT 管理帳戶是專用於 AFT 管理操作 AWS 帳戶 的 。這與您組織的管理帳戶不同。
- 已取代的帳戶是 AWS 帳戶 ，其中包含您選取的所有基準元件和控制項。AFT 使用 AWS Control Tower 來提供新帳戶。

如需此架構的詳細資訊，請參閱 AWS Control Tower 研討會中的 AFT 簡介。

工具

AWS 服務

- AWS Control Tower 可協助您設定和管理 AWS 多帳戶環境，並遵循規範最佳實務。
- AWS Account Factory for Terraform (AFT) 會設定 Terraform 管道，以協助您佈建和自訂其中的帳戶和資源 AWS Control Tower。
- AWS Organizations 當您擴展和擴展 AWS 資源時， 可協助您集中管理和控管您的環境。使用 Organizations，您可以建立帳戶並配置資源、分組帳戶來組織工作流程、套用控管政策，以及使用所有帳戶的單一付款方式來簡化帳單。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。此模式需要 IAM 角色和許可。

其他工具

- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。

最佳實務

- 如需 AWS Control Tower，請參閱 AWS Control Tower 文件中的AWS Control Tower 管理員最佳實務。
- 對於 IAM，請參閱 IAM 文件中的安全最佳實務。

史詩

刪除 AFT Management 帳戶中的 AFT 資源

任務	Description	所需的技能
刪除由 AFT 標籤識別的資源。	1. 使用管理員許可登入 AFT 管理帳戶。 2. 開啟 AWS Resource Groups 主控台。 3. 選取 AWS Control Tower 已部署 的區域。 4. 在導覽窗格中，選擇標籤編輯器。 5. 針對資源類型，選擇所有支援的資源類型。 6. 對於標籤，輸入 managed_by 作為標籤索引鍵，輸入 AFT 作為標籤值。 7. 選擇搜尋資源。 此搜尋會顯示 AFT 建立的所有資源。 8. 識別資源名稱，並使用對應的服務主控台將其刪除。例如，若要刪除參數存放區資源： a. 開啟 AWS Systems Manager 主控台。	AWS 管理員、AWS DevOps、DevOps 工程師

任務	Description	所需的技能
	b. 在導覽窗格中，選擇 Parameter Store (參數存放區)。 c. 在搜尋方塊中，按一下以顯示下拉式清單，選擇名稱，選擇等於，然後輸入 /aft。 d. 分 10 個批次刪除參數。（這是您可以同時刪除的最大數量。） 對於 AFT 1.12.2 版，大約會有 62 個參數存放區資源要刪除。所有參數名稱將以 /aft 開頭。 不過，並非所有資源都可以由 識別 AWS Resource Groups。在下列步驟中，您會找到並刪除剩餘的資源。	
刪除 IAM 角色。	1. 使用管理員許可登入 AFT 管理帳戶。 2. 開啟 IAM 主控台。 3. 依列出的順序刪除這些角色（順序很重要，因為相依性）： • `aft-*` • `AWSAFTAdmin` • `AWSAFTExecution` • `AWSAFTService` • `codebuild_trigger_role`	AWS 管理員、AWS DevOps、DevOps 工程師

任務	Description	所需的技能
刪除 AWS Backup 備份保存庫。	1. 開啟 AWS Backup 主控台。 2. 找到名為 的備份保存庫`aws_backup_vault` 。 3. 確認保存庫不包含任何作用中備份。 4. 刪除 `aws_backup_vault` 。	AWS 管理員、AWS DevOps、DevOps 工程師
刪除 Amazon CloudWatch 資源。	1. 開啟 CloudWatch 主控台。 2. 依列出的順序刪除下列資源： a. 事件匯流排：刪除 `aws_cloudwatch_event_bus` 。 b. 日誌：搜尋字首 AFT 並刪除所有相關日誌群組。 c. 查詢定義：刪除下列查詢： • `Customization Logs by Account ID` • `Customization Logs by Customization Request ID`	AWS 管理員、AWS DevOps、DevOps 工程師
刪除 AWS KMS 資源。	1. 切換到次要區域，做為 AFT 本身狀態的狀態追蹤後端。 2. 開啟 AWS KMS 主控台。 3. 刪除名為 AFT 的別名。	AWS 管理員、AWS DevOps、DevOps 工程師

刪除 Log Archive 帳戶中的 AFT 資源

任務	Description	所需的技能
刪除 S3 儲存貯體。	1. 使用管理員許可登入 Log Archive 帳戶。 2. 開啟 Amazon S3 主控台。 3. 清空下列儲存貯體： • `aws-aft-logs-471112509802-us-east-1` • `aws-aft-s3-access-logs-471112509802-us-east-1` (`111122223333` 以您的帳戶 ID 取代。) 4. 刪除兩個儲存貯體。	AWS 管理員、AWS DevOps、DevOps 工程師
刪除 IAM 角色。	1. 開啟 IAM 主控台。 2. 確認下列角色未由任何作用中服務使用： • `AWSAFTService` • `AWSAFTExecution` 3. 刪除這兩個角色。	AWS 管理員、AWS DevOps、DevOps 工程師

刪除稽核帳戶中的 AFT 資源

任務	Description	所需的技能
刪除 IAM 角色。	1. 使用管理員許可登入稽核帳戶。 2. 開啟 IAM 主控台。	AWS 管理員、AWS DevOps、DevOps 工程師

任務	Description	所需的技能
	3. 確認下列角色未由任何作用中服務使用： • `AWSAFTService` • `AWSAFTExecution` 4. 刪除這兩個角色。	

刪除 AWS Control Tower 管理帳戶中的 AFT 資源

任務	Description	所需的技能
刪除 IAM 角色。	1. 使用管理員許可登入 AWS Control Tower 管理帳戶。 2. 開啟 IAM 主控台。 3. 確認下列角色未由任何作用中服務使用： • `AWSAFTService` • `AWSAFTExecution` • `aft-control-tower-events-rule` 4. 刪除三個角色。	AWS 管理員、AWS DevOps、DevOps 工程師
刪除 EventBridge 規則。	1. 開啟 Amazon EventBridge 主控台。 2. 在左側導覽窗格中，選擇 Rules (規則)。 3. 尋找並選取名為 的規則`aft-capture-ct-events` 。 4. 選擇刪除，並在出現提示時確認刪除。	AWS 管理員、AWS DevOps、DevOps 工程師

疑難排解

問題	解決方案
分離網際網路閘道失敗。	當您刪除 AFT 標籤識別的資源時，如果您在分離或刪除網際網路閘道時遇到此問題，您必須先刪除 VPC 端點： 1. 登入 AFT 管理帳戶，然後開啟 Amazon VPC 主控台。 2. 在導覽窗格中，於依 VPC 篩選清單中，選擇名為 aft-management-vpc 的 VPC。 3. 在導覽窗格中選擇端點。 4. 選取與 VPC aft-management-vpc 相關聯的端點。 • 在刪除之前再次檢查 VPC ID 資料欄，以避免移除錯誤的端點。 • 請小心僅刪除與 AFT VPC 相關聯的端點。 5. 選擇 Actions (動作)、Delete VPC endpoints (刪除 VPC 端點)。 6. 在確認對話方塊中，輸入 Delete，然後選擇 Delete。 7. 等待端點狀態變更為已刪除。 刪除可能需要幾分鐘的時間才能完成。
找不到指定的 CloudWatch 查詢。	如果您找不到 AFT 建立的 CloudWatch 查詢，請依照下列步驟執行： 1. 登入 AFT 管理帳戶，然後開啟 CloudWatch 主控台。 2. 在導覽窗格中的日誌下，選擇日誌洞見。 3. 在右上角，選擇已儲存和範例查詢圖示。 您現在應該可以看到 AFT 查詢。如需螢幕擷取畫面，請參閱其他資訊一節。

問題	解決方案
	4. 選取下列查詢，然後選擇動作、刪除以移除它們。 • `Customization Logs by Account ID` • `Customization Logs by Customization Request ID`

相關資源

- AFT：
  - GitHub 儲存庫
  - 研討會
  - 文件
- AWS Control Tower 文件

其他資訊

若要在 CloudWatch Logs Insights 儀表板上檢視 AFT 查詢，請從右上角選擇已儲存和範例查詢圖示，如下列螢幕擷取畫面所示：

在不支援 AWS CodePipeline 的 AWS 區域中建立管道

Anand Krishna Varanasi，Amazon Web Services

摘要

注意： AWS CodeCommit 不再提供給新客戶。的現有客戶 AWS CodeCommit 可以繼續正常使用服務。進一步了解

AWS CodePipeline 是一種持續交付 (CD) 協調服務，屬於 Amazon Web Services (AWS) 的一組 DevOps 工具。它與各種來源（例如版本控制系統和儲存解決方案）、來自 AWS 和 AWS 合作夥伴的持續整合 (CI) 產品和服務，以及開放原始碼產品整合，以提供end-to-end工作流程服務，以進行快速的應用程式和基礎設施部署。

不過，並非所有 AWS 區域都支援 CodePipeline，而且具有連接 AWS CI/CD 服務的隱形協調器很有用。此模式說明如何使用 AWS CodeCommit、AWS CodeBuild 和 AWS CodeDeploy 等 AWS CI/CD 服務，在尚未支援 CodePipeline 的 AWS 區域中實作end-to-end工作流程管道。 AWS CodeDeploy

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- AWS 雲端開發套件 (AWS CDK) CLI 2.28 版或更新版本

Architecture

目標技術堆疊

下圖顯示在不支援 CodePipeline 的區域中建立的管道，例如非洲（開普敦）區域。開發人員會將 CodeDeploy 組態檔案（也稱為部署生命週期掛鉤指令碼) 推送至 CodeCommit 託管的 Git 儲存庫。（請參閱此模式隨附的 GitHub 儲存庫。) Amazon EventBridge 規則會自動啟動 CodeBuild。

CodeDeploy 組態檔案會擷取自 CodeCommit，做為管道來源階段的一部分，並傳輸至 CodeBuild。

在下一個階段，CodeBuild 會執行這些任務：

1. 下載應用程式原始碼 TAR 檔案。您可以使用 AWS Systems Manager 的 Parameter Store 功能來設定此檔案名稱。
2. 下載 CodeDeploy 組態檔案。

3. 建立特定於應用程式類型的應用程式原始碼和 CodeDeploy 組態檔案的合併封存。
4. 使用合併封存啟動 CodeDeploy 部署至 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。

工具

AWS 服務

- AWS CodeBuild 是一項全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。
- AWS CodeDeploy 會自動部署到 Amazon EC2 或內部部署執行個體、AWS Lambda 函數或 Amazon Elastic Container Service (Amazon ECS) 服務。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- AWS 雲端開發套件 (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。

Code

此模式的程式碼可在 GitHub CodePipeline 不支援的區域儲存庫中使用。

史詩

設定您的開發人員工作站

任務	Description	所需的技能
安裝 AWS CDK CLI。	如需說明，請參閱 AWS CDK 文件。	AWS DevOps
安裝 Git 用戶端。	若要建立遞交，您可以使用安裝在本機電腦上的 Git 用戶端，然後將遞交推送到 CodeCommit 儲存庫。若要使	AWS DevOps

任務	Description	所需的技能
	用 Git 用戶端設定 CodeCommit，請參閱 CodeCommit 文件。	
安裝 npm。	安裝 npm 套件管理員。如需詳細資訊，請參閱 npm 文件。	AWS DevOps

設定管道

任務	Description	所需的技能
複製程式碼儲存庫。	執行下列命令，將 GitHub CodePipeline 不支援的區域儲存庫複製到本機電腦。 `git clone https://github.com/aws-samples/invisible-codepipeline-unsupported-regions`	DevOps 工程師
在 cdk.json 中設定參數。	開啟 `cdk.json` 檔案，並提供下列參數的值： `"pipeline_account":"XXXXXXXXXXXX", "pipeline_region":"us-west-2", "repo_name": "app-dev-repo", "ec2_tag_key": "test-vm", "configName" : "cbdeployconfig", "deploymentGroupName": "cbdeploygroup", "applicationName" : "cbdeployapplication",`	AWS DevOps

任務	Description	所需的技能

	<pre>"projectName" :
	"CodeBuildProject"</pre>
	其中：
	• pipeline_account 是將建置管道的 AWS 帳戶。
	• pipeline_region 是將建置管道的 AWS 區域。
	• repo_name 是 CodeCommit 儲存庫的名稱。
	• ec2_tag_key 是連接至您要部署程式碼之 EC2 執行個體的標籤。
	• configName 是 CodeDeploy 組態檔案的名稱。
	• deploymentGroupName 是 CodeDeploy 部署群組的名稱。
	• applicationName 是 CodeDeploy 應用程式名稱。
	• projectName 是 CodeBuild 專案名稱。	

任務	Description	所需的技能

設定 AWS CDK 建構程式庫。	在複製的 GitHub 儲存庫中，使用下列命令來安裝 AWS CDK 建構程式庫、建置您的應用程式，以及合成以產生應用程式的 AWS CloudFormation 範本。
	`npm i aws-cdk-lib`
	`npm run build`
	`cdk synth`
		AWS DevOps

部署範例 AWS CDK 應用程式。	在不支援的區域（例如）中執行下列命令來部署程式碼`af-south-1` 。 `cdk deploy`	AWS DevOps

設定 CodeDeploy 的 CodeCommit CodeCommit 儲存庫

任務	Description	所需的技能
設定應用程式的 CI/CD。	複製您在 `cdk.json` 檔案中指定的 CodeCommit 儲存庫 (`app-dev-repo` 預設稱為)，以設定應用程式的 CI/CD 管道。 `git clone https://g it-codecommit.us-w est-2.amazonaws.com/ v1/repos/app-dev-repo`	AWS DevOps

任務	Description	所需的技能
	其中儲存庫名稱和區域取決於您在 `cdk.json` 檔案中提供的值。	

測試管道

任務	Description	所需的技能
使用部署指示測試管道。	GitHub CodePipeline 不支援區域儲存庫的`CodeDeploy_Files` 資料夾包含範例檔案，指示 CodeDeploy 部署應用程式。`appspec.yml` 檔案是 CodeDeploy 組態檔案，其中包含控制應用程式部署流程的掛鉤。您可以使用範例檔案 `index.html` 、`stop_server.sh` 、`start_server.sh` 和 `install_dependencies.sh` 來更新託管在 Apache 上的網站。這些是範例 - 您可以使用 GitHub 儲存庫中的程式碼來部署任何類型的應用程式。將檔案推送至 CodeCommit 儲存庫時，系統會自動啟動隱藏管道。如需部署結果，請檢查 CodeBuild 和 CodeDeploy 主控台中個別階段的結果。	AWS DevOps

相關資源

- 入門 (AWS CDK 文件)

- 雲端開發套件 (CDK) 簡介 (AWS Workshop Studio)
- AWS CDK 研討會

使用 AWS CDK 層面和逃生艙自訂預設角色名稱

SANDEEP SINGH 和 Amazon Web Services 的 James Jacob

總結

此模式示範如何自訂 AWS Cloud Development Kit (AWS CDK) 建構模組所建立角色的預設名稱。如果您的組織根據命名慣例有特定限制，通常需要自訂角色名稱。例如，您的組織可能會設定需要角色名稱中特定字首的 AWS Identity and Access Management (IAM) 許可界限或服務控制政策 (SCPs)。在這種情況下， AWS CDK 建構產生的預設角色名稱可能不符合這些慣例，而且可能需要修改。此模式透過使用 中的逃生艙和層面來解決這些需求 AWS CDK。您可以使用逃生艙來定義自訂角色名稱，以及將自訂名稱套用至所有角色的層面，以確保遵守組織的政策和限制條件。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- AWS CDK 文件中指定的先決條件

限制

- Aspects 根據資源類型篩選資源，因此所有角色都共用相同的字首。如果您需要不同角色的不同角色字首，則需要根據其他屬性進行其他篩選。例如，若要將不同的字首指派給與 AWS Lambda 函數相關聯的角色，您可以依特定角色屬性或標籤進行篩選，並為 Lambda 相關角色套用一個字首，為其他角色套用不同的字首。
- IAM 角色名稱的長度上限為 64 個字元，因此修改後的角色名稱必須修剪才能符合此限制。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

Architecture

目標技術堆疊

- AWS CDK
- AWS CloudFormation

目標架構

- AWS CDK 應用程式包含一或多個堆疊，這些 CloudFormation 堆疊會經過合成並部署以管理 AWS 資源。
- 若要修改未由第 2 層 (L2) 建構公開的 AWS CDK受管資源屬性，您可以使用逃生艙覆寫基礎 CloudFormation 屬性（在此案例中為角色名稱），以及在 AWS CDK 堆疊合成程序期間將角色套用至 AWS CDK 應用程式中的所有資源。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端 基礎設施。
- AWS CDK Command Line Interface (AWS CDK CLI)（也稱為 AWS CDK Toolkit) 是一種命令列雲端開發套件，可協助您與 AWS CDK 應用程式互動。CLI `cdk`命令是與您的 AWS CDK 應用程式互動的主要工具。它會執行您的應用程式、查詢您定義的應用程式模型，以及產生和部署由 產生的 CloudFormation 範本 AWS CDK。
- CloudFormation 可協助您設定 AWS 資源、快速一致地佈建資源，以及在整個 AWS 帳戶 和 區域的生命週期中管理資源。

程式碼儲存庫

此模式的原始程式碼和範本可在 GitHub CDK Aspects 覆寫儲存庫中使用。

最佳實務

請參閱 AWS 規範指引網站上的使用 TypeScript AWS CDK 中的 建立 IaC 專案的最佳實務。

史詩

安裝 AWS CDK CLI

任務	Description	所需的技能
安裝 AWS CDK CLI。	若要全域安裝 AWS CDK CLI，請執行 命令：	AWS DevOps

任務	Description	所需的技能
	`npm install -g aws-cdk`	
驗證版本。	執行 命令： `cdk --version` 確認您使用的是 CLI 第 2 AWS CDK 版。	AWS DevOps
引導 AWS CDK 環境。	部署 CloudFormation 範本之前，請先準備 AWS 區域 您要使用的 帳戶和 。執行 命令： `cdk bootstrap <account> /<Region>` 如需詳細資訊，請參閱 AWS 文件中的AWS CDK 引導。	AWS DevOps

部署 AWS CDK 應用程式以示範如何使用 面向

任務	Description	所需的技能
設定專案。	1. 將此模式的 GitHub 儲存庫複製到本機電腦： `git clone https://github.com/aws-samples/cdk-aspects-override` 2. 導覽至本機電腦上的專案目錄。 3. 安裝專案相依性：	AWS DevOps

任務	Description	所需的技能
	`npm ci`	
部署具有 指派之預設角色名稱的堆疊 AWS CDK。	部署兩個包含 Lambda 函數及其相關角色的 CloudFormation 堆疊 (`ExampleStack1` 和 `ExampleStack2`)： `npm run deploy:ExampleAppWithoutAspects` 程式碼不會明確傳遞角色屬性，因此角色名稱將由 建構 AWS CDK。 如需輸出範例，請參閱其他資訊一節。	AWS DevOps

任務	Description	所需的技能

使用 層面部署堆疊。	在此步驟中，您會將字首新增至 AWS CDK 專案中部署的所有 IAM 角色，以套用強制執行角色名稱慣例的 面向。在 `lib/aspects.ts` 檔案中定義 面向。方面使用逃生艙，透過新增字首來覆寫角色名稱。面向會套用至 `bin/app-with-aspects.ts` 檔案中的堆疊。此範例中使用的角色名稱字首為 `dev-unicorn` 。
	1. 編輯 `bin/app-with-aspects.ts` 檔案。
	2. 在 檔案中，更新字首`ROLE_NAME_PREFIX` 為的變數`dev-unicorn` ：
	`const app = new cdk.App();`
	`// Define a prefix for the role names`
	`const ROLE_NAME_PREFIX = 'dev-unicorn';`
	`// Instantiate the RoleNamingConventionAspect with the desired prefix`
	`const roleNamingConventionAspect = new RoleNamingConventionAspect(ROLE_NAME_PREFIX);`
		AWS DevOps

任務	Description	所需的技能
	3. 使用層面部署 AWS CDK 應用程式： `npm run deploy:ExampleAppWithAspects` 如需輸出範例，請參閱其他資訊一節。	

清除資源

任務	Description	所需的技能
刪除您的 AWS CloudFormation 堆疊。	使用此模式後，請執行下列命令來清除資源，以避免產生額外費用： `cdk destroy --all -f && cdk --app npx ts-node bin/app-with-aspects.ts' destroy --all -f`	AWS DevOps

疑難排解

問題	解決方案
您使用 時遇到問題 AWS CDK。	請參閱 AWS CDK 文件中的疑難排解 AWS CDK 常見問題。

相關資源

- AWS Cloud Development Kit (AWS CDK)

- AWS CDK 文件
- AWS CDK 在 GitHub 上
- 逃生艙
- Aspects 和 AWS CDK

其他資訊

由 CloudFormation 在沒有 層面的情況下建立的角色名稱

```
Outputs:
ExampleStack1WithoutAspects.Function1RoleName = example-stack1-without-as-
Function1LambdaFunctionSe-y7FYTY6FXJXA
ExampleStack1WithoutAspects.Function2RoleName = example-stack1-without-as-
Function2LambdaFunctionSe-dDZV4rkWqWnI
...

Outputs:
ExampleStack2WithoutAspects.Function3RoleName = example-stack2-without-as-
Function3LambdaFunctionSe-ygMv49iTyMq0
```

CloudFormation 使用 面向建立的角色名稱

```
Outputs:
ExampleStack1WithAspects.Function1RoleName = dev-unicorn-
Function1LambdaFunctionServiceRole783660DC
ExampleStack1WithAspects.Function2RoleName = dev-unicorn-
Function2LambdaFunctionServiceRole2C391181
...

Outputs:
ExampleStack2WithAspects.Function3RoleName = dev-unicorn-
Function3LambdaFunctionServiceRole4CAA721C
```

使用私有靜態 IPs 在 Amazon EC2 上部署 Cassandra 叢集，以避免重新平衡

Dipin Jain，Amazon Web Services

總結

Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的私有 IP 會在整個生命週期中保留。不過，私有 IP 可能會在計劃或非計劃的系統當機期間變更；例如，在 Amazon Machine Image (AMI) 升級期間。在某些情況下，保留私有靜態 IP 可以增強工作負載的效能和復原時間。例如，針對 Apache Cassandra 種子節點使用靜態 IP 可防止叢集產生重新平衡的額外負荷。

此模式說明如何將次要彈性網路界面連接至 EC2 執行個體，以在重新託管期間保持 IP 靜態。模式著重於 Cassandra 叢集，但您可以將此實作用於受益於私有靜態 IPs 的任何架構。

先決條件和限制

先決條件

- 作用中的 Amazon Web Service (AWS) 帳戶

產品版本

- DataStax 5.11.1 版
- 作業系統：Ubuntu 16.04.6 LTS

Architecture

來源架構

來源可以是現場部署虛擬機器 (VM) 或 AWS 雲端 EC2 執行個體上的 Cassandra 叢集。下圖說明第二個案例。此範例包含四個叢集節點：三個種子節點和一個管理節點。在來源架構中，每個節點都連接了單一網路介面。

目標架構

目的地叢集託管在 EC2 執行個體上，並將次要彈性網路界面連接到每個節點，如下圖所示。

自動化和擴展

您也可以自動將第二個彈性網路界面連接至 EC2 Auto Scaling 群組，如 AWS 知識中心影片中所述。

史詩

在 Amazon EC2 上設定 Cassandra 叢集

任務	Description	所需的技能
啟動 EC2 節點以託管 Cassandra 叢集。	在 Amazon EC2 主控台上，為 AWS 帳戶中的 Ubuntu 節點啟動四個 EC2 執行個體。Cassandra 叢集使用三個（種子）節點，而第四個節點充當叢集管理節點，您將在其中安裝 DataStax Enterprise (DSE) OpsCenter。如需說明，請參閱 Amazon EC2 文件。	雲端工程師
確認節點通訊。	請確定四個節點可以透過資料庫和叢集管理連接埠彼此通訊。	網路工程師
在管理節點上安裝 DSE OpsCenter。	從管理節點上的 Debian 套件安裝 DSE OpsCenter 6.1。如需說明，請參閱DataStax 文件。	DBA
建立次要網路界面。	Cassandra 會根據該節點的 EC2 執行個體 IP 地址，為每個節點產生通用唯一識別碼 (UUID)。此 UUID 用於在環上分佈虛擬節點 (vnodes)。在 EC2 執行個體上部署 Cassandra 時，IP 地址會在建立時自動指派給執行個體。 如果發生計劃或非計劃中斷，新 EC2 執行個體的 IP 地址會	雲端工程師

任務	Description	所需的技能
	變更、資料分佈會變更，而且必須重新平衡整個環。這不理想。若要保留指派的 IP 地址，請使用具有固定 IP 地址的次要彈性網路介面。 1. 在 Amazon EC2 主控台上，選擇網路界面、建立網路界面。 2. 針對子網路，選取您在其中建立 EC2 執行個體的子網路。 3. 針對私有 IPv4 地址，選擇自動指派。 4. 針對安全群組，選取安全群組，然後選擇建立網路介面。 如需建立網路介面的詳細資訊，請參閱 Amazon EC2 文件。	

任務	Description	所需的技能
將次要網路介面連接至叢集節點。	1. 在 Amazon EC2 主控台上，選擇執行個體。 2. 選取您先前建立之 EC2 執行個體的核取方塊。 3. 選擇 Actions (動作)、Networking (網路)、Attach network interface (連接網路介面)。 4. 選取您在上一個步驟中建立的網路界面，然後選擇連接。 如需連接網路介面的詳細資訊，請參閱 Amazon EC2 文件。	雲端工程師
在 Amazon EC2 中新增路由以處理非對稱路由。	當您連接第二個網路界面時，網路很可能會執行非對稱路由。若要避免這種情況，您可以為新的網路介面新增路由。 如需非對稱路由的深入說明和修復，請參閱 AWS 知識中心影片或多首頁伺服器上的克服非對稱路由（文章位於 Linux 日誌中，作者為 Patrick McManus，2004 年 4 月 5 日）。	網路工程師
更新 DNS 項目以指向次要網路介面 IP。	將節點的完整網域名稱 (FQDN) 指向次要網路介面的 IP。	網路工程師

任務	Description	所需的技能
使用 DSE OpsCenter 安裝和設定 Cassandra 叢集。	當叢集節點準備好次要網路介面時，您可以安裝和設定 Cassandra 叢集。	DBA

從節點失敗復原叢集

任務	Description	所需的技能
為叢集種子節點建立 AMI。	備份節點，以便在節點故障時，使用資料庫二進位檔還原它們。如需說明，請參閱 Amazon EC2 文件中的建立 AMI。	備份管理員
從節點失敗復原。	使用從 AMI 啟動的新 EC2 執行個體取代失敗的節點，並連接失敗節點的次要網路介面。	備份管理員
確認 Cassandra 叢集運作狀態良好。	當替代節點啟動時，請在 DSE OpsCenter 中驗證叢集運作狀態。	DBA

相關資源

- 從 Debian 套件安裝 DSE OpsCenter 6.1 (DataStax 文件）
- 如何讓次要網路界面在 Ubuntu EC2 執行個體中運作 (AWS 知識中心影片）
- 在 Amazon EC2 上執行 Apache Cassandra 的最佳實務 (AWS 部落格文章）

使用 AWS Transit Gateway Connect 將 VRFs 擴展至 AWS AWS Transit Gateway

Adam Till、Yashar Araghi、Vikas Dewangan 和 Mohideen HajaMohideen，Amazon Web Services

總結

虛擬路由和轉送 (VRF) 是傳統網路的一項功能。它使用隔離的邏輯路由網域，以路由表的形式分隔相同實體基礎設施內的網路流量。當您將內部部署網路連線至 AWS 時，您可以將 AWS Transit Gateway 設定為支援 VRF 隔離。此模式使用範例架構，將內部部署 VRFs 連接到不同的傳輸閘道路由表。

此模式使用 AWS Direct Connect 中的傳輸虛擬介面 (VIFs) 和傳輸閘道連線附件來擴展 VRFs。傳輸 VIF 用於存取與 Direct Connect 閘道相關聯的一或多個 Amazon VPC 傳輸閘道。傳輸閘道 Connect 連接會將傳輸閘道連接到在 VPC 中執行的第三方虛擬設備。傳輸閘道 Connect 連接支援一般路由封裝 (GRE) 通道通訊協定以實現高效能，並支援動態路由的邊界閘道通訊協定 (BGP)。

此模式中描述的方法具有下列優點：

- 使用 Transit Gateway Connect，您可以向 Transit Gateway Connect 對等公告最多 1，000 個路由，並從中接收最多 5，000 個路由。在沒有 Transit Gateway Connect 的情況下使用 Direct Connect 傳輸 VIF 功能，每個傳輸閘道限制為 20 個字首。
- 您可以維持流量隔離，並使用 Transit Gateway Connect 在 AWS 上提供託管服務，無論您的客戶使用的 IP 地址結構描述為何。
- VRF 流量不需要周遊公有虛擬介面。這可讓您更輕鬆地遵守許多組織中的合規和安全性要求。
- 每個 GRE 通道最多支援 5 Gbps，每個傳輸閘道 Connect 連接最多可有四個 GRE 通道。這比許多其他連線類型更快，例如支援高達 1.25 Gbps 的 AWS Site-to-Site VPN 連線。

先決條件和限制

先決條件

- 已建立必要的 AWS 帳戶 （ 如需詳細資訊，請參閱 架構 ）
- 在每個帳戶中擔任 AWS Identity and Access Management (IAM) 角色的許可。
- 每個帳戶中的 IAM 角色必須具有佈建 AWS Transit Gateway 和 AWS Direct Connect 資源的許可。如需詳細資訊，請參閱傳輸閘道的身分驗證和存取控制，以及請參閱 Direct Connect 的身分和存取管理。
- 已成功建立 Direct Connect 連線。如需詳細資訊，請參閱使用連線精靈建立連線。

限制

- 傳輸閘道連接到生產、QA 和開發帳戶中VPCs 有其限制。如需詳細資訊，請參閱傳輸閘道連接至 VPC。
- 建立與使用 Direct Connect 閘道均設有限制。如需詳細資訊，請參閱 AWS Direct Connect 配額。

Architecture

目標架構

下列範例架構提供可重複使用的解決方案，以使用 Transit Gateway Connect 連接部署傳輸 VIFs。此架構使用多個 Direct Connect 位置提供彈性。如需詳細資訊，請參閱 Direct Connect 文件中的最大彈性。內部部署網路具有延伸到 AWS 的生產、QA 和開發 VRFs，並使用專用路由表隔離。

在 AWS 環境中，兩個帳戶專用於擴展 VRFs：Direct Connect 帳戶和網路中樞帳戶。Direct Connect 帳戶包含每個路由器的連線和傳輸 VIFs。您可以從 Direct Connect 帳戶建立傳輸 VIFs，但將其部署到網路中樞帳戶，以便將其與網路中樞帳戶中的 Direct Connect 閘道建立關聯。網路中樞帳戶包含 Direct Connect 閘道和傳輸閘道。AWS 資源的連線方式如下：

1. 傳輸 VIFs會將 Direct Connect 位置中的路由器與 Direct Connect 帳戶中的 AWS Direct Connect 連接。
2. 傳輸 VIF 會將 Direct Connect 與網路中樞帳戶中的 Direct Connect 閘道連線。
3. 傳輸閘道關聯會將 Direct Connect 閘道與網路中樞帳戶中的傳輸閘道連線。
4. 傳輸閘道 Connect 連接會將傳輸閘道與生產、QA 和開發帳戶中VPCs 連線。

傳輸 VIF 架構

下圖顯示傳輸 VIFs組態詳細資訊。此範例架構使用通道來源的 VLAN，但您也可以使用迴路。

以下是傳輸 VIFs組態詳細資訊，例如自動系統號碼 (ASNs)。

資源	項目	詳細資訊
router-01	ASN	65534

router-02	ASN	65534
router-03	ASN	65534
router-04	ASN	65534
Direct Connect 閘道	ASN	64601
Transit Gateway	ASN	64600
	CIDR 區塊	10.100.254.0/24

傳輸閘道 Connect 架構

下圖和表格說明如何透過傳輸閘道 Connect 連接設定單一 VRF。對於其他 VRFs，請在 CIDR 區塊內指派唯一的通道 IDs、傳輸閘道 GRE IP 地址和 BGP。對等 GRE IP 地址符合來自傳輸 VIF 的路由器對等 IP 地址。

下表包含路由器組態詳細資訊。

路由器	通道	IP 位址	來源	目標
router-01	通道 1	169.254.101.17	VLAN 60	10.100.254.1
			169.254.100.1	
router-02	通道 11	169.254.101.81	VLAN 61	10.100.254.11
			169.254.100.5	
router-03	通道 21	169.254.101.145	VLAN 62	10.100.254.21
			169.254.100.9	
router-04	通道 31	169.254.101.209	VLAN 63	10.100.254.31
			169.254.100.13	

下表包含傳輸閘道組態詳細資訊。

通道	傳輸閘道 GRE IP 地址	對等 GRE IP 地址	CIDR 區塊內的 BGP
通道 1	10.100.254.1	VLAN 60 169.254.100.1	169.254.101.16/29
通道 11	10.100.254.11	VLAN 61 169.254.100.5	169.254.101.80/29
通道 21	10.100.254.21	VLAN 62 169.254.100.9	169.254.101.144/29
通道 31	10.100.254.31	VLAN 63 169.254.100.13	169.254.101.208/29

部署

Epics 區段說明如何在多個客戶路由器之間部署 單一 VRF 的範例組態。步驟 1-5 完成後，您可以針對要延伸到 AWS 的每個新 VRF，使用步驟 6-7 建立新的傳輸閘道 Connect 連接：

1. 建立傳輸閘道。
2. 為每個 VRF 建立 Transit Gateway 路由表。
3. 建立傳輸虛擬介面。
4. 建立 Direct Connect 閘道。
5. 使用允許的字首建立 Direct Connect 閘道虛擬介面和閘道關聯。
6. 建立傳輸閘道 Connect 連接。
7. 建立 Transit Gateway Connect 對等。
8. 將傳輸閘道 Connect 連接與路由表建立關聯。
9. 公告路由器的路由。

工具

AWS 服務

- AWS Direct Connect 透過標準乙太網路光纖纜線，將您的內部網路連結至 Direct Connect 位置。透過此連線，您可以直接建立與公有 AWS 服務的虛擬介面，同時略過網路路徑中的網際網路服務供應商。
- AWS Transit Gateway 是中央中樞，可連接虛擬私有雲端 (VPCs) 和內部部署網路。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路類似於您在自己的資料中心內操作的傳統網路，具有使用可擴展的 AWS 基礎設施的優勢。

史詩

規劃架構

任務	Description	所需的技能
建立自訂架構圖表。	1. 在附件區段中，下載圖表範本。 2. 在 Microsoft Office PowerPoint 中開啟連接的圖表。 3. 在架構概觀投影片上，為您的環境自訂架構圖。識別需要擴展到您的 AWS 環境的內部部署 VRFs。 4. 在傳輸 VIF 投影片上，自訂架構圖表。識別路由器、Direct Connect 閘道和傳輸閘道的 AS 編號。識別傳輸 VIF 每一端的 IP 地址。 5. 在 Transit Gateway Connect 投影片上，為每個 VRF 自訂架構圖。識別設定路由器和 Transit Gateway Connect 對等所需的所有必要 IP 地址。	雲端架構師、網路管理員

建立 Transit Gateway 資源

任務	Description	所需的技能
建立傳輸閘道。	1. 登入網路中樞帳戶。 2. 遵循建立傳輸閘道中的指示。此模式請注意下列事項： • 針對 Amazon 端自治系統編號 (ASN)，輸入唯一的 ASN。基於此範例的目的，ASN 為 64600。 • 選取 DNS 支援。 • 對於此範例架構，不需要 VPN ECMP 支援、預設路由表關聯、預設路由表探測和多點傳送支援。 • 針對傳輸閘道 CIDR 區塊，輸入傳輸閘道的 IPv4 CIDR 區塊。基於此範例的目的，CIDR 區塊為 `10.100.254.0/24` 。	網路管理員、雲端架構師
建立傳輸閘道路由表。	遵循建立傳輸閘道路由表中的指示。此模式請注意下列事項： • 針對名稱標籤，提供傳輸閘道路由表的名稱。建議使用對應至 VRF 的名稱，例如 `routetable-dev-vrf` 。 • 針對傳輸閘道 ID，選擇您先前建立的傳輸閘道。	雲端架構師、網路管理員

建立傳輸虛擬介面

任務	Description	所需的技能
建立傳輸虛擬介面。	1. 登入 Direct Connect 帳戶。 2. 遵循建立傳輸虛擬介面到 Direct Connect 閘道中的指示。此模式請注意下列事項： • 針對虛擬介面名稱，輸入傳輸 VIF 的名稱。建議使用對應至路由器的名稱，例如 `transit-vif-router01` 。 • 針對連線，選取路由器，例如 `router-01` 。 • 對於虛擬介面擁有者，輸入網路中樞帳戶的帳戶 ID。如需說明，請參閱檢視您的 AWS 帳戶 ID。 • 對於 Direct Connect 閘道，請勿進行任何選擇。您可以在後續步驟中連接 Direct Connect 閘道。 • 針對 VLAN，輸入路由器的 VLAN，例如 60。 • 針對 BGP ASN，輸入路由器的 ASN，例如 65534。 • 在 Additional settings (其他設定) 之下，執行下列動作： • 選擇 IPv4。 • 針對您的路由器對等 IP，輸入路由器	雲端架構師、網路管理員

任務	Description	所需的技能
	對等 IP 地址，例如 `169.254.100.1` 。 • 對於 Amazon 路由器對等 IP。輸入 Amazon 路由器對等 IP，例如 `169.254.100.2` 。 • 對於 BGP 身分驗證金鑰，需要密碼。如果保留空白，AWS 會建立只能在此帳戶中存取的金鑰。 3. 重複這些指示來建立 VIFs。	

建立 Direct Connect 資源

任務	Description	所需的技能
建立一個 Direct Connect 閘道。	1. 登入網路中樞帳戶。 2. 遵循建立 Direct Connect 閘道中的指示。此模式請注意下列事項： • 對於 Amazon 端 ASN，輸入 Direct Connect 閘道的 ASN，例如 64601。 • 請勿選擇虛擬私有閘道。	雲端架構師、網路管理員
將 Direct Connect 閘道連接至傳輸 VIFs。	1. 在網路中樞帳戶中，開啟位於 https：//https://console.aws.amazon.com/directconnect/v2/ 的 AWS Direct Connect 主控台。 2. 在導覽窗格中，選擇 Virtual Interfaces (虛擬介面)。	雲端架構師、網路管理員

任務	Description	所需的技能
	3. 選取新的傳輸 VIF，然後選擇接受。 4. 選擇您建立的 Direct Connect 閘道。 5. 為每個傳輸 VIF 重複這些指示。	
使用允許的字首建立 Direct Connect 閘道關聯。	在網路中樞帳戶中，遵循中的指示建立傳輸閘道的關聯。此模式請注意下列事項： • 針對閘道，選擇您先前建立的傳輸閘道。 • 針對允許的字首，輸入指派給傳輸閘道的 CIDR 區塊，例如 `10.100.254.0/24` 。 建立此關聯會自動建立具有 Direct Connect Gateway 資源類型的 Transit Gateway 附件。此附件不需要與傳輸閘道路由表相關聯。	雲端架構師、網路管理員

任務	Description	所需的技能
建立傳輸閘道 Connect 連接。	1. 在網路中樞帳戶中，開啟位於 https：//https://console.aws.amazon.com/vpc/ 的 Amazon VPC 主控台。 2. 在導覽窗格中，選擇傳輸閘道連接。 3. 選擇 Create transit gateway attachment (建立傳輸閘道連接)。 4. 在名稱標籤中，輸入附件的名稱。建議使用對應至 VRF 的名稱，例如 PROD-VRF。 5. 針對傳輸閘道 ID，選擇您先前建立的傳輸閘道。 6. 在 Attachment type (連接類型) 中，選擇 Connect (連線)。 7. 針對傳輸連接 ID，選擇您先前建立的 Direct Connect 閘道。 8. 選擇 Create transit gateway attachment (建立傳輸閘道連接)。 9. 針對您要延伸的每個 VRF 重複此步驟。	雲端架構師、網路管理員

任務	Description	所需的技能
建立 Transit Gateway Connect 對等。	1. 在網路中樞帳戶中，遵循建立 Transit Gateway Connect 對等 (GRE 通道) 中的指示。此模式請注意下列事項： • 在名稱標籤中，輸入 Transit Gateway Connect 對等的名稱。建議使用與路由器對應的名稱，例如 `connectpeer-router01` 。 • 對於傳輸閘道 GRE 地址，輸入傳輸閘道 CIDR 區塊中指派的 IP 地址，例如 `10.100.254.1` 。 • 針對對等 GRE 地址，輸入指派給在路由器上為傳輸 VIF 建立之 VLAN 的 IP 地址，例如 `169.254.100.1` 。如果 AWS 可以到達 IP 地址，您可以使用 VLAN 或 Loopback 等任何界面做為對等 GRE 地址。 • 對於 BGP 內部 CIDR 區塊 (IPv4)，輸入 CIDR 區塊 IP 地址內的 BGP，例如 `169.254.101.16/29` 。 • 針對對等 ASN，輸入路由器的 ASN，例如 65534。	

任務	Description	所需的技能
	2. 重複這些指示，為每個路由器建立 GRE 通道。	

向路由器公告路由

任務	Description	所需的技能
公告路由。	將新的 Transit Gateway Connect 連接與您先前為此 VRF 建立的路由表建立關聯。例如，將生產傳輸閘道 Connect 連接與`Production-VRF` 路由表建立關聯。 為公告至路由器的字首建立靜態路由。 1. 登入網路中樞帳戶。 2. 在 https://console.aws.amazon.com/vpc/ 開啟 Amazon VPC 主控台。 3. 在導覽窗格的傳輸閘道下，選擇傳輸閘道路由表。 4. 選取 `Production-VRF` 路由表。 5. 在動作功能表中，選擇建立靜態路由。 6. 針對 CIDR，輸入目標 VPC 中傳輸閘道連接之公告路由的 CIDR 區塊，例如 `10.100.1.0/24` 。 7. 針對選擇附件，選擇相關的傳輸閘道連線附件。	網路管理員、雲端架構師

任務	Description	所需的技能
	8. 選擇 Create static route (建立靜態路由)。	

相關資源

AWS 文件

- Direct Connect 文件
  - 使用 Direct Connect 閘道
  - 傳輸閘道關聯
  - AWS Direct Connect 虛擬介面
- Transit Gateway 文件
  - 使用傳輸閘道
  - 傳輸閘道連接至 Direct Connect 閘道
  - Transit Gateway Connect 連接和 Transit Gateway Connect 對等
  - 建立傳輸閘道 Connect 連接

AWS 部落格文章

- 使用 AWS Transit Gateway Connect 分割混合網路
- 使用 AWS Transit Gateway 連線來擴展 VRFs並增加 IP 字首公告

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

當 AWS KMS 金鑰的金鑰狀態變更時，取得 Amazon SNS 通知

Shubham Harsora、Aromal Raj Jayarajan 和 Navdeep Pareek，Amazon Web Services

總結

刪除該金鑰時，會遺失與 AWS Key Management Service (AWS KMS) 金鑰相關聯的資料和中繼資料。刪除是不可復原的，您無法復原遺失的資料（包括加密的資料）。您可以設定通知系統，提醒您 AWS KMS 金鑰的金鑰狀態變更，以防止資料遺失。

此模式說明如何使用 Amazon EventBridge 和 Amazon Simple Notification Service (Amazon SNS) 來監控 AWS KMS 金鑰的狀態變更，以便在 AWS KMS 金鑰的金鑰狀態變更為 `Disabled`或 時發出自動通知`PendingDeletion`。例如，如果使用者嘗試停用或刪除 AWS KMS 金鑰，您將收到一封電子郵件通知，其中包含嘗試狀態變更的詳細資訊。您也可以使用此模式來排程刪除 AWS KMS 金鑰。

先決條件和限制

先決條件

- 具有 AWS Identity and Access Management (IAM) 使用者的作用中 AWS 帳戶
- AWS KMS 金鑰

Architecture

技術堆疊

- Amazon EventBridge
- AWS Key Management Service (AWS KMS)
- Amazon Simple Notification Service (Amazon SNS)

目標架構

下圖顯示用於建置自動化監控和通知程序的架構，用於偵測 AWS KMS 金鑰狀態的任何變更。

該圖顯示以下工作流程：

1. 使用者停用或排程刪除 AWS KMS 金鑰。

2. EventBridge 規則會評估排程的 `Disabled`或 `PendingDeletion`事件。
3. EventBridge 規則會叫用 Amazon SNS 主題。
4. Amazon SNS 會傳送電子郵件通知訊息給使用者。

> **Note**
>
> 您可以自訂電子郵件訊息，以滿足組織的需求。我們建議您包含使用 AWS KMS 金鑰之實體的相關資訊。這可協助使用者了解刪除 AWS KMS 金鑰的影響。您也可以排定在刪除 AWS KMS 金鑰前一或兩天傳送的提醒電子郵件通知。

自動化和擴展

AWS CloudFormation 堆疊會部署所有必要的資源和服務，此模式才能運作。您可以在單一帳戶中獨立實作模式，或在 AWS Organizations 中使用多個獨立帳戶或組織單位的 AWS CloudFormation StackSets。 https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_ous.html AWS Organizations

工具

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在 AWS 帳戶和 AWS 區域的整個生命週期中管理這些資源。此模式的 CloudFormation 範本說明您想要的所有 AWS 資源，而 CloudFormation 會為您佈建和設定這些資源。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。EventBridge 會從您自己的應用程式和 AWS 服務提供即時資料串流，並將該資料路由到 AWS Lambda 等目標。EventBridge 可簡化建置事件驅動型架構的程序。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。

Code

此模式的程式碼可在 GitHub Monitor AWS KMS 金鑰停用和排程刪除儲存庫中使用。

史詩

部署 CloudFormation 範本

任務	Description	所需的技能
複製儲存庫。	執行下列命令，將 GitHub Monitor AWS KMS 金鑰停用並排程刪除儲存庫複製到本機電腦： `git clone https://github.com/aws-samples/aws-kms-deletion-notification`	AWS 管理員、雲端架構師
更新範本的參數。	在程式碼編輯器中，開啟您從儲存庫複製的 `Alerting-KMS-Events.yaml` CloudFormation 範本，然後更新下列參數： • 針對 `DestinationEmailAddress` ，輸入您計劃用於接收 SNS 通知的作用中電子郵件地址。 • 針對 `SNSTopicName` ，輸入 SNS 主題的名稱。	AWS 管理員、雲端架構師
部署 CloudFormation 範本。	1. 登入 AWS 管理主控台並開啟 CloudFormation 主控台。 2. 在導覽窗格中，選擇建立堆疊，然後選擇使用新資源（標準）。 3. 在識別資源頁面上，選擇下一步。	AWS 管理員、雲端架構師

任務	Description	所需的技能
	4. 在指定範本頁面上，針對範本來源， 選取上傳範本檔案。 5. 選擇選擇檔案，從複製的 GitHub 儲存庫中選取`Alerting-KMS-Events.yaml` 檔案，然後選擇下一步。 6. 針對堆疊名稱，輸入您的堆疊名稱。 7. 選擇提交。	

確認訂閱

任務	Description	所需的技能
確認訂閱電子郵件。	CloudFormation 範本成功部署後，Amazon SNS 會將訂閱確認訊息傳送至您在 CloudFormation 範本中提供的電子郵件地址。 若要接收通知，您必須確認此電子郵件訂閱。如需詳細資訊，請參閱《Amazon SNS 開發人員指南》中的確認訂閱。	AWS 管理員、雲端架構師

測試訂閱通知

任務	Description	所需的技能
停用 AWS KMS 金鑰。	1. 登入 AWS 管理主控台並開啟 AWS KMS 主控台。	AWS 管理員

任務	Description	所需的技能
	2. 若要變更區域，請選擇目前顯示區域的名稱，然後選擇您要切換的區域。 3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。 4. 選取您要啟用或停用之 AWS KMS 金鑰的核取方塊。 5. 若要停用 AWS KMS 金鑰，請選擇金鑰動作，然後選擇停用。	
驗證訂閱。	確認您已收到 Amazon SNS 通知電子郵件。	AWS 管理員

清除資源

任務	Description	所需的技能
刪除 CloudFormation 堆疊。	1. 登入 AWS 管理主控台並開啟 CloudFormation 主控台。 2. 在導覽窗格中，選擇 Stacks (堆疊)。 3. 選取您先前建立的堆疊，然後選擇刪除。	AWS 管理員

相關資源

- AWS CloudFormation (AWS 文件)
- 在 AWS CloudFormation 主控台上建立堆疊 (AWS CloudFormation 文件)
- 在 AWS 上建置事件驅動型架構 (AWS Workshop Studio 文件)

- AWS Key Management Service 最佳實務 (AWS 白皮書）
- AWS Key Management Service 的安全最佳實務 (AWS KMS 開發人員指南）

其他資訊

Amazon SNS 預設提供傳輸中加密。若要符合安全最佳實務，您也可以使用 AWS KMS 客戶受管金鑰啟用 Amazon SNS 的伺服器端加密。

在非工作負載子網路的多帳戶 VPC 設計中保留可路由 IP 空間

Adam Spicer，Amazon Web Services

總結

Amazon Web Services (AWS) 已發佈最佳實務，建議將虛擬私有雲端 (VPC) 中的專用子網路用於傳輸閘道附件和 Gateway Load Balancer 端點（以支援 AWS Network Firewall 或第三方設備）。這些子網路用於包含這些服務的彈性網路介面。如果您同時使用 AWS Transit Gateway 和 Gateway Load Balancer，則會在 VPC 的每個可用區域中建立兩個子網路。由於 VPCs的設計方式，這些額外的子網路不能小於 /28 遮罩，並且可以使用寶貴的可路由 IP 空間，否則可用於可路由工作負載。此模式示範如何針對這些專用子網路使用次要、不可路由的無類別網域間路由 (CIDR) 範圍，以協助保留可路由 IP 空間。

先決條件和限制

先決條件

- 可路由 IP 空間的多 VPC 策略
- 您正在使用之服務的不可路由 CIDR 範圍 (傳輸閘道附件和 Gateway Load Balancer 或 Network Firewall 端點)

Architecture

目標架構

此模式包含兩個參考架構：一個架構具有用於傳輸閘道 (TGW) 連接和閘道Load Balancer端點 (GWLBe) 的子網路，而第二個架構僅具有用於 TGW 連接的子網路。

架構 1 – 具有傳入路由至設備的 TGW 連接 VPC

下圖代表跨越兩個可用區域的 VPC 參考架構。在輸入時，VPC 會使用輸入路由模式，將目的地為公有子網路的流量導向至bump-in-the-wire設備以進行防火牆檢查。TGW 連接支援從私有子網路輸出到單獨的 VPC。

此模式會針對 TGW 連接子網路和 GWLBe 子網路使用不可路由的 CIDR 範圍。在 TGW 路由表中，此不可路由 CIDR 會使用一組更具體的路由，以黑洞（靜態）路由設定。如果路由要傳播到 TGW 路由表，則會套用這些更具體的黑洞路由。

在此範例中，/23 可路由 CIDR 會分割並完全配置給可路由子網路。

架構 2 – TGW 連接的 VPC

下圖代表跨越兩個可用區域的 VPC 的另一個參考架構。TGW 連接支援從私有子網路到個別 VPC 的傳出流量（輸出）。它只會針對 TGW 連接子網路使用不可路由的 CIDR 範圍。在 TGW 路由表中，此不可路由 CIDR 會使用一組更具體的路由，以黑洞路由設定。如果路由要傳播到 TGW 路由表，則會套用這些更具體的黑洞路由。

在此範例中，/23 可路由 CIDR 會分割並完全配置給可路由子網路。

工具

AWS 服務和資源

- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路類似於您在自己的資料中心內操作的傳統網路，具有使用可擴展的 AWS 基礎設施的優勢。在此模式中，VPC 次要 CIDRs用於保留工作負載 CIDRs 中的可路由 IP 空間。
- 網際網路閘道傳入路由（邊緣關聯）可與專用不可路由子網路的 Gateway Load Balancer 端點搭配使用。
- AWS Transit Gateway 是連接 VPCs和內部部署網路的中央中樞。在此模式中，VPCs會集中連接至傳輸閘道，而傳輸閘道附件則位於專用的不可路由子網路中。
- Gateway Load Balancer 可讓您部署、擴展和管理虛擬設備，如防火牆、入侵偵測與預防系統，以及深層封包檢查系統。閘道充當所有流量的單一入口和出口點。在此模式中，Gateway Load Balancer 的端點可用於專用的不可路由子網路。
- AWS Network Firewall 是 AWS 雲端中 VPCs 具狀態、受管的網路防火牆和入侵偵測和預防服務。在此模式中，防火牆的端點可用於專用的不可路由子網路。

程式碼儲存庫

此模式的 Runbook 和 AWS CloudFormation 範本可在 GitHub 不可路由次要 CIDR 模式儲存庫中使用。您可以使用範例檔案，在您的環境中設定工作實驗室。

最佳實務

AWS Transit Gateway

- 為每個傳輸閘道 VPC 連接使用個別子網路。
- 從傳輸閘道連接子網路的次要不可路由 CIDR 範圍配置 /28 子網路。
- 在每個傳輸閘道路由表中，將不可路由 CIDR 範圍的靜態、更具體的路由新增為黑洞。

Gateway Load Balancer 和輸入路由

- 使用輸入路由將流量從網際網路導向 Gateway Load Balancer 端點。
- 為每個 Gateway Load Balancer 端點使用單獨的子網路。
- 從閘道Load Balancer端點子網路的次要不可路由 CIDR 範圍配置 /28 子網路。

史詩

建立 VPCs

任務	Description	所需的技能
判斷不可路由的 CIDR 範圍。	決定不可路由的 CIDR 範圍，用於傳輸閘道連接子網路，以及（選擇性）用於任何 Gateway Load Balancer 或 Network Firewall 端點子網路。此 CIDR 範圍將用作 VPC 的次要 CIDR。它不能從 VPC 的主要 CIDR 範圍或更大的網路路由。	雲端架構師
判斷 VPCs 的可路由 CIDR 範圍。	決定一組將用於 VPCs可路由 CIDR 範圍。此 CIDR 範圍將用作 VPCs的主要 CIDR。	雲端架構師
建立 VPCs。	建立 VPCs並將其連接到傳輸閘道。每個 VPC 應具有可路由的主要 CIDR 範圍，以及不可路由的次要 CIDR 範圍，這取決於您在前兩個步驟中決定的範圍。	雲端架構師

設定 Transit Gateway 黑洞路由

任務	Description	所需的技能
將更具體的不可路由 CIDRs 建立為黑洞。	每個傳輸閘道路由表都需要為不可路由 CIDRs 建立一組黑洞路由。這些設定可確保來自次要 VPC CIDR 的任何流量都無法路由，也不會洩漏到較大的網路。這些路由應該比設定為 VPC 上次要 CIDR 的不可路由 CIDR 更具體。例如，如果次要不可路由 CIDR 為 100.64.0.0/26，則傳輸閘道路由表中的黑洞路由應為 100.64.0.0/27 和 100.64.0.32/27。	雲端架構師

相關資源

- 部署 Gateway Load Balancer 的最佳實務
- 具有 Gateway Load Balancer 的分散式檢查架構
- 網路沉浸日 – 網際網路到 VPC Firewall Lab
- 傳輸閘道設計最佳實務

其他資訊

在處理需要大量 IP 地址的擴展容器部署時，不可路由的次要 CIDR 範圍也很有用。您可以搭配私有 NAT Gateway 使用此模式，以使用不可路由的子網路來託管容器部署。如需詳細資訊，請參閱部落格文章如何使用私有 NAT 解決方案解決私有 IP 耗盡問題。

AWS Service Catalog 使用程式碼儲存庫在 中佈建 Terraform 產品

Rahul Sharad Gaikwad 醫生和 Tamilselvan P，Amazon Web Services

總結

AWS Service Catalog 支援對 HashiCorp Terraform 組態進行管控的自助式佈建。如果您使用 Terraform，則可以使用 Service Catalog 做為單一工具， AWS 在 內大規模組織、管理和分發 Terraform 組態。您可以存取 Service Catalog 金鑰功能，包括將標準化和預先核准的基礎設施編目為程式碼 (IaC) 範本、存取控制、具有最低權限存取的雲端資源佈建、版本控制、共用到數千個 AWS 帳戶和標記。工程師、資料庫管理員和資料科學家等最終使用者，會看到他們有權存取的產品和版本清單，而且可以透過單一動作進行部署。

此模式可協助您使用 Terraform 程式碼部署 AWS 資源。GitHub 儲存庫中的 Terraform 程式碼可透過 Service Catalog 存取。使用此方法，您可以將產品與現有的 Terraform 工作流程整合。管理員可以使用 Terraform 建立 Service Catalog 產品組合，並將 AWS Launch Wizard 產品新增至這些產品組合。

以下是此解決方案的優點：

- 由於 Service Catalog 中的復原功能，如果在部署期間發生任何問題，您可以將產品還原至先前的版本。
- 您可以輕鬆識別產品版本之間的差異。這可協助您解決部署期間的問題。
- 您可以在 Service Catalog 中設定儲存庫連線，例如 GitHub 或 GitLab。您可以直接透過儲存庫進行產品變更。

如需 整體優勢的詳細資訊 AWS Service Catalog，請參閱什麼是 Service Catalog。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 包含 ZIP 格式 Terraform 組態檔案的 GitHub、BitBucket 或其他儲存庫。
- AWS Serverless Application Model 命令列界面 (AWS SAM CLI)，已安裝。
- AWS Command Line Interface (AWS CLI)，已安裝並設定。
- Go，已安裝。
- Python 3.9 版，已安裝。 AWS SAM CLI 需要此版本的 Python。

- 寫入和執行 AWS Lambda 函數的許可，以及存取和管理 Service Catalog 產品和產品組合的許可。

Architecture

該圖顯示以下工作流程：

1. 當 Terraform 組態就緒時，開發人員會建立包含所有 Terraform 程式碼的 .zip 檔案。開發人員會將 .zip 檔案上傳至連線至 Service Catalog 的程式碼儲存庫。
2. 管理員會將 Terraform 產品與 Service Catalog 中的產品組合建立關聯。管理員也會建立啟動限制，允許最終使用者佈建產品。
3. 在 Service Catalog 中，最終使用者使用 Terraform 組態啟動 AWS 資源。他們可以選擇要部署的產品版本。

工具

AWS 服務

- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- AWS Service Catalog 可協助您集中管理已核准的 IT 服務目錄 AWS。最終使用者可在機構所設的限制範圍內，迅速地只部署自己需要且經核准的 IT 服務。

其他服務

- Go 是 Google 支援的開放原始碼程式設計語言。
- Python 是一種一般用途的電腦程式設計語言。

程式碼儲存庫

如果您需要可透過 Service Catalog 部署的範例 Terraform 組態，您可以使用 GitHub Amazon Macie Organization Setup using Terraform 儲存庫中的組態。不需要在此儲存庫中使用程式碼範例。

最佳實務

- 透過 Service Catalog 啟動產品時，不要在 Terraform 組態檔案 (`terraform.tfvars`) 中提供變數的值，而是設定變數值。

- 僅將產品組合的存取權授予特定使用者或管理員。
- 遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱《 AWS Identity and Access Management (IAM) 文件》中的授予最低權限和安全最佳實務。

史詩

設定您的本機工作站

任務	Description	所需的技能
（選用）安裝 Docker。	如果您想要在開發環境中執行 AWS Lambda 函數，請安裝 Docker。如需相關說明，請參閱 Docker 文件中的安裝 Docker 引擎。	DevOps 工程師

安裝適用於 Terraform 的 AWS Service Catalog 引擎。	1. 輸入下列命令以複製 AWS Service Catalog Engine for Terraform 儲存庫。
	`git clone https://github.com/aws-samples/service-catalog-engine-for-terraform-os.git`
	2. 導覽至複製儲存庫的根目錄。
	3. 輸入以下命令。這會安裝引擎。
	`run ./bin/bash/deploy-tre.sh -r`
	在自動安裝期間，不會使用預設設定檔中的 AWS 區域設定。相反地，您可以在執行此命令時提供 區域。
		DevOps 工程師、AWS 管理員

連接 GitHub 儲存庫

任務	Description	所需的技能
建立 GitHub 儲存庫的連線。	1. 登入 AWS 管理主控台，然後開啟開發人員工具主控台。您可以選擇 AWS CodePipeline 或 等服務來存取開發人員工具主控台 AWS CodeDeploy。 2. 在左側導覽窗格中，選擇設定，然後選擇連線。 3. 選擇建立連線。 4. 選取您維護 Terraform 原始程式碼的儲存庫。例如，您可以選擇 Bitbucket、GitHub 或 GitHub Enterprise Server。 5. 輸入連線的名稱，然後選擇連線。 6. 出現提示時，請驗證儲存庫。 身分驗證完成後，會建立連線，且狀態會變更為作用中。	AWS 管理員

在 Service Catalog 中建立 Terraform 產品

任務	Description	所需的技能
建立 Service Catalog 產品。	1. 開啟 AWS Service Catalog 主控台。 2. 導覽至管理區段，然後選擇產品清單。	AWS 管理員

任務	Description	所需的技能
	3. 選擇建立產品。 4. 在產品詳細資訊區段的建立產品頁面上，選擇外部產品類型。 Service Catalog 使用此產品類型來支援 Terraform Community Edition 產品。 5. 輸入 Service Catalog 產品的名稱和擁有者。 6. 選取使用 CodeStar 供應商指定程式碼儲存庫。 7. 輸入儲存庫的下列資訊： • 使用 連線至您的供應商 AWS CodeConnections – 選取您先前建立的連線。 • 儲存庫 – 選取儲存庫。 • 分支 – 選取分支。 • 範本檔案路徑 – 選擇儲存程式碼範本檔案的路徑。檔案名稱應以 結尾`tar.gz`。 8. 在版本名稱和描述下，提供產品版本的相關資訊。 9. 選擇建立產品。	

任務	Description	所需的技能
建立組合。	1. 開啟 AWS Service Catalog 主控台。 2. 導覽至管理區段，然後選擇產品組合。 3. 選擇建立產品組合。 4. 輸入下列值： • 產品組合名 – `Sample terraform` • 產品組合描述 – `Sample portfolio for Terraform configurations` • 擁有者 – 您的聯絡資訊，例如電子郵件地址 5. 選擇建立。	AWS 管理員
將 Terraform 產品新增至產品組合。	1. 開啟 AWS Service Catalog 主控台。 2. 導覽至管理區段，然後選擇產品清單。 3. 選取您先前建立的 Terraform 產品。 4. 選擇動作，然後選擇將產品新增至產品組合。 5. 選擇`Sample terraform` 產品組合。 6. 選擇將產品新增至產品組合。	AWS 管理員

任務	Description	所需的技能
建立存取政策。	1. 開啟 AWS Identity and Access Management (IAM) 主控台。 2. 在導覽窗格上選擇 Policies (政策)。 3. 在內容窗格中，選擇 Create policy (建立政策)。 4. 選擇 JSON 選項。 5. 在此模式額外資訊區段的存取政策中輸入範例 JSON 政策。 6. 選擇下一步。 7. 在檢閱和建立頁面上的政策名稱方塊中，輸入 `TerraformResourceCreationAndArtifactAccessPolicy` 。 8. 選擇建立政策。	AWS 管理員

任務	Description	所需的技能
建立自訂信任政策。	1. 開啟 IAM 主控台。 2. 在導覽窗格中，選擇 Roles (角色)。 3. 選擇 Create Role (建立角色)。 4. 在信任的實體類型下，選擇自訂信任政策。 5. 在 JSON 政策編輯器中，在此模式的其他資訊區段的信任政策中輸入範例 JSON 政策。 6. 選擇下一步。 7. 在許可政策下，選擇您先前建立`Terraform ResourceCreationAndArtifactAccessPolicy` 的 。 8. 選擇下一步。 9. 在角色詳細資訊下，於角色名稱方塊中，輸入 `SCLaunch-product` 。 **Important** 角色名稱必須以 開頭`SCLaunch`。 10選擇建立角色。	AWS 管理員

任務	Description	所需的技能
將啟動限制新增至 Service Catalog 產品。	1. 以具有管理許可的使用者 AWS 管理主控台 身分登入 。 2. 開啟 AWS Service Catalog 主控台。 3. 在導覽窗格中，選擇產品組合。 4. 選擇您先前建立的產品組合。 5. 在Portfolio 詳細資訊頁面上，選擇限制條件索引標籤，然後選擇建立限制條件。 6. ForProduct，選取您先前建立的 Terraform 產品。 7. 在啟動限制下，針對方法，選擇輸入角色名稱。 8. 在角色名稱方塊中，輸入 `SCLaunch-product` 。 9. 選擇建立。	AWS 管理員

任務	Description	所需的技能
授予產品存取權。	1. 開啟 AWS Service Catalog 主控台。 2. 在導覽窗格中，選擇產品組合。 3. 選擇您先前建立的產品組合。 4. 選擇Accesstab，然後選擇授予存取權。 5. 選擇 Rolestab，然後選擇應有權部署此產品的角色。 6. 選擇 Grant access (授與存取權)。	AWS 管理員
啟動產品。	1. 以具有部署 Service Catalog 產品許可的使用者 AWS 管理主控台 身分登入 。 2. 開啟 AWS Service Catalog 主控台。 3. 在導覽窗格中，選擇產品。 4. 選擇您先前建立的生產，然後選擇啟動產品。 5. 輸入產品名稱並定義任何必要的參數。 6. ChooseLaunch 產品。	DevOps 工程師

驗證部署

任務	Description	所需的技能
驗證部署。	Service Catalog 佈建工作流程有兩個 AWS Step Functions 狀態機器：	DevOps 工程師

任務	Description	所需的技能
	• `ManageProvisionedProductStateMachine` – Service Catalog 會在佈建新的 Terraform 產品和更新現有的 Terraform 佈建產品時叫用此狀態機器。 • `TerminateProvisionedProductStateMachine` – Service Catalog 會在終止現有的 Terraform 佈建產品時叫用此狀態機器。 您可以檢查`ManageProvisionedProductStateMachine` 狀態機器的日誌，以確認已佈建產品。 1. 登入 AWS 管理主控台，然後開啟 AWS Step Functions 主控台。 2. 在左側導覽窗格中，選擇狀態機器。 3. 選擇 `ManageProvisionedProductStateMachine` 。 4. 在執行清單中，輸入佈建的產品 ID 以尋找執行。 **Note** 狀態檔案後端儲存貯體名稱開頭為 `sc-terraf`	

任務	Description	所需的技能
	`orm-engine-state-`。 5. 驗證已在帳戶中建立所有必要的資源。	

清除基礎設施

任務	Description	所需的技能
刪除佈建的產品。	1. 以具有部署 Service Catalog 產品許可的使用者 AWS 管理主控台 身分登入 。 2. 開啟 AWS Service Catalog 主控台。 3. 在左側導覽中，選擇佈建的產品。 4. 選取您建立的產品。 5. 在動作清單中，選擇終止。 6. 在確認文字方塊中，輸入 `terminate` ，然後選擇終止佈建的產品。 7. 重複這些步驟來終止所有佈建的產品。	DevOps 工程師
移除適用於 Terraform 的 AWS Service Catalog 引擎。	1. 以具有管理許可的使用者 AWS 管理主控台 身分登入 。 2. 開啟 Amazon Simple Storage Service (Amazon S3) 主控台。 3. 在導覽窗格中，選擇 儲存貯體。	AWS 管理員

任務	Description	所需的技能
	4. 選取儲存sc-terraform-engine-logging-XXXX貯體。 5. 選擇空白。 6. 針對下列儲存貯體重複步驟 4–5： • sc-terraform-engine-state-XXXX • terraform-engine-bootstrap-XXXX 7. 開啟AWS CloudFormation 主控台，然後驗證您是否正確 AWS 區域。 8. 在左側導覽中，選擇堆疊。 9. 選取 SAM-TRE，然後選擇刪除。等待堆疊刪除。 10選取 Bootstrap-TRE ，然後選擇刪除。等待堆疊刪除。	

相關資源

AWS 文件

- Terraform 產品入門

Terraform 文件

- Terraform 安裝
- Terraform 後端組態
- Terraform AWS 提供者文件

其他資訊

存取政策

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:ExistingObjectTag/servicecatalog:provisioning": "true"
                }
            }
        },
        {
            "Action": [
                "s3:CreateBucket*",
                "s3:DeleteBucket*",
                "s3:Get*",
                "s3:List*",
                "s3:PutBucketTagging"
            ],
            "Resource": "arn:aws:s3:::*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "resource-groups:CreateGroup",
                "resource-groups:ListGroupResources",
                "resource-groups:DeleteGroup",
                "resource-groups:Tag"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "tag:GetResources",
                "tag:GetTagKeys",
                "tag:GetTagValues",
```

```
                "tag:TagResources",
                "tag:UntagResources"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

信任政策

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GivePermissionsToServiceCatalog",
            "Effect": "Allow",
            "Principal": {
                "Service": "servicecatalog.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::account_id:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringLike": {
                    "aws:PrincipalArn": [
                        "arn:aws:iam::accounti_id:role/TerraformEngine/
TerraformExecutionRole*",
                        "arn:aws:iam::accounti_id:role/TerraformEngine/
ServiceCatalogExternalParameterParserRole*",
                        "arn:aws:iam::accounti_id:role/TerraformEngine/
ServiceCatalogTerraformOSParameterParserRole*"
                    ]
                }
            }
        }
    ]
}
```

使用 Amazon SES AWS 帳戶 向單一電子郵件地址註冊多個

Joe Wozniak 和 Shubhangi Vishwakarma，Amazon Web Services

總結

此模式說明如何從與 相關聯的電子郵件地址分離真實電子郵件地址 AWS 帳戶。 AWS 帳戶 建立帳戶時需要提供唯一的電子郵件地址。在某些組織中， 管理的 團隊 AWS 帳戶 必須承擔與其簡訊團隊管理許多唯一電子郵件地址的負擔。對於管理許多的大型組織來說，這可能很困難 AWS 帳戶。此外，如果您的電子郵件系統不允許 Sieve Email Filtering： Subaddress Extension (RFC 5233) 中定義的加號地址或子地址，請在電子郵件地址的本機部分結尾加上加號 (+) 和識別符，例如`admin+123456789123@example.com`：此模式有助於克服此限制。

此模式提供唯一的電子郵件地址販賣解決方案，可讓 AWS 帳戶 擁有者將一個電子郵件地址與多個電子郵件地址建立關聯 AWS 帳戶。然後， AWS 帳戶 擁有者的真實電子郵件地址會與資料表中這些產生的電子郵件地址相關聯。解決方案會處理唯一電子郵件帳戶的所有傳入電子郵件、查詢每個帳戶的擁有者，然後將任何收到的訊息轉送給擁有者。

先決條件和限制

先決條件

- 對 的管理存取權 AWS 帳戶。
- 存取開發環境。
- （選用） 熟悉 AWS Cloud Development Kit (AWS CDK) 工作流程和 Python 程式設計語言，可協助您疑難排解任何問題或修改。

限制

- 整體發佈的電子郵件地址長度為 64 個字元。如需詳細資訊，請參閱 AWS Organizations API 參考中的CreateAccount。

產品版本

- Node.js 22.x 版或更新版本
- Python 3.13 或更新版本
- Python 套件 pip 和 virtualenv
- AWS CDK CLI 2.1019.2 版或更新版本

- Docker 20.10.x 或更新版本

Architecture

目標技術堆疊

- CloudFormation 堆疊
- AWS Lambda 函數
- Amazon Simple Email Service (Amazon SES) 規則和規則集
- AWS Identity and Access Management (IAM) 角色和政策
- Amazon Simple Storage Service (Amazon S3) 儲存貯體和儲存貯體政策
- AWS Key Management Service (AWS KMS) 金鑰和金鑰政策
- Amazon Simple Notification Service (Amazon SNS) 主題和主題政策
- Amazon DynamoDB 資料表

目標架構

此圖表顯示兩個流程：

- 電子郵件地址販賣流程：在圖表中，電子郵件地址販賣流程（下節）通常以帳戶販賣解決方案或外部自動化開始，或手動叫用。在請求中，呼叫 Lambda 函數時，承載會包含所需的中繼資料。函數會使用此資訊來產生唯一的帳戶名稱和電子郵件地址，將其存放在 DynamoDB 資料庫中，然後將值傳回給發起人。然後，這些值可用於建立新的 AWS 帳戶（通常是使用 AWS Organizations)。
- 電子郵件轉送流程：此流程如上圖的上一節所示。使用從電子郵件地址販賣流程產生的帳戶電子郵件建立 AWS 帳戶 時， 會將各種電子郵件 AWS 傳送至該電子郵件地址，例如帳戶註冊確認和定期通知。透過遵循此模式中的步驟，您可以使用 AWS 帳戶 Amazon SES 設定您的 以接收整個網域的電子郵件。此解決方案會設定轉送規則，允許 Lambda 處理所有傳入電子郵件、檢查T0地址是否在 DynamoDB 資料表中，並將訊息改為轉送至帳戶擁有者的電子郵件地址。使用此程序可讓帳戶擁有者將多個帳戶與一個電子郵件地址建立關聯。

自動化和擴展

此模式使用 AWS CDK 來完全自動化部署。解決方案使用 AWS 受管服務，可自動擴展（或設定為）以符合您的需求。Lambda 函數可能需要額外的組態，才能滿足您的擴展需求。如需詳細資訊，請參閱 Lambda 文件中的了解 Lambda 函數擴展。

工具

AWS 服務

- CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和 區域的生命週期中管理資源。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Email Service (Amazon SES) 可協助您使用自己的電子郵件地址和網域來傳送和接收電子郵件。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

部署所需的工具

- 具有 AWS CLI 和 IAM 存取您 的開發環境 AWS 帳戶。如需詳細資訊，請參閱相關資源區段中的連結。
- 在開發系統上，安裝下列項目：
  - Git 命令列工具，可從 Git 下載網站取得。
  - AWS CLI 設定 存取憑證的 AWS CDK。如需詳細資訊，請參閱 AWS CLI 文件。
  - Python 3.13 版或更新版本，可從 Python 下載網站取得。
  - UV for Python 套件管理。如需安裝說明，請參閱 UV 安裝指南。
  - Node.js 22.x 版或更新版本。如需安裝說明，請參閱 Node.js 文件。
  - AWS CDK CLI 2.1019.2 版或更新版本。如需安裝說明，請參閱 AWS CDK 文件。
  - Docker 20.10.x 版或更新版本。如需安裝說明，請參閱 Docker 文件。

Code

此模式的程式碼可在 GitHub AWS 帳戶 原廠電子郵件儲存庫中使用。

史詩

配置目標部署環境

任務	Description	所需的技能
識別或建立 AWS 帳戶。	識別您擁有完整管理存取權 AWS 帳戶 的現有或新 ，以部署電子郵件解決方案。	AWS 管理員、雲端管理員
設定部署環境。	請依照下列步驟，設定易於使用的部署環境並設定相依性： 1. 使用工具區段中列出的工具設定您的開發環境。 2. 使用 命令將 GitHub AWS 帳戶 原廠電子郵件儲存庫程式碼基礎複製到您的開發環境： `git clone https://github.com/aws-samples/aws-account-factory-email` 3. 在 `requirements.in` 檔案中 （在儲存庫的根目錄中），更新開頭為 的行`aws-cdk-lib==` ，以符合最新版本的 AWS CDK。若要識別什麼是最新版本，請執行 `npm view aws-cdk-lib` 並尋找關鍵字「最新：」。	AWS DevOps，應用程式開發人員

任務	Description	所需的技能
	4. 如果您在 中更新了 版本`requirements.in` ，請執行下列命令來使用 UV 更新`requirements.txt` 檔案： `uv pip compile requirements.in -o requirements.txt --python-version 3.13`	

設定已驗證的網域

任務	Description	所需的技能
識別和配置網域。	電子郵件轉送功能需要專用網域。識別並配置您可以使用 Amazon SES 驗證的網域或子網域。此網域應該可用於在部署電子郵件轉送解決方案 AWS 帳戶 的 內接收內送電子郵件。 網域需求： • 網域應該是標準網域或子網域。 • 網域應該可從外部 DNS 解析，因為它將用於接收來自組織外部的電子郵件。	雲端管理員、網路管理員、DNS 管理員
驗證網域。	確認已識別的網域可用於接受傳入電子郵件。 完成 Amazon SES 文件中驗證 Amazon SES 電子郵件接收網域的指示。 Amazon SES 這將	應用程式開發人員、AWS DevOps

任務	Description	所需的技能
	需要與負責網域 DNS 記錄的人員或團隊協調。	
設定 MX 記錄。	使用指向 AWS 帳戶 和 區域 中 Amazon SES 端點的 MX 記錄來設定您的網域。如需詳細資訊，請參閱《Amazon SES 文件》中的發佈 Amazon SES 電子郵件接收的 MX 記錄。Amazon SES	雲端管理員、網路管理員、DNS 管理員

部署電子郵件販賣和轉送解決方案

任務	Description	所需的技能
修改 中的預設值cdk.json。	編輯 cdk.json 檔案（在儲存庫根中）中的一些預設值，讓解決方案在部署之後可以正常運作。 1. 修改 SES_DOMAIN_NAME 值以符合您先前驗證的網域名稱。 2. 修改 ADDRESS_FROM 值，以包含與 中相同的網域SES_DOMAIN_NAME 。地址的本機部分應由您的雲端團隊決定。此地址會成為透過 解決方案轉送之每個電子郵件FROM的地址。 3. 修改 ADDRESS_ADMIN 值以符合任何不相符傳入訊息將轉送到的電子郵件地址。	應用程式開發人員、AWS DevOps

任務	Description	所需的技能
	此值必須是有效的操作電子郵件地址。	

任務	Description	所需的技能

部署電子郵件販賣和轉送解決方案。	1. 建立 Python 虛擬環境：
	`uv venv --python 3.13`
	2. 啟用 Python 虛擬環境：
	`source .venv/bin/activate`
	或者，在 Windows 平台上，使用：
	`% .venv\Scripts\activate.bat`
	3. 安裝所有 Python 需求，沒有錯誤：
	`uv pip install -r requirements.txt`
	4. 合成 CloudFormation 範本：
	`cdk synth`
	確認沒有錯誤，且完整的 CloudFormation 範本包含預期的輸出。
	5. （選用）如果您是第一次將 AWS CDK 程式碼部署到目前 AWS 帳戶 或 區域，請引導環境。如需詳細資訊，請參閱 AWS CDK 文件中的AWS CDK 引導。
		應用程式開發人員、AWS DevOps

任務	Description	所需的技能

	`cdk bootstrap aws://AWS-ACCOUNT-NUMBER/REGION`
	將 AWS-ACCOUNT-NUMBER 和 取代REGION為實際值。
	6. 部署解決方案：
	`cdk bootstrap`
	`cdk deploy`
	命令應該在沒有錯誤的情況下完成。	

任務	Description	所需的技能
確認已部署解決方案。	開始測試之前，請確認已成功部署解決方案： 1. 開啟 CloudFormation 主控台，並尋找包含名稱的 CloudFormation 堆疊`AwsMailFwdStack` 。 2. 確認此`AwsMailFwdStack` 堆疊具有下列資源： • Lambda 函式 • Amazon SES 規則和規則集 • (IAM) 角色和政策 • Amazon S3 儲存貯體和儲存貯體政策 • AWS KMS 金鑰和金鑰政策 • Amazon SNS 主題和主題政策 • DynamoDB 表	應用程式開發人員、AWS DevOps

驗證電子郵件販賣和轉送是否如預期般運作

任務	Description	所需的技能
驗證 API 是否正常運作。	在此步驟中，您將測試資料提交至解決方案的 API，並確認解決方案會產生預期的輸出，且後端操作已如預期般執行。 使用測試輸入手動執行 Vend Email Lambda 函數。（如	應用程式開發人員、AWS DevOps

任務	Description	所需的技能
	需範例，請參閱 sample_vend_request.json 檔案。) 對於 `OwnerAddress` ，請使用有效的電子郵件地址。API 應該如預期傳回具有值的帳戶名稱和帳戶電子郵件。	
確認電子郵件正在轉送。	在此步驟中，您會透過系統傳送測試電子郵件，並確認電子郵件已轉送給預期的收件人。 1. 從最後一個步驟取得帳戶電子郵件。 2. 使用測試主旨和內文文字傳送電子郵件至此地址。 3. 確認您已在帳戶擁有者的電子郵件地址收到電子郵件。 4. 確認您收到的電子郵件具有符合 中`ADDRESS_FROM` 設定`FROM`的地址`cdk.json`。 5. 確認收到的電子郵件主旨和內文與原始傳送的訊息相同。	應用程式開發人員、AWS DevOps

疑難排解

問題	解決方案
系統未如預期轉送電子郵件。	確認您的設定正確： 1. 您應該已完成網域的 Amazon SES 驗證程序。 2. 您的網域應以指向 AWS 帳戶 和 區域中 Amazon SES 端點的 MX 記錄正確設定。如

問題	解決方案
	需詳細資訊，請參閱《Amazon SES 文件》中的發佈 Amazon SES 電子郵件接收的 MX 記錄。 Amazon SES 驗證網域設定後，請依照下列步驟執行： 1. 為您部署解決方案的帳戶和區域開啟 Amazon CloudWatch 主控台，然後在導覽窗格中導覽至 CloudWatch 日誌群組。 2. 搜尋 的日誌群組清單`SesMailForwardLogGroup` 。 3. 調查此群組中的日誌，以查看電子郵件販賣和轉送程序期間是否產生任何錯誤。

問題	解決方案
當您嘗試部署 AWS CDK 堆疊時，您會收到類似以下的錯誤： 「範本格式錯誤：無法辨識的資源類型」	在大多數情況下，此錯誤訊息表示您要鎖定的區域沒有所有可用的 AWS 服務。如果您使用 Amazon EC2 執行個體來部署解決方案，則可能是以與執行個體執行所在區域不同的區域為目標。 **Note** 根據預設， AWS CDK 會部署到您在 中設定的 區域和帳戶 AWS CLI。 可能的解決方案： 1. 透過依區域檢閱 AWS 服務，調查此解決方案所需的所有服務（請參閱此模式稍早的目標技術堆疊一節）是否位於 AWS 區域 您目標的 中。 2. 如果您使用的是 EC2 執行個體，並以與執行個體執行所在區域不同的區域為目標，請務必先設定`AWS_DEFAULT_REGION` 環境變數，或使用 設定區域， AWS CLI 再部署解決方案。如需詳細資訊，請參閱 AWS CLI 文件中的設定 的環境變數 AWS CLI。或者，您可以依照AWS CDK 環境文件中的指示，修改儲存庫根目錄中`app.py`的檔案，以包含硬式編碼的帳戶 ID 和區域。

問題	解決方案
部署解決方案時，您會收到錯誤訊息： 「部署失敗：錯誤：AwsMailFwdStack：找不到 SSM 參數 /cdk-bootstrap/hnb659fds/version。環境是否已引導？ 請執行「cdk 引導」	如果您從未將任何 AWS CDK 資源部署到您要鎖定的 AWS 帳戶 和 區域，您必須先執行錯誤指示的 `cdk bootstrap` 命令。如果您在執行 bootstrapping 命令後繼續收到此錯誤，您可能會嘗試將解決方案部署到與開發環境執行所在區域不同的區域。 若要解決此問題，請先設定`AWS_DEFAULT_REGION` 環境變數或使用 設定區域，AWS CLI 再部署解決方案。或者，您可以依照AWS CDK 環境文件中的指示，修改儲存庫根目錄中`app.py`的檔案，以包含硬式編碼的帳戶 ID 和區域。

相關資源

- 如需安裝 的說明 AWS CLI，請參閱安裝或更新至最新版本的 AWS CLI。
- 如需 AWS CLI 使用 IAM 存取登入資料設定 的說明，請參閱設定 的設定 AWS CLI。
- 如需 的說明 AWS CDK，請參閱 入門 AWS CDK。

其他資訊

成本

當您部署此解決方案時， AWS 帳戶 持有者可能會產生與使用下列服務相關的成本。 請務必了解這些服務的計費方式，以便了解任何潛在的費用。如需定價資訊，請參閱下列頁面：

- Amazon SES 定價
- Amazon S3 定價
- AWS KMS 定價
- AWS Lambda 定價
- Amazon DynamoDB 定價

在單一帳戶 AWS 環境中設定混合網路的 DNS 解析

Abdullahi Olaoye，Amazon Web Services

總結

此模式說明如何設定完全混合網域名稱系統 (DNS) 架構，以啟用現場部署資源、AWS 資源和網際網路 DNS 查詢的end-to-end DNS 解析，而不會產生管理開銷。模式說明如何設定 Amazon Route 53 Resolver 轉送規則，以根據網域名稱判斷應從 AWS 傳送 DNS 查詢的位置。內部部署資源的 DNS 查詢會轉送至內部部署 DNS 解析程式。AWS 資源的 DNS 查詢和網際網路 DNS 查詢由 Route 53 Resolver 解析。

此模式涵蓋 AWS 單一帳戶環境中的混合 DNS 解析。如需有關在 AWS 多帳戶環境中設定傳出 DNS 查詢的資訊，請參閱在多帳戶 AWS 環境中設定混合網路的 DNS 解析模式。

先決條件和限制

先決條件

- 一個 AWS 帳戶
- AWS 帳戶中的虛擬私有雲端 (VPC)
- 內部部署環境與 VPC 之間的網路連線，透過 AWS Virtual Private Network (AWS VPN) 或 AWS Direct Connect
- 現場部署 DNS 解析程式的 IP 地址（可從 VPC 存取）
- 要轉送至內部部署解析程式的網域/子網域名稱（例如 onprem.mydc.com)
- AWS 私有託管區域的網域/子網域名稱（例如 myvpc.cloud.com)

Architecture

目標技術堆疊

- Amazon Route 53 私有託管區域
- Amazon Route 53 Resolver
- Amazon VPC
- AWS VPN 或 Direct Connect

目標架構

工具

- Amazon Route 53 Resolver 透過在整個混合雲端中啟用無縫 DNS 查詢解析，讓企業客戶更輕鬆地使用混合雲端。您可以建立 DNS 端點和條件式轉送規則，以解析現場部署資料中心和 VPCs 之間的 DNS 命名空間。
- Amazon Route 53 私有託管區域是一種容器，其中包含您希望 Route 53 如何回應您使用 Amazon VPC 服務建立之一或多個 VPCs 內網域及其子網域的 DNS 查詢的相關資訊。

史詩

設定私有託管區域

任務	Description	所需的技能
為 AWS 預留網域名稱建立 Route 53 私有託管區域，例如 myvpc.cloud.com。	此區域會保留應從內部部署環境解析之 AWS 資源的 DNS 記錄。如需說明，請參閱 Route 53 文件中的建立私有託管區域。	網路管理員、系統管理員
將私有託管區域與您的 VPC 建立關聯。	若要啟用 VPC 中的資源來解析此私有託管區域中的 DNS 記錄，您必須將 VPC 與託管區域建立關聯。如需說明，請參閱 Route 53 文件中的建立私有託管區域。	網路管理員、系統管理員

設定 Route 53 Resolver 端點

任務	Description	所需的技能
建立傳入端點。	Route 53 Resolver 使用傳入端點從內部部署 DNS 解析程式接收 DNS 查詢。如需說明，請參	網路管理員、系統管理員

任務	Description	所需的技能
	閱 Route 53 文件中的轉送傳入 DNS 查詢到您的 VPCs。記下傳入端點 IP 地址。	
建立傳出端點。	Route 53 Resolver 使用傳出端點將 DNS 查詢傳送至內部部署 DNS 解析程式。如需說明，請參閱 Route 53 文件中的轉送傳出 DNS 查詢到您的網路。記下輸出端點 ID。	網路管理員、系統管理員

設定轉送規則並將其與您的 VPC 建立關聯

任務	Description	所需的技能
為內部部署網域建立轉送規則。	此規則會指示 Route 53 Resolver 將內部部署網域（例如 onprem.mydc.com) 的任何 DNS 查詢轉送至內部部署 DNS 解析程式。若要建立此規則，您需要內部部署 DNS 解析程式的 IP 地址，以及 Route 53 Resolver 的傳出端點 ID。如需說明，請參閱 Route 53 文件中的管理轉送規則。	網路管理員、系統管理員
將轉送規則與您的 VPC 建立關聯。	若要讓轉送規則生效，您必須將規則與 VPC 建立關聯。Route 53 Resolver 接著會在解析網域時考慮規則。如需說明，請參閱 Route 53 文件中的管理轉送規則。	網路管理員、系統管理員

設定內部部署 DNS 解析程式

任務	Description	所需的技能
在內部部署 DNS 解析程式中設定條件式轉送。	若要從內部部署環境將 DNS 查詢傳送至 Route 53 私有託管區域，您必須在內部部署 DNS 解析程式中設定條件式轉送。這會指示 DNS 解析程式將所有 AWS 網域的 DNS 查詢（例如 myvpc.cloud.com) 轉送至 Route 53 Resolver 的傳入端點 IP 地址。	網路管理員、系統管理員

測試end-to-end解析

任務	Description	所需的技能
測試從 AWS 到內部部署環境的 DNS 解析。	從 VPC 中的伺服器，執行內部部署網域的 DNS 查詢（例如 server1.onprem.mydc.com)。	網路管理員、系統管理員
測試從內部部署環境到 AWS 的 DNS 解析。	從內部部署伺服器，執行 AWS 網域的 DNS 解析（例如 server1.myvpc.cloud.com)。	網路管理員、系統管理員

相關資源

- 使用 Amazon Route 53 和 AWS Transit Gateway 集中式 DNS 管理混合雲端 (AWS 網路與內容交付部落格）
- 使用 Route 53 Resolver 在多帳戶環境中簡化 DNS 管理 (AWS 安全部落格）
- 使用私有託管區域 (Route 53 文件）
- Route 53 Resolver 入門 (Route 53 文件）

使用 AWS CloudFormation 在 Amazon EC2 上自動設定 UiPath RPA 機器人

Rahul Sharad Gaikwad 醫生和 Tamilselvan P，Amazon Web Services

總結

此模式說明如何在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上部署機器人程序自動化 (RPA) 機器人。它使用 EC2 Image Builder 管道來建立自訂 Amazon Machine Image (AMI)。AMI 是預先設定的虛擬機器 (VM) 映像，其中包含作業系統 (OS) 和預先安裝的軟體來部署 EC2 執行個體。此模式使用 AWS CloudFormation 範本在自訂 AMI 上安裝 UiPath Studio Community Edition。UiPath 是一種 RPA 工具，可協助您設定機器人來自動化任務。

作為此解決方案的一部分，EC2 Windows 執行個體是使用基本 AMI 啟動，且 UiPath Studio 應用程式安裝在執行個體上。模式使用 Microsoft System Preparation (Sysprep) 工具來複製自訂 Windows 安裝。之後，它會移除主機資訊，並從執行個體建立最終 AMI。然後，您可以使用最終 AMI 搭配您自己的命名慣例和監控設定，以隨需啟動執行個體。

注意：此模式不會提供有關使用 RPA 機器人的任何資訊。如需該資訊，請參閱 UiPath 文件。您也可以使用此模式，根據您的需求自訂安裝步驟來設定其他 RPA 機器人應用程式。

此模式提供下列自動化和優點：

- 應用程式部署和共用：您可以建置應用程式部署的 Amazon EC2 AMIs，並透過 EC2 Image Builder 管道跨多個帳戶共用這些 AWS CloudFormation 範本做為基礎設施做為程式碼 (IaC) 指令碼。
- Amazon EC2 佈建和擴展：CloudFormation IaC 範本提供自訂電腦名稱序列和 Active Directory 聯結自動化。
- 可觀測性和監控： 模式會設定 Amazon CloudWatch 儀表板，以協助您監控 Amazon EC2 指標 （例如 CPU 和磁碟用量）。
- 業務的 RPA 優勢：由於機器人可以自動且一致地執行指派的任務，因此 RPA 可提高準確性。RPA 也會提高速度和生產力，因為它會移除不增加價值並處理重複活動的操作。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶

- 部署 CloudFormation 範本的 AWS Identity and Access Management (IAM) 許可
- 使用 EC2 Image Builder 設定跨帳戶 AMI 分佈的 IAM 政策

Architecture

1. 管理員在 `ec2-image-builder.yaml` 檔案中提供基本 Windows AMI，並在 CloudFormation 主控台中部署堆疊。
2. CloudFormation 堆疊部署 EC2 Image Builder 管道，其中包含下列資源：
   - `Ec2ImageInfraConfiguration`
   - `Ec2ImageComponent`
   - `Ec2ImageRecipe`
   - `Ec2AMI`
3. EC2 Image Builder 管道會使用基本 AMI 啟動暫時 Windows EC2 執行個體，並安裝必要的元件（在此情況下為 UiPath Studio)。
4. EC2 Image Builder 會移除所有主機資訊，並從 Windows Server 建立 AMI。
5. 您可以使用自訂 AMI 更新 `ec2-provisioning yaml` 檔案，並根據您的需求啟動多個 EC2 執行個體。
6. 您可以使用 CloudFormation 範本部署計數巨集。此巨集提供 CloudFormation 資源的 Count 屬性，因此您可以輕鬆指定相同類型的多個資源。
7. 您可以在 CloudFormation `ec2-provisioning.yaml`檔案中更新巨集的名稱，並部署堆疊。
8. 管理員會根據需求更新 `ec2-provisioning.yaml` 檔案，並啟動堆疊。
9. 範本會使用 UiPath Studio 應用程式部署 EC2 執行個體。

工具

AWS 服務

- AWS CloudFormation 可協助您以自動化且安全的方式建立和管理基礎設施資源的模型。
- Amazon CloudWatch 可協助您觀察和監控 AWS、內部部署和其他雲端上的資源和應用程式。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供安全且可調整大小的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- EC2 Image Builder 可簡化虛擬機器和容器映像的建置、測試和部署，以用於 AWS 或內部部署。

- Amazon EventBridge 可協助您跨 AWS、現有系統或軟體即服務 (SaaS) 應用程式大規模建置事件驅動型應用程式。
- AWS Identity and Access Management (IAM) 可協助您安全地控制對 AWS 資源的存取。透過 IAM，您可以集中管理許可，以控制使用者可以存取哪些 AWS 資源。您可以使用 IAM 來控制能通過身分驗證 (登入) 和授權使用資源的 (具有許可) 的人員。
- AWS Lambda 是一種無伺服器、事件驅動的運算服務，可讓您為幾乎任何類型的應用程式或後端服務執行程式碼，而無需佈建或管理伺服器。您可以從超過 200 個 AWS 服務和 SaaS 應用程式呼叫 Lambda 函數，並且只需為您使用的項目付費。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您存放、保護和擷取任何數量的資料。
- AWS Systems Manager Agent (SSM Agent) 可協助 Systems Manager 更新、管理和設定 EC2 執行個體、邊緣裝置、內部部署伺服器和虛擬機器 VMs)。

程式碼儲存庫

此模式的程式碼可在使用 CloudFormation 儲存庫的 GitHub UiPath RPA 機器人設定中使用。 UiPath CloudFormation 模式也會使用可從 AWS CloudFormation Macros 儲存庫取得的巨集。

最佳實務

- AWS 每月都會發行新的 Windows AMIs。這些包含最新的作業系統修補程式、驅動程式和啟動代理程式。我們建議您在啟動新執行個體或建置自己的自訂映像時，使用最新的 AMI。
- 在映像建置期間套用所有可用的 Windows 或 Linux 安全修補程式。

史詩

部署基礎映像的映像管道

任務	Description	所需的技能
設定 EC2 Image Builder 管道。	1. 使用 CloudFormation 儲存庫複製 UiPath RPA 機器人設定，或從儲存庫下載`ec2-image-builder.yaml` 範本。	AWS DevOps

任務	Description	所需的技能
	2. 登入 AWS 管理主控台，然後開啟 AWS CloudFormation 主控台。 3. 選擇建立堆疊。 4. 在 Specify template (指定範本) 區段中，選擇 Upload a template file (上傳範本檔案)。 5. 從您的電腦找到並上傳`ec2-image-builder.yaml` 範本，然後選擇下一步。 6. 提供堆疊的輸入參數或接受預設值。選擇下一步。 **Note** 參數的數量和值可能會根據您的輸入值而有所不同。 7. 或者，設定堆疊選項，然後選擇下一步。 8. 檢閱您的堆疊詳細資訊。 9. 在畫面結尾，選取核取方塊以確認功能，然後選擇提交。 10. 監控堆疊的進度。當狀態為`CREATE_COMPLETE` 時，部署已就緒。	

任務	Description	所需的技能
檢視 EC2 Image Builder 設定。	EC2 Image Builder 設定包括基礎設施組態、分佈設定和安全掃描設定。若要檢視設定： 1. 開啟 EC2 Image Builder 主控台。 2. 從導覽窗格中，導覽至各種映像建置器設定。 **Note** 最佳實務是，您應該僅透過 CloudFormation 範本對 EC2 Image Builder 進行任何更新。	AWS DevOps
檢視映像管道。	若要檢視部署的影像管道： 1. 在 EC2 Image Builder 主控台上，從導覽窗格中選擇映像管道。 2. 選取您建立的映像管道。 3. 檢視輸出映像、映像配方、基礎設施組態、分佈設定、Amazon EventBridge 規則和標籤的組態詳細資訊。	AWS DevOps

任務	Description	所需的技能
檢視映像建置器日誌。	EC2 Image Builder 日誌會在 CloudWatch 日誌群組中彙總。若要在 CloudWatch 中檢視日誌： 1. 開啟 CloudWatch 主控台。 2. 在導覽窗格中依序選擇 Logs (日誌)、Log groups (日誌群組)。 3. 選擇日誌群組名稱。EC2 Image Builder 日誌會在日誌群組 中彙總`/aws/imagebuilder/XXX` 。 4. 檢查個別日誌串流中的最新日誌，是否有在執行映像管道時遇到的任何錯誤。 EC2 Image Builder 日誌也會存放在 S3 儲存貯體中。若要檢視儲存貯體中的日誌： 1. 開啟 Amazon S3 主控台。 2. 在 Buckets (儲存貯體) 清單中，選擇您的儲存貯體名稱。日誌會彙總在 S3 儲存貯體 中`<stack-name>-XXXXXX` 。	AWS DevOps

任務	Description	所需的技能
將 UiPath 檔案上傳至 S3 儲存貯體。	1. 從位置 https：//https://download.uipath.com/UiPathStudioCommunity.msi 下載 UiPath Studio 的 `.msi` 檔案。 2. 上傳 至 S3 儲存貯體。 3. 在使用者資料區段的行號 310 中，更新`ec2-image-builder.yaml` 範本中的儲存貯體名稱和檔案金鑰。	AWS DevOps

部署和測試計數巨集

任務	Description	所需的技能
部署計數巨集。	1. 複製或下載 Count CloudFormation 巨集。 2. 導覽至 `Count` 資料夾。 3. 您需要 S3 儲存貯體來存放 CloudFormation 成品。如果您還沒有 S3 儲存貯體，請使用 名稱建立一個儲存貯體`aws s3 mb s3://<bucket name>` 。 4. 封裝 Count 巨集範本。範本使用 AWS Serverless Application Model (SAM)，因此必須先進行轉換，才能進行部署。 `aws cloudformation package \`	DevOps 工程師

任務	Description	所需的技能
	--template-file template.yaml \ --s3-bucket <your bucket name here> \ --output-template-file packaged.yaml	
	例如： aws cloudformation package \ --template-file template.yaml \ --s3-bucket count-macro-ec2 \ --output-template-file packaged.yaml	
	5. 部署封裝範本以建立 CloudFormation 堆疊。 aws cloudformation deploy \ --stack-name Count-macro \ --template-file packaged.yaml \ --capabilities CAPABILITY_IAM	
	如果您想要使用 主控台，請遵循上一個 epic 或 CloudFormation 文件中的指示。	

任務	Description	所需的技能
測試計數巨集。	若要測試巨集的功能，請嘗試啟動巨集隨附的範例範本。 `aws cloudformation deploy \ --stack-name Count-test \ --template-file test.yaml \ --capabilities CAPABILITY_IAM`	DevOps 工程師

部署 CloudFormation 堆疊以使用自訂映像佈建執行個體

任務	Description	所需的技能
部署 Amazon EC2 佈建範本。	若要使用 CloudFormation 部署 EC2 映像管道： 1. 從 GitHub 儲存庫下載`ec2-provisioning.yaml` 範本，或者如果您複製儲存庫，請在電腦上找到範本。 2. 開啟 CloudFormation 主控台。 3. 重複第一個史詩中的步驟（或遵循 CloudFormation 文件中的指示）來部署 `ec2-provisioning.yaml` 。	AWS DevOps
檢視 Amazon EC2 設定。	Amazon EC2 設定包括安全性、聯網、儲存、狀態檢查、監控和標籤組態。若要檢視這些組態：	AWS DevOps

任務	Description	所需的技能
	1. 開啟 Amazon EC2 主控台。 2. 在導覽窗格中，選擇執行個體，然後選取由 Amazon EC2 佈建範本建立的 EC2 執行個體。 Amazon EC2 3. 在執行個體摘要中，選取索引標籤以檢視對應的 Amazon EC2 設定。	
檢視 CloudWatch 儀表板。	1. 開啟 CloudWatch 主控台。 2. 在導覽窗格中，選擇 Dashboards (儀表板)。 3. 選擇具有堆疊名稱的儀表板。 **Note** 佈建堆疊之後，將指標填入儀表板需要一些時間。 儀表板提供這些指標：`CPUUtilization`、`DiskUtilization`、`MemoryUtilization`、`NetworkIn`、`NetworkOut`、`StatusCheckFailed`。	AWS DevOps

任務	Description	所需的技能
檢視記憶體和磁碟用量的自訂指標。	1. 在 CloudWatch 主控台上，選擇儀表板。 2. 在導覽窗格中，選擇 Metrics (指標)、All metrics (所有指標)。 3. 選擇自訂命名空間、CWAgent。	AWS DevOps
檢視記憶體和磁碟用量的警示。	1. 在 CloudWatch 主控台的導覽窗格中，選擇儀表板。 2. 選擇 所有警示 。	AWS DevOps
驗證快照生命週期規則。	1. 開啟 Amazon EC2 主控台。 2. 在導覽窗格中，選擇 Lifecycle Manager (生命週期管理器)。 3. 驗證 AMI 生命週期的設定。	AWS DevOps

刪除環境 （選用）

任務	Description	所需的技能
刪除堆疊。	當您的 PoC 或試行專案完成時，我們建議您刪除您建立的堆疊，以確保您不會支付這些資源的費用。 1. 開啟 AWS CloudFormation 主控台。 2. 在導覽窗格中，選擇堆疊，然後選取您先前建立並要刪除的一個或兩個堆疊。此堆疊目前必須正在執行。	AWS DevOps

任務	Description	所需的技能
	3. 在 stack details (堆疊詳細資訊) 窗格中，選擇 Delete (刪除)。 4. 出現提示時，選擇 Delete stack (刪除堆疊)。 **Important** 堆疊刪除操作開始後就無法停止。堆疊繼續進行到 DELETE_IN_PROGRESS (正在刪除) 狀態。 如果刪除失敗，堆疊將處於 DELETE_FAILED 狀態。如需解決方案，請參閱 AWS CloudFormation 疑難排解文件中的刪除堆疊失敗。 如需有關保護堆疊免於意外刪除的資訊，請參閱 AWS CloudFormation 文件中的保護堆疊免於遭到刪除。	

疑難排解

問題	解決方案
部署 Amazon EC2 佈建範本時，您會收到錯誤：從轉換 123xxxx： ：Count 收到格式不正確的回應。	這是已知問題。（請參閱 AWS CloudFormation 巨集儲存庫中的自訂解決方案和 PR。）

問題	解決方案
	若要修正此問題，請開啟 AWS Lambda 主控台`index.py`，並使用 GitHub 儲存庫中的內容進行更新。

相關資源

GitHub 儲存庫

- 使用 CloudFormation 的 UiPath RPA 機器人設定
- 計數 CloudFormation 巨集

AWS 參考

- 在 AWS CloudFormation 主控台上建立堆疊 (CloudFormation 文件）
- 故障診斷 CloudFormation (CloudFormation 文件）
- 監控Amazon EC2 執行個體forAmazon記憶體和磁碟指標 (Amazon EC2 文件）
- 如何使用 CloudWatch 代理程式在 Windows 伺服器上檢視效能監控的指標？ (AWS re：Post 文章）

其他參考

- UiPath 文件
- 在 SysPreped AMI 中設定主機名稱 (Brian Beach 部落格文章）
- 當參數變更時，如何讓 Cloudformation 使用巨集重新處理範本？ （堆疊溢位）

在 AWS 上設定高度可用的 PeopleSoft 架構

Ramanathan Muralidhar，Amazon Web Services

摘要

當您將 PeopleSoft 工作負載遷移至 AWS 時，彈性是重要的目標。它可確保 PeopleSoft 應用程式始終高度可用，並能夠快速從故障中復原。

此模式為 AWS 上的 PeopleSoft 應用程式提供架構，以確保網路、應用程式和資料庫層的高可用性 (HA)。它針對資料庫層使用 Amazon Relational Database Service (Amazon RDS) for Oracle 或 Amazon RDS for SQL Server 資料庫。此架構也包含 AWS 服務，例如 Amazon Route 53、Amazon Elastic Compute Cloud (Amazon EC2) Linux 執行個體、Amazon Elastic Block Storage (Amazon EBS)、Amazon Elastic File System (Amazon EFS) 和 Application Load Balancer，而且可擴展。

Oracle PeopleSoft 為人力資源管理和其他業務營運提供一套工具和應用程式。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 具有在 AWS 上設定的必要授權的 PeopleSoft 環境
- 在您的 AWS 帳戶中設定具有下列資源的虛擬私有雲端 (VPC)：
  - 至少兩個可用區域
  - 每個可用區域中一個公有子網路和三個私有子網路
  - NAT 閘道和網際網路閘道
  - 每個子網路路由表以路由流量
  - 網路存取控制清單 （ 網路 ACLs) 和定義的安全群組，可協助根據組織標準確保 PeopleSoft 應用程式的安全性

限制

- 此模式提供高可用性 (HA) 解決方案。它不支援災難復原 (DR) 案例。在 HA 實作的整個 AWS 區域極少停止運作的情況下，應用程式將無法使用。

產品版本

- 執行 PeopleTools 8.52 及更新版本的 PeopleSoft PeopleSoft 應用程式

Architecture

目標架構

PeopleSoft 生產應用程式的停機時間或中斷會影響應用程式的可用性，並對您的業務造成重大中斷。

我們建議您設計 PeopleSoft 生產應用程式，使其始終具有高可用性。您可以透過消除單一故障點、新增可靠的交叉或容錯移轉點，以及偵測故障來達成此目的。下圖說明 AWS 上 PeopleSoft 的 HA 架構。

此架構部署使用 Amazon RDS for Oracle 做為 PeopleSoft 資料庫，以及在 Red Hat Enterprise Linux (RHEL) 上執行的 EC2 執行個體。您也可以使用 Amazon RDS for SQL Server 做為 Peoplesoft 資料庫。

此架構包含下列元件：

- Amazon Route 53 用作網域名稱伺服器 (DNS)，用於將請求從網際網路路由到 PeopleSoft 應用程式。
- AWS WAF 可協助您防範可能影響可用性、危及安全性或消耗過多資源的常見 Web 入侵和機器人。AWS Shield Advanced （未說明）提供更廣泛的保護。
- Application Load Balancer 使用以 Web 伺服器為目標的進階請求路由來平衡 HTTP 和 HTTPS 流量。
- 支援 PeopleSoft 應用程式在多個可用區域中執行並使用 Amazon EC2 Auto Scaling 的 Web 伺服器、應用程式伺服器、程序排程器伺服器和 Elasticsearch 伺服器。
- PeopleSoft 應用程式使用的資料庫會在多可用區組態的 Amazon RDS 上執行。
- PeopleSoft 應用程式使用的檔案共用是在 Amazon EFS 上設定，用於跨執行個體存取檔案。
- Amazon EC2 Auto Scaling 會使用 Amazon Machine Image (AMI)，以確保在需要時快速複製 PeopleSoft 元件。 Amazon EC2 Auto Scaling
- NAT 閘道會將私有子網路中的執行個體連線至 VPC 外部的服務，並確保外部服務無法啟動與這些執行個體的連線。
- 網際網路閘道是一種水平擴展、備援且高可用性的 VPC 元件，可讓您的 VPC 與網際網路之間進行通訊。
- 公有子網路中的堡壘主機可讓您從外部網路存取私有子網路中的伺服器，例如網際網路或內部部署網路。堡壘主機提供私有子網路中伺服器的受控制且安全存取。

架構詳細資訊

PeopleSoft 資料庫存放在多可用區組態中的 Amazon RDS for Oracle （或 Amazon RDS for SQL Server) 資料庫中。Amazon RDS 多可用區域功能會跨兩個可用區域複寫資料庫更新，以提高耐用性和可用性。Amazon RDS 會自動容錯移轉至待命資料庫，以進行計劃維護和計劃外中斷。

PeopleSoft Web 和中間層安裝在 EC2 執行個體上。這些執行個體分散在多個可用區域，並由 Auto Scaling 群組繫結。這可確保這些元件始終具有高可用性。維護最低數量的必要執行個體，以確保應用程式隨時可用，並可在需要時進行擴展。

建議您針對 OEM EC2 執行個體使用最新一代的 EC2 執行個體類型。目前世代的執行個體類型，例如建置在 AWS Nitro 系統的執行個體，支援硬體虛擬機器 HVMs)。HVM AMIs 需要利用增強型聯網，而且也提供更高的安全性。屬於每個 Auto Scaling 群組的 EC2 執行個體會在取代或擴展執行個體時使用自己的 AMI。我們建議您根據希望 PeopleSoft 應用程式處理的負載，以及 Oracle 針對 PeopleSoft 應用程式和 PeopleTools 版本建議的最小值，選取 EC2 執行個體類型。如需硬體和軟體需求的詳細資訊，請參閱 Oracle 支援網站。

PeopleSoft Web 和中層共用 Amazon EFS 掛載，以共用報告、資料檔案和 （如有需要） `PS_HOME`目錄。基於效能和成本考量，Amazon EFS 會在每個可用區域中設定掛載目標。

Application Load Balancer 會佈建為支援存取 PeopleSoft 應用程式和負載平衡不同可用區域中 Web 伺服器之間流量的流量。Application Load Balancer 是一種網路裝置，可在至少兩個可用區域中提供 HA。Web 伺服器會使用負載平衡組態，將流量分配到不同的應用程式伺服器。Web 伺服器和應用程式伺服器之間的負載平衡可確保負載平均分散到執行個體，並協助避免因執行個體過載而造成瓶頸和服務中斷。

Amazon Route 53 用作 DNS 服務，將流量從網際網路路由到 Application Load Balancer。Route 53 是一種可用性高、可擴展性強的 DNS Web 服務。

HA 詳細資訊

- 資料庫：Amazon RDS 的異地同步複寫功能會在多個可用區域中操作兩個資料庫。這會建立具有自動容錯移轉的高可用性環境。Amazon RDS 具有容錯移轉事件偵測，並在發生這些事件時啟動自動容錯移轉。您也可以透過 Amazon RDS API 啟動手動容錯移轉。如需詳細說明，請參閱部落格文章 Amazon RDS under the Hood： Multi-AZ。容錯移轉是無縫的，應用程式會在發生時自動重新連線至資料庫。不過，容錯移轉期間的任何程序排程器任務都會產生錯誤，且必須重新提交。
- PeopleSoft 應用程式伺服器：應用程式伺服器分散在多個可用區域，並為其定義 Auto Scaling 群組。如果執行個體失敗，Auto Scaling 群組會立即將其取代為從應用程式伺服器範本的 AMI 複製的運作狀態良好的執行個體。具體而言，啟用震動集區，因此當應用程式伺服器執行個體關閉時，工作階段會自動容錯移轉至另一個應用程式伺服器，Auto Scaling 群組會自動啟動另一個執行個體、啟動

應用程式伺服器，並在 Amazon EFS 掛載中註冊。新建立的應用程式伺服器會使用 Web 伺服器中的PSSTRSETUP.SH指令碼，自動新增至 Web 伺服器。這可確保應用程式伺服器始終高度可用，並快速從故障中復原。

- 程序排程器：程序排程器伺服器分散在多個可用區域，並為其定義 Auto Scaling 群組。如果執行個體失敗，Auto Scaling 群組會立即將其取代為從程序排程器伺服器範本的 AMI 複製的運作狀態良好的執行個體。具體而言，當程序排程器執行個體關閉時，Auto Scaling 群組會自動啟動另一個執行個體並啟動程序排程器。執行個體失敗時正在執行的任何任務都必須重新提交。這可確保程序排程器始終高度可用，並快速從故障中復原。
- Elasticsearch 伺服器：Elasticsearch 伺服器具有為其定義的 Auto Scaling 群組。如果執行個體失敗，Auto Scaling 群組會立即將其取代為從 Elasticsearch 伺服器範本的 AMI 複製的運作狀態良好的執行個體。具體而言，當 Elasticsearch 執行個體故障時，向 提供請求的 Application Load Balancer 會偵測故障並停止傳送流量給該執行個體。Auto Scaling 群組會自動啟動另一個執行個體，並啟動 Elasticsearch 執行個體。當 Elasticsearch 執行個體備份時，Application Load Balancer 會偵測到運作狀態良好，並再次開始向其傳送請求。這可確保 Elasticsearch 伺服器始終高度可用，並快速從故障中復原。
- Web 伺服器：Web 伺服器具有為其定義的 Auto Scaling 群組。如果執行個體失敗，Auto Scaling 群組會立即將其取代為從 Web 伺服器範本的 AMI 複製的運作狀態良好的執行個體。具體而言，當 Web 伺服器執行個體故障時，提供請求給它的 Application Load Balancer 會偵測故障並停止傳送流量給它。Auto Scaling 群組會自動啟動另一個執行個體，並啟動 Web 伺服器執行個體。當 Web 伺服器執行個體備份時，Application Load Balancer 會偵測到運作狀態良好，並再次開始傳送請求。這可確保 Web 伺服器始終高度可用，並快速從故障中復原。

工具

AWS 服務

- Application Load Balancer 會將傳入的應用程式流量分散到多個可用區域中的多個目標，例如 EC2 執行個體。
- Amazon Elastic Block Store (Amazon EBS) 提供區塊層級儲存磁碟區，可與 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體搭配使用。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon Elastic File System (Amazon EFS) 可協助您在 AWS 雲端中建立和設定共用檔案系統。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。

- Amazon Route 53 是一種可用性高、可擴展性強的 DNS Web 服務。

最佳實務

操作最佳實務

- 當您在 AWS 上執行 PeopleSoft 時，請使用 Route 53 從網際網路和本機路由流量。如果主要資料庫執行個體無法使用，請使用容錯移轉選項將流量重新路由至災難復原 (DR) 網站。
- 一律在 PeopleSoft 環境前使用 Application Load Balancer。這可確保流量以安全的方式負載平衡至 Web 伺服器。
- 在 Application Load Balancer 目標群組設定中，請確定已使用負載平衡器產生的 Cookie 開啟黏性。

> **Note**
>
> 如果您使用外部單一登入 (SSO)，您可能需要使用應用程式型 Cookie。這可確保跨 Web 伺服器和應用程式伺服器的連線一致。

- 對於 PeopleSoft 生產應用程式，Application Load Balancer 閒置逾時必須符合您使用的 Web 設定檔中設定的內容。這可防止使用者工作階段在負載平衡器層過期。
- 對於 PeopleSoft 生產應用程式，將應用程式伺服器資源回收計數設定為可將記憶體流失降至最低的值。
- 如果您針對 PeopleSoft 生產應用程式使用 Amazon RDS 資料庫，如此模式所述，請以多可用區域格式執行資料庫，以獲得高可用性。
- 如果您的資料庫在 PeopleSoft 生產應用程式的 EC2 執行個體上執行，請確保待命資料庫在另一個可用區域上執行，以實現高可用性。
- 對於 DR，請確定您的 Amazon RDS 資料庫或 EC2 執行個體已在生產資料庫不同的 AWS 區域中設定待命。這可確保在 區域中發生災難時，您可以將應用程式切換到另一個區域。
- 對於 DR，使用 Amazon Elastic Disaster Recovery 在與生產元件不同的區域中設定應用程式層級元件。這可確保在 區域中發生災難時，您可以將應用程式切換到另一個區域。
- 使用 Amazon EFS（適用於中等 I/O 需求）或 Amazon FSx（適用於高 I/O 需求）來存放 PeopleSoft 報告、附件和資料檔案。這可確保內容存放在一個中央位置，並且可以從基礎設施中的任何位置存取。
- 使用 Amazon CloudWatch（基本和詳細）近乎即時地監控 PeopleSoft 應用程式正在使用的 AWS 雲端資源。這可確保您立即收到問題提醒，並在問題影響環境可用性之前快速解決這些問題。

- 如果您使用 Amazon RDS 資料庫做為 PeopleSoft 資料庫，請使用增強型監控。此功能可讓您存取超過 50 個指標，包括 CPU、記憶體、檔案系統 I/O 和磁碟 I/O。
- 使用 AWS CloudTrail 來監控 PeopleSoft 應用程式正在使用的 AWS 資源上的 API 呼叫。這可協助您執行安全分析、資源變更追蹤和合規稽核。

安全最佳實務

- 若要保護您的 PeopleSoft 應用程式免於 SQL Injection 或跨網站指令碼 (XSS) 等常見漏洞，請使用 AWS WAF。請考慮使用 AWS Shield Advanced 進行量身打造的偵測和緩解服務。
- 將規則新增至 Application Load Balancer，以自動將流量從 HTTP 重新導向至 HTTPS，以協助保護 PeopleSoft 應用程式。
- 為 Application Load Balancer 設定個別的安全群組。此安全群組應僅允許 HTTPS/HTTP 傳入流量，且不允許傳出流量。這可確保只允許預期的流量，並有助於保護您的應用程式。
- 將私有子網路用於應用程式伺服器、Web 伺服器和資料庫，並將 NAT 閘道用於傳出網際網路流量。這可確保支援應用程式的伺服器無法公開連線，同時僅向需要它的伺服器提供公開存取。
- 使用不同的 VPCs來執行 PeopleSoft 生產和非生產環境。使用 AWS Transit Gateway、VPC 對等互連、網路 ACLs 和安全群組來控制 VPC 和內部部署資料中心之間的流量。
- 遵循最低權限原則。僅將 PeopleSoft 應用程式使用的 AWS 資源存取權授予絕對需要它的使用者。僅授予執行任務所需的最低權限。如需詳細資訊，請參閱 AWS Well-Architected Framework 的安全支柱。
- 盡可能使用 AWS Systems Manager 來存取 PeopleSoft 應用程式使用的 EC2 執行個體。

可靠性最佳實務

- 當您使用 Application Load Balancer 時，請為每個啟用的可用區域註冊單一目標。這可讓負載平衡器更有效率。
- 我們建議您為每個 PeopleSoft 生產環境提供三個不同的 URLs：一個用於存取應用程式的 URL、一個用於服務整合代理程式，另一個用於檢視報告。如果可能，每個 URL 都應有自己的專用 Web 伺服器和應用程式伺服器。此設計有助於讓您的 PeopleSoft 應用程式更安全，因為每個 URL 都有不同的功能和受控制的存取。如果基礎服務失敗，也會將影響範圍降至最低。
- 建議您為 PeopleSoft 應用程式設定負載平衡器目標群組的運作狀態檢查。運作狀態檢查應該在 Web 伺服器上執行，而不是執行這些伺服器的 EC2 執行個體。這可確保如果 Web 伺服器當機或託管 Web 伺服器的 EC2 執行個體當機，Application Load Balancer 會準確反映該資訊。

- 對於 PeopleSoft 生產應用程式，我們建議您將 Web 伺服器分散到至少三個可用區域。這可確保即使其中一個可用區域故障，PeopleSoft 應用程式始終處於高可用性狀態。
- 對於 PeopleSoft 生產應用程式，啟用 jolt 集區 (`joltPooling=true`)。這可確保如果伺服器因修補或 VM 故障而停機，您的應用程式會容錯移轉至另一個應用程式伺服器。
- 對於 PeopleSoft 生產應用程式，將 `DynamicConfigReload` 設定為 1。PeopleTools 8.52 版及更新版本支援此設定。它會動態地將新的應用程式伺服器新增至 Web 伺服器，而不會重新啟動伺服器。
- 若要在套用 PeopleTools 修補程式時將停機時間降至最低，請針對 Web 和應用程式伺服器的 Auto Scaling 群組啟動組態使用藍/綠部署方法。如需詳細資訊，請參閱 AWS 上的部署選項概觀白皮書。
- 使用 AWS Backup 在 AWS 上備份 PeopleSoft 應用程式。AWS Backup 是符合成本效益、全受管、以政策為基礎的服務，能夠大規模簡化資料保護程序。

效能最佳實務

- 終止 Application Load Balancer 的 SSL 以獲得最佳的 PeopleSoft 環境效能，除非您的業務需要整個環境的加密流量。
- 為 Amazon Simple Notification Service (Amazon SNS) 和 CloudWatch 等 AWS 服務建立介面 VPC 端點，讓流量一律位於內部。這符合成本效益，有助於保護您的應用程式安全。

成本最佳化最佳實務

- 標記 PeopleSoft 環境使用的所有資源，並啟用成本分配標籤。這些標籤可協助您檢視和管理資源成本。
- 對於 PeopleSoft 生產應用程式，請為 Web 伺服器和應用程式伺服器設定 Auto Scaling 群組。這可維持最少數量的 Web 和應用程式伺服器，以支援您的應用程式。您可以使用 Auto Scaling 群組政策，視需要向上和向下擴展伺服器。
- 使用帳單警示，在成本超過您指定的預算閾值時收到提醒。

永續性最佳實務

- 使用基礎設施做為程式碼 (IaC) 來維護 PeopleSoft 環境。這可協助您建立一致的環境並維持變更控制。

史詩

將您的 PeopleSoft 資料庫遷移至 Amazon RDS

任務	Description	所需的技能
建立資料庫子網路群組。	在 Amazon RDS 主控台的導覽窗格中，選擇子網路群組，然後在多個可用區域中建立子網路為 的 Amazon RDS 資料庫子網路群組。這是 Amazon RDS 資料庫在多可用區組態中執行的必要項目。	雲端管理員
建立 Amazon RDS 資料庫。	在您為 PeopleSoft HA 環境選取的 AWS 區域的可用區域中建立 Amazon RDS 資料庫。當您建立 Amazon RDS 資料庫時，請務必選取異地同步備份選項 (建立待命執行個體) 和您在上一個步驟中建立的資料庫子網路群組。如需詳細資訊，請參閱 Amazon RDS 文件。	雲端管理員、Oracle 資料庫管理員
將您的 PeopleSoft 資料庫遷移至 Amazon RDS。	使用 AWS Database Migration Service (AWS DMS) 將現有的 PeopleSoft 資料庫遷移至 Amazon RDS 資料庫。如需詳細資訊，請參閱 AWS DMS 文件和 AWS 部落格文章使用 AWS DMS 在接近零停機時間的情況下遷移 Oracle 資料庫。	雲端管理員，PeopleSoft DBA

設定您的 Amazon EFS 檔案系統

任務	Description	所需的技能
建立檔案系統。	在 Amazon EFS 主控台上，為每個可用區域建立檔案系統和掛載目標。如需說明，請參閱 Amazon EFS 文件。建立檔案系統後，請記下其 DNS 名稱。當您掛載檔案系統時，將使用此資訊。	雲端管理員

設定您的 PeopleSoft 應用程式和檔案系統

任務	Description	所需的技能
啟動 EC2 執行個體。	為您的 PeopleSoft 應用程式啟動 EC2 執行個體。如需說明，請參閱 Amazon EC2 文件。 • 對於名稱，輸入 APP_TEMPLATE 。 • 針對作業系統映像，選擇 Red Hat。 • 針對執行個體類型，選擇適合您 PeopleSoft 應用程式的執行個體類型。如需詳細資訊，請參閱架構區段中的架構詳細資訊。 ???	雲端管理員、PeopleSoft 管理員
在執行個體上安裝 PeopleSoft。	在您建立的 EC2 執行個體上安裝 PeopleSoft 應用程式和 PeopleTools。如需說明，請參閱 Oracle 文件。	雲端管理員、PeopleSoft 管理員

任務	Description	所需的技能
建立應用程式伺服器。	為 AMI 範本建立應用程式伺服器，並確保其成功連線至 Amazon RDS 資料庫。	雲端管理員、PeopleSoft 管理員

掛載 Amazon EFS 檔案系統。	以根使用者身分登入 EC2 執行個體，並執行下列命令，將 Amazon EFS 檔案系統掛載到伺服器上名為 PSFTMNT的資料夾。
	`sudo su -`
	`mkdir /psftmnt`
	`cat /etc/fstab`
	將以下行附加至 `/etc/fstab` 檔案。使用您在建立檔案系統時記下的 DNS 名稱。
	`fs-09e064308f1145388.efs.us-east-1.amazonaws.com:/ /psftmnt nfs4 nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport,_netdev 0 0`
	`mount -a`
		雲端管理員、PeopleSoft 管理員

檢查許可。	請確定PSFTMNT資料夾具有適當的許可，以便 PeopleSoft 使用者可以正確存取。	雲端管理員、PeopleSoft 管理員

任務	Description	所需的技能
建立其他執行個體。	重複上述步驟，為程序排程器、Web 伺服器和 Elasticsearch 伺服器建立範本執行個體。為這些執行個體命名 PRCS_TEMPLATE 、 WEB_TEMPLATE 和 SRCH_TEMPLATE 。針對 Web 伺服器，設定joltPooling=true 和 DynamicConfigReloa d=1 。	雲端管理員、PeopleSoft 管理員

建立指令碼以設定伺服器

任務	Description	所需的技能

建立指令碼以安裝應用程式伺服器。	在 Amazon EC2 APP_TEMPLATE 執行個體中，以 PeopleSoft 使用者身分建立下列指令碼。將其命名appstart.sh 並放在PS_HOME目錄中。您將使用此指令碼來啟動應用程式伺服器，並在 Amazon EFS 掛載上記錄伺服器名稱。
	
	`#!/bin/ksh`
	`. /usr/homes/hcmdemo /.profile.`
	`psadmin -c configure -d HCMDEMO`
	`psadmin -c parallelb oot -d HCMDEMO`
	`touch /psftmnt/`echo $HOSTNAME`
		PeopleSoft 管理員

任務	Description	所需的技能

建立指令碼以安裝程序排程器伺服器。	在 Amazon EC2 PRCS_TEMPLATE 執行個體中，以 PeopleSoft 使用者身分建立下列指令碼。將其命名`prcsstart.sh` 並放在 PS_HOME目錄中。您將使用此指令碼來叫用程序排程器伺服器。
	
	`#!/bin/ksh`
	`. /usr/homes/hcmdemo/.profile`
	`/* The following line ensures that the process scheduler always has a unique name during replacement or scaling activity. */`
	`sed -i "s/.*PrcsServerName.*/`hostname -I | awk -F. '{print "PrcsServerName=PSUNX"$3$4}'`/" $HOME/appserv/prcs/*/psprcs.cfg`
	`psadmin -p configure -d HCMDEMO`
	`psadmin -p start -d HCMDEMO`
		PeopleSoft 管理員

任務	Description	所需的技能

建立指令碼以安裝 Elasticsearch 伺服器。	在 Amazon EC2 SRCH_TEMPLATE 執行個體中，以 Elasticsearch 使用者身分建立下列指令碼。將其命名`srchstart.sh` 並放在 HOME目錄中。
	
	`#!/bin/ksh`
	`/* The following line ensures that the correct IP is indicated in the elasticsearch.yaml file. */`
	`sed -i "s/.*network.host.*/`hostname -I
		awk '{print "host:"$0}'`/" $ES_HOME_DIR/config/elasticsearch.yaml`
		`nohup $ES_HOME_DIR/bin/elasticsearch &`
			PeopleSoft 管理員

任務	Description	所需的技能
建立指令碼以安裝 Web 伺服器。	在 Amazon EC2 WEB_TEMPLATE 執行個體中，以 Web 伺服器使用者身分，在 HOME目錄中建立下列指令碼。 renip.sh：此指令碼可確保 Web 伺服器在從 AMI 複製時擁有正確的 IP。 `#!/bin/ksh` `hn=`hostname`` `/* On the following` `line, change the IP` `with the hostname with` `the hostname of the web` `template. */` `for text_file in `find` `* -type f -exec grep` `-l '<hostname-of-the-` `web-template>' {} \;` `do` `sed -e 's/<hostn` `ame-of-the-web-tem` `plate>/'$hn'/g'` `$text_file > temp` `mv -f temp $text_file` `done`  psstrsetup.sh ：此指令碼可確保 Web 伺服器使用目前正在執行的正確應用程式伺服器 IPs。它會嘗試連線到 jolt 連接埠上的每個應用程式伺服器，並將其新增至組態檔案。  `#!/bin/ksh` `c2=""`	PeopleSoft 管理員

任務	Description	所需的技能

	<pre>for ctr in `ls -1 /psftmnt/*.internal`
	do
	c1=`echo $ctr | awk -F "/" '{print $3}'`
	/* In the following lines, 9000 is the jolt port. Change it if necessary. */
	if nc -z $c1 9000 2> /dev/null; then
	if [[$c2 = ""]]; then
	c2="psserver="`echo $c1`":9000"
	else
	c2=`echo $c2`","`echo $c1`":9000"
	fi
	fi
	done</pre>
	webstart.sh ：此指令碼會執行先前的兩個指令碼，並啟動 Web 伺服器。
	<pre>#!/bin/ksh
	/* Change the path in the following if necessary. */
	cd /usr/homes/hcmdemo
	./renip.sh
	./psstrsetup.sh
	webserv/peoplesoft/bin/startPIA.sh</pre>	

任務	Description	所需的技能
新增 crontab 項目。	在 Amazon EC2 WEB_TEMPLATE 執行個體中，作為 Web 伺服器使用者，將以下行新增至 crontab。變更時間和路徑，以反映您需要的值。此項目可確保 Web 伺服器在 configuration.properties 檔案中一律具有正確的應用程式伺服器項目。 `* * * * * /usr/homes/hcmdemo/psstrsetup.sh`	PeopleSoft 管理員

建立 AMIs和 Auto Scaling 群組範本

任務	Description	所需的技能
為應用程式伺服器範本建立 AMI。	在 Amazon EC2 主控台上，建立 Amazon EC2 APP_TEMPLATE 執行個體的 AMI 映像。將 AMI 命名為 PSAPPSRV-SCG-VER1 。如需說明，請參閱 Amazon EC2 文件。	雲端管理員、PeopleSoft 管理員
為其他伺服器建立 AMIs。	重複上一個步驟，為程序排程器、Elasticsearch 伺服器和 Web 伺服器建立 AMIs。	雲端管理員、PeopleSoft 管理員
為應用程式伺服器 Auto Scaling 群組建立啟動範本。	為應用程式伺服器 Auto Scaling 群組建立啟動範本。為範本命名 PSAPPSRV_TEMPLATE。 在範本中，選擇您為APP_TEMPLATE 執行	雲端管理員、PeopleSoft 管理員

任務	Description	所需的技能

	個體建立的 AMI。如需說明，請參閱 Amazon EC2 文件。
	• 在啟動範本中，根據您的需求選取執行個體類型。
	• 在進階詳細資訊區段的使用者資料欄位中，新增下列項目。請確定路徑和使用者資訊正確無誤。您在上一個步驟中建立`appstart.sh` 指令碼。
	`#! /bin/ksh`
	`su -c "/usr/homes/hcmdemo/appstart.sh" - hcmdemo`	

為程序排程器伺服器 Auto Scaling 群組建立啟動範本。	重複上一個步驟，為程序排程器伺服器 Auto Scaling 群組建立啟動範本。為範本命名 `PSPRCS_TEMPLATE` 。在範本中，選擇您為程序排程器建立的 AMI。
	• 在進階詳細資訊區段的使用者資料欄位中，新增下列項目。請確定路徑和使用者資訊正確無誤。您在上一個步驟中建立`prcsstart.sh` 指令碼。
	`#! /bin/ksh`
	`su -c "/usr/homes/hcmdemo/prcsstart.sh" - hcmdemo`
		雲端管理員、PeopleSoft 管理員

任務	Description	所需的技能

為 Elasticsearch 伺服器 Auto Scaling 群組建立啟動範本。	重複上述步驟，為 Elasticsearch 伺服器 Auto Scaling 群組建立啟動範本。為範本命名 SRCH_TEMPLATE 。在範本中，選擇您為搜尋伺服器建立的 AMI。
	• 在進階詳細資訊區段的使用者資料欄位中，新增下列項目。請確定路徑和使用者資訊正確無誤。您在上一個步驟中建立srchstart.sh 指令碼。
	`#! /bin/ksh`
	`su -c "/usr/homes/essearch/srchstart.sh" - essearch`
		雲端管理員、PeopleSoft 管理員

任務	Description	所需的技能

為 Web 伺服器 Auto Scaling 群組建立啟動範本。	重複上述步驟，為 Web 伺服器 Auto Scaling 群組建立啟動範本。為範本命名 `WEB_TEMPLATE` 。在範本中，選擇您為 Web 伺服器建立的 AMI。
	• 在進階詳細資訊區段的使用者資料欄位中，新增下列項目。請確定路徑和使用者資訊正確無誤。您在上一個步驟中建立`webstart.sh` 指令碼。
	`#! /bin/ksh`
	`su -c "/usr/homes/hcmdemo/webstart.sh" - hcmdemo`
		雲端管理員、PeopleSoft 管理員

建立 Auto Scaling 群組

任務	Description	所需的技能
為應用程式伺服器建立 Auto Scaling 群組。	在 Amazon EC2 主控台上，使用 `PSAPPSRV_TEMPLATE` 範本建立`PSAPPSRV_ASG` 名為 應用程式伺服器的 Auto Scaling 群組。如需說明，請參閱 Amazon EC2 文件。 • 在選擇執行個體啟動選項頁面上，選取正確的 VPC，然後從不同的可用區域選取多個子網路。 • 在設定進階選項頁面上，請勿選取負載平衡器。	雲端管理員、PeopleSoft 管理員

任務	Description	所需的技能
	• 在設定群組大小和擴展政策頁面上，根據您要架構系統的負載量，以及是否要使用擴展政策，選擇設定。我們建議您將所需容量和最小容量設定為至少 2，以便至少有一個執行個體可在任何時間點為流量提供服務。如需 Auto Scaling 政策的詳細資訊，請參閱 Amazon EC2 文件。	
為其他伺服器建立 Auto Scaling 群組。	重複上述步驟，為程序排程器、Elasticsearch 伺服器和 Web 伺服器建立 Auto Scaling 群組。	雲端管理員、PeopleSoft 管理員

建立和設定目標群組

任務	Description	所需的技能
為 Web 伺服器建立目標群組。	在 Amazon EC2 主控台上，為 Web 伺服器建立目標群組。如需說明，請參閱 Elastic Load Balancing 文件。將連接埠設定為 Web 伺服器接聽的連接埠。	雲端管理員
設定運作狀態檢查。	確認運作狀態檢查具有正確的值，以反映您的業務需求。如需詳細資訊，請參閱 Elastic Load Balancing 說明文件。	雲端管理員
為 Elasticsearch 伺服器建立目標群組。	重複上述步驟，為 Elasticsearch 伺服器建立名為 PSFTSRCH 的目標群組，並設	雲端管理員

任務	Description	所需的技能
	定正確的 Elasticsearch 連接埠。	
將目標群組新增至 Auto Scaling 群組。	開啟您先前建立PSPIA_ASG 的 Web 伺服器 Auto Scaling 群組。在負載平衡索引標籤上，選擇編輯，然後將PSFTWEB目標群組新增至 Auto Scaling 群組。 針對 Elasticsearch Auto Scaling 群組重複此步驟PSSRCH_ASG ，以新增PSFTSRCH您先前建立的目標群組。	雲端管理員
設定工作階段黏性。	在目標群組 中PSFTWEB，選擇屬性索引標籤，選擇編輯，然後設定工作階段黏性。針對黏性類型，選擇負載平衡器產生的 Cookie，並將持續時間設定為 1。如需詳細資訊，請參閱 Elastic Load Balancing 說明文件。 針對目標群組 重複此步驟PSFTSRCH。	雲端管理員

建立和設定應用程式負載平衡器

任務	Description	所需的技能
為 Web 伺服器建立負載平衡器。	建立名為 的 Application Load Balancer，PSFTLB以將流量負載平衡至 Web 伺服器。如	雲端管理員

任務	Description	所需的技能
	需說明，請參閱 Elastic Load Balancing 文件。 • 提供負載平衡器名稱。 • 對於 Scheme (結構描述)，選擇 Internet-facing (面向網際網路)。 • 在網路映射區段中，從不同的可用區域選取正確的 VPC 和至少兩個公有子網路。 • 在接聽程式和路由區段中，選取目標群組，PSFTWEB並指定正確的通訊協定和連接埠號碼。	
為 Elasticsearch 伺服器建立負載平衡器。	建立名為 的 Application Load BalancerPSFTSCH，以負載平衡傳送至 Elasticsearch 伺服器的流量。 • 提供負載平衡器名稱。 • 針對結構描述，選擇內部。 • 在網路映射區段中，選取正確的 VPC 和私有子網路。 • 在接聽程式和路由區段中，選取目標群組，PSFTSRCH並指定正確的通訊協定和連接埠號碼。	雲端管理員

任務	Description	所需的技能
設定 Route 53。	在 Amazon Route 53 主控台上，在託管區域中建立記錄，以服務 PeopleSoft 應用程式。如需說明，請參閱 Amazon Route 53 文件。這可確保所有流量通過PSFTLB負載平衡器。	雲端管理員

相關資源

- Oracle PeopleSoft 網站
- AWS 文件

使用 AWS Elastic Disaster Recovery 為 Oracle JD Edwards EnterpriseOne 設定災難復原

Thanigaivel Thirumalai，Amazon Web Services

摘要

由自然災難、應用程式故障或服務中斷觸發的災難會損害收入 ，並導致企業應用程式停機。為了減少此類事件的後果，災難復原 (DR) 的規劃對於採用 JD Edwards EnterpriseOne 企業資源規劃 (ERP) 系統和其他關鍵任務和業務軟體的公司至關重要。

此模式說明企業如何使用 AWS Elastic Disaster Recovery 作為其 JD Edwards EnterpriseOne 應用程式的 DR 選項。 它還概述了使用 Elastic Disaster Recovery 容錯移轉和容錯恢復來為 AWS 雲端中託管的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的資料庫建構跨區域 DR 策略的步驟。

> **Note**
>
> 此模式需要在 AWS 上託管跨區域 DR 實作的主要和次要區域。

Oracle JD Edwards EnterpriseOne 是整合式的 ERP 軟體解決方案，適用於各種產業中的中型到大型公司。

AWS Elastic Disaster Recovery 使用經濟實惠的儲存、最小的運算和point-in-time復原，透過快速、可靠的內部部署和雲端應用程式復原，將停機時間和資料遺失降至最低。

AWS 提供四個核心 DR 架構模式。本文件著重於使用指示燈策略的設定、組態和最佳化。此策略可協助您建立成本較低的 DR 環境，其中您最初佈建複寫伺服器以從來源資料庫複寫資料，而且只有在您啟動 DR 演練和復原時才佈建實際的資料庫伺服器。此策略可免除在 DR 區域中維護資料庫伺服器的費用。反之，您需要為做為複寫伺服器的小型 EC2 執行個體付費。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在 Oracle 資料庫或 Microsoft SQL Server 上執行的 JD Edwards EnterpriseOne 應用程式，其受管 EC2 執行個體上的支援資料庫處於執行狀態。此應用程式應包含安裝在一個 AWS 區域的所有 JD Edwards EnterpriseOne 基本元件 (Enterprise Server、HTML Server 和 Database Server)。

- 用於設定 Elastic Disaster Recovery 服務的 AWS Identity and Access Management (IAM) 角色。
- 根據所需的連線設定來執行 Elastic Disaster Recovery 的網路。

限制

- 您可以使用此模式複寫所有層，除非資料庫託管在 Amazon Relational Database Service (Amazon RDS) 上，在這種情況下，我們建議您使用 Amazon RDS 的跨區域複製功能。
- Elastic Disaster Recovery 與 CloudEndure Disaster Recovery 不相容，但您可以從 CloudEndure Disaster Recovery 升級。如需詳細資訊，請參閱 Elastic Disaster Recovery 文件中的常見問答集。
- Amazon Elastic Block Store (Amazon EBS) 會限制您可以拍攝快照的速率。您可以使用 Elastic Disaster Recovery 在單一 AWS 帳戶中複寫最多 300 個伺服器。若要複寫更多伺服器，您可以使用多個 AWS 帳戶或多個目標 AWS 區域。（您必須為每個帳戶和區域分別設定彈性災難復原。）如需詳細資訊，請參閱 Elastic Disaster Recovery 文件中的最佳實務。
- 來源工作負載 (JD Edwards EnterpriseOne 應用程式和資料庫） 必須託管在 EC2 執行個體上。此模式不支援內部部署或其他雲端環境中的工作負載。
- 此模式著重於 JD Edwards EnterpriseOne 元件。完整的 DR 和業務持續性計劃 (BCP) 應包含其他核心服務，包括：
  - 網路 （虛擬私有雲端、子網路和安全群組）
  - Active Directory
  - Amazon WorkSpaces
  - Elastic Load Balancing
  - 受管資料庫服務，例如 Amazon Relational Database Service (Amazon RDS)

如需先決條件、組態和限制的其他資訊，請參閱彈性災難復原文件。

產品版本

- Oracle JD Edwards EnterpriseOne （根據 Oracle 最低技術需求，Oracle 和 SQL Server 支援的版本）

Architecture

目標技術堆疊

- 生產和非生產的單一區域和單一虛擬私有雲端 (VPC)，以及 DR 的第二個區域

- 單一可用區域，以確保伺服器之間的低延遲
- Application Load Balancer 可分配網路流量，以提高應用程式跨多個可用區域的可擴展性和可用性
- Amazon Route 53 提供網域名稱系統 (DNS) 組態
- Amazon WorkSpaces 為使用者提供雲端中的桌面體驗
- Amazon Simple Storage Service (Amazon S3) 用於儲存備份、檔案和物件
- Amazon CloudWatch 用於應用程式記錄、監控和警示
- Amazon Elastic Disaster Recovery for disaster Recovery

目標架構

下圖顯示使用 Elastic Disaster Recovery 的 JD Edwards EnterpriseOne 跨區域災難復原架構。

程序

以下是程序的高階檢閱。如需詳細資訊，請參閱 Epics 區段。

- Elastic Disaster Recovery 複寫從初始同步開始。在初始同步期間，AWS 複寫代理程式會將來源磁碟中的所有資料複寫到預備區域子網路中的適當資源。
- 初始同步完成後，持續複寫會持續無限期。
- 安裝代理程式並開始複寫後，您可以檢閱啟動參數，其中包括服務特定的組態和 Amazon EC2 啟動範本。當來源伺服器指示為已準備好進行復原時，您可以啟動執行個體。
- 當 Elastic Disaster Recovery 發出一系列 API 呼叫以開始啟動操作時，復原執行個體會根據您的啟動設定立即在 AWS 上啟動。服務會在啟動期間自動啟動轉換伺服器。
- 轉換完成後，新的執行個體會在 AWS 上旋轉並可供使用。啟動時的來源伺服器狀態由與啟動的執行個體相關聯的磁碟區表示。轉換程序涉及驅動程式、網路和作業系統授權的變更，以確保執行個體在 AWS 上原生開機。
- 啟動後，新建立的磁碟區不會再與來源伺服器保持同步。AWS 複寫代理程式會繼續定期將對來源伺服器所做的變更複寫至暫存區域磁碟區，但啟動的執行個體不會反映這些變更。
- 當您啟動新的演練或復原執行個體時，資料一律會反映在從來源伺服器複寫至預備區域子網路的最新狀態中。
- 當來源伺服器標記為準備進行復原時，您可以啟動執行個體。

> **Note**
>
> 此程序可同時運作：從主要 AWS 區域容錯移轉至 DR 區域，並在復原時容錯移轉回主要網站。您可以透過完全協調的方式，將資料複寫的方向從目標機器反轉回來源機器，以準備容錯回復。

此模式中描述的此程序優點包括：

- 彈性：複寫伺服器會根據資料集和複寫時間橫向擴展和縮減，因此您可以執行 DR 測試，而不會中斷來源工作負載或複寫。
- 可靠性：複寫功能強大、不中斷且持續。
- 自動化：此解決方案為測試、復原和容錯回復提供了統一的自動化程序。
- 成本最佳化：您只能複寫所需的磁碟區並支付這些磁碟區的費用，並僅在啟用這些資源時支付 DR 網站上的運算資源費用。您可以針對多個來源或具有大型 EBS 磁碟區的單一來源使用成本最佳化複寫執行個體（我們建議您使用運算最佳化執行個體類型）。

自動化和擴展

當您大規模執行災難復原時，JD Edwards EnterpriseOne 伺服器將相依於環境中的其他伺服器。例如：

- 在開機時連線至 JD Edwards EnterpriseOne 支援資料庫的 JD Edwards EnterpriseOne 應用程式伺服器與該資料庫有相依性。
- 需要身分驗證且需要在開機時連線到網域控制器以啟動服務的 JD Edwards EnterpriseOne 伺服器，對網域控制器具有相依性。

因此，建議您將容錯移轉任務自動化。例如，您可以使用 AWS Lambda 或 AWS Step Functions 自動化 JD Edwards EnterpriseOne 啟動指令碼和負載平衡器變更，以自動化end-to-end容錯移轉程序。如需詳細資訊，請參閱部落格文章使用 AWS Elastic Disaster Recovery 建立可擴展的災難復原計劃。

工具

AWS 服務

- Amazon Elastic Block Store (Amazon EBS) 提供區塊層級儲存體磁碟區，可與 EC2 執行個體搭配使用。

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- AWS Elastic Disaster Recovery 使用經濟實惠的儲存體、最少的運算和point-in-time復原，快速、可靠地復原內部部署和雲端應用程式，將停機時間和資料遺失降至最低。
- Amazon Virtual Private Cloud (Amazon VPC) 可讓您完全控制虛擬聯網環境，包括資源配置、連線和安全性。

最佳實務

一般最佳實務

- 制定書面計畫，說明發生實際復原事件時該怎麼做。
- 正確設定 Elastic Disaster Recovery 之後，請建立 AWS CloudFormation 範本，以便在需要時隨需建立組態。決定伺服器和應用程式的啟動順序，並將其記錄在復原計劃中。
- 執行定期演練 （適用標準 Amazon EC2 費率 ）。
- 使用 Elastic Disaster Recovery 主控台或以程式設計方式監控進行中複寫的運作狀態。
- 保護point-in-time快照，並在終止執行個體之前確認。
- 為 AWS Replication Agent 安裝建立 IAM 角色。
- 在實際 DR 案例中啟用復原執行個體的終止保護。
- 請勿對您啟動復原執行個體的伺服器使用 Elastic Disaster Recovery 主控台中的從 AWS 中斷連線動作，即使發生實際復原事件也一樣。執行中斷連線會終止與這些來源伺服器相關的所有複寫資源，包括point-in-time(PIT) 復原點。
- 變更 PIT 政策以變更快照保留的天數。
- 在 Elastic Disaster Recovery 啟動設定中編輯啟動範本，為目標伺服器設定正確的子網路、安全群組和執行個體類型。
- 使用 Lambda 或 Step Functions 自動化end-to-end容錯移轉程序，以自動化 JD Edwards EnterpriseOne 啟動指令碼和負載平衡器變更。

JD Edwards EnterpriseOne 最佳化和考量事項

- 將 PrintQueue 移至資料庫。
- 將 MediaObjects移至資料庫。
- 從批次和邏輯伺服器排除日誌和暫存資料夾。

- 從 Oracle WebLogic 排除暫存資料夾。
- 建立指令碼以在容錯移轉後啟動。
- 排除 SQL Server 的 tempdb。
- 排除 Oracle 的暫存檔案。

史詩

執行初始任務和組態

任務	Description	所需的技能
設定複寫網路。	在主要 AWS 區域中實作 JD Edwards EnterpriseOne 系統，並識別 DR 的 AWS 區域。 遵循 Elastic Disaster Recovery 文件的複寫網路需求一節中的步驟來規劃和設定複寫和 DR 網路。	AWS 管理員
判斷 RPO 和 RTO。	識別應用程式伺服器和資料庫的復原時間目標 (RTO) 和復原點目標 (RPO)。	雲端架構師、DR 架構師
啟用 Amazon EFS 的複寫。	如果適用，請使用 AWS DataSync、rsync 或其他適當的工具，為 Amazon Elastic File System (Amazon EFS) 等共用檔案系統啟用從 AWS 主要伺服器的複寫至 DR 區域。	雲端管理員
在 DR 的情況下管理 DNS。	識別在 DR 演練或實際 DR 期間更新網域名稱系統 (DNS) 的程序。	雲端管理員
建立 IAM 角色以進行設定。	遵循 Elastic Disaster Recovery 文件的 Elastic Disaster Recovery 初始化和許可一節中	雲端管理員

任務	Description	所需的技能
	的指示，建立 IAM 角色以初始化和管理 AWS 服務。	
設定 VPC 對等互連。	確定來源和目標 VPCs彼此對等且可存取。如需組態指示，請參閱 Amazon VPC 文件。	AWS 管理員

設定彈性災難復原複寫設定

任務	Description	所需的技能
初始化彈性災難復原。	開啟 Elastic Disaster Recovery 主控台，選擇目標 AWS 區域（您將在其中複寫資料並啟動復原執行個體），然後選擇設定預設複寫設定。	AWS 管理員
設定複寫伺服器。	1. 在設定複寫伺服器窗格中，輸入暫存區域子網路和複寫伺服器執行個體類型。預設選取 `t3.small` 執行個體類型。根據您的需求設定此設定，並請記得考慮執行個體定價。如需詳細資訊，請參閱 Amazon EC2 定價。 2. 在服務存取區段中，選擇檢視詳細資訊以檢閱服務連結角色，以及在服務初始化期間建立的其他政策。 3. 選擇下一步。	AWS 管理員
設定磁碟區和安全群組。	1. 在磁碟區和安全群組窗格中，選取複寫伺服器的 EBS 磁碟區類型，並將 Amazon EBS 加密設定為預設。	AWS 管理員

任務	Description	所需的技能
	2. 選取一律使用 AWS Elastic Disaster Recovery 安全群組，讓 Elastic Disaster Recovery 自動連接並監控預設安全群組。 3. 選擇下一步。	
設定其他設定。	1. 在其他設定窗格中，設定資料路由和限流、PIT 政策和標籤。 • 資料路由和限流控制資料如何從外部伺服器流向複寫伺服器。選擇使用私有 IP 進行資料複寫。否則，複寫伺服器會自動指派公有 IP，資料會透過公有網際網路流動。 • 在時間點 (PIT) 政策區段中，設定保留政策，以決定不需要快照的持續時間。預設保留期間為七天。 • 在標籤區段中，將自訂標籤新增至您 AWS 帳戶中 Elastic Disaster Recovery 建立的資源。 2. 選擇下一步，檢閱下一個窗格中的設定，然後選擇建立預設以建立預設範本。	AWS 管理員

安裝 AWS 複寫代理程式

任務	Description	所需的技能
建立 IAM 角色。	建立包含 `AWSElasticDisasterRecoveryAgentInstallationPolicy` 政策的 IAM 角色。在選取 AWS 存取類型區段中，啟用程式設計存取。請記下存取金鑰 ID 和私密存取金鑰。在安裝 AWS 複寫代理程式期間，您將需要此資訊。	AWS 管理員
檢查需求。	檢查並完成 Elastic Disaster Recovery 文件中的先決條件，以安裝 AWS Replication Agent。	AWS 管理員
安裝 AWS 複寫代理程式。	遵循您作業系統的安裝指示，並安裝 AWS Replication Agent。 • 對於 Microsoft Windows：下載安裝檔案並以管理員身分執行 .exe 檔案。 回應提示以完成安裝。 • 針對 Linux：複製下列命令（依照顯示的順序），並將其貼到您的 Secure Shell (SSH) 工作階段。第一個命令會下載安裝程式，第二個命令則會執行它。 `wget -O ./aws-replication-installer-init.py https://aws-elastic-disaste`	AWS 管理員

任務	Description	所需的技能
	r-recovery-us-west-2.s3.amazonaws.com/latest/linux/aws-replication-installer-init.py sudo python3 aws-replication-installer-init.py 回應提示以完成安裝。 **Note** 變更 URL 以反映您的區域。 針對剩餘的伺服器重複這些步驟。	
監控複寫。	返回彈性災難復原來源伺服器窗格以監控複寫狀態。初始同步需要一些時間，取決於資料傳輸的大小。 當來源伺服器完全同步時，伺服器狀態會更新為就緒。這表示複寫伺服器已在預備區域中建立，而 EBS 磁碟區已從來源伺服器複寫至預備區域。	AWS 管理員

設定啟動設定

任務	Description	所需的技能
編輯啟動設定。	若要更新演練和復原執行個體的啟動設定，請在 Elastic Disaster Recovery 主控台上選取來源伺服器，然後選擇動作、編輯啟動設定。或者，您可以從來源伺服器頁面選擇複寫來源機器，然後選擇啟動設定索引標籤。此索引標籤有兩個區段：一般啟動設定和 EC2 啟動範本。	AWS 管理員
設定一般啟動設定。	根據您的需求修改一般啟動設定。 • 執行個體類型適當調整大小：如果您選擇基本，Elastic Disaster Recovery 會略過您在 Amazon EC2 啟動範本中選取的執行個體類型，並根據來源伺服器的作業系統、CPU 和 RAM 自動選擇執行個體類型。 • 複製私有 IP：選擇是否要 Elastic Disaster Recovery，以確保演練或復原執行個體所使用的私有 IP 符合來源伺服器所使用的私有 IP。如果您選擇是，請確定您在 Amazon EC2 啟動範本中設定的子網路 IP 範圍包含私有 IP 地址。	AWS 管理員

任務	Description	所需的技能
	如需詳細資訊，請參閱 Elastic Disaster Recovery 文件中的一般啟動設定。	
設定 Amazon EC2 啟動範本。	Elastic Disaster Recovery 使用 Amazon EC2 啟動範本，為每個來源伺服器啟動演練和復原執行個體。安裝 AWS 複寫代理程式後，系統會自動為您新增至 Elastic Disaster Recovery 的每個來源伺服器建立啟動範本。 如果您想要搭配 Elastic Disaster Recovery 使用 Amazon EC2 啟動範本，則必須將其設定為預設啟動範本。 如需詳細資訊，請參閱彈性災難復原文件中的 EC2 啟動範本。	AWS 管理員

啟動 DR 演練和容錯移轉

任務	Description	所需的技能
啟動演練	1. 在 Elastic Disaster Recovery 主控台上，開啟來源伺服器頁面，並確認來源伺服器的狀態為就緒。 2. 選取您要為其執行 DR 演練的所有來源伺服器。 3. 從起始復原任務功能表中，選擇起始演練，然後選取適當的point-in-time快照。這	AWS 管理員

任務	Description	所需的技能
	會啟動所選來源伺服器的復原任務。您可以在復原任務歷史記錄索引標籤上監控任務的狀態。 啟動的演練執行個體也會出現在復原執行個體頁面上。 **Note** 來源伺服器的進一步變更將同步至複寫伺服器，而非演練執行個體。 4. 測試並驗證 DR 演練執行個體。 5. 在復原執行個體頁面上，選取演練執行個體，然後選擇動作、中斷與 AWS 的連線。這會從復原執行個體中刪除 AWS 複寫代理程式，並從 Elastic Disaster Recovery 移除與復原執行個體相關聯的所有資源。 6. 選擇刪除復原執行個體。這會從 Elastic Disaster Recovery 主控台刪除執行個體的表示，並完全取消執行個體與 Elastic Disaster Recovery 服務的關聯。它不會刪除基礎 EC2 執行個體。 7. 從 Amazon EC2 主控台終止 DR 演練執行個體。	

任務	Description	所需的技能
	如需詳細資訊，請參閱 Elastic Disaster Recovery 文件中的準備容錯移轉。	
驗證演練。	在上一個步驟中，您在 DR 區域中啟動了新的目標執行個體。根據啟動時拍攝的快照，目標執行個體是來源伺服器的複本。 在此程序中，您會連線至 Amazon EC2 目標機器，以確認它們如預期般執行。 1. 開啟 Amazon EC2 主控台。 2. 選擇執行個體 （執行中）。 3. 選取目標執行個體，並記下其私有 IPv4 地址。 4. 請確定您可以連線至 EC2 執行個體，而且 JD Edwards EnterpriseOne 和相關元件會如預期複寫。	

任務	Description	所需的技能
啟動容錯移轉。	容錯移轉是將流量從主要系統重新導向至次要系統。Elastic Disaster Recovery 透過在 AWS 上啟動復原執行個體，協助您執行容錯移轉。啟動復原執行個體後，您可以將流量從主要系統重新導向至這些執行個體。 1. 在 Elastic Disaster Recovery 主控台上，開啟來源伺服器頁面，並確認來源伺服器的準備復原欄顯示就緒，而資料複寫狀態欄顯示良好。 2. 選取來源伺服器。從起始復原任務功能表中，選擇起始復原。 3. 選取要從中啟動復原執行個體的point-in-time快照，然後選擇啟動復原。 這會啟動復原任務。您可以在復原執行個體頁面上監控任務的狀態。 4. 測試並驗證復原執行個體。如有需要，請調整 DNS 組態，並將 JD Edwards EnterpriseOne 應用程式連線至資料庫。 5. 您現在可以中斷和停用來源 JD Edwards EnterpriseOne 伺服器，因為所有變更都已寫入新的復原執行個體。	AWS 管理員

任務	Description	所需的技能
	6. 遵循安裝 AWS 複寫代理程式範例中所述的程序，將復原執行個體註冊為 DR 區域中的來源伺服器。 如需詳細資訊，請參閱彈性災難復原文件中的執行容錯移轉。	

任務	Description	所需的技能
啟動容錯回復。	啟動容錯回復的程序類似於啟動容錯移轉的程序。 1. 在主要區域中開啟 Elastic Disaster Recovery 主控台。導覽至復原執行個體頁面，選取演練執行個體，然後選擇動作、中斷與 AWS 的連線、刪除復原執行個體。 2. 在 DR 區域中開啟 Elastic Disaster Recovery 主控台。安裝 AWS 複寫代理程式，將新的 JD Edwards EnterpriseOne 伺服器註冊為 DR 區域中的來源伺服器。資料將與佈建在新預備子網路中的新複寫伺服器同步。 **Note** 當新的 JD Edwards EnterpriseOne 伺服器註冊為來源伺服器時，您可能會在 Elastic Disaster Recovery 主控台中看到兩個來源伺服器：一個從主要 EC2 執行個體建立的伺服器，以及從復原執行個體建立的新伺服器。建議您正確標記伺服器以避免混	AWS 管理員

任務	Description	所需的技能
	淆，最好將新伺服器新增至啟動範本。 3. 若要從主要區域重新啟動 DR 複寫，請取消啟動的復原執行個體與 DR 區域中 Elastic Disaster Recovery 主控台的關聯，並將主機註冊為主要區域中的來源伺服器。 如需詳細資訊，請參閱彈性災難復原文件中的執行容錯回復。	

任務	Description	所需的技能
啟動 JD Edwards EnterpriseOne 元件。	1. 登入資料庫伺服器以啟動 JD Edwards EnterpriseOne 資料庫。 2. 當資料庫執行時，啟動 JD Edwards EnterpriseOne 邏輯和批次伺服器。 3. 在 Web 伺服器上啟動 WebLogic，並在 JAS 伺服器上啟動 JAS 執行個體。 4. 在佈建伺服器和 SM 主控台的伺服器上啟動 WebLogic。 5. 在伺服器上啟動 SM 代理程式。 6. 確認登入 JD Edwards EnterpriseOne 正常運作。 您需要在 Route 53 和 Application Load Balancer 中進行變更，JD Edwards EnterpriseOne 連結才能運作。 您可以使用 Lambda、Step Functions 和 Systems Manager（執行命令）來自動化這些步驟。 **Note** Elastic Disaster Recovery 會執行託管作業系統和檔案系統的來源 EC2 執行個體 EBS 磁碟區的	JD Edwards EnterpriseOne CNC

任務	Description	所需的技能
	區塊層級複寫。使用 Amazon EFS 建立的共用檔案系統不屬於此複寫。您可以使用 AWS DataSync 將共用檔案系統複寫到 DR 區域，如第一個範例所述，然後在 DR 系統中掛載這些複寫的檔案系統。	

疑難排解

問題	解決方案
來源伺服器資料複寫狀態為停滯且複寫延遲。如果您檢查詳細資訊，資料複寫狀態會顯示未顯示客服人員。	檢查 以確認停滯的來源伺服器正在執行。 **Note** 如果來源伺服器故障，複寫伺服器會自動終止。 如需延遲問題的詳細資訊，請參閱 Elastic Disaster Recovery 文件中的複寫延遲問題。
掃描磁碟後，在來源 EC2 執行個體中安裝 AWS Replication Agent 會在 RHEL 8.2 中失敗。 `aws_replication_agent_installer.log` 顯示核心標頭遺失。	在 RHEL 8、CentOS 8 或 Oracle Linux 8 上安裝 AWS 複寫代理程式之前，請執行： `sudo yum install elfutils-libelf-devel` 如需詳細資訊，請參閱 Elastic Disaster Recovery 文件中的 Linux 安裝需求。

問題	解決方案
在 Elastic Disaster Recovery 主控台上，您會看到來源伺服器為就緒，且延遲和資料複寫狀態為停滯。 根據 AWS 複寫代理程式無法使用的時間長度，狀態可能表示高延遲，但問題保持不變。	使用作業系統命令來確認 AWS 複寫代理程式正在來源 EC2 執行個體中執行，或確認執行個體正在執行。 修正任何問題後，Elastic Disaster Recovery 會重新啟動掃描。等到所有資料同步且複寫狀態為良好，再開始 DR 演練。
具有高延遲的初始複寫。在 Elastic Disaster Recovery 主控台上，您可以看到來源伺服器的初始同步狀態非常慢。	檢查 Elastic Disaster Recovery 文件的複寫延遲問題區段中記錄的複寫延遲問題。 由於內部運算操作，複寫伺服器可能無法處理負載。在這種情況下，請在諮詢 AWS 技術支援團隊後嘗試升級執行個體類型。

相關資源

- AWS Elastic Disaster Recovery 使用者指南
- 使用 AWS Elastic Disaster Recovery 建立可擴展的災難復原計劃 (AWS 部落格文章)
- AWS Elastic Disaster Recovery - 技術簡介 (AWS Skill Builder 課程；需要登入)
- AWS Elastic Disaster Recovery 快速入門指南

在多區域、多帳戶組織中設定 CloudFormation 偏離偵測

Ram Kandaswamy，Amazon Web Services

總結

Amazon Web Services (AWS) 使用者通常會尋找有效的方法來偵測資源組態不相符，包括 AWS CloudFormation 堆疊中的漂移，並盡快修正它們。 AWS Control Tower 使用 時尤其如此。

此模式提供方案解決方案，透過使用合併的資源組態變更並對這些變更採取行動來產生結果，有效率地解決問題。此解決方案專為在多個堆疊中建立多個堆疊 AWS 區域，或在多個帳戶中建立多個 CloudFormation 堆疊，或兩者結合的情況而設計。解決方案的目標如下：

- 簡化偏離偵測程序
- 設定通知和提醒
- 設定合併報告

先決條件和限制

先決條件

- AWS Config 在所有必須監控的區域和帳戶中啟用

限制

- 產生的報告僅支援逗號分隔值 (CSV) 和 JSON 輸出格式。

Architecture

下圖顯示 AWS Organizations 設定多個帳戶。 AWS Config 規則會在帳戶之間進行通訊。

工作流程包含下列步驟：

1. AWS Config 規則會偵測偏離。
2. 在其他帳戶中找到的偏離偵測結果會傳送至管理帳戶。
3. Amazon CloudWatch 規則會呼叫 AWS Lambda 函數。

4. Lambda 函數會查詢彙總結果的 AWS Config 規則。
5. Lambda 函數會通知 Amazon Simple Notification Service (Amazon SNS)，其會傳送偏離的電子郵件通知。

自動化和擴展

此處提供的解決方案可以同時針對其他區域和帳戶進行擴展。

工具

AWS 服務

- AWS Config 提供 中 AWS 資源組態的詳細檢視 AWS 帳戶。這包含資源彼此之間的關係和之前的組態方式，所以您可以看到一段時間中組態和關係的變化。
- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您在 上執行的應用程式。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。

史詩

自動化 的偏離偵測 CloudFormation

任務	Description	所需的技能
建立彙總工具。	1. 登入 AWS 管理主控台 並開啟位於 https：//https://console.aws.amazon.com/config 的 AWS Config 主控台。 2. 在管理帳戶中建立彙總工具。 3. 確保資料複寫已開啟，以便 AWS Config 可從來源帳戶擷取資料。	雲端架構師

任務	Description	所需的技能
	4. 選取所有適用的區域和帳戶。您可以根據 選取帳戶 AWS Organizations。我們建議您使用此方法，因為組織中的新帳戶會自動成為彙總工具的一部分。	
建立 AWS 受管規則。	新增 `cloudformation-stack-drift-detection-check` AWS受管規則。規則需要一個參數值：`cloudformationArn` 。 輸入具有偵測堆疊偏離許可的 IAM 角色 Amazon Resource Name (ARN)。角色必須具有可讓 AWS Config 擔任角色的信任政策。	雲端架構師
建立彙總工具的進階查詢區段。	若要從多個來源擷取漂移堆疊，請建立下列查詢： `SELECT resourceId, configuration.driftInformation.stackDriftStatus WHERE resourceType = 'AWS::CloudFormation::Stack' AND configuration.driftInformation.stackDriftStatus IN ('DRIFTED')`	雲端架構師、開發人員

任務	Description	所需的技能
自動化執行查詢並發佈。	1. 使用連接的程式碼建立 Lambda 函數。Lambda 會將結果發佈至在 Lambda 函數中做為環境變數提供的 SNS 主題。 2. 若要接收提醒，請建立 SNS 主題的電子郵件訂閱。	雲端架構師、開發人員
建立 CloudWatch 規則。	建立排程型 CloudWatch 規則來呼叫負責提醒的 Lambda 函數。	雲端架構師

相關資源

資源

- 什麼是 AWS Config？
- 多帳戶多區域資料彙總
- 偵測堆疊和資源的未受管組態變更
- IAM：將 IAM 角色傳遞至特定 AWS 服務
- 什麼是 Amazon SNS？

其他資訊

考量

我們建議使用此模式中顯示的解決方案，而不是使用涉及特定間隔 API 呼叫的自訂解決方案，來在每個 CloudFormation 堆疊或堆疊集上啟動偏離偵測。以特定間隔使用 API 呼叫的自訂解決方案可能會導致大量的 API 呼叫並影響效能。由於 API 呼叫的數量，可能會發生限流。如果僅根據排程識別資源變更，則另一個潛在問題是偵測延遲。

由於堆疊集是由堆疊組成，因此您可以使用此解決方案。堆疊執行個體詳細資訊也可作為解決方案的一部分使用。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

成功將 S3 儲存貯體匯入為 AWS CloudFormation 堆疊

Ram Kandaswamy，Amazon Web Services

總結

如果您使用 Amazon Web Services (AWS) 資源，例如 Amazon Simple Storage Service (Amazon S3) 儲存貯體，並想要使用基礎設施做為程式碼 (IaC) 方法，則可以將資源匯入 AWS CloudFormation，並將其做為堆疊管理。

此模式提供將 S3 儲存貯體成功匯入為 AWS CloudFormation 堆疊的步驟。透過使用此模式的方法，您可以避免在單一動作中匯入 S3 儲存貯體時可能發生的錯誤。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 現有的 S3 儲存貯體和 S3 儲存貯體政策。如需詳細資訊，請參閱 AWS 知識中心中的我應該使用哪些 S3 儲存貯體政策來符合 AWS Config 規則 s3-bucket-ssl-requests-onlyAWS Config。
- 現有的 AWS Key Management Service (AWS KMS) 金鑰及其別名。如需詳細資訊，請參閱 AWS KMS 文件中的使用別名。
- 下載到本機電腦的範例 `CloudFormation-template-S3-bucket` AWS CloudFormation 範本（已連接）。

Architecture

該圖顯示以下工作流程：

1. 使用者會建立 JSON 或 YAML 格式的 AWS CloudFormation 範本。
2. 範本會建立 AWS CloudFormation 堆疊來匯入 S3 儲存貯體。
3. AWS CloudFormation 堆疊會管理您在範本中指定的 S3 儲存貯體。

技術堆疊

- AWS CloudFormation

- AWS Identity and Access Management (IAM)
- AWS KMS
- Amazon S3

工具

- AWS CloudFormation – AWS CloudFormation 可協助您以可預測且重複的方式建立和佈建 AWS 基礎設施部署。
- AWS Identity and Access Management (IAM) – IAM 是一種 Web 服務，可安全地控制對 AWS 服務的存取。
- AWS KMS – AWS Key Management Service (AWS KMS) 是一種針對雲端擴展的加密和金鑰管理服務。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是網際網路的儲存體。

史詩

以 AWS CloudFormation 堆疊形式匯入具有 AWS KMS key型加密的 S3 儲存貯體

任務	Description	所需的技能

建立範本以匯入 S3 儲存貯體和 KMS 金鑰。	在本機電腦上，使用下列範例範本建立範本以匯入 S3 儲存貯體和 KMS 金鑰：
	`AWSTemplateFormatVersion: 2010-09-09`
	`Parameters:`
	`  bucketName:`
	`    Type: String`
	`Resources:`
	`  S3Bucket:`
		AWS DevOps

任務	Description	所需的技能

	<pre> Type: 'AWS::S3:
	:Bucket'
	
	    DeletionPolicy:
	 Retain
	
	    Properties:
	
	      BucketName: !Ref
	 bucketName
	
	      BucketEncryption:
	
	        ServerSid
	eEncryptionConfigu
	ration:
	
	          - ServerSid
	eEncryptionByDefault:
	
	              SSEAlgori
	thm: 'aws:kms'
	
	              KMSMaster
	KeyID: !GetAtt
	
	                -
	 KMSS3Encryption
	
	                - Arn
	
	  KMSS3Encryption:
	
	    Type: 'AWS::KMS
	::Key'
	
	    DeletionPolicy:
	 Retain
	
	    Properties:
	
	      Enabled: true</pre>	

任務	Description	所需的技能

	KeyPolicy: !Sub |-
	{
	"Id": "key-consolepolicy-3",
	"Version": "2012-10-17",
	"Statement": [
	{
	"Sid": "Enable IAM User Permissions",
	"Effect": "Allow",
	"Principal": {
	"AWS": ["arn:aws:iam::${AWS::AccountId}:root"]
	},
	"Action": "kms:*",
	"Resource": "*"
	}	

任務	Description	所需的技能

	<pre>}
	
	]
	
	}
	
	EnableKey
	Rotation: true</pre>	

建立堆疊。	1. 登入 AWS 管理主控台，開啟 AWS CloudFormation 主控台，選擇檢視堆疊，選擇建立堆疊，然後選擇使用現有資源（匯入資源）。 2. 選擇上傳範本檔案，然後上傳您先前建立的範本檔案。 3. 輸入堆疊的名稱，並根據您的需求設定其餘選項。 4. 選擇建立堆疊，並等待堆疊的狀態變更為 IMPORT_COMPLETE 。	AWS DevOps

任務	Description	所需的技能

建立 KMS 金鑰別名。	1. 在 AWS CloudFormation 主控台上，選擇 Stacks，選擇您先前建立的堆疊名稱，選擇範本窗格，然後選擇在設計工具中檢視。
	2. 將下列程式碼片段新增至範本的 `Resource`區段，然後選擇建立堆疊並完成精靈：
	`KMSS3EncryptionAlias:`
	`  Type: 'AWS::KMS ::Alias'`
	`  DeletionPolicy: Retain`
	`  Properties:`
	`  AliasName: alias/ S3BucketKey`
	`  TargetKeyId: !Ref KMSS3Encryption`
	如需詳細資訊，請參閱 AWS CloudFormation 文件中的 AWS CloudFormation 堆疊更新。 AWS CloudFormation
		AWS DevOps

任務	Description	所需的技能
更新堆疊以包含 S3 儲存貯體政策。	1. 在 AWS CloudFormation 主控台上，選擇 Stacks，選擇您先前建立的堆疊名稱，選擇範本窗格，然後選擇在設計工具中檢視。 2. 將下列程式碼片段新增至範本的 Resource區段，然後選擇建立堆疊並完成精靈： S3BucketPolicy: Type: 'AWS::S3: :BucketPolicy' Properties: Bucket: !Ref S3Bucket PolicyDocument: ! Sub |- { "Version": "2008-10- 17", "Id": "restricthttp", "Statement": [{ "Sid": "denyhttp",	AWS DevOps

任務	Description	所需的技能

	<pre>"Effect": "Deny",
	"Principal": {
	"AWS": "*"
	},
	"Action": "s3:*",
	"Resource": ["arn:aws:s3:::${S3Bucket}","arn:aws:s3:::${S3Bucket}/*"],
	"Condition": {
	"Bool": {
	"aws:SecureTransport": "false"
	}
	}
	}
	]</pre>	

任務	Description	所需的技能
	} **Note** 此 S3 儲存貯體政策具有拒絕陳述式，可限制不安全的 API 呼叫。	
更新金鑰政策。	1. 在 AWS CloudFormation 主控台上，選擇 Stacks，選擇您先前建立的堆疊名稱，選擇範本窗格，然後選擇在設計工具中檢視。 2. 修改範本的 KMS 資源，以包含允許管理員管理 KMS 金鑰的金鑰政策。 3. 選擇建立堆疊，選擇下一步，然後根據您的需求完成精靈。 如需詳細資訊，請參閱 AWS KMS 文件中的 中的金鑰政策 AWS KMS。	AWS 管理員

任務	Description	所需的技能

新增資源層級標籤。	1. 在 AWS CloudFormation 主控台上，選擇 Stacks，選擇您先前建立的堆疊名稱，選擇範本窗格，然後選擇在設計工具中檢視。
	2. 將下列程式碼片段新增至範本的 Amazon S3 資源`Properties` 區段，然後選擇建立堆疊並完成精靈：
	`Tags:`
	`  - Key: createdBy`
	`    Value: Cloudformation`
		AWS DevOps

相關資源

- 將現有資源帶入 AWS CloudFormation 管理
- AWS re：Invent 2017：深入探討 AWS CloudFormation （影片）

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS DataSync 同步不同 AWS 區域中 Amazon EFS 檔案系統之間的資料 DataSync

Sarat Chandra Pothula 和 Aditya Ambati，Amazon Web Services

總結

此解決方案提供強大的架構，可在不同 AWS 區域中的 Amazon Elastic File System (Amazon EFS) 執行個體之間有效且安全地進行資料同步。此方法可擴展，並提供受控的跨區域資料複寫。此解決方案可以增強您的災難復原和資料備援策略。

透過使用 AWS Cloud Development Kit (AWS CDK)，此模式會使用 做為基礎設施的程式碼 (IaC) 方法來部署解決方案資源。AWS CDK 應用程式會部署必要的 AWS DataSync、Amazon EFS、Amazon Virtual Private Cloud (Amazon VPC) 和 Amazon Elastic Compute Cloud (Amazon EC2) 資源。此 IaC 提供可重複且版本控制的部署程序，完全符合 AWS 最佳實務。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- AWS Command Line Interface (AWS CLI) 2.9.11 版或更新版本，已安裝並設定
- AWS CDK 2.114.1 版或更新版本，已安裝並已引導
- NodeJS 20.8.0 版或更新版本，已安裝

限制

- 解決方案會繼承 DataSync 和 Amazon EFS 的限制，例如資料傳輸率、大小限制和區域可用性。如需詳細資訊，請參閱 AWS DataSync 配額和 Amazon EFS 配額。
- 此解決方案僅支援 Amazon EFS。DataSync 支援其他 AWS 服務，例如 Amazon Simple Storage Service (Amazon S3) 和 Amazon FSx for Lustre。不過，此解決方案需要修改，才能與這些其他服務同步資料。

Architecture

此解決方案會部署下列 AWS CDK 堆疊：

- Amazon VPC 堆疊 – 此堆疊會在主要和次要 AWS 區域中設定虛擬私有雲端 (VPC) 資源，包括子網路、網際網路閘道和 NAT 閘道。
- Amazon EFS 堆疊 – 此堆疊會將 Amazon EFS 檔案系統部署到主要和次要區域，並將其連接到各自的 VPCs。
- Amazon EC2 堆疊 – 此堆疊會在主要和次要區域中啟動 EC2 執行個體。這些執行個體設定為掛載 Amazon EFS 檔案系統，允許其存取共用儲存體。
- DataSync 位置堆疊 – 此堆疊使用名為 的自訂建構`DataSyncLocationConstruct`，在主要和次要區域中建立 DataSync 位置資源。這些資源定義用於資料同步的端點。
- DataSync 任務堆疊 – 此堆疊使用名為 的自訂建構`DataSyncTaskConstruct`，在主要區域中建立 DataSync 任務。此任務設定為使用 DataSync 來源和目的地位置，在主要和次要區域之間同步資料。

工具

AWS 服務

- AWS 雲端開發套件 (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。
- AWS DataSync 是一種線上資料傳輸和探索服務，可協助您在 AWS 儲存服務之間來回移動檔案或物件資料。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon Elastic File System (Amazon EFS) 可協助您在 AWS 雲端中建立和設定共用檔案系統。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路類似於您在自己的資料中心內操作的傳統網路，具有使用可擴展的 AWS 基礎設施的優勢。

程式碼儲存庫

此模式的程式碼可在 GitHub Amazon EFS 跨區域 DataSync 專案儲存庫中使用。

最佳實務

遵循 TypeScript 中使用 AWS CDK 建立 IaC 專案的最佳實務中所述的最佳實務。

史詩

部署 AWS CDK 應用程式

任務	Description	所需的技能
複製專案儲存庫。	輸入下列命令以複製 Amazon EFS 跨區域 DataSync 專案儲存庫。 `git clone https://github.com/aws-samples/aws-efs-crossregion-datasync.git`	AWS DevOps
安裝 npm 相依性。	輸入以下命令。 `npm ci`	AWS DevOps
選擇主要和次要區域。	在複製的儲存庫中，導覽至 `src/infa`目錄。在 `Launcher.ts` 檔案中，更新 `PRIMARY_AWS_REGION` 和 `SECONDARY_AWS_REGION` 值。使用對應的區域代碼。 `const primaryRegion = { account: account, region: '<PRIMARY_AWS_REGION>' }; const secondaryRegion = { account: account, region: '<SECONDARY_AWS_REGION>' };`	AWS DevOps
引導環境。	輸入下列命令以引導您要使用的 AWS 帳戶和 AWS 區域。	AWS DevOps

任務	Description	所需的技能
	`cdk bootstrap <aws_account>/<aws_region>` 如需詳細資訊，請參閱 AWS CDK 文件中的引導。	
列出 AWS CDK 堆疊。	輸入下列命令以檢視應用程式中 AWS CDK 堆疊的清單。 `cdk ls`	AWS DevOps
合成 AWS CDK 堆疊。	輸入下列命令，為 AWS CDK 應用程式中定義的每個堆疊產生 AWS CloudFormation 範本。 `cdk synth`	AWS DevOps
部署 AWS CDK 應用程式。	輸入下列命令，將所有堆疊部署到您的 AWS 帳戶，而不需要任何變更的手動核准。 `cdk deploy --all --require-approval never`	AWS DevOps

驗證部署

任務	Description	所需的技能
登入主要區域中的 EC2 執行個體。	1. 使用 AWS Systems Manager 的功能 Session Manager 登入主要區域中的 EC2 執行個體。如需說明，請參閱使用 AWS Systems	AWS DevOps

任務	Description	所需的技能
	Manager Session Manager 連線至 Linux 執行個體。 2. 將目錄變更為 Amazon EFS 掛載路徑。 `cd /mnt/efs`	

建立暫存檔案。	輸入下列命令以在 Amazon EFS 掛載路徑中建立暫存檔案。
	`sudo dd if=/dev/zero \`
	`of=tmptst.dat \`
	`bs=1G \`
	`seek=5 \`
	`count=0`
	
	`ls -lrt tmptst.dat`
		AWS DevOps

任務	Description	所需的技能

啟動 DataSync 任務。	輸入下列命令，將暫存檔案從主要區域複寫到次要區域，其中 <ARN-task> 是 DataSync 任務的 Amazon Resource Name (ARN)。
	`aws datasync start-task-execution \`
	`  --task-arn <ARN-task>`
	命令會以下列格式傳回任務執行的 ARN。
	`arn:aws:datasync:<region>:<account-ID>:task/task-execution/<exec-ID>`
		AWS DevOps

檢查資料傳輸的狀態。	輸入下列命令來描述 DataSync 執行任務，其中 <ARN-task-execution> 是任務執行的 ARN。
	`aws datasync describe-task-execution \`
	`  --task-execution-arn <ARN-task-execution>`
	當 PrepareStatus 、和 VerifyStatus 都有值 時TransferStatus ，DataSync 任務即完成SUCCESS。
		AWS DevOps

任務	Description	所需的技能
登入次要區域中的 EC2 執行個體。	1. 使用 AWS Systems Manager 的功能 Session Manager，登入次要區域中的 EC2 執行個體。如需說明，請參閱使用 AWS Systems Manager Session Manager 連線至 Linux 執行個體。 2. 將目錄變更為 Amazon EFS 掛載路徑。 `cd /mnt/efs`	AWS DevOps

驗證複寫。	輸入下列命令，以確認暫存檔案存在於 Amazon EFS 檔案系統中。
	`ls -lrt`
	`tmptst.dat`
		AWS DevOps

相關資源

AWS 文件

- AWS CDK API 參考
- 使用 Amazon EFS 設定 AWS DataSync 傳輸
- 針對 AWS DataSync 傳輸的問題進行故障診斷

其他 AWS 資源

- AWS DataSync FAQs

使用 LocalStack 和 Terraform Tests 測試 AWS 基礎設施

Ivan Girardi 和 Ioannis Kalyvas，Amazon Web Services

總結

此模式可協助您在 Terraform AWS 中以程式碼 (IaC) 形式對基礎設施進行本機測試，而無需在您的 AWS 環境中佈建基礎設施。它將 Terraform Tests 架構與 LocalStack 整合。LocalStack Docker 容器提供模擬各種 的本機開發環境 AWS 服務。這可協助您在基礎設施部署上測試和反覆執行，而不會在中產生成本 AWS 雲端。

此解決方案提供下列優點：

- 成本最佳化 – 針對 LocalStack 執行測試不需要使用 AWS 服務。這可避免產生與建立、操作和修改這些 AWS 資源相關的成本。
- 速度和效率 – 在本機進行測試通常也比部署 AWS 資源更快。此快速回饋迴圈可加速開發和偵錯。由於 LocalStack 在本機執行，因此您可以開發和測試 Terraform 組態檔案，而無需網際網路連線。您可以在本機偵錯 Terraform 組態檔案，並接收立即的意見回饋，以簡化開發程序。
- 一致性和重現性 – LocalStack 提供一致的測試環境。此一致性有助於確保無論外部 AWS 變更或網路問題為何，測試都會產生相同的結果。
- 隔離 – 使用 LocalStack 進行測試可防止意外影響即時 AWS 資源或生產環境。此隔離可讓您安全地實驗和測試各種組態。
- 自動化 – 與持續整合和持續交付 (CI/CD) 管道整合，可協助您自動測試 Terraform 組態檔案。管道會在部署之前徹底測試 IaC。
- 彈性 – 您可以模擬不同的 AWS 區域 AWS 帳戶和服務組態，以更接近您的生產環境。

先決條件和限制

先決條件

- 安裝 Docker
- 啟用對預設 Docker 通訊端 () 的存取`/var/run/docker.sock`。如需詳細資訊，請參閱 LocalStack 文件。
- 安裝 Docker Compose
- 安裝 Terraform 1.6.0 版或更新版本

- 安裝 Terraform CLI
- 設定 Terraform AWS 提供者
- （選用）安裝和設定 AWS Command Line Interface (AWS CLI)。如需如何 AWS CLI 搭配 LocalStack 使用 的範例，請參閱使用 LocalStack 和 Terraform Tests 儲存庫的 GitHub 測試基礎設施。 AWS LocalStack

限制

- 此模式提供測試 Amazon Simple Storage Service (Amazon S3) AWS Lambda AWS Step Functions 和 Amazon DynamoDB 資源的明確範例。不過，您可以擴展此解決方案以包含其他 AWS 資源。
- 此模式提供在本機執行 Terraform Tests 的說明，但您可以將測試整合到任何 CI/CD 管道。
- 此模式提供使用 LocalStack Community 映像的說明。如果您使用的是 LocalStack Pro 映像，請參閱 LocalStack Pro 文件。
- LocalStack 為不同的 AWS APIs提供模擬服務。如需完整清單，請參閱AWS 服務功能涵蓋範圍。有些進階功能可能需要訂閱 LocalStack Pro。

Architecture

下圖顯示此解決方案的架構。主要元件是原始程式碼儲存庫、CI/CD 管道和 LocalStack Docker 容器。LocalStack Docker 容器會在 AWS 服務 本機託管下列項目：

- 用於儲存檔案的 Amazon S3 儲存貯體
- 用於監控和記錄的 Amazon CloudWatch
- 用於執行無伺服器程式碼的 AWS Lambda 函數
- 用於協調多步驟工作流程 AWS Step Functions 的狀態機器
- 用於存放 NoSQL 資料的 Amazon DynamoDB 資料表

該圖顯示以下工作流程：

1. 您可以將 Terraform 組態檔案新增並遞交至原始程式碼儲存庫。
2. CI/CD 管道會偵測變更，並啟動靜態 Terraform 程式碼分析的建置程序。管道會建置並執行 LocalStack Docker 容器。然後，管道會啟動測試程序。

3. 管道會將物件上傳至託管在 LocalStack Docker 容器中的 Amazon S3 儲存貯體。
4. 上傳物件會叫用 AWS Lambda 函數。
5. Lambda 函數會將 Amazon S3 事件通知存放在 CloudWatch 日誌中。
6. Lambda 函數會啟動 AWS Step Functions 狀態機器。
7. 狀態機器會將 Amazon S3 物件的名稱寫入 DynamoDB 資料表。
8. CI/CD 管道中的測試程序會驗證上傳的物件名稱是否符合 DynamoDB 資料表中的項目。它也會驗證 S3 儲存貯體是否以指定的名稱部署，以及 AWS Lambda 函數是否已成功部署。

工具

AWS 服務

- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您執行的應用程式。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務 來建置業務關鍵型應用程式。

其他工具

- Docker 是一組平台即服務 (PaaS) 產品，可在作業系統層級使用虛擬化在容器中交付軟體。
- Docker Compose 是一種用於定義和執行多容器應用程式的工具。
- LocalStack 是在單一容器中執行的雲端服務模擬器。透過使用 LocalStack，您可以在使用的本機電腦上執行工作負載 AWS 服務，而無需連線到 AWS 雲端。
- Terraform 是 HashiCorp 的 IaC 工具，可協助您建立和管理雲端和內部部署資源。
- Terraform Tests 可協助您透過類似整合或單元測試的測試來驗證 Terraform 模組組態更新。

程式碼儲存庫

此模式的程式碼可在 GitHub Test AWS 基礎設施中使用 LocalStack 和 Terraform Tests 儲存庫。

最佳實務

- 此解決方案會測試 Terraform 組態檔案中指定的 AWS 基礎設施，而且不會在 中部署這些資源 AWS 雲端。如果您想要部署資源，請遵循最低權限 (IAM 文件） 原則，並正確設定 Terraform 後端 (Terraform 文件）。
- 在 CI/CD 管道中整合 LocalStack 時，建議您不要在權限模式下執行 LocalStack Docker 容器。如需詳細資訊，請參閱執行期權限和 Linux 功能 (Docker 文件） 和自我管理執行器的安全性 (GitLab 文件）。

史詩

部署解決方案

任務	Description	所需的技能
複製儲存庫。	在 bash shell 中，輸入下列命令。這會使用 LocalStack 和 Terraform Tests 儲存庫從 GitHub 複製測試 AWS 基礎設施： GitHub `git clone https://github.com/aws-samples/localstack-terraform-test.git`	DevOps 工程師

執行 LocalStack 容器。	1. 輸入下列命令以導覽至複製的儲存庫：
	`cd localstack-terraform-test`
	2. 輸入下列命令以分離模式啟動 LocalStack Docker 容器：
	`docker-compose up -d`
		DevOps 工程師

任務	Description	所需的技能
	3. 等待 LocalStack Docker 容器正常運作。	
初始化 Terraform。	輸入下列命令來初始化 Terraform： `terraform init`	DevOps 工程師

執行 Terraform 測試。	1. 輸入下列命令來執行 Terraform Tests：
	`terraform test`
	2. 驗證所有測試是否成功完成。輸出格式應類似以下內容：
	`Success! 3 passed, 0 failed.`
		DevOps 工程師

清除資源。	輸入下列命令來銷毀 LocalStack 容器： `docker-compose down`	DevOps 工程師

疑難排解

問題	解決方案
`Error: reading DynamoDB Table Item (Files|README.md): empty` 執行 `terraform test`命令時的結果。	1. 重新輸入 `terraform test`命令。 2. 如果這無法解決錯誤，請編輯 main.tf 檔案，將睡眠逾時增加到大於 15 秒的值： `resource "time_sleep" "wait" { create_duration = "15s"`

問題	解決方案

	<pre> triggers = {
	    s3_object = local.key_json
	  }
	}</pre>

相關資源

- Terraform 入門： AWS CDK 和 AWS CloudFormation 專家指引 (AWS 方案指引)
- 使用 Terraform AWS 提供者的最佳實務 (AWS 方案指引)
- Terraform CI/CD，並使用新的 Terraform 測試架構 AWS 在 上進行測試 (AWS 部落格文章)
- 從 （部落格文章） 使用 LocalStack Cloud Emulator 加速軟體交付 AWS MarketplaceAWS

其他資訊

與 GitHub 動作整合

您可以使用 GitHub 動作，在 CI/CD 管道中整合 LocalStack 和 Terraform 測試。如需詳細資訊，請參閱 GitHub 動作文件。以下是範例 GitHub 動作組態檔案：

```
name: LocalStack Terraform Test

on:
  push:
    branches:
      - '**'

  workflow_dispatch: {}

jobs:
  localstack-terraform-test:
    runs-on: ubuntu-latest

    steps:
    - uses: actions/checkout@v4

    - name: Build and Start LocalStack Container
      run: |
```

```
        docker compose up -d

    - name: Setup Terraform
      uses: hashicorp/setup-terraform@v3
      with:
        terraform_version: latest

    - name: Run Terraform Init and Validation
      run: |
        terraform init
        terraform validate
        terraform fmt --recursive --check
        terraform plan
        terraform show

    - name: Run Terraform Test
      run: |
        terraform test

    - name: Stop and Delete LocalStack Container
      if: always()
      run: docker compose down
```

從 ENSA1 升級 SAP Pacemaker 叢集到 ENSA2

Gergely Cserdi 和 Balazs Sandor Skublics，Amazon Web Services

總結

此模式說明將基於獨立 Enqueue Server (ENSA1) 的 SAP Pacemaker 叢集升級至 ENSA2 的步驟和考量。此模式中的資訊同時適用於 SUSE Linux Enterprise Server (SLES) 和 Red Hat Enterprise Linux (RHEL) 作業系統。

SAP NetWeaver 7.52 或 S/4HANA 1709 及更舊版本的 Pacemaker 叢集會在 ENSA1 架構上執行，並專門針對 ENSA1 設定。如果您在 Amazon Web Services (AWS) 上執行 SAP 工作負載，而且有興趣移至 ENSA2，您可能會發現 SAP、SUSE 和 RHEL 文件不提供完整的資訊。此模式說明重新設定 SAP 參數和 Pacemaker 叢集以從 ENSA1 升級到 ENSA2 所需的技術步驟。它提供 SUSE 系統的範例，但 RHEL 叢集的概念相同。

> **Note**
>
> ENSA1 和 ENSA2 是僅適用於 SAP 應用程式的概念，因此此模式中的資訊不適用於 SAP HANA 或其他類型的叢集。

> **Note**
>
> 在技術上，ENSA2 可以搭配或不搭配 Enqueue Replicator 2 使用。不過，高可用性 (HA) 和容錯移轉自動化（透過叢集解決方案）需要 Enqueue Replicator 2。此模式使用 ENSA2 叢集一詞來參考具有獨立 Enqueue Server 2 和 Enqueue Replicator 2 的叢集。

先決條件和限制

先決條件

- 在 SLES 或 RHEL 上使用 Pacemaker 和 Corosync 的工作 ENSA1-based叢集。
- 至少兩個執行 (ABAP) SAP Central Services (ASCS/SCS) 和 Enqueue Replication Server (ERS) 執行個體的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。
- 具備管理 SAP 應用程式和叢集的知識。
- 以根使用者身分存取 Linux 環境。

限制

- ENSA1-based叢集僅支援雙節點架構。
- ENSA2-based叢集無法部署至 7.52 之前的 SAP NetWeaver 版本。
- 叢集中的 EC2 執行個體應該位於不同的 AWS 可用區域。

產品版本

- SAP NetWeaver 7.52 版或更新版本
- 從 S/4HANA 2020 開始，僅支援 ENSA2 叢集
- 支援 ENSA2 和 Enqueue Replicator 2 的核心 7.53 或更新版本
- SLES for SAP 應用程式第 12 版或更新版本
- RHEL for SAP 搭配高可用性 (HA) 7.9 版或更新版本

Architecture

來源技術堆疊

- SAP NetWeaver 7.52 搭配 SAP 核心 7.53 或更新版本
- SLES 或 RHEL 作業系統

目標技術堆疊

- SAP NetWeaver 7.52 搭配 SAP Kernel 7.53 或更新版本，包括 S/4HANA 2020 搭配 ABAP 平台
- SLES 或 RHEL 作業系統

目標架構

下圖顯示以 ENSA2 叢集為基礎的 ASCS/SCS 和 ERS 執行個體的 HA 組態。

ENSA1 和 ENSA2 叢集的比較

SAP 推出 ENSA2 做為 ENSA1 的後續版本。ENSA1-based叢集支援雙節點架構，當發生錯誤時，ASCS/SCS 執行個體會容錯移轉至 ERS。此限制源自於 ASCS/SCS 執行個體在容錯移轉後如何

從 ERS 節點的共用記憶體中重新取得鎖定資料表資訊。ENSA2-based叢集搭配 Enqueue Replicator 2 可消除此限制，因為 ASCS/SCS 執行個體可以透過網路從 ERS 執行個體收集鎖定資訊。ENSA2-based叢集可以有兩個以上的節點，因為 ASCS/SCS 執行個體不再需要容錯移轉至 ERS 節點。（不過，在雙節點 ENSA2 叢集環境中，ASCS/SCS 執行個體仍會容錯移轉至 ERS 節點，因為叢集中沒有其他節點可容錯移轉。）從 SAP Kernel 7.50 開始支援 ENSA2，但有一些限制。對於支援佇列複寫器 2 的 HA 設定，最低需求為 NetWeaver 7.52（請參閱 SAP OSS 備註 2630416)。S/4HANA 1809 隨附預設建議的 ENSA2 架構，而 S/4HANA 僅支援從 2020 版開始的 ENSA2。

自動化和擴展

目標架構中的 HA 叢集可讓 ASCS 自動容錯移轉至其他節點。

移至 ENSA2-based叢集的案例

升級至 ENSA2-based叢集有兩種主要案例：

- 案例 1：假設您的 SAP 版本和核心版本支援 ENSA2，您可以選擇在沒有隨附的 SAP 升級或 S/4HANA 轉換的情況下升級至 ENSA2。
- 案例 2：您使用 SUM 移至 ENSA2 作為升級或轉換的一部分（例如，移至 S/4HANA 1809 或更新版本）。

Epics 區段涵蓋這兩個案例的步驟。第一個案例需要您手動設定 SAP 相關參數，才能變更 ENSA2 的叢集組態。在第二個案例中，二進位檔和 SAP 相關參數是由 SUM 部署，而您剩下的唯一任務是更新 HA 的叢集組態。我們仍建議您在使用 SUM 之後驗證 SAP 參數。在大多數情況下，S/4HANA 轉換是叢集升級的主要原因。

工具

- 對於作業系統套件管理員，我們建議使用 Zypper（適用於 SLES) 或 YUM（適用於 RHEL) 工具。
- 針對叢集管理，我們建議使用 crm（適用於 SLES) 或 pcs（適用於 RHEL) shell。
- SAP 執行個體管理工具，例如 SAPControl。
- （選用）S/4HANA 轉換升級的 SUM 工具。

最佳實務

- 如需在 AWS 上使用 SAP 工作負載的最佳實務，請參閱 AWS Well-Architected Framework 的 SAP Lens。

- 考慮 ENSA2 多節點架構中的叢集節點數量（舊節點或甚至節點）。
- 設定 SLES 15 的 ENSA2 叢集，以符合 SAP S/4-HA-CLU 1.0 認證標準。
- 升級至 ENSA2 之前，請務必儲存或備份現有的叢集和應用程式狀態。

史詩

為 ENSA2 手動設定 SAP 參數（僅限案例 1)

任務	Description	所需的技能

在預設設定檔中設定參數。	如果您想要在保持相同 SAP 版本時升級至 ENSA2，或目標版本預設為 ENSA1，請將預設設定檔 (DEFAULT.PFL 檔案）中的參數設定為下列值。
	`enq/enable=TRUE`
	`enq/serverhost=sapascsvirt`
	`enq/serverinst=10`
	`(instance number of ASCS/SCS instance)`
	`enque/process_location=REMOTESA`
	`enq/replicatorhost=sapersvirt`
	`enq/replicatorinst=11`
	`(instance number of ERS instance)`
	其中 `sapascsvirt` 是 ASCS 執行個體的虛擬主機名稱，而 `sapersvirt` 是 ERS 執行個體的虛擬主機名稱。您可以變更這些項目以符合您的目標環境。
		SAP

任務	Description	所需的技能
	Note 若要使用此升級選項，您的 SAP 版本和核心版本必須支援 ENSA2 和 Enqueue Replicator 2。	

任務	Description	所需的技能

設定 ASCS/SCS 執行個體描述檔。	如果您想要在保持相同 SAP 版本時升級至 ENSA2，或目標版本預設為 ENSA1，請在 ASCS/SCS 執行個體描述檔中設定下列參數。
	
	定義 ENSA1 的設定檔區段如下所示。
	
	`#----------`
	`Start SAP enqueue server`
	`#----------`
	`_EN = en.sap$(SAPSYSTEMNAME)$(INSTANCE_NAME)`
	`Execute_04 = local rm -f $(_EN)`
	`Execute_05 = local ln -s -f $(DIR_EXECUTABLE)/enserver$(FT_EXE) $(_EN)`
	`Start_Program_01 = local $(_EN) pf=$(_PF)`
	
	若要為 ENSA2 重新設定本節：
	
	1. _ENQ 根據來自 SAP 的最新資訊，將_EN程式字首變更為 (OSS 備註 2501860；
		SAP

任務	Description	所需的技能
	需要 SAP ONE Support Launchpad 使用者帳戶)。 2. 將佇列伺服器的二進位檔從 enserver變更為 enq_server 。 3. 將新參數enq/server/replication/enable 設定為 TRUE。 4. 確定 Autostart = 0。 變更後，此設定檔區段看起來會類似以下內容。 `#------------------ ------------------ ------------------ --------- Start SAP enqueue server #------------------ ------------------ ------------------ ------- _ENQ = enq.sap$(SAPSYSTEMNAME)$(INSTANCE_NAME) Execute_04 = local rm -f $(_ENQ) Execute_05 = local ln -s -f $(DIR_EXECUTABLE)/enq_server$(FT_EXE) $(_ENQ) Start_Program_01 = local $(_ENQ) pf=$(_PF) ... enq/server/replication/enable = TRUE`	

任務	Description	所需的技能
	`Autostart = 0` **Important** _ENQ 不得啟用重新啟動選項。如果 RestartProgram_01 設定為 _ENQ，請將其變更為 StartProgram_01 。這可防止 SAP 重新啟動服務或干擾叢集管理的資源。	

任務	Description	所需的技能

設定 ERS 設定檔。	如果您想要在保持相同 SAP 版本時升級至 ENSA2，或目標版本預設為 ENSA1，請在 ERS 執行個體設定檔中設定下列參數。
	
	尋找定義佇列複寫器的區段。它將類似於以下內容。
	
	`#----------`
	`----------`
	`----------`
	`Start enqueue replicati`
	`on server`
	`#----------`
	`----------`
	`----------`
	`_ER = er.sap$(S`
	`APSYSTEMNAME)$(INS`
	`TANCE_NAME)`
	`Execute_03 = local rm -`
	`f $(_ER)`
	`Execute_04 = local ln -`
	`s -f $(DIR_EXECUTABLE)/`
	`enrepserver$(FT_EXE)`
	` $(_ER)`
	`Start_Program_00 =`
	` local $(_ER) pf=$(_PF)`
	` NR=$(SCSID)`
	
	若要為 Enqueue Replicator 2 重新設定本節：
	
	1. _ENQR 根據 SAP 的最新備註，將_ER程式字首變更為 (OSS 備註 2501860；
		SAP

任務	Description	所需的技能
	需要 SAP ONE Support Launchpad 使用者帳戶)。 2. 將佇列複寫器的二進位檔變更為 `enq_replicator` ，而非 `enrepserver` 。 3. 確定 `Autostart = 0`。 變更後，此設定檔區段看起來應該類似以下內容。 `#----------------- ----------------- ------------------- Start enqueue replicati on server #------------------- ------------------ ----------------- _ENQR = enqr.sap$ (SAPSYSTEMNAME)$(I NSTANCE_NAME) Execute_01 = local rm - f $(_ENQR) Execute_02 = local ln - s -f $(DIR_EXECUTABLE)/ enq_replicator$(FT _EXE) $(_ENQR) Start_Program_00 = local $(_ENQR) pf= $(_PF) NR=$(SCSID) … Autostart = 0`  **⚠ Important** _ENQR 不得啟用重新啟動選項。如果 `RestartPr`	

任務	Description	所需的技能
	ogram_01 設定為 _ENQR，請將其變更為 StartProgram_01 。這可防止 SAP 重新啟動服務或干擾叢集管理的服務。	

重新啟動 SAP Start Services。	變更本史詩先前所述的設定檔後，請重新啟動 ASCS/SCS 和 ERS 的 SAP Start Services。
	
	`sapcontrol -nr 10 -function RestartService SCT`
	
	`sapcontrol -nr 11 -function RestartService SCT`
	
	其中 SCT是指 SAP 系統 ID，並假設 10 和 11 分別是 ASCS/SCS 和 ERS 執行個體的執行個體編號。
		SAP

重新設定 ENSA2 的叢集 （兩個案例都需要）

任務	Description	所需的技能
驗證 SAP 資源代理程式中的版本編號。	當您使用 SUM 將 SAP 升級到 S/4HANA 1809 或更新版本時，SUM 會處理 SAP 設定檔中的參數變更。只有叢集需要手動調整。不過，我們建議您先驗證參數設定，再對叢集進行任何變更。	AWS 系統管理員

任務	Description	所需的技能

	Note
	此史詩中的範例假設您使用的是 SUSE 作業系統。如果您使用的是 RHEL，您將需要使用 YUM 和 pcs shell 等工具，而不是 Zypper 和 crm。
	
	檢查架構中的兩個節點，確認`resource-agents` 套件符合 SAP 建議的最低版本。對於 SLES，請檢查 SAP OSS 備註 2641019。對於 RHEL，請檢查 SAP OSS 備註 2641322。(SAP 備註需要 SAP ONE Support Launchpad 使用者帳戶。)
	
	`sapers:sctadm 23>`
	`zypper search -s -i`
	`resource-agents`
	`Loading repository`
	`data...`
	`Reading installed`
	`packages...`
	`S | Name | Type |`
	`Version | Arch |`
	`Repository`
	`--+-------------`
	`----+---------+---`
	`------------------`
	`---------------+--`
	`------+-----------`
	`------------------`	

任務	Description	所需的技能
	`i | resource-agents | package | 4.8.0+git30.d0077df0-150300.8.28.1 | x86_64 | SLE-Product-HA15-SP3-Updates` 視需要更新resource-agents 版本。	
備份叢集組態。	備份 CRM 叢集組態，如下所示。 `crm configure show > /tmp/cluster_config_backup.txt`	AWS 系統管理員
設定維護模式。	將叢集設定為維護模式。 `crm configure property maintenance-mode="true"`	AWS 系統管理員

任務	Description	所需的技能

檢查叢集組態。	檢查目前的叢集組態。
	
	`crm configure show`
	
	以下是完整輸出的摘錄：
	
	`node 1: sapascs`
	`node 2: sapers`
	`...`
	`primitive rsc_sap_S`
	`CT_ASCS10 SAPInstance`
	` \`
	`operations $id=rsc_s`
	`ap_SCT_ASCS10-oper`
	`ations \`
	`op monitor interval=120`
	` timeout=60 on-fail=r`
	`estart \`
	`params InstanceN`
	`ame=SCT_ASCS10_sap`
	`ascsvirt START_PRO`
	`FILE="/sapmnt/SCT/`
	`profile/SCT_ASCS10`
	`_sapascsvirt" \`
	`  AUTOMATIC_RECOVER=`
	`false \`
	`meta resource-stickines`
	`s=5000 failure-t`
	`imeout=60 migration-`
	`threshold=1 priority=`
	`10`
	`primitive rsc_sap_S`
	`CT_ERS11 SAPInstance \`
	`operations $id=rsc_s`
	`ap_SCT_ERS11-opera`
	`tions \`
	`op monitor interval=120`
	` timeout=60 on-fail=r`
	`estart \`
	`params InstanceN`
	`ame=SCT_ERS11_sape`
		AWS 系統管理員

任務	Description	所需的技能
	rsvirt START_PRO FILE="/sapmnt/SCT/ profile/SCT_ERS11_ sapersvirt" \ AUTOMATIC_RECOVER= false IS_ERS=true \ meta priority=1000 ... colocation col_sap_S CT_no_both -5000: grp_SCT_ERS11 grp_SCT_ASCS10 location loc_sap_S CT_failover_to_ers rsc_sap_SCT_ASCS10 \ rule 2000: runs_ers_SCT eq 1 order ord_sap_S CT_first_start_asc s Optional: rsc_sap_S CT_ASCS10:start rsc_sap_SCT_ERS11: stop symmetrical=false ... 其中 sapascsvirt 是指 ASCS 執行個體的虛擬主機名稱，sapersvirt 是指 ERS 執行個體的虛擬主機名稱，而 SCT是指 SAP 系統 ID。	

任務	Description	所需的技能
移除容錯移轉主機代管限制條件。	在上述範例中，位置限制條件`loc_sap_SCT_failover_to_ers` 指定 ASCS 的 ENSA1 功能在容錯移轉時應一律遵循 ERS 執行個體。使用 ENSA2，ASCS 應該能夠自由容錯移轉至任何參與的節點，因此您可以移除此限制條件。 `crm configure delete loc_sap_SCT_failover_to_ers`	AWS 系統管理員

任務	Description	所需的技能

調整基本概念。	您也需要對 ASCS 和 ERS SAPInstance 基本概念進行次要變更。
	
	以下是針對 ENSA1 設定的 ASCS SAPInstance 基本範例。
	
	`primitive rsc_sap_SCT_ASCS10 SAPInstance \`
	`operations $id=rsc_sap_SCT_ASCS10-operations \`
	`op monitor interval=120 timeout=60 on-fail=restart \`
	`params InstanceName=SCT_ASCS10_sapascsvirt START_PROFILE="/sapmnt/SCT/profile/SCT_ASCS10_sapascsvirt" \`
	`AUTOMATIC_RECOVER=false \`
	`meta resource-stickiness=5000 failure-timeout=60 migration-threshold=1 priority=10`
	
	若要升級至 ENSA2，請將此組態變更為以下內容。
	
	`primitive rsc_sap_SCT_ASCS10 SAPInstance \`
		AWS 系統管理員

任務	Description	所需的技能
	operations $id=rsc_sap_SCT_ASCS10-operations \ op monitor interval=120 timeout=60 on-fail=restart \ params InstanceName=SCT_ASCS10_sapascsvirt START_PROFILE="/sapmnt/SCT/profile/SCT_ASCS10_sapascsvirt" \ AUTOMATIC_RECOVER=false \ meta resource-stickiness=3000 這是針對 ENSA1 設定的 ERS SAPInstance 基本範例。 primitive rsc_sap_SCT_ERS11 SAPInstance \ operations $id=rsc_sap_SCT_ERS11-operations \ op monitor interval=120 timeout=60 on-fail=restart \ params InstanceName=SCT_ERS11_sapersvirt START_PROFILE="/sapmnt/SCT/profile/SCT_ERS11_sapersvirt" \ AUTOMATIC_RECOVER=false IS_ERS=true \ meta priority=1000 若要升級至 ENSA2，請將此組態變更為以下內容。	

任務	Description	所需的技能

	`primitive rsc_sap_SCT_ERS11 SAPInstance \ operations $id=rsc_sap_SCT_ERS11-operations \ op monitor interval=120 timeout=60 on-fail=restart \ params InstanceName=SCT_ERS11_sapersvirt START_PROFILE="/sapmnt/SCT/profile/SCT_ERS11_sapersvirt" \ AUTOMATIC_RECOVER=false IS_ERS=true`
	
	您可以透過各種方式變更基本概念。例如，您可以在 vi 等編輯器中修改它們，如下列範例所示。
	
	`crm configure edit rsc_sap_SCT_ERS11`	

停用維護模式。	在叢集上停用維護模式。 `crm configure property maintenance-mode="false"` 當叢集停止維護模式時，它會嘗試使用新的 ENSA2 設定讓 ASCS 和 ERS 執行個體上線。	AWS 系統管理員

（選用）新增叢集節點

任務	Description	所需的技能
檢閱最佳實務。	在新增更多節點之前，請務必了解最佳實務，例如使用奇數或甚至數量的節點。	AWS 系統管理員
新增節點。	新增更多節點涉及一系列任務，例如更新作業系統、安裝符合現有節點的軟體套件，以及提供掛載。您可以使用 SAP 軟體佈建管理員 (SWPM) 中的準備其他主機選項來建立主機的 SAP 特定基準。如需詳細資訊，請參閱下一節中列出的 SAP 指南。	AWS 系統管理員

相關資源

SAP 和 SUSE 參考

若要存取 SAP Notes，您必須擁有 SAP ONE Support Launchpad 使用者帳戶。如需詳細資訊，請參閱 SAP 支援網站。

- SAP Note 2501860 – SAP NetWeaver Application Server for ABAP 7.52 文件
- SAP Note 2641019 – 在 SUSE HA 環境中安裝 ENSA2 並從 ENSA1 更新至 ENSA2
- SAP Note 2641322 – 使用適用於 SAP 的 Red Hat HA 解決方案時，安裝 ENSA2 並從 ENSA1 更新至 ENSA2
- SAP Note 2711036 – 在 HA 環境中使用獨立佇列伺服器 2
- 獨立 Enqueue Server 2 (SAP 文件）
- SAP S/4 HANA – 佇列複寫 2 高可用性叢集 - 設定指南 (SUSE 文件）

AWS 參考

- AWS 上的 SAP HANA：SLES 和 RHEL 的高可用性組態指南

- SAP Lens - AWS Well-Architected Framework

在不同 AWS 帳戶中使用 VPCs 中的一致可用區域

Adam Spicer，Amazon Web Services

總結

在 Amazon Web Services (AWS) 雲端上，可用區域的名稱可能會因您的 AWS 帳戶和識別其位置的可用區域 ID (AZ ID) 而有所不同。如果您使用 AWS CloudFormation 建立虛擬私有雲端 (VPCs)，則必須在建立子網路時指定可用區域的名稱或 ID。如果您在多個帳戶中建立 VPCs，則會隨機選取可用區域名稱，這表示子網路在每個帳戶中使用不同的可用區域。

若要跨帳戶使用相同的可用區域，您必須將每個帳戶中的可用區域名稱對應至相同的可用區域 ID。例如，下圖顯示 `use1-az6` AZ ID 在 `us-east-1a` AWS 帳戶 A 和 AWS 帳戶 Z `us-east-1c`中命名。

此模式透過提供跨帳戶、可擴展的解決方案，以便在子網路中使用相同的可用區域，協助確保區域一致性。區域一致性可確保您的跨帳戶網路流量避免跨可用區域網路路徑，這有助於降低資料傳輸成本並降低工作負載之間的網路延遲。

此模式是 AWS CloudFormation AvailabilityZoneId 屬性的替代方法。

先決條件和限制

先決條件

- 同一 AWS 區域中至少有兩個作用中的 AWS 帳戶。
- 評估需要多少可用區域來支援區域中的 VPC 需求。
- 識別並記錄您需要支援的每個可用區域的 AZ ID。如需詳細資訊，請參閱 AWS Resource Access Manager 文件中的 AWS 資源可用區域 IDs。
- 以逗號分隔的有序 AZ IDs 清單。例如，您清單中的第一個可用區域會映射為 `az1`，第二個可用區域會映射為 `az2`，而此映射結構會持續到您的逗號分隔清單完全映射為止。沒有可映射的 AZ IDs 數目上限。
- 來自 GitHub 多帳戶可用區域映射儲存庫`az-mapping.yaml`的檔案，複製到您的本機電腦

Architecture

下圖顯示部署在 帳戶中並建立 AWS Systems Manager 參數存放區值的架構。當您在帳戶中建立 VPC 時，會使用這些參數存放區值。

該圖顯示以下工作流程：

1. 此模式的解決方案會部署到需要 VPC 區域一致性的所有帳戶。
2. 解決方案會為每個 AZ ID 建立參數存放區值，並存放新的可用區域名稱。
3. AWS CloudFormation 範本使用存放在每個參數存放區值中的可用區域名稱，這可確保區域一致性。

下圖顯示使用此模式的解決方案建立 VPC 的工作流程。

該圖顯示以下工作流程：

1. 提交範本以建立 VPC 至 AWS CloudFormation。
2. AWS CloudFormation 會解析每個可用區域的參數存放區值，並傳回每個可用區域 ID 的可用區域名稱。
3. 使用區域一致性所需的正確 AZ IDs 建立 VPC。

部署此模式的解決方案之後，您可以建立參考參數存放區值的子網路。如果您使用 AWS CloudFormation，您可以從下列 YAML 格式的範例程式碼參考可用區域映射參數值：

```
Resources:
    PrivateSubnet1AZ1:
        Type: AWS::EC2::Subnet
        Properties:
            VpcId: !Ref VPC
            CidrBlock: !Ref PrivateSubnetAZ1CIDR
            AvailabilityZone:
                !Join
                    - ''
                    - - '{{resolve:ssm:/az-mapping/az1:1}}'
```

此範例程式碼包含在 GitHub 多帳戶可用區域映射儲存庫的 `vpc-example.yaml` 檔案中。它說明如何建立符合參數存放區值的 VPC 和子網路，以實現區域一致性。

技術堆疊

- AWS CloudFormation
- AWS Lambda
- AWS Systems Manager 參數存放區

自動化和擴展

您可以使用 AWS CloudFormation StackSets 或 Customizations for AWS Control Tower 解決方案，將此模式部署到所有 AWS 帳戶。如需詳細資訊，請參閱 AWS CloudFormation 文件中的使用 AWS CloudFormation StackSets，以及 AWS 解決方案程式庫中的 AWS Control Tower 自訂。

部署 AWS CloudFormation 範本之後，您可以更新範本以使用參數存放區值，並在管道中或根據您的需求部署 VPCs。

工具

AWS 服務

- AWS CloudFormation 可協助您建立模型並設定 AWS 資源、快速一致地佈建資源，以及在整個生命週期中進行管理。您可以使用範本來描述資源及其相依性，並將它們一起啟動和設定為堆疊，而不是個別管理資源。您可以管理和佈建跨多個 AWS 帳戶和 AWS 區域的堆疊。
- AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。只需為使用的運算時間支付費用，一旦未執行程式碼，就會停止計費。
- AWS Systems Manager 參數存放區是 AWS Systems Manager 的功能。它為組態資料管理和秘密管理提供安全的階層式儲存。

Code

此模式的程式碼會在 GitHub 多帳戶可用區域映射儲存庫中提供。

史詩

部署 az-mapping.yaml 檔案

任務	Description	所需的技能
判斷區域所需的可用區域。	1. 確定必須在您的區域中一致使用的 AZ IDs。	雲端架構師

任務	Description	所需的技能
	2. 將這些 AZ IDs 記錄在逗號分隔的清單中，並依照您希望套用的順序進行。例如，您清單中的第一個可用區域會映射為 `az1`，第二個則映射為 `az2`。沒有可映射的 AZ IDs 數目上限。	
部署 az-mapping.yaml 檔案。	使用 `az-mapping.yaml` 檔案在所有必要的 AWS 帳戶中建立 AWS CloudFormation 堆疊。在 `AZIds` 參數中，使用您先前建立的逗號分隔清單。 我們建議您使用 AWS CloudFormation StackSets 或 Customizations for AWS Control Tower Solution。	雲端架構師

在帳戶中部署 VPCs

任務	Description	所需的技能
自訂 AWS CloudFormation 範本。	當您使用 AWS CloudFormation 建立子網路時，自訂範本以使用您先前建立的參數存放區值。 如需範例範本，請參閱 GitHub 多帳戶可用區域映射儲存庫中的 `vpc-example.yaml` 檔案。	雲端架構師
部署 VPCs。	將自訂的 AWS CloudFormation 範本部署到您的帳戶。區	雲端架構師

任務	Description	所需的技能
	域中的每個 VPC 接著會在子網路使用的可用區域中具有區域一致性	

相關資源

- AWS 資源的可用區域 IDs (AWS Resource Access Manager 文件）
- AWS::EC2::Subnet (AWS CloudFormation 文件）

在 IAM 政策中使用使用者 IDs 進行存取控制和自動化

Srinivas Ananda Babu 和 Ram Kandaswamy，Amazon Web Services

總結

此模式說明在 AWS Identity and Access Management (IAM) 中使用使用者名稱型政策的潛在陷阱、使用使用者 IDs的好處，以及如何將此方法與 整合 AWS CloudFormation 以進行自動化。

在 中 AWS 雲端，IAM 服務可協助您精確管理使用者身分和存取控制。不過，依賴 IAM 政策建立中的使用者名稱可能會導致無法預見的安全風險和存取控制問題。例如，請考慮此案例：新員工 John Doe 加入您的團隊，而您建立使用者名稱為 的 IAM 使用者帳戶`j.doe`，透過參考使用者名稱的 IAM 政策授予他們許可。當 John 離開公司時，帳戶會被刪除。當新員工 Jane Doe 加入您的團隊並重新建立`j.doe`使用者名稱時，問題就會開始。現有政策現在授予 Jane Doe 與 John Doe 相同的許可。這會產生潛在的安全和合規噩遊。

手動更新每個政策以反映新的使用者詳細資訊是一個耗時、容易出錯的程序，尤其是隨著組織的成長。解決方案是使用唯一且不變的使用者 ID。當您建立 IAM 使用者帳戶時， 會 AWS 為 IAM 使用者指派唯一的使用者 ID （或主體 ID)。您可以在 IAM 政策中使用這些使用者 IDs，以確保不受使用者名稱變更或重複使用影響的一致且可靠的存取控制。

例如，使用使用者 ID 的 IAM 政策可能如下所示：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::example-bucket",
            "Principal": { "AWS": "arn:aws:iam::123456789012:user/abcdef01234567890" }
        }
      ]
}
```

在 IAM 政策中使用使用者 IDs 的優點包括：

- 唯一性。使用者 IDs 在所有 中都是唯一的 AWS 帳戶，因此它們提供正確且一致的許可應用程式。
- 抗擾性。使用者 IDs 無法變更，因此它們提供在政策中參考使用者的穩定識別符。
- 稽核和 compliance. AWS 服務 通常在日誌和稽核追蹤中包含使用者 IDs，這可讓您輕鬆地將動作追蹤回特定使用者。

- 自動化和整合。使用 AWS APIs、 SDKs 或自動化指令碼中的使用者 IDs，可確保程序不受使用者名稱變更的影響。
- 面向未來。從一開始在政策中使用使用者 IDs 可以防止潛在的存取控制問題或廣泛的政策更新。

自動化

當您使用基礎設施做為程式碼 (IaC) 工具 AWS CloudFormation，例如，使用者名稱型 IAM 政策的陷阱仍可能導致問題。當您呼叫`Ref`內部 函數時，IAM 使用者資源會傳回使用者名稱。隨著組織的基礎設施演進，如果您重複使用使用者名稱，建立和刪除資源的週期，包括 IAM 使用者帳戶，可能會導致意外的存取控制問題。

若要解決此問題，建議您將使用者 IDs納入 CloudFormation 範本。不過，為此目的取得使用者 IDs 可能具有挑戰性。這就是自訂資源可以提供幫助的地方。您可以使用 CloudFormation 自訂資源，透過整合 AWS APIs或外部服務來擴展服務的功能。透過建立擷取指定 IAM 使用者之使用者 ID 的自訂資源，您可以在 CloudFormation 範本中提供使用者 ID。此方法可簡化參考使用者 IDs的程序，並確保您的自動化工作流程保持強大且符合未來需求。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 雲端管理員執行 CloudFormation 範本的 IAM 角色

限制

- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

Architecture

目標架構

下圖顯示 如何使用 支援的 CloudFormation 自訂資源 AWS Lambda 來擷取 IAM 使用者 ID。

自動化和擴展

您可以針對不同的 AWS 區域 和 帳戶多次使用 CloudFormation 範本。您只需要在每個區域或帳戶中執行一次。

工具

AWS 服務

- IAM – AWS Identity and Access Management (IAM) 是一種 Web 服務，可協助您安全地控制對 AWS 資源的存取。您可以使用 IAM 來控制能通過身分驗證 (登入) 和授權使用資源的 (具有許可) 的人員。
- AWS CloudFormation – AWS CloudFormation 可協助您建立和設定 AWS 資源的模型，以減少管理這些資源的時間，並有更多時間專注於執行的應用程式 AWS。您可以建立描述所需 AWS 資源的範本，CloudFormation 會為您佈建和設定這些資源。
- AWS Lambda – AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。

最佳實務

如果您是從頭開始或規劃綠地部署，強烈建議您使用 AWS IAM Identity Center進行集中式使用者管理。IAM Identity Center 會與您現有的身分提供者 （ 例如 Active Directory 或 Okta) 整合，以聯合使用者身分 AWS，因此不需要直接建立和管理 IAM 使用者。這種方法不僅可確保一致的存取控制，還簡化了使用者生命週期管理，並有助於增強整個 AWS 環境的安全性和合規性。

史詩

驗證許可

任務	Description	所需的技能
驗證您的 AWS 帳戶 和 IAM 角色。	確認您的 IAM 角色具有在 中部署 CloudFormation 範本的許可 AWS 帳戶。 如果您打算在此程序的最後一個步驟中使用 AWS CLI 而非 CloudFormation 主控台來部署範本，您也應該設定暫時登入	雲端架構師

任務	Description	所需的技能
	資料來執行 AWS CLI 命令。如需說明，請參閱 IAM 文件。	

建置 CloudFormation 範本

任務	Description	所需的技能
建立 CloudFormation 範本。	1. 遵循 CloudFormation 文件中的指示建立 CloudFormation 範本。您可以使用 JSON 或 YAML 格式。此模式假設您使用 YAML 格式。 2. 儲存名為 的範本`get_unique_user_id.yaml` 。	AWS DevOps，雲端架構師

新增使用者名稱的輸入參數。	將下列程式碼新增至 CloudFormation 範本的 `Parameters` 區段：
	`Parameters:`
	`  NewIamUserName:`
	`  Type: String`
	`  Description: Unique username for the new IAM user`
	此參數會提示使用者輸入使用者名稱。
		AWS DevOps，雲端架構師

新增自訂資源以建立 IAM 使用者。	將下列程式碼新增至 CloudFormation 範本的 `Resources` 區段：
	`Resources:`
	`  rNewIamUser:`
		AWS DevOps，雲端架構師

任務	Description	所需的技能

	<pre> Type: 'AWS::IAM
	::User'
	    Properties:
	      UserName: !Ref
	 NewIamUserName</pre>
	此程式碼會新增 CloudFormation 資源，以 NewIamUserName 參數提供的名稱建立 IAM 使用者。	

新增 Lambda 函數的執行角色。	在此步驟中，您會建立 IAM 角色，授予 AWS Lambda 函數取得 IAM 的許可UserId。指定下列 Lambda 執行所需的最低許可： • logs:CreateLogStream • logs:PutLogEvents • CreateLogGroup • iam:GetUser • 適用於 lambda.amazonaws.com 的 AssumeRole 如需建立執行角色的說明，請參閱 Lambda 文件。當您建立 Lambda 函數時，您將在下一個步驟中參考此角色。	AWS 管理員、雲端架構師

任務	Description	所需的技能

新增 Lambda 函數以取得唯一的 IAM `UserId`。	在此步驟中，您會定義具有 Python 執行時間的 Lambda 函數，以取得唯一的 IAM `UserId`。若要這樣做，請將下列程式碼新增至 CloudFormation 範本的 `Resources` 區段。`<<ROLENAME>>` 將取代為您在最後一個步驟中建立的執行角色名稱。
	
	`GetUserLambdaFunction:`
	`  Type: 'AWS::Lambda::Function'`
	`  Properties:`
	`  Handler: index.handler`
	`  Role: <<ROLENAME>>`
	`  Timeout: 30`
	`  Runtime: python3.11`
	`  Code:`
	`  ZipFile: |`
	`  import cfnresponse, boto3`
	`  def handler(event, context):`
	`  try:`
	`  print(event)`
	`  user = boto3.client('iam').get_user(UserName=event['ResourceProperties']['NewIamUserName'])['User']`
	`  cfnresponse.send(event,`
		AWS DevOps，雲端架構師

任務	Description	所需的技能

	<pre>context, cfnrespon
	se.SUCCESS, {'NewIamU
	serId': user['Use
	rId'], 'NewIamUs
	erPath': user['Pat
	h'], 'NewIamUserArn':
	user['Arn']})
	except
	Exception as e:
	cfnrespon
	se.send(event,
	context, cfnrespon
	se.FAILED, {'NewIamU
	ser': str(e)})</pre>	

新增自訂資源。	將下列程式碼新增至 CloudFormation 範本的 `Resources` 區段：
	<pre>rCustomGetUniqueUs
	erId:
	Type: 'Custom::
	rCustomGetUniqueUs
	erIdWithLambda'
	Properties:
	ServiceToken: !
	GetAtt GetUserLa
	mbdaFunction.Arn
	NewIamUserName: !
	Ref NewIamUserName</pre>
	此自訂資源會呼叫 Lambda 函數以取得 IAM `UserID`。
		AWS DevOps，雲端架構師

任務	Description	所需的技能

定義 CloudFormation 輸出。	將下列程式碼新增至 CloudFormation 範本的 `Outputs`區段：
	
	`Outputs:`
	`  NewIamUserId:`
	`  Value: !GetAtt`
	`  rCustomGetUniqueUs`
	`erId.NewIamUserId`
	
	這會顯示`UserID`新 IAM 使用者的 IAM。
		AWS DevOps，雲端架構師

儲存範本。	將變更儲存至 CloudFormation 範本。	AWS DevOps，雲端架構師

部署 CloudFormation 範本

任務	Description	所需的技能

部署 CloudFormation 範本。	若要使用 CloudFormation 主控台部署`get_uniqu`
	`e_user_id.yaml` 範本，請遵循 CloudFormation 文件中的指示。
	
	或者，您可以執行下列 AWS CLI 命令來部署範本：
	
	`aws cloudformation`
	`  create-stack \`
	`--stack-name DemoNewUs`
	`er \`
	`--template-body file://`
	`get_unique_user_id.y`
	`aml \`
		AWS DevOps，雲端架構師

任務	Description	所需的技能
	`--parameters Parameter Key=NewIamUserName ,ParameterValue=de mouser \ --capabilities CAPABILITY_NAMED_IAM`	

相關資源

- 從 CloudFormation 主控台建立堆疊 (CloudFormation 文件)
- Lambda 支援的自訂資源 (CloudFormation 文件)
- 唯一識別符 (IAM 文件)
- 搭配 AWS 資源使用臨時登入資料 (IAM 文件)

在本機驗證帳戶工廠的 Terraform (AFT) 程式碼

Alexandru Pop 和 Michal Gorniak，Amazon Web Services

總結

此模式說明如何在本機測試由 AWS Control Tower Account Factory for Terraform (AFT) 管理的 HashiCorp Terraform 程式碼。Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。AFT 會設定 Terraform 管道，協助您 AWS 帳戶 佈建和自訂多個管道 AWS Control Tower。

在程式碼開發期間，在 AFT 管道外部以程式碼 (IaC) 形式在本機測試 Terraform 基礎設施會很有幫助。此模式顯示如何執行下列動作：

- 擷取儲存在 AFT 管理帳戶中 AWS CodeCommit 儲存庫中 Terraform 程式碼的本機副本。
- 使用擷取的程式碼在本機模擬 AFT 管道。

此程序也可以用來執行不屬於正常 AFT 管道的 Terraform 命令。例如，您可以使用此方法執行命令，例如 `terraform validate`、`terraform destroy`、 `terraform plan`和 `terraform import`。

先決條件和限制

先決條件

- 使用 的作用中 AWS 多帳戶環境 AWS Control Tower
- 完全部署的 AFT 環境
- AWS Command Line Interface (AWS CLI)，已安裝並設定
- AWS CLI 的登入資料協助程式 AWS CodeCommit、安裝和設定
- Python 3.x
- 在本機電腦上安裝和設定的 Git
- `git-remote-commit` 公用程式，已安裝和設定
- 安裝並設定 Terraform （本機 Terraform 套件版本必須與 AFT 部署中使用的版本相符）

限制

- 此模式不包含 AWS Control Tower、AFT 或任何特定 Terraform 模組所需的部署步驟。

- 在此程序期間在本機產生的輸出不會儲存在 AFT 管道執行時間日誌中。

Architecture

目標技術堆疊

- 部署內部署的 AFT 基礎設施 AWS Control Tower
- Terraform
- Git
- AWS CLI 第 2 版

自動化和擴展

此模式示範如何針對單一 AFT 受管的 AFT 全域帳戶自訂，在本機叫用 Terraform 程式碼 AWS 帳戶。驗證 Terraform 程式碼之後，您可以將其套用至多帳戶環境中的其餘帳戶。如需詳細資訊，請參閱 AWS Control Tower 文件中的重新叫用自訂。

您也可以使用類似的程序，在本機終端機中執行 AFT 帳戶自訂。若要從 AFT 帳戶自訂本機叫用 Terraform 程式碼，請在 AFT 管理帳戶中從 CodeCommit 複製 aft-account-customizations 儲存庫，而非 aft-global-account-customizations 儲存庫。

工具

AWS 服務

- AWS Control Tower 可協助您設定和管理 AWS 多帳戶環境，並遵循規範最佳實務。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。

其他服務

- HashiCorp Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。
- Git 是開放原始碼的分散式版本控制系統。

Code

以下是範例 bash 指令碼，可用於在本機執行由 AFT 管理的 Terraform 程式碼。若要使用指令碼，請遵循此模式的 Epics 區段中的指示。

```
#! /bin/bash
# Version: 1.1 2022-06-24 Unsetting AWS_PROFILE since, when set, it interferes with
 script operation
#          1.0 2022-02-02 Initial Version
#
# Purpose: For use with AFT: This script runs the local copy of TF code as if it were
 running within AFT pipeline.
#        * Facilitates testing of what the AFT pipline will do
#           * Provides the ability to run terraform with custom arguments (like 'plan'
 or 'move') which are currently not supported within the pipeline.
#
# © 2021 Amazon Web Services, Inc. or its affiliates. All Rights Reserved.
# This AWS Content is provided subject to the terms of the AWS Customer Agreement
# available at http://aws.amazon.com/agreement or other written agreement between
# Customer and either Amazon Web Services, Inc. or Amazon Web Services EMEA SARL or
 both.
#
# Note: Arguments to this script are passed directly to 'terraform' without parsing nor
 validation by this script.
#
# Prerequisites:
#    1. local copy of ct GIT repositories
#    2. local backend.tf and aft-providers.tf filled with data for the target account
 on which terraform is to be run
#       Hint: The contents of above files can be obtain from the logs of a previous
 execution of the AFT pipeline for the target account.
#    3. 'terraform' binary is available in local PATH
#    4. Recommended: .gitignore file containing 'backend.tf', 'aft_providers.tf' so the
 local copy of these files are not pushed back to git

readonly credentials=$(aws sts assume-role \
    --role-arn arn:aws:iam::$(aws sts get-caller-identity --query "Account" --output
 text ):role/AWSAFTAdmin \
    --role-session-name AWSAFT-Session \
    --query Credentials )

unset AWS_PROFILE
export AWS_ACCESS_KEY_ID=$(echo $credentials | jq -r '.AccessKeyId')
export AWS_SECRET_ACCESS_KEY=$(echo $credentials | jq -r '.SecretAccessKey')
export AWS_SESSION_TOKEN=$(echo $credentials | jq -r '.SessionToken')
```

```
terraform "$@"
```

史詩

將範例程式碼儲存為本機檔案

任務	Description	所需的技能
將範例程式碼儲存為本機檔案。	1. 複製此模式程式碼 區段中的範例 bash 指令碼，並將其貼到程式碼編輯器中。 2. 命名檔案 `ct_terraform.sh` ，然後將檔案儲存在本機專用資料夾中，例如 `~/scripts` 或 `~/bin`。	AWS 管理員
讓範例程式碼可執行。	開啟終端機視窗，並執行下列其中一項操作來驗證您的 AWS AFT 管理帳戶： • 使用已設定存取 AFT 管理帳戶所需許可的現有AWS CLI設定檔。若要使用 設定檔，您可以執行下列命令： `export AWS_PROFILE=<aft account profile name>` • 如果您的組織使用 SSO 存取 AWS，請在組織的 SSO 頁面上輸入 AFT 管理帳戶的登入資料。 **Note** 您的組織可能也有自訂工具，可為您的 AWS	AWS 管理員

任務	Description	所需的技能
	環境提供身分驗證憑證。	

在正確的 中驗證對 AFT 管理帳戶的存取 AWS 區域。	**⚠ Important**
	請確定您使用與您向 AFT 管理帳戶驗證的相同終端機工作階段。
	
	1. 執行下列命令 AWS 區域 ，導覽至您 AFT 部署的 ：
	`export AWS_REGIO N=<aft_region>`
	2. 請確定您位於正確的 帳戶。
	a. 執行以下命令：
	`aws code-commit list-repositories`
	b. 確認輸出中列出的儲存庫符合您 AFT 管理帳戶中的儲存庫名稱。
		AWS 管理員

建立新的本機目錄來存放 AFT 儲存庫程式碼。	在相同的終端機工作階段中，執行下列命令：
	`mkdir my_aft`
	`cd my_aft`
		AWS 管理員

任務	Description	所需的技能

複製遠端 AFT 儲存庫程式碼。	1. 在本機終端機中，執行下列命令：
	`git clone codecommit::$AWS_REGION://aft-global-customizations`
	Note
	為了簡化，此程序和 AFT 僅使用主程式碼分支。若要使用程式碼分支，您也可以在此處輸入程式碼分支命令。不過，當 AFT 自動化從主分支套用程式碼時，從非主分支套用的任何變更都會復原。
	2. 導覽至複製的目錄：
	`cd aft-global-customizations/terraform`
		AWS 管理員

建立 AFT 管道在本機執行所需的 Terraform 組態檔案

任務	Description	所需的技能
開啟先前執行的 AFT 管道，並將 Terraform 組態檔案複製到本機資料夾。	**Note** AFT 管道需要在此史詩中建立的 `backend.t`	AWS 管理員

任務	Description	所需的技能
	f 和 aft-providers.tf 組態檔案，才能在本機執行。這些檔案會在雲端型 AFT 管道內自動建立，但必須手動建立，管道才能在本機執行。在本機執行 AFT 管道需要一組檔案，代表在單一 中執行管道 AWS 帳戶。 1. 使用您的 AWS Control Tower 管理帳戶登入資料，登入 AWS 管理主控台，然後開啟 AWS CodePipeline 主控台。請確定您位於部署 AFT AWS 區域 的相同位置。 2. 在左側導覽窗格中，選擇 Pipelines (管道)。 3. 選擇 ########-customizations-pipeline。(############ 是您用來在本機執行 Terraform 程式碼的 AWS 帳戶 ID。) 4. 確定最近標示的執行顯示成功值。如果值不同，您必須在 AFT 管道中重新叫用自訂項目。如需詳細資訊，請參閱 AWS Control Tower 文件中的重新叫用自訂。 5. 選擇最新的執行時間，以顯示其詳細資訊。	

任務	Description	所需的技能

	6. 在 Apply-AFT-Global-Customizations 區段中，尋找 Apply-Terraform 階段。
	7. 選取 Apply-Terraform 階段的詳細資訊區段。
	8. 尋找 Apply-Terraform 階段的執行時間日誌。
	9. 在執行時間日誌中，尋找以下列行開頭和結尾的區段：
	`"\n\n aft-providers.tf ... "\n \n backend.tf"`
	10. 複製這兩個標籤之間的輸出，並將其儲存為本機 Terraform 資料夾（終端機工作階段目前的工作目錄）`aft-providers.tf` 中名為 的本機檔案。
	自動產生的 providers.tf 陳述式範例
	`## Autogenerated providers.tf ## ## Updated on: 2022-05-31 16:27:45 ## provider "aws" { region = "us-east-2" assume_role { role_arn = "arn:aws:iam::######## :role/AWSAFTExecution"`	

任務	Description	所需的技能

	<pre> }
	  default_tags {
	    tags = {
	      managed_by
	              = "AFT"
	    }
	  }
	}</pre>
	11.在執行時間日誌中，尋找以下列行開頭和結尾的區段：
	<pre>"\n\n tf
	...
	"\n \n backend.tf"</pre>
	12.複製這兩個標籤之間的輸出，並將其儲存為本機 Terraform 資料夾（終端機工作階段目前的工作目錄）tf中名為 的本機檔案。
	
	自動產生的 backend.tf 陳述式範例
	<pre>## Autogenerated
	 backend.tf ##
	## Updated on: 2022-05-3
	1 16:27:45 ##
	terraform {
	  required_version =
	 ">= 0.15.0"
	  backend "s3" {
	    region =
	 "us-east-2"
	    bucket =
	 "aft-backend-#####
	#######-primary-re
	gion"</pre>	

任務	Description	所需的技能

	<pre>key = "############-aft-global-customizations/terraform.tfstate"
	dynamodb_table = "aft-backend-############"
	encrypt = "true"
	kms_key_id = "########-####-####-####-############"
	role_arn = "arn:aws:iam::############:role/AWSAFTExecution"
	}
	}</pre>
	Note
	backend.tf 和 aft-providers.tf 檔案繫結至特定 AWS 帳戶、AFT 部署和資料夾。這些檔案也不同，取決於它們是否位於相同 AFT 部署中的 aft-global-customizations 和 aft-account-customizations 儲存庫中。請務必從相同的執行時間清單中產生這兩個檔案。	

使用範例 bash 指令碼在本機執行 AFT 管道

任務	Description	所需的技能
實作您要驗證的 Terraform 組態變更。	1. 執行下列命令，導覽至複製的 aft-global-customizations 儲存庫： `cd aft-global-customizations/terraform` **Note** 檔案 `backend.tf` 和 `aft-providers.tf` 位於此目錄中。目錄也包含來自 aft-global-customizations 儲存庫的 Terraform 檔案。 2. 將您要在本機測試的 Terraform 程式碼變更併入組態檔案。	AWS 管理員

執行`ct_terraform.sh` 指令碼並檢閱輸出。	1. 導覽至包含 sh 指令碼的本機資料夾。
	2. 若要驗證修改後的 Terraform 程式碼，請執行下列命令來執行`ct_terraform.sh` 指令碼：
	`~/scripts/ct_terraform.sh apply`
	`terraform --help`
		AWS 管理員

任務	Description	所需的技能
	Note 您可以在此步驟期間執行任何 Terraform 命令。若要查看 Terraform 命令的完整清單，請執行下列命令： 3. 檢閱命令輸出，然後在本機偵錯程式碼變更，然後遞交變更並將其推回 AFT 儲存庫。 **Important** • 在本機進行且未推回遠端儲存庫的任何變更都是暫時的，而且可能隨時由執行中的 AFT 管道自動化復原。 • AFT 自動化可以隨時執行，因為其他使用者和 AFT 自動化觸發可以調用它。 • AFT 一律會從儲存庫的主分支套用程式碼，復原任何未遞交的變更。	

將您的本機程式碼變更推回 AFT 儲存庫

任務	Description	所需的技能

將 backend.tf 和 aft-providers.tf 檔案的參考新增至 .gitignore 檔案。	執行下列命令，將您所建立的 backend.tf 和 aft-providers.tf 檔案新增至.gitignore 檔案：
	`echo backend.tf >> .gitignore`
	`echo aft-providers.tf >>.gitignore`
	Note
	將檔案移至.gitignore 檔案可確保檔案不會遞交並推回遠端 AFT 儲存庫。
		AWS 管理員

遞交程式碼變更並將其推送至遠端 AFT 儲存庫。	1. 若要將任何新的 Terraform 組態檔案新增至儲存庫，請執行下列命令：
	`git add <filename>`
	2. 若要遞交變更並將其推送至 CodeCommitt 中的遠端 AFT 儲存庫，請執行下列命令：
	`git commit -a`
	`git push`
		AWS 管理員

任務	Description	所需的技能
	Important 您引入的程式碼會依照此程序進行變更，直到這個時間點 AWS 帳戶僅套用至一個。	

將變更推展到多個帳戶

任務	Description	所需的技能
將變更推展到由 AFT 管理的所有帳戶。	若要將變更推展到由 AFT 管理 AWS 帳戶 的多個 ，請遵循 AWS Control Tower 文件中重新叫用自訂中的指示。	AWS 管理員

更多模式

- 使用僅供讀取複本將 HA 新增至 Amazon RDS Custom 上的 Oracle PeopleSoft
- 自動稽核允許從公有 IP 地址存取 AWS 的安全群組
- 在 上使用登陸區域加速器自動建立帳戶 AWS
- 使用 AWS Systems Manager 自動化新增或更新 Windows 登錄項目
- 自動化 AWS 資源評估
- 使用 AWS CDK 自動化 AWS Service Catalog 產品組合和產品部署
- 使用 DR Orchestrator Framework 自動化跨區域容錯移轉和容錯回復
- 自動化刪除 AWS CloudFormation 堆疊和相關聯的資源
- 使用 Terraform 在 Amazon Managed Grafana 上自動化 Amazon MWAA 自訂指標的擷取和視覺化
- 在 Amazon MQ 中自動化 RabbitMQ 組態 Amazon MQ
- 使用 GitHub Actions、Artifactory 和 Terraform 在多儲存庫設定中自動化 AWS Supply Chain 資料湖的部署
- 自動化跨 的 Amazon RDS 執行個體複寫 AWS 帳戶
- 使用 Cloud Custodian 和 AWS CDK 將 Systems Manager 的 AWS 受管政策自動連接至 EC2 執行個體設定檔
- 使用 AWS CDK 自動為微服務建置 CI/CD 管道和 Amazon ECS 叢集
- 在 CodeCommit 中自動偵測變更並啟動單一儲存庫的不同 CodePipeline 管道 CodeCommit
- 建置資料管道，以使用 AWS DataOps 開發套件擷取、轉換和分析 Google Analytics 資料
- 使用 Amazon EC2 Auto Scaling 和 Systems Manager 建置 Micro Focus Enterprise Server PAC
- 使用 GitHub 動作和 Terraform 建置 Docker 映像並將其推送至 Amazon ECR
- 建置包含 MongoDB Atlas 的 AWS 登陸區域
- 使用 Terraform 在 AWS Organizations 中集中管理 IAM 存取金鑰
- 使用 Terraform 集中 AWS Organizations 中的軟體套件分佈
- 使用 在 Amazon Bedrock 中設定模型調用記錄 AWS CloudFormation
- 在 AWS 上 SQL Server 的 Always On 可用性群組中設定唯讀路由
- 使用、 AWS Amplify Angular 和 Module Federation 建立微型前端的入口網站
- 使用 GitHub Actions 和 Terragrunt 建立 API 驅動的資源協同運作架構
- 在組織中建立跨帳戶 Amazon EventBridge 連線
- 自動為 Java 和 Python 專案建立動態 CI 管道

- 使用私有端點和 Application Load Balancer 在內部網站上部署 Amazon API Gateway API
- 使用 和 AWS CDK CloudFormation 部署和管理 AWS Control Tower 控制項
- 使用 Terraform 部署和管理 AWS Control Tower 控制項
- 使用 Terraform 部署 CloudWatch Synthetics Canary
- 使用 Terraform 在 Amazon EKS 中部署 CockroachDB 叢集
- 使用 Terraform 和 DRA 部署 Lustre 檔案系統以進行高效能資料處理
- AWS 使用 Terraform 和 Amazon Bedrock 在 上部署 RAG 使用案例
- 使用 Terraform 在 AWS Wavelength 區域中部署資源
- 使用 Terraform 在 Amazon EC2 和 Amazon FSx 上部署 SQL Server 容錯移轉叢集執行個體
- 使用 Terraform 部署 AWS WAF 解決方案的安全自動化
- 偵測具有即將到期 CA 憑證的 Amazon RDS 和 Aurora 資料庫執行個體
- 記錄您的 AWS 登陸區域設計
- 從 AWS Organizations 中的整個組織將 AWS Backup 報告匯出為 CSV 檔案 AWS Organizations
- 使用 Amazon Personalize 產生個人化和重新排名的建議
- 使用 Account Factory for Terraform 管理多個帳戶的許可集
- 當 Amazon Data Firehose 資源未使用 AWS KMS 金鑰加密時，識別和提醒
- 使用引導管道實作 Account Factory for Terraform (AFT)
- 在 Amazon API Gateway 中使用自訂網域實作路徑型 API 版本控制
- 使用 Kubernetes DaemonSet 在 Amazon EKS 工作者節點上安裝 SSM Agent
- 使用 preBootstrapCommands 在 Amazon EKS 工作者節點上安裝 SSM 代理程式和 CloudWatch 代理程式
- 使用 以程式碼形式管理 AWS IAM Identity Center 許可集 AWS CodePipeline
- 使用 Terraform 動態管理 AWS 許可集
- 在多個 AWS 帳戶和 AWS 區域中管理 AWS Service Catalog 產品
- 使用 AWS CDK 設定 Amazon ECS Anywhere 來管理內部部署容器應用程式
- 使用 AWS CodePipeline 和 Amazon Bedrock 以程式碼形式管理 AWS Organizations 政策
- 將大量 DNS 記錄遷移至 Amazon Route 53 私有託管區域
- 將 Oracle PeopleSoft 遷移至 Amazon RDS Custom
- 使用 AWS MGN 將 RHEL BYOL 系統遷移至包含 AWS 授權的執行個體
- 設定最低可行的資料空間以在組織之間共用資料
- 監控 Amazon ElastiCache 叢集的靜態加密

- 使用 CloudWatch Logs Insights 監控應用程式活動
- 使用 AWS 服務監控 SAP RHEL Pacemaker 叢集
- AWS 使用 Terraform 在 上建立階層式多區域 IPAM 架構
- 使用 AWS CDK 和 GitHub Actions 工作流程最佳化多帳戶無伺服器部署
- 使用 GitHub 動作根據 AWS CloudFormation 範本佈建 AWS Service Catalog 產品
- 透過部署角色販賣機解決方案來佈建最低權限的 IAM 角色
- 使用 AWS Lambda 自動化 AWS 帳戶 AWS Managed Microsoft AD 從 移除 Amazon EC2 項目
- 使用 AWS Lambda 自動化 AWS 帳戶 從 移除相同 中的 Amazon EC2 AWS Managed Microsoft AD 項目
- 使用 Transfer Family、Amazon Cognito 和 GuardDuty 保護檔案傳輸
- 建立 IAM 使用者時傳送通知
- 為以儲存格為基礎的架構設定無伺服器儲存格路由器
- 使用 AWS CDK 和 GitLab 在 Amazon ECS Anywhere 上設定混合工作負載的 CI/CD 管道
- 使用作用中待命資料庫為 Amazon RDS Custom 上的 Oracle 電子商務套件設定 HA/DR 架構
- 在多帳戶 AWS 環境中設定混合網路的 DNS 解析
- 使用 Amazon FSx 設定 SQL Server Always On FCI 的異地同步備份基礎設施
- 在 Aurora PostgreSQL 相容上設定 Oracle UTL_FILE 功能
- 使用 Application Load Balancer 在 Amazon ECS 中使用交互 TLS 簡化應用程式身分驗證
- 使用 AWS Private CA 和 AWS RAM 簡化私有憑證管理
- 使用 SageMaker AI 和 hydra 簡化從本機開發到可擴展實驗的機器學習工作流程
- 使用 AWS Organizations 自動標記 Transit Gateway 連接
- Amazon RDS Custom for Oracle 上 Oracle PeopleSoft 應用程式的轉換角色
- 使用 Amazon Q Developer 作為編碼助理，以提高您的生產力

Web 和行動應用程式

主題

- 使用 Amazon Cognito 和 AWS Amplify UI 驗證現有的 React 應用程式使用者
- 使用 AWS Amplify 建立 React 應用程式，並使用 Amazon Cognito 新增身分驗證
- 使用、 AWS Amplify Angular 和 Module Federation 建立微型前端的入口網站
- 將以 React 為基礎的單一頁面應用程式部署至 Amazon S3 和 CloudFront
- 使用私有端點和 Application Load Balancer 在內部網站上部署 Amazon API Gateway API
- 使用 Amazon Cognito 和 IaC 自動化將 Amazon Quick Sight 視覺化元件內嵌至 Web 應用程式 IaC
- 使用 Green Boost 探索完整堆疊的雲端原生 Web 應用程式開發
- 使用 AWS CodeBuild 從 GitHub 執行 Node.js 應用程式的單元測試
- 使用 AWS Lambda 在六邊形架構中建構 Python 專案
- 更多模式

使用 Amazon Cognito 和 AWS Amplify UI 驗證現有的 React 應用程式使用者

Daniel Kozhemyako，Amazon Web Services

總結

此模式說明如何使用 AWS Amplify UI 程式庫和 Amazon Cognitouser 集區，將身分驗證功能新增至現有的前端 React 應用程式。

模式使用 Amazon Cognito 為應用程式提供身分驗證、授權和使用者管理。它還使用 Amplify UI 中的元件，這是將 功能擴展 AWS Amplify 到使用者介面 (UI) 開發的開放原始碼程式庫。驗證器 UI 元件會管理登入工作階段，並執行透過 Amazon Cognito 驗證使用者的雲端連線工作流程。

實作此模式之後，使用者可以使用下列任何登入資料來登入：

- 使用者名稱和密碼
- 社交身分提供者，例如 Apple、Facebook、Google 和 Amazon
- 與 SAML 2.0 相容或 OpenID Connect (OIDC) 相容的企業身分提供者

> **Note**
>
> 若要建立自訂身分驗證 UI 元件，您可以在無周邊模式下執行驗證器 UI 元件。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- React 18.2.0 或更新版本的 Web 應用程式
- Node.js 和 npm 6.14.4 或更新版本，已安裝

限制

- 此模式僅適用於 React Web 應用程式。
- 此模式使用預先建置的 Amplify UI 元件。解決方案不涵蓋實作自訂 UI 元件所需的步驟。

產品版本

- Amplify UI 6.1.3 或更新版本（第 1 代）
- Amplify 6.0.16 或更新版本（第 1 代）

Architecture

目標架構

下圖顯示使用 Amazon Cognito 驗證 React Web 應用程式使用者的架構。

工具

AWS 服務

- Amazon Cognito 為 Web 和行動應用程式提供身分驗證、授權和使用者管理。

其他工具

- Amplify UI 是一個開放原始碼 UI 程式庫，提供您可以連接到雲端的可自訂元件。
- Node.js 是一種事件驅動的 JavaScript 執行期環境，旨在建置可擴展的網路應用程式。
- npm 是在 Node.js 環境中執行的軟體登錄檔，用於共用或借用套件和管理私有套件的部署。

最佳實務

如果您要建立新的應用程式，建議您使用 Amplify Gen 2。

史詩

建立 Amazon Cognito 使用者集區

任務	Description	所需的技能
建立使用者集區。	建立 Amazon Cognito 使用者集區。設定使用者集區的登入選項和安全性需求，以符合您的使用案例。	應用程式開發人員

任務	Description	所需的技能
新增應用程式用戶端。	設定使用者集區應用程式用戶端。您的應用程式需要此用戶端，才能與 Amazon Cognito 使用者集區互動。	應用程式開發人員

將您的 Amazon Cognito 使用者集區與 Authenticator UI 元件整合

任務	Description	所需的技能
安裝依存項目。	若要安裝 `aws-amplify` 和 `@aws-amplify/ui-react` 套件，請從應用程式的根目錄執行下列命令： `npm i @aws-amplify/ui-react aws-amplify`	應用程式開發人員

設定使用者集區。	根據下列範例，建立 `aws-exports.js` 檔案並儲存在 `src` 資料夾中。檔案應包含下列資訊：
	• AWS 區域 Amazon Cognito 使用者集區所在的
	• Amazon Cognito 使用者集區 ID
	• 應用程式用戶端 ID
	`// replace the user pool region, id, and app client id details`
	`const awsmobile = {`
		應用程式開發人員

任務	Description	所需的技能

	<pre>"aws_project_region": "put_your_region_here",
	"aws_cognito_region": "put_your_region_here",
	"aws_user_pools_id": "put_your_user_pool_id_here",
	"aws_user_pools_web_client_id": "put_your_user_pool_app_id_here"
	}
	
	export default awsmobile;</pre>	

任務	Description	所需的技能

匯入並設定 Amplify 服務。	1. 在應用程式的進入點檔案中（例如 App.js），輸入以下幾行程式碼匯入和載入aws-exports.js 檔案：
	`import { Amplify } from 'aws-amplify'; import awsExports from './aws-exports';`
	2. 根據下列範例，使用 aws-exports.js 檔案設定 Amplify 用戶端：
	`// Configure Amplify in index file or root file Amplify.configure({ ...awsExports });`
	如需詳細資訊，請參閱 Amplify 文件中的設定 Amplify 類別。
		應用程式開發人員

任務	Description	所需的技能

新增驗證器 UI 元件。	若要顯示 Authenticator UI 元件，請將以下幾行程式碼新增至應用程式的進入點檔案 (App.js)：
	
	`import { Authenticator } from '@aws-amplify/ui-react';`
	`import '@aws-amplify/ui-react/styles.css';`
	
	Note
	程式碼片段範例會匯入 Authenticator UI 元件和 Amplify UI styles.css 檔案，這是使用元件預先建置的主題時的必要項目。
	
	UI Authenticator 元件提供兩個傳回值：
	
	• 使用者詳細資訊
	• 可叫用以將使用者登出的函數
	
	請參閱下列範例元件：
	
	`function App() {`
	`return (`
	`<Authenticator>`
	`{({ signOut, user }) => (`
	`<div>`
		應用程式開發人員

任務	Description	所需的技能

	<pre><p>Welcome {user.username}</p>
	
	<button onClick={signOut}>Sign out</button>
	</div>
	)}
	</Authenticator>
	);
	}</pre>
	Note
	如需範例App.js檔案，請參閱此模式的其他資訊區段。	

（選用）擷取工作階段資訊。	驗證使用者之後，您可以從 Amplify 用戶端擷取有關其工作階段的資料。例如，您可以從使用者的工作階段擷取 JSON Web 字符 (JWT)，以便驗證從其工作階段到後端 API 的請求。
	
	請參閱下列包含 JWT 的請求標頭範例：
	
	<pre>import { fetchAuthSession } from 'aws-amplify/auth';
	(await fetchAuthSession()).tokens?.idToken?.toString();</pre>
		應用程式開發人員

疑難排解

問題	解決方案
新使用者無法註冊應用程式。	如下所示，請確定您的 Amazon Cognito 使用者集區已設定為允許使用者註冊自己在使用者集區中： • 登入 AWS 管理主控台，然後開啟 Amazon Cognito 主控台。 • 在左側導覽窗格中，選擇使用者集區。 • 從清單中選擇您的使用者集區。 • 在一般設定下，選擇政策。 • 選擇允許使用者自行註冊。
從 v5 升級到 v6 後，驗證元件停止運作。	`Auth` 類別已移至 Amplify v6 中的功能方法和具名參數。您現在必須直接從 `aws-amplify/auth` 路徑匯入功能 APIs。如需詳細資訊，請參閱 Amplify 文件中的從 v5 遷移至 v6。

相關資源

- Amazon Cognito 入門 (AWS 網站)
- 建立新的 React 應用程式 (React 文件)
- 什麼是 Amazon Cognito？ (Amazon Cognito 文件)
- Amplify UI 程式庫 (Amplify 文件)

其他資訊

`App.js` 檔案應包含下列程式碼：

```
import './App.css';
import { Amplify } from 'aws-amplify';
import awsExports from './aws-exports';
import { fetchAuthSession } from 'aws-amplify/auth';
import { Authenticator } from '@aws-amplify/ui-react';
```

```
import '@aws-amplify/ui-react/styles.css';
Amplify.configure({ ...awsExports });
let token = (await fetchAuthSession()).tokens?.idToken?.toString();
function App() {
  return (
      <Authenticator>
        {({ signOut, user }) => (
            <div>
              <p>Welcome {user.username}</p>
                <p>Your token is: {token}</p>
              <button onClick={signOut}>Sign out</button>
            </div>
        )}
      </Authenticator>
  );
}

export default App;
```

使用 AWS Amplify 建立 React 應用程式，並使用 Amazon Cognito 新增身分驗證

Rishi Singla，Amazon Web Services

摘要

此模式示範如何使用 AWS Amplify 建立以 React 為基礎的應用程式，以及如何使用 Amazon Cognito 將身分驗證新增至前端。AWS Amplify 包含一組工具（開放原始碼架構、視覺化開發環境、主控台）和服務 (Web 應用程式和靜態網站託管），以加速 AWS 上的行動和 Web 應用程式開發。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 安裝在機器上的 Node.js 和 npm

產品版本

- Node.js 10.x 版或更新版本（若要驗證您的版本，`node -v`請在終端機視窗中執行）
- npm 6.x 版或更新版本（若要驗證您的版本，`npm -v`請在終端機視窗中執行）

Architecture

目標技術堆疊

- AWS Amplify
- Amazon Cognito

工具

- Amplify 命令列界面 (CLI)
- Amplify 程式庫（開放原始碼用戶端程式庫）
- Amplify Studio（視覺化界面）

史詩

安裝 AWS Amplify CLI

任務	Description	所需的技能

安裝 Amplify CLI。	Amplify CLI 是為 React 應用程式建立 AWS 雲端服務的統一工具鏈。若要安裝 Amplify CLI，請執行：
	`npm install -g @aws-amplify/cli`
	npm 會在有新的主要版本可用時通知您。若是如此，請使用下列命令來升級您的 npm 版本：
	`npm install -g npm@9.8.0`
	其中 9.8.0 是指您要安裝的版本。
		應用程式開發人員

建立 React 應用程式

任務	Description	所需的技能
建立 React 應用程式。	若要建立新的 React 應用程式，請使用 命令： `npx create-react-app amplify-react-application`	應用程式開發人員

任務	Description	所需的技能
	其中 `ampify-react-application` 是應用程式的名稱。 成功建立應用程式後，您會看到訊息： `Success! Created amplify-react-application` 系統會為 React 應用程式建立具有各種子資料夾的目錄。	
在本機電腦上啟動應用程式。	移至上一個步驟中建立`amplify-react-application` 的目錄，然後執行命令： `amplify-react-application% npm start` 這會在您的本機電腦上啟動 React 應用程式。	應用程式開發人員

設定 Amplify CLI

任務	Description	所需的技能
設定 Amplify 以連線至您的 AWS 帳戶。	執行 命令來設定 Amplify： `amplify-react-application % amplify configure`	General AWS，應用程式開發人員

任務	Description	所需的技能
	Amplify CLI 會要求您遵循以下步驟來設定對 AWS 帳戶的存取： 1. 登入您的 AWS 管理員帳戶。 2. 指定您要使用的 AWS 區域。 3. 建立具有程式設計存取權的 AWS Identity and Access Management (IAM) 使用者，並將`AdministratorAccess-Amplify` 許可政策連接至使用者。 4. 建立並複製存取金鑰 ID 和私密存取金鑰。 5. 在終端機中輸入這些詳細資訊。 6. 建立設定檔名稱或使用預設設定檔。 ⚠ Warning 此案例需要具有程式設計存取和長期登入資料的 IAM 使用者，這會造成安全風險。為了協助降低此風險，建議您只為這些使用者提供執行任務所需的許可，並在不再需要這些使用者時將其移除。必要時可以更新存取金鑰。如需詳細資訊，請參閱	

任務	Description	所需的技能

	《IAM 使用者指南》中的更新存取金鑰。
	
	這些步驟會顯示在終端機中，如下所示。
	
	`Follow these steps to set up access to your AWS account:`
	`Sign in to your AWS administrator account: https://console.aws.amazon.com/`
	`Press Enter to continue`
	`Specify the AWS Region`
	`? region:  us-east-1`
	`Follow the instructions at https://docs.amplify.aws/cli/start/install/#configure-the-amplify-cli`
	`to complete the user creation in the AWS console`
	`https://console.aws.amazon.com/iamv2/home#/users/create`
	`Press Enter to continue`
	`Enter the access key of the newly created user:`
	`? accessKeyId: ********************`
	`? secretAccessKey: ****************************************`
	`****`	

任務	Description	所需的技能
	`This would update/cr eate the AWS Profile in your local machine ? Profile Name: new  Successfully set up the new user.` 如需這些步驟的詳細資訊，請參閱 Amplify 開發中心中的 文件。	

初始化 Amplify

任務	Description	所需的技能
初始化 Amplify。	1. 若要在新目錄中初始化 Amplify，請執行： `amplify init` Amplify 會提示您輸入專案名稱和組態參數 2. 指定所有參數，然後按 Y 以指定組態初始化專案。 `Project information | Name: amplifyre actproject  | Environment: dev  | Default editor: Visual Studio Code  | App type: javascrip t`	應用程式開發人員，一般 AWS

任務	Description	所需的技能

```
| Javascript
 framework: react

| Source Directory
 Path: src

| Distribution
 Directory Path: build

| Build Command: npm
 run-script build

| Start Command: npm
 run-script start
```

3. 選取您在上一個步驟中建立的設定檔。資源將部署到您建立的 Amplify 專案中的dev環境中。
4. 若要確認資源已建立，您可以開啟 AWS Amplify 主控台，並檢視用來建立資源和詳細資訊的 AWS CloudFormation 範本。

```
Deploying root stack
 amplifyreactproject
 [ ==================
==----------------
---- ] 2/4
amplify-amplif
yreactproject-d…
 AWS::CloudFormatio
n::Stack
 CREATE_IN_PROGRESS


UnauthRole
            AWS::IAM:
```

任務	Description	所需的技能
	:Role CREATE_COMPLETE DeploymentBucket AWS::S3:: Bucket CREATE_IN_PROGRESS AuthRole AWS::IAM: :Role CREATE_COMPLETE	

將身分驗證新增至前端

任務	Description	所需的技能
新增身分驗證。	您可以使用 `amplify add <category>` 命令來新增功能，例如使用者登入或後端 API。在此步驟中，您將使用命令來新增身分驗證。 Amplify 提供後端身分驗證服務，其中包含 Amazon Cognito、前端程式庫和插入式 Authenticator UI 元件。功能包括使用者註冊、使用者登入、多重驗證、使用者登出和無密碼登入。您也可以透過與 Amazon、Google 和 Facebook 等聯合身分提供者整合來驗證使用者。Amplify 身分驗證類別與其他 Amplify 類	應用程式開發人員，一般 AWS

任務	Description	所需的技能

	別無縫整合，例如 API、分析和儲存，因此您可以為已驗證和未驗證的使用者定義授權規則。
	
	1. 若要設定 React 應用程式的身分驗證，請執行 命令：
	
	`amplify-react-application1 % amplify add auth`
	
	這會顯示下列資訊和提示。您可以根據您的業務和安全需求選擇適當的組態。
	
	`Using service: Cognito, provided by: awscloudformation The current configured provider is Amazon Cognito. Do you want to use the default authentication and security configuration? (Use arrow keys)`
	
	`# Default configuration`
	
	`Default configuration with Social Provider (Federation)`
	
	`Manual configuration`	

任務	Description	所需的技能

	<pre> I want to learn
	 more.</pre>
	2. 如需簡單範例，請選擇預設組態，然後選取使用者的登入機制（在此情況下為電子郵件）：
	<pre>How do you want users
	 to be able to sign
	 in?
	 Username
	
	# Email
	
	  Phone Number
	
	  Email or Phone
	 Number
	
	  I want to learn
	 more.</pre>
	3. 略過進階設定以完成新增身分驗證資源：
	<pre>Do you want to
	 configure advanced
	 settings? (Use arrow
	 keys)
	# No, I am done.
	
	  Yes, I want to make
	 some additional
	 changes.</pre>
	4. 建置您的本機後端資源，並在雲端中佈建它們：	

任務	Description	所需的技能

	<pre>amplify-react-appl
	ication1 % amplify
	 push</pre>
	此命令會對您帳戶中的 Congito 使用者集區進行適當的變更。
	5. 按 Y 以使用 CloudForm ation 設定auth資源。
	
	這會設定下列資源：
	
	<pre>UserPool
	           AWS::Cogn
	ito::UserPool
	  CREATE_COMPLETE
	UserPoolClientWeb
	            AWS::Cogn
	ito::UserPoolClient
	   CREATE_COMPLETE
	
	UserPoolClientWeb
	            AWS::Cogn
	ito::UserPoolClient
	   CREATE_COMPLETE
	
	UserPoolClientRole
	            AWS::IAM:
	:Role
	 CREATE_COMPLETE
	
	UserPoolClientLambda
	           AWS::Lamb
	da::Function
	  CREATE_COMPLETE
	UserPoolClientLam
	bdaPolicy
	 AWS::IAM::Policy
	            CREATE_CO</pre>	

任務	Description	所需的技能
	MPLETE UserPoolClientLog Policy AWS::IAM::Policy CREATE_IN _PROGRESS 您也可以使用 AWS Cognito 主控台來檢視這些資源（尋找 Cognito 使用者集區和身分集區）。 此步驟會使用 Cognito 使用者集區和身分集區組態，更新 React 應用程式`src`資料夾中`aws-exports.js` 的檔案。	

變更 App.js 檔案

任務	Description	所需的技能
變更 App.js 檔案。	在 `src`資料夾中，開啟並修訂 `App.js` 檔案。修改後的檔案看起來應該如下所示： { App.Js File after modifications: import React from 'react'; import logo from './ logo.svg'; import './App.css'; import { Amplify } from 'aws-amplify';	應用程式開發人員

任務	Description	所需的技能

	`import { withAuthenticator, Button, Heading } from '@aws-amplify/ui-react';`
	`import awsconfig from './aws-exports';`
	`Amplify.configure(awsconfig);`
	`function App({ signOut }) {`
	`return (`
	`<div>`
	`<h1>Thankyou for doing verification</h1>`
	`<h2>My Content</h2>`
	`<button onClick={signOut}>Sign out</button>`
	`</div>`
	`);`
	`}`
	`export default withAuthenticator(App);`	

匯入 React 套件。	App.js 檔案會匯入兩個 React 套件。使用 命令安裝這些套件： `amplify-react-application1 % npm install --save aws-amplify @aws-amplify/ui-react`	應用程式開發人員

啟動 React 應用程式並檢查身分驗證

任務	Description	所需的技能
啟動應用程式。	在本機電腦上啟動 React 應用程式： `amplify-react-application1 % npm start`	應用程式開發人員，一般 AWS
檢查身分驗證。	檢查應用程式是否提示驗證參數。（在我們的範例中，我們已將電子郵件設定為登入方法。） 前端 UI 應該會提示您輸入登入憑證，並提供建立 帳戶的選項。 您也可以設定 Amplify 建置程序，將後端新增為連續部署工作流程的一部分。不過，此模式不會涵蓋該選項。	應用程式開發人員，一般 AWS

相關資源

- 入門 (npm 文件）
- 建立獨立的 AWS 帳戶 (AWS 帳戶管理文件）
- AWS Amplify 文件
- Amazon Cognito 文件

使用、 AWS Amplify Angular 和 Module Federation 建立微型前端的入口網站

Milena Godau 和 Pedro Garcia，Amazon Web Services

總結

微型前端架構可讓多個團隊獨立處理前端應用程式的不同部分。每個團隊都可以開發、建置和部署前端的片段，而不會干擾應用程式的其他部分。從最終使用者的角度來看，它似乎是一個單一且具凝聚力的應用程式。不過，他們正在與數個由不同團隊發佈的獨立應用程式互動。

本文件說明如何使用 AWS Amplify、角度前端架構和模組聯合來建立微型前端架構。在此模式中，微型前端在用戶端由 shell（或父系）應用程式組合。shell 應用程式充當擷取、顯示和整合微型前端的容器。shell 應用程式會處理全域路由，這會載入不同的微型前端。@angular-architects/module-federation 外掛程式整合 Module Federation 與 Angular。您可以使用 部署 shell 應用程式和微型前端 AWS Amplify。最終使用者透過 Web 入口網站存取應用程式。

入口網站會垂直分割。這表示微型前端是整個檢視或檢視群組，而不是相同檢視的一部分。因此，殼層應用程式一次只會載入一個微型前端。

微型前端會實作為遠端模組。shell 應用程式會延遲載入這些遠端模組，將微前端初始化延遲到需要為止。此方法透過僅載入必要的模組來最佳化應用程式效能。這可減少初始載入時間，並改善整體使用者體驗。此外，您可以透過 webpack 組態檔案 (webpack.config.js) 跨模組共用常見的相依性。此實務可提升程式碼重複使用、減少重複，並簡化綁定程序。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- Node.js 和 npm，已安裝
- Amplify CLI，已安裝
- 角 CLI，已安裝
- 使用 的許可 AWS Amplify
- 熟悉角度

產品版本

- 角度 CLI 13.1.2 版或更新版本
- @angular-architects/module-federation 14.0.1 版或更新版本
- Webpack 5.4.0 版或更新版本
- AWS Amplify 第 1 代

限制

微型前端架構是一種強大的方法來建置可擴展性和彈性的 Web 應用程式。不過，在採用此方法之前，請務必了解下列潛在挑戰：

- 整合 – 其中一個主要挑戰是相較於單體前端，潛在的複雜性增加。協調多個微型前端、處理它們之間的通訊和管理共用相依性可能更為複雜。此外，可能會有與微型前端之間的通訊相關聯的效能額外負荷。此通訊可以增加延遲並降低效能。這需要透過有效的傳訊機制和資料共用策略來解決。
- 程式碼重複 – 由於每個微型前端都是獨立開發的，因此存在複製常見功能或共用程式庫程式碼的風險。這可能會增加整體應用程式大小，並帶來維護挑戰。
- 協調和管理 – 協調跨多個微型前端的開發和部署程序可能具有挑戰性。確保一致的版本控制、管理相依性，以及維護元件之間的相容性在分散式架構中變得更加重要。建立明確的控管、指導方針和自動化測試和部署管道，對於無縫協作和交付至關重要。
- 測試 – 測試微型前端架構可能比測試單體前端更複雜。它需要額外的工作量和專門的測試策略，以執行跨元件整合測試和end-to-end測試，並驗證多個微型前端的一致使用者體驗。

在承諾微前端方法之前，我們建議您檢閱了解和實作微前端 AWS。

Architecture

在微型前端架構中，每個團隊都會獨立開發和部署功能。下圖顯示多個 DevOps 團隊如何一起運作。入口網站團隊開發 shell 應用程式。shell 應用程式充當容器。它會擷取、顯示和整合由其他 DevOps 團隊發佈的微型前端應用程式。您可以使用 AWS Amplify 來發佈 shell 應用程式和微型前端應用程式。

架構圖顯示下列工作流程：

1. 入口網站團隊會開發和維護 shell 應用程式。shell 應用程式會協調微型前端的整合和轉譯，以構成整體入口網站。
2. 團隊 A 和 B 開發和維護整合到入口網站的一或多個微型前端或功能。每個團隊可以在各自的微型前端獨立工作。

3. 最終使用者使用 Amazon Cognito 進行身分驗證。
4. 最終使用者存取入口網站，並載入 shell 應用程式。當使用者導覽時，殼層應用程式會處理路由，並擷取請求的微型前端，並載入其套件。

工具

AWS 服務

- AWS Amplify 是一組專用工具和功能，可協助前端 Web 和行動開發人員快速建置完整堆疊的應用程式 AWS。在此模式中，您可以使用 Amplify CLI 部署 Amplify 微型前端應用程式。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。

其他工具

- @angular-architects/module-federation 是整合 Angular 與 Module Federation 的外掛程式。
- Angular 是一種開放原始碼 Web 應用程式架構，用於建置現代、可擴展且可測試的單一頁面應用程式。它遵循可促進程式碼重複使用和維護的模組化和元件型架構。
- Node.js 是一種事件驅動的 JavaScript 執行期環境，專為建置可擴展的網路應用程式而設計。
- npm 是在 Node.js 環境中執行的軟體登錄檔，用於共用或借用套件和管理私有套件的部署。
- Webpack Module Federation 可協助您將獨立編譯和部署的程式碼載入應用程式，例如微型前端或外掛程式。

程式碼儲存庫

此模式的程式碼可在 Micro-frontend 入口網站中使用 Angular 和 Module Federation GitHub 儲存庫。此儲存庫包含下列兩個資料夾：

- `shell-app` 包含 shell 應用程式的程式碼。
- `feature1-app` 包含範例微型前端。shell 應用程式會擷取此微型前端，並將其顯示為入口網站應用程式中的頁面。

最佳實務

微型前端架構提供許多優勢，但也會帶來複雜性。以下是順暢開發、高品質程式碼和良好使用者體驗的一些最佳實務：

- 規劃和溝通 – 為了簡化協同合作，請投資前期規劃、設計和明確的溝通管道。
- 設計一致性 – 使用設計系統、樣式指南和元件程式庫，跨微型前端強制執行一致的視覺化樣式。這可提供有凝聚力的使用者體驗並加速開發。
- 相依性管理 – 由於微型前端獨立發展，請採用標準化合約和版本控制策略，以有效管理相依性並防止相容性問題。
- 微型前端架構 – 若要啟用獨立開發和部署，每個微型前端都應對封裝功能負有明確且明確的責任。
- 整合和通訊 – 為了促進順暢整合並將衝突降至最低，請定義微型前端之間的明確合約和通訊協定，包括 APIs、事件和共用資料模型。
- 測試和品質保證 – 實作微型前端的測試自動化和持續整合管道。這可改善整體品質、減少手動測試工作，並驗證微型前端互動之間的功能。
- 效能最佳化 – 持續監控效能指標並追蹤微型前端之間的相依性。這可協助您識別瓶頸並維持最佳應用程式效能。為此使用效能監控和相依性分析工具。
- 開發人員體驗 – 透過提供清晰的文件、工具和範例，專注於開發人員體驗。這可協助您簡化開發並加入新的團隊成員。

史詩

建立 shell 應用程式

任務	Description	所需的技能

建立 shell 應用程式。	1. 在角度 CLI 中，輸入下列命令：
	`ng new shell --routing`
	2. 輸入下列命令以導覽至專案資料夾：
	`cd shell`
	Note
	shell 和微型前端應用程式的資料夾和
		應用程式開發人員

任務	Description	所需的技能
	專案結構可以完全獨立。它們可以做為獨立的 Angular 應用程式處理。	
安裝 外掛程式。	在 Angular CLI 中，輸入下列命令來安裝 @angular-architects/module-federation 外掛程式： `ng add @angular-architects/module-federation --project shell --port 4200`	應用程式開發人員
新增微型前端 URL 做為環境變數。	1. 開啟 environment.ts 檔案。 2. 將 `mfe1URL: 'http://localhost:5000'` 新增至`environment` 物件： `export const environment = { production: false, mfe1URL: 'http://localhost:5000', };` 3. 儲存並關閉 environment.ts 檔案。	應用程式開發人員

任務	Description	所需的技能

定義路由。	1. 開啟 app-routing.module.ts 檔案。
	2. 在角度 CLI 中，輸入下列命令，從 @angular-architects/module-federation 外掛程式匯入`loadRemoteModule` 模組：
	`import { loadRemoteModule } from '@angular-architects/module-federation';`
	3. 將預設路由設定為下列項目：
	`{ path: '', pathMatch: 'full', redirectTo: 'mfe1' },`
	4. 設定微前端的路由：
	`{ path: 'mfe1', loadChildren: () => loadRemoteModule({ type: 'module', remoteEntry: `${environment.mfe1URL}/remoteEntry.js`,`
		應用程式開發人員

任務	Description	所需的技能

	<pre>exposedMo
	dule: './Module'
	})
	.then(m =>
	m.Mfe1Module)
	},</pre>
	5. 儲存並關閉 app-routing.module.ts 檔案。	

宣告mfe1模組。	1. 在 src資料夾中，建立一個名為 decl.d.ts 的新檔案。 2. 開啟 decl.d.ts 檔案。 3. 將下列項目新增至 檔案： <pre>declare module 'mfe1/Module';</pre> 4. 儲存並關閉 decl.d.ts 檔案。	應用程式開發人員

任務	Description	所需的技能

準備微前端的預先載入。	預先載入微型前端有助於 Webpack 正確交涉共用程式庫和套件。
	1. 開啟main.ts 檔案。
	2. 將內容取代為下列項目：
	`import { loadRemoteEntry } from '@angular-architects/module-federation';`
	`Promise.all([ loadRemoteEntry(`${environment.mfe1URL}/remoteEntry.js`, 'mfe1'), ]) .catch(err => console.error('Error loading remote entries', err)) .then(() => import('./bootstrap')) .catch(err => console.error(err) );`
	3. 儲存並關閉 main.ts 檔案。
		應用程式開發人員

任務	Description	所需的技能
調整 HTML 內容。	1. 開啟 app.component.html 檔案。 2. 將內容取代為下列項目： `<h1>Shell application is running!</h1><router-outlet></router-outlet>` 3. 儲存並關閉 app.component.html 檔案。	應用程式開發人員

建立微型前端應用程式

任務	Description	所需的技能

建立微型前端。	1. 在角度 CLI 中，輸入下列命令：
	`ng new mfe1 --routing`
	2. 輸入下列命令以導覽至專案資料夾：
	`cd mfe1`
		應用程式開發人員

安裝 外掛程式。	輸入下列命令來安裝 @angular-architects/module-federation 外掛程式： `ng add @angular-architects/module-federation --project mfe1 --port 5000`	應用程式開發人員

任務	Description	所需的技能

建立模組和元件。	輸入下列命令來建立模組和元件，並將其匯出為遠端項目模組：
	`ng g module mfe1 --routing`
	`ng g c mfe1`
		應用程式開發人員

設定預設路由路徑。	1. 開啟 mfe-routing.module.ts 檔案。 2. 將預設路由設定為下列項目： `{ path: '', component: Mfe1Component },` 3. 儲存並關閉 mfe-routing.module.ts 檔案。	應用程式開發人員

任務	Description	所需的技能

新增mfe1路由。	1. 開啟app-routing.module.ts 檔案。
	2. 將預設路由設定為下列項目：
	`{ path: '', pathMatch: 'full', redirectTo: 'mfe1' },`
	3. 新增下列mfe1路由：
	`{ path: 'mfe1', loadChildren: () => import('./mfe1/mfe1.module').then((m) => m.Mfe1Module), },`
	4. 儲存並關閉 app-routing.module.ts 檔案。
		應用程式開發人員

任務	Description	所需的技能
編輯 webpack.config.js 檔案。	1. 開啟 webpack.config.js 檔案。 2. 編輯 For remotes區段以符合下列項目： 3. 在 shared區段中，新增mfe1應用程式與 shell 應用程式共用的任何相依性：	應用程式開發人員

```
// For remotes (please
 adjust)
    name: "mfe1",
    filename:
 "remoteEntry.js",
    exposes: {
        './Module':
 './src/app/mfe1/mf
e1.module.ts',
    },
```

```
shared: share({
        "@angular/
core": { singleton
: true, strictVer
sion: true, requiredV
ersion: 'auto' },
        "@angular/
common": { singleton
: true, strictVer
sion: true, requiredV
ersion: 'auto' },
        "@angular
/common/http":
 { singleton: true,
 strictVersion: true,
 requiredVersion:
 'auto' },
        "@angular/
router": { singleton
: true, strictVer
```

任務	Description	所需的技能
	sion: true, requiredV ersion: 'auto' }, ...shared Mappings.getDescri ptors() })	
	4. 儲存並關閉 webpack.c onfig.js 檔案。	
調整 HTML 內容。	1. 開啟 app.component.html 檔案。 2. 將內容取代為下列項目： <router-outlet></r outer-outlet> 3. 儲存並關閉 app.compo nent.html 檔案。	應用程式開發人員

在本機執行應用程式

任務	Description	所需的技能
執行mfe1應用程式。	1. 輸入下列命令以啟動mfe1應 用程式： npm start 2. 在 Web 瀏覽器中，存取 http://localhost:5 000 。 3. 確認微型前端可以獨立執 行。mfe1 應用程式應該正 確轉譯，沒有任何錯誤。	應用程式開發人員

任務	Description	所需的技能
執行 shell 應用程式。	1. 輸入下列命令以啟動 shell 應用程式： `npm start` 2. 在 Web 瀏覽器中，存取 `http://localhost:4200/mfe1` 。 3. 確認mfe1微型前端內嵌在 shell 應用程式中。入口網站應用程式應該正確轉譯，沒有任何錯誤，而且mfe1應用程式應該內嵌在其中。	應用程式開發人員

重構 shell 應用程式以處理微前端載入錯誤

任務	Description	所需的技能

建立模組和元件。	在 shell 應用程式的根資料夾中，輸入下列命令來建立錯誤頁面的模組和元件：
	`ng g module error-page --routing`
	`ng g c error-page`
		應用程式開發人員

調整 HTML 內容。	1. 開啟 error-page.component.html 檔案。 2. 將內容取代為下列項目： `<p>Sorry, this page is not available.</p>` 3. 儲存並關閉 error-page.component.html 檔案。	應用程式開發人員

任務	Description	所需的技能
設定預設路由路徑。	1. 開啟 error-page-routing .module.ts 檔案。 2. 將預設路由設定為下列項目： `{ path: '', component: ErrorPageComponent },` 3. 儲存並關閉 error-page-routing.module.ts 檔案。	應用程式開發人員

任務	Description	所需的技能

建立 函數以載入微型前端。	1. 開啟 app-routing.module.ts 檔案。
	2. 建立下列函數：
	`function loadMFE(url: string) { return loadRemoteModule({ type: 'module', remoteEntry: `${url}/remoteEntry.js`, exposedModule: './Module' }) .then(m => m.Mfe1Module) .catch( () => import('./error-page/error-page.module').then(m => m.ErrorPageModule) ); }`
	3. 將mfe1路由修改為下列項目：
	`{ path: 'mfe1', loadChildren: () => loadMFE(environment.mfe1URL) },`
	4. 儲存並關閉 app-routing.module.ts 檔案。
		應用程式開發人員

任務	Description	所需的技能

測試錯誤處理。	1. 如果尚未執行，請輸入下列命令來啟動 shell 應用程式：
	`npm start`
	2. 在 Web 瀏覽器中，存取 `http://localhost:4200/mfe1` 。
	3. 確認已轉譯錯誤頁面。您應該會看到下列文字：
	`Sorry, this page is not available.`
		應用程式開發人員

使用 部署應用程式 AWS Amplify

任務	Description	所需的技能
部署微型前端。	1. 在 Amplify CLI 中，導覽至微型前端應用程式的根資料夾。 2. 輸入下列命令來初始化 Amplify： `amplify init` 3. 當系統提示您輸入 Amplify 專案的名稱時，請按 Enter。這會重複使用 package.json 檔案中的名稱。	應用程式開發人員、AWS DevOps

任務	Description	所需的技能

	4. 當系統提示您使用上述組態初始化專案時，請輸入 `Yes`。
	5. 當您收到選取身分驗證方法的提示時，請選擇 `AWS Profile`。
	6. 選取您要使用的設定檔。
	7. 等待 Amplify 初始化專案。當此程序完成時，您會在終端機中收到確認訊息。
	8. 輸入下列命令，將 Amplify 託管類別新增至微型前端：
	`amplify add hosting`
	9. 當您收到選取外掛程式模組的提示時，請選擇 `Hosting with Amplify Console`。
	10. 當系統提示您選擇類型時，請選擇 `Manual deployment` 。
	11. 輸入下列命令來安裝專案 npm 相依性：
	`npm install`
	12. 輸入下列命令，將應用程式發佈至 Amplify 主控台：
	`amplify publish -y`
	發佈完成後，Amplify 會傳回微型前端的 URL。	

任務	Description	所需的技能
	13.複製 URL。您需要此值才能更新 shell 應用程式。	

任務	Description	所需的技能

部署 shell 應用程式。	1. 在 src/app/environments 資料夾中，開啟 environment.prod.ts 檔案。
	2. 將`mfe1URL`值取代為已部署微型前端的 URL：
	`export const environment = { production: true, mfe1URL: 'https://<env>.<Amplify-app-ID>.amplifyapp.com' };`
	3. 儲存並關閉 environment.prod.ts 檔案。
	4. 在 Amplify CLI 中，導覽至 shell 應用程式的根資料夾。
	5. 輸入下列命令來初始化 Amplify：
	`amplify init`
	6. 當系統提示您輸入 Amplify 專案的名稱時，請按 Enter。這會重複使用 package.json 檔案中的名稱。
	7. 當系統提示您使用上述組態初始化專案時，請輸入 `Yes`。
	8. 當您收到選取身分驗證方法的提示時，請選擇 `AWS Profile`。
	9. 選取您要使用的設定檔。
		應用程式開發人員、應用程式擁有者

任務	Description	所需的技能

	10.等待 Amplify 初始化專案。當此程序完成時，您會在終端機中收到確認訊息。
	11.將 Amplify 託管類別新增至 shell 應用程式：
	`amplify add hosting`
	12.當您收到選取外掛程式模組的提示時，請選擇 `Hosting with Amplify Console`。
	13.當系統提示您選擇類型時，請選擇 `Manual deployment` 。
	14.輸入下列命令來安裝專案 npm 相依性：
	`npm install`
	15.輸入下列命令，將 shell 應用程式發佈至 Amplify 主控台：
	`amplify publish -y`
	發佈完成時，Amplify 會傳回已部署 shell 應用程式的 URL。
	16.請記下 shell 應用程式的 URL。	

任務	Description	所需的技能

啟用 CORS。	由於 shell 和微型前端應用程式是獨立託管在不同網域上，因此您必須在微型前端上啟用跨來源資源共用 (CORS)。這可讓 shell 應用程式從不同的原始伺服器載入內容。若要啟用 CORS，您可以新增自訂標頭。
	
	1. 在 Amplify CLI 中，導覽至微型前端的根資料夾。
	2. 輸入以下命令：
	
	`amplify configure hosting`
	
	3. 當系統提示您設定自訂設定時，請輸入 Y。
	4. 登入 AWS 管理主控台，然後開啟 Amplify 主控台。
	5. 選擇微型前端。
	6. 在導覽窗格中，選擇託管，然後選擇自訂標頭。
	7. 選擇編輯。
	8. 在編輯自訂標頭視窗中，輸入下列內容：
	
	`customHeaders:`
	`  - pattern: '*.js'`
	`    headers:`
	`      - key: Access-Control-Allow-Origin`
	`        value: '*'`
	`      - key: Access-Control-Allow-Methods`
		應用程式開發人員、AWS DevOps

任務	Description	所需的技能

	<pre>value: 'GET, OPTIONS'
	- key: Access-Control-Allow-Headers
	value: '*'</pre>
	9. 選擇儲存。
	10重新部署微型前端以套用新的自訂標頭。	

任務	Description	所需的技能

在 shell 應用程式中建立重寫規則。	角度 shell 應用程式設定為使用 HTML5 路由。如果使用者執行硬重新整理，Amplify 會嘗試從目前的 URL 載入頁面。這會產生 403 錯誤。若要避免這種情況，您可以在 Amplify 主控台中新增重寫規則。
	
	若要建立重寫規則，請遵循下列步驟：
	
	1. 在 Amplify CLI 中，導覽至 shell 應用程式的根資料夾。
	2. 輸入以下命令：
	
	`amplify configure hosting`
	
	3. 當系統提示您設定自訂設定時，請輸入 Y。
	4. 開啟 Amplify 主控台。
	5. 選擇 shell 應用程式。
	6. 在導覽窗格中，選擇託管，然後選擇重寫和重新導向。
	7. 在重寫和重新導向頁面上，選擇管理重新導向。
	8. 選擇開啟文字編輯器。
	9. 在 JSON 編輯器中，輸入下列重新導向：
	
	`[ { "source": "/<*>",`
		應用程式開發人員、AWS DevOps

任務	Description	所需的技能

	<pre>"target": "/index.html",
	"status": "404-200",
	"condition": null
	}
	]</pre>
	10選擇儲存。	

測試 Web 入口網站。	1. 在 Web 瀏覽器中，輸入已部署 shell 應用程式的 URL。 2. 驗證 shell 應用程式和微型前端負載是否正確。	應用程式開發人員

清除資源

任務	Description	所需的技能
刪除應用程式。	如果您不再需要 shell 和微型前端應用程式，請將其刪除。這有助於避免您未使用之資源的費用。 1. 登入 AWS 管理主控台，然後開啟 Amplify 主控台。 2. 選擇微型前端。 3. 在導覽窗格中，選擇應用程式設定，然後選擇一般設定。 4. 選擇刪除應用程式。 5. 在確認視窗中，輸入 `delete`，然後選擇刪除應用程式。	一般 AWS

任務	Description	所需的技能
	6. 重複這些步驟來刪除 shell 應用程式。	

疑難排解

問題	解決方案
執行 `amplify init`命令時沒有可用的 AWS 設定檔	如果您沒有設定 AWS 設定檔，您仍然可以繼續執行 `amplify init`命令。不過，當系統提示您輸入身分驗證方法時，您需要選取 `AWS access keys`選項。準備好您的 AWS 存取金鑰和私密金鑰。 或者，您可以為 設定具名設定檔 AWS CLI。如需說明，請參閱 AWS CLI 文件中的組態和登入資料檔案設定。
載入遠端項目時發生錯誤	如果您在 Shell 應用程式的 main.ts 檔案中載入遠端項目時發生錯誤，請確定已正確設定`environment.mfe1URL` 變數。此變數的值應該是微型前端的 URL。
存取微型前端時發生錯誤 404	如果您在嘗試存取本機微型前端時收到 404 錯誤，例如在 `http://localhost:4200/mfe1`，請檢查下列項目： • 對於 shell 應用程式，請確定 app-routing.module.ts 檔案中的路由組態設定正確，並確認`loadRemoteModule` 函數已正確呼叫微型前端。 • 對於微型前端，請確認 webpack.config.js 檔案具有正確的`exposes`組態，並確認正確產生 remoteEntry.js 檔案。

其他資訊

AWS 文件

- 了解和實作 上的微型前端 AWS(AWS 方案指引)
- Amplify CLI (Amplify 文件)
- Amplify 託管 (Amplify 文件)

其他參考

- 模組聯合
- Node.js
- 角度
- @angular-architects/module-federation

將以 React 為基礎的單一頁面應用程式部署至 Amazon S3 和 CloudFront

Jean-Baptiste Guillois，Amazon Web Services

摘要

單頁應用程式 (SPA) 是使用 JavaScript APIs 動態更新所顯示網頁內容的網站或 Web 應用程式。這種方法可增強網站的使用者體驗和效能，因為它只會更新新資料，而不是從伺服器重新載入整個網頁。

此模式提供step-by-step方法來編碼和託管以 Amazon Simple Storage Service (Amazon S3) 和 Amazon CloudFront 上的 React 撰寫的 SPA。此模式中的 SPA 使用在 Amazon API Gateway 中設定並透過 Amazon CloudFront 分佈公開的 REST API，以簡化跨來源資源共用 (CORS) 管理。 Amazon CloudFront

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- Node.js 和 npm，已安裝並設定。如需詳細資訊，請參閱 Node.js 文件的下載一節。
- Yarn、已安裝和設定。如需詳細資訊，請參閱 Yarn 文件。
- Git，已安裝並設定。如需詳細資訊，請參閱 Git 文件。

Architecture

此架構會使用 AWS CloudFormation (infrastructure as code) 自動部署。它使用區域服務，例如 Amazon S3 來存放靜態資產，並使用 Amazon CloudFront Amazon API Gateway 來公開區域 API (REST) 端點。應用程式日誌是透過使用 Amazon CloudWatch 收集。所有 AWS API 呼叫都會接受稽核 AWS CloudTrail。所有安全組態（例如身分和許可）都會在 AWS Identity and Access Management (IAM) 中管理。靜態內容會透過 Amazon CloudFront 內容交付網路 (CDN) 交付，而 DNS 查詢則由 Amazon Route 53 處理。

工具

AWS 服務

- Amazon API Gateway 可協助您建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和區域的生命週期進行管理。
- Amazon CloudFront 透過全球資料中心網路提供 Web 內容，進而降低延遲並改善效能，進而加快 Web 內容的發佈速度。
- AWS CloudTrail 可協助您稽核 的控管、合規和營運風險 AWS 帳戶。
- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您執行的應用程式。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon Route 53 是一種可用性高、可擴展性強的 DNS Web 服務。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

Code

此模式的範例應用程式程式碼可在 GitHub React 型 CORS 單頁應用程式儲存庫中使用。

最佳實務

透過使用 Amazon S3 物件儲存，您可以使用安全、高彈性、高效能且符合成本效益的方式存放應用程式的靜態資產。此任務不需要使用專用容器或 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。

透過使用 Amazon CloudFront 內容交付網路，您可以減少使用者存取應用程式時可能遇到的延遲。您也可以連接 Web 應用程式防火牆 (AWS WAF) 來保護資產免受惡意攻擊。

史詩

在本機建置和部署您的應用程式

任務	Description	所需的技能
複製儲存庫。	執行下列命令來複製範例應用程式的儲存庫： `git clone https://github.com/aws-samples/react-cors-spa`	應用程式開發人員、AWS DevOps

任務	Description	所需的技能
	`react-cors-spa && cd react-cors-spa`	
在本機部署應用程式。	1. 在專案目錄中，執行 `npm install`命令來啟動應用程式相依性。 2. 執行 `yarn dev`命令以在本機啟動應用程式。	應用程式開發人員、AWS DevOps
在本機存取應用程式。	開啟瀏覽器視窗並輸入 `http://localhost:3000` URL 以存取應用程式。	應用程式開發人員、AWS DevOps

部署應用程式

任務	Description	所需的技能
部署 AWS CloudFormation 範本。	1. 登入 AWS 管理主控台，然後開啟 AWS CloudFormation 主控台。 2. 選擇建立堆疊，然後選擇使用新資源 （標準）。 3. 選擇 Upload a template file (上傳範本檔案)。 4. 選擇選擇檔案， 從複製的儲存庫中選擇`react-cors-spa-stack.yaml` 檔案，然後選擇下一步。 5. 輸入堆疊的名稱，然後選擇下一步。 6. 保留所有預設選項，然後選擇下一步。	應用程式開發人員、AWS DevOps

任務	Description	所需的技能
	7. 檢閱堆疊的最終設定，然後選擇建立堆疊。	
自訂您的應用程式來源檔案。	1. 部署堆疊後，開啟輸出索引標籤並識別Bucket名稱和APIDomain 值。 2. 複製 REST API 的 CloudFront 分佈網域。 3. 導覽至 <project_root>/src/pages/index.tsx ，然後將此網域插入或貼到index.tsx 檔案第 13 行的APIEndPoint 變數值。	應用程式開發人員
建置應用程式套件。	在您的專案目錄中，執行 yarn build命令來建置應用程式套件。	應用程式開發人員

任務	Description	所需的技能
部署應用程式套件。	1. 開啟 Amazon S3 主控台。 2. 識別並選擇 CloudFormation 堆疊先前建立的 S3 儲存貯體。 3. 選擇上傳，然後選擇新增檔案。 4. 選擇out資料夾的內容。 5. 選擇新增資料夾，然後選擇_next目錄。 **Important** 選擇_next目錄，而非內容。 6. 選擇上傳，將檔案和目錄上傳至 S3 儲存貯體。	應用程式開發人員、AWS DevOps

測試應用程式。

任務	Description	所需的技能
存取和測試應用程式。	開啟瀏覽器視窗，然後貼上 CloudFront 分佈網域（您先前部署的 CloudFormation 堆疊SPADomain 輸出）以存取應用程式。	應用程式開發人員、AWS DevOps

清除資源

任務	Description	所需的技能
刪除 S3 儲存貯體內容。	1. 開啟 Amazon S3 主控台，然後選擇堆疊先前建立的儲存貯體（名稱開頭為 的第一個儲存貯體`react-cors-spa-`）。 2. 選擇空白以刪除儲存貯體的內容。 3. 選擇堆疊先前建立的第二個儲存貯體（名稱開頭為 `react-cors-spa-` 並以 結尾的第二個儲存貯體`-logs`）。 4. 選擇空白以刪除儲存貯體的內容。	AWS DevOps，應用程式開發人員
刪除 CloudFormation 堆疊。	1. 開啟 CloudFormation 主控台，然後選擇您先前建立的堆疊。 2. 選擇刪除以刪除堆疊和所有相關資源。	AWS DevOps，應用程式開發人員

相關資源

若要部署和託管您的 Web 應用程式，您也可以使用 AWS Amplify Hosting，它提供 Git 型工作流程來託管具有持續部署的完整堆疊、無伺服器 Web 應用程式。Amplify Hosting 是 的一部分AWS Amplify，提供一組專門建置的工具和功能，讓前端 Web 和行動開發人員能夠快速輕鬆地在其中建置完整的堆疊應用程式 AWS。

其他資訊

若要處理可能產生 403 錯誤的使用者請求的無效 URLs，CloudFront 分佈中設定的自訂錯誤頁面會擷取 403 個錯誤，並將其重新導向至應用程式進入點 (`index.html`)。

為了簡化 CORS 的管理，REST API 會透過 CloudFront 分佈公開。

使用私有端點和 Application Load Balancer 在內部網站上部署 Amazon API Gateway API

Saurabh Kothari，Amazon Web Services

總結

此模式說明如何在可從內部部署網路存取的內部網站上部署 Amazon API Gateway API。您學習使用以私有端點、Application Load Balancer、AWS PrivateLink 和 Amazon Route 53 設計的架構，為私有 API 建立自訂網域名稱。此架構可防止使用自訂網域名稱和代理伺服器在 API 上協助以網域為基礎的路由的意外後果。例如，如果您在不可路由的子網路中部署虛擬私有雲端 (VPC) 端點，您的網路就無法連線到 API Gateway。常見的解決方案是使用自訂網域名稱，然後在可路由子網路中部署 API，但這可能會在代理組態將流量 (`execute-api.{region}.vpce.amazonaws.com`) 傳遞至 AWS Direct Connect 時破壞其他內部網站。最後，此模式可協助您滿足使用無法從網際網路和自訂網域名稱連線的私有 API 的組織需求。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 網站和 API 的伺服器名稱指示 (SNI) 憑證
- 使用 AWS Direct Connect 或 AWS Site-to-Site VPN，從內部部署環境連線到已設定的 AWS 帳戶
- 具有對應網域（例如 domain.com) 的私有託管區域，已從內部部署網路解析，並將 DNS 查詢轉送至 Route 53
- 可從內部部署網路連線的可路由私有子網路

限制

如需負載平衡器、規則和其他資源配額（先前稱為限制）的詳細資訊，請參閱 Elastic Load Balancing 文件中的 Application Load Balancer 配額。

Architecture

技術堆疊

- Amazon API Gateway
- Amazon Route 53

- Application Load Balancer
- AWS Certificate Manager
- AWS PrivateLink

目標架構

下圖顯示 Application Load Balancer 如何部署在根據 Application Load Balancer 接聽程式規則將 Web 流量導向網站目標群組或 API Gateway 目標群組的 VPC 中。API Gateway 目標群組是 API Gateway 中 VPC 端點的 IP 地址清單。API Gateway 已設定為使用其資源政策讓 API 私有。此政策會拒絕來自特定 VPC 端點的所有呼叫。API 閘道中的自訂網域名稱會更新為針對 API 及其階段使用 api.domain.com。Application Load Balancer 規則會新增，以根據主機名稱路由流量。

該圖顯示以下工作流程：

1. 內部部署網路的使用者嘗試存取內部網站。請求會傳送至 ui.domain.com 和 api.domain.com。然後，請求會解析為可路由私有子網路的內部 Application Load Balancer。SSL 會在 ui.domain.com 和 api.domain.com 的 Application Load Balancer 中終止。
2. 在 Application Load Balancer 上設定的接聽程式規則會檢查主機標頭。

   a. 如果主機標頭是 api.domain.com，請求會轉送到 API Gateway 目標群組。Application Load Balancer 透過連接埠 443 啟動 API Gateway 的新連線。

   b. 如果主機標頭是 ui.domain.com，請求會轉送到網站目標群組。
3. 當請求到達 API Gateway 時，在 API Gateway 中設定的自訂網域映射會決定主機名稱和要執行的 API。

自動化和擴展

此模式中的步驟可以使用 AWS CloudFormation 或 AWS Cloud Development Kit (AWS CDK) 自動化。若要設定 API Gateway 呼叫的目標群組，您必須使用自訂資源來擷取 VPC 端點的 IP 地址。describe-vpc-endpoints 和 describe-network-interfaces 的 API 呼叫會傳回 IP 地址和安全群組，可用於建立 IP 地址的 API 目標群組。

工具

- Amazon API Gateway 可協助您建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。

- Amazon Route 53 是一種可用性高、可擴展性強的 DNS Web 服務。
- AWS Certificate Manager (ACM) 可協助您建立、存放和續約公有和私有 SSL/TLS X.509 憑證和金鑰，以保護 AWS 網站和應用程式。
- AWS 雲端開發套件 (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。
- AWS PrivateLink 可協助您建立從 VPCs 到 VPC 外部服務的單向私有連線。

史詩

建立 SNI 憑證

任務	Description	所需的技能
建立 SNI 憑證並將憑證匯入 ACM。	1. 為 ui.domain.com 和 api.domain.com 建立 SNI 憑證。如需詳細資訊，請參閱 Amazon CloudFront 文件中的選擇 CloudFront 如何提供 HTTPS 請求。 Amazon CloudFront 2. 將 SNI 憑證匯入 AWS Certificate Manager (ACM)。如需詳細資訊，請參閱 ACM 文件中的將憑證匯入 AWS Certificate Manager。	網路管理員

在不可路由的私有子網路中部署 VPC 端點

任務	Description	所需的技能
在 API Gateway 中建立介面 VPC 端點。	若要建立介面 VPC 端點，請遵循 Amazon Virtual Private Cloud (Amazon VPC) 文件	雲端管理員

任務	Description	所需的技能
	中的使用介面 VPC 端點存取 AWS 服務的指示。	

設定 Application Load Balancer

任務	Description	所需的技能
為您的應用程式建立目標群組。	為應用程式的 UI 資源建立目標群組。	雲端管理員
建立 API Gateway 端點的目標群組。	1. 使用 IP 地址類型建立目標群組，然後將 API Gateway 端點的 VPC 端點 IP 地址新增至目標群組。 2. 使用成功代碼 403 為您的目標群組設定運作狀態檢查。403 是必要的，因為 API Gateway 的 VPC 端點會在被目標群組運作狀態檢查調用時傳回 403 代碼，而沒有任何標頭。	雲端管理員
建立 Application Load Balancer。	1. 在可路由的私有子網路中建立 Application Load Balancer（內部）。 2. 將 443 接聽程式新增至 Application Load Balancer，然後從 ACM 選擇憑證。	雲端管理員
建立接聽程式規則。	建立接聽程式規則以執行下列動作： 1. 將主機 api.domain.com 轉送至 API Gateway 目標群組	雲端管理員

任務	Description	所需的技能
	2. 將主機 ui.domain.com 轉送至 UI 資源的目標群組	

設定 Route 53

任務	Description	所需的技能
建立私有託管區域。	為 建立私有託管區域。domain.com	雲端管理員
建立網域記錄。	建立下列項目的 CNAME 記錄： • 值設為 Application Load Balancer DNS 名稱的 API • 值設為 Application Load Balancer DNS 名稱的 UI	雲端管理員

在 API Gateway 中建立私有 API 端點

任務	Description	所需的技能
建立和設定私有 API 端點。	1. 若要建立私有 API 端點，請遵循 API Gateway 文件中的在 Amazon API Gateway 中建立私有 API 的指示。 2. 設定資源政策以僅允許從 VPC 端點呼叫 API。如需詳細資訊，請參閱 API Gateway 文件中的使用 API Gateway 資源政策控制對 API 的存取。	應用程式開發人員、雲端管理員

任務	Description	所需的技能
建立自訂網域名稱。	1. 為 api.domain.com 建立自訂網域名稱。如需詳細資訊，請參閱 API Gateway 文件中的設定 REST APIs 的自訂網域名稱。 2. 選取已建立的 API 和階段。如需詳細資訊，請參閱 API Gateway 文件中的使用 REST APIs API 映射。	雲端管理員

相關資源

- Amazon API Gateway
- Amazon Route 53
- Application Load Balancer
- AWS PrivateLink
- AWS Certificate Manager

使用 Amazon Cognito 和 IaC 自動化將 Amazon Quick Sight 視覺化元件內嵌至 Web 應用程式 IaC

Ishita Gupta、Saurabh Singh 和 Srishti Wadhwa，Amazon Web Services

摘要

此模式透過使用已註冊的使用者內嵌搭配簡化的 Amazon Cognito 身分驗證，提供將 Amazon Quick Sight 視覺化元件內嵌至 React 應用程式的特殊方法。然後，這些資源會透過基礎設施即程式碼 (IaC) 範本進行部署。與傳統儀表板嵌入不同，此解決方案會隔離特定圖表和圖形，以直接整合到 React 應用程式，大幅改善效能和使用者體驗。

架構會在 Amazon Cognito 使用者管理和 Quick Sight 許可之間建立有效的身分驗證流程：使用者透過 Amazon Cognito 進行身分驗證，並根據 Quick Sight 中的儀表板共用規則存取其授權的視覺化。這種簡化的方法不需要直接存取 Quick Sight 主控台，同時維持強大的安全控制。

透過佈建所有必要基礎設施元件的單一 AWS CloudFormation 範本部署完整環境，包括：

- 使用 AWS Lambda 和 Amazon API Gateway 的無伺服器後端
- 透過 Amazon CloudFront、Amazon Simple Storage Service (Amazon S3) 和 保護前端託管 AWS WAF
- 使用 Amazon Cognito 進行身分管理

所有元件的設定方式是遵循具有最低權限 AWS Identity and Access Management (IAM) 政策、 AWS WAF 保護和end-to-end加密的安全最佳實務。

此解決方案非常適合希望將安全、互動式分析整合到應用程式中的開發團隊和組織，同時保持對使用者存取的精細控制。解決方案使用 AWS 受管服務和自動化來簡化內嵌程序、增強安全性，並確保可擴展性以滿足不斷變化的業務需求。

目標對象和使用案例：

- 想要將分析嵌入 React 應用程式的前端開發人員
- 希望提供每個使用者或以角色為基礎的資料視覺化的軟體即服務 (SaaS) 產品團隊
- 有興趣將 AWS 分析整合到自訂入口網站的解決方案架構師
- 商業智慧 (BI) 開發人員想要向已驗證的使用者公開視覺效果，而不需要完整儀表板存取
- 想要在內部工具中嵌入互動式 Quick Sight 圖表的企業團隊

先決條件和限制

先決條件

若要成功實作此模式，請確定已備妥下列項目：

- 作用中 AWS 帳戶 – AWS 帳戶 具有部署 CloudFormation 堆疊和建立 Lambda、API Gateway、Amazon Cognito、CloudFront 和 Amazon S3 資源之許可的 。
- Amazon Quick Sight 帳戶 – 作用中的 Quick Sight 帳戶，具有至少一個包含視覺效果的儀表板。如需設定說明，請參閱《Amazon Quick 文件》中的教學課程：使用範例資料建立 Amazon Quick Sight 儀表板。
- 開發環境包含：
  - Node.js (16 版或更新版本)
  - npm 或已安裝的 yarn
  - Vite 作為 React 建置工具
  - React (19.1.1 版)
- 儀表板共用 – 儀表板必須在 Quick Sight 中共用，實作器必須登入才能存取內嵌視覺效果或儀表板。

限制

- 此模式使用已註冊的使用者內嵌方法，這需要實作者擁有作用中的 Quick Sight 帳戶。
- 存取權僅限於與實作此模式的已驗證 Quick Sight 使用者明確共用的儀表板和視覺效果。如果實作器沒有正確的存取權限，內嵌 URL 產生將會失敗，而且視覺效果也不會載入。
- CloudFormation 堆疊必須部署在支援 Quick Sight、API Gateway 和 Amazon Cognito AWS 區域 的中。如需區域可用性，請參閱AWS 服務 依區域。

產品版本

- Quick Sight 內嵌 SDK 2.10.1 版
- React 19.1.1 版
- Node.js 16 版或更新版本，以確保與此解決方案中使用的最新 React 和 Vite 版本相容

Architecture

目標架構

下圖顯示此模式的架構和工作流程。

在此工作流程中：

1. 使用者存取應用程式。使用者使用瀏覽器開啟 React Web 應用程式。請求會路由至 CloudFront 分佈，做為應用程式的內容交付網路。
2. AWS WAF 會篩選惡意請求。 在請求到達 CloudFront 之前，它會通過 AWS WAF。 會 AWS WAF 檢查流量，並根據安全規則封鎖任何惡意或可疑的請求。
3. Amazon S3 提供靜態檔案。如果請求是乾淨的，CloudFront 會使用原始存取控制 (OAC) 從私有 S3 儲存貯體擷取靜態前端資產 (HTML、JS、CSS)，並將其交付至瀏覽器。
4. 使用者登入。載入應用程式後，使用者會透過 Amazon Cognito 登入，以驗證使用者並傳回安全 JSON Web 字符 (JWT) 進行授權 API 存取。
5. 應用程式會發出 API 請求。登入後，React 應用程式會對 API Gateway 上的`/get-embed-url`端點進行安全呼叫，並在請求標頭中傳遞 JWT 字符以進行身分驗證。
6. 字符已驗證。API Gateway 會使用 Amazon Cognito 授權方來驗證權杖。如果權杖有效，則請求會繼續；否則，會使用 401 （未授權） 回應拒絕該權杖。
7. 請求會導向 Lambda 進行處理。然後，驗證的請求會轉送到後端 Lambda 函數。此函數負責為請求的 Quick Sight 視覺效果產生內嵌 URL。
8. Lambda 會從 Quick Sight 產生內嵌 URL。IAM 使用具有適當許可的 IAM 角色來呼叫 Quick Sight `GenerateEmbedUrlForRegisteredUser` API，以產生安全、使用者範圍的視覺化 URL。
9. Lambda 會將內嵌 URL 傳回 API Gateway。Lambda 會將產生的內嵌 URL 傳回 API Gateway，做為 JSON 回應的一部分。然後，此回應會準備好交付到前端。
10. 內嵌 URL 會傳送至瀏覽器。內嵌 URL 會傳回瀏覽器做為 API 回應。
11. 視覺效果會顯示給使用者。 React 應用程式會收到回應，並使用 Quick Sight Embedding SDK 將特定視覺效果呈現給使用者。

自動化和擴展

後端和前端部署使用 完全自動化 CloudFormation，可在單一部署中佈建所有必要 AWS 的資源，包括 Amazon Cognito、Lambda、API Gateway、Amazon S3、CloudFront、IAM AWS WAF角色和 Amazon CloudWatch。

此自動化可確保所有環境的基礎設施一致且可重複。所有元件都會自動擴展：Lambda 會根據函數叫用進行調整、CloudFront 全域提供快取的內容，以及 API Gateway 隨傳入請求進行擴展。

工具

AWS 服務

- Amazon Quick Sight 是一種雲端原生商業智慧服務，可協助您建立、管理和嵌入互動式儀表板和視覺效果。
- Amazon API Gateway 會管理做為 React 應用程式與後端服務之間橋接器的 APIs。
- AWS Lambda是一種無伺服器運算服務，此模式用於動態產生安全的 Quick Sight 內嵌 URLs，並根據請求自動擴展。
- Amazon Cognito 為使用者提供身分驗證和授權，並為 API 存取發行安全字符。
- Amazon S3 託管此模式的靜態前端資產，並透過 CloudFront 安全地提供服務。
- Amazon CloudFront 以低延遲提供全球前端內容，並與 整合 AWS WAF 以進行流量篩選。
- AWS WAF 透過套用安全規則和速率限制，保護 Web 應用程式免於惡意流量。
- AWS CloudFormation會在單一部署中自動佈建和設定所有應用程式資源。
- Amazon CloudWatch 會從 Lambda、API Gateway 和 收集日誌和指標， AWS WAF 以進行監控和故障診斷。

開發工具

- React JS 是一種前端架構，此模式用於建置 Web 應用程式並整合內嵌 Quick Sight 視覺效果。
- Vite 是一種建置工具，用於快速開發和最佳化 React 應用程式的生產建置。
- Quick Sight 嵌入 SDK 有助於將 Quick Sight 視覺效果嵌入 React 應用程式，並實現應用程式和視覺效果之間的無縫互動。

程式碼儲存庫

此模式的程式碼可在 React 儲存庫的 GitHub Amazon Quick Sight 視覺化內嵌中找到。 https://github.com/aws-samples/sample-quicksight-visual-embedding

最佳實務

此模式會自動實作下列安全最佳實務：

- 使用 Amazon Cognito 使用者集區進行 JWT 型身分驗證，以及選用的多重要素驗證 (MFA)。
- 使用 Amazon Cognito 授權方保護 APIs，並強制執行所有服務的最低權限 IAM 政策。

- 實作 Quick Sight 註冊的使用者內嵌，並使用讀取器角色自動佈建使用者。
- 透過 CloudFront 和 HTTPS 強制執行支援 TLS 1.2 和更新版本的傳輸中加密。
- 使用 AES-256 for Amazon S3 搭配版本控制和 OAC 加密靜態資料。
- 使用限流和配額設定 API Gateway 用量計劃。
- 使用預留並行和環境變數保護來保護 Lambda。
- 啟用 Amazon S3、CloudFront、Lambda 和 API Gateway 的記錄；使用 CloudWatch 監控服務。
- 加密日誌、套用存取控制，並強制執行非 HTTPS 或未加密上傳的拒絕政策。

此外，我們建議下列事項：

- 使用 CloudFormation 來自動化部署和維護跨環境的一致組態。
- 確定每個使用者都有正確的 Quick Sight 許可來存取內嵌視覺效果。
- 使用 Amazon Cognito 授權方保護 API Gateway 端點，並強制執行所有 IAM 角色的最低權限原則。
- 在環境變數中存放敏感資訊，例如 Amazon Resource Name (ARNs) IDs，而不是硬式編碼。
- 透過減少相依性並改善冷啟動效能來最佳化 Lambda 函數。如需詳細資訊，請參閱 AWS 部落格文章最佳化搭配 SnapStart AWS Lambda 使用進階入門策略的冷啟動效能。
- 將您的 CloudFront 網域新增至 Quick Sight 允許清單，以啟用安全的視覺化內嵌。
- 使用 CloudWatch 和 AWS WAF 來監控記錄、提醒和流量保護的效能和安全性。

其他建議的最佳實務

- 使用自訂網域搭配來自 的 SSL 憑證 AWS Certificate Manager ，以提供安全且品牌化的使用者體驗。
- 使用客戶受管 AWS Key Management Service (AWS KMS) 金鑰加密 Amazon S3 資料和 CloudWatch 日誌，以更好地控制加密。
- 使用地理封鎖、SQL Injection (SQLi)、跨網站指令碼 (XSS) 保護和自訂篩選條件來擴展 AWS WAF 規則，以增強威脅預防。
- 啟用 CloudWatch 警示 AWS Config，以及 AWS CloudTrail 以實現即時監控、稽核和組態合規。
- 套用精細的 IAM 政策、強制執行 API 金鑰輪換，並只在絕對必要時才允許跨帳戶存取。
- 執行定期安全性評估，以確保符合系統和組織控制 2 (SOC 2)、一般資料保護法規 (GDPR) 和健康保險流通與責任法案 (HIPAA) 等合規架構。

史詩

準備環境

任務	Description	所需的技能

複製儲存庫。	將此解決方案的 GitHub 儲存庫複製到本機系統，然後導覽至專案目錄：
	
	`git clone https://github.com/aws-samples/sample-quicksight-visual-embedding.git`
	
	`cd sample-quicksight-visual-embedding`
	
	此儲存庫包含部署解決方案所需的 CloudFormation 範本和 React 原始程式碼。
		應用程式開發人員

部署 CloudFormation 堆疊

任務	Description	所需的技能
部署 範本。	1. 登入 AWS 管理主控台 並開啟 AWS CloudFormation 主控台。 2. 在堆疊頁面上，選擇右上角的建立堆疊，然後選擇使用新資源 （標準）。 3. 在建立堆疊頁面上，針對先決條件 – 準備範本，選擇選擇現有範本。	AWS 管理員

任務	Description	所需的技能
	4. 對於 Specify template (指定範本)，選擇 Upload a template file (上傳範本檔案)。 5. 從本機 GitHub 儲存庫上傳`template.yaml` 檔案，然後選擇下一步。 6. 在指定堆疊詳細資訊頁面上，輸入堆疊名稱（例如，`quicksight-embedding-stack`)。 7. 在設定堆疊選項頁面上，保留預設設定，然後勾選方塊以確認建立 IAM 資源。選擇下一步。 8. 在檢閱和建立頁面上，選擇提交。 如需詳細資訊，請參閱 CloudFormation 文件中的建立和管理堆疊。	
監控堆疊建立。	在事件索引標籤中監控堆疊，直到其狀態為 CREATE_COMPLETE。	AWS 管理員

任務	Description	所需的技能
擷取堆疊輸出。	1. 部署完成時，請選擇輸出索引標籤。 2. 記下金鑰輸出：`ApiGatewayUrl` 、`UserPoolId` 、`CloudFrontDomainName` 、`UserPoolClientId` 和 `S3BucketName` 。您將在後續步驟中使用此資訊。	AWS 管理員

設定前端環境

任務	Description	所需的技能
擷取 Quick Sight 視覺識別符。	1. 開啟 Quick Sight 儀表板。 2. 請注意您計劃嵌入 Web 應用程式中的每個視覺效果的儀表板 ID、工作表 ID 和視覺效果 ID。	Quick Sight 管理員
設定您的本機 React 環境。	若要設定本機 React 環境並將其連結至 AWS 資源，請在本機 GitHub 儲存庫的 `my-app/` 資料夾中建立 `.env` 檔案。將檔案填入： • 您的 AWS 區域 • Amazon Cognito 集區資訊（來自 CloudFormation 堆疊輸出） • API Gateway 端點（來自 CloudFormation 堆疊輸出）	應用程式開發人員

任務	Description	所需的技能

	• Quick Sight 視覺效果 IDs (`DashboardId` 、`SheetId`、`VisualId`)
	
	以下是範例`.env`檔案：
	
	`VITE_AWS_REGION=us-east-1`
	
	`# Cognito Configuration (from CloudFormation outputs)`
	`VITE_USER_POOL_ID=us-east-1_xxxxxxxx`
	`x VITE_USER_POOL_WEB_CLIENT_ID=xxxxxxx`
	`xxxxxxxxxxxxxxxxxx`
	
	`# API Configuration (from CloudFormation outputs)`
	`VITE_API_URL=https:/your-api-id.execute-api.us-east-1.amazonaws.com/prod`
	
	`# QuickSight Visual Configuration`
	`VITE_DASHBOARD_ID=your-dashboard-id`
	`VITE_SHEET_ID=your-sheet-id`
	`VITE_VISUAL_ID=your-visual-id`	

設定使用者身分驗證

任務	Description	所需的技能
在 Cognito 中建立或管理使用者	若要啟用已驗證的使用者存取內嵌 Quick Sight 視覺效果，您必須先在 Amazon Cognito 中建立使用者： 1. 在 Amazon Cognito 主控台上，選擇使用者集區，然後選擇符合的集區 `UserPoolId` （從 CloudFormation 輸出）。 2. 將使用者新增至集區。您可以從兩個選項中選擇： • 在 Amazon Cognito 中，使用電子郵件地址和臨時密碼手動建立使用者（如需詳細資訊，請參閱 Amazon Cognito 文件） —或— • 透過在 `my-app/src/auth/AuthWrapper.jsx` 檔案中設定 在 UI `hideSignUp={false}` 中啟用自我註冊。	AWS 管理員
提供 Quick Sight 儀表板存取	若要提供 Quick Sight 視覺效果的存取權，請將檢視者許可存取權提供給已驗證的使用者： 1. 在 Quick Sight 主控台上，選擇儀表板。 2. 選擇您要共用的儀表板名稱，以選取其名稱。	Quick Sight 管理員

任務	Description	所需的技能
	3. 在儀表板頁面的右上角，選擇共用。 4. 在共用面板中，選擇與特定使用者共用。 5. 若要新增使用者，請選擇邀請使用者或加號 (+) 圖示。 6. 在電子郵件欄位中，輸入收件人的完整電子郵件地址。（此電子郵件必須與使用者的 Amazon Cognito 登入完全相符） 7. 從電子郵件欄位旁的許可選單中，選擇檢視器。 8. 若要傳送邀請和授予存取權，請選擇共用。 每個使用者都會收到一封電子郵件，其中包含儀表板的連結。您可以隨時透過共用功能表修改許可。 如需詳細資訊，請參閱《Amazon Quick 文件》中的授予個別 Amazon Quick Sight 使用者和群組對 Amazon Quick Sight 中儀表板的存取權。	

建置和部署 React 前端

任務	Description	所需的技能

安裝相依性並建置專案。	在 React 應用程式目錄中，執行下列命令來產生最佳化的生產檔案：
	`cd my-app`
	`npm install`
	`npm run build`
		應用程式開發人員

將建置檔案上傳至 Amazon S3。	將所有檔案從`my-app/dist/` 目錄上傳至由 佈建的 S3 儲存貯體 CloudFormation （請勿上傳資料夾本身）。	應用程式開發人員
建立 CloudFront 無效。	在 CloudFront 主控台上，為部署後`/*`重新整理快取內容的路徑建立失效。如需說明，請參閱 CloudFront 文件中的使檔案失效。	AWS 管理員

設定 Quick Sight 允許清單

任務	Description	所需的技能
將 CloudFront 網域新增至 Quick Sight 允許清單。	若要讓您的 CloudFront 網域安全地嵌入 Quick Sight 視覺效果： 1. 在 Quick Sight 主控台上，選擇管理 Quick Sight、網域和內嵌。 2. 將 CloudFront 網域新增至網域方塊（例如，`https://d`	Quick Sight 管理員

任務	Description	所需的技能
	`1234567890abc.cloudfront.net`)。 3. 選擇新增網域。	

存取應用程式並驗證功能

任務	Description	所需的技能
開啟 React 應用程式。	使用 CloudFront 網域（來自 CloudFormation 輸出）在瀏覽器中開啟部署的 React Web 應用程式。	應用程式擁有者
驗證身分驗證。	使用 Amazon Cognito 登入資料來登入應用程式，以透過 API Gateway 驗證身分驗證流程和 JWT 驗證。	應用程式擁有者
驗證內嵌視覺效果。	根據使用者特定的存取許可，確認 Quick Sight 視覺效果在應用程式中正確載入。	應用程式擁有者
驗證 API 和 Lambda 連線。	確認應用程式可以成功呼叫 `/get-embed-url` API 並擷取有效的 Quick Sight 內嵌 URLs 而不會發生錯誤。	應用程式擁有者

監控和維護應用程式

任務	Description	所需的技能
使用 CloudWatch 進行監控。	您可以使用可 AWS 觀測性工具來監控應用程式，並在生產	AWS 管理員

任務	Description	所需的技能
	環境中維持安全、可擴展的效能。 在 CloudWatch 中檢閱 Lambda 日誌、API Gateway 指標和 Amazon Cognito 身分驗證事件，以確保應用程式運作狀態並偵測異常。	
追蹤 AWS WAF 和 CloudFront 日誌。	檢查 AWS WAF 日誌是否有封鎖或可疑的請求，以及 CloudFront 存取日誌是否有效能和快取指標。	AWS 管理員

疑難排解

問題	解決方案
「不允許網域」錯誤	1. 在 Quick Sight 主控台上，選擇管理 Quick Sight、網域和內嵌。 2. 新增您的 CloudFront 網域（例如 `https://d1234567890abc.cloudfront.net`）。 3. 儲存組態並重新載入應用程式。
身分驗證錯誤	可能原因： • `.env` 檔案中的 Amazon Cognito 使用者集區 ID 或應用程式用戶端 ID 不正確。 • 未在 Amazon Cognito 使用者集區中確認使用者。 • API 請求中的 JWT 字符無效或缺少。 解決方案：

問題	解決方案
	• 驗證 .env 檔案中的 Amazon Cognito 組態值。 • 在 Amazon Cognito 主控台上，檢查使用者索引標籤以確認使用者處於作用中狀態。 • 確認授權標頭包含有效的 JWT 字符。 • 在 API Gateway 主控台上，檢查授權方以確認 API Gateway 使用 Amazon Cognito 授權方進行身分驗證。
API Gateway 錯誤	可能原因： • API Gateway 中遺失或設定錯誤的 Amazon Cognito 授權方。 • 呼叫 Quick Sight API 的 Lambda 許可不足。 • .env 與 CloudFormation 輸出之間的 API URL 不相符。 解決方案： • 重新確認 API Gateway 調用 URL 在您的 .env 檔案中。 • 在 API Gateway 主控台上，檢查授權方以驗證 Amazon Cognito 設定。 • 將必要的 IAM 政策與 quicksight:GenerateEmbedUrlForRegisteredUser 連接至 Lambda 執行角色。 • 檢閱 CloudWatch 日誌以取得詳細的 API 或許可錯誤。如需詳細資訊，請參閱 API Gateway 文件。

問題	解決方案
Quick Sight 視覺效果不會載入	可能原因： • 環境組態中的不正確 Quick Sight 儀表板 ID、工作表 ID 或視覺效果 ID。 • 沒有 Quick Sight 使用者存取視覺效果或儀表板的權限。 • API Gateway 中缺少跨來源資源共用 (CORS) 組態。 解決方案： • 從儀表板視覺效果右上角的三點式功能表中確認 Quick Sight IDs。 • 確認 Quick Sight 使用者具有共用儀表板的檢視器角色。 • 在 API Gateway 中啟用 CORS，並重新部署 API。如需詳細資訊，請參閱 API Gateway 文件。 • 檢查瀏覽器主控台是否有 CORS 的網路相關錯誤。

問題	解決方案
「使用者沒有存取權」錯誤	可能原因： • 使用者的電子郵件未與 Quick Sight 儀表板共用。 • Quick Sight 中的使用者角色或許可不正確。 解決方案： 1. 在 Quick Sight 主控台上，開啟儀表板，然後選擇共用、共用儀表板。 2. 新增使用者的電子郵件地址（符合 Amazon Cognito 登入電子郵件）。 3. 指派檢視器許可角色。 4. 要求使用者登出並再次登入以重新整理許可。

相關資源

AWS 文件

- 註冊 Amazon Quick 訂閱
- Amazon Quick Sight 的內嵌分析
- Quick Sight API 參考 – GenerateEmbedUrlForRegisteredUser
- Amazon Cognito 使用者集區
- AWS Lambda 開發人員指南
- Amazon API Gateway 開發人員指南
- CloudWatch 中的基本監控和詳細監控
- AWS CloudFormation 使用者指南
- 《Amazon CloudFront 開發人員指南》
- AWS WAF 開發人員指南

教學課程和影片

- Amazon Quick Sight 的內嵌分析
- Amazon Quick Sight YouTube 教學課程

使用 Green Boost 探索完整堆疊的雲端原生 Web 應用程式開發

Ben Stickley 和 Amiin Samatar，Amazon Web Services

總結

為了回應開發人員不斷演進的需求，Amazon Web Services (AWS) 認識到有效開發雲端原生 Web 應用程式的關鍵需求。AWS 的重點在於協助您克服與在 AWS 雲端部署 Web 應用程式相關的常見障礙。透過利用 TypeScript、AWS Cloud Development Kit (AWS CDK)、React 和 Node.js 等現代技術的功能，此模式旨在簡化和加速開發程序。

此模式以 Green Boost (GB) 工具組為基礎，提供建構 Web 應用程式的實際指南，以充分利用 AWS 的廣泛功能。它可做為全面的藍圖，引導您部署與 Amazon Aurora PostgreSQL 相容版本整合的基本 CRUD（建立、讀取、更新、刪除）Web 應用程式。這可透過使用 Green Boost 命令列界面 (Green Boost CLI) 並建立本機開發環境來完成。

在成功部署應用程式之後，模式會深入探索 Web 應用程式的關鍵元件，包括基礎設施設計、後端和前端開發，以及基本工具，例如用於視覺化的 cdk-dia，促進有效率的專案管理。

先決條件和限制

先決條件

- 已安裝 Git
- 已安裝 Visual Studio Code (VS Code)
- 已安裝 AWS Command Line Interface (AWS CLI)
- 已安裝 AWS CDK Toolkit
- 已安裝 Node.js 18，或已啟用 pnpm 的 Node.js 18
- 已安裝 pnpm，如果它不是 Node.js 安裝的一部分
- 對 TypeScript、AWS CDK、Node.js 和 React 的基本熟悉度
- 作用中的 AWS 帳戶
- 在 中使用 AWS CDK 引導的 AWS 帳戶`us-east-1`。Amazon CloudFront Lambda@Edge 函數的支援 需要 `us-east-1` AWS 區域。
- 在終端機環境中正確設定 AWS 安全登入資料`AWS_ACCESS_KEY_ID`，包括 。
- 對於 Windows 使用者，終端機處於管理員模式（以適應 pnpm 處理節點模組的方式）

產品版本

- 適用於 JavaScript 的 AWS 開發套件第 3 版
- AWS CDK 第 2 版
- AWS CLI 2.2 版
- Node.js 第 18 版
- React 第 18 版

Architecture

目標技術堆疊

- Amazon Aurora PostgreSQL-Compatible Edition
- Amazon CloudFront
- Amazon CloudWatch
- Amazon Elastic Compute Cloud (Amazon EC2)
- AWS Lambda
- AWS Secrets Manager
- Amazon Simple Notification Service (Amazon SNS)
- Amazon Simple Storage Service (Amazon S3)
- AWS WAF

目標架構

下圖顯示使用者請求在與 S3 儲存貯體、Aurora 資料庫、EC2 執行個體互動，最終到達開發人員之前，會先通過 Amazon CloudFront、AWS WAF 和 AWS Lambda。另一方面，管理員使用 Amazon SNS 和 Amazon CloudWatch 進行通知和監控。

若要在部署後深入了解應用程式，您可以使用 cdk-dia 建立圖表，如下列範例所示。

這些圖表展示兩個不同角度的 Web 應用程式架構。cdk-dia 圖表提供 AWS CDK 基礎設施的詳細技術檢視，強調特定的 AWS 服務，例如 Amazon Aurora PostgreSQL 相容和 AWS Lambda。相反地，另一個圖表採用更廣泛的視角，強調資料和使用者互動的邏輯流程。關鍵區別在於細節層級：cdk-dia 深入探索技術複雜性，而第一個圖表提供更以使用者為中心的檢視。

使用 AWS CDK 建立 cdk-dia 圖表涵蓋在史詩中。了解應用程式基礎設施。

工具

AWS 服務

- Amazon Aurora PostgreSQL 相容版本是完全受管的 ACID 相容關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。
- AWS 雲端開發套件 (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- Amazon CloudFront 透過全球資料中心網路提供 Web 內容，進而降低延遲並改善效能，進而加快 Web 內容的發佈速度。
- Amazon CloudWatch 可協助您即時監控 AWS 資源的指標，以及您在 AWS 上執行的應用程式。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Secrets Manager 可協助您以 API 呼叫 Secrets Manager，以程式設計方式擷取秘密，取代程式碼中的硬式編碼登入資料，包括密碼。
- AWS Systems Manager 可協助您管理在 AWS 雲端中執行的應用程式和基礎設施。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。此模式使用 AWS Systems Manager Session Manager。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您存放、保護和擷取任意數量的資料。Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- AWS WAF 是一種 Web 應用程式防火牆，可協助您監控轉送至受保護 Web 應用程式資源的 HTTP 和 HTTPS 請求

其他工具

- Git 是一種開放原始碼的分散式版本控制系統。
- Green Boost 是一種工具組，可在 AWS 上建置 Web 應用程式。
- Next.js 是用於新增功能和最佳化的 React 架構。

- Node.js 是一種事件驅動的 JavaScript 執行期環境，旨在建置可擴展的網路應用程式。
- pgAdmin 是 PostgreSQL 的開放原始碼管理工具。它提供圖形界面，可協助您建立、維護和使用資料庫物件。
- pnpm 是 Node.js 專案相依性的套件管理員。

最佳實務

如需下列建議的詳細資訊，請參閱 Epics 一節：

- 使用 Amazon CloudWatch Dashboards 和警示來監控基礎設施。
- 使用 cdk-nag 執行靜態基礎設施即程式碼 (IaC) 分析，以強制執行 AWS 最佳實務。
- 使用 Systems Manager Session Manager 透過 SSH （安全殼層） 通道建立資料庫連接埠轉送，這比擁有公開的 IP 地址更安全。
- 執行 來管理漏洞`pnpm audit`。
- 使用 ESLint 執行靜態 TypeScript 程式碼分析和 Prettier 標準化程式碼格式，以強制執行最佳實務。

史詩

使用 Aurora PostgreSQL 相容部署 CRUD Web 應用程式

任務	Description	所需的技能
安裝 Green Boost CLI。	若要安裝 Green Boost CLI，請執行下列命令。 `pnpm add -g gboost`	應用程式開發人員
建立 GB 應用程式。	1. 若要使用 Green Boost 建立應用程式，請執行命令`gboost create`。 2. 選擇`CRUD App with Aurora PostgreSQL` 範本。	應用程式開發人員
安裝相依性並部署應用程式。	1. 導覽至專案目錄：`cd <your directory>` 。	應用程式開發人員

任務	Description	所需的技能
	2. 若要安裝相依性，請執行命令 `pnpm i`。 3. 導覽至基礎設施目錄：`cd infra`。 4. 若要在本機部署應用程式，請執行命令 `pnpm deploy:local` 。 這是 中定義之`cdk deploy ...`命令的別名`infra/package.json` 。 等待部署完成（約 20 分鐘）。當您等待時，請在 AWS CloudFormation CloudFormation 堆疊。請注意程式碼中定義的建構如何映射到部署的資源。在 CloudFormation 主控台中檢閱 CDK 建構樹檢視。	

任務	Description	所需的技能
存取應用程式。	在本機部署您的 GB 應用程式之後，您可以使用 CloudFront URL 存取它。URL 列印在終端機輸出中，但可能難以找到。若要更快速地找到它，請使用下列步驟： 1. 開啟執行`pnpm deploy:local` 命令的終端機。 2. 在終端機輸出中尋找類似下列文字的區段。 `myapp5stickbui9C39A55A.CloudFrontDomainName = d1q16n5pof924c.cloudfront.net` 此 URL 將專屬於您的部署。 或者，您也可以存取 Amazon CloudFront 主控台來尋找 CloudFront URL： Amazon CloudFront 1. 登入 AWS 管理主控台並導覽至 CloudFront 服務。 2. 在清單中尋找最新部署的分佈。 複製與分佈相關聯的網域名稱。它看起來類似於	應用程式開發人員

任務	Description	所需的技能
	`your-unique-id.cloudfront.net` 。	

使用 Amazon CloudWatch 進行監控

任務	Description	所需的技能
檢視 CloudWatch Dashboard 。	1. 開啟 CloudWatch 主控台，然後選擇儀表板。 2. 選取名稱為 <appId>-<stageName>-dashboard 的儀表板。 3. 檢閱儀表板。正在監控哪些資源？ 正在記錄哪些指標？此儀表板由開放原始碼建構 cdk-monitoring-constructs 提供。	應用程式開發人員
啟用提醒。	CloudWatch Dashboard 可協助您主動監控 Web 應用程式。若要被動監控 Web 應用程式，您可以啟用提醒。 1. 導覽至 `/infra/src/app/stateless/monitor-stack.ts` ，定義監視器堆疊。 2. 取消以下行的註解，並以`admin@example.com` 您的電子郵件地址取代 。 `onAlarmTopic.addSubscription(new EmailSubscription(`	應用程式開發人員

任務	Description	所需的技能
	"admin@example.com ")); 3. 將下列匯入資訊新增至 檔案頂端。 import { EmailSubs cription } from "aws-cdk-lib/aws-sns-subscriptions"; 4. 在 中infra/，執行下列命令。 cdk deploy "*/monito r" --exclusively. 5. 若要確認您訂閱的 SNS 主題在啟動監控警示時已初始化，請選擇電子郵件訊息中的連結。	

使用 AWS CDK 了解應用程式基礎設施

任務	Description	所需的技能
建立架構圖。	使用 cdk-dia 產生 Web 應用程式的架構圖。視覺化架構有助於改善團隊成員之間的理解和溝通。它提供系統元件及其關係的清晰概觀。 1. 安裝 Graphviz。 2. 在 中infra/，執行命令 pnpm cdk-dia。	應用程式開發人員

任務	Description	所需的技能
	3. 檢視您的 `infra/diagram.png` 。	

任務	Description	所需的技能
使用 cdk-nag 強制執行最佳實務。	使用 cdk-nag 透過強制執行最佳實務，降低安全漏洞和設定錯誤的風險，協助您維護安全且合規的基礎設施。 1. 透過其規則區段探索 cdk-nag 的最佳實務強制執行，包括來自 AWS 解決方案程式庫規則套件的檢查。 2. 若要查看 cdk-nag 如何強制執行規則，請在程式碼中進行變更。例如，在 中`infra/src/app/stateful/data-stacks.ts` ，`storageEncrypted: true` 將變更為 `storageEncrypted: false`。 3. 在 中`infra/`，執行命令 `cdk synth "*/data"`。在合成期間，您會遇到表示違反規則的建置錯誤。 `AwsSolutions-RDS2: The RDS instance or Aurora DB cluster does not have storage encryption enabled.` 此錯誤展示 cdk-nag 如何成為安全機制，以強制執行基礎設施最佳實務並防止安全錯誤設定。	應用程式開發人員

任務	Description	所需的技能

	4. 如有需要，您也可以隱藏不同範圍的規則。例如，若要隱藏 AwsSolutions-RDS2，請在 的執行個體化下方新增下列程式碼DbIamCluster 。
	
	`NagSuppressions.addResourceSuppressions(`
	`  cluster.node.findChild("Resource"),`
	`  [`
	`  {`
	`  id: "AwsSolutions-RDS2",`
	`  reason: "Customer requirement necessitates having unencrypted DB storage",`
	`  },`
	`  ],`
	`);`
	
	5. 禁止之後，請cdk synth "*/data"再次執行 。您的 AWS CDK 應用程式現在應該已成功合成。您可以在 中找到所有隱藏的規則infra/cdk.out/assembly-<appId>-<stageName>/AwsSolutions-<appId>-<stageName>-${stackId}-NagReport.csv 。	

評估資料庫組態和結構描述

任務	Description	所需的技能
取得環境變數。	若要取得所需的環境變數，請使用下列步驟： 1. 若要尋找 `DB_BASTION_ID` ，請登入 主控台，然後導覽至 EC2 主控台。選擇執行個體 （執行中），然後尋找包含 <stageName>-ssm-db-bastion Name 的資料列。執行個體 ID 以 i- 開頭。 2. 若要尋找 `DB_ENDPOINT` ，請在 Amazon Relational Database Service (Amazon RDS) 主控台上，選擇資料庫執行個體，然後選取資料庫識別符開頭為 <appId>-<stageName>-data- 的區域叢集。找到以 rds.amazonaws.com 結尾的寫入器執行個體端點。	應用程式開發人員
建立連接埠轉送。	若要建立連接埠轉送，請使用下列步驟： 1. 安裝 AWS Systems Manager Session Manager 外掛程式。 2. 在 `pnpm db:connect` 中執行以透過堡壘主機建立安全連線`core/`，以啟動連接埠轉送。	應用程式開發人員

任務	Description	所需的技能
	3. 在終端機`Waiting for connections...`，中看到文字後，您的本機電腦和 Aurora 伺服器之間已成功透過 EC2 堡壘主機建立 SSH 通道。	
調整 Systems Manager Session Manager 逾時。	（選用）如果預設的 20 分鐘工作階段逾時太短，您可以在 Systems Manager 主控台中選擇工作階段管理員、偏好設定、編輯、閒置工作階段逾時，將其增加至最多 60 分鐘。	應用程式開發人員

任務	Description	所需的技能
視覺化資料庫。	pgAdmin 是一種易於使用的開放原始碼工具，用於管理 PostgreSQL 資料庫。它可簡化資料庫任務，讓您有效率地建立、管理和最佳化資料庫。本節會逐步引導您安裝 pgAdmin，並使用其功能進行 PostgreSQL 資料庫管理。 1. 在 Object Explorer 中，開啟伺服器的內容（按一下滑鼠右鍵）選單，然後選擇註冊、伺服器。 2. 在一般索引標籤上，輸入名稱欄位的 <appId>-<stageName>。 3. 若要擷取資料庫密碼，請開啟 AWS Secrets Manager 主控台，選取具有由堆疊的 CDK 產生描述的秘密：<appId>-<stageName>-data，然後選擇秘密值卡。選擇擷取秘密值，然後使用密碼金鑰複製秘密值。 4. 在連線索引標籤上，在主機名稱/地址欄位中輸入 0.0.0，然後在使用者名稱欄位中輸入 <appId>_admin。在密碼欄位中，使用您先前擷取的秘密。針對儲存密碼？ 欄位選擇是。 5. 選擇儲存。 6. 若要檢視資料表，請導覽至 <appId>-<stageName	應用程式開發人員

任務	Description	所需的技能
	>、 Databases、<appId>_db、 Schemas、<appId>、Tables。 7. 開啟項目資料表的內容（按一下滑鼠右鍵）選單，然後選取檢視/編輯資料、所有資料列。 8. 探索資料表。	

使用 Node.js 偵錯

任務	Description	所需的技能

對建立項目使用案例進行偵錯。	若要偵錯建立項目使用案例，請遵循下列步驟：
	1. 開啟 `core/src/modules/item/create-item.use-case.ts` 檔案，然後插入下列程式碼。
	`import { fileURLToPath } from "node:url";`
	`// existing create-item.use-case.ts code here`
	`if (process.argv[1] === fileURLToPath(import.meta.url)) {`
	`createItemUseCase({`
		應用程式開發人員

任務	Description	所需的技能
	description: "Item 1's Description", name: "Item 1", }); } 2. 上一個步驟中新增的程式碼可確保在直接執行此模組時呼叫 `createItemUseCase` 函數。在此程式碼區塊中的各行上設定中斷點，以啟動line-by-line偵錯。 1. 開啟 VS 程式碼 JavaScript 偵錯終端機，然後執行 `pnpm tsx core/src/modules/item/create-item.use-case.ts` 以逐line-by-line偵錯執行程式碼。或者，您可以使用`console.log` 陳述式，但當您使用複雜的商業邏輯時，列印陳述式可能不足。Line-by-line偵錯可為您提供更多內容。	

開發前端

任務	Description	所需的技能
設定開發伺服器。	1. 導覽至 `ui/`，然後執行 `pnpm dev`以啟動 Next.js 開發伺服器。	應用程式開發人員

任務	Description	所需的技能
	2. 在 本機存取您的 Web 應用程式 `http://localhost:3000` 。Next.js 開發伺服器設定了對 React 元件進行編輯的快速重新整理即時意見回饋。 3. 嘗試自訂應用程式列顏色。開啟 `ui/src/components/theme/theme.tsx` 檔案並找到定義應用程式列主題的 區段。在 `colorSchemes.light.palette.primary` 區段中，將主值從 更新`colors.lagoon` 為 `colors.carrot` 。進行此變更後，請儲存 檔案並觀察瀏覽器中的更新。 4. 透過修改文字、元件和新增頁面進行實驗。	

Green Boost 工具

任務	Description	所需的技能
設定 monorepo 和 pnpm 套件管理員。	1. 在 GB 儲存庫的根`pnpm-workspace.yaml` 目錄中檢閱 ，並注意如何定義工作區。如需工作區的詳細資訊，請參閱 pnpm 文件。 2. 檢閱 `ui/package.json` ，並注意其如何使用`core/` 套件名稱參考	應用程式開發人員

任務	Description	所需的技能
	中的工作區 "<appId>/core": "workspace:^", 。 3. 觀察 TypeScript 和 ESLint 組態如何集中在 中定義的公用程式套件中packages/ 。然後，應用程式套件會使用此組態core/，例如 infra/、 和 ui/。當您的應用程式擴展並定義更多應用程式套件時，這很有幫助，可以參考公用程式套件，而無需複製組態程式碼。	
執行 pnpm 指令碼。	在儲存庫的根目錄中執行下列命令： 1. 執行 pnpm lint。此命令會使用 ESLint 執行靜態程式碼分析。 2. 執行 pnpm typecheck 。此命令會執行 TypeScript 編譯器來檢查程式碼的類型。 3. 執行 pnpm test。此命令會執行 Vitest 來執行單元測試。 請注意如何在所有工作區中執行這些命令。命令會在每個工作區的 package.json#scripts 欄位中定義。	應用程式開發人員

任務	Description	所需的技能

使用 ESLint 進行靜態程式碼分析。	若要測試 ESLint 的靜態程式碼分析功能，請執行下列動作：
	
	1. 首先，請確定已安裝 VS Code ESLint 延伸模組 (ID：dbaeumer.vscode-eslint)。我們也建議您安裝 VS 程式碼錯誤鏡頭 (ID：usernamehw.errorlens)，以查看內嵌錯誤。
	2. 在您的程式碼中，刻意包含一行使用 eval()函數的程式碼，如下列範例所示。
	
	`const userInput = "import("fs").then((fs) => console.log(fs.readFileSync("/etc/passwd", { encoding: "utf8" })))";`
	`eval(userInput);`
	
	Important
	這僅用於測試目的。使用 eval() 被視為潛在危險，且由於安全風險應予以避免。
	
	3. 包含該eval()行之後，請開啟程式碼編輯器，確認 ESLint 使用紅色魷魚指出程式碼的嗅覺。
		應用程式開發人員

任務	Description	所需的技能
	4. 在 檢閱 ESLint 外掛程式和組態packages/eslint-config-{node,next}/.eslintrc.cjs 。	
管理相依性和漏洞。	1. 若要識別任何常見漏洞與暴露 (CVEs)，請在儲存庫的根pnpm audit目錄中執行 。 您應該會看到找不到已知的漏洞。 2. core/ 透過執行 ，在 中安裝有意易受攻擊的套件pnpm add minimist@0.2.3 ，然後執行 pnpm audit。請注意要報告的漏洞。 3. core/ 執行 ，在 中解除安裝易受攻擊的套件pnpm remove minimist。	應用程式開發人員

任務	Description	所需的技能
使用 Husky 預先遞交掛鉤。	1. 在整個儲存庫中對 TypeScript 檔案進行一些小變更。這些變更可以和新增註解一樣基本。 2. 使用 和 來暫存`git add -A`和遞交這些變更`git commit -m "test husky"`。 Husky 預先遞交勾點觸發器在 中定義`.husky/pre-commit`，執行命令 `pnpm lint-staged` 。 3. 觀察 lint-staged 如何在 Git 已暫存的檔案上執行 `*/.lintstagedrc.js` 整個儲存庫檔案中指定的命令。 這些工具是有助於防止錯誤程式碼進入您應用程式的機制。	應用程式開發人員

向下拉動基礎設施

任務	Description	所需的技能
從您的帳戶中移除部署。	1. 若要縮減您在第一個史詩中佈建的基礎設施，請在 `pnpm destroy:local` 中執行 `infra/`。 2. 在 `pnpm destroy:local` 完成後等待 15 分鐘，然後在 Lambda 主控台	應用程式開發人員

任務	Description	所需的技能
	中搜尋您的應用程式 ID，以刪除保留的 Lambda@Edge 函數。Lambda@Edge 函數會複寫。這使得它們難以刪除。如需刪除 Lambda@Edge 函數的詳細資訊，請參閱 CloudFront 文件。	

疑難排解

問題	解決方案
無法建立連接埠轉送	確保您的 AWS 登入資料已正確設定並具有必要的許可。 再次檢查堡壘主機 ID (`DB_BASTION_ID`) 和資料庫端點 (`DB_ENDPOINT`) 環境變數是否已正確設定。 如果您仍然遇到問題，請參閱 AWS 文件來疑難排解 SSH 連線和 Session Manager。
網站未在 上載入 `localhost:3000`	確認終端機輸出指出連接埠轉送成功，包括轉送地址。 確保本機電腦上沒有使用連接埠 3000 的衝突程序。 確認 Green Boost 應用程式已正確設定並在預期的連接埠 (3000) 上執行。 檢查您的 Web 瀏覽器是否有任何可能封鎖本機連線的安全擴充功能或設定。
本機部署期間的錯誤訊息 (`pnpm deploy:local`)	請仔細檢閱錯誤訊息，以找出問題的原因。

問題	解決方案
	確認已正確設定必要的環境變數和組態檔案。

相關資源

- AWS CDK 文件
- Green Boost 文件
- Next.js 文件
- Node.js 文件
- React 文件
- TypeScript 文件

使用 AWS CodeBuild 從 GitHub 執行 Node.js 應用程式的單元測試

Thomas Scott 和 Jean-Baptiste Guillois，Amazon Web Services

總結

此模式提供 Node.js 遊戲 API 的範例原始程式碼和金鑰單位測試元件。它還包含使用 AWS CodeBuild 從 GitHub 儲存庫執行這些單元測試的說明，作為持續整合和持續交付 (CI/CD) 工作流程的一部分。

單元測試是一種軟體開發程序，其中應用程式的不同部分稱為單元，會個別獨立測試以正確操作。測試會驗證程式碼的品質，並確認其如預期般運作。其他開發人員也可以透過諮詢測試，輕鬆熟悉您的程式碼庫。單元測試可減少未來的重構時間，協助工程師更快地趕上您的程式碼基礎，並提供對預期行為的信心。

單元測試涉及測試個別函數，包括 AWS Lambda 函數。若要建立單元測試，您需要測試架構和驗證測試的方式（聲明）。此模式中的程式碼範例使用Mocha 測試架構和中國聲明程式庫。

如需單元測試和測試元件範例的詳細資訊，請參閱其他資訊一節。

先決條件和限制

- 具有正確 CodeBuild 許可的作用中 AWS 帳戶
- GitHub 帳戶（請參閱註冊說明)
- Git（請參閱安裝說明)
- 用於進行變更並將程式碼推送至 GitHub 的程式碼編輯器

Architecture

此模式會實作下圖所示的架構。

工具

工具

- Git 是一種版本控制系統，可用於程式碼開發。
- AWS CodeBuild 是全受管的持續整合服務，可編譯原始程式碼、執行測試，並產生準備好部署的軟體套件。使用 CodeBuild，您不需要佈建、管理和擴展自己的建置伺服器。CodeBuild 會持續擴展並同時處理多個組建，所以您的組建不必排入佇列中等候。您可以利用預先封裝好的組建環境立即開始

使用，或是建立自訂的組建環境來使用您自己的組建工具。使用 CodeBuild 時，將依據您使用運算資源的分鐘數計費。

Code

此模式的原始碼可在 GitHub 範例遊戲單位測試應用程式儲存庫中取得。您可以從此範例 （選項 1) 建立自己的 GitHub 儲存庫，或針對此模式直接使用範例儲存庫 （選項 2)。請遵循下一節中每個選項的指示。您遵循的選項將取決於您的使用案例。

史詩

選項 1 - 使用 CodeBuild 在您的個人 GitHub 儲存庫上執行單元測試

任務	Description	所需的技能
根據範例專案建立您自己的 GitHub 儲存庫。	1. 登入 GitHub。 2. 建立新的儲存庫。如需說明，請參閱 GitHub 文件。 3. 複製範例儲存庫，並將其推送至您帳戶中的新儲存庫。	應用程式開發人員、AWS 管理員、AWS DevOps
建立新的 CodeBuild 專案。	1. 登入 AWS 管理主控台，並在 https：//https://console.aws.amazon.com/codesuite/codebuild/home 開啟 CodeBuild 主控台。 2. 選擇 Create build project (建立建置專案)。 3. 在專案組態區段中，針對專案名稱，輸入 aws-tests-sample-node-js。 4. 在來源區段中，針對來源提供者選擇 GitHub。 5. 對於儲存庫，選擇 GitHub 帳戶中的儲存庫，然後將 URL 貼到新建立的 GitHub 儲存庫。	應用程式開發人員、AWS 管理員、AWS DevOps

任務	Description	所需的技能
	6. 在主要來源 Webhook 事件區段中，每次將程式碼變更推送至此儲存庫時，selectRebuild。 7. 針對事件類型，選擇 PUSH。 8. 在環境區段中，選擇受管映像、Amazon Linux 和最新的映像。 9. 保留所有其他選項的預設設定，然後選擇建立建置專案。	
啟動建置。	在 Review (檢閱) 頁面上，選擇 Start build (開始建置) 來執行建置。	應用程式開發人員、AWS 管理員、AWS DevOps

選項 2 - 使用 CodeBuild 在公有儲存庫上執行單位測試

任務	Description	所需的技能
建立新的 CodeBuild 組建專案。	1. 登入 AWS 管理主控台，並在 https：//https://console.aws.amazon.com/codesuite/codebuild/home 開啟 CodeBuild 主控台。 2. 選擇 Create build project (建立建置專案)。 3. 在專案組態區段中，針對專案名稱，輸入 aws-tests-sample-node-js。 4. 在來源區段中，針對來源提供者選擇 GitHub。	應用程式開發人員、AWS 管理員、AWS DevOps

任務	Description	所需的技能
	5. 針對儲存庫，選擇公有儲存庫，然後貼上 URL：https：//https://github.com/aws-samples/node-js-tests-sample。 6. 在環境區段中，選擇受管映像、Amazon Linux 和最新的映像。 7. 保留所有其他選項的預設設定，然後選擇建立建置專案。	
啟動建置。	在 Review (檢閱) 頁面上，選擇 Start build (開始建置) 來執行建置。	應用程式開發人員、AWS 管理員、AWS DevOps

分析單位測試

任務	Description	所需的技能
檢視測試結果。	在 CodeBuild 主控台中，檢閱 CodeBuild 任務的單位測試結果。它們應該符合其他資訊區段中顯示的結果。 這些結果會驗證 GitHub 儲存庫與 CodeBuild 的整合。	應用程式開發人員、AWS 管理員、AWS DevOps
套用 Webhook。	您現在可以套用 Webhook，因此每當您將程式碼變更推送至儲存庫的主分支時，就會自動啟動組建。如需說明，請參閱CodeBuild 文件。	應用程式開發人員、AWS 管理員、AWS DevOps

相關資源

- 遊戲單位測試應用程式範例 （具有範本程式碼的 GitHub 儲存庫）
- AWS CodeBuild 文件
- GitHub Webhook 事件 (CodeBuild 文件）
- 建立新的儲存庫 (GitHub 文件）

其他資訊

單位測試結果

在 CodeBuild 主控台中，您應該會在專案成功建置後看到下列測試結果。

單元測試元件範例

本節說明用於單元測試的四種測試元件類型：聲明、間葉、短軸和模擬。它包含每個元件的簡短說明和程式碼範例。

聲明

宣告用於驗證預期結果。這是重要的測試元件，因為它會驗證指定函數的預期回應。下列範例聲明會在初始化新遊戲時，驗證傳回的 ID 介於 0 到 1000 之間。

```
const { expect }  = require('chai');
const { Game } = require('../src/index');

describe('Game Function Group', () => {
 it('Check that the Game ID is between 0 and 1000', function() {
      const game = new Game();
      expect(game.id).is.above(0).but.below(1000)
 });
});
```

間葉

間諜軟體用於觀察函數執行時所發生的情況。例如，您可能想要驗證已正確呼叫函數。下列範例顯示在遊戲類別物件上呼叫啟動和停止方法。

```
const { expect }  = require('chai');
```

```
const { spy } = require('sinon');

const { Game } = require('../src/index');

describe('Game Function Group', () => {
   it('should verify that the correct function is called', () => {
      const spyStart = spy(Game.prototype, "start");
      const spyStop = spy(Game.prototype, "stop");

      const game = new Game();
      game.start();
      game.stop();

      expect(spyStart.called).to.be.true
      expect(spyStop.called).to.be.true
    });
});
```

Stubs

Stub 用於覆寫函數的預設回應。這在函數提出外部請求時特別有用，因為您想要避免從單位測試提出外部請求。（外部請求更適合用於整合測試，這可以實際測試不同元件之間的請求。）在下列範例中，stub 會從 getId 函數強制傳回 ID。

```
const { expect }  = require('chai');
const {.stub } = require('sinon');

const { Game } = require('../src/index');

describe('Game Function Group', () =>  {
   it('Check that the Game ID is between 0 and 1000', function() {
      let generateIdStub = stub(Game.prototype, 'getId').returns(999999);

      const game = new Game();

      expect(game.getId).is.equal(999999);

      generateIdStub.restore();
    });
});
```

模擬

模擬是一種仿造方法，具有用於測試不同案例的預先程式設計行為。模擬可以被視為長短的格式，並且可以同時執行多個任務。在下列範例中，模擬會用來驗證三個案例：

- 函數稱為
- 函數使用引數呼叫
- 函數會傳回整數 9

```
const { expect }  = require('chai');
const {.mock } = require('sinon');

const { Game } = require('../src/index');

describe('Game Function Group', () =>  {
   it('Check that the Game ID is between 0 and 1000', function() {
      let mock = mock(Game.prototype).expects('getId').withArgs().returns(9);

      const game = new Game();
      const id = get.getId();

      mock.verify();
      expect(id).is.equal(9);
    });
});
```

使用 AWS Lambda 在六邊形架構中建構 Python 專案

Furkan Oruc、Dominik Goby、Darius Kunce 和 Michal Ploski，Amazon Web Services

總結

此模式說明如何使用 AWS Lambda 在六邊形架構中建構 Python 專案。模式使用 AWS 雲端開發套件 (AWS CDK) 做為基礎設施做為程式碼 (IaC) 工具、使用 Amazon API Gateway 做為 REST API，以及使用 Amazon DynamoDB 做為持久性層。六角形架構遵循網域驅動的設計原則。在六邊形架構中，軟體包含三個元件：網域、連接埠和轉接器。如需六邊形架構及其優點的詳細資訊，請參閱 指南在 AWS 上建置六邊形架構。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Python 的經驗
- 熟悉 AWS Lambda、AWS CDK、Amazon API Gateway 和 DynamoDB
- GitHub 帳戶 （ 請參閱註冊說明)
- Git （ 請參閱安裝說明)
- 用於進行變更並將程式碼推送至 GitHub 的程式碼編輯器 （ 例如，Visual Studio Code 或 JetBrains PyCharm)
- 已安裝 Docker，且 Docker 協助程式已啟動並執行

產品版本

- Git 2.24.3 版或更新版本
- Python 3.7 版或更新版本
- AWS CDK v2
- Poetry 1.1.13 版或更新版本
- AWS Lambda Powertools for Python 1.25.6 版或更新版本
- pytest 7.1.1 版或更新版本
- Moto 3.1.9 版或更新版本
- pydantic 1.9.0 版或更新版本
- Boto3 1.22.4 版或更新版本

- mypy-boto3-dynamodb 1.24.0 版或更新版本

Architecture

目標技術堆疊

目標技術堆疊包含使用 API Gateway、Lambda 和 DynamoDB 的 Python 服務。服務使用 DynamoDB 轉接器來保留資料。它提供使用 Lambda 做為進入點的函數。服務使用 Amazon API Gateway 公開 REST API。API 使用 AWS Identity and Access Management (IAM) 進行用戶端身分驗證。

目標架構

為了說明實作，此模式會部署無伺服器目標架構。用戶端可以將請求傳送至 API Gateway 端點。API Gateway 會將請求轉送至實作六邊形架構模式的目標 Lambda 函數。Lambda 函數會在 DynamoDB 資料表上執行建立、讀取、更新和刪除 (CRUD) 操作。

> **Important**
>
> 此模式已在 PoC 環境中測試。您必須先執行安全性審查，以識別威脅模型並建立安全程式碼庫，才能將任何架構部署至生產環境。

API 支援產品實體的五個操作：

- `GET /products` 會傳回所有產品。
- `POST /products` 會建立新的產品。
- `GET /products/{id}` 會傳回特定產品。
- `PUT /products/{id}` 會更新特定產品。
- `DELETE /products/{id}` 會刪除特定產品。

您可以使用下列資料夾結構來組織專案，以遵循六邊形架構模式：

```
app/  # application code
|--- adapters/  # implementation of the ports defined in the domain
     |--- tests/  # adapter unit tests
|--- entrypoints/  # primary adapters, entry points
```

```
     |--- api/  # api entry point
          |--- model/  # api model
          |--- tests/  # end to end api tests
|--- domain/  # domain to implement business logic using hexagonal architecture
     |--- command_handlers/  # handlers used to execute commands on the domain
     |--- commands/  # commands on the domain
     |--- events/  # events triggered via the domain
     |--- exceptions/  # exceptions defined on the domain
     |--- model/  # domain model
     |--- ports/  # abstractions used for external communication
     |--- tests/  # domain tests
|--- libraries/  # List of 3rd party libraries used by the Lambda function
infra/  # infrastructure code
simple-crud-app.py  # AWS CDK v2 app
```

工具

AWS 服務

- Amazon API Gateway 是一項全受管服務，可讓開發人員輕鬆建立、發佈、維護、監控和保護任何規模APIs。
- Amazon DynamoDB 是全受管、無伺服器、鍵值的 NoSQL 資料庫，旨在以任何規模執行高效能應用程式。
- AWS Lambda 是一種無伺服器、事件驅動的運算服務，可讓您為幾乎任何類型的應用程式或後端服務執行程式碼，而無需佈建或管理伺服器。您可以從超過 200 個 AWS 服務和軟體即服務 (SaaS) 應用程式啟動 Lambda 函數，並且只需支付使用的費用。

工具

- Git 會使用此模式做為程式碼開發的版本控制系統。
- Python 用作此模式的程式設計語言。Python 提供高階資料結構和物件導向程式設計的方法。AWS Lambda 提供內建 Python 執行期，可簡化 Python 服務的操作。
- Visual Studio Code 用作開發和測試此模式的 IDE。您可以使用支援 Python 開發的任何 IDE（例如 PyCharm)。
- AWS Cloud Development Kit (AWS CDK) 是一種開放原始碼軟體開發架構，可讓您使用熟悉的程式設計語言來定義雲端應用程式資源。此模式使用 CDK 將雲端基礎設施寫入和部署為程式碼。
- Poetry 用於管理 模式中的相依性。
- AWS CDK 會使用 Docker 來建置 Lambda 套件和 layer。

Code

此模式的程式碼可在 GitHub Lambda 六邊形架構範例儲存庫中使用。

最佳實務

若要在生產環境中使用此模式，請遵循下列最佳實務：

- 在 AWS Key Management Service (AWS KMS) 中使用客戶受管金鑰來加密 Amazon CloudWatch 日誌群組和 Amazon DynamoDB 資料表。
- 設定 Amazon API Gateway 的 AWS WAF，以僅允許從組織的網路存取。
- 如果 IAM 不符合您的需求，請考慮 API Gateway 授權的其他選項。例如，您可以使用 Amazon Cognito 使用者集區或 API Gateway Lambda 授權方。
- 使用 DynamoDB 備份。
- 使用虛擬私有雲端 (VPC) 部署設定 Lambda 函數，以將網路流量保留在雲端內。
- 更新允許跨來源資源共用 (CORS) 預檢的原始伺服器組態，以限制只能存取請求的原始伺服器網域。
- 使用 cdk-nag 檢查 AWS CDK 程式碼的安全性最佳實務。
- 請考慮使用程式碼掃描工具來尋找程式碼中常見的安全問題。例如，Bandit 是一種旨在尋找 Python 程式碼中常見安全問題的工具。Pip-audit 會掃描 Python 環境是否有已知漏洞的套件。

此模式使用 AWS X-Ray 透過應用程式的進入點、網域和轉接器追蹤請求。AWS X-Ray 可協助開發人員識別瓶頸並判斷高延遲，以改善應用程式效能。

史詩

初始化專案

任務	Description	所需的技能
建立您自己的儲存庫。	1. 登入 GitHub。 2. 建立新的儲存庫。如需說明，請參閱 GitHub 文件。 3. 複製此模式的範例儲存庫，並將其推送至您帳戶中的新儲存庫。	應用程式開發人員

任務	Description	所需的技能

安裝依存項目。	1. 安裝 Poetry。
	`pip install poetry`
	2. 從根目錄安裝套件。下列命令會安裝應用程式和 AWS CDK 套件。它也會安裝執行單元測試所需的開發套件。所有已安裝的套件都會放置在新的虛擬環境中。
	`poetry install`
	3. 若要查看已安裝套件的圖形表示，請執行下列命令。
	`poetry show --tree`
	4. 更新所有相依性。
	`poetry update`
	5. 在新建立的虛擬環境中開啟新的 shell。它包含所有已安裝的相依性。
	`poetry shell`
		應用程式開發人員

任務	Description	所需的技能

設定您的 IDE。	我們建議 Visual Studio Code，但您可以使用任何支援 Python 的 IDE。下列步驟適用於 Visual Studio Code。
	
	1. 更新 `.vscode/settings` 檔案。
	
	`{ "python.testing.pytestArgs": [ "app/adapters/tests", "app/entrypoints/api/tests", "app/domain/tests" ], "python.testing.unittestEnabled": false, "python.testing.pytestEnabled": true, "python.envFile": "${workspaceFolder}/.env", }`
	
	2. 在專案的根目錄中建立 `.env` 檔案。這可確保專案的根目錄包含在中，PYTHONPATH 以便 `pytest` 可以找到它並正確探索所有套件。
	
	`PYTHONPATH=.`
		應用程式開發人員

任務	Description	所需的技能
執行單位測試，選項 1：使用 Visual Studio 程式碼。	1. 選擇由 Poetry 管理之虛擬環境的 Python 解譯器。 2. 從 Test Explorer 執行測試。	應用程式開發人員

執行單元測試，選項 2：使用 shell 命令。	1. 在虛擬環境中啟動新的 shell。
	`poetry shell`
	2. 從根目錄執行 `pytest`命令。
	`python -m pytest`
	或者，您也可以直接從 Poetry 執行命令。
	`poetry run python -m pytest`
		應用程式開發人員

部署和測試應用程式

任務	Description	所需的技能
請求臨時登入資料。	若要在執行 時在 Shell 上擁有 AWS 登入資料`cdk deploy`，請使用 AWS IAM Identity Center (AWS Single Sign-On 的後續版本）建立臨時登入資料。如需說明，請參閱部落格文章如何擷取 CLI 搭配 AWS IAM Identity Center 使用的短期憑證。	應用程式開發人員、AWS DevOps

任務	Description	所需的技能

部署應用程式。	1. 安裝 AWS CDK v2。
	`npm install -g aws-cdk`
	如需詳細資訊，請參閱 AWS CDK 文件。
	2. 將 AWS CDK 引導到您的帳戶和區域。
	`cdk bootstrap aws://12345678900/us-east-1 --profile aws-profile-name`
	3. 使用 AWS 設定檔將應用程式部署為 AWS CloudFormation 堆疊。
	`cdk deploy --profile aws-profile-name`
		應用程式開發人員、AWS DevOps

測試 API，選項 1：使用 主控台。	使用 API Gateway 主控台來測試 API。如需 API 操作和請求/回應訊息的詳細資訊，請參閱 GitHub 儲存庫中 readme 檔案的 API 用量區段。	應用程式開發人員、AWS DevOps

任務	Description	所需的技能
測試 API，選項 2：使用 Postman。	如果您想要使用 Postman 等工具： 1. 安裝 Postman 做為獨立應用程式或瀏覽器擴充功能。 2. 複製 API Gateway 的端點 URL。其格式如下。 `https://{api-id}.execute-api.{region}.amazonaws.com/{stage}/{path}` 3. 在授權索引標籤中設定 AWS 簽章。如需說明，請參閱有關啟用 API Gateway REST APIs AWS re：Post 文章。 4. 使用 Postman 將請求傳送至您的 API 端點。	應用程式開發人員、AWS DevOps

開發服務

任務	Description	所需的技能
撰寫商業網域的單位測試。	1. 使用檔案名稱字首在 `app/domain/tests` 資料夾中建立 Python `test_` 檔案。 2. 使用下列範例建立新的測試方法，以測試新的商業邏輯。 `def test_create_product_should_`	應用程式開發人員

任務	Description	所需的技能

	store_in_repository():
	    # Arrange
	    command = create_product_command.CreateProductCommand(
	        name="Test Product",
	        description="Test Description",
	    )
	    # Act
	    create_product_command_handler.handle_create_product_command(
	        command=command, unit_of_work=mock_unit_of_work
	    )
	# Assert
	3. 在 app/domain/commands 資料夾中建立命令類別。
	4. 如果功能是新的，請在 app/domain/command_handlers 資料夾中為命令處理常式建立 stub。
	5. 執行單元測試以查看失敗，因為仍然沒有商業邏輯。
	python -m pytest	

任務	Description	所需的技能

實作命令和命令處理常式。	1. 在新建立的命令處理常式檔案中實作商業邏輯。
	2. 對於與外部系統互動的每個相依性，請在 app/domain/ports 資料夾中宣告抽象類別。
	
	`class ProductsRepository(ABC):`
	`  @abstractmethod`
	`  def add(self, product: product.Product) -> None:`
	`  ...`
	
	`class UnitOfWork(ABC):`
	`  products: ProductsRepository`
	
	`  @abstractmethod`
	`  def commit(self) -> None:`
	`  ...`
	
	`  @abstractmethod`
	`  def __enter__(self) -> typing.Any:`
	`  ...`
	
	`  @abstractmethod`
	`  def __exit__(self, *args) -> None:`
	`  ...`
	
	3. 使用抽象連接埠類別做為類型註釋，更新命令處理常式
		應用程式開發人員

任務	Description	所需的技能

	簽章以接受新宣告的相依性。
	`def handle_create_product_command(`
	`  command: create_product_command.CreateProductCommand,`
	`  unit_of_work: unit_of_work.UnitOfWork,`
	`) -> str:`
	`  ...`
	4. 更新單位測試，以模擬命令處理常式所有宣告相依性的行為。
	`  # Arrange`
	`  mock_unit_of_work = unittest.mock.create_autospec(`
	`  spec=unit_of_work.UnitOfWork, instance=True`
	`  )`
	`  mock_unit_of_work.products = unittest.mock.create_autospec(`
	`  spec=unit_of_work.ProductsRepository, instance=True`
	`  )`
	5. 更新測試中的聲明邏輯，以檢查預期的相依性叫用。	

任務	Description	所需的技能

	<pre> # Assert
	    mock_unit
	_of_work.commit.as
	sert_called_once()
	    product =
	 mock_unit_of_work.
	products.add.call_
	args.args[0]
	
	    assertpy.
	assert_that(produc
	t.name).is_equal_t
	o("Test Product")
	    assertpy.
	assert_that(produc
	t.description).is_
	equal_to("Test
	 Description")</pre> 6. 執行單元測試以查看是否成功。
	<pre>python -m pytest</pre>	

任務	Description	所需的技能

撰寫次要轉接器的整合測試。	1. 使用 `test_` 做為檔案名稱字首，在 `app/adapters/tests` 資料夾中建立測試檔案。
	2. 使用 Moto 程式庫模擬 AWS 服務。
	`@pytest.fixture def mock_dynamodb(): with moto.mock_dynamodb(): yield boto3.resource("dynamodb", region_name="eu-central-1")`
	3. 為轉接器的整合測試建立新的測試方法。
	`def test_add_and_commit_should_store_product(mock_dynamodb): # Arrange unit_of_work = dynamodb_unit_of_work.DynamoDBUnitOfWork( table_name=TEST_TABLE_NAME, dynamodb_client=mock_dynamodb.meta.client ) current_time = datetime.datetime.now(datetime.timezone.utc).isoformat()`
		應用程式開發人員

任務	Description	所需的技能

	<pre> new_product_id = str(uuid.uuid4())
	    new_product = product.Product(
	        id=new_product_id,
	        name="test-name",
	        description="test-description",
	        createDate=current_time,
	        lastUpdateDate=current_time,
	    )
	
	    # Act
	    with unit_of_work:
	        unit_of_work.products.add(new_product)
	        unit_of_work.commit()
	
	    # Assert</pre> 4. 在 app/adapters 資料夾中建立轉接器類別。使用連接埠資料夾中的抽象類別做為基礎類別。
	5. 執行單元測試以查看失敗，因為仍然沒有邏輯。
	<pre>python -m pytest</pre>	

任務	Description	所需的技能

實作次要轉接器。	1. 在新建立的轉接器檔案中實作邏輯。
	2. 更新測試聲明。
	
	`# Assert`
	`    with unit_of_work_readonly:`
	`        product_from_db = unit_of_work_readonly.products.get(new_product_id)`
	
	`    assertpy.assert_that(product_from_db).is_not_none()`
	`    assertpy.assert_that(product_from_db.dict()).is_equal_to(`
	`        {`
	`            "id": new_product_id,`
	`            "name": "test-name",`
	`            "description": "test-description",`
	`            "createDate": current_time,`
	`            "lastUpdateDate": current_time,`
	`        }`
	`    )`
	
	3. 執行單元測試以查看是否成功。
		應用程式開發人員

任務	Description	所需的技能
	`python -m pytest`	

任務	Description	所需的技能
撰寫end-to-end測試。	1. 使用 `test_`做為檔案名稱字首，在 `app/entry points/api/tests` 資料夾中建立測試檔案。 2. 建立 Lambda 內容固定裝置，供測試用來呼叫 Lambda。 `@pytest.fixture def lambda_co ntext(): @dataclass class LambdaCon text: function_ name: str = "test" memory_li mit_in_mb: int = 128 invoked_f unction_arn: str = "arn:aws:lambda:eu -west-1:809313241: function:test" aws_reque st_id: str = "52fdfc07-2182-154 f-163f-5f0f9a621d7 2" return LambdaCon text()` 3. 建立 API 調用的測試方法。 `def test_crea te_product(lambda_ context): # Arrange name = "TestName"`	應用程式開發人員

任務	Description	所需的技能

	<pre> description =
	 "Test description"
	    request =
	 api_model.CreatePr
	oductRequest(name=
	name, descripti
	on=description)
	
	    minimal_event =
	 api_gateway_proxy_
	event.APIGatewayPr
	oxyEvent(
	        {
	            "path": "/
	products",
	            "httpMeth
	od": "POST",
	            "requestC
	ontext": { #
	 correlation ID
	
	  "requestId":
	 "c6af9ac6-7b61-11e
	6-9a41-93e8deadbee
	f"
	            },
	            "body":
	 json.dumps(request
	.dict()),
	        }
	    )
	
	    create_pr
	oduct_func_mock =
	 unittest.mock.crea
	te_autospec(
	        spec=crea
	te_product_command
	_handler.handle_cr
	eate_product_comma
	nd
	    )</pre>	

任務	Description	所需的技能

	<pre> handler.c
	reate_product_comm
	and_handler.handle
	_create_product_co
	mmand = (
	        create_pr
	oduct_func_mock
	    )
	
	    # Act
	    handler.h
	andler(minimal_eve
	nt, lambda_context)</pre>	

	4. 執行單元測試以查看失敗，因為仍然沒有邏輯。 <pre>python -m pytest</pre>	

任務	Description	所需的技能

實作主要轉接器。	1. 建立 API 商業邏輯的函數，並將其宣告為 API 資源。
	`@tracer.capture_method`
	`@app.post("/products")`
	`@utils.parse_event(model=api_model.CreateProductRequest, app_context=app)`
	`def create_product(`
	`request: api_model.CreateProductRequest,`
	`) -> api_model.CreateProductResponse:`
	`"""Creates a product."""`
	`...`
	Note
	您看到的所有裝飾項目都是適用於 Python 的 AWS Lambda Powertools 程式庫的功能。如需詳細資訊，請參閱 AWS Lambda Powertools for Python 網站。
	2. 實作 API 邏輯。
	`id=create_product_command_handler.ha`
		應用程式開發人員

任務	Description	所需的技能
	ndle_create_produc t_command(command=c reate_product_comm and.CreateProductC ommand(name=requ est.name, descripti on=request.descrip tion,), unit_of_w ork=unit_of_work,) response = api_model.CreatePr oductResponse(id=i d) return response. dict() 3. 執行單元測試以查看是否成功。 python -m pytest	

相關資源

APG 指南

- 在 AWS 上建置六邊形架構

AWS 參考

- AWS Lambda 文件
- AWS CDK 文件
  - 您的第一個 AWS CDK 應用程式

- API Gateway 文件
  - 使用 IAM 許可控制對 API 的存取
  - 使用 API Gateway 主控台測試 REST API 方法
- Amazon DynamoDB 文件

工具

- git-scm.com 網站
- 安裝 Git
- 建立新的 GitHub 儲存庫
- Python 網站
- 適用於 Python 的 AWS Lambda Powertools
- Postman 網站
- Python 模擬物件程式庫
- Poetry 網站

IDE

- Visual Studio Code 網站
- PyCharm 網站

更多模式

- 使用 AWS Fargate、AWS PrivateLink 和 Network Load Balancer，在 Amazon ECS 上私下存取容器應用程式
- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon ECS 上私下存取容器應用程式
- 將儲存 AWS CodeCommit 庫與另一個帳戶中 AWS 帳戶 的 Amazon SageMaker AI Studio Classic 建立關聯
- 自動化刪除 AWS CloudFormation 堆疊和相關聯的資源
- 使用 根據 IP 地址或地理位置限制存取 AWS WAF
- 使用 AWS Amplify 建置無伺服器 React Native 行動應用程式
- 使用 AWS CodeCommit、AWS CodePipeline 和 AWS Device Farm 建置和測試 iOS 應用程式
- 使用 NLog 在 Amazon CloudWatch Logs 中設定 .NET 應用程式的記錄
- 使用與靜態 IP 地址相關聯的端點，整合 Amazon S3 預先簽章的 URL 產生和物件下載
- 使用 Amazon EFS 在 EC2 執行個體上建立 Amazon ECS 任務定義並掛載檔案系統
- 在 Amazon EKS 叢集上部署以 gRPC 為基礎的應用程式，並使用 Application Load Balancer 存取它
- 在聊天應用程式自訂動作和 中使用 Amazon Q Developer 部署 ChatOps 解決方案來管理 SAST 掃描結果 CloudFormation
- 使用 Terraform 部署 CloudWatch Synthetics Canary
- 使用 AWS Fargate 在 Amazon ECS 上部署 Java 微服務
- AWS 使用 Terraform 和 Amazon Bedrock 在 上部署 RAG 使用案例
- 使用 Terraform 在 AWS Wavelength 區域中部署資源
- 在 Amazon API Gateway 中使用自訂網域實作路徑型 API 版本控制
- 將訊息佇列從 Microsoft Azure Service Bus 遷移至 Amazon SQS
- 將 .NET 應用程式從 Microsoft Azure App Service 遷移至 AWS Elastic Beanstalk
- 使用二進位方法將內部部署 Go Web 應用程式遷移至 AWS Elastic Beanstalk
- AWS 使用 將內部部署 SFTP 伺服器遷移至 AWS Transfer for SFTP
- 從 IBM WebSphere Application Server 遷移至 Amazon EC2 上的 Apache Tomcat
- 使用 Auto Scaling 從 IBM WebSphere Application Server 遷移至 Amazon EC2 上的 Apache Tomcat
- 使用 AWS App2Container 將內部部署 Java 應用程式遷移至 AWS
- 使用 ACM 將 Windows SSL 憑證遷移至 Application Load Balancer

- 在 AWS 上現代化 ASP.NET Web Forms 應用程式
- 使用 CloudWatch Logs Insights 監控應用程式活動
- 在 Amazon EC2 Linux 執行個體上執行 ASP.NET Core Web API Docker 容器
- 將自訂屬性傳送至 Amazon Cognito，並將其插入字符
- 使用 Amazon CloudFront 在 Amazon S3 儲存貯體中透過 VPC 提供靜態內容
- 在 AWS 上設定高度可用的 PeopleSoft 架構
- 使用自動化工作流程簡化 Amazon Lex 機器人開發和部署
- AWS Step Functions 使用 Amazon Bedrock 對 中的狀態進行故障診斷
- 使用 Network Firewall 從傳出流量的伺服器名稱指示擷取 DNS 網域名稱
- 使用 Flask 和 AWS Elastic Beanstalk 視覺化 AI/ML 模型結果

IoT

主題

- 在資料湖中擷取和查詢 AWS IoT SiteWise 中繼資料屬性
- 使用用戶端裝置設定 AWS IoT Greengrass 並進行疑難排解
- 更多模式

在資料湖中擷取和查詢 AWS IoT SiteWise 中繼資料屬性

Ambarish Dongaonkar，Amazon Web Services

總結

AWS IoT SiteWise 使用資產模型和階層來代表工業設備、程序和設施。每個模型或資產可以有多個專屬於您環境的屬性。中繼資料屬性範例包括資產的網站或實體位置、工廠詳細資訊和設備識別符。這些屬性值補充資產測量資料，以最大化商業價值。機器學習 (ML) 可以提供此中繼資料的其他洞見，並簡化工程任務。

不過，中繼資料屬性無法直接從 AWS IoT SiteWise 服務查詢。若要查詢屬性，您必須擷取屬性並將其擷取至資料湖。此模式使用 Python 指令碼擷取所有 AWS IoT SiteWise 資產的屬性，並將其擷取到 Amazon Simple Storage Service (Amazon S3) 儲存貯體中的資料湖。完成此程序後，您可以使用 Amazon Athena 中的 SQL 查詢來存取 AWS IoT SiteWise 中繼資料屬性和其他資料集，例如測量資料集。中繼資料屬性資訊在使用 AWS IoT SiteWise 監視器或儀表板時也很有用。您也可以使用 Amazon S3 儲存貯體中擷取的屬性來建置 Amazon Quick Sight 儀表板。 Amazon S3

模式具有參考程式碼，您可以使用 AWS Lambda 或 等使用案例的最佳運算服務來實作程式碼 AWS Glue。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 設定 AWS Lambda 函數或 AWS Glue 任務的許可。
- Amazon S3 儲存貯體。
- 資產模型和階層是在 中設定 AWS IoT SiteWise。如需詳細資訊，請參閱 AWS IoT SiteWise 文件中的建立資產模型。

Architecture

您可以使用 Lambda 函數或 AWS Glue 任務來完成此程序。如果您有少於 100 個模型，且每個模型平均有 15 個或更少的屬性，建議您使用 Lambda。對於所有其他使用案例，建議使用 AWS Glue。

下圖顯示解決方案架構和工作流程。

1. 排程 AWS Glue 任務或 Lambda 函數會執行。它會從 擷取資產中繼資料屬性 AWS IoT SiteWise ，並將其擷取至 Amazon S3 儲存貯體。
2. AWS Glue 爬蟲程式會爬取 Amazon S3 儲存貯體中擷取的資料，並在 中建立資料表 AWS Glue Data Catalog。
3. Amazon Athena 使用標準 SQL 查詢 中的資料表 AWS Glue Data Catalog。

自動化和擴展

您可以根據 AWS IoT SiteWise 資產組態的更新頻率，將 Lambda 函數或 AWS Glue 任務排程為每天或每週執行。

範例程式碼可處理的 AWS IoT SiteWise 資產數目沒有限制，但大量資產可能會增加完成程序所需的時間。

工具

- Amazon Athena 是一種互動式查詢服務，可協助您使用標準 SQL 直接在 Amazon S3 中分析資料。
- AWS Glue 是全受管擷取、轉換和載入 (ETL) 服務。它可協助您可靠地分類、清理、擴充和移動資料存放區和資料串流之間的資料。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS IoT SiteWise 可協助您大規模收集、建模、分析和視覺化工業設備的資料。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- 適用於 Python (Boto3) 的 AWS SDK 是一種軟體開發套件，可協助您整合 Python 應用程式、程式庫或指令碼 AWS 服務。

史詩

設定任務或函數

任務	Description	所需的技能
在 IAM 中設定許可。	在 IAM 主控台中，將許可授予 Lambda 函數或 AWS Glue 任務擔任的 IAM 角色，以執行下列動作： • 從 AWS IoT SiteWise 服務讀取 • 寫入 Amazon S3 儲存貯體 如需詳細資訊，請參閱《IAM 文件》中的為 建立角色 AWS 服務。	一般 AWS
建立 Lambda 函數或 AWS Glue 任務。	如果您使用的是 Lambda，請建立新的 Lambda 函數。針對執行時間，選擇 Python。如需詳細資訊，請參閱 Lambda 文件中的使用 Python 建置 Lambda 函數。 如果您使用的是 AWS Glue，請在 AWS Glue 主控台中建立新的 Python shell 任務。如需詳細資訊，請參閱 AWS Glue 文件中的新增 Python shell 任務。	一般 AWS
更新 Lambda 函數或 AWS Glue 任務。	修改新的 Lambda 函數或 AWS Glue 任務，然後在其他資訊區段中輸入程式碼範例。視需要為您的使用案例修改程式碼。如需詳細資訊，請參閱	一般 AWS

任務	Description	所需的技能
	Lambda 文件中的使用主控台編輯器編輯程式碼，以及文件中的使用指令碼AWS Glue 。	

執行任務或函數

任務	Description	所需的技能
執行 Lambda 函數或 AWS Glue 任務。	執行 Lambda 函數或 AWS Glue 任務。如需詳細資訊，請參閱 Lambda 文件中的叫用 Lambda 函數，或參閱 AWS Glue 文件中的使用觸發程序啟動任務。這會擷取 AWS IoT SiteWise 階層中資產和模型的中繼資料屬性，並將其存放在指定的 Amazon S3 儲存貯體中。	一般 AWS
設定 AWS Glue 爬蟲程式。	使用 CSV 格式檔案的必要格式分類器來設定 AWS Glue 爬蟲程式。使用 Lambda 函數或 AWS Glue 任務中使用的 Amazon S3 儲存貯體和字首詳細資訊。如需詳細資訊，請參閱 AWS Glue 文件中的定義爬蟲程式。	一般 AWS
執行 AWS Glue 爬蟲程式。	執行爬蟲程式來處理 Lambda 函數或 AWS Glue 任務建立的資料檔案。爬蟲程式會在指定的 中建立資料表 AWS Glue Data Catalog。如需詳細資訊，請參閱 或 AWS Glue 文件	一般 AWS

任務	Description	所需的技能
	中的使用觸發程序啟動爬蟲程式。	
查詢中繼資料屬性。	使用 Amazon Athena，根據您的使用案例 AWS Glue Data Catalog 使用標準 SQL 來查詢 。您可以將中繼資料屬性資料表與其他資料庫和資料表聯結。如需詳細資訊，請參閱 Amazon Athena 文件中的入門。	一般 AWS

相關資源

- Amazon Athena 文件
- AWS Glue 文件
- AWS IoT SiteWise API 參考
- AWS IoT SiteWise 使用者指南
  - 開始使用
  - 建立工業資產的模型
  - 定義資產模型之間的關係（階層）
  - 關聯和取消關聯資產
  - 建立 AWS IoT SiteWise 示範
- IOTSiteWise（適用於 Python 的 SDK 文件）
- Lambda 文件

其他資訊

Code

提供的範例程式碼僅供參考，您可以視需要針對使用案例自訂此程式碼。

```
# Following code can be used in an AWS Lambda function or in an AWS Glue Python shell
 job.
# IAM roles used for this job need read access to the AWS IoT SiteWise service and
 write access to the S3 bucket.
sw_client = boto3.client('iotsitewise')
s3_client = boto3.client('s3')
output = io.StringIO()

attribute_list=[]
bucket = '{3_bucket name}'
prefix = '{s3_bucket prefix}'
output.write("model_id,model_name,asset_id,asset_name,attribuet_id,attribute_name,attribute_val
\n")

m_resp = sw_client.list_asset_models()
for m_rec in m_resp['assetModelSummaries']:
     model_id = m_rec['id']
     model_name = m_rec['name']

     attribute_list.clear()
     dam_response = sw_client.describe_asset_model(assetModelId=model_id)
     for rec in dam_response['assetModelProperties']:
         if 'attribute' in rec['type']:
            attribute_list.append(rec['name'])

     response = sw_client.list_assets(assetModelId=model_id, filter='ALL')
     for asset in response['assetSummaries']:
         asset_id = asset['id']
         asset_name = asset['name']
         resp = sw_client.describe_asset(assetId=asset_id)
         for rec in resp['assetProperties']:
            if rec['name'] in attribute_list:
                p_resp = sw_client.get_asset_property_value(assetId=asset_id,
 propertyId=rec['id'])
                if 'propertyValue' in p_resp:
                    if p_resp['propertyValue']['value']:
                        if 'stringValue' in p_resp['propertyValue']['value']:
                             output.write(model_id + "," + model_name + ","
 + asset_id + "," + asset_name + "," + rec['id'] + "," + rec['name'] + "," +
 str(p_resp['propertyValue']['value']['stringValue']) + "\n")

                        if 'doubleValue' in p_resp['propertyValue']['value']:
```

```
                            output.write(model_id + "," + model_name + ","
 + asset_id + "," + asset_name + "," + rec['id'] + "," + rec['name'] + "," +
 str(p_resp['propertyValue']['value']['doubleValue']) + "\n")
                        if 'integerValue' in p_resp['propertyValue']['value']:
                            output.write(model_id + "," + model_name + ","
 + asset_id + "," + asset_name + "," + rec['id'] + "," + rec['name'] + "," +
 str(p_resp['propertyValue']['value']['integerValue']) + "\n")
                         if 'booleanValue' in p_resp['propertyValue']['value']:
                            output.write(model_id + "," + model_name + ","
 + asset_id + "," + asset_name + "," + rec['id'] + "," + rec['name'] + "," +
 str(p_resp['propertyValue']['value']['booleanValue']) + "\n")

output.seek(0)
s3_client.put_object(Bucket=bucket, Key= prefix + '/data.csv', Body=output.getvalue())
output.close()
```

使用用戶端裝置設定 AWS IoT Greengrass 並進行疑難排解

Marouane Sefiani 和 Akalanka De Silva，Amazon Web Services

總結

AWS IoT Greengrass 是一種開放原始碼節點執行期和雲端服務，可在邊緣裝置上建置、部署和管理物聯網 (IoT) 軟體。AWS IoT Greengrass 的使用案例包括：

- 使用 AWS IoT Greengrass 閘道做為家庭自動化中樞的智慧家庭
- AWS IoT Greengrass 可促進從生產區擷取和本機處理資料的智慧工廠

AWS IoT Greengrass 可以做為其他邊緣裝置 （ 也稱為用戶端裝置) 的安全、已驗證 MQTT 連線端點，否則通常會直接連線至 AWS IoT Core。當用戶端裝置無法直接存取 AWS IoT Core 端點時，此功能非常有用。

您可以針對下列使用案例設定 AWS IoT Greengrass 以搭配用戶端裝置使用：

- 讓用戶端裝置將資料傳送至 AWS IoT Greengrass
- 讓 AWS IoT Greengrass 將資料轉送至 AWS IoT Core
- 利用進階 AWS IoT Core 規則引擎功能

這些功能需要在 AWS IoT Greengrass 裝置上安裝和設定下列元件：

- MQTT 代理程式
- MQTT 橋接器
- 用戶端裝置身分驗證
- IP 偵測器

此外，來自用戶端裝置的已發佈訊息必須是 JSON 格式或通訊協定緩衝區 (protobuf) 格式。

此模式說明如何安裝和設定這些必要的元件，並提供疑難排解秘訣和最佳實務。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶

- AWS 命令列界面 (AWS CLI) 第 2 版
- 執行 Python 3.7 或更新版本的兩個用戶端裝置
- 執行 Java 執行期環境 (JRE) 第 8 版或更新版本的單一核心裝置，以及 Amazon Corretto 11 或 OpenJDK 11

限制

- 您必須選擇可使用 AWS IoT Core 的 AWS 區域。如需 AWS IoT Core 區域的最新清單，請參閱依區域的 AWS 服務。
- 核心裝置必須至少有 172 MB RAM 和 512 MB 的磁碟空間。

Architecture

下圖顯示此模式的解決方案架構。

架構包括：

- 兩個用戶端裝置。每個裝置都包含私有金鑰、裝置憑證和根憑證授權單位 (CA) 憑證。包含 MQTT 用戶端的 AWS IoT 裝置 SDK 也會安裝在每個用戶端裝置上。
- 使用下列元件部署 AWS IoT Greengrass 的核心裝置：
  - MQTT 代理程式
  - MQTT 橋接器
  - 用戶端裝置身分驗證
  - IP 偵測器

此架構支援下列案例：

- 用戶端裝置可以使用其 MQTT 用戶端，透過核心裝置的 MQTT 代理程式彼此通訊。
- 用戶端裝置也可以透過核心裝置的 MQTT 代理程式和 MQTT 橋接器，與雲端中的 AWS IoT Core 通訊。
- 雲端中的 AWS IoT Core 可以透過 MQTT 測試用戶端和核心裝置的 MQTT 橋接器和 MQTT 代理程式，將訊息傳送至用戶端裝置。

如需用戶端裝置與核心裝置之間通訊的詳細資訊，請參閱其他資訊一節。

工具

AWS 服務

- AWS IoT Greengrass 是開放原始碼物聯網 (IoT) 邊緣執行期和雲端服務，可協助您在裝置上建置、部署和管理 IoT 應用程式。
- AWS IoT Core 為連線網際網路的裝置提供安全的雙向通訊，以連線至 AWS 雲端。
- AWS IoT Device SDK 是一種軟體開發套件，其中包含開放原始碼程式庫、包含範例的開發人員指南，以及移植指南，讓您可以在所選的硬體平台上建置創新的 IoT 產品或解決方案。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。

最佳實務

- 來自用戶端裝置的訊息承載應該採用 JSON 或 Protobuf 格式，以便利用 AWS IoT Core 規則引擎的進階功能，例如轉換和條件式動作。
- 設定 MQTT 橋接器以允許雙向通訊。
- 在 AWS IoT Greengrass 中設定和部署 IP 偵測器元件，以確保核心裝置的 IP 地址包含在 MQTT 代理程式憑證的主體別名 (SAN) 欄位中。

史詩

設定核心裝置

任務	Description	所需的技能
在核心裝置上設定 AWS IoT Greengrass。	請依照開發人員指南中的指示安裝 AWS IoT Greengrass Core 軟體。	AWS IoT Greengrass
檢查安裝的狀態。	使用下列命令來檢查核心裝置上的 AWS IoT Greengrass 服務狀態： `sudo systemctl status greengrass.service`	一般 AWS

任務	Description	所需的技能
	命令的預期輸出為： `Launched Nucleus successfully`	

任務	Description	所需的技能
設定 IAM 政策並將其連接到 Greengrass 服務角色。	1. 建立 IAM 政策，以允許往返 MQTT 橋接器的通訊。以下是範例政策： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "iot:*" ], "Resource": "*" }, { "Sid": "GreengrassActions", "Effect": "Allow", "Action": [ "greengrass:*" ], "Resource": "*" } ] }` 2. 將政策連接至 Greengrass 服務角色。若要取得服務角色，請使用 命令：	一般 AWS

任務	Description	所需的技能
	`aws greengrassv2 get-service-role-for-account --region <region>` 其中 `<region>`是指您的 AWS 區域。	
在 AWS IoT Greengrass 核心裝置中設定和部署必要的元件。	設定和部署下列元件： • `greengrass.clientdevices.mqtt.Moquette`（請參閱組態詳細資訊） • `greengrass.clientdevices.mqtt.Bridge`（請參閱組態詳細資訊和下一個任務） • `greengrass.clientdevices.Auth`（請參閱組態詳細資訊和下一個組態詳細資訊之後的任務） • `aws.greengrass.clientdevices.IPDetector`（請參閱組態詳細資訊）	AWS IoT Greengrass

任務	Description	所需的技能
確認 MQTT 橋接器允許雙向通訊。	若要在用戶端裝置和 AWS IoT Core 之間轉送 MQTT 訊息，請設定和部署 MQTT 橋接器元件，並指定要轉送的主題。範例如下： `{ "mqttTopicMapping": { "ClientDevicesToCloud": { "topic": "dt/#", "source": "LocalMqtt", "target": "IotCore" }, "CloudToClientDevices": { "topic": "cmd/#", "source": "IotCore", "target": "LocalMqtt" } } }`	AWS IoT Greengrass

任務	Description	所需的技能
確認身分驗證元件允許用戶端裝置連接和發佈或訂閱主題。	下列`aws.greengrass.clientdevices.Auth` 組態允許所有用戶端裝置連線、發佈訊息和訂閱所有主題。 `{ "deviceGroups": { "formatVersion": "2021-03-05", "definitions": { "MyPermissiveDeviceGroup": { "selectionRule": "thingName: *", "policyName": "MyPermissivePolicy" } }, "policies": { "MyPermissivePolicy": { "AllowAll": { "statementDescription": "Allow client devices to perform all actions.", "operations": ["*"], "resources": ["*"] } } } } }`	AWS IoT Greengrass

任務	Description	所需的技能
	`}`	

設定用戶端裝置

任務	Description	所需的技能

安裝 AWS IoT 裝置 SDK。	在用戶端裝置上安裝 AWS IoT 裝置 SDK。如需支援的語言和相關 SDKs 的完整清單，請參閱 AWS IoT Core 文件。
	
	例如，適用於 Python 的 AWS IoT 裝置 SDK 位於 GitHub。若要安裝此 SDK：
	
	1. 確認已安裝 Python 3.7 或更新版本，如 GitHub 儲存庫的先決條件頁面所述。
	2. 使用 pip 命令來安裝 SDK。
	
	對於 MacOS 和 Linux：
	
	`python3 -m pip install awsiotsdk`
	
	針對 Windows：
	
	`python -m pip install awsiotsdk`
	
	或者，您可以從來源儲存庫安裝 SDK：
		一般 AWS IoT

任務	Description	所需的技能

	`# Create a workspace directory to hold all the SDK files`
	`mkdir sdk-workspace`
	`cd sdk-workspace`
	`# Clone the repository`
	`git clone https://github.com/aws/aws-iot-device-sdk-python-v2.git`
	`# Install using Pip (use 'python' instead of 'python3' on Windows)`
	`python3 -m pip install ./aws-iot-device-sdk-python-v2`	

任務	Description	所需的技能
建立物件。	1. 在 AWS IoT 主控台中，如果出現入門按鈕，請選擇它。否則，在導覽窗格中，選擇安全性、政策。 2. 如果尚未顯示任何政策對話方塊，請選擇建立政策。否則，請選擇 Create (建立)。 3. 輸入 AWS IoT 政策的名稱（例如 `ClientDevicePolicy`）。 4. 在新增陳述式區段中，使用下列 JSON 程式碼取代現有的政策。將 `<region>` 和 `<account>` 取代為您的 AWS 區域和 AWS 帳戶號碼。 `{ "Version": "2012-10-17", "Statement": [{ "Effect": "Allow", "Action": "iot:Connect", "Resource": "arn:aws:iot:region:account:client/*" }, { "Effect": "Allow", "Action": "iot:Publish", "Resource": "*" }, ...`	AWS IoT Core

任務	Description	所需的技能

	<pre>{
	"Effect": "Allow",
	"Action": "iot:Receive",
	"Resource": "*"
	},
	{
	"Effect": "Allow",
	"Action": "iot:Subscribe",
	"Resource": "*"
	},
	{
	"Effect": "Allow",
	"Action": [
	"iot:GetThingShadow",
	"iot:UpdateThingShadow",
	"iot:DeleteThingShadow"
	],
	"Resource": "arn:aws:iot:region:account:thing/*"
	
	}
	]
	}</pre>
	5. 選擇建立。
	6. 在 AWS IoT 主控台的導覽窗格中，選擇管理、事物。
	7. 如果顯示您還沒有任何物件對話方塊，請選擇註冊物	

任務	Description	所需的技能
	件。否則，請選擇 Create (建立)。 8. 在 Creating AWS IoT things (建立 AWS IoT 物件) 頁面上，選擇 Create a single thing (建立單一物件)。 9. 在 Add your device to the device registry (將裝置新增至裝置登錄檔) 頁面上，輸入您 IoT 物件的名稱 (例如 `ClientDevice1`)，然後選擇 Next (下一步)。 **Note** 您無法在建立之後變更物件的名稱。若要變更名稱，您必須建立新的物件、提供新名稱，然後刪除舊物件。 10. 在 Add a certificate for your thing (新增物件的憑證) 頁面上，選擇 Create certificate (建立憑證)。 11. 選擇 Download (下載) 連結來下載憑證、私有金鑰和根憑證授權機構憑證。 **Important** 這是您下載憑證和私有金鑰的唯一機會。	

任務	Description	所需的技能
	12.若要啟用憑證，請選擇 Activate (啟用)。憑證必須處於作用中狀態，裝置才能連線至 AWS IoT。 13.選擇 Attach a policy (連接政策)。 14.針對為物件新增政策，選擇 ClientDevicePolicy、註冊物件。	
從 Greengrass 核心裝置下載 CA 憑證。	如果您預期 Greengrass 核心裝置可在離線環境中運作，您必須將 Greengrass 核心 CA 憑證提供給用戶端裝置，才能驗證 MQTT 代理程式的憑證（由 Greengrass 核心 CA 發行）。因此，請務必取得此憑證的副本。使用下列其中一種方法來下載 CA 憑證： • 如果您可以從 PC 存取 AWS IoT Greengrass 裝置的網路，請在 Web 瀏覽器`https://<device IP>:8883`中輸入 ，並檢視 MQTT 代理程式憑證和 CA 憑證。您也可以將 CA 憑證儲存到用戶端裝置。 • 或者，您可以使用 OpenSSL 命令列： `openssl s_client -showcerts -connect <device IP>:8883`	一般 AWS

任務	Description	所需的技能
在用戶端裝置中複製登入資料。	在用戶端裝置中複製 Greengrass 核心 CA 憑證、裝置憑證和私有金鑰。	一般 AWS
將用戶端裝置與核心裝置建立關聯。	將用戶端裝置與核心裝置建立關聯，以便他們可以探索核心裝置。然後，用戶端裝置可以使用 Greengrass 探索 API 來擷取其相關聯核心裝置的連線資訊和憑證。如需詳細資訊，請參閱 AWS IoT Greengrass 文件中的關聯用戶端裝置。 1. 在 AWS IoT Greengrass 主控台上，選擇 Core 裝置。 2. 選擇要管理的核心裝置。 3. 在核心裝置的詳細資訊頁面上，選擇用戶端裝置索引標籤。 4. 在關聯的用戶端裝置區段中，選擇關聯用戶端裝置。 5. 在將用戶端裝置與核心裝置模式建立關聯中，為每個要建立關聯的用戶端裝置執行下列動作： a. 輸入要關聯為用戶端裝置的 AWS IoT 物件名稱。 b. 選擇新增。 6. 選擇關聯。 您相關聯的用戶端裝置現在可以使用 Greengrass 探索 API 來探索此核心裝置。	AWS IoT Greengrass

傳送和接收資料

任務	Description	所需的技能
將資料從一個用戶端裝置傳送至另一個用戶端裝置。	使用裝置中的 MQTT 用戶端來發佈`dt/client1/sensor`主題的訊息。	一般 AWS
將資料從用戶端裝置傳送至 AWS IoT Core。	使用裝置中的 MQTT 用戶端來發佈`dt/client1/sensor`主題的訊息。 在 MQTT 測試用戶端中，訂閱裝置傳送訊息的主題，或為所有主題訂閱 `#`（請參閱詳細資訊）。	一般 AWS
從 AWS IoT Core 傳送訊息至用戶端裝置。	在 MQTT 測試用戶端頁面上，在發佈至主題索引標籤的主題名稱欄位中，輸入訊息的主題名稱。在此範例中，針對 主題使用 `cmd/client1` 。	一般 AWS

疑難排解

問題	解決方案
無法驗證伺服器憑證錯誤	當 MQTT 用戶端無法驗證 MQTT 代理程式在 TLS 交握期間提供的憑證時，會發生此錯誤。最常見的原因是 MQTT 用戶端沒有 CA 憑證。請依照下列步驟，確認 CA 憑證已提供給 MQTT 用戶端。 1. 如果您可以從 PC 存取 AWS IoT Greengrass 裝置的網路，請在瀏覽器視窗`https://<device IP>:8883` 中輸入 以檢視 MQTT

問題	解決方案

	代理程式憑證和 CA 憑證。您也可以將 CA 憑證儲存到用戶端裝置。
	
	或者，使用 OpenSSL 命令列：
	
	`openssl s_client -showcerts -connect <device IP>:8883`
	
	2. 將 Moquette CA 和 Greengrass Core CA 憑證的內容儲存到檔案中，然後使用 命令檢視解碼的內容：
	
	`openssl x509 -in <Name of CA>.pem -text`
	
	Moquette CA 憑證應會顯示 SAN 欄位，如本範例所示：
	
	`X509v3 Subject Alternative Name: IP Address:XXX.XXX.XXX.XXX, IP Address:127.0.0.1, DNS:localhost`

無法驗證伺服器名稱錯誤	當 MQTT 用戶端無法驗證其是否連線至正確的伺服器時，就會發生此錯誤。最常見的原因是 Greengrass 裝置的 IP 地址未列在憑證的 SAN 欄位中。 依照先前解決方案中的指示取得 MQTT 代理程式憑證，並確認 SAN 欄位包含 AWS IoT Greengrass 裝置的 IP 地址，如其他資訊一節所述。如果沒有，請確認 IP 偵測器元件已正確安裝，然後重新啟動核心裝置。

問題	解決方案
只有在從內嵌用戶端裝置連線時，才能驗證伺服器名稱	Mbed TLS 是內嵌裝置中使用的熱門 TLS 程式庫，目前僅支援憑證的 SAN 欄位中的 DNS 名稱驗證，如 Mbed TLS 程式庫程式碼所示。由於核心裝置沒有自己的網域名稱，且取決於 IP 地址，因此使用 Mbed TLS 的 TLS 用戶端會在 TLS 交握期間失敗伺服器名稱驗證，導致連線失敗。建議您在 x509_crt_check_san 函數將 SAN IP 地址驗證新增至 Mbed TLS 程式庫。

相關資源

- AWS IoT Greengrass 文件
- AWS IoT Core 文件
- MQTT 代理程式元件
- MQTT 橋接器元件
- 用戶端裝置身分驗證元件
- IP 偵測器元件
- AWS IoT 裝置 SDK
- 使用 AWS IoT Greengrass 實作本機用戶端裝置 (AWS 部落格文章)
- RFC 5280 – 網路 X.509 公有金鑰基礎設施憑證和憑證撤銷清單 (CRL) 設定檔

其他資訊

本節提供有關用戶端裝置與核心裝置之間通訊的其他資訊。

MQTT 代理程式會在核心裝置中的連接埠 8883 上接聽 TLS 用戶端連線嘗試。下圖顯示 MQTT 代理程式的伺服器憑證範例。

憑證範例會顯示下列詳細資訊：

- 憑證由 AWS IoT Greengrass Core CA 發行，該 CA 是本機且專屬於核心裝置；也就是說，它充當本機 CA。

- 用戶端身分驗證元件每週會自動輪換此憑證，如下圖所示。您可以在用戶端身分驗證元件組態中設定此間隔。

- 主體替代名稱 (SAN) 在 TLS 用戶端的伺服器名稱驗證中扮演重要角色。它有助於 TLS 用戶端確保連接到正確的伺服器，並有助於避免 man-in-the-middle攻擊。在範例憑證中，SAN 欄位表示此伺服器正在接聽 localhost （本機 Unix 網域通訊端），且網路介面具有 IP 地址 192.168.1.12。

TLS 用戶端使用憑證中的 SAN 欄位來驗證它在伺服器驗證期間是否連線到合法伺服器。相反地，在 HTTP 伺服器和瀏覽器之間的典型 TLS 交握期間，常用名稱 (CN) 欄位或 SAN 欄位中的網域名稱用於交叉檢查瀏覽器在伺服器驗證程序期間實際連接的網域。如果核心裝置沒有網域名稱，SAN 欄位中包含的 IP 地址具有相同的用途。如需詳細資訊，請參閱 RFC 5280 – 網路 X.509 公有金鑰基礎設施憑證和憑證撤銷清單 (CRL) 設定檔的主體別名一節。

AWS IoT Greengrass 中的該 IP 偵測器元件可確保正確的 IP 地址包含在憑證的 SAN 欄位中。

範例中的憑證由做為本機 CA 的 AWS IoT Greengrass 裝置簽署。TLS 用戶端 (MQTT 用戶端） 不知道此 CA，因此我們必須提供如下所示的 CA 憑證。

更多模式

- 使用 AWS IoT Greengrass，以經濟實惠的方式直接將 IoT 資料擷取至 Amazon S3 AWS IoT
- 使用 Terraform 在 Amazon Bedrock 上部署代理程式系統，搭配CrewAI 架構
- 在做為 Docker 容器 AWS IoT Greengrass V2 執行時部署容器化應用程式

遷移與現代化

主題

- 遷移
- 現代化
- 大型主機

遷移

主題

- 使用 Microsoft Excel 和 Python 為 AWS DMS 任務建立 AWS CloudFormation 範本
- 開始使用自動化產品組合探索
- 將內部部署 Cloudera 工作負載遷移至 AWS 上的 Cloudera 資料平台
- 解決將 Microsoft SQL Server 遷移至 AWS 雲端後的連線錯誤
- 重新啟動 RHEL 來源伺服器後，在不停用 SELinux 的情況下自動重新啟動 AWS 複寫代理程式
- 重新架構師
- 重新託管
- 重新定位
- 平台重建
- 依工作負載的遷移模式
- 更多模式

使用 Microsoft Excel 和 Python 為 AWS DMS 任務建立 AWS CloudFormation 範本

Venkata Naveen Koppula，Amazon Web Services

總結

此模式概述使用 Microsoft Excel 和 Python 為 AWS Database Migration Service (AWS DMS) 自動建立 AWS CloudFormation 範本的步驟。

使用 AWS DMS 遷移資料庫通常需要建立 AWS CloudFormation 範本來佈建 AWS DMS 任務。先前，建立 AWS CloudFormation 範本需要 JSON 或 YAML 程式設計語言的知識。使用此工具，您只需具備 Excel 的基本知識，以及如何使用終端機或命令視窗執行 Python 指令碼。

做為輸入，工具會採用 Excel 工作手冊，其中包含要遷移的資料表名稱、AWS DMS 端點的 Amazon Resource Name (ARNs)，以及 AWS DMS 複寫執行個體。然後，該工具會為所需的 AWS DMS 任務產生 AWS CloudFormation 範本。

如需詳細步驟和背景資訊，請參閱 AWS 資料庫部落格中的部落格文章使用 Microsoft Excel 建立 AWS DMS 任務的 AWS CloudFormation 範本。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Microsoft Excel 2016 版或更新版本
- Python 2.7 版或更新版本
- xlrd Python 模組 （ 安裝在命令提示中，使用 命令：pip install xlrd)
- AWS DMS 來源和目標端點和 AWS DMS 複寫執行個體

限制

- 結構描述、資料表和關聯資料欄的名稱會在目的地端點轉換為小寫字元。
- 此工具不會處理 AWS DMS 端點和複寫執行個體的建立。
- 目前， 工具只支援每個 AWS DMS 任務的一個結構描述。

Architecture

來源技術堆疊

- 內部部署資料庫
- Microsoft Excel

目標技術堆疊

- AWS CloudFormation 範本
- AWS 雲端中的資料庫

架構

工具

- Pycharm IDE，或任何支援 Python 3.6 版的整合式開發環境 (IDE)
- Microsoft Office 2016 （適用於 Microsoft Excel)

史詩

設定網路、AWS DMS 複寫執行個體和端點

任務	Description	所需的技能
如有必要，請請求提高服務配額。	視需要請求提高 AWS DMS 任務的服務配額。	一般 AWS
設定 AWS 區域、虛擬私有雲端 (VPCs)、CIDR 範圍、可用區域和子網路。		一般 AWS
設定 AWS DMS 複寫執行個體。	AWS DMS 複寫執行個體可以連線到內部部署和 AWS 資料庫。	一般 AWS

任務	Description	所需的技能
設定 AWS DMS 端點。	設定來源和目標資料庫的端點。	一般 AWS

準備 AWS DMS 任務和標籤的工作表

任務	Description	所需的技能
設定資料表清單。	列出遷移中涉及的所有資料表。	資料庫
準備任務工作表。	使用您設定的資料表清單準備 Excel 工作表。	General AWS、Microsoft Excel
準備標籤工作表。	詳細說明要連接到 AWS DMS 任務的 AWS 資源標籤。	General AWS、Microsoft Excel

下載並執行 工具

任務	Description	所需的技能
從 GitHub 儲存庫下載並擷取範本產生工具。	GitHub 儲存庫：https：//https://github.com/aws-samples/dms-cloudformation-templates-generator/	
執行 工具。	請遵循「參考和說明」下列出的部落格文章中的詳細指示。	

相關資源

- 使用 Microsoft Excel 為 AWS DMS 任務建立 AWS CloudFormation 範本 （部落格文章）
- DMS CloudFormation 範本產生器 (GitHub 儲存庫）
- Python 文件

- xlrd 描述和下載
- AWS DMS 文件
- AWS CloudFormation 文件

開始使用自動化產品組合探索

Pratik Chunawala 和 Rodolfo Jr. Cerrada，Amazon Web Services

總結

將應用程式和伺服器遷移至 Amazon Web Services (AWS) 雲端時，評估產品組合和收集中繼資料是一項重大挑戰，特別是對於具有 300 多個伺服器的大型遷移。使用自動化產品組合探索工具可協助您收集應用程式的相關資訊，例如使用者數量、使用頻率、相依性，以及應用程式基礎設施的相關資訊。在規劃遷移波紋時，此資訊至關重要，因此您可以正確排定優先順序，並將具有類似特徵的應用程式分組。使用探索工具可簡化產品組合團隊與應用程式擁有者之間的通訊，因為產品組合團隊可以驗證探索工具的結果，而不是手動收集中繼資料。此模式討論選擇自動化探索工具的重要考量，以及如何在環境中部署和測試工具的資訊。

此模式包含 範本，這是建立自己高階活動檢查清單的起點。檢查清單旁邊是負責、負責、諮詢、告知 (RACI) 矩陣的範本。您可以使用此 RACI 矩陣來判斷誰負責檢查清單中的每個任務。

史詩

選取探索工具

任務	Description	所需的技能
判斷探索工具是否適合您的使用案例。	探索工具可能不是適合您使用案例的最佳解決方案。考慮選取、購買、準備和部署探索工具所需的時間量。在您的環境中為無代理程式探索工具設定掃描設備，或將代理程式安裝到所有範圍內工作負載，可能需要 4-8 週的時間。部署後，您必須允許 4 到 12 週的時間，讓探索工具透過掃描應用程式工作負載和執行應用程式堆疊分析來收集中繼資料。如果您遷移的伺服器少於 100 個，您可能可以手動收集中繼資料，並比使用自動探索工具	遷移負責人，遷移工程師

任務	Description	所需的技能
	部署和收集中繼資料所需的時間更快地分析相依性。	
選取探索工具。	檢閱其他資訊區段中選取自動探索工具的考量事項。判斷為您的使用案例選擇探索工具的適當條件，然後根據這些條件評估每個工具。如需自動化探索工具的完整清單，請參閱探索、規劃和建議遷移工具。	遷移負責人，遷移工程師

準備安裝

任務	Description	所需的技能
準備部署前檢查清單。	建立您在部署工具之前必須完成的任務檢查清單。如需範例，請參閱 Flexera 文件網站上的部署前檢查清單。	組建主管、遷移工程師、遷移主管、網路管理員
準備網路需求。	佈建工具執行和存取目標伺服器所需的連接埠、通訊協定、IP 地址和路由。如需詳細資訊，請參閱 探索工具的安裝指南。如需範例，請參閱 Flexera 文件網站上的部署需求。	遷移工程師、網路管理員、雲端架構師
準備帳戶和登入資料需求。	識別存取目標伺服器和安裝所有工具元件所需的登入資料。	雲端管理員、一般 AWS、遷移工程師、遷移主管、網路管理員、AWS 管理員
準備您要安裝工具的設備。	確定您要安裝工具元件的設備符合工具的規格和平台需求。	遷移工程師、遷移主管、網路管理員
準備變更訂單。	根據您組織中的變更管理程序，準備所需的任何變更訂	組建潛在客戶、遷移潛在客戶

任務	Description	所需的技能
	單，並確保這些變更訂單獲得核准。	
將需求傳送給利益相關者。	將部署前檢查清單和網路需求傳送給利益相關者。在繼續部署之前，利益相關者應該檢閱、評估和準備必要的要求。	組建潛在客戶、遷移潛在客戶

部署 工具

任務	Description	所需的技能
下載安裝程式。	下載安裝程式或虛擬機器映像。虛擬機器映像通常採用開放虛擬化格式 (OVF)。	組建潛在客戶、遷移潛在客戶
將檔案解壓縮。	如果您使用的是 安裝程式，則必須在內部部署伺服器上下載並執行安裝程式。	組建潛在客戶、遷移潛在客戶
在伺服器上部署 工具。	在目標內部部署伺服器上部署探索工具，如下所示： • 如果您的來源檔案是虛擬機器映像，請將其部署到您的虛擬機器環境，例如 VMware。 • 如果您的來源檔案是安裝程式，請執行安裝程式來安裝和設定工具。	組建領導、遷移領導、網路管理員
登入 探索工具。	遵循畫面上的提示，並登入以開始使用 工具。	遷移潛在客戶、建置潛在客戶
啟用產品。	輸入您的授權金鑰。	組建潛在客戶、遷移潛在客戶

任務	Description	所需的技能
設定工具。	輸入存取目標伺服器所需的任何登入資料，例如 Windows、VMware、Simple Network Management Protocol (SNMP) 和 Secure Shell Protocol (SSH) 或資料庫的登入資料。	組建潛在客戶、遷移潛在客戶

測試工具

任務	Description	所需的技能
選取測試伺服器。	識別一組可用來測試探索工具的非生產子網路或 IP 地址。這可協助您快速驗證掃描、快速識別任何錯誤並進行疑難排解，以及將您的測試與生產環境隔離。	組建領導、遷移領導、網路管理員
開始掃描選取的測試伺服器。	對於無代理程式探索工具，請在探索工具主控台中輸入所選測試伺服器的子網路或 IP 地址，然後開始掃描。 對於代理程式型探索工具，請在選取的測試伺服器上安裝代理程式。	組建領導、遷移領導、網路管理員
檢閱掃描結果。	檢閱測試伺服器的掃描結果。如果發現任何錯誤，請疑難排解並修正錯誤。記錄錯誤和解決方案。您可以在未來參考此資訊，也可以將此資訊新增至您的產品組合 Runbook。	組建領導、遷移領導、網路管理員

任務	Description	所需的技能
重新掃描測試伺服器。	重新掃描完成後，請重複掃描，直到沒有錯誤為止。	組建領導、遷移領導、網路管理員

相關資源

AWS resources

- AWS 雲端 遷移的應用程式產品組合評估指南
- 探索、規劃和建議遷移工具

常用所選探索工具的部署指南

- 部署 RN150 虛擬設備 (Flexera 文件）
- 收集器安裝 (modelizeIT 文件）
- 內部部署分析伺服器安裝 (modelizeIT 文件）

其他資訊

選取自動化探索工具的考量事項

每個探索工具都有優點和限制。為您的使用案例選取適當的工具時，請考慮下列事項：

- 選取探索工具，可收集您實現產品組合評估目標所需的大部分中繼資料，如果不是全部。
- 識別您需要手動收集的任何中繼資料，因為工具不支援它。
- 將探索工具需求提供給利益相關者，讓他們可以根據其內部安全和合規需求來檢閱和評估工具，例如伺服器、網路和憑證需求。
  - 此工具是否需要您在範圍內工作負載中安裝 代理程式？
  - 工具是否需要您在環境中設定虛擬設備？
- 判斷您的資料駐留需求。有些組織不想將資料存放在環境之外。若要解決此問題，您可能需要在內部部署環境中安裝工具的一些元件。
- 確定工具支援範圍內工作負載的作業系統 (OS) 和作業系統版本。

- 判斷您的產品組合是否包含大型主機、中範圍和舊版伺服器。大多數探索工具可以將這些工作負載偵測為相依性，但有些工具可能無法取得裝置詳細資訊，例如使用率和伺服器相依性。Device42 和 modernizeIT 探索工具都支援大型主機和中階伺服器。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

將內部部署 Cloudera 工作負載遷移至 AWS 上的 Cloudera 資料平台

Battulga Purevragchaa 和 Nidhi Gupta，Amazon Web Services

Nijjwol Lamsal， Cloudera， Inc.

總結

此模式說明將內部部署 Cloudera 分散式 Hadoop (CDH)、Hortonworks 資料平台 (HDP) 和 Cloudera 資料平台 (CDP) 工作負載遷移至 CDP Public Cloud on AWS 的高階步驟。我們建議您與 Cloudera Professional Services 和系統整合商 (SI) 合作實作這些步驟。

Cloudera 客戶想要將內部部署 CDH、HDP 和 CDP 工作負載移至雲端的原因有很多。一些典型原因包括：

- 簡化新資料平台範例的採用，例如資料湖區或資料網格
- 提高業務敏捷性、普及現有資料資產的存取和推論
- 降低總擁有成本 (TCO)
- 增強工作負載彈性
- 提供更高的可擴展性；相較於舊版內部部署安裝基礎，可大幅縮短佈建資料服務的時間
- 淘汰舊版硬體；大幅減少硬體重新整理週期
- 利用pay-as-you-go定價透過 Cloudera 授權模型 (CCU) 擴展到 AWS 上的 Cloudera 工作負載
- 利用更快的部署和改善與持續整合和持續交付 (CI/CD) 平台的整合
- 針對多個工作負載使用單一統一平台 (CDP)

Cloudera 支援所有主要工作負載，包括Machine Learning、資料工程、資料倉儲、操作資料庫、串流處理 (CSP)，以及資料安全和控管。Cloudera 已在內部部署提供這些工作負載多年，您可以使用 CDP Public Cloud 搭配 Workload Manager 和 Replication Manager，將這些工作負載遷移至 AWS 雲端。

Cloudera 共用資料體驗 (SDX) 提供跨這些工作負載的共用中繼資料目錄，以促進一致的資料管理和操作。SDX 也包含可防範威脅的全方位精細安全性，以及稽核和搜尋功能的統一控管，以符合支付卡產業資料安全標準 (PCI DSS) 和 GDPR 等標準。

CDP 遷移一目了然

來源工作負載 CDH、HDP 和 CDP 私有雲端

工作負載	來源環境	• Windows，Linux • 內部部署、主機代管或任何非 AWS 環境
	目的地工作負載	AWS 上的 CDP 公有雲端
	目的地環境	• 部署模型：客戶帳戶 • 操作模型：客戶/Cloudera 控制平面
	遷移策略 (7R)	重新託管、轉換或重構
	這是工作負載版本中的升級嗎？	是
移轉	遷移持續時間	• 部署：約 1 週可建立客戶帳戶、虛擬私有雲端 (VPC) 和 CDP 公有雲端客戶受管環境。 • 遷移持續時間：1-4 個月，取決於工作負載的複雜性和大小。

成本	在 AWS 上執行工作負載的成本	• 在高層級上，CDH 工作負載遷移至 AWS 的成本假設您會在 AWS 上建立新的環境。它包括考慮人員時間和精力，以及為新環境佈建運算資源和授權軟體。 • Cloudera 雲端消費型定價模式可讓您靈活地利用爆量和自動擴展功能。如需詳細資訊，請參閱 Cloudera 網站上的 CDP 公有雲端服務費率。 • Cloudera Enterprise Data Hub 是以 Amazon Elastic Compute Cloud (Amazon EC2) 為基礎，並緊密建立傳統叢集的模型。Data Hub 可以自訂，但這會影響成本。 • CDP 公有雲端資料倉儲、Cloudera Machine Learning 和 Cloudera Data Engineering (CDE) 是以容器為基礎，可設定為自動擴展。
基礎設施協議和架構	系統要求	請參閱先決條件一節。
	SLA	請參閱 CDP 公有雲端的 Cloudera 服務水準協議。
	DR	請參閱 Cloudera 文件中的災難復原。
	授權和操作模型（適用於目標 AWS 帳戶）	使用自有授權 (BYOL) 模型
合規	安全要求	請參閱 Cloudera 文件中的 Cloudera 安全性概觀。

其他合規認證	請參閱 Cloudera 網站上有關一般資料保護法規 (GDPR) 合規和 CDP 信任中心的資訊。

先決條件和限制

先決條件

- AWS 帳戶需求，包括帳戶、資源、服務和許可，例如 AWS Identity and Access Management (IAM) 角色和政策設定
- 從 Cloudera 網站部署 CDP 的先決條件

遷移需要下列角色和專業知識：

Role	技能和責任
遷移潛在客戶	確保執行支援、團隊協作、規劃、實作和評估
Cloudera 中小企業	CDH、HDP 和 CDP 管理、系統管理和架構方面的專業技能
AWS 架構師	AWS 服務、聯網、安全和架構的技能

Architecture

建置適當的架構是確保遷移和效能符合您期望的關鍵步驟。為了滿足此手冊的假設，AWS 雲端中的目標資料環境，無論是在虛擬私有雲端 (VPC) 託管執行個體或 CDP 上，在作業系統和軟體版本以及主要機器規格方面，都必須與來源環境相當。

下圖（透過 Cloudera 共享資料體驗資料表的許可而重新產生）顯示 CDP 環境的基礎設施元件，以及層或基礎設施元件如何互動。

架構包含下列 CDP 元件：

- Data Hub 是一項服務，用於啟動和管理採用 Cloudera Runtime 技術的工作負載叢集。您可以使用 Data Hub 中的叢集定義，為自訂使用案例佈建和存取工作負載叢集，並定義自訂叢集組態。如需詳細資訊，請參閱 Cloudera 網站。
- 資料流程和串流可解決企業在處理資料時面臨的主要挑戰。它會管理下列項目：
  - 處理大量和大規模的即時資料串流
  - 追蹤串流資料的資料來源和歷程
  - 管理和監控邊緣應用程式和串流來源

  如需詳細資訊，請參閱 Cloudera 網站上的 Cloudera DataFlow 和 CSP。
- 資料工程包括資料整合、資料品質和資料控管，可協助組織建置和維護資料管道和工作流程。如需詳細資訊，請參閱 Cloudera 網站。了解 Spot 執行個體的支援，以協助節省 AWS for Cloudera Data Engineering 工作負載的成本。
- 資料倉儲 可讓您建立獨立的資料倉儲和資料封送，以自動擴展以滿足工作負載需求。此服務為每個資料倉儲和資料智慧提供隔離的運算執行個體和自動化最佳化，並協助您在符合 SLAs 時節省成本。如需詳細資訊，請參閱 Cloudera 網站。了解如何管理 AWS 上 Cloudera Data Warehouse 的成本和自動擴展。
- CDP 中的操作資料庫為可擴展、高效能應用程式提供了可靠且靈活的基礎。它提供即時、始終可用、可擴展的資料庫，可在統一的操作和倉儲平台上提供傳統的結構化資料以及新的非結構化資料。如需詳細資訊，請參閱 Cloudera 網站。
- Machine Learning 是一種雲端原生機器學習平台，可將自助式資料科學和資料工程功能合併到企業資料雲端內的單一可攜式服務。它可在資料上的任何地方進行可擴展的機器學習和人工智慧 (AI) 部署。如需詳細資訊，請參閱 Cloudera 網站。

AWS 上的 CDP

下圖（採用 Cloudera 網站的許可）顯示 AWS 上 CDP 的高階架構。CDP 實作自己的安全模型來管理帳戶和資料流程。這些透過使用跨帳戶角色與 IAM 整合。

CDP 控制平面位於 Cloudera 主帳戶中自己的 VPC。每個客戶帳戶都有自己的子帳戶和唯一的 VPC。跨帳戶 IAM 角色和 SSL 技術會將進出控制平面的管理流量路由到位於每個客戶 VPC 內網際網路可路由公有子網路上的客戶服務。在客戶的 VPC 上，Cloudera 共享資料體驗 (SDX) 以統一的控管和合規提供企業級安全性，讓您可以更快地從資料中取得洞見。SDX 是併入所有 Cloudera 產品的設計理念。如需適用於 AWS 的 SDX 和 CDP 公有雲端網路架構的詳細資訊，請參閱 Cloudera 文件。 https://docs.cloudera.com/cdp-public-cloud/cloud/aws-refarch/topics/cdp-pc-aws-refarch-overview.html

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon Elastic Kubernetes Service (Amazon EKS) 可協助您在 AWS 上執行 Kubernetes，而無需安裝或維護您自己的 Kubernetes 控制平面或節點。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

自動化和工具

- 如需其他工具，您可以使用 Cloudera Backup Data Recovery (BDR)、AWS Snowball 和 AWS Snowmobile，協助將資料從內部部署 CDH、HDP 和 CDP 遷移到 AWS 託管的 CDP。
- 對於新的部署，我們建議您使用適用於 CDP 的 AWS 合作夥伴解決方案。

史詩

準備遷移

任務	Description	所需的技能
與 Cloudera 團隊互動。	Cloudera 會與客戶一起追求標準化的參與模式，並可與您的系統整合商 (SI) 合作，以推廣相同的方法。請聯絡 Cloudera 客戶團隊，讓他們可以提供指引和必要的技術資源，以開始專案。聯絡 Cloudera 團隊可確保所有必要的團隊都能在日期接近時準備遷移。	遷移潛在客戶

任務	Description	所需的技能
	您可以聯絡 Cloudera Professional Services，以較低成本和最高效能快速地將 Cloudera 部署從試行移至生產環境。如需方案的完整清單，請參閱 Cloudera 網站。	
在 AWS 上為您的 VPC 建立 CDP 公有雲端環境。	使用 Cloudera Professional Services 或您的 SI，在 AWS 上的 VPC 中規劃和部署 CDP 公有雲端。	Cloudera SME 雲端架構師
排定優先順序並評估工作負載以進行遷移。	評估所有現場部署工作負載，以判斷最容易遷移的工作負載。非關鍵任務的應用程式最好先移動，因為它們對您的客戶的影響最小。在您成功遷移其他工作負載之後，請儲存任務關鍵工作負載以供上次使用。 **Note** 暫時性 (CDP Data Engineering) 工作負載比持久性 (CDP Data Warehouse) 工作負載更容易遷移。在遷移時考慮資料磁碟區和位置也很重要。挑戰可能包括持續將資料從內部部署環境複寫到雲端，以及變更資料擷取管道以將資料直接匯入雲端。	遷移潛在客戶

任務	Description	所需的技能
討論 CDH、HDP、CDP 和舊版應用程式遷移活動。	考慮並開始規劃下列 Cloudera Workload Manager 活動： • 要複製到 AWS 環境的資料和工作負載 • 雲端就緒資料 • 雜訊鄰，使用資源並為其他租戶建立問題 • 彈性工作負載 • 具有高營運開銷的小型叢集	遷移潛在客戶

任務	Description	所需的技能
完成 Cloudera Replication Manager 要求和建議。	使用 Cloudera Professional Services 和您的 SI 準備將工作負載遷移到 AWS 上的 CDP 公有雲端環境。了解下列要求和建議可協助您避免在安裝 Replication Manager 服務期間和之後的常見問題。 • 檢閱 Replication Manager 支援文件，以確認您符合環境和系統需求。如需詳細資訊，請參閱 Cloudera 網站上的 CDP Public Cloud Replication Manager 支援矩陣。 • 您不需要對要安裝 Replication Manager 應用程式和 Data Lifecycle Manager (DLM) 引擎的節點進行根存取。 • 在 Replication Manager 的初始安裝期間安裝 Apache Hive，除非您確定未來不會使用 Hive 複寫。如果您在 Replication Manager 中建立 HDFS 複寫政策之後決定安裝 Hive，則必須在新增 Hive 之後刪除並重新建立所有 HDFS 複寫政策。 • Replication Manager 中使用的叢集必須具有對稱組態。針對安全性 (Kerberos)、使用者管理 (LDAP/AD) 和 Knox Proxy，複寫關係	遷移潛在客戶

任務	Description	所需的技能
	中的每個叢集的設定必須完全相同。Hadoop 分散式檔案系統 (HDFS)、Apache Hive、Apache Knox、Apache Ranger 和 Apache Atlas 等叢集服務可以具有不同的組態，以實現高可用性 (HA)。例如，來源和目標叢集可能具有單獨的 HA 和非 HA 組態。	

將 CDP 遷移至 AWS

任務	Description	所需的技能
使用 Cloudera Workload Manager 遷移開發/測試環境的第一個工作負載。	您的 SI 可協助您將第一個工作負載遷移至 AWS 雲端。這應該是非面向客戶或關鍵任務的應用程式。開發/測試遷移的理想候選項目是具有雲端可輕鬆擷取資料的應用程式，例如 CDP Data Engineering 工作負載。這是一種暫時性工作負載，與 CDP Data Warehouse 工作負載等持續性工作負載相比，存取它的使用者通常較少，而 CDP Data Warehouse 工作負載可能有許多需要不間斷存取的使用者。Data Engineering 工作負載並非持久性，如果發生錯誤，這可將業務影響降至最低。不過，這些任務對於生產報告至關重要，因此請先排定	遷移潛在客戶

任務	Description	所需的技能
	低影響資料工程工作負載的優先順序。	
視需要重複遷移步驟。	Cloudera Workload Manager 可協助識別最適合雲端的工作負載。它提供諸如雲端效能評分、目標環境的大小/容量計劃，以及複寫計劃的指標。遷移的最佳候選項目是季節性工作負載、臨機操作報告，以及不會耗用許多資源的間歇性任務。 Cloudera Replication Manager 會將資料從內部部署移至雲端，以及從雲端移至內部部署。 使用 Workload Manager 主動最佳化資料倉儲、資料工程和機器學習的工作負載、應用程式、效能和基礎設施容量。如需如何現代化資料倉儲的完整指南，請參閱 Cloudera 網站。	Cloudera 中小企業

相關資源

Cloudera 文件：

- 向 CDP、Cloudera Manager 和 Replication Manager 註冊傳統叢集：
  - 管理主控台
  - Replication Manager hive 複寫
- Sentry 複寫
- Sentry 許可
- Data Hub 叢集規劃檢查清單

- Workload Manager 架構
- 複寫管理員需求
- Cloudera 資料平台可觀測性
- AWS 需求

AWS 文件：

- 雲端資料遷移

解決將 Microsoft SQL Server 遷移至 AWS 雲端後的連線錯誤

Premkumar Chelladurai，Amazon Web Services

總結

將 Windows Server 2008 R2、2012 或 2012 R2 上執行的 Microsoft SQL Server 遷移至 Amazon Web Services (AWS) Cloud 上的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體後，SQL Server 的連線會失敗，並出現下列錯誤：

- `[Microsoft][ODBC SQL Server Driver][DBNETLIB] General Network error`
- `ERROR [08S01] [Microsoft][SQL Native Client]Communication link failure. System.Data.SqlClient.SqlException: A transport-level error has occurred when sending the request to the server. (provider: TCP Provider, error: 0 - An existing connection was forcibly closed by the remote host.)`
- `TCP Provider: The semaphore timeout period has expired`

此模式說明如何透過在作業系統 (OS) 和在 Windows Server 2008 R2、2012 或 2012 R2 上執行的 SQL Server 網路介面層級關閉 Windows Scalable Networking Pack (SNP) 功能來解決這些錯誤R2。

先決條件和限制

先決條件

- Windows Server 的管理員權限。
- 如果您使用 AWS Application Migration Service 做為遷移工具，則需要下列其中一個 Windows Server 版本：
  - Windows Server 2008 R2 Service Pack 1、2012 或 2012 R2
- 如果您使用 CloudEndure Migration 做為遷移工具，則需要下列其中一個 Windows Server 版本：
  - Windows Server 2003 R2 Service Pack 3、2008、2008 R2 Service Pack 1、2012 或 2012 R2

工具

- Amazon EC2 – Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以使用 Amazon EC2 視需要啟動任意數量或任意數量的虛擬伺服器，也可以向外擴展或向內擴展。
- Windows Server – Windows Server 是建置連線應用程式、網路和 Web 服務基礎設施的平台。

史詩

在作業系統和彈性網路介面層級關閉 SNP 功能

任務	Description	所需的技能
在作業系統層級關閉 SNP 功能。	1. 登入 Windows Server 並以管理員身分開啟命令提示。 2. 執行 `netsh int tcp show global` 命令。 3. 在輸出中，檢查 `Receive-Side Scaling` 或 `Chimney Offload` 是否處於 `enabled` 模式。如果其中一個是 `enabled`，請執行下列命令： • `netsh int tcp set global chimney=disabled` • `netsh int tcp set global rss=disabled`	AWS 管理員、AWS 系統管理員、遷移工程師、雲端管理員
在彈性網路介面層級關閉 SNP 功能。	1. 選擇開始，輸入 `ncpa.cpl`，然後按 Enter。 2. 在彈性網路轉接器上按一下滑鼠右鍵。 3. 在快顯功能表中，選擇屬性。 4. 在乙太網路轉接器屬性視窗中，選擇設定。 5. 在 Amazon Elastic Network Adapter Properties 快顯視窗中，選擇進階索引標籤。	AWS 管理員、雲端管理員、AWS 系統管理員

任務	Description	所需的技能
	6. 在屬性區段中，關閉所有卸載和 RSS。	

相關資源

- 如何疑難排解進階網路效能功能，例如 RSS 和 NetDMA

重新啟動 RHEL 來源伺服器後，在不停用 SELinux 的情況下自動重新啟動 AWS 複寫代理程式

Anil Kunapareddy、Venkatramana Chintha 和 Shanmugam Shanker，Amazon Web Services

總結

AWS Application Migration Service 有助於簡化、加速和自動化 Red Hat Enterprise Linux (RHEL) 工作負載遷移至 Amazon Web Services (AWS) 雲端。若要將來源伺服器新增至 Application Migration Service，請在伺服器上安裝 AWS 複寫代理程式。

Application Migration Service 提供即時、非同步的區塊層級複寫。這表示在整個複寫過程中，您可以繼續正常的 IT 操作。這些 IT 操作可能需要您在遷移期間重新啟動或重新啟動 RHEL 來源伺服器。如果發生這種情況，AWS 複寫代理程式不會自動重新啟動，而且您的資料複寫將會停止。一般而言，您可以將 Security-Enhanced Linux (SELinux) 設定為停用或寬鬆模式，以自動重新啟動 AWS Replication Agent。不過，您組織的安全政策可能禁止停用 SELinux，而且您可能也必須重新標記檔案。

此模式說明如何在 RHEL 來源伺服器在遷移期間重新啟動或重新啟動時，在不關閉 SELinux 的情況下自動重新啟動 AWS 複寫代理程式。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 您希望遷移至 AWS 雲端的現場部署 RHEL 工作負載。
- 從 Application Migration Service 主控台初始化的 Application Migration Service。只有在您第一次使用此服務時，才需要初始化。如需說明，請參閱 Application Migration Service 文件。
- Application Migration Service 的現有 AWS Identity and Access Management (IAM) 政策。如需詳細資訊，請參閱 Application Migration Service 文件。

版本

- RHEL 第 7 版或更新版本

工具

AWS 服務

- AWS Application Migration Service 是一種高度自動化lift-and-shift（重新託管）解決方案，可簡化、加速和降低將應用程式遷移至 AWS 的成本。

Linux 命令

下表提供您將在 RHEL 來源伺服器上執行的 Linux 命令清單。這些也會在此模式的 epics 和故事中說明。

命令	Description
`#systemctl –version`	識別系統版本。
`#systemctl list-units --type=service`	列出 RHEL 伺服器上可用的所有作用中服務。
`#systemctl list-units --type=service | grep running`	列出目前在 RHEL 伺服器上執行的所有服務。
`#systemctl list-units --type=service | grep failed`	列出 RHEL 伺服器重新啟動或重新啟動後無法載入的所有服務。
`restorecon -Rv /etc/rc.d/init.d/aws-replication-service`	將內容變更為 `aws-replication-service` 。
`yum install policycoreutils*`	安裝 SELinux 系統操作所需的政策核心公用程式。
`ausearch -c "insmod" --raw | audit2allow -M my-modprobe`	搜尋稽核日誌並建立 政策的模組。
`semodule -i my-modprobe.pp`	啟用政策。
`cat my-modprobe.te`	顯示 `my-modprobe.te` 檔案的內容。
`semodule -l | grep my-modprobe`	檢查政策是否已載入 SELinux 模組。

史詩

安裝 AWS 複寫代理程式並重新啟動 RHEL 來源伺服器

任務	Description	所需的技能
使用存取金鑰和私密存取金鑰建立 Application Migration Service 使用者。	若要安裝 AWS 複寫代理程式，您必須使用所需的 AWS 登入資料建立 Application Migration Service 使用者。如需說明，請參閱 Application Migration Service 文件。	遷移工程師
安裝 AWS 複寫代理程式。	1. 登入 AWS 管理主控台，並在 https：//https://console.aws.amazon.com/mgn/home 開啟 AWS Migration Service 主控台。 2. 遵循 Application Migration Service 文件中的指示來設定複寫設定。 3. 遵循 Application Migration Service 文件中的指示安裝 AWS Replication Agent。 4. 在來源伺服器頁面上，選擇 RHEL 來源伺服器，然後選擇複寫以開始初始複寫。如需詳細資訊，請參閱 Application Migration Service 文件。	遷移工程師
重新啟動或重新啟動 RHEL 來源伺服器。	當 RHEL 來源伺服器的資料複寫狀態顯示在遷移儀表板上停滯時，請重新啟動或重新啟動 RHEL 來源伺服器。	遷移工程師

任務	Description	所需的技能
檢查資料複寫狀態。	等待一小時，然後在遷移儀表板上再次檢查資料複寫狀態。它應該處於運作狀態。	遷移工程師

檢查 RHEL 來源伺服器上的 AWS 複寫代理程式狀態

任務	Description	所需的技能
識別系統版本。	開啟 RHEL 來源伺服器的命令列界面，並執行下列命令來識別系統版本： `#systemctl –version`	遷移工程師
列出所有作用中的服務。	若要列出 RHEL 伺服器上可用的所有作用中服務，請執行 命令： `#systemctl list-units --type=service`	遷移工程師
列出所有執行中的服務。	若要列出目前在 RHEL 伺服器上執行的所有服務，請使用 命令： `#systemctl list-units --type=service | grep running`	遷移工程師
列出所有無法載入的服務。	若要列出 RHEL 伺服器重新啟動或重新啟動後無法載入的所有服務，請執行 命令： `#systemctl list-units --type=service | grep failed`	遷移工程師

建立並執行 SELinux 模組

任務	Description	所需的技能
變更安全內容。	在 RHEL 來源伺服器的命令列界面中，執行下列命令，將安全內容變更為 AWS 複寫服務： `restorecon -Rv /etc/rc.d/init.d/aws-replication-service`	遷移工程師
安裝核心公用程式。	若要安裝 SELinux 系統操作所需的核心公用程式及其政策，請執行 命令： `yum install policycoreutils*`	遷移工程師
搜尋稽核日誌並建立 政策的模組。	執行 命令： `ausearch -c "insmod" --raw | audit2allow -M my-modprobe`	遷移工程師
顯示 my-modprobe-te 檔案的內容。	`my-modprobe.te` 檔案是由 audit2allow 命令產生。它包含 SELinux 網域、政策來源目錄和子目錄，並指定與網域相關聯的存取向量規則和轉換。若要顯示檔案的內容，請執行 命令： `cat my modprobe.te`	遷移工程師
啟用政策。	若要插入模組並將政策套件設為作用中，請執行 命令：	遷移工程師

任務	Description	所需的技能
	`semodule -i my-modprobe.pp`	
檢查是否已載入模組。	執行 命令： `semodule -l | grep my-modprobe` 載入 SELinux 模組後，您不再需要在遷移期間將 SELinux 設定為已停用或寬鬆模式。	遷移工程師
重新啟動或重新啟動 RHEL 來源伺服器，並確認資料複寫狀態。	開啟 AWS Migration Service 主控台，導覽至資料複寫進度，然後重新啟動或重新啟動 RHEL 來源伺服器。資料複寫現在應在 RHEL 來源伺服器重新啟動後自動恢復。	遷移工程師

相關資源

- 應用程式遷移服務文件
- 技術訓練資料
- 故障診斷 AWS 複寫代理程式問題
- Application Migration Service 政策

重新架構師

主題

- 將 Oracle 的 VARCHAR2(1) 資料類型轉換為 Amazon Aurora PostgreSQL 的布林資料類型
- 在 Aurora PostgreSQL 相容中建立應用程式使用者和角色
- 使用 PostgreSQL 相容 Aurora 全域資料庫模擬 Oracle DR
- 從 SQL Server 遷移至 PostgreSQL 時，實作 PII 資料的 SHA1 雜湊
- 使用 Oracle SQL Developer 和 AWS SCT，逐步從 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL
- 在 Aurora PostgreSQL 相容中使用檔案編碼將 BLOB 檔案載入 TEXT
- 使用 AWS SCT 和 AWS DMS 將 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL AWS CLI CloudFormation
- 使用 AWS DMS，以 SSL 模式將 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL
- 將 Oracle SERIALLY_REUSABLE pragma 套件遷移至 PostgreSQL
- 將 Oracle 外部資料表遷移至 Amazon Aurora PostgreSQL 相容
- 將函數型索引從 Oracle 遷移至 PostgreSQL
- 使用延伸模組將 Oracle 原生函數遷移至 PostgreSQL
- 使用 AWS DMS 將 Db2 資料庫從 Amazon EC2 遷移至 Aurora MySQL 相容
- 使用 AWS DMS 將 Microsoft SQL Server 資料庫從 Amazon EC2 遷移至 Amazon DocumentDB
- 將內部部署 ThoughtSpot Falcon 資料庫遷移至 Amazon Redshift
- 使用 Oracle GoldenGate 從 Oracle 資料庫遷移至 Amazon RDS for PostgreSQL
- 使用 AWS DMS 將 Oracle 分割的資料表遷移至 PostgreSQL
- 從 Amazon RDS for Oracle 遷移至 Amazon RDS for MySQL
- 使用 AWS DMS 和 AWS SCT 從 Amazon EC2 上的 IBM Db2 遷移至 Aurora PostgreSQL 相容 Amazon EC2
- 使用 SharePlex 和 AWS DMS 從 Oracle 8i 或 9i 遷移至 Amazon RDS for PostgreSQL
- 使用具體化視觀表和 AWS DMS，從 Oracle 8i 或 9i 遷移至 Amazon RDS for PostgreSQL
- 使用 AWS DMS 和 AWS SCT 從 Oracle on Amazon EC2 遷移至 Amazon RDS for MySQL
- 使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫從 Amazon EC2 遷移至 Amazon RDS for MariaDB
- 使用 AWS DMS 和 AWS SCT 將內部部署 Oracle 資料庫遷移至 Amazon RDS for MySQL
- 使用 Oracle 旁觀者和 AWS DMS 將內部部署 Oracle 資料庫遷移至 Amazon RDS for PostgreSQL

- 使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫遷移至 Amazon Redshift
- 使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫遷移至 Aurora PostgreSQL
- 將資料從現場部署 Oracle 資料庫遷移至 Aurora PostgreSQL
- 使用 AWS DMS 從 SAP ASE 遷移至 Amazon RDS for SQL Server
- 使用 AWS DMS 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon Redshift
- 使用 AWS SCT 資料擷取代理程式將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon Redshift
- 將舊版應用程式從 Oracle Pro*C 遷移至 ECPG
- 將虛擬產生的資料欄從 Oracle 遷移至 PostgreSQL
- 在 Aurora PostgreSQL 相容上設定 Oracle UTL_FILE 功能
- 從 Oracle 遷移到 Amazon Aurora PostgreSQL 後驗證資料庫物件

將 Oracle 的 VARCHAR2(1) 資料類型轉換為 Amazon Aurora PostgreSQL 的布林資料類型

Naresh Damera，Amazon Web Services

總結

從 Amazon Relational Database Service (Amazon RDS) for Oracle 遷移至 Amazon Aurora PostgreSQL 相容版本期間，您可能會在驗證 AWS Database Migration Service () 中的遷移時遇到資料不符的情況AWS DMS。若要防止此不相符，您可以將 VARCHAR2(1) 資料類型轉換為布林資料類型。

VARCHAR2 資料類型存放可變長度文字字串，而 VARCHAR2(1) 表示字串長度為 1 個字元或 1 個位元組。如需 VARCHAR2 的詳細資訊，請參閱 Oracle 內建資料類型 (Oracle 文件）。

在此模式中，在範例來源資料表欄中，VARCHAR2(1) 資料為 Y，表示是，或 N，表示否。 此模式包含使用 AWS DMS 和 AWS Schema Conversion Tool (AWS SCT) 將此資料類型從 VARCHAR2(1) 中的 Y 和 N 值轉換為布林值中的 true 或 false 值的指示。

目標對象

對於使用 將 Oracle 資料庫遷移至 Aurora PostgreSQL 相容資料庫的使用者，建議使用此模式 AWS DMS。當您完成遷移時，請遵循將 Oracle 轉換為 Amazon RDS for PostgreSQL 或 Amazon Aurora PostgreSQL (AWS SCT 文件） 中的建議。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 確認您的環境已為 Aurora 做好準備，包括設定登入資料、許可和安全群組。如需詳細資訊，請參閱設定 Amazon Aurora 的環境 (Aurora 文件）。
- 來源 Amazon RDS for Oracle 資料庫，其中包含具有 VARCHAR2(1) 資料的資料表資料欄。
- 目標 Amazon Aurora PostgreSQL 相容資料庫執行個體。如需詳細資訊，請參閱建立資料庫叢集並連線至 Aurora PostgreSQL 資料庫叢集上的資料庫 (Aurora 文件）。

產品版本

- Amazon RDS for Oracle 12.1.0.2 版或更新版本。

- AWS DMS 3.1.4 版或更新版本。如需詳細資訊，請參閱使用 Oracle 資料庫做為 的來源 AWS DMS和使用 PostgreSQL 資料庫做為 （ 文件 ） 的目標 AWS DMS。AWS DMS 我們建議您使用最新版本的 AWS DMS ，以獲得最全面的版本和功能支援。
- AWS Schema Conversion Tool (AWS SCT) 1.0.632 版或更新版本。我們建議您使用最新版本的 AWS SCT ，以獲得最全面的版本和功能支援。
- Aurora 支援 Aurora PostgreSQL 相容資料庫引擎版本中列出的 PostgreSQL 版本 (Aurora 文件 ）。PostgreSQL

Architecture

來源技術堆疊

Amazon RDS for Oracle 資料庫執行個體

目標技術堆疊

Amazon Aurora PostgreSQL 相容資料庫執行個體

來源和目標架構

工具

AWS 服務

- Amazon Aurora PostgreSQL 相容版本是完全受管的 ACID 相容關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。
- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移到 AWS 雲端 或在雲端和內部部署設定的組合之間遷移。
- Amazon Relational Database Service (Amazon RDS) for Oracle 可協助您在 中設定、操作和擴展 Oracle 關聯式資料庫 AWS 雲端。
- AWS Schema Conversion Tool (AWS SCT) 透過自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，支援異質資料庫遷移。

其他服務

- Oracle SQL Developer 是一種整合的開發環境，可簡化傳統和雲端部署中 Oracle 資料庫的開發和管理。在此模式中，您會使用此工具連線至 Amazon RDS for Oracle 資料庫執行個體並查詢資料。

- pgAdmin 是 PostgreSQL 的開放原始碼管理工具。它提供圖形界面，可協助您建立、維護和使用資料庫物件。在此模式中，您會使用此工具連線至 Aurora 資料庫執行個體並查詢資料。

史詩

準備遷移

任務	Description	所需的技能
建立資料庫遷移報告。	1. 在 中 AWS SCT，建立資料庫遷移評估報告。如需詳細資訊，請參閱建立遷移評估報告。 2. 檢閱並執行遷移評估報告中的動作項目。如需詳細資訊，請參閱評估報告動作項目。	DBA、開發人員
在目標資料庫上停用外部金鑰限制條件。	在 PostgreSQL 中，外部金鑰是透過使用觸發程序來實作。在完全載入階段期間， 一次 AWS DMS 載入一個資料表。我們強烈建議您使用下列其中一種方法，在完全載入期間停用外部金鑰限制條件： • 暫時停用執行個體的所有觸發，並完成完全載入。 • 在 PostgreSQL 中使用 `session_replication_role` 參數。 如果無法停用外部金鑰限制，請為父資料表和子資料表特有的主要資料建立 AWS DMS 遷移任務。	DBA、開發人員

任務	Description	所需的技能

停用目標資料庫上的主索引鍵和唯一索引鍵。	使用下列命令，停用目標資料庫上的主索引鍵和限制條件。這有助於改善初始載入任務的效能。
	`ALTER TABLE <table> DISABLE PRIMARY KEY;`
	`ALTER TABLE <table> DISABLE CONSTRAINT <constraint_name>;`
		DBA、開發人員

建立初始載入任務。	在 中 AWS DMS，建立初始載入的遷移任務。如需說明，請參閱建立任務。針對遷移方法，選擇遷移現有資料。此遷移方法在 API `Full Load`中呼叫。尚未啟動此任務。	DBA、開發人員

任務	Description	所需的技能
編輯初始載入任務的任務設定。	編輯任務設定以新增資料驗證。這些驗證設定必須在 JSON 檔案中建立。如需說明和範例，請參閱指定任務設定。新增下列驗證： • 若要驗證 VARCHAR2(1) 資料是否準確轉換為目標資料庫中的布林值，請在此模式的其他資訊區段中的資料驗證指令碼中新增程式碼。驗證指令碼會將目標資料表中的布林值 1 轉換為 Y，並將 0 轉換為 N，然後將目標資料表中的值與來源資料表進行比較。 若要驗證其餘的資料遷移，請在任務中啟用資料驗證。如需詳細資訊，請參閱資料驗證任務設定。	AWS 管理員，DBA
建立進行中複寫任務。	在 中 AWS DMS，建立讓目標資料庫與來源資料庫保持同步的遷移任務。如需說明，請參閱建立任務。針對遷移方法，選擇僅複寫資料變更。尚未啟動此任務。	DBA

測試遷移任務

任務	Description	所需的技能
建立測試的範例資料。	在來源資料庫中，建立包含資料的範例資料表以供測試之用。	開發人員
確認沒有衝突的活動。	使用 `pg_stat_activity` 來檢查伺服器上可能影響遷移的任何活動。如需詳細資訊，請參閱統計資料收集器 (PostgreSQL 文件）。	AWS 管理員
啟動 AWS DMS 遷移任務。	在 AWS DMS 主控台的儀表板頁面上，啟動您在上一個史詩中建立的初始載入和持續複寫任務。	AWS 管理員
監控任務和資料表載入狀態。	在遷移期間，監控任務狀態和資料表狀態。當初始載入任務完成時，在資料表統計資料索引標籤上： • 載入狀態應為資料表完成。 • 應驗證驗證狀態。	AWS 管理員
驗證遷移結果。	使用 pgAdmin 查詢目標資料庫上的資料表。成功的查詢表示資料已成功遷移。	開發人員
在目標資料庫上新增主索引鍵和外部索引鍵。	在目標資料庫上建立主索引鍵和外部索引鍵。如需詳細資訊，請參閱 ALTER TABLE (PostgreSQL 網站）。	DBA
清除測試資料。	在來源和目標資料庫上，清除為單位測試而建立的資料。	開發人員

剪下

任務	Description	所需的技能
完成遷移。	使用實際來源資料重複上述史詩測試遷移任務。這會將資料從來源遷移到目標資料庫。	開發人員
驗證來源和目標資料庫是否同步。	驗證來源和目標資料庫是否同步。如需詳細資訊和說明，請參閱AWS DMS 資料驗證。	開發人員
停止來源資料庫。	停止 Amazon RDS for Oracle 資料庫。如需說明，請參閱暫時停止 Amazon RDS 資料庫執行個體。當您停止來源資料庫時， 中的初始載入和持續複寫任務 AWS DMS 會自動停止。停止這些任務不需要其他動作。	開發人員

相關資源

AWS 參考

- 使用 AWS DMS 和 將 Oracle 資料庫遷移至 Aurora PostgreSQL AWS SCT (AWS 方案指引)
- 將 Oracle 轉換為 Amazon RDS for PostgreSQL 或 Amazon Aurora PostgreSQL (AWS SCT 文件)
- 運作方式 AWS DMS(AWS DMS 文件)

其他參考

- 布林資料類型 (PostgreSQL 文件)
- Oracle 內建資料類型 (Oracle 文件)
- pgAdmin (pgAdmin 網站)
- SQL 開發人員 (Oracle 網站)

教學課程和影片

- 入門 AWS DMS
- Amazon RDS 入門
- 簡介 AWS DMS（影片）
- 了解 Amazon RDS （影片）

其他資訊

資料驗證指令碼

下列資料驗證指令碼會將 1 轉換為 Y，並將 0 轉換為 N。 這有助於 AWS DMS 任務成功完成並通過資料表驗證。

```
{
"rule-type": "validation",
"rule-id": "5",
"rule-name": "5",
"rule-target": "column",
"object-locator": {
"schema-name": "ADMIN",
"table-name": "TEMP_CHRA_BOOL",
"column-name": "GRADE"
},
"rule-action": "override-validation-function",
"target-function": "case grade when '1' then 'Y' else 'N' end"
        }
```

指令碼中的 `case`陳述式會執行驗證。如果驗證失敗， 會在目標資料庫執行個體的 public.awsdms_validation_failures_v1 資料表中 AWS DMS 插入記錄。此記錄包含資料表名稱、錯誤時間，以及來源和目標資料表中不相符值的詳細資訊。

如果您未將此資料驗證指令碼新增至 AWS DMS 任務，且資料已插入目標資料表，則 AWS DMS 任務會將驗證狀態顯示為不相符的記錄。

在 AWS SCT 轉換期間， AWS DMS 遷移任務會將 VARCHAR2(1) 資料類型變更為布林值，並在資料"NO"欄上新增主索引鍵限制條件。

在 Aurora PostgreSQL 相容中建立應用程式使用者和角色

Abhishek Verma，Amazon Web Services

總結

當您遷移至 Amazon Aurora PostgreSQL 相容版本時，來源資料庫上存在的資料庫使用者和角色必須在 Aurora PostgreSQL 相容資料庫中建立。您可以使用兩種不同的方法，在 Aurora PostgreSQL 相容中建立使用者和角色：

- 在目標中使用與來源資料庫中類似的使用者和角色。在此方法中，會從來源資料庫擷取使用者和角色的資料定義語言 (DDLs)。然後，它們會轉換並套用至目標 Aurora PostgreSQL 相容資料庫。例如，部落格文章使用 SQL 將使用者、角色和授予從 Oracle 映射到 PostgreSQL 涵蓋使用從 Oracle 來源資料庫引擎擷取。
- 使用在開發、管理和執行資料庫中其他相關操作時常用的標準化使用者和角色。這包括由個別使用者執行的唯讀、讀寫、開發、管理和部署操作。

此模式包含在標準化使用者和角色方法所需的 Aurora PostgreSQL 相容中建立使用者和角色所需的授予。使用者和角色建立步驟符合授予最低權限給資料庫使用者的安全政策。下表列出使用者、其對應的角色，以及其在資料庫上的詳細資訊。

使用者	角色	用途
APP_read	APP_RO	用於結構描述上的唯讀存取 APP
APP_WRITE	APP_RW	用於結構描述上的寫入和讀取操作 APP
APP_dev_user	APP_DEV	用於結構描述 上的開發用途APP_DEV，具有結構描述上的唯讀存取權 APP
Admin_User	rds_superuser	用於在資料庫上執行管理員操作
APP	APP_DEP	用於在APP結構描述下建立物件，以及在APP結構描述中部署物件

先決條件和限制

先決條件

- 作用中的 Amazon Web Services (AWS) 帳戶
- PostgreSQL 資料庫、Amazon Aurora PostgreSQL 相容版本資料庫，或 PostgreSQL 資料庫的 Amazon Relational Database Service (Amazon RDS)

產品版本

- PostgreSQL 的所有版本

Architecture

來源技術堆疊

- 任何資料庫

目標技術堆疊

- Amazon Aurora PostgreSQL 相容

目標架構

下圖顯示 Aurora PostgreSQL 相容資料庫中的使用者角色和結構描述架構。

自動化和擴展

此模式包含使用者、角色和結構描述建立指令碼，您可以執行多次，而不會影響來源或目標資料庫的現有使用者。

工具

AWS 服務

- Amazon Aurora PostgreSQL 相容版本是完全受管的 ACID 相容關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。

其他服務

- psql 是一種終端型前端工具，會隨每次 PostgreSQL 資料庫安裝一起安裝。它具有執行 SQL、PL-PGSQL 和作業系統命令的命令列界面。
- pgAdmin 是 PostgreSQL 的開放原始碼管理工具。它提供圖形界面，可協助您建立、維護和使用資料庫物件。

史詩

建立使用者和角色

任務	Description	所需的技能

建立部署使用者。	部署使用者APP將用於在部署期間建立和修改資料庫物件。使用下列指令碼在結構描述APP_DEP中建立部署使用者角色APP。驗證存取權，以確保此使用者只有在所需結構描述中建立物件的權限APP。
	1. 連線至管理員使用者，並建立結構描述。
	`CREATE SCHEMA APP;`
	2. 建立使用者。
	`CREATE USER APP WITH PASSWORD <password> ;`
	3. 建立角色。
	`CREATE ROLE APP_DEP ; GRANT all on schema APP to APP_DEP ; GRANT USAGE ON SCHEMA APP to APP_DEP ;`
		DBA

任務	Description	所需的技能
	GRANT connect on database <db_name> to APP_DEP ; GRANT APP_DEP to APP; 4. 若要測試權限，請連線至APP並建立資料表。 set search_path to APP; SET CREATE TABLE test(id integer) ; CREATE TABLE 5. 檢查權限。 select schemaname , tablename , tableowne r from pg_tables where tablename like 'test' ; schemaname | tablename | tableowner APP | test | APP	

任務	Description	所需的技能

建立唯讀使用者。	唯讀使用者APP_read將用於在結構描述 中執行唯讀操作APP。使用下列指令碼來建立唯讀使用者。驗證存取權，以確保此使用者僅具有讀取結構描述中物件的權限，APP並自動授予在結構描述中建立之任何新物件的讀取存取權APP。
	
	1. 建立使用者 APP_read。
	
	`create user APP_read ;`
	`alter user APP_read`
	` with password`
	` 'your_password' ;`
	
	2. 建立角色。
	
	`CREATE ROLE APP_ro ;`
	`GRANT SELECT ON ALL`
	` TABLES IN SCHEMA APP`
	`  TO APP_RO  ;`
	`GRANT USAGE ON SCHEMA`
	` APP  TO APP_RO`
	`GRANT CONNECT ON`
	` DATABASE testdb TO`
	` APP_RO ;`
	`GRANT APP_RO TO`
	` APP_read;`
	
	3. 若要測試權限，請使用APP_read使用者登入。
	
	`set search_path to`
	` APP ;`
	`create table test1( id`
	` integer) ;`
		DBA

任務	Description	所需的技能
	ERROR: permission denied for schema APP LINE 1: create table test1(id integer) ; insert into test values (34) ; ERROR: permission denied for table test SQL state: 42501 select from test no rows selected	

任務	Description	所需的技能

建立讀取/寫入使用者。	讀取/寫入使用者APP_WRITE 將用於對結構描述 執行讀取和寫入操作APP。使用下列指令碼來建立讀取/寫入使用者，並授予該APP_RW角色。驗證存取權，以確保此使用者僅對結構描述中的物件具有讀取和寫入權限，APP並自動為結構描述 中建立的任何新物件授予讀取和寫入存取權APP。
	
	1. 建立使用者。
	
	`CREATE USER APP_WRITE ; alter user APP_WRITE with password 'your_password' ;`
	
	2. 建立角色。
	
	`CREATE ROLE APP_RW; GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA APP TO APP_RW ; GRANT CONNECT ON DATABASE postgres to APP_RW ; GRANT USAGE ON SCHEMA APP to APP_RW ; ALTER DEFAULT PRIVILEGES IN SCHEMA APP GRANT SELECT, INSERT, UPDATE, DELETE ON TABLES TO APP_RW ;`	

任務	Description	所需的技能
	GRANT APP_RW to APP_WRITE 3. 若要測試權限，請使用 APP_WRITE 使用者登入。 SET SEARCH_PATH to APP; CREATE TABLE test1(id integer) ; ERROR: permission denied for schema APP LINE 1: create table test1(id integer) ; SELECT * FROM test ; id ---- 12 INSERT INTO test values (31) ; INSERT 0 1	

任務	Description	所需的技能

建立管理員使用者。	管理員使用者Admin_User 將用於對資料庫執行管理員操作。這些操作的範例為CREATE ROLE和 CREATE DATABASE。 Admin_User 使用內建角色rds_superuser 在資料庫上執行管理員操作。使用下列指令碼來建立和測試Admin_User 資料庫中管理員使用者的權限。
	
	1. 建立使用者並授予角色。
	
	`create user Admin_User WITH PASSWORD ‘Your password’ ALTER user Admin_user CREATEDB; ALTER user Admin_user CREATEROLE;`
	
	2. 若要測試權限，請從 Admin_User 使用者登入。
	
	`SELECT * FROM APP.test ; id ---- 31 CREATE ROLE TEST ; CREATE DATABASE test123 ;`
		DBA

任務	Description	所需的技能

建立開發使用者。	開發使用者APP_dev_user 將有權在其本機結構描述中建立物件，APP_DEV並在結構描述 中建立讀取存取權APP。使用下列指令碼來建立和測試APP_dev_user 資料庫中使用者的權限。
	
	1. 建立使用者。
	
	`CREATE USER APP1_dev_user with password ‘your password’;`
	
	2. 建立 APP_DEV 的結構描述App_dev_user 。
	
	`CREATE SCHEMA APP1_DEV ;`
	
	3. 建立 APP_DEV 角色。
	
	`CREATE ROLE APP1_DEV ;`
	`GRANT APP1_RO to APP1_DEV ;`
	`GRANT SELECT ON ALL TABLES IN SCHEMA APP1_DEV to APP1_dev_user`
	`GRANT USAGE, CREATE ON SCHEMA APP1_DEV to APP1_DEV_USER`
	`GRANT APP1_DEV to APP1_DEV_USER ;`
	
	4. 若要測試權限，請從 登入APP_dev_user 。
		DBA

任務	Description	所需的技能

	`CREATE TABLE APP1_dev.test1( id integer ) ;`
	`CREATE TABLE`
	`INSERT into APP1_dev.test1 ( select * from APP1.test );`
	`INSERT 0 1`
	`CREATE TABLE APP1.test4 ( id int) ;`
	`ERROR: permissio n denied for schema APP1`
	`LINE 1: create table APP1.test4 ( id int) ;`	

相關資源

PostgreSQL 文件

- 建立角色
- 建立使用者
- 預先定義的角色

其他資訊

PostgreSQL 14 增強功能

PostgreSQL 14 提供一組預先定義的角色，可讓您存取某些常用的特權功能和資訊。管理員（包括具有 CREATE ROLE權限的角色）可以將這些角色或其環境中的其他角色授予使用者，讓他們能夠存取指定的功能和資訊。

管理員可以使用 GRANT命令授予使用者對這些角色的存取權。例如，若要將pg_signal_backend角色授予 Admin_User，您可以執行下列命令。

```
GRANT pg_signal_backend TO Admin_User;
```

此pg_signal_backend角色旨在允許管理員啟用信任的非超級使用者角色，將訊號傳送至其他後端。如需詳細資訊，請參閱 PostgreSQL 14 增強功能。

微調存取權

在某些情況下，可能需要為使用者提供更精細的存取（例如，資料表型存取或資料欄型存取）。在這種情況下，可以建立其他角色，將這些權限授予使用者。如需詳細資訊，請參閱 PostgreSQL Grants。

使用 PostgreSQL 相容 Aurora 全域資料庫模擬 Oracle DR

HariKrishna Boorgadda，Amazon Web Services

總結

企業災難復原 (DR) 的最佳實務基本上包含設計和實作容錯硬體和軟體系統，這些系統可以承受災難 (業務持續性） 並恢復正常操作 (業務恢復)，且介入最少，理想情況下不會遺失資料。建置容錯環境以滿足企業 DR 目標可能既昂貴又耗時，而且需要業務的強大承諾。

Oracle Database 提供三種不同的 DR 方法，相較於任何其他保護 Oracle 資料的方法，可提供最高層級的資料保護和可用性。

- Oracle 零資料遺失復原設備
- Oracle Active Data Guard
- Oracle GoldenGate

此模式提供使用 Amazon Aurora 全域資料庫模擬 Oracle GoldenGate DR 的方法。參考架構跨三個 AWS 區域使用 Oracle GoldenGate 進行 DR。模式會逐步解說來源架構到以 Amazon Aurora PostgreSQL 相容版本為基礎的雲端原生 Aurora 全域資料庫。

Aurora 全域資料庫專為具有全域足跡的應用程式而設計。單一 Aurora 資料庫跨越多個 AWS 區域，最多可有五個次要區域。Aurora 全域資料庫提供下列功能：

- 實體儲存層級複寫
- 低延遲全域讀取
- 從整個區域的中斷快速復原災難
- 快速跨區域遷移
- 跨區域的低複寫延遲
- 對資料庫的效能影響Little-to-no

如需 Aurora 全域資料庫功能和優點的詳細資訊，請參閱使用 Amazon Aurora 全域資料庫。如需意外和受管容錯移轉的詳細資訊，請參閱在 Amazon Aurora 全域資料庫中使用容錯移轉。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 用於應用程式連線的 Java Database Connectivity (JDBC) PostgreSQL 驅動程式
- 以 Amazon Aurora PostgreSQL 相容版本為基礎的 Aurora 全域資料庫
- 基於 Aurora PostgreSQL 相容，遷移至 Aurora 全域資料庫的 Oracle Real Application Clusters (RAC) 資料庫

Aurora 全域資料庫的限制

- 並非所有 AWS 區域都提供 Aurora 全域資料庫。如需支援區域的清單，請參閱 Aurora PostgreSQL 的 Aurora 全域資料庫。
- 如需有關不支援的功能以及 Aurora 全域資料庫的其他限制的資訊，請參閱 Amazon Aurora 全域資料庫的限制。

產品版本

- Amazon Aurora PostgreSQL 相容版本 10.14 版或更新版本

Architecture

來源技術堆疊

- Oracle RAC 四節點資料庫
- Oracle GoldenGate

來源架構

下圖顯示使用 Oracle GoldenGate 複寫之不同 AWS 區域中具有四節點 Oracle RAC 的三個叢集。

目標技術堆疊

- 以 Aurora PostgreSQL 相容為基礎的三個叢集 Amazon Aurora 全域資料庫，主要區域中有一個叢集，不同次要區域中有兩個叢集

目標架構

工具

AWS 服務

- Amazon Aurora PostgreSQL 相容版本是完全受管的 ACID 相容關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。
- Amazon Aurora 全域資料庫橫跨多個 AWS 區域，提供低延遲的全域讀取，並從可能影響整個 AWS 區域的罕見中斷中快速復原。

史詩

使用讀取器資料庫執行個體新增區域

任務	Description	所需的技能
連接一或多個次要 Aurora 叢集。	在 AWS 管理主控台上，選擇 Amazon Aurora。選取主要叢集，選擇動作，然後從下拉式清單中選擇新增區域。	DBA
選取執行個體類別。	您可以變更次要叢集的執行個體類別。不過，我們建議您保持與主要叢集執行個體類別相同的狀態。	DBA
新增第三個區域。	重複此史詩中的步驟，在第三個區域中新增叢集。	DBA

容錯移轉 Aurora 全域資料庫

任務	Description	所需的技能
從 Aurora 全域資料庫中移除主要叢集。	1. 在資料庫頁面上，選擇主要叢集。 2. 選擇從全域移除以容錯移轉至次要叢集。	DBA

任務	Description	所需的技能
重新設定您的應用程式，將寫入流量轉向新提升的叢集。	使用新提升叢集的端點修改應用程式中的端點。	DBA
停止對無法使用的叢集發出任何寫入操作。	停止應用程式和您移除之叢集的任何資料處理語言 (DML) 活動。	DBA
建立新的 Aurora 全域資料庫。	現在，您可以使用新提升的叢集建立 Aurora 全域資料庫，做為主要叢集。	DBA

啟動主要叢集

任務	Description	所需的技能
選取要從全域資料庫啟動的主要叢集。	在 Amazon Aurora 主控台的全域資料庫設定中，選擇主要叢集。	DBA
啟動叢集。	在動作下拉式清單中，選擇開始。此程序可能需要一些時間。重新整理畫面以查看狀態，或在操作完成後檢查叢集目前狀態的狀態欄。	DBA

清除資源

任務	Description	所需的技能
刪除剩餘的次要叢集。	容錯移轉試行完成後，從全域資料庫中移除次要叢集。	DBA
刪除主要叢集。	移除叢集。	DBA

相關資源

- 使用 Amazon Aurora 全球資料庫
- 使用 Amazon Aurora Global Database 的 Aurora PostgreSQL 災難復原解決方案 （部落格文章）

從 SQL Server 遷移至 PostgreSQL 時，實作 PII 資料的 SHA1 雜湊

Rajkumar Raghuwanshi 和 Jagadish Kantubugata，Amazon Web Services

總結

此模式說明如何在從 SQL Server 遷移至 Amazon RDS for PostgreSQL 或 Amazon Aurora PostgreSQL 相容時，為電子郵件地址實作安全雜湊演算法 1 (SHA1) 雜湊。電子郵件地址是個人身分識別資訊 (PII) 的範例。PII 是當直接檢視或與其他相關資料配對時，可用來合理推斷個人身分的資訊。

此模式涵蓋在不同資料庫定序和字元編碼之間維持一致雜湊值的挑戰，並提供使用 PostgreSQL 函數和觸發程序的解決方案。雖然此模式著重於 SHA1 雜湊，但可以適應 PostgreSQL `pgcrypto`模組支援的其他雜湊演算法。一律考慮雜湊策略的安全性影響，並在處理敏感資料時諮詢安全專家。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 來源 SQL Server 資料庫
- Target PostgreSQL 資料庫 (Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 相容)
- PL/pgSQL 編碼專業知識

限制

- 此模式需要根據使用案例進行資料庫層級定序變更。
- 尚未評估對大型資料集的效能影響。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

- Microsoft SQL Server 2012 或更新版本

Architecture

來源技術堆疊

- SQL Server

- .NET Framework

目標技術堆疊

- PostgreSQL
- `pgcrypto` 延伸模組

自動化和擴展

- 請考慮實作雜湊函數作為預存程序，以便於維護。
- 對於大型資料集，請評估效能，並考慮批次處理或索引策略。

工具

AWS 服務

- Amazon Aurora PostgreSQL 相容是完全受管且符合 ACID 規範的關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。
- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移到 AWS 雲端 或在雲端和內部部署設定的組合之間遷移。
- Amazon Relational Database Service Amazon RDS for PostgreSQL 可協助您在 中設定、操作和擴展 PostgreSQL 關聯式資料庫 AWS 雲端。
- AWS Schema Conversion Tool (AWS SCT) 透過自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，支援異質資料庫遷移。

其他工具

- pgAdmin 是 PostgreSQL 的開放原始碼管理工具。它提供圖形界面，可協助您建立、維護和使用資料庫物件。
- SQL Server Management Studio (SSMS) 是用於管理任何 SQL 基礎設施的整合環境。

最佳實務

- 使用適當的定序設定來處理目標資料庫端的特殊字元。
- 使用各種電子郵件地址進行徹底測試，包括非 ASCII 字元的地址。

- 在應用程式和資料庫層之間維持大小寫處理的一致性。
- 使用雜湊值對查詢效能進行基準測試。

史詩

分析來源雜湊實作

任務	Description	所需的技能
檢閱 SQL Server 程式碼。	若要檢閱產生 SHA1 雜湊的 SQL Server 程式碼，請執行下列動作： • 分析 SHA1 雜湊的現有 SQL Server 實作。 • 識別用於產生雜湊的確切方法。 • 記錄輸入參數和輸出格式。 • 檢閱任何資料類型轉換或轉換。 • 檢查定序設定及其影響。	資料工程師、DBA、應用程式開發人員
記錄雜湊演算法和資料轉換。	若要記錄確切的雜湊演算法和資料轉換，請執行下列動作： • 建立雜湊程序的詳細技術文件。 • 記錄step-by-step轉換邏輯。 • 指定輸入和輸出格式和資料類型。 • 包含邊緣案例和特殊字元處理。	應用程式開發人員、資料工程師、DBA

建立 PostgreSQL 雜湊函數

任務	Description	所需的技能
建立pgcrypto擴充功能。	若要建立pgcrypto擴充功能，請使用 pgAdmin/psql 執行下列命令： `CREATE EXTENSION pgcrypto;`	DBA，資料工程師
實作 PostgreSQL 函數。	實作下列 PostgreSQL 函數來複寫 SQL Server 雜湊邏輯。在高階，此函數使用下列步驟： 1. 選擇性地將輸入轉換為大寫。 2. 建立輸入的 SHA1 雜湊。 3. 此雜湊的最後 10 個位元組 (80 位元）。 4. 將這些位元組轉換為 64 位元整數。 `CREATE OR REPLACE FUNCTION utility.hex_to_bigint ( par_val character varying, par_upper character varying DEFAULT 'lower'::character varying) RETURNS bigint LANGUAGE 'plpgsql' AS $BODY$ DECLARE retnumber bigint;`	資料工程師、DBA、應用程式開發人員

任務	Description	所需的技能

	`    digest_bytes bytea;`
	`BEGIN`
	`    if lower(par_upper)`
	` = 'upper'`
	`    then`
	`        digest_bytes :=`
	` digest(upper(par_v`
	`al), 'sha1');`
	`    else`
	`        digest_bytes :=`
	` digest((par_val),`
	` 'sha1');`
	`    end if;`
	`    retnumber := ('x'`
	` || encode(substring(d`
	`igest_bytes, length(di`
	`gest_bytes)-10+1),`
	` 'hex'))::bit(64)::`
	`bigint;`
	`    RETURN retnumber;`
	`END;`
	`$BODY$;`	

任務	Description	所需的技能

測試函數。	若要測試函數，請使用 SQL Server 的範例資料來驗證相符的雜湊值。執行以下命令：
	
	`select 'alejandro_rosalez@example.com' as Email, utility.hex_to_bigint('alejandro_rosalez@example.com', 'upper') as HashValue;`
	
	`--OUTPUT`
	`/*`
	`email hashvalue`
	`"alejandro_rosalez@example.com" 451397011176045063`
	`*/`
		應用程式開發人員、DBA、資料工程師

實作自動雜湊的觸發

任務	Description	所需的技能

在相關資料表上建立觸發條件。	若要在相關資料表上建立觸發，以在插入或更新時自動產生雜湊值，請執行下列命令：
	
	`CREATE OR REPLACE FUNCTION update_email_hash()`
	`RETURNS TRIGGER`
	`AS $$`
	`BEGIN`
	`NEW.email_hash = utility.hex_to_big`
		應用程式開發人員、資料工程師、DBA

任務	Description	所需的技能

	`int(NEW.email, 'upper'); RETURN NEW; END; $$ LANGUAGE plpgsql;`
	
	`CREATE TRIGGER email_hash_trigger BEFORE INSERT OR UPDATE ON users FOR EACH ROW EXECUTE FUNCTION update_email_hash();`	

遷移現有資料

任務	Description	所需的技能
開發遷移指令碼或使用 AWS DMS。	開發遷移指令碼或使用 AWS DMS 來填入現有資料的雜湊值（包括存放在BIGINT來源系統中的雜湊值）。完成下列任務： • 使用雜湊值建立資料傳輸的遷移指令碼。 • 使用適當的轉換規則設定 AWS DMS 任務。 • 在 中設定來源和目標端點 AWS DMS。 • 實作錯誤處理和記錄機制。 • 為大型資料集設計批次處理策略。 • 建立驗證查詢以進行資料驗證。	資料工程師、應用程式開發人員、DBA

任務	Description	所需的技能
使用新的 PostgreSQL 雜湊函數。	若要使用新的 PostgreSQL 雜湊函數以確保一致性，請執行下列動作： • 實作驗證程序來驗證雜湊一致性。 • 在來源和目標系統之間建立比較指令碼。 • 設定雜湊值驗證的自動測試。 • 記錄任何差異和解決步驟。	應用程式開發人員、DBA、DevOps 工程師

更新應用程式查詢

任務	Description	所需的技能
識別應用程式查詢。	若要識別使用雜湊值的應用程式查詢，請執行下列動作： • 使用雜湊值分析查詢的應用程式程式碼庫。 • 檢閱參考雜湊操作的預存程序和函數。 • 記錄查詢效能指標和執行計畫。 • 識別雜湊型查詢的相依性。 • 映射受影響的應用程式元件。	應用程式開發人員、DBA、資料工程師
修改查詢。	如有必要，請修改查詢以使用新的 PostgreSQL 雜湊函數。請執行下列操作：	應用程式開發人員、DBA、資料工程師

任務	Description	所需的技能
	• 重構現有的查詢以使用 PostgreSQL 雜湊函數。 • 更新預存程序和函數。 • 實作和測試新的查詢模式。 • 針對效能最佳化修改後的查詢。	

測試和驗證

任務	Description	所需的技能
執行測試。	若要使用生產資料子集執行徹底測試，請執行下列動作： • 建立資料子集驗證的測試計畫。 • 擷取生產資料的代表性範例。 • 使用適當的組態設定測試環境。 • 執行資料載入和轉換測試。 • 執行容積和壓力測試。	應用程式開發人員、資料工程師、DBA
驗證雜湊值是否相符。	若要驗證 SQL Server 和 PostgreSQL 之間的雜湊值是否相符，請執行下列動作： • 開發雜湊值的比較指令碼。 • 建立雜湊比對的驗證報告。 • 實作自動化驗證程序。 • 記錄發現的任何差異。 • 分析和解決雜湊不相符的問題。	應用程式開發人員、資料工程師、DBA

任務	Description	所需的技能
驗證應用程式功能。	若要使用遷移的資料和新的雜湊實作來驗證應用程式功能，請執行下列動作： • 執行end-to-end應用程式測試。 • 使用雜湊資料驗證所有應用程式功能。 • 使用新實作測試應用程式效能。 • 驗證 API 整合和相依性。	應用程式開發人員、DBA、資料工程師

疑難排解

問題	解決方案
雜湊值不相符。	驗證來源和目標之間的字元編碼和定序。如需詳細資訊，請參閱在 Amazon Aurora 和 Amazon RDS （部落格） 上管理 PostgreSQL 中的定序變更。AWS

相關資源

AWS 部落格

- 在 Amazon Aurora 和 Amazon RDS 上的 PostgreSQL 中管理定序變更
- 使用從 欄位學到的最佳實務和經驗，將 SQL Server 遷移至 Amazon Aurora PostgreSQL

其他資源

- PostgreSQL pgcrypto 模組 (PostgreSQL 文件）
- PostgreSQL 觸發函數 (PostgreSQL 文件）
- SQL Server HASHBYTES 函數 (Microsoft 文件）

使用 Oracle SQL Developer 和 AWS SCT，逐步從 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL

Pinesh Singal，Amazon Web Services

總結

許多遷移策略和方法會分多個階段執行，持續數週到數個月。在此期間，您可能會因為要遷移至 PostgreSQL 資料庫執行個體的來源 Oracle 資料庫執行個體中的修補或升級而遇到延遲。為了避免這種情況，建議您將剩餘的 Oracle 資料庫程式碼逐步遷移至 PostgreSQL 資料庫程式碼。

此模式為在初始遷移後執行大量交易且必須遷移至 PostgreSQL 資料庫的多 TB Oracle 資料庫執行個體提供無停機時間的增量遷移策略。您可以使用此模式的step-by-step方法，將 Amazon Relational Database Service (Amazon RDS) for Oracle 資料庫執行個體逐步遷移至 Amazon RDS for PostgreSQL 資料庫執行個體，而無需登入 Amazon Web Services (AWS) 管理主控台。

模式使用 Oracle SQL Developer 尋找來源 Oracle 資料庫中兩個結構描述之間的差異。然後，您可以使用 AWS Schema Conversion Tool (AWS SCT) 將 Amazon RDS for Oracle 資料庫結構描述物件轉換為 Amazon RDS for PostgreSQL 資料庫結構描述物件。然後，您可以在 Windows 命令提示字元中執行 Python 指令碼，為來源資料庫物件的增量變更建立 AWS SCT 物件。

> **Note**
>
> 遷移生產工作負載之前，建議您在測試或非生產環境中，針對此模式的方法執行概念驗證 (PoC)。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 現有的 Amazon RDS for Oracle 資料庫執行個體。
- 現有的 Amazon RDS for PostgreSQL 資料庫執行個體。
- AWS SCT，安裝並設定適用於 Oracle 和 PostgreSQL 資料庫引擎的 JDBC 驅動程式。如需詳細資訊，請參閱 AWS SCT 文件中的安裝 AWS SCT 和安裝所需的資料庫驅動程式。
- Oracle SQL Developer，已安裝並設定。如需詳細資訊，請參閱 Oracle SQL Developer 文件。
- `incremental-migration-sct-sql.zip` 檔案（已連接），下載到您的本機電腦。

限制

- 來源 Amazon RDS for Oracle 資料庫執行個體的最低需求為：
  - Oracle 10.2 版和更新版本（適用於 10.x 版）、11g 版 (11.2.0.3.v1 版和更新版本）和最高 12.2 版，以及 Enterprise、Standard、Standard One 和 Standard Two 版 18c 版
- 您目標 Amazon RDS for PostgreSQL 資料庫執行個體的最低需求為：
  - PostgreSQL 9.4 版和更新版本（適用於 9.x 版）、10.x 版和 11.x 版
- 此模式使用 Oracle SQL Developer。如果您使用其他工具來尋找和匯出結構描述差異，結果可能會有所不同。
- Oracle SQL Developer 產生的 SQL 指令碼可能會引發轉換錯誤，這表示您需要執行手動遷移。
- 如果 AWS SCT 來源和目標測試連線失敗，請確定您已設定虛擬私有雲端 (VPC) 安全群組的 JDBC 驅動程式版本和傳入規則，以接受傳入流量。

產品版本

- Amazon RDS for Oracle 資料庫執行個體 12.1.0.2 版 (10.2 版及更新版本）
- Amazon RDS for PostgreSQL 資料庫執行個體 11.5 版 (9.4 版及更新版本）
- Oracle SQL Developer 19.1 版及更新版本
- AWS SCT 1.0.632 版及更新版本

Architecture

來源技術堆疊

- Amazon RDS for Oracle 資料庫執行個體

目標技術堆疊

- Amazon RDS for PostgreSQL 資料庫執行個體

來源和目標架構

下圖顯示 Amazon RDS for Oracle 資料庫執行個體遷移至 Amazon RDS for PostgreSQL 資料庫執行個體。

圖表顯示下列遷移工作流程：

1. 開啟 Oracle SQL Developer 並連線至來源和目標資料庫。
2. 產生差異報告，然後產生結構描述差異物件的 SQL 指令碼檔案。如需 diff 報告的詳細資訊，請參閱 Oracle 文件中的詳細 diff 報告。
3. 設定 AWS SCT 並執行 Python 程式碼。
4. SQL 指令碼檔案會從 Oracle 轉換為 PostgreSQL。
5. 在目標 PostgreSQL 資料庫執行個體上執行 SQL 指令碼檔案。

自動化和擴展

您可以將單一程式中多個功能的其他參數和安全相關變更新增至 Python 指令碼，以自動化此遷移。

工具

- AWS SCT – AWS Schema Conversion Tool (AWS SCT) 會將您現有的資料庫結構描述從一個資料庫引擎轉換為另一個資料庫引擎。
- Oracle SQL Developer – Oracle SQL Developer 是一種整合的開發環境 (IDE)，可簡化傳統和雲端部署中 Oracle 資料庫的開發和管理。

Code

`incremental-migration-sct-sql.zip` 檔案（已連接）包含此模式的完整原始碼。

史詩

建立來源資料庫結構描述差異的 SQL 指令碼檔案

任務	Description	所需的技能
在 Oracle SQL Developer 中執行資料庫差異。	1. 登入來源 Oracle 資料庫執行個體，選擇工具，然後選擇資料庫差異。 2. 在來源連線中選擇來源資料庫。 3. 在目的地連線中選擇更新或修補的來源資料庫。	DBA

任務	Description	所需的技能
	4. 根據您的需求設定其餘選項，選擇下一步，然後選擇完成以產生差異報告。	
產生 SQL 指令碼檔案。	選擇產生指令碼以產生 SQL 檔案中的差異。 這會產生 SQL 指令碼檔案，供 AWS SCT 用來將資料庫從 Oracle 轉換為 PostgreSQL。	DBA

使用 Python 指令碼在 AWS SCT 中建立目標資料庫物件

任務	Description	所需的技能
使用 Windows 命令提示字元設定 AWS SCT。	1. 從預先安裝的 AWS SCT 資料夾複製`AWSSchemaConversionToolBatch.jar` 檔案，並將其貼到您的工作目錄中。 2. 從 `incremental-migration-sct-sql.zip` 資料夾（已連接）從 `run_aws_sct_sql.py` 檔案部署 Python 程式碼。這會使用來源和目標資料庫環境組態詳細資訊，在 `projects`目錄中建立 .xml 檔案和 .sct 檔案。它也會讀取您在 Oracle SQL Developer 中產生的 SQL 指令碼檔案。最後，它會在 `output`目錄中建立 .sql 檔案物件。	DBA

任務	Description	所需的技能

	3. 使用下列格式在 `database_migration.txt` 檔案中設定來源和目標環境組態詳細資訊：
	
	`#source_vendor,source_hostname,source_dbname,source_user,source_pwd,source_schema,source_port,source_sid,target_vendor,target_hostname,target_user,target_pwd,target_dbname,target_port`
	
	`ORACLE,myoracledb.cokmvis0v46q.us-east-1.rds.amazonaws.com,ORCL,orcl,orcl1234,orcl,1521,ORCL,POSTGRESQL,mypgdbinstance.cokmvis0v46q.us-east-1.rds.amazonaws.com,pguser,pgpassword,pgdb,5432`
	
	4. 根據您的需求修改 AWS SCT 組態參數，然後將 SQL 指令碼檔案複製到`input`子目錄中的工作目錄。	

任務	Description	所需的技能
執行 Python 指令碼。	1. 使用下列命令執行 Python 指令碼：`$ python run_aws_sct_sql.py database_migration.txt` 2. 這會建立資料庫物件 SQL 檔案。具有轉換錯誤的非轉換程式碼可以手動轉換。	DBA
在 Amazon RDS for PostgreSQL 中建立物件	執行 SQL 檔案，並在 Amazon RDS for PostgreSQL 資料庫執行個體中建立物件。	DBA

相關資源

- Amazon RDS 上的 Oracle
- Amazon RDS 上的 PostgreSQL
- 使用 AWS SCT 使用者介面
- 使用 Oracle 做為 AWS SCT 的來源

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

在 Aurora PostgreSQL 相容中使用檔案編碼將 BLOB 檔案載入 TEXT

Bhanu Ganesh Gudivada 和 Jeevan Shetty，Amazon Web Services

總結

通常在遷移期間，在某些情況下，您必須處理從本機檔案系統的檔案載入的非結構化和結構化資料。資料也可能位於與資料庫字元集不同的字元集中。

這些檔案會保存以下類型的資料：

- 中繼資料 – 此資料說明檔案結構。
- 半結構化資料 – 這些是特定格式的文字字串，例如 JSON 或 XML。您可以對這類資料進行聲明，例如「一律以「 < 」」或「不包含任何換行字元」。
- 全文 – 此資料通常包含所有類型的字元，包括換行字元和引號字元。它也可能由 UTF-8 中的多位元組字元組成。
- 二進位資料 – 此資料可能包含位元組或位元組組合，包括 null 和end-of-file標記。

載入這些資料類型的混合可能是一項挑戰。

此模式可與現場部署 Oracle 資料庫 、Amazon Web Services (AWS) 雲端上 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的 Oracle 資料庫，以及 Oracle 資料庫的 Amazon Relational Database Service (Amazon RDS) 搭配使用。例如，此模式使用 Amazon Aurora PostgreSQL 相容版本。

在 Oracle Database 中，透過 `BFILE`（二進位檔案） 指標、`DBMS_LOB`套件和 Oracle 系統函數的協助，您可以從檔案載入並使用字元編碼轉換為 CLOB。由於 PostgreSQL 在遷移至 Amazon Aurora PostgreSQL 相容版本資料庫時不支援 BLOB 資料類型，因此這些函數必須轉換為 PostgreSQL 相容指令碼。

此模式提供兩種方法，可將檔案載入 Amazon Aurora PostgreSQL 相容資料庫中的單一資料庫資料欄：

- 方法 1 – 您可以使用 `aws_s3`延伸的 `table_import_from_s3`函數搭配 編碼選項，從 Amazon Simple Storage Service (Amazon S3) 儲存貯體匯入資料。
- 方法 2 – 您在資料庫外部編碼為十六進位，然後解碼以在`TEXT`資料庫中檢視。

我們建議您使用方法 1，因為 Aurora PostgreSQL 相容與`aws_s3`延伸模組直接整合。

此模式使用範例，將包含多位元組字元和不同格式的電子郵件範本，載入 Amazon Aurora PostgreSQL 相容資料庫。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Amazon RDS 執行個體或 Aurora PostgreSQL 相容執行個體
- 對 SQL 和關聯式資料庫管理系統 (RDBMS) 的基本了解
- Amazon Simple Storage Service (Amazon S3) 儲存貯體。
- Oracle 和 PostgreSQL 中的系統函數知識
- RPM 套件 HexDump-XXD-0.1.1 （ 隨附於 Amazon Linux 2)

> **Note**
>
> Amazon Linux 2 即將終止支援。如需詳細資訊，請參閱 Amazon Linux 2 FAQs。

限制

- 對於 TEXT資料類型，可儲存的最長字元字串約為 1 GB。

產品版本

- Aurora 支援 Amazon Aurora PostgreSQL 更新中列出的 PostgreSQL 版本。 PostgreSQL

Architecture

目標技術堆疊

- Aurora PostgreSQL 相容

目標架構

方法 1 – 使用 aws_s3.table_import_from_s3

從內部部署伺服器，包含多位元組字元和自訂格式的電子郵件範本的檔案會傳輸至 Amazon S3。此模式提供的自訂資料庫函數使用 `aws_s3.table_import_from_s3`函數搭配 `file_encoding` 將檔案載入資料庫，並以TEXT資料類型傳回查詢結果。

1. 檔案會傳輸至預備 S3 儲存貯體。
2. 檔案會上傳至 Amazon Aurora PostgreSQL 相容資料庫。
3. 使用 pgAdmin 用戶端，自訂函數`load_file_into_clob`會部署到 Aurora 資料庫。
4. 自訂函數會在內部`table_import_from_s3`搭配 file_encoding 使用 。函數的輸出是透過使用 `array_to_string`和 `array_agg`做為TEXT輸出來取得。

方法 2 – 在資料庫外部編碼為十六進位，並解碼以檢視資料庫內的 TEXT

來自內部部署伺服器或本機檔案系統的檔案會轉換為十六進位傾印。然後，檔案會匯入 PostgreSQL 做為TEXT欄位。

1. 使用 `xxd -p`選項，將檔案轉換為命令列中的十六進位傾印。
2. 使用 `\copy`選項將十六進位傾印檔案上傳至 Aurora PostgreSQL 相容，然後將十六進位傾印檔案解碼為二進位檔案。
3. 將二進位資料編碼為傳回為 TEXT。

工具

AWS 服務

- Amazon Aurora PostgreSQL 相容版本是完全受管的 ACID 相容關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。

其他工具

- pgAdmin4 是 PostgreSQL 的開放原始碼管理和開發平台。pgAdmin4 可用於 Linux、Unix、mac OS 和 Windows 來管理 PostgreSQL。

史詩

方法 1：將資料從 Amazon S3 匯入 Aurora PostgreSQL 相容

任務	Description	所需的技能
啟動 EC2 執行個體。	如需啟動執行個體的指示，請參閱啟動執行個體。	DBA
安裝 PostgreSQL 用戶端 pgAdmin 工具。	下載並安裝 pgAdmin。	DBA
建立 IAM 政策。	建立名為 的 AWS Identity and Access Management (IAM) 政策`aurora-s3-access-pol` ，授予儲存檔案的 S3 儲存貯體存取權。使用下列程式碼，`<bucket-name>` 將 取代為 S3 儲存貯體的名稱。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "s3:GetObject", "s3:AbortMultipartUpload", "s3:DeleteObject", "s3:ListMultipartUploadParts", "s3:PutObject",`	DBA

任務	Description	所需的技能

	<pre>"s3:ListBucket"
	],
	"Resource":
	[
	
	"arn:aws:s3:::<buc
	ket-name>/*",
	
	"arn:aws:s3:::<buc
	ket-name>"
	]
	}
	]
	}</pre>	

建立 IAM 角色，以將物件從 Amazon S3 匯入 Aurora PostgreSQL 相容。	使用以下程式碼建立名為的 IAM 角色`aurora-s3-import-role` 與 AssumeRole 信任關係。 `AssumeRole` 允許 Aurora 代表您存取其他 AWS 服務。
	
	<pre>{
	"Version": "2012-10-
	17",
	"Statement": [
	{
	"Effect":
	"Allow","Principal":
	{
	"Service":
	"rds.amazonaws.com"
	},"Action":
	"sts:AssumeRole"
	}
	]
	}</pre>
		DBA

任務	Description	所需的技能
將 IAM 角色與叢集建立關聯。	若要將 IAM 角色與 Aurora PostgreSQL 相容資料庫叢集建立關聯，請執行下列 AWS CLI 命令。<Account-ID> 變更為託管 Aurora PostgreSQL 相容資料庫的 AWS 帳戶 ID。這可讓 Aurora PostgreSQL 相容資料庫存取 S3 儲存貯體。 `aws rds add-role-to-db-cluster --db-cluster-identifier aurora-postgres-cl --feature-name s3Import --role-arn arn:aws:iam::<Account-ID>:role/aurora-s3-import-role`	DBA
將範例上傳至 Amazon S3。	1. 在此模式的其他資訊區段中，將電子郵件範本程式碼複製到名為 的檔案`salary.event.notification.email.vm` 。 2. 上傳至 檔案至 S3 儲存貯體。	DBA、應用程式擁有者

任務	Description	所需的技能
部署自訂 函數。	1. 從其他資訊區段中，將自訂函數 `load_file_into_clob` SQL 檔案內容複製到暫時資料表。 2. 使用 pgAdmin 用戶端登入 Aurora PostgreSQL 相容資料庫，並將其部署至資料庫結構描述。	應用程式擁有者、DBA

執行自訂函數，將資料匯入資料庫。	執行下列 SQL 命令，將角括號中的項目取代為適當的值。
	`select load_file_into_clob('aws-s3-import-test'::text,'us-west-1'::text,'employee.salary.event.notification.email.vm'::text);`
	在執行 命令之前，將角括號中的項目取代為適當的值，如下列範例所示。
	`Select load_file_into_clob('aws-s3-import-test'::text,'us-west-1'::text,'employee.salary.event.notification.email.vm'::text);`
	命令會從 Amazon S3 載入檔案，並將輸出傳回為 TEXT。
		應用程式擁有者、DBA

方法 2：將範本檔案轉換為本機 Linux 系統中的十六進位傾印

任務	Description	所需的技能

將範本檔案轉換為十六進位傾印。	**Note**
	Hexdump 公用程式會以十六進位、小數、八進位或 ASCII 顯示二進位檔案的內容。hexdump 命令是 `util-linux` 套件的一部分，並預先安裝在 Linux 發行版本中。Hexdump RPM 套件也是 Amazon Linux 2 的一部分。(：Amazon Linux 2 即將結束支援。 如需詳細資訊，請參閱 Amazon Linux 2 FAQs。)
	
	若要將檔案內容轉換為十六進位傾印，請執行下列 shell 命令。
	`xxd -p </path/file.vm> | tr -d '\n' > </path/file.hex>`
	
	將路徑和檔案取代為適當的值，如下列範例所示。
	`xxd -p employee.salary.event.notification.email.vm | tr -d '\n' > employee.`
		DBA

任務	Description	所需的技能
	`salary.event.notification.email.vm.hex`	

任務	Description	所需的技能
將 hexdump 檔案載入資料庫結構描述。	使用下列命令將 hexdump 檔案載入 Aurora PostgreSQL 相容資料庫。 1. 登入 Aurora PostgreSQL 資料庫，並建立新的名為 的資料表`email_template_hex` 。 ``` CREATE TABLE email_template_hex(hex_data TEXT); ``` 2. 使用下列命令，將檔案從本機檔案系統載入資料庫結構描述。 ``` \copy email_template_hex FROM '/path/file.hex'; ``` 將路徑取代為本機檔案系統上的位置。 ``` \copy email_template_hex FROM '/tmp/employee.salary.event.notification.email.vm.hex'; ``` 3. 建立另一個名為 的資料表`email_template_bytea` 。 ``` CREATE TABLE email_template_bytea(hex_data bytea); ```	DBA

任務	Description	所需的技能

	4. 將資料從 email_template_hex 插入 email_template_bytea 。
	
	`INSERT INTO email_template_bytea (hex_data)`
	
	`(SELECT decode(hex_data, 'hex') FROM email_template_hex limit 1);`
	
	5. 若要將十六進位 bytea 程式碼傳回為TEXT資料，請執行下列命令。
	
	`SELECT encode(hex_data::bytea, 'escape') FROM email_template_bytea;`	

相關資源

參考

- 使用 PostgreSQL 資料庫做為 AWS Database Migration Service 的目標
- 具有 PostgreSQL 相容性 (12.4) 遷移手冊的 Oracle Database 19c 到 Amazon Aurora
- 建立 IAM 政策
- 將 IAM 角色與 Amazon Aurora MySQL 資料庫叢集建立關聯
- pgAdmin

教學課程

- Amazon RDS 入門
- 從 Oracle 遷移至 Amazon Aurora

其他資訊

load_file_into_clob 自訂函數

```
CREATE OR REPLACE FUNCTION load_file_into_clob(
    s3_bucket_name text,
    s3_bucket_region text,
    file_name text,
    file_delimiter character DEFAULT '&'::bpchar,
    file_encoding text DEFAULT 'UTF8'::text)
    RETURNS text
    LANGUAGE 'plpgsql'
    COST 100
    VOLATILE PARALLEL UNSAFE
AS $BODY$
DECLARE
    blob_data BYTEA;
    clob_data TEXT;
    l_table_name CHARACTER VARYING(50) := 'file_upload_hex';
    l_column_name CHARACTER VARYING(50) := 'template';
    l_return_text TEXT;
    l_option_text CHARACTER VARYING(150);
    l_sql_stmt CHARACTER VARYING(500);

BEGIN

    EXECUTE format ('CREATE TEMPORARY TABLE %I (%I text, id_serial serial)',
 l_table_name, l_column_name);

    l_sql_stmt := 'select ''(format text, delimiter ''''' || file_delimiter || ''''',
 encoding ''''' || file_encoding ||  ''''')'' ';

    EXECUTE FORMAT(l_sql_stmt)
    INTO l_option_text;

    EXECUTE FORMAT('SELECT aws_s3.table_import_from_s3($1,$2,$6,
 aws_commons.create_s3_uri($3,$4,$5))')
    INTO l_return_text
```

```
    USING l_table_name, l_column_name, s3_bucket_name,
 file_name,s3_bucket_region,l_option_text;

    EXECUTE format('select array_to_string(array_agg(%I order by id_serial),E''\n'')
 from %I', l_column_name, l_table_name)
    INTO clob_data;

    drop table file_upload_hex;

    RETURN clob_data;
END;
$BODY$;
```

電子郵件範本

```
##############################################################################################
##
 ##
##    johndoe Template Type: email
 ##
##    File: johndoe.salary.event.notification.email.vm
 ##
##    Author: Aimée Étienne    Date 1/10/2021
      ##
##  Purpose: Email template used by EmplmanagerEJB to inform a johndoe they         ##
##        have been given access to a salary event
 ##
##    Template Attributes:
 ##
##        invitedUser - PersonDetails object for the invited user
  ##
##        salaryEvent - OfferDetails object for the event the user was given access
 ##
##        buyercollege - CompDetails object for the college owning the salary event
 ##
##        salaryCoordinator - PersonDetails of the salary coordinator for the event
 ##
##        idp - Identity Provider of the email recipient
 ##
##        httpWebRoot - HTTP address of the server
 ##
```

```
##
 ##
################################################################################################

$!invitedUser.firstname $!invitedUser.lastname,

Ce courriel confirme que vous avez ete invite par $!salaryCoordinator.firstname $!
salaryCoordinator.lastname de $buyercollege.collegeName a participer a l'evenement
 "$salaryEvent.offeringtitle" sur johndoeMaster Sourcing Intelligence.

Votre nom d'utilisateur est $!invitedUser.username

Veuillez suivre le lien ci-dessous pour acceder a l'evenement.

${httpWebRoot}/myDashboard.do?idp=$!{idp}

Si vous avez oublie votre mot de passe, utilisez le lien "Mot de passe oublie" situe
 sur l'ecran de connexion et entrez votre nom d'utilisateur ci-dessus.

Si vous avez des questions ou des preoccupations, nous vous invitons a
 communiquer avec le coordonnateur de l'evenement $!salaryCoordinator.firstname $!
salaryCoordinator.lastname au ${salaryCoordinator.workphone}.

*******

johndoeMaster Sourcing Intelligence est une plateforme de soumission en ligne pour les
 equipements, les materiaux et les services.

Si vous avez des difficultes ou des questions, envoyez un courriel a
 support@johndoeMaster.com pour obtenir de l'aide.
```

使用 AWS SCT 和 AWS DMS 將 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL AWS CLI CloudFormation

Pinesh Singal，Amazon Web Services

總結

此模式說明如何使用 () 將 Oracle 資料庫執行個體的多 TB Amazon Relational Database Service (Amazon RDS) 遷移至 Amazon RDS for PostgreSQL 資料庫執行個體AWS CLI。 AWS Command Line Interface 此方法提供最短的停機時間，不需要登入 AWS 管理主控台。

此模式使用 AWS Schema Conversion Tool (AWS SCT) 和 AWS Database Migration Service (AWS DMS) 主控台，有助於避免手動組態和個別遷移。解決方案會為多個資料庫設定一次性組態，並在 AWS DMS 中使用 AWS SCT 和 執行遷移 AWS CLI。

模式使用 AWS SCT 將資料庫結構描述物件從 Amazon RDS for Oracle 轉換為 Amazon RDS for PostgreSQL，然後使用 AWS DMS 遷移資料。在 中使用 Python 指令碼 AWS CLI，您可以使用 CloudFormation 範本建立 AWS SCT 物件和 AWS DMS 任務。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 現有的 Amazon RDS for Oracle 資料庫執行個體。
- 現有的 Amazon RDS for PostgreSQL 資料庫執行個體。
- 執行指令碼的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體或具有 Windows 或 Linux 作業系統的本機電腦。
- 了解下列 AWS DMS 遷移任務類型：`full-load`、`cdc`、`full-load-and-cdc`。 如需詳細資訊，請參閱 AWS DMS 文件中的建立任務。
- AWS SCT，安裝並設定適用於 Oracle 和 PostgreSQL 資料庫引擎的 Java Database Connectivity (JDBC) 驅動程式。如需詳細資訊，請參閱 AWS SCT 文件中的安裝和設定 AWS SCT。
- 來自已安裝 AWS SCT 資料夾`AWSSchemaConversionToolBatch.jar`的檔案，複製到您的工作目錄。
- `cli-sct-dms-cft.zip` 檔案 （ 已連接 ），在您的工作目錄中下載並解壓縮。
- 最新的 AWS DMS 複寫執行個體引擎版本。如需詳細資訊，請參閱 AWS 支援 文件和AWS DMS 版本備註中的如何建立 AWS DMS 複寫執行個體。

- AWS CLI 第 2 版，安裝並設定您的存取金鑰 ID、私密存取金鑰，以及執行指令碼之 EC2 執行個體或作業系統的預設 AWS 區域 名稱。如需詳細資訊，請參閱 AWS CLI 文件中的安裝或更新至最新版本的 AWS CLI和設定 的設定 AWS CLI。
- 熟悉 CloudFormation 範本。如需詳細資訊，請參閱 CloudFormation 文件中的 CloudFormation 運作方式。
- Python 第 3 版，安裝在執行指令碼的 EC2 執行個體或作業系統上並進行設定。如需詳細資訊，請參閱 Python 文件。

限制

- 來源 Amazon RDS for Oracle 資料庫執行個體的最低需求為：
  - 適用於 Enterprise、Standard、Standard One 和 Standard Two 版本的 Oracle 版本 12c (12.1.0.2、12.2.0.1)、18c (18.0.0.0) 和 19c (19.0.0.0)。
  - 雖然 Amazon RDS 支援 Oracle 18c (18.0.0.0)，但此版本處於棄用路徑，因為 Oracle 在end-of-support日期後不再提供 18c 的修補程式。如需詳細資訊，請參閱 Amazon RDS 文件中的 Amazon RDS for Oracle。
  - 不再支援 Amazon RDS for Oracle 11g。
- 您目標 Amazon RDS for PostgreSQL 資料庫執行個體的最低需求為：
  - PostgreSQL 第 9 版 (9.5 和 9.6)、10.x、11.x、12.x 和 13.x

產品版本

- Amazon RDS for Oracle 資料庫執行個體 12.1.0.2 版及更新版本
- Amazon RDS for PostgreSQL 資料庫執行個體 11.5 版及更新版本
- AWS CLI 第 2 版
- 的最新版本 AWS SCT
- Python 3 的最新版本

Architecture

來源技術堆疊

- Amazon RDS for Oracle

目標技術堆疊

- Amazon RDS for PostgreSQL

來源和目標架構

下圖顯示使用 AWS DMS 和 Python 指令碼將 Amazon RDS for Oracle 資料庫執行個體遷移至 Amazon RDS for PostgreSQL 資料庫執行個體。

圖表顯示下列遷移工作流程：

1. Python 指令碼使用 AWS SCT 連線到來源和目標資料庫執行個體。
2. 使用者 AWS SCT 從 Python 指令碼開始，將 Oracle 程式碼轉換為 PostgreSQL 程式碼，並在目標資料庫執行個體上執行。
3. Python 指令碼會為來源和目標資料庫執行個體建立 AWS DMS 複寫任務。
4. 使用者部署 Python 指令碼來啟動 AWS DMS 任務，然後在資料遷移完成後停止任務。

自動化和擴展

您可以將參數和安全相關變更新增至 Python 指令碼，以提供其他功能，以自動化此遷移。

工具

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和 區域的生命週期中管理這些資源。此模式會使用 Python 指令碼將`.csv`輸入檔案轉換為`.json`輸入檔案。`.json` 檔案用於 AWS CLI 命令，以建立使用 Amazon Resource Name (ARNs)、遷移類型、任務設定和資料表映射建立多個 AWS DMS 複寫任務的 CloudFormation 堆疊。
- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端 或在雲端和內部部署設定的組合之間遷移。此模式使用 AWS DMS 在命令列上執行的 Python 指令碼來建立、啟動和停止任務，以及建立 CloudFormation 範本。
- AWS Schema Conversion Tool (AWS SCT) 透過自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，支援異質資料庫遷移。此模式需要安裝 AWS SCT 目錄中`AWSSchemaConversionToolBatch.jar`的檔案。

Code

cli-sct-dms-cft.zip 檔案 （已連接）包含此模式的完整原始碼。

史詩

在 中設定 AWS SCT 和建立資料庫物件 AWS CLI

任務	Description	所需的技能

AWS SCT 設定 從 執行 AWS CLI。	1. 使用下列格式在 database_migration.txt 檔案中設定來源和目標環境組態詳細資訊：
	`#source_vendor,source_hostname,source_dbname,source_user,source_pwd,source_schema,source_port,source_sid,target_vendor,target_hostname,target_user,target_pwd,target_dbname,target_port`
	`ORACLE,myoracledb.cokmvis0v46q.us-east-1.rds.amazonaws.com,ORCL,orcl,orcl1234,orcl,1521,ORCL,POSTGRESQL,mypgdbinstance.cokmvis0v46q.us-east-1.rds.amazonaws.com,pguser,pgpassword,pgdb,5432`
	2. 根據您的需求修改下列檔案中的 AWS SCT 組態參數：project_settings.xml 、
		DBA

任務	Description	所需的技能
	`Oracle_PG_Test_Batch.xml` 和 `ORACLE-ocl-to-POSTGRESQL.xml` 。	
執行 `run_aws_sct.py` Python 指令碼。	使用以下命令執行 `run_aws_sct.py` Python 指令碼： `$ python run_aws_sct.py database_migration.txt` Python 指令碼會將資料庫物件從 Oracle 轉換為 PostgreSQL，並以 PostgreSQL 格式建立 SQL 檔案。指令碼也會建立 PDF 檔案 `Database migration assessment report`，為您提供資料庫物件的詳細建議和轉換統計資料。	DBA
在 Amazon RDS for PostgreSQL 中建立物件。	1. AWS SCT視需要手動修改產生的 SQL 檔案。 2. 執行 SQL 檔案，並在 Amazon RDS for PostgreSQL 資料庫執行個體中建立物件。	DBA

使用 和 設定 AWS CLI 和建立 AWS DMS 任務 CloudFormation

任務	Description	所需的技能
建立 AWS DMS 複寫執行個體。	登入 AWS 管理主控台，開啟 AWS DMS 主控台，並建立根	DBA

任務	Description	所需的技能
	據您的需求設定的複寫執行個體。 如需詳細資訊，請參閱 AWS DMS 文件中的建立複寫執行個體和 AWS 支援 文件中的如何建立 AWS DMS 複寫執行個體。	
建立來源端點。	在 AWS DMS 主控台上，選擇端點，然後根據您的需求建立 Oracle 資料庫的來源端點。 **Note** 額外的連線屬性必須`numberDataTypeScale` 具有`-2`值。 如需詳細資訊，請參閱 AWS DMS 文件中的建立來源和目標端點。	DBA
建立目標端點。	在 AWS DMS 主控台上，選擇端點，然後根據您的需求建立 PostgreSQL 資料庫的目標端點。 如需詳細資訊，請參閱 AWS DMS 文件中的建立來源和目標端點。	DevOps 工程師

任務	Description	所需的技能

設定複 AWS DMS 寫詳細資訊以從 執行 AWS CLI。	使用下列格式，使用 AWS DMS 來源端點 ARN、目標端點 ARN 和複寫執行個體 ARN 設定 `dms-arn-list.txt` 檔案中的來源和目標端點和複寫詳細資訊：
	
	`#sourceARN,targetARN,repARN`
	`arn:aws:dms:us-east-1:123456789012:endpoint:EH7AINRUDZ5GOYIY6HVMXECMCQ`
	`arn:aws:dms:us-east-1:123456789012:endpoint:HHJVUV57N7O3CQF4PJZKGIOYY5`
	`arn:aws:dms:us-east-1:123456789012:rep:LL57N77AQQAHHJF4PJFHNEDZ5G`
		DBA

任務	Description	所需的技能
執行 dms-create-task.py Python 指令碼來建立 AWS DMS 任務。	1. 使用以下命令執行 dms-create-task.py Python 指令碼： `$ python dms-create-task.py database_migration.txt dms-arn-list.txt <cft-stack-name> <migration-type>` 其中： • database_migration.txt 是資料庫遷移文字檔案。 • dms-arn-list.txt 是 的 ARN 清單 AWS DMS。 • <cft-stack-name> 是使用者定義的 CloudFormation 堆疊名稱。 • <migration-type> 是 full-load 、cdc 或 full-load-and-cdc。 2. 根據您的遷移類型，您可以使用下列命令來建立三種類型的 AWS DMS 任務： • $ python dms-create-task.py database_migration.txt dms-arn-list.txt dms-cli-c	DBA

任務	Description	所需的技能
	ft-stack full-load • $ python dms-create-task.py database_migration.txt dms-arn-list.txt dms-cli-cft-stack cdc • $ python dms-create-task.py database_migration.txt dms-arn-list.txt dms-cli-cft-stack full-load-and-cdc	
確認 AWS DMS 任務已就緒。	在 AWS DMS 主控台上，檢查Ready狀態區段中的 AWS DMS 任務是否處於 狀態。	DBA

使用 啟動和停止 AWS DMS 任務 AWS CLI

任務	Description	所需的技能
啟動 AWS DMS 任務。	使用以下命令執行 dms-start-task.py Python 指令碼： $ python dms-start-task.py start '<cdc-start-datetime>'	DBA

任務	Description	所需的技能
	Note 開始日期和時間必須是 'DD-MON-YYYY' 或 'YYYY-MM-DDTHH:MI:SS' 格式（例如 '01-Dec-2019' 或 '2018-03-08T12:12:12' ）。 您可以在 AWS DMS 主控台 AWS DMS 的任務頁面上的資料表統計資料索引標籤中檢閱任務狀態。	

任務	Description	所需的技能
驗證資料。	1. 完全載入遷移完成後，任務會持續為 CDC 執行。 2. 當 CDC 完成或不再需要更多變更時，請檢閱並驗證 Oracle 和 PostgreSQL 資料庫中的遷移任務結果和資料。 您可以在 AWS DMS 主控台的任務頁面上的資料庫遷移任務的資料表統計資料索引標籤中檢查狀態和計數資料欄`Validation state` `Validation pending` `Validation failed` (`Validation suspended` 、、和 `Validation details`)，以驗證您的資料。 如需詳細資訊，請參閱 AWS DMS 文件中的AWS DMS 資料驗證。	DBA

任務	Description	所需的技能
停止 AWS DMS 任務。	使用下列命令執行 Python 指令碼： `$ python dms-start-task.py stop` **Note** AWS DMS 任務可能會停止`failed`狀態，視驗證狀態而定。如需詳細資訊，請參閱下一節。	DBA

疑難排解

問題	解決方案
AWS SCT 來源和目標測試連線失敗。	設定 JDBC 驅動程式版本和 VPC 安全群組傳入規則，以接受傳入流量。
來源或目標端點測試執行失敗。	檢查端點設定和複寫執行個體是否處於 `Available` 狀態。檢查端點連線狀態是否為 `Successful` 。 如需詳細資訊，請參閱 AWS 支援 文件中的如何對 AWS DMS 端點連線失敗進行疑難排解。
完全載入執行失敗。	檢查來源和目標資料庫是否有相符的資料類型和大小。 如需詳細資訊，請參閱 AWS DMS 文件中的 中的遷移任務疑難排解 AWS DMS。

問題	解決方案
您遇到驗證執行錯誤。	檢查資料表是否具有主索引鍵，因為未驗證非主索引鍵資料表。 如果資料表有主索引鍵和錯誤，請檢查來源端點中的額外連線屬性是否具有 `numberDataTypeScale=-2` 。 如需詳細資訊，請參閱 AWS DMS 文件中的使用 Oracle 做為來源、OracleSettings 和故障診斷時的端點設定 AWS DMS。 OracleSettings https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Validating.html#CHAP_Validating.Troubleshooting

相關資源

- 安裝和設定 AWS SCT
- 簡介 AWS DMS（影片）
- AWS CLI 和 PowerShell 的 CloudFormation 堆疊操作命令範例
- 導覽 的使用者介面 AWS SCT
- 使用 Oracle 資料庫做為 的來源 AWS DMS
- 使用 連線至 Oracle 資料庫 AWS SCT
- 使用 PostgreSQL 資料庫做為 的目標 AWS DMS
- 資料遷移的來源
- 資料遷移的目標
- cloudformation (AWS CLI 文件）
- create-stack (AWS CLI 文件）
- dms (AWS CLI 文件）

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS DMS，以 SSL 模式將 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL

Pinesh Singal，Amazon Web Services

總結

此模式提供將 Amazon Relational Database Service (Amazon RDS) for Oracle 資料庫執行個體遷移至 Amazon Web Services (AWS) 雲端上 Amazon RDS for PostgreSQL 資料庫的指引。為了加密資料庫之間的連線， 模式會在 Amazon RDS 和 AWS Database Migration Service (AWS DMS) 中使用憑證授權單位 (CA) 和 SSL 模式。

模式描述了線上遷移策略，對於具有大量交易的多 TB Oracle 來源資料庫而言，幾乎沒有停機時間。為了資料安全，模式會在傳輸資料時使用 SSL。

此模式使用 AWS Schema Conversion Tool (AWS SCT) 將 Amazon RDS for Oracle 資料庫結構描述轉換為 Amazon RDS for PostgreSQL 結構描述。然後， 模式會使用 AWS DMS 將資料從 Amazon RDS for Oracle 資料庫遷移至 Amazon RDS for PostgreSQL 資料庫。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 僅以 rds-ca-rsa2048-g1 設定的 Amazon RDS 資料庫憑證授權單位 (CA)
  - rds-ca-2019 憑證已於 2024 年 8 月過期。
  - rds-ca-2015 憑證已於 2020 年 3 月 5 日過期。
- AWS SCT
- AWS DMS
- pgAdmin
- SQL 工具 （ 例如 SQL Developer 或 SQL*Plus)

限制

- Amazon RDS for Oracle 資料庫 – 企業版和標準二版的 Oracle 19c 最低需求。
- Amazon RDS for PostgreSQL 資料庫 – 最低需求為 PostgreSQL 第 12 版及更新版本 （ 適用於 9.x 及更新版本 ）。

產品版本

- Amazon RDS for Oracle 資料庫版本 12.1.0.2 執行個體
- Amazon RDS for PostgreSQL 資料庫 11.5 版執行個體

Architecture

來源技術堆疊

- 版本為 12.1.0.2.v18 的 Amazon RDS for Oracle 資料庫執行個體。

目標技術堆疊

- AWS DMS
- 具有 11.5 版的 Amazon RDS for PostgreSQL 資料庫執行個體。

目標架構

下圖顯示 Oracle （來源） 和 PostgreSQL （目標） 資料庫之間資料遷移架構的架構。架構包含下列項目：

- 虛擬私有雲端 (VPC)
- 可用區域
- 私有子網路
- Amazon RDS for Oracle 資料庫
- AWS DMS 複寫執行個體
- RDS for PostgreSQL 資料庫

若要加密來源和目標資料庫的連線，必須在 Amazon RDS 和 AWS DMS 中啟用 CA 和 SSL 模式。

工具

AWS 服務

- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。

- 適用於 Oracle 的 Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展 Oracle 關聯式資料庫。
- 適用於 PostgreSQL 的 Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展 PostgreSQL 關聯式資料庫。
- AWS Schema Conversion Tool (AWS SCT) 會自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，以支援異質資料庫遷移。

其他服務

- pgAdmin 是 PostgreSQL 的開放原始碼管理工具。它提供圖形界面，可協助您建立、維護和使用資料庫物件。

最佳實務

Amazon RDS 提供新的憑證授權機構憑證，做為 AWS 安全最佳實務。如需新憑證和支援之 AWS 區域的相關資訊，請參閱使用 SSL/TLS 加密與資料庫執行個體或叢集的連線。

如果您的 RDS 執行個體目前在 CA 憑證 上`rds-ca-2019`，且您想要升級至 `rds-ca-rsa2048-g1`，請遵循透過修改資料庫執行個體或叢集來更新 CA 憑證，或透過套用維護來更新 CA 憑證中的指示。

史詩

設定 Amazon RDS for Oracle 執行個體

任務	Description	所需的技能
建立 Oracle 資料庫執行個體。	登入您的 AWS 帳戶，開啟 AWS 管理主控台，然後導覽至 Amazon RDS 主控台。在 主控台上，選擇建立資料庫，然後選擇 Oracle。	一般 AWS、DBA
設定安全群組。	設定傳入和傳出安全群組。	一般 AWS
建立選項群組。	在與 Amazon RDS for Oracle 資料庫相同的 VPC 和安全群組中建立選項群組。在選項中，	一般 AWS

任務	Description	所需的技能
	選擇 SSL。針對連接埠，選擇 2484 （針對 SSL 連線）。	
設定 選項設定。	請使用下列設定： • SQLNET.CIPHER_SUITE ：SSL_RSA_WITH_AES_256_CBC_SHA • SQLNET.SSL_VERSION ：1.2 or 1.0	一般 AWS
修改 RDS for Oracle 資料庫執行個體。	將 CA 憑證設定為 rds-ca-rsa2048-g1。 在選項群組下，連接先前建立的選項群組。	DBA、一般 AWS

任務	Description	所需的技能

確認 RDS for Oracle 資料庫執行個體可用。	確定 Amazon RDS for Oracle 資料庫執行個體已啟動並執行，而且可存取資料庫結構描述。
	
	若要連線至 RDS for Oracle 資料庫，請使用`sqlplus`命令列中的 命令。
	
	`$ sqlplus orcl/**** @myoracledb.cokmvis0v46q.us-east-1.rds.amazonaws.com:1521/ORCL`
	`SQL*Plus: Release 12.1.0.2.0 Production on Tue Oct 15 18:11:07 2019`
	`Copyright (c) 1982, 2016, Oracle. All rights reserved.`
	`Last Successful login time: Mon Dec 16 2019 23:17:31 +05:30`
	`Connected to:`
	`Oracle Database 12c Enterprise Edition Release 12.1.0.2.0 - 64bit Production`
	`With the Partitioning, OLAP, Advanced Analytics and Real Application Testing options`
	`SQL>`
		DBA

在 RDS for Oracle 資料庫中建立物件和資料。	在結構描述中建立物件並插入資料。	DBA

設定 Amazon RDS for PostgreSQL 執行個體

任務	Description	所需的技能
建立 RDS for PostgreSQL 資料庫。	在 Amazon RDS 主控台建立資料庫頁面上，選擇 PostgreSQL 以建立 Amazon RDS for PostgreSQL 資料庫執行個體。	DBA、一般 AWS
設定安全群組。	設定傳入和傳出安全群組。	一般 AWS
建立參數群組。	如果您使用的是 PostgreSQL 11.x 版，請建立參數群組來設定 SSL 參數。在 PostgreSQL 第 12 版中，預設會啟用 SSL 參數群組。	一般 AWS
編輯參數。	將 `rds.force_ssl` 參數變更為 1（開啟）。 根據預設，`ssl` 參數為 1（開啟）。透過將 `rds.force_ssl` 參數設定為 1，您可以強制所有連線僅透過 SSL 模式進行連線。	一般 AWS
修改 RDS for PostgreSQL 資料庫執行個體。	將 CA 憑證設定為 rds-ca-rsa2048-g1。 根據您的 PostgreSQL 版本，連接預設參數群組或先前建立的參數群組。	DBA、一般 AWS
確認 RDS for PostgreSQL 資料庫執行個體可用。	確定 Amazon RDS for PostgreSQL 資料庫已啟動並執行。	DBA

任務	Description	所需的技能

	psql 命令會從命令列使用 sslmode集合 建立 SSL 連線。
	
	其中一個選項是在 參數群組sslmode=1 中設定 ，並使用 psql 連線，而不在 命令中包含 sslmode 參數。
	
	下列輸出顯示已建立 SSL 連線。
	
	`$ psql -h mypgdbinstance.cokmvis0v46q.us-east-1.rds.amazonaws.com -p 5432 "dbname=pgdb user=pguser"`
	`Password for user pguser:`
	`psql (11.3, server 11.5)`
	`SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256, compression: off)`
	`Type "help" for help.`
	`pgdb=>`
	
	第二個選項是在參數群組sslmode=1 中設定 ，並在psql命令中包含 sslmode 參數。
	
	下列輸出顯示已建立 SSL 連線。	

任務	Description	所需的技能
	`$ psql -h mypgdbinstance.cokmvis0v46q.us-east-1.rds.amazonaws.com -p 5432 "dbname=pgdb user=pguser sslmode=require" Password for user pguser: psql (11.3, server 11.5) SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256, compression: off) Type "help" for help. pgdb=>`	

設定和執行 AWS SCT

任務	Description	所需的技能
安裝 AWS SCT。	安裝最新版本的 AWS SCT 應用程式。	一般 AWS
使用 JDBC 驅動程式設定 AWS SCT。	下載適用於 Oracle (ojdbc8.jar) 和 PostgreSQL (postgresql-42.2.5.jar) 的 Java Database Connectivity (JDBC) 驅動程式。 若要在 AWS SCT 中設定驅動程式，請選擇設定、全域設定、驅動程式。	一般 AWS
建立 AWS SCT 專案。	使用 Oracle 作為來源資料庫引擎和 Amazon RDS for	一般 AWS

任務	Description	所需的技能
	PostgreSQL 作為目標資料庫引擎，建立 AWS SCT 專案和報告： 1. 透過提供連線詳細資訊，測試來源 Oracle 資料庫和目標 Amazon RDS for PostgreSQL 資料庫的連線。 對於來源 Oracle 資料庫，需要下列許可或權限： • `CONNECT` • `SELECT_CATALOG_ROLE` • `SELECT ANY DICTIONARY` • `SELECT on SYS.USER$ TO <sct_user>` 如需詳細資訊，請參閱使用 Oracle 資料庫做為 AWS SCT 的來源。 來源和目標連線都必須成功，AWS SCT 才能啟動遷移報告。 2. 在報告之後，輸入要轉換的結構描述，然後選擇完成。	

任務	Description	所需的技能
驗證資料庫物件。	1. 選擇載入結構描述。 AWS SCT 會顯示來源和轉換後的目標物件，包括發生錯誤的物件。更新目標資料庫上任何不正確的物件。 2. 檢閱錯誤，並使用手動介入清除錯誤。 3. 清除所有錯誤後，再次選擇載入結構描述。 4. 選擇套用至資料庫。 5. 連線至 pgAdmin 或任何支援 PostgreSQL 資料庫連線的工具，並檢查結構描述和物件。	DBA、一般 AWS

設定和執行 AWS DMS

任務	Description	所需的技能
建立複寫執行個體。	1. 登入您的帳戶，開啟 AWS 管理主控台，然後導覽至 AWS DMS 主控台。 2. 使用 VPC、安全群組、可用區域和額外連線屬性的有效設定建立複寫執行個體。	一般 AWS
匯入憑證。	為您的 AWS 區域下載憑證套件 (PEM)。 套件同時包含`rds-ca-2019` 中繼憑證和根憑證。套件也包含 `rds-ca-rsa2048-g1`、 `rds-ca-rsa4096-`	一般 AWS

任務	Description	所需的技能
	g1和rds-ca-ecc384-g1 根 CA 憑證。您的應用程式信任存放區只需要註冊根 CA 憑證。	
建立來源端點。	1. 選擇選取 RDS 資料庫執行個體，然後選擇您建立的 RDS for Oracle 資料庫執行個體，為 Amazon RDS for Oracle 建立來源端點。會自動填入端點組態詳細資訊。 2. 選擇手動提供存取資訊。針對連接埠，請確定您輸入 2484。 3. 在 Secure Socket Layer (SSL) 模式下，選擇 verify-ca ，然後選擇您先前建立的 CA 憑證。 4. 在端點設定下，新增額外的連線屬性NumberDataTypeScale=-2 ，以支援沒有大小的NUMBER資料類型。 如需詳細資訊，請參閱使用 Oracle 資料庫做為 AWS Database Migration Service 的來源。	一般 AWS

任務	Description	所需的技能
建立目標端點。	1. 選擇選取 RDS 資料庫執行個體，然後選擇 RDS for PostgreSQL 資料庫執行個體，為 Amazon RDS for PostgreSQL 建立目標端點。會自動填入端點組態詳細資訊。 2. 選擇手動提供存取資訊。針對連接埠，請確定您輸入 2484。 如需詳細資訊，請參閱使用 PostgreSQL 資料庫做為 AWS Database Migration Service 的目標。	一般 AWS
測試端點。	1. 測試來源和目標端點，確認兩者都成功且可用。 2. 如果測試失敗，請確定安全群組傳入規則有效。	一般 AWS

任務	Description	所需的技能
建立遷移任務。	若要為完全載入和變更資料擷取 (CDC) 或資料驗證建立遷移任務，請執行下列動作： 1. 若要建立資料庫遷移任務，請選擇複寫執行個體、來源資料庫端點、目標資料庫端點。指定遷移類型為下列其中一項： • 遷移現有資料（完全載入） • 僅複寫資料變更 (CDC) • 遷移現有資料並複寫持續變更（完全載入和 CDC) 2. 在資料表映射下，您可以設定 GUI 或 JSON 格式的選擇規則和轉換規則： • 在選取規則下，選取結構描述，輸入資料表名稱，然後選取要設定的動作（包含或排除）；例如，結構描述 ORCL、資料表名稱 %、動作包含。 • 在轉換規則下，執行下列其中一項： • 選取結構描述，然後選擇動作（大小寫、字首、尾碼）；例如，目標結構描述 ORCL、動作製作小寫。 • 選取結構描述，輸入資料表名稱，然後選擇動作（大小寫、字首、	一般 AWS

任務	Description	所需的技能
	尾碼）；例如，目標結構描述 ORCL、資料表 %、動作製作小寫。 3. 開啟 Amazon CloudWatch Logs 監控。 4. 針對映射規則，新增下列 JSON 程式碼。 `{ "rules": [ { "rule-type": "transformation", "rule-id": "1", "rule-name": "1", "rule-target": "table", "object-locator": { "schema-name": "%", "table-name": "%" }, "rule-action": "convert-lowercase", "value": null, "old-value": null }, { "rule-type": "transformation",`	

任務	Description	所需的技能
	"rule-id" : "2", "rule-name": "2", "rule-target": "schema", "object-locator": { "schema-name": "ORCL", "table-name": "%" }, "rule-action": "convert-lowercase", "value": null, "old-value": null }, { "rule-type": "selection", "rule-id" : "3", "rule-name": "3", "object-locator": { "schema-name": "ORCL", "table-name": "DEPT" }, "rule-action": "include", "filters" : [] }	

任務	Description	所需的技能
	] }	
規劃生產執行。	與應用程式擁有者等利益相關者確認停機時間，以在生產系統中執行 AWS DMS。	遷移潛在客戶

任務	Description	所需的技能

執行 遷移任務。	1. 啟動狀態為就緒的 AWS DMS 任務，並監控 Amazon CloudWatch 中的遷移任務日誌是否有任何錯誤。
	
	如果您選擇遷移現有資料，並將持續變更複寫為遷移類型，且狀態為載入完成持續複寫，則會完成具有 CDC 資料遷移的完整載入，且驗證正在進行中。
	2. 開始遷移後，您可以在 CloudWatch 中取得其他 SSL 連線資訊。對於 Oracle，CloudWatch 會顯示下列連線字串。
	
	`2019-12-17T09:15:11 [SOURCE_UNLOAD ]I: Connecting to Oracle: Beginning session (oracle_endpoint_conn.c:834)`
	
	PostgreSQL 連線字串將類似於下列範例。
	
	`2019-12-17T09:15:11 [TARGET_LOAD ]I: Going to connect to ODBC connection string: PROTOCOL=7.4-0;DRIVER={PostgreSQL};SERVER=myp`
		一般 AWS

任務	Description	所需的技能
	`gdbinstance.cokmvis0v46q.us-east-1.rds.amazonaws.com;DATABASE=pgdb;PORT=5432;sslmode=require;UID=pguser;(odbc_endpoint_imp.c:2218)`	
驗證資料。	檢閱來源 Oracle 和目標 PostgreSQL 資料庫中的遷移任務結果和資料： 1. 連線至 pgAdmin，並使用結構描述 檢查 PostgreSQL 資料庫中的資料ORCL。 2. 對於 CDC，透過在來源 Oracle 資料庫中插入或更新資料來檢查進行中的變更。	DBA
停止遷移任務。	成功完成資料驗證後，請停止遷移任務。	一般 AWS

清除資源

任務	Description	所需的技能
刪除 AWS DMS 任務。	1. 在 AWS DMS 主控台上，導覽至資料庫遷移任務，然後停止任何進行中或執行中的 AWS DMS 任務。 2. 選取任務，選擇動作，然後選擇刪除。	一般 AWS

任務	Description	所需的技能
刪除 AWS DMS 端點。	選取您建立的來源和目標端點，選擇動作，然後選擇刪除。	一般 AWS
刪除 AWS DMS 複寫執行個體。	選擇複寫執行個體，選擇動作，然後選擇刪除。	一般 AWS
刪除 PostgreSQL 資料庫。	1. 在 Amazon RDS 主控台上，選擇資料庫。 2. 選取您建立的 PostgreSQL 資料庫執行個體，選擇動作，然後選擇刪除。	一般 AWS
刪除 Oracle 資料庫。	在 Amazon RDS 主控台上，選取 Oracle 資料庫執行個體，選擇動作，然後選擇刪除。	一般 AWS

疑難排解

問題	解決方案
AWS SCT 來源和目標測試連線失敗。	設定 JDBC 驅動程式版本和 VPC 安全群組傳入規則，以接受傳入流量。
Oracle 來源端點測試執行失敗。	檢查端點設定以及複寫執行個體是否可用。
AWS DMS 任務完全載入執行失敗。	檢查來源和目標資料庫是否具有相符的資料類型和大小。
AWS DMS 驗證遷移任務會傳回錯誤。	1. 檢查資料表是否有主索引鍵。沒有主索引鍵的資料表不會進行驗證。 2. 如果資料表有主索引鍵但傳回錯誤，請檢查來源端點中的額外連線屬性。額外的連線屬性必須根據資料表中可用的NUMBER資料，動

問題	解決方案
	態numberDataTypeScale=-2 支援沒有大小的資料類型。

相關資源

資料庫

- Amazon RDS for Oracle
- Amazon RDS for PostgreSQL

SSL 資料庫連線

- 使用 SSL/TLS 加密與資料庫執行個體的連線
  - 搭配 RDS for Oracle 資料庫執行個體使用 SSL
  - 使用 SSL/TLS 保護 RDS for PostgreSQL 的連線
  - 下載特定 AWS 區域的憑證套件
    - 下載 CA-2019 根憑證（已於 2024 年 8 月過期）
- 使用選項群組
  - 將選項新增至 Oracle 資料庫執行個體
  - Oracle Secure Sockets Layer
- 使用參數群組
- PostgreSQL sslmode 連線參數
- 從 JDBC 使用 SSL
- 輪換您的 SSL/TLS 憑證
  - 透過修改資料庫執行個體或叢集來更新您的 CA 憑證
  - 套用維護來更新您的 CA 憑證

AWS SCT

- AWS Schema Conversion Tool
- AWS Schema Conversion Tool使用者指南
- 使用 AWS SCT 使用者介面

- 使用 Oracle 資料庫做為 AWS SCT 的來源

AWS DMS

- AWS Database Migration Service
- AWS Database Migration Service 使用者指南
  - 使用 Oracle 資料庫做為 AWS DMS 的來源
  - 使用 PostgreSQL 資料庫做為 AWS DMS 的目標
- 搭配 AWS Database Migration Service 使用 SSL
- 將執行關聯式資料庫的應用程式遷移至 AWS

其他資訊

Amazon RDS Certificate Authority 憑證已於 2024 年 8 月`rds-ca-2019`過期。如果您使用或計劃使用 SSL 或 TLS 搭配憑證驗證來連線至 RDS 資料庫執行個體或多可用區域資料庫叢集，請考慮使用其中一個新的 CA 憑證：`rds-ca-rsa2048-g1`、 `rds-ca-rsa4096-g1`或 `rds-ca-ecc384-g1`。

將 Oracle SERIALLY_REUSABLE pragma 套件遷移至 PostgreSQL

Vinay Paladi，Amazon Web Services

總結

此模式提供step-by-step方法，將定義為 SERIALLY_REUSABLE pragma 的 Oracle 套件遷移至 Amazon Web Services (AWS) 上的 PostgreSQL。此方法會維護 SERIALLY_REUSABLE pragma 的功能。

PostgreSQL 不支援套件和 SERIALLY_REUSABLE pragma 的概念。若要在 PostgreSQL 中取得類似的功能，您可以為套件建立結構描述，並在結構描述內部署所有相關物件（例如函數、程序和類型）。為了實現 SERIALLY_REUSABLE pragma 的功能，此模式中提供的範例包裝函式指令碼使用 AWS Schema Conversion Tool (AWS SCT) 延伸套件。

如需詳細資訊，請參閱 Oracle 文件中的 SERIALLY_REUSABLE Pragma。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 最新版本的 AWS SCT 和必要的驅動程式
- Amazon Aurora PostgreSQL 相容版本資料庫或 Amazon Relational Database Service (Amazon RDS) for PostgreSQL 資料庫

產品版本

- Oracle 資料庫 10g 版及更新版本

Architecture

來源技術堆疊

- 內部部署的 Oracle 資料庫

目標技術堆疊

- Aurora PostgreSQL 相容或 Amazon RDS for PostgreSQL

- AWS SCT

遷移架構

工具

AWS 服務

- AWS Schema Conversion Tool (AWS SCT) 會自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，以支援異質資料庫遷移。
- Amazon Aurora PostgreSQL 相容版本是完全受管且符合 ACID 規範的關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。
- 適用於 PostgreSQL 的 Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展 PostgreSQL 關聯式資料庫。

其他工具

- pgAdmin 是 PostgreSQL 的開放原始碼管理工具。它提供圖形界面，可協助您建立、維護和使用資料庫物件。

史詩

使用 AWS SCT 遷移 Oracle 套件

任務	Description	所需的技能
設定 AWS SCT。	設定來源資料庫的 AWS SCT 連線。如需詳細資訊，請參閱使用 Oracle 資料庫做為 AWS SCT 的來源。	DBA、開發人員
轉換指令碼。	使用 AWS SCT 將目標資料庫選取為 Aurora PostgreSQL 相容，以轉換 Oracle 套件。	DBA、開發人員
儲存 .sql 檔案。	儲存 .sql 檔案之前，請將 AWS SCT 中的專案設定選項修改	DBA、開發人員

任務	Description	所需的技能
	為每個階段的單一檔案。AWS SCT 會根據物件類型，將 .sql 檔案分成多個 .sql 檔案。	
變更程式碼。	開啟 AWS SCT 產生的`init`函數，並如其他資訊區段中的範例所示進行變更。它會新增變數來實現功能 `pg_serialize = 0` 。	DBA、開發人員
測試轉換。	將`init`函數部署至 Aurora PostgreSQL 相容資料庫，並測試結果。	DBA、開發人員

相關資源

- AWS Schema Conversion Tool
- Amazon RDS
- Amazon Aurora 功能
- SERIALLY_REUSABLE Pragma

其他資訊

```
Source Oracle Code:

CREATE OR REPLACE PACKAGE test_pkg_var
IS
PRAGMA SERIALLY_REUSABLE;
PROCEDURE function_1
 (test_id number);
PROCEDURE function_2
 (test_id number
 );
END;

CREATE OR REPLACE PACKAGE BODY test_pkg_var
```

```
IS
PRAGMA SERIALLY_REUSABLE;
v_char VARCHAR2(20) := 'shared.airline';
v_num number := 123;

PROCEDURE function_1(test_id number)
IS
begin
dbms_output.put_line( 'v_char-'|| v_char);
dbms_output.put_line( 'v_num-'||v_num);
v_char:='test1';
function_2(0);
END;

PROCEDURE function_2(test_id number)
is
begin
dbms_output.put_line( 'v_char-'|| v_char);
dbms_output.put_line( 'v_num-'||v_num);
END;
END test_pkg_var;

Calling the above functions

set serveroutput on


EXEC test_pkg_var.function_1(1);


EXEC test_pkg_var.function_2(1);


Target Postgresql Code:


CREATE SCHEMA test_pkg_var;

CREATE OR REPLACE FUNCTION test_pkg_var.init(pg_serialize IN INTEGER DEFAULT 0)

RETURNS void
AS
$BODY$
```

```
DECLARE

BEGIN

if aws_oracle_ext.is_package_initialized( 'test_pkg_var' ) AND pg_serialize = 0

then

return;

end if;

PERFORM aws_oracle_ext.set_package_initialized( 'test_pkg_var' );

PERFORM aws_oracle_ext.set_package_variable( 'test_pkg_var', 'v_char',
 'shared.airline.basecurrency'::CHARACTER

VARYING(100));

PERFORM aws_oracle_ext.set_package_variable('test_pkg_var', 'v_num', 123::integer);

END;

$BODY$

LANGUAGE plpgsql;


CREATE OR REPLACE FUNCTION test_pkg_var.function_1(pg_serialize int default 1)

RETURNS void
AS

$BODY$
DECLARE

BEGIN

PERFORM test_pkg_var.init(pg_serialize);

raise notice 'v_char%',aws_oracle_ext.get_package_variable( 'test_pkg_var', 'v_char');

raise notice 'v_num%',aws_oracle_ext.get_package_variable( 'test_pkg_var', 'v_num');
```

```
PERFORM aws_oracle_ext.set_package_variable( 'test_pkg_var', 'v_char',
 'test1'::varchar);

PERFORM test_pkg_var.function_2(0);
END;

$BODY$
LANGUAGE plpgsql;


CREATE OR REPLACE FUNCTION test_pkg_var.function_2(IN pg_serialize integer default 1)

RETURNS void

AS

$BODY$

DECLARE

BEGIN

PERFORM test_pkg_var.init(pg_serialize);

raise notice 'v_char%',aws_oracle_ext.get_package_variable( 'test_pkg_var', 'v_char');

raise notice 'v_num%',aws_oracle_ext.get_package_variable( 'test_pkg_var', 'v_num');

END;
$BODY$
LANGUAGE plpgsql;


Calling the above functions

select test_pkg_var.function_1()

 select test_pkg_var.function_2()
```

將 Oracle 外部資料表遷移至 Amazon Aurora PostgreSQL 相容

anuradha chintha 和 Rakesh Raghav，Amazon Web Services

總結

外部資料表可讓 Oracle 查詢以一般檔案存放在資料庫外部的資料。您可以使用 ORACLE_LOADER 驅動程式來存取以 SQL*Loader 公用程式可載入的任何格式儲存的任何資料。您無法在外部資料表上使用資料處理語言 (DML)，但您可以使用外部資料表進行查詢、聯結和排序操作。

Amazon Aurora PostgreSQL 相容版本不提供類似於 Oracle 中外部資料表的功能。反之，您必須使用現代化來開發符合功能需求的可擴展解決方案，而且是無法的。

此模式提供使用 aws_s3擴充功能，將不同類型的 Oracle 外部資料表遷移至 Amazon Web Services (AWS) 雲端上的 Aurora PostgreSQL 相容版本的步驟。

我們建議在生產環境中實作此解決方案之前，先徹底測試此解決方案。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- AWS 命令列界面 (AWS CLI)
- 可用的 Aurora PostgreSQL 相容資料庫執行個體。
- 具有外部資料表的現場部署 Oracle 資料庫
- pg.Client API
- 資料檔案

限制

- 此模式不提供可取代 Oracle 外部資料表的功能。不過，您可以進一步增強步驟和範本程式碼，以實現資料庫現代化目標。
- 檔案不應包含在aws_s3匯出和匯入函數中以分隔符號傳遞的字元。

產品版本

- 若要從 Amazon S3 匯入 RDS for PostgreSQL，資料庫必須執行 PostgreSQL 10.7 版或更新版本。

Architecture

來源技術堆疊

- Oracle

來源架構

目標技術堆疊

- Amazon Aurora PostgreSQL 相容
- Amazon CloudWatch
- AWS Lambda
- AWS Secrets Manager
- Amazon Simple Notification Service (Amazon SNS)
- Amazon Simple Storage Service (Amazon S3)

目標架構

下圖顯示解決方案的高階表示法。

1. 檔案會上傳至 S3 儲存貯體。
2. Lambda 函數已啟動。
3. Lambda 函數會啟動資料庫函數呼叫。
4. Secrets Manager 提供資料庫存取的登入資料。
5. 根據資料庫函數，會建立 SNS 警示。

自動化和擴展

外部資料表的任何新增或變更都可以使用中繼資料維護來處理。

工具

- Amazon Aurora PostgreSQL 相容 – Amazon Aurora PostgreSQL 相容版本是全受管、PostgreSQL 相容且 ACID 相容的關係資料庫引擎，結合了高階商業資料庫的速度和可靠性，以及開放原始碼資料庫的成本效益。
- AWS CLI – AWS Command Line Interface (AWS CLI) 是管理 AWS 服務的統一工具。只需下載和設定一個工具，您就可以從命令列控制多個 AWS 服務，並透過指令碼將其自動化。
- Amazon CloudWatch – Amazon CloudWatch 會監控 Amazon S3 資源和使用率。
- AWS Lambda – AWS Lambda 是一種無伺服器運算服務，支援執行程式碼，無需佈建或管理伺服器、建立工作負載感知叢集擴展邏輯、維護事件整合，或管理執行時間。在此模式中，每當檔案上傳至 Amazon S3 時，Lambda 都會執行資料庫函數。
- AWS Secrets Manager – AWS Secrets Manager 是一種用於憑證儲存和擷取的服務。使用 Secrets Manager，您可以使用以程式設計方式呼叫 Secrets Manager 擷取秘密的 API，取代程式碼中的硬式編碼登入資料，包括密碼。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 提供儲存層來接收和存放檔案，以供取用和往返 Aurora PostgreSQL 相容叢集傳輸。
- aws_s3 – `aws_s3`延伸模組整合 Amazon S3 和 Aurora PostgreSQL 相容。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 會協調和管理發佈者和用戶端之間的訊息傳遞或傳送。在此模式中，Amazon SNS 用於傳送通知。

Code

每當檔案放入 S3 儲存貯體時，都必須從處理應用程式或 Lambda 函數建立和呼叫資料庫函數。如需詳細資訊，請參閱 程式碼 （已連接）。

史詩

建立外部檔案

任務	Description	所需的技能
將外部檔案新增至來源資料庫。	建立外部檔案，並將其移至`oracle`目錄。	DBA

設定目標 (Aurora PostgreSQL 相容)

任務	Description	所需的技能
建立 Aurora PostgreSQL 資料庫。	在 Amazon Aurora PostgreSQL 相容叢集中建立資料庫執行個體。	DBA
建立結構描述、aws_s3 延伸模組和資料表。	在其他資訊區段`ext_tbl_scripts` 中使用 下的程式碼。資料表包括實際資料表、預備資料表、錯誤和日誌資料表，以及可轉移。	DBA、開發人員
建立 資料庫函數。	若要建立資料庫函數，請使用其他資訊區段中函數下的`load_external_table_latest` 程式碼。	DBA、開發人員

建立並設定 Lambda 函式

任務	Description	所需的技能
建立角色。	建立具有存取 Amazon S3 和 Amazon Relational Database Service (Amazon RDS) 許可的角色。此角色將指派給 Lambda 以執行模式。	DBA
建立 Lambda 函數。	建立從 Amazon S3 讀取檔案名稱的 Lambda 函數 （例如`file_key = info.get('object', {}).get('key')`），並使用檔案名稱做為輸入參數來呼叫資料庫函數 （例如 `curs.callproc("load_externa`	DBA

任務	Description	所需的技能
	l_tables", [file_key]))。 根據函數呼叫結果，將會啟動 SNS 通知（例如 client.publish(TopicArn='arn:',Message='fileloadsuccess',Subject='fileloadsuccess'))。 根據您的業務需求，您可以視需要建立具有額外程式碼的 Lambda 函數。如需詳細資訊，請參閱 Lambda 文件。	
設定 S3 儲存貯體事件觸發。	設定機制，針對 S3 儲存貯體中的所有物件建立事件呼叫 Lambda 函數。	DBA
建立秘密。	使用 Secrets Manager 建立資料庫登入資料的秘密名稱。在 Lambda 函數中傳遞秘密。	DBA
上傳 Lambda 支援檔案。	上傳 .zip 檔案，其中包含 Lambda 支援套件和連接的 Python 指令碼，以連線至 Aurora PostgreSQL 相容。Python 程式碼會呼叫您在資料庫中建立的 函數。	DBA
建立 SNS 主題。	建立 SNS 主題以傳送郵件，確保資料載入成功或失敗。	DBA

新增與 Amazon S3 的整合

任務	Description	所需的技能
建立 S3 儲存貯體。	在 Amazon S3 主控台上，使用不包含正斜線的唯一名稱建立 S3 儲存貯體。S3 儲存貯體名稱全域唯一，且命名空間由所有 AWS 帳戶共用。	DBA
建立 IAM 政策。	若要建立 AWS Identity and Access Management (IAM) 政策，請使用其他資訊區段`s3bucketpolicy_for_import` 中 下的程式碼。	DBA
建立角色。	為 Aurora PostgreSQL 相容建立兩個角色，一個用於匯入的角色，另一個用於匯出的角色。將對應的政策指派給角色。	DBA
將角色連接至 Aurora PostgreSQL 相容叢集。	在管理角色下，將匯入和匯出角色連接至 Aurora PostgreSQL 叢集。	DBA
為 Aurora PostgreSQL 相容建立支援物件。	對於資料表指令碼，請在其他資訊區段`ext_tbl_scripts` 中使用 下的程式碼。 對於自訂函數，請在其他資訊區段`load_external_Table_latest` 中使用 下的程式碼。	DBA

處理測試檔案

任務	Description	所需的技能
將檔案上傳至 S3 儲存貯體。	若要將測試檔案上傳至 S3 儲存貯體，請使用主控台或在 AWS CLI 中使用下列命令。 `aws s3 cp /Users/Desktop/ukpost/exttbl/"testing files"/aps s3://s3importtest/inputext/aps` 一旦上傳檔案，儲存貯體事件就會啟動 Lambda 函數，執行 Aurora PostgreSQL 相容函數。	DBA
檢查資料以及日誌和錯誤檔案。	Aurora PostgreSQL 相容函數會將檔案載入主資料表，並在 S3 儲存貯體中建立 `.log`和 `.bad` 檔案。	DBA
監控解決方案。	在 Amazon CloudWatch 主控台中，監控 Lambda 函數。	DBA

相關資源

- Amazon S3 整合
- Amazon S3
- 使用 Amazon Aurora PostgreSQL 相容版本
- AWS Lambda
- Amazon CloudWatch
- AWS Secrets Manager
- 設定 Amazon SNS 通知

其他資訊

ext_table_scripts

```
CREATE EXTENSION aws_s3 CASCADE;
CREATE TABLE IF NOT EXISTS meta_EXTERNAL_TABLE
(
    table_name_stg character varying(100) ,
    table_name character varying(100)  ,
    col_list character varying(1000)  ,
    data_type character varying(100)  ,
    col_order numeric,
    start_pos numeric,
    end_pos numeric,
    no_position character varying(100)  ,
    date_mask character varying(100)  ,
    delimeter character(1)  ,
    directory character varying(100)  ,
    file_name character varying(100)  ,
    header_exist character varying(5)
);
CREATE TABLE IF NOT EXISTS ext_tbl_stg
(
    col1 text
);
CREATE TABLE IF NOT EXISTS error_table
(
    error_details text,
    file_name character varying(100),
    processed_time timestamp without time zone
);
CREATE TABLE IF NOT EXISTS log_table
(
    file_name character varying(50) COLLATE pg_catalog."default",
    processed_date timestamp without time zone,
    tot_rec_count numeric,
    proc_rec_count numeric,
    error_rec_count numeric
);
sample insert scripts of meta data:
INSERT INTO meta_EXTERNAL_TABLE (table_name_stg, table_name, col_list, data_type,
 col_order, start_pos, end_pos, no_position, date_mask, delimeter, directory,
 file_name, header_exist) VALUES ('F_EX_APS_TRANSACTIONS_STG', 'F_EX_APS_TRANSACTIONS',
```

```
 'source_filename', 'character varying', 2, 8, 27, NULL, NULL, NULL, 'databasedev',
 'externalinterface/loaddir/APS', 'NO');
INSERT INTO meta_EXTERNAL_TABLE (table_name_stg, table_name, col_list, data_type,
 col_order, start_pos, end_pos, no_position, date_mask, delimeter, directory,
 file_name, header_exist) VALUES ('F_EX_APS_TRANSACTIONS_STG', 'F_EX_APS_TRANSACTIONS',
 'record_type_identifier', 'character varying', 3, 28, 30, NULL, NULL, NULL,
 'databasedev', 'externalinterface/loaddir/APS', 'NO');
INSERT INTO meta_EXTERNAL_TABLE (table_name_stg, table_name, col_list, data_type,
 col_order, start_pos, end_pos, no_position, date_mask, delimeter, directory,
 file_name, header_exist) VALUES ('F_EX_APS_TRANSACTIONS_STG', 'F_EX_APS_TRANSACTIONS',
 'fad_code', 'numeric', 4, 31, 36, NULL, NULL, NULL, 'databasedev', 'externalinterface/
loaddir/APS', 'NO');
INSERT INTO meta_EXTERNAL_TABLE (table_name_stg, table_name, col_list, data_type,
 col_order, start_pos, end_pos, no_position, date_mask, delimeter, directory,
 file_name, header_exist) VALUES ('F_EX_APS_TRANSACTIONS_STG', 'F_EX_APS_TRANSACTIONS',
 'session_sequence_number', 'numeric', 5, 37, 42, NULL, NULL, NULL, 'databasedev',
 'externalinterface/loaddir/APS', 'NO');
INSERT INTO meta_EXTERNAL_TABLE (table_name_stg, table_name, col_list, data_type,
 col_order, start_pos, end_pos, no_position, date_mask, delimeter, directory,
 file_name, header_exist) VALUES ('F_EX_APS_TRANSACTIONS_STG', 'F_EX_APS_TRANSACTIONS',
 'transaction_sequence_number', 'numeric', 6, 43, 48, NULL, NULL, NULL, 'databasedev',
 'externalinterface/loaddir/APS', 'NO');
```

s3bucketpolicy_for 匯入

```
---Import role policy
--Create an IAM policy to allow, Get,  and list actions on S3 bucket
 {
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "s3import",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::s3importtest",
                "arn:aws:s3:::s3importtest/*"
            ]
        }
    ]
```

```
}
--Export Role policy
--Create an IAM policy to allow, put,  and list actions on S3 bucket
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "s3export",
            "Action": [
                "S3:PutObject",
                "s3:ListBucket"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::s3importtest/*"
            ]
        }
    ]
}
```

資料庫函數 load_external_tables_latest 範例

```
CREATE OR REPLACE FUNCTION public.load_external_tables(pi_filename text)
 RETURNS character varying
 LANGUAGE plpgsql
AS $function$
/* Loading data from S3 bucket into a APG table */
DECLARE
 v_final_sql TEXT;
 pi_ext_table TEXT;
 r refCURSOR;
 v_sqlerrm text;
 v_chunk numeric;
 i integer;
 v_col_list TEXT;
 v_postion_list CHARACTER VARYING(1000);
 v_len  integer;
 v_delim varchar;
 v_file_name CHARACTER VARYING(1000);
 v_directory CHARACTER VARYING(1000);
 v_table_name_stg CHARACTER VARYING(1000);
 v_sql_col TEXT;
 v_sql TEXT;
```

```
 v_sql1 TEXT;
 v_sql2 TEXT;
 v_sql3 TEXT;
 v_cnt integer;
 v_sql_dynamic TEXT;
 v_sql_ins TEXT;
 proc_rec_COUNT integer;
 error_rec_COUNT integer;
 tot_rec_COUNT integer;
 v_rec_val integer;
 rec record;
 v_col_cnt integer;
 kv record;
 v_val text;
 v_header text;
 j integer;
 ERCODE VARCHAR(5);
 v_region text;
 cr CURSOR FOR
 SELECT distinct DELIMETER,
   FILE_NAME,
   DIRECTORY
 FROM  meta_EXTERNAL_TABLE
 WHERE table_name = pi_ext_table
   AND DELIMETER IS NOT NULL;


 cr1 CURSOR FOR
   SELECT   col_list,
   data_type,
   start_pos,
   END_pos,
   concat_ws('',' ',TABLE_NAME_STG) as TABLE_NAME_STG,
   no_position,date_mask
 FROM  meta_EXTERNAL_TABLE
 WHERE table_name = pi_ext_table
 order by col_order asc;
cr2 cursor FOR
SELECT  distinct table_name,table_name_stg
   FROM  meta_EXTERNAL_TABLE
   WHERE upper(file_name) = upper(pi_filename);


BEGIN
```

```
 -- PERFORM utl_file_utility.init();
   v_region := 'us-east-1';
   /* find tab details from file name */


   --DELETE FROM  ERROR_TABLE WHERE file_name= pi_filename;
  -- DELETE FROM  log_table WHERE file_name= pi_filename;


 BEGIN


   SELECT distinct table_name,table_name_stg INTO strict pi_ext_table,v_table_name_stg
   FROM  meta_EXTERNAL_TABLE
   WHERE upper(file_name) = upper(pi_filename);
 EXCEPTION
   WHEN NO_DATA_FOUND THEN
    raise notice 'error 1,%',sqlerrm;
    pi_ext_table := null;
    v_table_name_stg := null;
      RAISE USING errcode = 'NTFIP' ;
    when others then
        raise notice 'error others,%',sqlerrm;
 END;
 j :=1 ;

for rec in  cr2
 LOOP




  pi_ext_table     := rec.table_name;
  v_table_name_stg := rec.table_name_stg;
  v_col_list := null;


 IF pi_ext_table IS NOT NULL
  THEN
    --EXECUTE concat_ws('','truncate table  ' ,pi_ext_table) ;
   EXECUTE concat_ws('','truncate table  ' ,v_table_name_stg) ;
```

```
      SELECT distinct DELIMETER INTO STRICT v_delim
      FROM  meta_EXTERNAL_TABLE
      WHERE table_name = pi_ext_table;


      IF v_delim IS NOT NULL THEN
    SELECT distinct DELIMETER,
      FILE_NAME,
      DIRECTORY ,
      concat_ws('',' ',table_name_stg),
      case  header_exist when 'YES' then 'CSV HEADER' else 'CSV' end as header_exist
    INTO STRICT v_delim,v_file_name,v_directory,v_table_name_stg,v_header
    FROM  meta_EXTERNAL_TABLE
    WHERE table_name = pi_ext_table
      AND DELIMETER IS NOT NULL;


    IF    upper(v_delim) = 'CSV'
    THEN
      v_sql := concat_ws('','SELECT aws_s3.table_import_FROM_s3 ( ''',
      v_table_name_stg,''','''',
      ''DELIMITER '''','''' CSV HEADER QUOTE '''''''''', aws_commons.create_s3_uri
( ''',
      v_directory,''','''',v_file_name,''', ''',v_region,'''))');
      ELSE
      v_sql := concat_ws('','SELECT aws_s3.table_import_FROM_s3(''',
          v_table_name_stg, ''','''', ''DELIMITER AS ''''^''''',''',','
         aws_commons.create_s3_uri
          ( ''',v_directory, ''','''',
          v_file_name, ''',',
           '''',v_region,''')
         )');
         raise notice 'v_sql , %',v_sql;
      begin
       EXECUTE  v_sql;
      EXCEPTION
        WHEN OTHERS THEN
          raise notice 'error 1';
        RAISE USING errcode = 'S3IMP' ;
      END;


      select count(col_list) INTO v_col_cnt
```

```
      from  meta_EXTERNAL_TABLE where table_name = pi_ext_table;









       -- raise notice 'v_sql 2, %',concat_ws('','update ',v_table_name_stg, ' set
col1 = col1||''',v_delim,''''');


      execute concat_ws('','update ',v_table_name_stg, ' set col1 =
col1||''',v_delim,''''');




      i :=1;
      FOR rec in cr1
      loop
      v_sql1 := concat_ws('',v_sql1,'split_part(col1,''',v_delim,''',', i,')',' as
',rec.col_list,',');
      v_sql2 := concat_ws('',v_sql2,rec.col_list,',');
  --    v_sql3 := concat_ws('',v_sql3,'rec.',rec.col_list,'::',rec.data_type,',');


      case
        WHEN upper(rec.data_type) = 'NUMERIC'
        THEN v_sql3 := concat_ws('',v_sql3,' case WHEN
length(trim(split_part(col1,''',v_delim,''',', i,'))) =0
               THEN null
                ELSE
                coalesce((trim(split_part(col1,''',v_delim,''',',
i,')))::NUMERIC,0)::',rec.data_type,' END as ',rec.col_list,',') ;
        WHEN UPPER(rec.data_type) = 'TIMESTAMP WITHOUT TIME ZONE' AND rec.date_mask =
'YYYYMMDD'
        THEN v_sql3 := concat_ws('',v_sql3,' case WHEN
length(trim(split_part(col1,''',v_delim,''',', i,'))) =0
               THEN null
                ELSE
                to_date(coalesce((trim(split_part(col1,''',v_delim,''',',
i,'))),''99990101''),''YYYYMMDD'')::',rec.data_type,' END as ',rec.col_list,',');
        WHEN UPPER(rec.data_type) = 'TIMESTAMP WITHOUT TIME ZONE' AND rec.date_mask =
'MM/DD/YYYY hh24:mi:ss'
```

```
        THEN v_sql3 := concat_ws('',v_sql3,' case WHEN
length(trim(split_part(col1,''',v_delim,''',', i,'))) =0
               THEN null
                ELSE
                to_date(coalesce((trim(split_part(col1,''',v_delim,''',',
i,'))),''01/01/9999 0024:00:00''),''MM/DD/YYYY hh24:mi:ss'')::',rec.data_type,' END as
',rec.col_list,',');
         ELSE
       v_sql3 := concat_ws('',v_sql3,' case WHEN
length(trim(split_part(col1,''',v_delim,''',', i,'))) =0
               THEN null
                ELSE
                 coalesce((trim(split_part(col1,''',v_delim,''',',
i,'))),''''):',rec.data_type,' END as ',rec.col_list,',') ;
      END case;


      i :=i+1;
      end loop;


        -- raise notice 'v_sql 3, %',v_sql3;


      SELECT trim(trailing ' ' FROM v_sql1) INTO v_sql1;
      SELECT trim(trailing ',' FROM v_sql1) INTO v_sql1;


      SELECT trim(trailing ' ' FROM v_sql2) INTO v_sql2;
      SELECT trim(trailing ',' FROM v_sql2) INTO v_sql2;


      SELECT trim(trailing ' ' FROM v_sql3) INTO v_sql3;
      SELECT trim(trailing ',' FROM v_sql3) INTO v_sql3;


      END IF;
     raise notice 'v_delim , %',v_delim;


    EXECUTE concat_ws('','SELECT COUNT(*) FROM ',v_table_name_stg)  INTO v_cnt;


   raise notice 'stg cnt , %',v_cnt;
```

```
    /* if upper(v_delim) = 'CSV' then
       v_sql_ins := concat_ws('', ' SELECT * from ' ,v_table_name_stg );
     else
      -- v_sql_ins := concat_ws('',' SELECT ',v_sql1,'  from (select col1 from
 ' ,v_table_name_stg , ')sub ');
       v_sql_ins := concat_ws('',' SELECT ',v_sql3,'  from (select col1 from
 ' ,v_table_name_stg , ')sub ');
       END IF;*/


v_chunk := v_cnt/100;




for i in 1..101
loop
     BEGIN
    -- raise notice 'v_sql , %',v_sql;
       -- raise notice 'Chunk number , %',i;
       v_sql_ins := concat_ws('',' SELECT ',v_sql3,'  from (select col1 from
 ' ,v_table_name_stg , ' offset ',v_chunk*(i-1), ' limit ',v_chunk,') sub ');


     v_sql := concat_ws('','insert into  ', pi_ext_table ,' ', v_sql_ins);
     -- raise notice 'select statement , %',v_sql_ins;
          -- v_sql := null;
     -- EXECUTE concat_ws('','insert into  ', pi_ext_table ,' ', v_sql_ins, 'offset
 ',v_chunk*(i-1), ' limit ',v_chunk );
     --v_sql := concat_ws('','insert into  ', pi_ext_table ,' ', v_sql_ins );


     -- raise notice 'insert statement , %',v_sql;


    raise NOTICE 'CHUNK START %',v_chunk*(i-1);
   raise NOTICE 'CHUNK END %',v_chunk;


     EXECUTE v_sql;
```

```
 EXCEPTION
      WHEN OTHERS THEN
      -- v_sql_ins := concat_ws('',' SELECT ',v_sql1, '  from (select col1 from
' ,v_table_name_stg , ' )sub ');
        -- raise notice 'Chunk number for cursor , %',i;


   raise NOTICE 'Cursor - CHUNK START %',v_chunk*(i-1);
  raise NOTICE 'Cursor -  CHUNK END %',v_chunk;
        v_sql_ins := concat_ws('',' SELECT ',v_sql3, '  from (select col1 from
' ,v_table_name_stg , ' )sub ');


        v_final_sql := REPLACE (v_sql_ins, ''''::text, ''''''::text);
       -- raise notice 'v_final_sql %',v_final_sql;
        v_sql :=concat_ws('','do $a$ declare  r refcursor;v_sql text; i
numeric;v_conname text;  v_typ  ',pi_ext_table,'[]; v_rec  ','record',';
          begin









          open r for execute ''select col1 from ',v_table_name_stg ,'  offset
',v_chunk*(i-1), ' limit ',v_chunk,''';
          loop
          begin
          fetch r into v_rec;
          EXIT WHEN NOT FOUND;




          v_sql := concat_ws('''','insert into  ',pi_ext_table,' SELECT ',REPLACE
(v_sql3, ''''::text, ''''''::text) , '  from ( select ''''''',v_rec.col1,''''''' as
col1) v'');
           execute v_sql;


          exception
           when others then
         v_sql := ''INSERT INTO  ERROR_TABLE VALUES (concat_ws(''''''''','''Error
Name: ''',$$''||SQLERRM||''$$,'''Error State: ''','''''||
```

```
SQLSTATE||''''''','''record : '''',$$''||v_rec.col1||''$$),'''''||
pi_filename||''''',now())'';


              execute v_sql;
            continue;
          end ;
          end loop;
          close r;
          exception
          when others then
        raise;
          end ; $a$');
     -- raise notice ' inside excp v_sql %',v_sql;
         execute v_sql;
     --  raise notice 'v_sql %',v_sql;
      END;
  END LOOP;
     ELSE


    SELECT distinct DELIMETER,FILE_NAME,DIRECTORY ,concat_ws('',' ',table_name_stg),
      case  header_exist when 'YES' then 'CSV HEADER' else 'CSV' end as header_exist
      INTO STRICT v_delim,v_file_name,v_directory,v_table_name_stg,v_header
    FROM  meta_EXTERNAL_TABLE
    WHERE table_name = pi_ext_table                  ;
    v_sql := concat_ws('','SELECT aws_s3.table_import_FROM_s3(''',
      v_table_name_stg, ''','''', ''DELIMITER AS ''''#'''' ',v_header,' '','','
     aws_commons.create_s3_uri
      ( ''',v_directory, ''','''',
      v_file_name, ''',',
       '''',v_region,''')
     )');
        EXECUTE  v_sql;


    FOR rec in cr1
    LOOP


     IF rec.start_pos IS NULL AND rec.END_pos IS NULL AND rec.no_position = 'recnum'
     THEN
       v_rec_val := 1;
     ELSE
```

```
      case
        WHEN upper(rec.data_type) = 'NUMERIC'
        THEN v_sql1 := concat_ws('',' case WHEN length(trim(substring(COL1,
',rec.start_pos ,',', rec.END_pos,'-',rec.start_pos ,'+1))) =0
               THEN null
                ELSE
                coalesce((trim(substring(COL1, ',rec.start_pos ,',',
rec.END_pos,'-',rec.start_pos ,'+1)))::NUMERIC,0)::',rec.data_type,' END as
',rec.col_list,',') ;
        WHEN UPPER(rec.data_type) = 'TIMESTAMP WITHOUT TIME ZONE' AND rec.date_mask =
'YYYYMMDD'
        THEN v_sql1 := concat_ws('','case WHEN length(trim(substring(COL1,
',rec.start_pos ,',', rec.END_pos,'-',rec.start_pos ,'+1))) =0
               THEN null
                ELSE
                to_date(coalesce((trim(substring(COL1, ',rec.start_pos ,',',
rec.END_pos,'-',rec.start_pos ,'+1))),''99990101''),''YYYYMMDD'')::',rec.data_type,'
END as ',rec.col_list,',');
        WHEN UPPER(rec.data_type) = 'TIMESTAMP WITHOUT TIME ZONE' AND rec.date_mask =
'YYYYMMDDHH24MISS'
        THEN v_sql1 := concat_ws('','case WHEN length(trim(substring(COL1,
',rec.start_pos ,',', rec.END_pos,'-',rec.start_pos ,'+1))) =0
               THEN null
                ELSE
                to_date(coalesce((trim(substring(COL1, ',rec.start_pos ,',',
rec.END_pos,'-',rec.start_pos ,'+1))),''9999010100240000''),''YYYYMMDDHH24MISS'')::',rec.data_
END as ',rec.col_list,',');
         ELSE
       v_sql1 := concat_ws('',' case WHEN length(trim(substring(COL1,
',rec.start_pos ,',', rec.END_pos,'-',rec.start_pos ,'+1))) =0
               THEN null
                ELSE
                 coalesce((trim(substring(COL1, ',rec.start_pos ,',',
rec.END_pos,'-',rec.start_pos ,'+1))),''''):',rec.data_type,' END as
',rec.col_list,',') ;
      END case;


     END IF;
     v_col_list := concat_ws('',v_col_list ,v_sql1);
    END LOOP;
```

```
          SELECT trim(trailing ' ' FROM v_col_list) INTO v_col_list;
          SELECT trim(trailing ',' FROM v_col_list) INTO v_col_list;


          v_sql_col   :=  concat_ws('',trim(trailing ',' FROM v_col_list) , ' FROM
',v_table_name_stg,' WHERE col1 IS NOT NULL AND length(col1)>0 ');




          v_sql_dynamic := v_sql_col;


          EXECUTE  concat_ws('','SELECT COUNT(*) FROM ',v_table_name_stg) INTO v_cnt;




        IF v_rec_val = 1 THEN
            v_sql_ins := concat_ws('',' select row_number() over(order by ctid) as
line_number ,' ,v_sql_dynamic) ;


        ELSE
              v_sql_ins := concat_ws('',' SELECT' ,v_sql_dynamic) ;
          END IF;


    BEGIN
      EXECUTE concat_ws('','insert into  ', pi_ext_table ,' ', v_sql_ins);
          EXCEPTION
             WHEN OTHERS THEN
         IF v_rec_val = 1 THEN
                v_final_sql := ' select row_number() over(order by ctid) as
line_number ,col1 from ';
               ELSE
                v_final_sql := ' SELECT col1 from';
              END IF;
      v_sql :=concat_ws('','do $a$ declare  r refcursor;v_rec_val numeric :=
',coalesce(v_rec_val,0),';line_number numeric; col1 text; v_typ  ',pi_ext_table,'[];
v_rec  ',pi_ext_table,';
```

```
            begin
            open r for execute ''',v_final_sql, ' ',v_table_name_stg,' WHERE col1 IS
NOT NULL AND length(col1)>0 '' ;
            loop
            begin
            if   v_rec_val = 1 then
            fetch r into line_number,col1;
            else
            fetch r into col1;
            end if;


            EXIT WHEN NOT FOUND;
             if v_rec_val = 1 then
             select line_number,',trim(trailing ',' FROM v_col_list) ,' into v_rec;
             else
               select ',trim(trailing ',' FROM v_col_list) ,' into v_rec;
             end if;


            insert into  ',pi_ext_table,' select v_rec.*;
             exception
             when others then
              INSERT INTO  ERROR_TABLE VALUES (concat_ws(''''',''Error Name:
'',SQLERRM,''Error State: '',SQLSTATE,''record : '',v_rec),''',pi_filename,''',now());
              continue;
             end ;
              end loop;
            close r;
             exception
             when others then
             raise;
             end ; $a$');
        execute v_sql;


    END;


        END IF;


  EXECUTE concat_ws('','SELECT COUNT(*) FROM  ' ,pi_ext_table)   INTO proc_rec_COUNT;
```

```
   EXECUTE concat_ws('','SELECT COUNT(*) FROM  error_table WHERE file_name
 ='''',pi_filename,''' and processed_time::date = clock_timestamp()::date')  INTO
 error_rec_COUNT;


   EXECUTE concat_ws('','SELECT COUNT(*) FROM ',v_table_name_stg)   INTO tot_rec_COUNT;


   INSERT INTO  log_table values(pi_filename,now(),tot_rec_COUNT,proc_rec_COUNT,
 error_rec_COUNT);


   raise notice 'v_directory, %',v_directory;


   raise notice 'pi_filename, %',pi_filename;


   raise notice 'v_region, %',v_region;


  perform aws_s3.query_export_to_s3('SELECT
 replace(trim(substring(error_details,position(''('' in
 error_details)+1),'')''),'','','';''),file_name,processed_time FROM  error_table WHERE
 file_name = '''||pi_filename||'''',
   aws_commons.create_s3_uri(v_directory, pi_filename||'.bad', v_region),
   options :='FORmat csv, header, delimiter $$,$$'
   );


raise notice 'v_directory, %',v_directory;


   raise notice 'pi_filename, %',pi_filename;


   raise notice 'v_region, %',v_region;


  perform aws_s3.query_export_to_s3('SELECT * FROM  log_table WHERE file_name = '''||
pi_filename||'''',
   aws_commons.create_s3_uri(v_directory, pi_filename||'.log', v_region),
   options :='FORmat csv, header, delimiter $$,$$'
```

```
   );




   END IF;
 j := j+1;
 END LOOP;


       RETURN 'OK';
EXCEPTION
    WHEN  OTHERS THEN
  raise notice 'error %',sqlerrm;
   ERCODE=SQLSTATE;
   IF ERCODE = 'NTFIP' THEN
     v_sqlerrm := concat_Ws('',sqlerrm,'No data for the filename');
   ELSIF ERCODE = 'S3IMP' THEN
    v_sqlerrm := concat_Ws('',sqlerrm,'Error While exporting the file from S3');
   ELSE
      v_sqlerrm := sqlerrm;
   END IF;


 select distinct directory into v_directory from  meta_EXTERNAL_TABLE;




 raise notice 'exc v_directory, %',v_directory;


   raise notice 'exc pi_filename, %',pi_filename;


   raise notice 'exc v_region, %',v_region;


  perform aws_s3.query_export_to_s3('SELECT * FROM  error_table WHERE file_name = '''||
pi_filename||'''',
   aws_commons.create_s3_uri(v_directory, pi_filename||'.bad', v_region),
   options :='FORmat csv, header, delimiter $$,$$'
   );
    RETURN null;
```

```
END;
$function$
```

將函數型索引從 Oracle 遷移至 PostgreSQL

Veeranjaneyulu Grandhi 和 Navakanth Talluri，Amazon Web Services

總結

索引是增強資料庫效能的常見方式。索引可讓資料庫伺服器比沒有索引時更快地尋找和擷取特定資料列。但索引也會為資料庫系統整體增加額外負荷，因此應該合理地使用它們。以函數為基礎的索引，以函數或表達式為基礎，可以涉及多個欄和數學表達式。以函數為基礎的索引可改善使用索引表達式的查詢效能。

原本，PostgreSQL 不支援使用波動定義為穩定的函數建立以函數為基礎的索引。不過，您可以建立具有波動的類似函數，IMMUTABLE並在建立索引時使用它們。

IMMUTABLE 函數無法修改資料庫，並保證永遠傳回相同的結果與相同的引數。此類別可讓最佳化工具在查詢使用常數引數呼叫函數時預先評估函數。

此模式有助於將 Oracle 函數型索引與 to_char、 to_date和 等函數搭配使用時遷移to_number至 PostgreSQL 對等項目。

先決條件和限制

先決條件

- 作用中的 Amazon Web Services (AWS) 帳戶
- 具有接聽程式服務設定和執行的來源 Oracle 資料庫執行個體
- 熟悉 PostgreSQL 資料庫

限制

- 資料庫大小限制為 64 TB。
- 用於建立索引的函數必須為 IMMUTABLE。

產品版本

- 11g 版 (11.2.0.3.v1 版及更新版本） 及最高 12.2 版和 18c 版的所有 Oracle 資料庫版本
- PostgreSQL 9.6 版及更新版本

Architecture

來源技術堆疊

- 內部部署或 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體或 Amazon RDS for Oracle 資料庫執行個體上的 Oracle 資料庫

目標技術堆疊

- 任何 PostgreSQL 引擎

工具

- pgAdmin 4 是 Postgres 的開放原始碼管理工具。pgAdmin 4 工具提供圖形界面，用於建立、維護和使用資料庫物件。
- Oracle SQL Developer 是整合式開發環境 (IDE)，用於在傳統和雲端部署中開發和管理 Oracle 資料庫。

史詩

使用預設函數建立以函數為基礎的索引

任務	Description	所需的技能

使用 to_char 函數在資料欄上建立以函數為基礎的索引。	使用下列程式碼來建立以函數為基礎的索引。
	`postgres=# create table funcindex( col1 timestamp without time zone);`
	`CREATE TABLE`
	`postgres=# insert into funcindex values (now());`
	`INSERT 0 1`
	`postgres=# select * from funcindex;`
	`col1`
		DBA、App 開發人員

任務	Description	所需的技能
	``` ---------- ------ 2022-08-09 16:00:57. 77414 (1 rows) postgres=# create index funcindex_idx on funcindex(to_char(col1,'DD-MM-YYYY HH24:MI:SS')); ERROR: functions in index expression must be marked IMMUTABLE ```	
	Note PostgreSQL 不允許在沒有 `IMMUTABLE` 子句的情況下建立以函數為基礎的索引。	
檢查函數的波動。	若要檢查函數波動，請使用其他資訊區段中的程式碼。	DBA

使用包裝函式建立以函式為基礎的索引

任務	Description	所需的技能
建立包裝函式。	若要建立包裝函式，請使用其他資訊區段中的程式碼。	PostgreSQL 開發人員
使用包裝函式建立索引。	使用其他資訊區段中的程式碼，在與應用程式相同的結構描述`IMMUTABLE` 中建立具有	DBA、PostgreSQL 開發人員

任務	Description	所需的技能
	關鍵字的使用者定義函數，並在索引建立指令碼中參考它。 如果在一般結構描述中建立使用者定義的函數（從先前的範例），請更新顯示的 `search_path` 。 `ALTER ROLE <ROLENAME> set search_path=$user, COMMON;`	

驗證索引建立

任務	Description	所需的技能
驗證索引建立。	驗證是否需要根據查詢存取模式建立索引。	DBA
驗證索引是否可以使用。	若要檢查 PostgreSQL Optimizer 是否收取函數型索引，請使用 explain 或 explain analysis 執行 SQL 陳述式。使用其他資訊區段中的程式碼。如果可能，也請收集資料表統計資料。 **Note** 如果您注意到解釋計畫，PostgreSQL 最佳化工具已因為述詞條件而選擇以函數為基礎的索引。	DBA

相關資源

- 以函數為基礎的索引 (Oracle 文件）
- 運算式上的索引 (PostgreSQL 文件）
- PostgreSQL 波動 (PostgreSQL 文件）
- PostgreSQL search_path (PostgreSQL 文件）
- Oracle 資料庫 19c 至 Amazon Aurora PostgreSQL 遷移手冊

其他資訊

建立包裝函式

```
CREATE OR REPLACE FUNCTION myschema.to_char(var1 timestamp without time zone, var2
 varchar) RETURNS varchar AS $BODY$ select to_char(var1, 'YYYYMMDD'); $BODY$ LANGUAGE
 sql IMMUTABLE;
```

使用包裝函式建立索引

```
postgres=# create function common.to_char(var1 timestamp without time zone, var2
 varchar) RETURNS varchar AS $BODY$ select to_char(var1, 'YYYYMMDD'); $BODY$ LANGUAGE
 sql IMMUTABLE;
CREATE FUNCTION
postgres=# create index funcindex_idx on funcindex(common.to_char(col1,'DD-MM-YYYY
 HH24:MI:SS'));
CREATE INDEX
```

檢查函數的波動

```
SELECT DISTINCT p.proname as "Name",p.provolatile as "volatility" FROM
 pg_catalog.pg_proc p
 LEFT JOIN pg_catalog.pg_namespace n ON n.oid = p.pronamespace
 LEFT JOIN pg_catalog.pg_language l ON l.oid = p.prolang
 WHERE n.nspname OPERATOR(pg_catalog.~) '^(pg_catalog)$' COLLATE pg_catalog.default AND
 p.proname='to_char'GROUP BY p.proname,p.provolatile
ORDER BY 1;
```

驗證索引是否可以使用

```
explain analyze <SQL>
```

```
postgres=# explain select col1 from funcindex where common.to_char(col1,'DD-MM-YYYY
 HH24:MI:SS') = '09-08-2022 16:00:57';
                                                        QUERY PLAN
-----------------------------------------------------------------------------------------------------------
 Index Scan using funcindex_idx on funcindex  (cost=0.42..8.44 rows=1 width=8)
   Index Cond: ((common.to_char(col1, 'DD-MM-YYYY HH24:MI:SS'::character
 varying))::text = '09-08-2022 16:00:57'::text)
(2 rows)
```

使用延伸模組將 Oracle 原生函數遷移至 PostgreSQL

Pinesh Singal，Amazon Web Services

總結

此遷移模式提供step-by-step指引，透過修改 aws_oracle_ext和orafce擴充功能至 PostgreSQL () 原生內建程式碼，將 Amazon Relational Database Service (Amazon RDSpsql) for Oracle 資料庫執行個體遷移至 Amazon RDS for PostgreSQL 或 Amazon Aurora PostgreSQL 相容版本資料庫。這可節省處理時間。

模式說明離線手動遷移策略，多 TB Oracle 來源資料庫具有大量交易時不會停機。

遷移程序使用 AWS Schema Conversion Tool (AWS SCT) 搭配 aws_oracle_ext和 orafce延伸模組，將 Amazon RDS for Oracle 資料庫結構描述轉換為 Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 相容資料庫結構描述。然後，程式碼會手動變更為 PostgreSQL 支援的原生psql內建程式碼。這是因為延伸呼叫會影響 PostgreSQL 資料庫伺服器上的程式碼處理，而且並非所有延伸程式碼都完全抱怨或與 PostgreSQL 程式碼相容。

此模式主要著重於使用 AWS SCT 和擴充功能 aws_oracle_ext 和 手動遷移 SQL 程式碼orafce。您可以將已使用的延伸模組轉換為原生 PostgreSQL (psql) 內建。然後，您移除延伸模組的所有參考，並相應地轉換程式碼。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 作業系統 (Windows 或 Mac) 或 Amazon EC2 執行個體 （啟動和執行中）
- Orafce

限制

並非所有使用 aws_oracle_ext或 orafce擴充功能的 Oracle 函數都可以轉換為原生 PostgreSQL 函數。它可能需要手動重新作業，以便使用 PostgreSQL 程式庫進行編譯。

使用 AWS SCT 擴充功能的一個缺點是執行和擷取結果時效能緩慢。您可以從 Oracle SYSDATE函數遷移到所有三個代碼 (aws_oracle_ext、 和psql預設值） 之間的 PostgreSQL NOW()函數的簡易 PostgreSQL EXPLAIN 計劃 （陳述式的執行計劃） 了解其成本orafce，如隨附文件中效能比較檢查一節所述。

產品版本

- Source：Amazon RDS for Oracle 資料庫 10.2 和更新版本（適用於 10.x)、11g (11.2.0.3.v1 和更新版本），以及 Enterprise Edition、Standard Edition、Standard Edition 1 和 Standard Edition 2 最多 12.2、18c 和 19c（和更新版本）
- 目標：Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 相容資料庫 9.4 及更新版本（適用於 9.x)、10.x、11.x、12.x、13.x 及 14.x（及更新版本）
- AWS SCT：最新版本（此模式已使用 1.0.632 進行測試）
- Orafce：最新版本（此模式已使用 3.9.0 進行測試）

Architecture

來源技術堆疊

- 版本為 12.1.0.2.v18 的 Amazon RDS for Oracle 資料庫執行個體

目標技術堆疊

- 具有 11.5 版的 Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 相容資料庫執行個體

資料庫遷移架構

下圖代表來源 Oracle 和目標 PostgreSQL 資料庫之間的資料庫遷移架構。架構涉及 AWS Cloud、虛擬私有雲端 (VPC)、可用區域、私有子網路、Amazon RDS for Oracle 資料庫、AWS SCT、Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 相容資料庫、Oracle (`aws_oracle_ext` 和 `orafce`) 擴充功能，以及結構化查詢語言 (SQL) 檔案。

1. 啟動 Amazon RDS for Oracle 資料庫執行個體（來源資料庫）。
2. 使用 AWS SCT 搭配 `aws_oracle_ext`和 `orafce`延伸套件，將原始碼從 Oracle 轉換為 PostreSQL。
3. 轉換會產生 PostgreSQL 支援的遷移 .sql 檔案。
4. 手動將未轉換的 Oracle 延伸程式碼轉換為 PostgreSQL (`psql`) 程式碼。
5. 手動轉換會產生 PostgreSQL 支援的轉換後 .sql 檔案。
6. 在 Amazon RDS for PostgreSQL 資料庫執行個體（目標資料庫）上執行這些 .sql 檔案。

工具

工具

AWS 服務

- AWS SCT - AWS Schema Conversion Tool (AWS SCT) 會將您現有的資料庫結構描述從一個資料庫引擎轉換為另一個資料庫引擎。您可以轉換關聯式線上交易處理 (OLTP) 結構描述或資料倉儲結構描述。轉換後的結構描述適用於 Amazon RDS for MySQL 資料庫執行個體、Amazon Aurora 資料庫叢集、Amazon RDS for PostgreSQL 資料庫執行個體或 Amazon Redshift 叢集。轉換後的結構描述也可以與 Amazon EC2 執行個體上的資料庫搭配使用，或儲存為 Amazon S3 儲存貯體中的資料。

  AWS SCT 提供專案型使用者介面，可將來源資料庫的資料庫結構描述自動轉換為與您目標 Amazon RDS 執行個體相容的格式。

  您可以使用 AWS SCT 從 Oracle 來源資料庫遷移至上述任何目標。您可以使用 AWS SCT 匯出來源資料庫物件定義，例如結構描述、檢視、預存程序和函數。

  您可以使用 AWS SCT 將資料從 Oracle 轉換為 Amazon RDS for PostgreSQL 或 Amazon Aurora PostgreSQL 相容版本。

  在此模式中，您可以使用 AWS SCT，使用延伸模組 和 將 Oracle 程式碼轉換`aws_oracle_ext`並遷移至 PostgreSQL`orafce`，並手動將延伸程式碼遷移至`psql`預設或原生內建程式碼。
- AWS SCT 延伸套件是一種附加元件模組，可模擬來源資料庫中存在的函數，而這些函數在將物件轉換為目標資料庫時需要。您需要轉換資料庫結構描述，才能安裝 AWS SCT 延伸套件。

  當您轉換資料庫或資料倉儲結構描述時，AWS SCT 會將額外的結構描述新增至目標資料庫。此結構描述會實作來源資料庫的 SQL 系統功能，當您將已轉換的結構描述寫入至目標資料庫時需要這些功能。這個額外的結構描述稱為延伸套件結構描述。

  OLTP 資料庫的延伸套件結構描述會根據來源資料庫命名。對於 Oracle 資料庫，延伸套件結構描述為 `AWS_ORACLE_EXT`。

其他工具

- Orafce – Orafce 是實作 Oracle 相容函數、資料類型和套件的模組。這是具有 Berkeley 來源分佈 (BSD) 授權的開放原始碼工具，因此任何人都可以使用它。`orafce` 模組適用於從 Oracle 遷移到 PostgreSQL，因為它在 PostgreSQL 中實作了許多 Oracle 函數。

Code

如需從 Oracle 到 PostgreSQL 的所有常用和遷移程式碼清單，以避免使用 AWS SCT 延伸程式碼，請參閱隨附的文件。

史詩

設定 Amazon RDS for Oracle 來源資料庫

任務	Description	所需的技能
建立 Oracle 資料庫執行個體。	從 Amazon RDS 主控台建立 Amazon RDS for Oracle 或 Aurora PostgreSQL 相容資料庫執行個體。	一般 AWS、DBA
設定安全群組。	設定傳入和傳出安全群組。	一般 AWS
建立資料庫。	使用所需的使用者和結構描述建立 Oracle 資料庫。	一般 AWS、DBA
建立物件。	在結構描述中建立物件和插入資料。	DBA

設定 Amazon RDS for PostgreSQL 目標資料庫

任務	Description	所需的技能
建立 PostgreSQL 資料庫執行個體。	從 Amazon RDS 主控台建立 Amazon RDS for PostgreSQL 或 Amazon Aurora PostgreSQL 資料庫執行個體。	一般 AWS、DBA
設定安全群組。	設定傳入和傳出安全群組。	一般 AWS
建立資料庫。	使用所需的使用者和結構描述建立 PostgreSQL 資料庫。	一般 AWS、DBA

任務	Description	所需的技能
驗證擴充功能。	確定aws_oracle_ext orafce已在 PostgreSQL 資料庫中正確安裝和設定 和 。	DBA
確認 PostgreSQL 資料庫可用。	確定 PostgreSQL 資料庫已啟動並執行。	DBA

使用 AWS SCT 和擴充功能將 Oracle 結構描述遷移至 PostgreSQL

任務	Description	所需的技能
安裝 AWS SCT。	安裝最新版本的 AWS SCT。	DBA
設定 AWS SCT。	使用適用於 Oracle () 和 PostgreSQL () 的 Java Database Connectivity (JDBCojdbc8.jar) 驅動程式設定 AWS SCTpostgresql-42.2.5.jar 。	DBA
啟用 AWS SCT 延伸套件或範本。	在 AWS SCT 專案設定下，使用 Oracle 資料庫結構描述的 aws_oracle_ext 和 orafce擴充功能啟用內建函數實作。	DBA
轉換結構描述。	在 AWS SCT 中，選擇轉換結構描述，將結構描述從 Oracle 轉換為 PostgreSQL，並產生 .sql 檔案。	DBA

將 AWS SCT 延伸程式碼轉換為 psql 程式碼

任務	Description	所需的技能
手動轉換程式碼。	手動將延伸支援程式碼的每一行轉換為psql預設內建程式碼，如附加文件中詳述。例如，將 AWS_ORACLE_EXT.SYSDATE() 或 ORACLE.SYSDATE() 變更為 NOW()。	DBA
驗證程式碼	（選用）在 PostgreSQL 資料庫中暫時執行程式碼，以驗證每一行程式碼。	DBA
在 PostgreSQL 資料庫中建立物件。	若要在 PostgreSQL 資料庫中建立物件，請執行由 AWS SCT 產生並在前兩個步驟中修改的 .sql 檔案。	DBA

相關資源

- 資料庫
  - Amazon RDS 上的 Oracle
  - Amazon RDS 上的 PostgreSQL
  - 使用 Amazon Aurora PostgreSQL
  - PostgreSQL EXPLAIN 計畫
- AWS SCT
  - AWS Schema Conversion Tool概觀
  - AWS SCT 使用者指南
  - 使用 AWS SCT 使用者介面
  - 使用 Oracle 資料庫做為 AWS SCT 的來源
- AWS SCT 的延伸模組
  - 使用 AWS SCT 延伸套件

- Oracle 功能 (en)
- PGXN 或afce
- GitHub 或afce

其他資訊

如需詳細資訊，請遵循詳細的命令搭配語法和範例，以手動轉換附加文件中的程式碼。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS DMS 將 Db2 資料庫從 Amazon EC2 遷移至 Aurora MySQL 相容

Pinesh Singal，Amazon Web Services

總結

將 IBM Db2 for LUW 資料庫遷移至 Amazon Elastic Compute Cloud (Amazon EC2) 之後，請考慮移至 Amazon Web Services (AWS) 雲端原生資料庫來重新架構資料庫。此模式涵蓋將 Amazon ECDb22 遷移至 AWS 上的 Amazon Aurora MySQL 相容版本資料庫。 EC2

模式說明線上遷移策略，對於具有大量交易的多 TB Db2 來源資料庫，停機時間最短。

此模式使用 AWS Schema Conversion Tool (AWS SCT) 將 Db2 資料庫結構描述轉換為 Aurora MySQL 相容結構描述。然後， 模式會使用 AWS Database Migration Service (AWS DMS) 將資料從 Db2 資料庫遷移至 Aurora MySQL 相容資料庫。非由 AWS SCT 轉換的程式碼需要手動轉換。

先決條件和限制

先決條件

- 具有虛擬私有雲端 (VPC) 的作用中 AWS 帳戶
- AWS SCT
- AWS DMS

產品版本

- AWS SCT 最新版本
- 適用於 Linux 的 Db2 11.1.4.4 版及更新版本

Architecture

來源技術堆疊

- 安裝在 EC2 執行個體上的 DB2/Linux x86-64 位元 EC2

目標技術堆疊

- Amazon Aurora MySQL 相容版本資料庫執行個體

來源和目標架構

下圖顯示來源 Db2 與目標 Aurora MySQL 相容資料庫之間的資料遷移架構。AWS 雲端上的架構包含虛擬私有雲端 (VPC)（虛擬私有雲端）、可用區域、Db2 執行個體和 AWS DMS 複寫執行個體的公有子網路，以及 Aurora MySQL 相容資料庫的私有子網路。

工具

AWS 服務

- Amazon Aurora 是全受管關聯式資料庫引擎，專為雲端而建置，並與 MySQL 和 PostgreSQL 相容。
- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- AWS Schema Conversion Tool (AWS SCT) 會自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，以支援異質資料庫遷移。AWS SCT 支援 LUW 9.1、9.5、9.7、10.1、10.5、11.1 和 11.5 版的來源 IBM Db2。

最佳實務

如需最佳實務，請參閱 AWS Database Migration Service 的最佳實務。

史詩

設定來源 IBM Db2 資料庫

任務	Description	所需的技能
在 Amazon EC2 上建立 IBM Db2 資料庫。 Amazon EC2	您可以使用來自 AWS Marketplace 的 Amazon Machine Image (AMI)，或在 EC2Db2 執行個體上安裝 Db2 軟體，在 EC2 執行個體上建立 IBM Db2 資料庫。 EC2	DBA、一般 AWS

任務	Description	所需的技能
	透過選取與內部部署資料庫類似的 AMI for IBM Db2 （例如 IBM Db2 v11.5.7 RHEL 7.9) 來啟動 EC2 執行個體。	
設定安全群組。	分別使用連接埠 22 和 50000 設定 SSH （安全殼層） 和 TCP 的 VPC 安全群組傳入規則。	一般 AWS

任務	Description	所需的技能
建立資料庫執行個體。	建立新的執行個體（使用者）和資料庫（結構描述），或使用預設db2inst1執行個體和範例資料庫。 1. 使用終端機連線至 Db2 資料庫，以連線至 EC2 執行個體。 Db2 或者，您可以安裝任何將連接到 Db2 資料庫的資料庫用戶端軟體。 2. 若要設定 db2inst1 使用者的密碼，請執行命令 `sudo passwd db2inst1`。 3. 若要連線至 db2inst1 執行個體，請執行命令 `sudo su - db2inst1`。 4. 若要連線至 Db2 資料庫，請執行命令 `db2`。 5. 若要連線至範例資料庫，請使用命令 `connect to sample`。或者，連線到您建立的資料庫。 6. 連線至資料庫執行個體後，請使用 Db2 SQL 陳述式建立物件並將資料插入這些物件。	DBA
確認 Db2 資料庫執行個體可用。	若要確認 Db2 資料庫執行個體已啟動並執行，請使用 `Db2pd -`命令。	DBA

設定目標 Aurora MySQL 相容資料庫

任務	Description	所需的技能
建立 Aurora MySQL 相容資料庫。	從 AWS RDS 服務建立具有 MySQL 相容性資料庫的 Amazon Aurora • 使用 MySQL 相容性和您選擇的版本在 Amazon Aurora 上建立資料庫，例如 Aurora (MySQL)–5.6.10a • 安裝 MySQL Workbench 應用程式或您偏好的資料庫用戶端軟體，可讓您連線至 MySQL 資料庫	DBA、一般 AWS
設定安全群組。	設定 SSH 和 TCP 連線的 VPC 安全群組傳入規則。	一般 AWS
確認 Aurora 資料庫可用。	若要確保 Aurora MySQL 相容資料庫已啟動並執行，請執行下列動作： 1. 透過 SSH 連線至 EC2 執行個體。 2. 從 MySQL Workbench 設定並連線至 Aurora MySQL 相容執行個體。使用端點做為主機名稱，如下列範例所示。 `mysql-cluster-instance-1.cokmvis0v46q.us-east-1.rds.amazonaws.com`	DBA

任務	Description	所需的技能
	3. 建立並連線至新的結構描述（例如 `mysql-sample-db2`）。 4. 執行 MySQL 陳述式來檢查資料庫中的結構描述和物件。	

設定和執行 AWS SCT

任務	Description	所需的技能
安裝 AWS SCT。	下載並安裝最新版本的 AWS SCT（最新版本 1.0.628）。	一般 AWS
設定 AWS SCT。	1. 下載適用於 IBM Db2 (4.22.X 版本）和 MySQL (8.x) 的 Java Database Connectivity (JDBC) 驅動程式。 2. 若要在 AWS SCT 中設定驅動程式，請選擇設定、全域設定、驅動程式。	一般 AWS
建立 AWS SCT 專案。	建立 AWS SCT 專案和報告，使用 Db2 for LUW 做為來源資料庫引擎，並使用 Aurora MySQL 相容於目標資料庫引擎。 若要識別連線至 Db2 for LUW 資料庫所需的權限，請參閱使用 Db2 LUW 做為 AWS SCT 的來源。	一般 AWS

任務	Description	所需的技能
驗證物件。	選擇載入結構描述，驗證物件。更新目標資料庫上任何不正確的物件： 1. 提供連線詳細資訊，然後選擇測試連線，以連接至 Amazon Aurora MySQL 相容伺服器。 來源和目標連線都必須成功，AWS SCT 才能啟動遷移報告。 2. 報告完成後，輸入要轉換的結構描述，然後選擇完成。 AWS SCT 會列出已轉換且發生錯誤的任何來源和目標物件。 3. 檢閱錯誤，並手動清除。 4. 清除所有錯誤後，開啟結構描述的內容（按一下滑鼠右鍵）選單，然後選擇載入結構描述。 5. 選擇套用至資料庫。 6. 在 MySQL Workbench 中，連線至 Aurora MySQL 相容資料庫，並檢查結構描述和物件。	DBA、一般 AWS

設定和執行 AWS DMS

任務	Description	所需的技能
建立複寫執行個體。	登入 AWS 管理主控台，導覽至 AWS DMS 服務，並使用您為來源和目標資料庫設定的 VPC 安全群組的有效設定來建立複寫執行個體。	一般 AWS
建立端點。	建立 Db2 資料庫的來源端點，並為 Aurora MySQL 相容資料庫建立目標端點： 1. 選擇選取 RDS 資料庫執行個體，然後選擇您建立的 Db2 執行個體，以建立 IBM Db2 的端點做為來源。會自動填入端點組態詳細資訊。 2. 在端點特定設定中，新增下列額外的連線屬性。 `CurrentLSN=<scan>; MaxKBytesPerRead=64; SetDataCaptureChanges=true` 如果您未提及這些屬性，來源端點測試連線將不會成功。如需詳細資訊，請參閱使用 IBM Db2 LUW 做為 AWS DMS 的來源。 3. 選擇選取 RDS 資料庫執行個體，然後選擇您建立的 Aurora MySQL 相容執行個體，以建立 Aurora MySQL 相容做為目標的端點。會自	一般 AWS

任務	Description	所需的技能
	動填入端點組態詳細資訊。如需詳細資訊，請參閱使用 MySQL 相容資料庫做為 AWS Database Migration Service 的目標。 4. 測試來源和目標端點。確認兩者都成功且可用 5. 如果測試失敗，請確定安全群組傳入規則有效。	

任務	Description	所需的技能
建立遷移任務。	為完全載入和 CDC 或資料驗證建立單一遷移任務或多個遷移任務： 1. 若要建立資料庫遷移任務，請選擇複寫執行個體、來源資料庫端點、目標資料庫端點。將遷移類型指定為遷移現有資料（完全載入）、僅複寫資料變更 (CDC) 或遷移現有資料，並複寫持續變更（完全載入和 CDC)。 2. 在資料表映射下，您可以設定 GUI 或 JSON 格式的選擇規則和轉換規則。 3. 在選取規則下，選取結構描述，輸入資料表名稱，然後選取要設定的動作（包含/排除）（例如，結構描述：範例；資料表名稱：%，動作：包含）。 4. 在轉換規則下，選取目標（結構描述、資料表或資料欄）。選取結構描述名稱，然後選擇動作（大小寫、字首、尾碼）；例如，目標：結構描述；`mysql-sample-db` ；動作：小寫。 5. 開啟 Amazon CloudWatch Logs 監控。	一般 AWS
規劃生產執行。	與應用程式擁有者等利益相關者確認停機時間，以在生產系統中執行 AWS DMS。	遷移潛在客戶

任務	Description	所需的技能
執行遷移任務。	1. 啟動狀態為就緒的 AWS DMS 任務。 2. 監控 Amazon CloudWatch Logs 中的遷移任務日誌是否有任何錯誤。	一般 AWS
驗證資料。	檢閱來源 Db2 和目標 MySQL 資料庫中的遷移任務結果和資料： 1. 如果狀態為載入完成持續複寫，則具有 CDC 資料遷移的完整載入已完成，且驗證正在進行中。 2. 連線至 Aurora MySQL 相容資料庫，並檢查資料。 3. 在 Db2 資料庫中插入或更新資料，以檢查進行中的變更。	DBA
停止遷移任務。	成功完成資料驗證後，停止驗證遷移任務。	一般 AWS

疑難排解

問題	解決方案
AWS SCT 來源和目標測試連線失敗。	設定 JDBC 驅動程式版本和 VPC 安全群組傳入規則，以接受傳入流量。
Db2 來源端點測試執行失敗。	設定額外的連線設定 `CurrentLSN=<scan>;` 。
AWSDMS 任務無法連線至 Db2 來源，並傳回下列錯誤。	若要避免錯誤，請執行下列命令：

問題	解決方案

database is recoverable if either or both of the database configuration parameters LOGARCHMETH1 and LOGARCHMETH2 are set to ON	1. $ db2 update db cfg for sample using LOGARCHMETH1 DISK:/home/db2inst1/logs
	2. $ db2stop
	3. $ db2start
	4. $ db2 connect to sample
	`SQL1116N A connection to or activation of database "SAMPLE" cannot be made because of BACKUP PENDING. SQLSTATE=57019`
	5. $ db2 backup database sample to ../logs
	`SQL2036N The path for the file or device "../logs" is not valid`
	6. $ cd
	7. $ pwd
	`/home/db2inst1`
	8. $ mkdir /tmp/backup
	9. $ db2 backup database sample to /tmp/backup
	`Backup successful. The timestamp for this backup image is : 20190530084921`
	10$ db2 connect to sample
	`Database Connection Information`
	`Database server = DB2/LINUX 9.7.1`
	`SQL authorization ID = DB2INST1`

問題	解決方案
	`Local database alias = SAMPLE`

相關資源

Amazon EC2

- Amazon EC2
- Amazon EC2 使用者指南

資料庫

- IBM Db2 資料庫
- Amazon Aurora
- 使用 Amazon Aurora MySQL

AWS SCT

- AWS DMS 結構描述轉換
- AWS Schema Conversion Tool使用者指南
- 使用 AWS SCT 使用者介面
- 使用 IBM Db2 LUW 做為 AWS SCT 的來源

AWS DMS

- AWS Database Migration Service
- AWS Database Migration Service 使用者指南
- 資料遷移的來源
- 資料遷移的目標
- AWS Database Migration Service 和 AWS Schema Conversion Tool 現在支援 IBM Db2 LUW 作為來源（部落格文章）
- 將執行關聯式資料庫的應用程式遷移至 AWS

使用 AWS DMS 將 Microsoft SQL Server 資料庫從 Amazon EC2 遷移至 Amazon DocumentDB

Umamaheswara Nooka，Amazon Web Services

總結

此模式說明如何使用 AWS Database Migration Service (AWS DMS) 將 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上託管的 Microsoft SQL Server 資料庫遷移至 Amazon DocumentDB（具有 MongoDB 相容性）資料庫。

AWS DMS 複寫任務會讀取 SQL Server 資料庫的資料表結構、在 Amazon DocumentDB 中建立對應的集合，以及執行完全載入遷移。

您也可以使用此模式將現場部署 SQL Server 或 Amazon Relational Database Service (Amazon RDS) for SQL Server 資料庫執行個體遷移至 Amazon DocumentDB。如需詳細資訊，請參閱 AWS 規範指引網站上的將 Microsoft SQL Server 資料庫遷移至 AWS 雲端指南。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- EC2 執行個體上現有的 SQL Server 資料庫。
- 修正 SQL Server 資料庫中指派給 AWS DMS 的資料庫 (db_owner) 角色。如需詳細資訊，請參閱 SQL Server 文件中的資料庫層級角色。
- 熟悉使用 `mongodump`、`mongoexport`、 `mongorestore`和 `mongoimport`公用程式將資料移入和移出 Amazon DocumentDB 叢集。
- Microsoft SQL Server Management Studio，已安裝並設定。

限制

- Amazon DocumentDB 中的叢集大小限制為 64 TB。如需詳細資訊，請參閱 Amazon DocumentDB 文件中的叢集限制。
- AWS DMS 不支援將多個來源資料表合併為單一 Amazon DocumentDB 集合。
- 如果 AWS DMS 在沒有主索引鍵的情況下處理來源資料表的任何變更，它會忽略來源資料表中的大型物件 (LOB) 資料欄。

Architecture

來源技術堆疊

- Amazon EC2

目標技術堆疊

- Amazon DocumentDB

目標架構

工具

- AWS DMS – AWS Database Migration Service (AWS DMS) 可協助您輕鬆安全地遷移資料庫。
- Amazon DocumentDB – Amazon DocumentDB （與 MongoDB 相容） 是一種快速、可靠且全受管的資料庫服務。
- Amazon EC2 – Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。
- Microsoft SQL Server – SQL Server 是一種關聯式資料庫管理系統。
- SQL Server Management Studio (SSMS) – SSMS 是管理 SQL Server 的工具，包括存取、設定和管理 SQL Server 元件。

史詩

建立和設定 VPC

任務	Description	所需的技能
建立 VPC。	登入 AWS 管理主控台並開啟 Amazon VPC 主控台。建立具有 IPv4 CIDR 區塊範圍的虛擬私有雲端 (VPC)。	系統管理員
建立安全群組和網路 ACLs。	在 Amazon VPC 主控台上，根據您的需求為您的 VPC 建立	系統管理員

任務	Description	所需的技能
	安全群組和網路存取控制清單（網路 ACLs)。您也可以使用這些組態的預設設定。如需此案例和其他案例的詳細資訊，請參閱「相關資源」一節。	

建立和設定 Amazon DocumentDB 叢集

任務	Description	所需的技能
建立 Amazon DocumentDB 叢集。	開啟 Amazon DocumentDB 主控台，然後選擇「叢集」。選擇「建立」，並使用一個執行個體建立 Amazon DocumentDB 叢集。重要：請務必使用 VPC 的安全群組來設定此叢集。	系統管理員
安裝 mongo shell。	mongo Shell 是一個命令行公用程式，可以使用它來連線和查詢 Amazon DocumentDB 叢集。若要安裝它，請執行 "/etc/yum.repos.d/mongodb-org-3.6.repo" 命令來建立儲存庫檔案。執行「sudo yum install -y mongodb-org-shell」命令來安裝 mongo shell。若要加密傳輸中的資料，請下載 Amazon DocumentDB 的公有金鑰，然後連接至您的 Amazon DocumentDB 執行個體。如需這些步驟的詳細資訊，請參閱「相關資源」一節。	系統管理員

任務	Description	所需的技能
在 Amazon DocumentDB 叢集中建立資料庫。	使用資料庫的名稱執行「使用」命令，在 Amazon DocumentDB 叢集中建立資料庫。	系統管理員

建立和設定 AWS DMS 複寫執行個體

任務	Description	所需的技能
建立 AWS DMS 複寫執行個體。	開啟 AWS DMS 主控台，然後選擇「建立複寫執行個體」。輸入複寫任務的名稱和描述。選擇執行個體類別、引擎版本、儲存體、VPC、異地同步備份，並使其可公開存取。選擇「進階」索引標籤來設定網路和加密設定。指定維護設定，然後選擇「建立複寫執行個體」。	系統管理員
設定 SQL Server 資料庫。	登入 Microsoft SQL Server 並新增傳入規則，以在來源端點和 AWS DMS 複寫執行個體之間進行通訊。使用複寫執行個體的私有 IP 地址做為來源。重要：複寫執行個體和目標端點應該位於相同的 VPC 上。如果來源和複寫執行個體VPCs 不同，請使用安全群組中的替代來源。	系統管理員

在 AWS DMS 中建立和測試來源和目標端點

任務	Description	所需的技能
建立來源和目標資料庫端點。	開啟 AWS DMS 主控台，然後選擇「連接來源和目標資料庫端點」。指定來源和目標資料庫的連線資訊。如有必要，請選擇「進階」索引標籤來設定「額外連線屬性」的值。在端點組態中下載並使用憑證套件。	系統管理員
測試端點連線。	選擇「執行測試」來測試連線。透過驗證安全群組設定以及來自來源和目標資料庫執行個體的 AWS DMS 複寫執行個體連線，對任何錯誤訊息進行故障診斷。	系統管理員

遷移資料

任務	Description	所需的技能
建立 AWS DMS 遷移任務。	在 AWS DMS 主控台上，選擇「任務」、「建立任務」。指定任務選項，包括來源和目的地端點名稱，以及複寫執行個體名稱。在「遷移類型」下，選擇「遷移現有資料」和「僅複寫資料變更」。選擇「開始任務」。	系統管理員
執行 AWS DMS 遷移任務。	在「任務設定」下，指定資料表準備模式的設定，例如「不執行任何動作」、「目標上的捨棄資料表」、「截斷」	系統管理員

任務	Description	所需的技能
	和「在複寫中包含 LOB 資料欄」。設定 AWS DMS 將接受的最大 LOB 大小，然後選擇「啟用記錄」。將「進階設定」保留為預設值，然後選擇「建立任務」。	
監控遷移。	在 AWS DMS 主控台上，選擇「任務」，然後選擇您的遷移任務。選擇「任務監控」來監控您的任務。完成完全載入遷移並套用快取變更時，任務會停止。	系統管理員

測試並驗證遷移

任務	Description	所需的技能
使用 mongo shell 連線至 Amazon DocumentDB 叢集。	開啟 Amazon DocumentDB 主控台，選擇「叢集」下的叢集。在「連線和安全性」索引標籤中，選擇「使用 mongo shell 連線至此叢集」。	系統管理員
驗證遷移的結果。	使用資料庫的名稱執行「使用」命令，然後執行「顯示集合」命令。使用資料庫的名稱執行 "db. .count() ; " 命令。如果結果與您的來源資料庫相符，則遷移成功。	系統管理員

相關資源

建立和設定 VPC

- 為您的 VPC 建立安全群組
- 建立網路 ACL

建立和設定 Amazon DocumentDB 叢集

- 建立 Amazon DocumentDB 叢集
- 安裝 Amazon DocumentDB 的 mongo shell
- 連接至 Amazon DocumentDB 叢集

建立和設定 AWS DMS 複寫執行個體

- 使用公有和私有複寫執行個體

在 AWS DMS 中建立和測試來源和目標端點

- 使用 Amazon DocumentDB 做為 AWS DMS 的目標
- 使用 SQL Server 資料庫做為 AWS DMS 的來源
- 使用 AWS DMS 端點

遷移資料

- 遷移至 Amazon DocumentDB

其他資源

- 使用 SQL Server 做為 AWS DMS 來源的限制
- 如何使用 Amazon DocumentDB 大規模建置和管理應用程式

將內部部署 ThoughtSpot Falcon 資料庫遷移至 Amazon Redshift

Battulga Purevragchaa 和 Antony Prasad Thevaraj，Amazon Web Services

摘要

內部部署資料倉儲需要大量的管理時間和資源，尤其是大型資料集。建置、維護和成長這些倉儲的財務成本也非常高。為了協助管理成本、降低擷取、轉換和載入 (ETL) 複雜性，並隨著資料成長提供效能，您必須持續選擇要載入的資料和要封存的資料。

透過將內部部署 ThoughtSpot Falcon 資料庫遷移至 Amazon Web Services (AWS) 雲端，除了降低整體基礎設施成本之外，您還可以存取雲端型資料湖和資料倉儲，以提高業務敏捷性、安全性和應用程式可靠性。Amazon Redshift 有助於大幅降低資料倉儲的成本和營運開銷。您也可以使用 Amazon Redshift Spectrum 來分析其原生格式的大量資料，而無需載入資料。

此模式說明將 ThoughtSpot Falcon 資料庫從現場部署資料中心遷移至 AWS 雲端上 Amazon Redshift 資料庫的步驟和程序。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在內部部署資料中心託管的 ThoughtSpot Falcon 資料庫

產品版本

- ThoughtSpot 7.0.1 版

Architecture

該圖顯示以下工作流程：

1. 資料託管在內部部署關聯式資料庫中。
2. AWS Schema Conversion Tool (AWS SCT) 會轉換與 Amazon Redshift 相容的資料定義語言 (DDL)。
3. 建立資料表之後，您可以使用 AWS Database Migration Service (AWS DMS) 遷移資料。

4. 資料會載入 Amazon Redshift。
5. 如果您使用 Redshift Spectrum 或已在 Amazon S3 中託管資料，資料會儲存在 Amazon S3) 中。

工具

- AWS DMS – AWS Data Migration Service (AWS DMS) 可協助您快速且安全地將資料庫遷移至 AWS。
- Amazon Redshift – Amazon Redshift 是一種快速、全受管的 PB 級資料倉儲服務，可讓您使用現有的商業智慧工具有效率地分析所有資料，既簡單又經濟實惠。
- AWS SCT – AWS Schema Conversion Tool (AWS SCT) 會將您現有的資料庫結構描述從一個資料庫引擎轉換為另一個資料庫引擎。

史詩

準備遷移

任務	Description	所需的技能
識別適當的 Amazon Redshift 組態。	根據您的需求和資料磁碟區識別適當的 Amazon Redshift 叢集組態。 如需詳細資訊，請參閱 Amazon Redshift 文件中的 Amazon Redshift 叢集。	DBA
研究 Amazon Redshift 以評估是否符合您的需求。	使用 Amazon Redshift FAQs來了解和評估 Amazon Redshift 是否符合您的需求。	DBA

準備目標 Amazon Redshift 叢集

任務	Description	所需的技能
建立 Amazon Redshift 叢集。	登入 AWS 管理主控台，開啟 Amazon Redshift 主控台，然	DBA

任務	Description	所需的技能
	後在虛擬私有雲端 (VPC) 中建立 Amazon Redshift 叢集。 如需詳細資訊，請參閱 Amazon Redshift 文件中的在 VPC 中建立叢集。	
為您的 Amazon Redshift 資料庫設計執行 PoC。	執行資料庫設計的概念驗證 (PoC)，以遵循 Amazon Redshift 最佳實務。 如需詳細資訊，請參閱《Amazon Redshift 文件》中的為 Amazon Redshift 執行概念驗證。	DBA
建立資料庫使用者。	在 Amazon Redshift 資料庫中建立使用者，並授予適當的角色以存取結構描述和資料表。 如需詳細資訊，請參閱《Amazon Redshift 文件》中的授予使用者或使用者群組的存取權限。	DBA
將組態設定套用至目標資料庫。	根據您的需求將組態設定套用至 Amazon Redshift 資料庫。 如需啟用資料庫、工作階段和伺服器層級參數的詳細資訊，請參閱 Amazon Redshift 文件中的組態參考。	DBA

在 Amazon Redshift 叢集中建立物件

任務	Description	所需的技能
在 Amazon Redshift 中使用 DDL 手動建立資料表。	（選用）如果您使用 AWS SCT，系統會自動建立資料表。不過，如果複寫 DDLs時發生失敗，您必須手動建立資料表	DBA
為 Redshift Spectrum 建立外部資料表。	使用 Amazon Redshift Spectrum 的外部結構描述建立外部資料表。若要建立外部資料表，您必須是外部結構描述或資料庫超級使用者的擁有者。 如需詳細資訊，請參閱《Amazon Redshift 文件》中的為 Amazon Redshift Spectrum 建立外部資料表。	DBA

使用 AWS DMS 遷移資料

任務	Description	所需的技能
使用 AWS DMS 遷移資料。	在 Amazon Redshift 資料庫中建立資料表的 DDL 之後，請使用 AWS DMS 將資料遷移至 Amazon Redshift。 如需詳細步驟和說明，請參閱 AWS DMS 文件中的使用 Amazon Redshift 資料庫做為 AWS DMS 的目標。	DBA

任務	Description	所需的技能
使用 COPY 命令載入資料。	使用 Amazon Redshift COPY命令將資料從 Amazon S3 載入至 Amazon Redshift。 如需詳細資訊，請參閱《Amazon Redshift 文件》中的使用 COPY 命令從 Amazon S3 載入。	DBA

驗證 Amazon Redshift 叢集

任務	Description	所需的技能
驗證來源和目標記錄。	驗證從來源系統載入的來源和目標記錄的資料表計數。	DBA
實作效能調校的 Amazon Redshift 最佳實務。	實作資料表和資料庫設計的 Amazon Redshift 最佳實務。 如需詳細資訊，請參閱部落格文章 Amazon Redshift 的前 10 大效能調校技術。	DBA
最佳化查詢效能。	Amazon Redshift 使用 SQL 型查詢與系統中的資料和物件互動。資料處理語言 (DML) 是 SQL 的子集，可用來檢視、新增、變更和刪除資料。DDL 是您用來新增、變更和刪除資料庫物件的 SQL 子集，例如資料表和檢視。 如需詳細資訊，請參閱 Amazon Redshift 文件中的調校查詢效能。	DBA

任務	Description	所需的技能
實作 WLM。	您可以使用工作負載管理 (WLM) 來定義多個查詢佇列，並在執行時間將查詢路由至適當的佇列。 如需詳細資訊，請參閱《Amazon Redshift 文件》中的實作工作負載管理。	DBA
使用並行擴展。	透過使用 並行擴展 功能，您可以支援幾乎無限制的並行使用者和並行查詢，並具有一致的快速查詢效能。 如需詳細資訊，請參閱《Amazon Redshift 文件》中的使用並行擴展。	DBA
使用 Amazon Redshift 最佳實務進行資料表設計。	當您規劃資料庫時，某些重要的資料表設計決策可能會大幅影響整體查詢效能。 如需選擇最適合的資料表設計選項的詳細資訊，請參閱 Amazon Redshift 文件中的設計資料表的 Amazon Redshift 最佳實務。	DBA

任務	Description	所需的技能
在 Amazon Redshift 中建立具體化視觀表。	具體化檢視包含根據一或多個基礎資料表上的 SQL 查詢預先計算的結果集。您可以發出SELECT陳述式，以與查詢資料庫中其他資料表或檢視相同的方式查詢具體化檢視。 如需詳細資訊，請參閱《Amazon Redshift 文件》中的在 Amazon Redshift 中建立具體化視觀表。	DBA
定義資料表之間的聯結。	若要在 ThoughtSpot 中同時搜尋多個資料表，您必須透過指定包含兩個資料表中相符資料的欄來定義資料表之間的聯結。這些欄代表聯結foreign key的 primary key和 。 您可以在 Amazon Redshift 或 ThoughtSpot 中使用 ALTER TABLE命令來定義它們。如需詳細資訊，請參閱 Amazon Redshift 文件中的 ALTER TABLE。	DBA

設定與 Amazon Redshift 的 ThoughtSpot 連線

任務	Description	所需的技能
新增 Amazon Redshift 連線。	將 Amazon Redshift 連線新增至您的內部部署 ThoughtSpot Falcon 資料庫。	DBA

任務	Description	所需的技能
	如需詳細資訊，請參閱 ThoughtSpot 文件中的新增 Amazon Redshift 連線。	
編輯 Amazon Redshift 連線。	您可以 編輯 Amazon Redshift 連線來新增資料表和資料欄。 如需詳細資訊，請參閱 ThoughtSpot 文件中的編輯 Amazon Redshift 連線。	DBA
重新映射 Amazon Redshift 連線。	透過編輯新增 Amazon Redshift 連線時建立的來源映射 .yaml 檔案來修改連線參數。 例如，您可以將現有的資料表或資料欄重新映射至現有資料庫連線中的不同資料表或資料欄。ThoughtSpot 建議您在重新映射連線中的資料表或資料欄之前和之後檢查相依性，以確保它們視需要顯示。 如需詳細資訊，請參閱 ThoughtSpot 文件中的重新對應 Amazon Redshift 連線。	DBA

任務	Description	所需的技能
從 Amazon Redshift 連線刪除資料表。	（選用）如果您嘗試移除 Amazon Redshift 連線中的資料表，ThoughtSpot 會檢查相依性，並顯示相依物件的清單。您可以選擇列出的物件來刪除它們，或移除相依性。然後，您可以移除資料表。 如需詳細資訊，請參閱 ThoughtSpot 文件中的從 Amazon Redshift 連線刪除資料表。	DBA
從 Amazon Redshift 連線刪除具有相依物件的資料表。	（選用）如果您嘗試刪除具有相依物件的資料表，操作會遭到封鎖。`Cannot delete` 視窗隨即出現，其中包含相依物件的連結清單。移除所有相依性後，您就可以刪除資料表 如需詳細資訊，請參閱 ThoughtSpot 文件中的從 Amazon Redshift 連線刪除具有相依物件的資料表。	DBA
刪除 Amazon Redshift 連線。	（選用）由於連線可用於多個資料來源或視覺化效果，您必須先刪除使用該連線的所有來源和任務，才能刪除 Amazon Redshift 連線。 如需詳細資訊，請參閱 ThoughtSpot 文件中的刪除 Amazon Redshift 連線。	DBA

任務	Description	所需的技能
檢查 Amazon Redshift 的連線參考。	請務必使用 ThoughtSpot 文件中的連線參考，提供 Amazon Redshift 連線所需的資訊。	DBA

其他資訊

- 使用 ThoughtSpot 和 Amazon Redshift 進行任何規模的 AI 驅動分析
- Amazon Redshift 定價
- AWS SCT 入門
- Amazon Redshift 入門
- 使用資料擷取代理程式
- Chick-fil-A 使用 ThoughtSpot 和 AWS 改善洞見的速度

使用 Oracle GoldenGate 從 Oracle 資料庫遷移至 Amazon RDS for PostgreSQL

Dhairya Jindani、Sindhusha Paturu 和 Rajeshkumar Sabankar，Amazon Web Services

總結

此模式說明如何使用 Oracle Cloud Infrastructure (OCI) GoldenGate 將 Oracle 資料庫遷移至 PostgreSQL 的 Amazon Relational Database Service (Amazon RDS)。

透過使用 Oracle GoldenGate，您可以在來源資料庫與一或多個目的地資料庫之間複寫資料，並將停機時間降至最低。

> **Note**
>
> 來源 Oracle 資料庫可以是內部部署或 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。使用內部部署複寫工具時，您可以使用類似的程序。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Oracle GoldenGate 授權
- 連接至 PostgreSQL 資料庫的 Java Database Connectivity (JDBC) 驅動程式
- 在目標 Amazon RDS for PostgreSQL 資料庫上使用 AWS 結構描述Schema Conversion Tool和資料表

限制

- Oracle GoldenGate 只能複寫現有的資料表資料 （初始載入） 和持續變更 （變更資料擷取）

產品版本

- Oracle Database Enterprise Edition 10g 或更新版本
- OracleGoldenGate12.2.0.1.1 for Oracle 或更新版本
- 適用於 PostgreSQLor 較新版本的 OracleGoldenGate12.2.0.1.1

Architecture

下圖顯示使用 Oracle GoldenGate 將 Oracle 資料庫遷移至 Amazon RDS for PostgreSQL 的範例工作流程：

該圖顯示以下工作流程：

1. Oracle GoldenGate 擷取程序會針對來源資料庫執行，以擷取資料。
2. Oracle GoldenGate Replicat 程序會將擷取的資料交付至目標 Amazon RDS for PostgreSQL 資料庫。

工具

- Oracle GoldenGate 可協助您在 Oracle Cloud Infrastructure 中設計、執行、協調和監控資料複寫和串流資料處理解決方案。
- 適用於 PostgreSQL 的 Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展 PostgreSQL 關聯式資料庫。

史詩

下載並安裝 Oracle GoldenGate

任務	Description	所需的技能
下載 Oracle GoldenGate。	下載下列版本的 Oracle GoldenGate： • OracleGoldenGate12.2.0.1.1 for Oracle 或更新版本 • 適用於 PostgreSQL 或較新版本的 OracleGoldenGate12.2.0.1.1	DBA

任務	Description	所需的技能
	若要下載軟體，請參閱 Oracle 網站上的 Oracle GoldenGate Downloads。	
在來源 Oracle 資料庫伺服器上安裝 Oracle GoldenGate for Oracle。	如需說明，請參閱 Oracle GoldenGate 文件。	DBA
在 Amazon EC2 執行個體上安裝 Oracle GoldenGate for PostgreSQL 資料庫。	如需說明，請參閱 Oracle GoldenGate 文件。	DBA

在來源和目標資料庫上設定 Oracle GoldenGate

任務	Description	所需的技能
在來源資料庫上設定 Oracle GoldenGate for Oracle 資料庫。	如需說明，請參閱 Oracle GoldenGate 文件。 請務必設定下列項目： • 補充記錄 • Oracle GoldenGate 使用者 • 任何必要的授予和許可 • 參數檔案 • 管理員程序 • 目錄 • GLOBALS 檔案 • Oracle 錢包	DBA
在目標資料庫上設定 Oracle GoldenGate for PostgreSQL。	如需說明，請參閱 Oracle 網站上的使用 Oracle GoldenGate for PostgreSQL 的第 VI 部分。 請務必設定下列項目：	DBA

任務	Description	所需的技能
	• 管理員程序 • GLOBALS 檔案 • Oracle 錢包	

設定資料擷取

任務	Description	所需的技能
在來源資料庫中設定擷取程序。	在來源 Oracle 資料庫中，建立擷取檔案以擷取資料。 如需說明，請參閱 Oracle 文件中的新增 EXTRACT。 **Note** 擷取檔案包含建立擷取參數檔案和追蹤檔案目錄。	DBA
設定資料幫浦，將追蹤檔案從來源傳輸到目標資料庫。	遵循 Oracle 網站上的資料庫公用程式 PARFILE 中的指示，建立 EXTRACT 參數檔案和線索檔案目錄。 如需詳細資訊，請參閱 Oracle 網站上的 Fusion Middlewar e Understanding Oracle GoldenGate 中的什麼是線索？。	DBA
在 Amazon EC2 執行個體上設定複寫。	建立複寫參數檔案和追蹤檔案目錄。	DBA

任務	Description	所需的技能
	如需建立複寫參數檔案的詳細資訊，請參閱 Oracle 資料庫文件中的第 3.5 節驗證參數檔案。 如需建立追蹤檔案目錄的詳細資訊，請參閱 Oracle Cloud 文件中的建立追蹤。 **Important** 請務必在目標的 GLOBALS 檔案中新增檢查點資料表項目。 如需詳細資訊，請參閱 Oracle 網站上的 Fusion Middlewar e Understanding Oracle GoldenGate 中的什麼是複本？。	

設定資料複寫

任務	Description	所需的技能
在來源資料庫中，建立參數檔案以擷取初始載入的資料。	遵循 Oracle Cloud 文件中在 GGSCI 中建立參數檔案的指示。 **Important** 確定 Manager 正在目標上執行。	DBA

任務	Description	所需的技能
在目標資料庫中，建立參數檔案以複寫初始載入的資料。	遵循 Oracle Cloud 文件中在 GGSCI 中建立參數檔案的指示。 **Important** 請務必新增並啟動複寫程序。	DBA

切換到 Amazon RDS for PostgreSQL 資料庫

任務	Description	所需的技能
停止複寫程序，並確保來源和目標資料庫處於同步狀態。	比較來源和目標資料庫之間的資料列計數，以確保資料複寫成功。	DBA
設定資料定義語言 (DDL) 支援。	執行 DDL 指令碼，以在 PostgreSQL 上建立觸發條件、序列、同義詞和參考金鑰。 **Note** 您可以使用任何標準 SQL 用戶端應用程式來連線至資料庫叢集中的資料庫。例如，您可以使用 pgAdmin 連線到資料庫執行個體。	DBA

相關資源

- Amazon RDS for PostgreSQL (Amazon RDS 使用者指南)
- Amazon EC2 文件
- Oracle GoldenGate 支援的處理方法和資料庫 (Oracle 文件)

使用 AWS DMS 將 Oracle 分割的資料表遷移至 PostgreSQL

Saurav Mishra 和 Eduardo Valentim，Amazon Web Services

總結

此模式說明如何使用不支援原生分割的 AWS Database Migration Service (AWS DMS)，加速將分割的資料表從 Oracle 載入 PostgreSQL。目標 PostgreSQL 資料庫可以安裝在 Amazon Elastic Compute Cloud (Amazon EC2) 上，也可以是 PostgreSQL 的 Amazon Relational Database Service (Amazon RDS) 或 Amazon Aurora PostgreSQL 相容版本資料庫執行個體。

上傳分割的資料表包含下列步驟：

1. 建立類似於 Oracle 分割區資料表的父資料表，但不包含任何分割區。
2. 建立將繼承自您在步驟 1 中建立之父資料表的子資料表。
3. 建立程序函數和觸發程序來處理父資料表中的插入。

不過，由於每次插入都會觸發觸發，因此使用 AWS DMS 的初始負載可能會非常慢。

為了加速從 Oracle 到 PostgreSQL 9.0 的初始載入，此模式會為每個分割區建立個別的 AWS DMS 任務，並載入對應的子資料表。然後，您可以在切換期間建立觸發。

PostgreSQL 第 10 版支援原生分割。不過，在某些情況下，您可能會決定使用繼承的分割。如需詳細資訊，請參閱其他資訊一節。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 具有分割資料表的來源 Oracle 資料庫
- AWS 上的 PostgreSQL 資料庫

產品版本

- PostgreSQL 9.0

Architecture

來源技術堆疊

- Oracle 中的分割資料表

目標技術堆疊

- PostgreSQL 中的分割資料表 （ 在 Amazon EC2、Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 上 ）

目標架構

工具

- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。

史詩

設定 AWS DMS

任務	Description	所需的技能
在 PostgreSQL 中建立資料表。	在 PostgreSQL 中建立父資料表和對應的子資料表，其中包含分割區所需的檢查條件。	DBA
為每個分割區建立 AWS DMS 任務。	在 AWS DMS 任務中包含分割區的篩選條件。將分割區對應至對應的 PostgreSQL 子資料表。	DBA
使用完全載入和變更資料擷取 (CDC) 執行 AWS DMS 任務。	請確定 `StopTaskCachedChangesApplied` 參數設定為 `true`，且 `StopTaskCachedChangesNotApplied` 參數設定為 `false`。	DBA

剪下

任務	Description	所需的技能
停止複寫任務。	在您停止任務之前，請確認來源和目的地是同步的。	DBA
在父資料表上建立觸發。	由於父資料表會收到所有插入和更新命令，請建立觸發程序，根據分割條件將這些命令路由至個別的子資料表。	DBA

相關資源

- AWS DMS
- 資料表分割 (PostgreSQL 文件)

其他資訊

雖然 PostgreSQL 第 10 版支援原生分割區，但您可能會決定將繼承的分割區用於下列使用案例：

- 分割會強制執行規則，表示所有分割區都必須具有與父系相同的資料欄集，但資料表繼承支援具有額外資料欄的子系。
- 資料表繼承支援多個繼承。
- 宣告式分割僅支援清單和範圍分割。使用資料表繼承，您可以根據需要分割資料。不過，如果限制排除無法有效剔除分割區，則查詢效能會受到影響。
- 與使用資料表繼承相比，某些操作在使用宣告式分割時需要更強大的鎖定。例如，在分割資料表中新增或移除分割區時，需要在父資料表上`ACCESS EXCLUSIVE`鎖定，而`SHARE UPDATE EXCLUSIVE`鎖定足以進行一般繼承。

使用個別任務分割區時，如果有任何 AWS DMS 驗證問題，您也可以重新載入分割區。為了獲得更好的效能和複寫控制，請在不同的複寫執行個體上執行任務。

從 Amazon RDS for Oracle 遷移至 Amazon RDS for MySQL

Jitender Kumar、Srini Ramaswamy 和 Neha Sharma，Amazon Web Services

總結

此模式提供將 Amazon Relational Database Service (Amazon RDS) for Oracle 資料庫執行個體遷移至 Amazon Web Services (AWS) 上 Amazon RDS for MySQL 資料庫執行個體的指引。模式使用 AWS Database Migration Service (AWS DMS) 和 AWS Schema Conversion Tool (AWS SCT)。

模式提供處理預存程序遷移的最佳實務。它還涵蓋 和程式碼變更，以支援應用程式層。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Amazon RDS for Oracle 來源資料庫。
- Amazon RDS for MySQL 目標資料庫。來源和目標資料庫應位於相同的虛擬私有雲端 (VPC) 中。如果您使用的是多個 VPCs，或者您必須擁有必要的存取許可。
- 允許來源和目標資料庫、AWS SCT、應用程式伺服器和 AWS DMS 之間連線的安全群組。
- 具有在來源資料庫上執行 AWS SCT 所需權限的使用者帳戶。
- 啟用補充記錄，以在來源資料庫上執行 AWS DMS。

限制

- 來源和目標 Amazon RDS 資料庫大小限制為 64 TB。如需 Amazon RDS 大小資訊，請參閱 AWS 文件。
- Oracle 對資料庫物件不區分大小寫，但 MySQL 不區分大小寫。AWS SCT 可以在建立物件時處理此問題。不過，需要一些手動工作才能支援全案例不敏感。
- 此遷移不會使用 MySQL 擴充功能來啟用 Oracle 原生函數。AWS SCT 會處理大多數轉換，但需要一些工作才能手動變更程式碼。
- 應用程式中需要 Java Database Connectivity (JDBC) 驅動程式變更。

產品版本

- Amazon RDS for Oracle 12.2.0.1 及更新版本。如需目前支援的 RDS for Oracle 版本，請參閱 AWS 文件。

- Amazon RDS for MySQL 8.0.15 及更新版本。如需目前支援的 RDS for MySQL 版本，請參閱 AWS 文件。
- AWS DMS 3.3.0 版及更新版本。如需 AWS DMS 支援的來源端點和目標端點的詳細資訊，請參閱 AWS 文件。
- AWS SCT 1.0.628 版及更新版本。 請參閱 AWS 文件中的 AWS SCT 來源和目標端點支援矩陣。

Architecture

來源技術堆疊

- Amazon RDS for Oracle。如需詳細資訊，請參閱使用 Oracle 資料庫做為 AWS DMS 的來源。

目標技術堆疊

- Amazon RDS for MySQL。如需詳細資訊，請參閱使用 MySQL 相容資料庫做為 AWS DMS 的目標。

遷移架構

在下圖中，AWS SCT 會從 Amazon RDS for Oracle 來源資料庫複製和轉換結構描述物件，並將物件傳送至 Amazon RDS for MySQL 目標資料庫。AWS DMS 會從來源資料庫複寫資料，並將其傳送至 Amazon RDS for MySQL 執行個體。

工具

- AWS Data Migration Service 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間進行遷移。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。此模式使用 Amazon RDS for Oracle 和 Amazon RDS for MySQL。
- AWS Schema Conversion Tool (AWS SCT) 會自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，以支援異質資料庫遷移。

史詩

準備遷移

任務	Description	所需的技能
驗證來源和目標資料庫版本和引擎。		DBA
識別目標伺服器執行個體的硬體需求。		DBA、SysAdmin
識別儲存需求（儲存類型和容量）。		DBA、SysAdmin
選擇適當的執行個體類型（容量、儲存功能、網路功能）。		DBA、SysAdmin
識別來源和目標資料庫的網路存取安全需求。		DBA、SysAdmin
選擇應用程式遷移策略。	考慮您是否希望將完全停機時間或部分停機時間用於切換活動。	DBA、SysAdmin、應用程式擁有者

設定基礎設施

任務	Description	所需的技能
建立 VPC 和子網路。		SysAdmin
建立安全群組和網路存取控制清單 ACLs)。		SysAdmin
設定和啟動 Amazon RDS for Oracle 執行個體。		DBA、SysAdmin
設定和啟動 Amazon RDS for MySQL 執行個體。		DBA、SysAdmin

任務	Description	所需的技能
準備測試案例以驗證程式碼轉換。	這將有助於對轉換後的程式碼進行單位測試。	DBA、開發人員
設定 AWS DMS 執行個體。		
在 AWS DMS 中設定來源和目標端點。		

遷移資料

任務	Description	所需的技能
使用 AWS SCT 產生目標資料庫指令碼。	檢查 AWS SCT 轉換之程式碼的準確性。需要一些手動工作。	DBA、開發人員
在 AWS SCT 中，選擇「不區分大小寫」設定。	在 AWS SCT 中，選擇專案設定、目標案例敏感度、不區分大小寫。	DBA、開發人員
在 AWS SCT 中，選擇不使用 Oracle 原生函數。	在專案設定中，檢查函數 TO_CHAR/TO_NUMBER/ TO_DATE。	DBA、開發人員
變更 "sql%notfound" 程式碼。	您可能需要手動轉換程式碼。	
查詢預存程序中的資料表和物件（使用小寫查詢）。		DBA、開發人員
完成所有變更後建立主要指令碼，然後在目標資料庫上部署主要指令碼。		DBA、開發人員
使用範例資料對預存程序和應用程式呼叫進行單元測試。		

任務	Description	所需的技能
清除在單位測試期間建立的資料。		DBA、開發人員
捨棄目標資料庫上的外部金鑰限制。	載入初始資料需要此步驟。如果您不想捨棄外部金鑰限制，您必須為主要和次要資料表特定的資料建立遷移任務。	DBA、開發人員
在目標資料庫上捨棄主索引鍵和唯一索引鍵。	此步驟可為初始載入提供更好的效能。	DBA、開發人員
在來源資料庫上啟用補充記錄。		DBA
在 AWS DMS 中建立初始載入的遷移任務，然後執行它。	選擇 選項以遷移現有資料。	DBA
將主索引鍵和外部索引鍵新增至目標資料庫。	初始載入後需要新增限制條件。	DBA、開發人員
建立遷移任務以進行持續複寫。	持續複寫可讓目標資料庫與來源資料庫保持同步。	DBA

遷移應用程式

任務	Description	所需的技能
將 Oracle 原生函數取代為 MySQL 原生函數。		應用程式擁有者
請確定 SQL 查詢中的資料庫物件只使用小寫名稱。		DBA、SysAdmin、應用程式擁有者

切換到目標資料庫

任務	Description	所需的技能
關閉應用程式伺服器。		應用程式擁有者
驗證來源和目標資料庫是否同步。		DBA、應用程式擁有者
停止 Amazon RDS for Oracle 資料庫執行個體。		DBA
停止遷移任務。	這會在您完成上一個步驟後自動停止。	DBA
將 JDBC 連線從 Oracle 變更為 MySQL。		應用程式擁有者、DBA
啟動應用程式。		DBA、SysAdmin、應用程式擁有者

關閉專案

任務	Description	所需的技能
檢閱並驗證專案文件。		DBA、SysAdmin
收集遷移時間、手動與工具任務的百分比、節省成本等指標。		DBA、SysAdmin
停止和刪除 AWS DMS 執行個體。		DBA
移除來源和目標端點。		DBA
移除遷移任務。		DBA

任務	Description	所需的技能
拍攝 Amazon RDS for Oracle 資料庫執行個體的快照。		DBA
刪除 Amazon RDS for Oracle 資料庫執行個體。		DBA
關閉並刪除您使用的任何其他臨時 AWS 資源。		DBA、SysAdmin
關閉專案並提供任何意見回饋。		DBA

相關資源

- AWS DMS
- AWS SCT
- Amazon RDS 定價
- AWS DMS 入門
- Amazon RDS 入門

使用 AWS DMS 和 AWS SCT 從 Amazon EC2 上的 IBM Db2 遷移至 Aurora PostgreSQL 相容 Amazon EC2

Sirsendu Halder 和 Abhimanyu Chhabra，Amazon Web Services

總結

此模式提供將 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的 IBM Db2 資料庫遷移至 Amazon Aurora PostgreSQL 相容版本資料庫執行個體的指引。此模式使用 AWS Database Migration Service (AWS DMS) 和 AWS Schema Conversion Tool (AWS SCT) 進行資料遷移和結構描述轉換。

此模式針對具有大量交易的多 TB IBM Db2 資料庫，以幾乎沒有停機時間的線上遷移策略為目標。建議您將主索引鍵 PKs) 和外部索引鍵 (FKs) 中的資料欄與資料類型轉換為 PostgreSQL `NUMERIC`或 `INT` `BIGINT`，以獲得更好的效能。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- EC2 執行個體上的來源 IBM Db2 資料庫 EC2

產品版本

- DB2/LINUXX8664 11.1.4.4 版及更新版本

Architecture

來源技術堆疊

- EC2 執行個體上的 Db2 資料庫 EC2

目標技術堆疊

- Aurora PostgreSQL 相容版本 10.18 或更新版本的資料庫執行個體

資料庫遷移架構

工具

- AWS Database Migration Service (AWS DMS) 可協助您將資料庫遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。來源資料庫在遷移期間保持完全運作，將依賴資料庫的應用程式停機時間降至最低。您可以使用 AWS DMS 在最廣泛使用的商業和開放原始碼資料庫之間遷移資料。AWS DMS 支援在不同資料庫平台之間進行異質遷移，例如 IBM Db2 到 Aurora PostgreSQL 相容版本 10.18 或更新版本。如需詳細資訊，請參閱 AWS DMS 文件中的資料遷移來源和資料遷移目標。
- AWS Schema Conversion Tool (AWS SCT) 透過自動將來源資料庫結構描述和大部分資料庫程式碼物件，包括檢視、預存程序和函數，轉換為與目標資料庫相容的格式，來支援異質資料庫遷移。任何未自動轉換的物件都會清楚標示，以便手動轉換以完成遷移。AWS SCT 也可以掃描內嵌 SQL 陳述式的應用程式原始碼，並進行轉換。

史詩

設定環境

任務	Description	所需的技能
建立 Aurora PostgreSQL 相容資料庫執行個體。	若要建立資料庫執行個體，請遵循 AWS 文件中的指示。針對引擎類型，選擇 Amazon Aurora。針對版本，選擇 Amazon Aurora PostgreSQL 相容版本。 Aurora PostgreSQL 相容版本 10.18 或更新版本的資料庫執行個體應與來源 IBM Db2 資料庫位於相同的虛擬私有雲端 (VPC) 中。	Amazon RDS

轉換資料庫結構描述

任務	Description	所需的技能
安裝並驗證 AWS SCT。	1. 遵循 AWS SCT 文件中的步驟安裝 AWS SCT。	AWS 管理員、DBA、遷移工程師

任務	Description	所需的技能
	2. 遵循 AWS SCT 文件中的程序來驗證安裝。	
啟動 AWS SCT 並建立專案。	若要啟動 AWS SCT 工具並建立新專案以執行資料庫遷移評估報告，請遵循 AWS SCT 文件中的指示。	遷移工程師
新增資料庫伺服器並建立映射規則。	1. 遵循 AWS SCT 文件中的指示新增來源和目標資料庫伺服器。 2. 建立映射規則來定義來源資料庫的目標資料庫平台。如需說明，請參閱 AWS SCT 文件。	遷移工程師
建立資料庫遷移評估報告。	依照 AWS SCT 文件中的步驟建立資料庫遷移評估報告。	遷移工程師
檢視評估報告。	使用資料庫遷移評估報告的摘要索引標籤來檢視報告並分析資料。此分析將協助您判斷遷移的複雜性。如需詳細資訊，請參閱 AWS SCT 文件。	遷移工程師

任務	Description	所需的技能
轉換結構描述。	若要轉換來源資料庫結構描述： 1. 在 AWS SCT 主控台上，選擇檢視，然後選擇主檢視。 2. 從來源結構描述中選取物件或父節點，開啟內容（按一下滑鼠右鍵）選單，然後選擇轉換結構描述。 如需詳細資訊，請參閱 AWS SCT 文件。	遷移工程師
將轉換後的資料庫結構描述套用至目標資料庫執行個體。	1. 在顯示您目標資料庫執行個體之計劃結構描述的專案右側面板中，選擇結構描述元素。 2. 開啟結構描述元素的內容 (按一下右鍵) 選單，然後選擇 Apply to database (套用至資料庫)。 如需詳細資訊，請參閱 AWS SCT 文件。	遷移工程師

遷移您的資料

任務	Description	所需的技能
設定 VPC 和資料庫參數群組。	設定 VPC 和資料庫參數群組，並設定遷移所需的傳入規則和參數。如需說明，請參閱 AWS DMS 文件。	遷移工程師

任務	Description	所需的技能
	針對 VPC 安全群組，選取 Db2 的 EC2 執行個體和 Aurora PostgreSQL 相容資料庫執行個體。此複寫執行個體必須與來源和目標資料庫執行個體位於相同的 VPC 中。	
準備來源和目標資料庫執行個體。	準備來源和目標資料庫執行個體以進行遷移。在生產環境中，來源資料庫將已存在。 對於來源資料庫，伺服器名稱必須是執行 Db2 之 EC2 執行個體的公有網域名稱系統 (DNS)。對於使用者名稱，您可以使用 `db2inst1`，後面接著連接埠，這會是 IBM Db2 的 5000。	遷移工程師

任務	Description	所需的技能
建立 Amazon EC2 用戶端和端點。	1. 建立 Amazon EC2 用戶端。您可以使用此用戶端將要複寫的資料填入來源資料庫。您也可以使用此用戶端，透過在目標資料庫上執行查詢來驗證複寫。 2. 為來源資料庫和目標資料庫執行個體建立端點，以用於後續步驟。如需說明，請參閱 AWS DMS 文件。您必須為來源和目標資料庫建立個別的端點。對於 Aurora PostgreSQL 相容版本 10.18 或更新版本，連接埠將為 5432，而且您可以從資料庫執行個體的端點取得伺服器名稱。	遷移工程師
建立複寫執行個體。	使用 AWS DMS 主控台建立複寫執行個體，並指定來源和目標端點。複寫執行個體會在端點之間執行資料遷移。如需詳細資訊，請參閱 AWS DMS 文件。	遷移工程師

任務	Description	所需的技能
建立 AWS DMS 任務以遷移資料。	依照 AWS DMS 文件中的步驟建立任務，將來源 IBM Db2 資料表載入目標 PostgreSQL 資料庫執行個體。 • 對於來源和目標，請使用來源和目的地端點名稱。 • 遷移類型可以是完全載入。 • 對於結構描述規則，您可以使用 Db2 資料庫的`inst1`結構描述。 • 針對資料表名稱，指定 `%`來遷移所有資料表。當載入完成時，您會看到結構描述的 Db2 `inst1` 資料表出現在 Aurora PostgreSQL 相容資料庫中。	遷移工程師

相關資源

參考

- Amazon Aurora 文件
- PostgreSQL 外部資料包裝函式 (FDW) 文件
- PostgreSQL IMPORT FOREIGN SCHEMA 文件
- AWS DMS 文件
- AWS SCT 文件

教學課程和影片

- AWS DMS 入門（逐步解說）
- Amazon EC2 簡介 - Elastic Cloud Server & Hosting with AWS（影片）

使用 SharePlex 和 AWS DMS 從 Oracle 8i 或 9i 遷移至 Amazon RDS for PostgreSQL

Kumar Babu P G，Amazon Web Services

總結

此模式說明如何將內部部署 Oracle 8i 或 9i 資料庫遷移至 Amazon Relational Database Service (Amazon RDS) for PostgreSQL 或 Amazon Aurora PostgreSQL。AWS Database Migration Service (AWS DMS) 不支援 Oracle 8i 或 9i 作為來源，因此 Quest SharePlex 會將內部部署 8i 或 9i 資料庫的資料複寫到與 AWS DMS 相容的中繼 Oracle 資料庫 (Oracle 10g 或 11g)。

從中繼 Oracle 執行個體，使用 AWS Schema Conversion Tool (AWS SCT) 和 AWS DMS，將結構描述和資料遷移至 AWS 上的 PostgreSQL 資料庫。此方法有助於將資料從來源 Oracle 資料庫持續串流到具有最小複寫延遲的目標 PostgreSQL 資料庫執行個體。在此實作中，停機時間僅限於建立或驗證目標 PostgreSQL 資料庫上所有外部金鑰、觸發條件和序列所需的時間長度。

遷移使用已安裝 Oracle 10g 或 11g 的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體來託管來源 Oracle 資料庫的變更。AWS DMS 使用此中繼 Oracle 執行個體做為來源，將資料串流至 Amazon RDS for PostgreSQL 或 Aurora PostgreSQL。資料複寫可以從現場部署 Oracle 資料庫暫停並繼續到中繼 Oracle 執行個體。它也可以暫停並繼續從中繼 Oracle 執行個體到目標 PostgreSQL 資料庫，以便您可以使用 AWS DMS 資料驗證或自訂資料驗證工具來驗證資料。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署資料中心中的來源 Oracle 8i 或 9i 資料庫
- 在內部部署資料中心和 AWS 之間設定的 AWS Direct Connect
- 在本機電腦或安裝 AWS SCT 的 EC2 執行個體上安裝的 AWS SCT 連接器的 Java Database Connectivity (JDBC) 驅動程式
- 熟悉使用 Oracle 資料庫做為 AWS DMS 來源
- 熟悉使用 PostgreSQL 資料庫做為 AWS DMS 目標
- 熟悉 Quest SharePlex 資料複寫

限制

- 資料庫大小限制為 64 TB

- 內部部署 Oracle 資料庫必須是 Enterprise Edition

產品版本

- 來源資料庫的 Oracle 8i 或 9i
- 適用於中繼資料庫的 Oracle 10g 或 11g
- PostgreSQL 9.6 或更新版本

Architecture

來源技術堆疊

- Oracle 8i 或 9i 資料庫
- Quest SharePlex

目標技術堆疊

- Amazon RDS for PostgreSQL 或 Aurora PostgreSQL

來源和目標架構

工具

- AWS DMS – AWS Database Migration Service (AWS DMS) 可協助您快速安全地遷移資料庫。來源資料庫在遷移期間保持完全運作，將依賴資料庫的應用程式停機時間降到最低。AWS DMS 可以在最廣泛使用的商業和開放原始碼資料庫之間遷移您的資料。
- AWS SCT – AWS Schema Conversion Tool (AWS SCT) 會自動將來源資料庫結構描述和大部分資料庫程式碼物件，包括檢視、預存程序和函數，轉換為與目標資料庫相容的格式，讓異質資料庫遷移成為可預測。無法自動轉換的物件會清楚標示，以便手動轉換以完成遷移。AWS SCT 也可以掃描應用程式原始碼以取得內嵌 SQL 陳述式，並將其轉換為資料庫結構描述轉換專案的一部分。在此過程中，AWS SCT 會透過將舊版 Oracle 和 SQL Server 函數轉換為其 AWS 對等函數來執行雲端原生程式碼最佳化，協助您在遷移資料庫的同時現代化應用程式。當結構描述轉換完成時，AWS SCT 可以使用內建的資料遷移代理程式，協助將資料從各種資料倉儲遷移到 Amazon Redshift。

- Quest SharePlex – Quest SharePlex 是一種 Oracle-to-Oracle 資料複寫工具，可在最短的停機時間下移動資料，而不會遺失資料。

史詩

建立 EC2 執行個體並安裝 Oracle

任務	Description	所需的技能
設定 Amazon EC2 的網路。	建立虛擬私有雲端 (VPC)、子網路、網際網路閘道、路由表和安全群組。	AWS SysAdmin
建立新的 EC2 執行個體。	選取 EC2 執行個體的 Amazon Machine Image (AMI)。選擇執行個體大小並設定執行個體詳細資訊：執行個體數量 (1)、上一個步驟的 VPC 和子網路、自動指派公有 IP 和其他選項。新增儲存體、設定安全群組，以及啟動執行個體。出現提示時，請建立並儲存下一個步驟的金鑰對。	AWS SysAdmin
在 EC2 執行個體上安裝 Oracle。	取得授權和必要的 Oracle 二進位檔，並在 EC2 執行個體上安裝 Oracle 10g 或 11g。	DBA

在 EC2 執行個體上設定 SharePlex 並設定資料複寫

任務	Description	所需的技能
設定 SharePlex。	建立 Amazon EC2 執行個體並安裝與 Oracle 8i 或 9i 相容的 SharePlex 二進位檔。	AWS SysAdmin、DBA

任務	Description	所需的技能
設定資料複寫。	遵循 SharePlex 最佳實務，設定從內部部署 Oracle 8i/9i 資料庫到 Oracle 10g/11g 執行個體的資料複寫。	DBA

將 Oracle 資料庫結構描述轉換為 PostgreSQL

任務	Description	所需的技能
設定 AWS SCT。	建立新的報告，然後連接至 Oracle 做為來源，而 PostgreSQL 做為目標。在專案設定中，開啟 SQL 指令碼索引標籤，並將目標 SQL 指令碼變更為多個檔案。	DBA
轉換 Oracle 資料庫結構描述。	在動作索引標籤中，選擇產生報告、轉換結構描述，然後儲存為 SQL。	DBA
修改 AWS SCT 產生的 SQL 指令碼。		DBA

建立和設定 Amazon RDS 資料庫執行個體

任務	Description	所需的技能
建立 Amazon RDS 資料庫執行個體。	在 Amazon RDS 主控台中，建立新的 PostgreSQL 資料庫執行個體。	AWS SysAdmin、DBA
設定資料庫執行個體。	指定資料庫引擎版本、資料庫執行個體類別、異地同步備份部署、儲存類型和配置的儲	AWS SysAdmin、DBA

任務	Description	所需的技能
	存。輸入資料庫執行個體識別符、主要使用者名稱和主要密碼。	
設定網路和安全性。	指定 VPC、子網路群組、公有可存取性、可用區域偏好設定和安全群組。	AWS SysAdmin、DBA
設定資料庫選項。	指定資料庫名稱、連接埠、參數群組、加密和主金鑰。	AWS SysAdmin、DBA
設定備份。	指定備份保留期、備份時段、開始時間、持續時間，以及是否要將標籤複製到快照。	AWS SysAdmin、DBA
設定監控選項。	啟用或停用增強型監控和效能洞察。	AWS SysAdmin、DBA
設定維護選項。	指定自動次要版本升級、維護時段，以及開始日期、時間和持續時間。	AWS SysAdmin、DBA
從 AWS SCT 執行預遷移指令碼。	在 Amazon RDS 執行個體上執行這些指令碼：create_database.sql、create_sequence.sql、create_table.sql、create_view.sql 和 create_function.sql。	AWS SysAdmin、DBA

使用 AWS DMS 遷移資料

任務	Description	所需的技能
在 AWS DMS 中建立複寫執行個體。	完成名稱、執行個體類別、VPC （與 EC2 執行個體相同）、異地同步備份和	AWS SysAdmin、DBA

任務	Description	所需的技能
	公有可存取性的欄位。在進階組態區段中，指定配置的儲存、子網路群組、可用區域、VPC 安全群組和 AWS Key Management Service (AWS KMS) 根金鑰。	
建立來源資料庫端點。	指定端點名稱、類型、來源引擎 (Oracle)、伺服器名稱 (Amazon EC2 私有 DNS 名稱）、連接埠、SSL 模式、使用者名稱、密碼、SID、VPC （指定具有複寫執行個體的 VPC) 和複寫執行個體。若要測試連線，請選擇執行測試，然後建立端點。您也可以設定下列進階設定：maxFileSize 和 numberDataTypeScale。	AWS SysAdmin、DBA
建立 AWS DMS 複寫任務。	指定任務名稱、複寫執行個體、來源和目標端點，以及複寫執行個體。針對遷移類型，選擇「遷移現有資料並複寫持續變更」。清除「建立時啟動任務」核取方塊。	AWS SysAdmin、DBA
設定 AWS DMS 複寫任務設定。	針對目標資料表準備模式，選擇「不執行任何動作」。在完全載入完成後停止任務，以建立主索引鍵。指定有限或完整 LOB 模式，並啟用控制資料表。或者，您可以設定 CommitRate 進階設定。	DBA

任務	Description	所需的技能
設定資料表映射。	在資料表映射區段中，為遷移中包含的所有結構描述中的所有資料表建立包含規則，然後建立排除規則。新增三個轉換規則，將結構描述、資料表和資料欄名稱轉換為小寫，並新增此特定遷移所需的任何其他規則。	DBA
啟動 任務。	啟動複寫任務。確定完全載入正在執行中。在主要 Oracle 資料庫上執行 ALTER SYSTEM SWITCH LOGFILE，以啟動任務。	DBA
從 AWS SCT 執行中遷移指令碼。	在 Amazon RDS for PostgreSQL 中，執行這些指令碼：create_index.sql 和 create_constraint.sql。	DBA
重新啟動任務以繼續變更資料擷取 (CDC)。	在 Amazon RDS for PostgreSQL 資料庫執行個體中，執行 VACUUM，然後重新啟動 AWS DMS 任務以套用快取的 CDC 變更。	DBA

切換到 PostgreSQL 資料庫

任務	Description	所需的技能
檢查 AWS DMS 日誌和中繼資料表。	驗證任何錯誤並視需要修正。	DBA
停止所有 Oracle 相依性。	關閉 Oracle 資料庫上的接聽程式，並執行 ALTER SYSTEM	DBA

任務	Description	所需的技能
	SWITCH LOGFILE。在未顯示活動時停止 AWS DMS 任務。	
從 AWS SCT 執行遷移後指令碼。	在 Amazon RDS for PostgreSQL 中，執行這些指令碼：create_foreign_key_constraint.sql 和 create_triggers.sql。	DBA
完成任何其他 Amazon RDS for PostgreSQL 步驟。	視需要遞增序列以符合 Oracle，執行 VACUUM 和 ANALYZE，並拍攝快照以符合合規。	DBA
開啟 Amazon RDS for PostgreSQL 的連線。	從 Amazon RDS for PostgreSQL 移除 AWS DMS 安全群組、新增生產安全群組，並將您的應用程式指向新的資料庫。	DBA
清除 AWS DMS 資源。	移除端點、複寫任務、複寫執行個體和 EC2 執行個體。	SysAdmin、DBA

相關資源

- AWS DMS 文件
- AWS SCT 文件
- Amazon RDS for PostgreSQL 定價
- 使用 Oracle 資料庫做為 AWS DMS 的來源
- 使用 PostgreSQL 資料庫做為 AWS DMS 的目標
- Quest SharePlex 文件

使用具體化視觀表和 AWS DMS，從 Oracle 8i 或 9i 遷移至 Amazon RDS for PostgreSQL

Kumar Babu P G 和 Pragnesh Patel，Amazon Web Services

總結

此模式說明如何將內部部署舊版 Oracle 8i 或 9i 資料庫遷移至 Amazon Relational Database Service (Amazon RDS) for PostgreSQL 或 Amazon Aurora PostgreSQL 相容版本。

AWS Database Migration Service (AWS DMS) 不支援 Oracle 8i 或 9i 作為來源，因此此模式使用與 AWS DMS 相容的中繼 Oracle 資料庫執行個體，例如 Oracle 10g 或 11g。它也會使用具體化視觀表功能，將資料從來源 Oracle 8i/9i 執行個體遷移至中繼 Oracle 10g/11g 執行個體。

AWS Schema Conversion Tool (AWS SCT) 會轉換資料庫結構描述，而 AWS DMS 會將資料遷移至目標 PostgreSQL 資料庫。

此模式可協助想要從舊版 Oracle 資料庫遷移的使用者，將資料庫停機時間降至最低。在此實作中，停機時間會受限於在目標資料庫上建立或驗證所有外部金鑰、觸發條件和序列所需的時間長度。

模式使用已安裝 Oracle 10g/11g 資料庫的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，以協助 AWS DMS 串流資料。您可以暫停從現場部署 Oracle 資料庫到中繼 Oracle 執行個體的串流複寫，讓 AWS DMS 能夠跟上資料驗證的進度，或使用其他資料驗證工具。當 AWS DMS 已完成遷移目前的變更時，PostgreSQL 資料庫執行個體和中繼 Oracle 資料庫將具有相同的資料。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署資料中心中的來源 Oracle 8i 或 9i 資料庫
- 在內部部署資料中心和 AWS 之間設定的 AWS Direct Connect
- 在本機電腦或安裝 AWS SCT 的 EC2 執行個體上安裝的 AWS SCT 連接器的 Java Database Connectivity (JDBC) 驅動程式
- 熟悉使用 Oracle 資料庫做為 AWS DMS 來源
- 熟悉使用 PostgreSQL 資料庫做為 AWS DMS 目標

限制

- 資料庫大小限制為 64 TB

產品版本

- 來源資料庫的 Oracle 8i 或 9i
- 適用於中繼資料庫的 Oracle 10g 或 11g
- PostgreSQL 10.17 或更新版本

Architecture

來源技術堆疊

- Oracle 8i 或 9i 資料庫

目標技術堆疊

- Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 相容

目標架構

工具

- AWS DMS 有助於快速安全地遷移資料庫。來源資料庫在遷移期間保持完全運作，將依賴資料庫的應用程式停機時間降至最低。AWS DMS 可以在最廣泛使用的商業和開放原始碼資料庫之間遷移您的資料。
- AWS SCT 會自動將來源資料庫結構描述和大多數資料庫程式碼物件，包括檢視、預存程序和函數，轉換為與目標資料庫相容的格式。無法自動轉換的物件會清楚標示，以便手動轉換以完成遷移。AWS SCT 也可以掃描應用程式原始碼以取得內嵌 SQL 陳述式，並將其轉換為資料庫結構描述轉換專案的一部分。在此過程中，AWS SCT 會透過將舊版 Oracle 和 SQL Server 函數轉換為其 AWS 對等函數來執行雲端原生程式碼最佳化，協助您在遷移資料庫的同時現代化應用程式。當結構描述轉換完成時，AWS SCT 可以使用內建的資料遷移代理程式，協助將資料從各種資料倉儲遷移到 Amazon Redshift。

最佳實務

如需重新整理具體化視觀表的最佳實務，請參閱下列 Oracle 文件：

- 重新整理具體化視觀表

- 具體化視觀表的快速重新整理

史詩

在 EC2 執行個體上安裝 Oracle 並建立具體化視觀表

任務	Description	所需的技能
設定 EC2 執行個體的網路。	建立虛擬私有雲端 (VPC)、子網路、網際網路閘道、路由表和安全群組。	AWS SysAdmin
建立 EC2 執行個體。	選取 EC2 執行個體的 Amazon Machine Image (AMI)。選擇執行個體大小並設定執行個體詳細資訊：執行個體數量 (1)、上一個步驟的 VPC 和子網路、自動指派公有 IP 和其他選項。新增儲存體、設定安全群組，以及啟動執行個體。出現提示時，請建立並儲存下一個步驟的金鑰對。	AWS SysAdmin
在 EC2 執行個體上安裝 Oracle。	取得授權和必要的 Oracle 二進位檔，並在 EC2 執行個體上安裝 Oracle 10g 或 11g。	DBA
設定 Oracle 聯網。	在 中修改或新增項目`listener.ora` ，以連線至內部部署來源 Oracle 8i/9i 資料庫，然後建立資料庫連結。	DBA
建立具體化視觀表。	識別要在來源 Oracle 8i/9i 資料庫中複寫的資料庫物件，然後使用資料庫連結為所有物件建立具體化檢視。	DBA

任務	Description	所需的技能
部署指令碼，以所需的間隔重新整理具體化視觀表。	開發和部署指令碼，以在 Amazon EC2 Oracle 10g/11g 執行個體上以所需的間隔重新整理具體化視觀表。使用增量重新整理選項來重新整理具體化視觀表。	DBA

將 Oracle 資料庫結構描述轉換為 PostgreSQL

任務	Description	所需的技能
設定 AWS SCT。	建立新的報告，然後連接至 Oracle 做為來源，而 PostgreSQL 做為目標。在專案設定中，開啟 SQL 指令碼索引標籤。將目標 SQL 指令碼變更為多個檔案。(AWS SCT 不支援 Oracle 8i/9i 資料庫，因此您必須在中繼 Oracle 10g/11g 執行個體上還原僅限結構描述的傾印，並將其用作 AWS SCT 的來源。)	DBA
轉換 Oracle 資料庫結構描述。	在動作索引標籤上，選擇產生報告、轉換結構描述，然後儲存為 SQL。	DBA
修改 SQL 指令碼。	根據最佳實務進行修改。例如，切換到適當的資料類型，並為 Oracle 特定函數開發 PostgreSQL 對等項目。	DBA、DevDBA

建立和設定 Amazon RDS 資料庫執行個體以託管轉換後的資料庫

任務	Description	所需的技能
建立 Amazon RDS 資料庫執行個體。	在 Amazon RDS 主控台中，建立新的 PostgreSQL 資料庫執行個體。	AWS SysAdmin、DBA
設定資料庫執行個體。	指定資料庫引擎版本、資料庫執行個體類別、異地同步備份部署、儲存類型和配置的儲存。輸入資料庫執行個體識別符、主要使用者名稱和主要密碼。	AWS SysAdmin、DBA
設定網路和安全性。	指定 VPC、子網路群組、公有可存取性、可用區域偏好設定和安全群組。	DBA、SysAdmin
設定資料庫選項。	指定資料庫名稱、連接埠、參數群組、加密和主金鑰。	DBA、AWS SysAdmin
設定備份。	指定備份保留期、備份時段、開始時間、持續時間，以及是否要將標籤複製到快照。	AWS SysAdmin、DBA
設定監控選項。	啟用或停用增強型監控和效能洞察。	AWS SysAdmin、DBA
設定維護選項。	指定自動次要版本升級、維護時段，以及開始日期、時間和持續時間。	AWS SysAdmin、DBA
從 AWS SCT 執行預遷移指令碼。	在目標 Amazon RDS for PostgreSQL 執行個體上，使用 AWS SCT 中的 SQL 指令碼和其他修改來建立資料庫結構描述。這些可能包括執行多個指令碼，包括使用者建立、資	AWS SysAdmin、DBA

任務	Description	所需的技能
	料庫建立、結構描述建立、資料表、檢視、函數和其他程式碼物件。	

使用 AWS DMS 遷移資料

任務	Description	所需的技能
在 AWS DMS 中建立複寫執行個體。	完成名稱、執行個體類別、VPC （與 EC2 執行個體相同）、異地同步備份和公有可存取性的欄位。在進階組態區段中，指定配置的儲存、子網路群組、可用區域、VPC 安全群組和 AWS Key Management Service (AWS KMS) 金鑰。	AWS SysAdmin、DBA
建立來源資料庫端點。	指定端點名稱、類型、來源引擎 (Oracle)、伺服器名稱 (EC2 執行個體的私有 DNS 名稱）、連接埠、SSL 模式、使用者名稱、密碼、SID、VPC （指定具有複寫執行個體的 VPC) 和複寫執行個體。若要測試連線，請選擇執行測試，然後建立端點。您也可以設定下列進階設定：maxFileSize 和 numberDataTypeScale。	AWS SysAdmin、DBA
將 AWS DMS 連線至 Amazon RDS for PostgreSQL。	如果您的 PostgreSQL 資料庫位於另一個 VPCs，請建立跨 VPC 連線的遷移安全群組。	AWS SysAdmin、DBA

任務	Description	所需的技能
建立目標資料庫端點。	指定端點名稱、類型、來源引擎 (PostgreSQL)、伺服器名稱 (Amazon RDS 端點）、連接埠、SSL 模式、使用者名稱、密碼、資料庫名稱、VPC（指定具有複寫執行個體的 VPC) 和複寫執行個體。若要測試連線，請選擇執行測試，然後建立端點。您也可以設定下列進階設定：maxFileSize 和 numberDataTypeScale。	AWS SysAdmin、DBA
建立 AWS DMS 複寫任務。	指定任務名稱、複寫執行個體、來源和目標端點，以及複寫執行個體。針對遷移類型，選擇遷移現有資料並複寫持續變更。清除建立時啟動任務核取方塊。	AWS SysAdmin、DBA
設定 AWS DMS 複寫任務設定。	針對目標資料表準備模式，選擇不執行任何動作。完全載入完成後停止任務（以建立主索引鍵）。指定有限或完整 LOB 模式，並啟用控制資料表。或者，您可以設定 CommitRate 進階設定。	DBA
設定資料表映射。	在資料表映射區段中，為遷移中包含的所有結構描述中的所有資料表建立包含規則，然後建立排除規則。新增三個轉換規則，將結構描述、資料表和資料欄名稱轉換為小寫，並新增此特定遷移所需的任何其他規則。	DBA

任務	Description	所需的技能
啟動 任務。	啟動複寫任務。確定完全載入正在執行中。在主要 Oracle 資料庫`ALTER SYSTEM SWITCH LOGFILE`上執行 以啟動任務。	DBA
從 AWS SCT 執行中遷移指令碼。	在 Amazon RDS for PostgreSQL 中，執行下列指令碼： `create_index.sql` 和 `create_constraint.sql` （如果最初未建立完整的結構描述）。	DBA
繼續任務以繼續變更資料擷取 (CDC)。	在 Amazon RDS for PostgreSQL 資料庫執行個體`VACUUM`上執行，然後重新啟動 AWS DMS 任務以套用快取的 CDC 變更。	DBA

切換到 PostgreSQL 資料庫

任務	Description	所需的技能
檢查 AWS DMS 日誌和驗證資料表。	檢查並修正任何複寫或驗證錯誤。	DBA
停止使用現場部署 Oracle 資料庫及其相依性。	停止所有 Oracle 相依性、關閉 Oracle 資料庫上的接聽程式，然後執行 `ALTER SYSTEM SWITCH LOGFILE`。在未顯示活動時停止 AWS DMS 任務。	DBA
從 AWS SCT 執行遷移後指令碼。	在 Amazon RDS for PostgreSQL 中，執行這些指令碼：`create_foreign_key_constrai`	DBA

任務	Description	所需的技能
	`nt.sql and create_tr iggers.sql` 。請確定序列是最新的。	
完成其他 Amazon RDS for PostgreSQL 步驟。	視需要遞增序列以符合 Oracle、執行 `VACUUM`和 `ANALYZE`，並拍攝快照以符合規範。	DBA
開啟 Amazon RDS for PostgreSQL 的連線。	從 Amazon RDS for PostgreSQL 移除 AWS DMS 安全群組、新增生產安全群組，並將您的應用程式指向新的資料庫。	DBA
清除 AWS DMS 物件。	移除端點、複寫任務、複寫執行個體和 EC2 執行個體。	SysAdmin、DBA

相關資源

- AWS DMS 文件
- AWS SCT 文件
- Amazon RDS for PostgreSQL 定價
- 使用 Oracle 資料庫做為 AWS DMS 的來源
- 使用 PostgreSQL 資料庫做為 AWS DMS 的目標

使用 AWS DMS 和 AWS SCT 從 Oracle on Amazon EC2 遷移至 Amazon RDS for MySQL

Anil Kunapareddy，Amazon Web Services

Harshad Gohil，無

摘要

在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上管理 Oracle 資料庫需要 資源，而且成本高昂。將這些資料庫移至適用於 MySQL 資料庫執行個體的 Amazon Relational Database Service (Amazon RDS)，可透過最佳化整體 IT 預算來簡化您的任務。Amazon RDS for MySQL 也提供異地同步備份、可擴展性和自動備份等功能。

此模式會逐步引導您在 Amazon EC2 上將來源 Oracle 資料庫遷移至目標 Amazon RDS for MySQL 資料庫執行個體。它使用 AWS Database Migration Service (AWS DMS) 遷移資料，並使用 AWS Schema Conversion Tool (AWS SCT) 將來源資料庫結構描述和物件轉換為與 Amazon RDS for MySQL 相容的格式。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在 ARCHIVELOG 模式下執行執行個體和接聽程式服務的來源資料庫
- 目標 Amazon RDS for MySQL 資料庫，具有足夠的資料遷移儲存空間

限制

- AWS DMS 不會在目標資料庫上建立結構描述；您必須這麼做。目標的結構描述名稱必須已存在。來源結構描述中的資料表會匯入至使用者/結構描述，AWS DMS 會使用此結構描述來連線至目標執行個體。如果您必須遷移多個結構描述，即必須建立多項複寫任務。

產品版本

- 版本 10.2 及更新版本、11g 及最高 12.2 和 18c 的所有 Oracle 資料庫版本。如需支援版本的最新清單，請參閱使用 Oracle 資料庫做為 AWS DMS 的來源和使用 MySQL 相容資料庫做為 AWS DMS 的目標。我們建議您使用最新版本的 AWS DMS，以獲得最全面的版本和功能支援。如需 AWS SCT 支援的 Oracle 資料庫版本的相關資訊，請參閱 AWS SCT 文件。

- AWS DMS 支援 MySQL 5.5、5.6 和 5.7 版。

Architecture

來源技術堆疊

- EC2 執行個體上的 Oracle 資料庫

目標技術堆疊

- Amazon RDS for MySQL 資料庫執行個體

資料遷移架構

來源和目標架構

工具

- AWS DMS - AWS Database Migration Service (AWS DMS) 是一種 Web 服務，可用來將資料從內部部署、Amazon RDS 資料庫執行個體或 EC2 執行個體上的資料庫遷移到 AWS 服務上的資料庫，例如 Amazon RDS for MySQL 或 EC2 執行個體。您也可以將資料庫從 AWS 服務遷移至內部部署資料庫。您可以在異質或同質資料庫引擎之間遷移資料。
- AWS SCT - AWS Schema Conversion Tool (AWS SCT) 會自動將來源資料庫結構描述和大部分資料庫程式碼物件，包括檢視、預存程序和函數，轉換為與目標資料庫相容的格式，使異質資料庫遷移可預測。使用 AWS SCT 轉換資料庫結構描述和程式碼物件之後，您可以使用 AWS DMS 將資料從來源資料庫遷移至目標資料庫，以完成遷移專案。

史詩

規劃遷移

任務	Description	所需的技能
識別來源和目標資料庫版本和引擎。		DBA/開發人員

任務	Description	所需的技能
識別 DMS 複寫執行個體。		DBA/開發人員
識別儲存需求，例如儲存類型和容量。		DBA/開發人員
識別網路需求，例如延遲和頻寬。		DBA/開發人員
識別來源和目標伺服器執行個體的硬體需求（根據 Oracle 相容性清單和容量需求）。		DBA/開發人員
識別來源和目標資料庫的網路存取安全需求。		DBA/開發人員
安裝 AWS SCT 和 Oracle 驅動程式。		DBA/開發人員
決定備份策略。		DBA/開發人員
判斷可用性需求。		DBA/開發人員
識別應用程式遷移和切換策略。		DBA/開發人員
根據容量、儲存體和網路功能，選取適當的資料庫執行個體類型。		DBA/開發人員

設定環境

任務	Description	所需的技能
建立 Virtual Private Cloud (VPC) 來源、目標和複寫執行個體應該位於相同的 VPC 中。		開發人員

任務	Description	所需的技能
最好在相同的可用區域中擁有這些項目。		
建立資料庫存取所需的安全群組。		開發人員
產生和設定金鑰對。		開發人員
設定子網路、可用區域和 CIDR 區塊。		開發人員

設定來源：EC2 執行個體上的 Oracle 資料庫

任務	Description	所需的技能
使用必要的使用者和角色在 Amazon EC2 上安裝 Oracle Database。		DBA
執行下一欄中的三個步驟，從 EC2 執行個體外部存取 Oracle。	1. 將 中的本機主機`tnsnames`變更為 Amazon EC2 公有 DNS。 2. 將 中的本機主機`listener`變更為 Amazon EC2 公有 DNS。 3. 停止並重新啟動接聽程式。	DBA
重新啟動 Amazon EC2 時，公有 DNS 會變更。請務必更新 'tnsnames' 和 'listener' 中的 Amazon EC2 公有 DNS，或使用彈性 IP 地址。		DBA/開發人員

任務	Description	所需的技能
設定 EC2 執行個體安全群組，讓複寫執行個體和必要的用戶端可以存取來源資料庫。		DBA/開發人員

設定目標：Amazon RDS for MySQL

任務	Description	所需的技能
設定和啟動 Amazon RDS for MySQL 資料庫執行個體。		開發人員
在 Amazon RDS for MySQL 資料庫執行個體中建立必要的資料表空間。		DBA
設定安全群組，讓複寫執行個體和必要的用戶端可以存取目標資料庫。		開發人員

設定 AWS SCT 並在目標資料庫中建立結構描述

任務	Description	所需的技能
安裝 AWS SCT 和 Oracle 驅動程式。		開發人員
輸入適當的參數並連接到來源和目標。		開發人員
產生結構描述轉換報告。		開發人員
視需要更正程式碼和結構描述，特別是資料表空間和引號，並在目標資料庫上執行。		開發人員

任務	Description	所需的技能
在遷移資料之前，驗證來源與目標上的結構描述。		開發人員

使用 AWS DMS 遷移資料

任務	Description	所需的技能
對於完全載入和變更資料擷取 (CDC) 或僅 CDC，您必須設定額外的連線屬性。		開發人員
AWS DMS 來源 Oracle 資料庫定義中指定的使用者必須獲得所有必要的權限。如需完整清單，請參閱 https：//https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Source.Oracle.html#CHAP_Source.Oracle.Self-Managed。		DBA/開發人員
在來源資料庫中啟用補充記錄。		DBA/開發人員
對於完全載入和變更資料擷取 (CDC) 或僅 CDC，請在來源資料庫中啟用 ARCHIVELOG 模式。		DBA
建立來源和目標端點，並測試連線。		開發人員
成功連接端點時，請建立複寫任務。		開發人員
在任務中選取僅限 CDC （或） 完全載入加上 CDC，		開發人員

任務	Description	所需的技能
以分別擷取僅限連續複寫（或）完全載入加上持續變更的變更。		
執行複寫任務並監控 Amazon CloudWatch logs。		開發人員
驗證來源和目標資料庫中的資料。		開發人員

遷移您的應用程式並切換

任務	Description	所需的技能
請遵循應用程式遷移策略的步驟。		DBA、開發人員、應用程式擁有者
請遵循應用程式切換/切換策略的步驟。		DBA、開發人員、應用程式擁有者

關閉專案

任務	Description	所需的技能
驗證來源資料庫與目標資料庫中的結構描述和資料。		DBA/開發人員
收集遷移時間的指標、手動與工具的百分比、節省成本等。		DBA/Developer/AppOwner
檢閱專案文件和成品。		DBA/Developer/AppOwner
關閉臨時 AWS 資源。		DBA/開發人員
關閉專案並提供意見回饋。		DBA/Developer/AppOwner

相關資源

- AWS DMS 文件
- AWS DMS 網站
- AWS DMS 部落格文章
- 將 Oracle Database 遷移到 AWS 的策略
- Amazon RDS for Oracle FAQs
- Oracle 常見問答集
- Amazon EC2
- Amazon EC2 FAQs
- 在雲端運算環境中授權 Oracle 軟體

使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫從 Amazon EC2 遷移至 Amazon RDS for MariaDB

Veeranjaneyulu Grandhi 和 vinod kumar，Amazon Web Services

總結

此模式會逐步引導您將 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的 Oracle 資料庫遷移至適用於 MariaDB 資料庫執行個體的 Amazon Relational Database Service (Amazon RDS)。模式使用 AWS Data Migration Service (AWS DMS) 進行資料遷移，並使用 AWS Schema Conversion Tool (AWS SCT) 進行結構描述轉換。

在 EC2 執行個體上管理 Oracle 資料庫需要更多資源，而且比在 Amazon RDS 上使用資料庫更昂貴。Amazon RDS 可讓您輕鬆地在雲端中設定、操作和擴展關聯式資料庫。Amazon RDS 提供經濟實惠且可調整大小的容量，同時自動化耗時的管理任務，例如硬體佈建、資料庫設定、修補和備份。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 啟動並執行執行個體和接聽程式服務的來源 Oracle 資料庫。此資料庫應處於 ARCHIVELOG 模式。
- 熟悉使用 Oracle 資料庫做為 AWS DMS 來源。
- 熟悉使用 Oracle 做為 AWS SCT 來源。

限制

- 資料庫大小限制：64 TB

產品版本

- 版本 10.2 及更新版本、11g 及最高 12.2 和 18c 的所有 Oracle 資料庫版本。如需支援版本的最新清單，請參閱 AWS 文件中的使用 Oracle 資料庫做為 AWS DMS 的來源和 AWS SCT 版本資料表。
- Amazon RDS 支援 MariaDB Server Community Server 10.3、10.4、10.5 和 10.6 版。如需支援版本的最新清單，請參閱Amazon RDS 文件。

Architecture

來源技術堆疊

- EC2 執行個體上的 Oracle 資料庫

目標技術堆疊

- Amazon RDS for MariaDB

資料遷移架構

目標架構

工具

- AWS Schema Conversion Tool (AWS SCT) 會自動將來源資料庫結構描述和大部分資料庫程式碼物件，包括檢視、預存程序和函數，轉換為與目標資料庫相容的格式，讓異質資料庫遷移可預測。使用 AWS SCT 轉換資料庫結構描述和程式碼物件之後，您可以使用 AWS DMS 將資料從來源資料庫遷移到目標資料庫，以完成遷移專案。如需詳細資訊，請參閱 AWS SCT 文件中的使用 Oracle 做為 AWS SCT 的來源。
- AWS Database Migration Service (AWS DMS) 可協助您快速安全地將資料庫遷移至 AWS。來源資料庫在遷移期間保持完全運作，將依賴資料庫的應用程式停機時間降至最低。AWS DMS 可以在最廣泛使用的商業和開放原始碼資料庫之間遷移您的資料。AWS DMS 支援同質遷移，例如 Oracle 到 Oracle，以及不同資料庫平台之間的異質遷移，例如 Oracle 或 Microsoft SQL Server 到 Amazon Aurora。若要進一步了解遷移 Oracle 資料庫，請參閱 AWS DMS 文件中的使用 Oracle 資料庫做為 AWS DMS 的來源。

史詩

規劃遷移

任務	Description	所需的技能
識別版本和資料庫引擎。	識別來源和目標資料庫版本和引擎。	DBA、開發人員
識別複寫執行個體。	識別 AWS DMS 複寫執行個體。	DBA、開發人員

任務	Description	所需的技能
識別儲存需求。	識別儲存類型和容量。	DBA、開發人員
識別網路需求。	識別網路延遲和頻寬。	DBA、開發人員
識別硬體需求。	識別來源和目標伺服器執行個體的硬體需求（根據 Oracle 相容性清單和容量需求）。	DBA、開發人員
識別安全需求。	識別來源和目標資料庫的網路存取安全需求。	DBA、開發人員
安裝驅動程式。	安裝最新的 AWS SCT 和 Oracle 驅動程式。	DBA、開發人員
決定備份策略。		DBA、開發人員
判斷可用性需求。		DBA、開發人員
選擇應用程式遷移/切換策略。		DBA、開發人員
選取 執行個體類型。	根據容量、儲存體和網路功能選取適當的執行個體類型。	DBA、開發人員

設定環境

任務	Description	所需的技能
建立 Virtual Private Cloud (VPC)	來源、目標和複寫執行個體應位於相同的 VPC 和相同的可用區域（建議）。	開發人員
建立安全群組。	建立資料庫存取所需的安全群組。	開發人員
產生金鑰對。	產生和設定金鑰對。	開發人員

任務	Description	所需的技能
設定其他資源。	設定子網路、可用區域和 CIDR 區塊。	開發人員

設定來源

任務	Description	所需的技能
啟動 EC2 執行個體。	如需說明，請參閱 Amazon EC2 文件。	開發人員
安裝 Oracle 資料庫。	在 EC2 執行個體上安裝 Oracle 資料庫，其中包含必要的使用者和角色。	DBA
請依照任務描述中的步驟，從 EC2 執行個體外部存取 Oracle。	1. 將 中的本機主機`tnsnames`變更為 Amazon EC2 公有 DNS。 2. 將 中的本機主機`listener`變更為 Amazon EC2 公有 DNS。 3. 停止並重新啟動接聽程式。	DBA
更新 Amazon EC2 公有 DNS。	EC2 執行個體重新啟動後，公有 DNS 會變更。請務必更新 `tnsnames`和 中的 Amazon EC2 公有 DNS`listener`，或使用彈性 IP 地址。	DBA、開發人員
設定 EC2 執行個體安全群組。	設定 EC2 執行個體安全群組，讓複寫執行個體和必要的用戶端可以存取來源資料庫。	DBA、開發人員

設定目標 Amazon RDS for MariaDB 環境

任務	Description	所需的技能
啟動 RDS 資料庫執行個體。	設定和啟動 Amazon RDS for MariaDB 資料庫執行個體。	開發人員
建立資料表空間。	在 Amazon RDS MariaDB 資料庫中建立任何必要的資料表空間。	DBA
設定安全群組。	設定安全群組，讓複寫執行個體和必要的用戶端可以存取目標資料庫。	開發人員

設定 AWS SCT

任務	Description	所需的技能
安裝驅動程式。	安裝最新的 AWS SCT 和 Oracle 驅動程式。	開發人員
連接。	輸入適當的參數，然後連接到來源和目標。	開發人員
產生結構描述轉換報告。	產生 AWS SCT 結構描述轉換報告。	開發人員
視需要更正程式碼和結構描述。	對程式碼和結構描述進行任何必要的更正（特別是資料表空間和引號）。	DBA、開發人員
驗證結構描述。	在載入資料之前，驗證來源與目標上的結構描述。	開發人員

使用 AWS DMS 遷移資料

任務	Description	所需的技能
設定連線屬性。	對於完全載入和變更資料擷取 (CDC)，或僅針對 CDC，請設定額外的連線屬性。如需詳細資訊，請參閱 Amazon RDS 文件。	開發人員
啟用補充記錄。	在來源資料庫上啟用補充記錄。	DBA、開發人員
啟用封存日誌模式。	對於完全載入和 CDC （或僅適用於 CDC)，請在來源資料庫上啟用封存日誌模式。	DBA
建立和測試端點。	建立來源和目標端點並測試連線。如需詳細資訊，請參閱 Amazon DMS 文件。	開發人員
建立複寫任務。	成功連接端點時，請建立複寫任務。如需詳細資訊，請參閱 Amazon DMS 文件。	開發人員
選擇複寫類型。	在任務中選擇僅限 CDC 或完全載入加上 CDC，以擷取僅限連續複寫的變更，或分別針對完全載入和持續變更進行的變更。	開發人員
啟動並監控任務。	啟動複寫任務並監控 Amazon CloudWatch logs。如需詳細資訊，請參閱 Amazon DMS 文件。	開發人員
驗證資料。	驗證來源和目標資料庫中的資料。	開發人員

遷移應用程式並切換到目標資料庫

任務	Description	所需的技能
遵循所選的應用程式遷移策略。		DBA、應用程式擁有者、開發人員
遵循所選的應用程式切換/切換策略。		DBA、應用程式擁有者、開發人員

關閉專案

任務	Description	所需的技能
驗證結構描述和資料。	確保在專案關閉之前，在來源與目標中成功驗證結構描述和資料。	DBA、開發人員
收集指標。	收集遷移時間、手動與工具任務的百分比、節省成本和類似條件的指標。	DBA、應用程式擁有者、開發人員
檢閱文件。	檢閱專案文件和成品。	DBA、應用程式擁有者、開發人員
關閉資源。	關閉臨時 AWS 資源。	DBA、開發人員
關閉專案。	關閉遷移專案並提供任何意見回饋。	DBA、應用程式擁有者、開發人員

相關資源

- MariaDB Amazon RDS 概觀
- Amazon RDS for MariaDB 產品詳細資訊
- 使用 Oracle 資料庫做為 AWS DMS 的來源
- 將 Oracle 資料庫遷移至 AWS 的策略

- 在雲端運算環境中授權 Oracle 軟體
- Amazon RDS for Oracle FAQs
- AWS DMS 概觀
- AWS DMS 部落格文章
- Amazon EC2 概觀
- Amazon EC2 FAQs
- AWS SCT 文件

使用 AWS DMS 和 AWS SCT 將內部部署 Oracle 資料庫遷移至 Amazon RDS for MySQL

Sergey Dmitriev 和 Naresh Damera，Amazon Web Services

總結

此模式會逐步引導您將現場部署 Oracle 資料庫遷移至 MySQL 資料庫執行個體的 Amazon Relational Database Service (Amazon RDS)。它使用 AWS Database Migration Service (AWS DMS) 遷移資料，並使用 AWS Schema Conversion Tool (AWS SCT) 將來源資料庫結構描述和物件轉換為與 Amazon RDS for MySQL 相容的格式。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署資料中心中的來源 Oracle 資料庫

限制

- 資料庫大小限制：64 TB

產品版本

- 版本 11g （版本 11.2.0.3.v1 和更新版本） 和最高 12.2 和 18c 的所有 Oracle 資料庫版本。如需支援版本的最新清單，請參閱使用 Oracle 資料庫做為 AWS DMS 的來源。我們建議您使用最新版本的 AWS DMS，以獲得最全面的版本和功能支援。如需 AWS SCT 支援的 Oracle 資料庫版本的相關資訊，請參閱 AWS SCT 文件。
- AWS DMS 目前支援 MySQL 5.5、5.6 和 5.7 版。如需支援版本的最新清單，請參閱 AWS 文件中的使用 MySQL 相容資料庫做為 AWS DMS 的目標。

Architecture

來源技術堆疊

- 內部部署 Oracle 資料庫

目標技術堆疊

- Amazon RDS for MySQL 資料庫執行個體

資料遷移架構

工具

- AWS DMS - AWS Database Migration Services (AWS DMS) 可協助您遷移關聯式資料庫、資料倉儲、NoSQL 資料庫和其他類型的資料存放區。您可以使用 AWS DMS 將資料遷移至 AWS 雲端，可在現場部署執行個體 (透過 AWS 雲端設定) 或在雲端和現場部署設定之間進行。
- AWS SCT - AWS Schema Conversion Tool (AWS SCT) 用於將您的資料庫結構描述從一個資料庫引擎轉換為另一個資料庫引擎。工具轉換的自訂程式碼包含檢視、預存程序和函數。工具無法自動轉換的任何程式碼都會清楚標示，讓您可以自行轉換。

史詩

規劃遷移

任務	Description	所需的技能
驗證來源和目標資料庫版本和引擎。		DBA
識別目標伺服器執行個體的硬體需求。		DBA、SysAdmin
識別儲存需求（儲存類型和容量）。		DBA、SysAdmin
根據容量、儲存功能和網路功能選擇適當的執行個體類型。		DBA、SysAdmin
識別來源和目標資料庫的網路存取安全需求。		DBA、SysAdmin
識別應用程式遷移策略。		DBA、SysAdmin、應用程式擁有者

設定基礎設施

任務	Description	所需的技能
建立虛擬私有雲端 (VPC) 和子網路。		SysAdmin
建立安全群組和網路存取控制清單 ACLs)。		SysAdmin
設定和啟動 Amazon RDS 資料庫執行個體。		DBA、SysAdmin

遷移資料

任務	Description	所需的技能
使用 AWS SCT 遷移資料庫結構描述。		DBA
使用 AWS DMS 遷移資料。		DBA

遷移應用程式

任務	Description	所需的技能
使用 AWS SCT 來分析和轉換應用程式程式碼內的 SQL 程式碼。	如需詳細資訊，請參閱 https：//https://docs.aws.amazon.com/SchemaConversionTool/latest/userguide/CHAP_Converting.App.html。	應用程式擁有者
遵循應用程式遷移策略。		DBA、SysAdmin、應用程式擁有者

切換

任務	Description	所需的技能
將應用程式用戶端切換到新的基礎設施。		DBA、SysAdmin、應用程式擁有者

關閉專案

任務	Description	所需的技能
關閉臨時 AWS 資源。		DBA、SysAdmin
檢閱並驗證專案文件。		DBA、SysAdmin
收集遷移時間的指標、手動與工具的 %、節省成本等。		DBA、SysAdmin
關閉專案並提供意見回饋。		

相關資源

參考

- AWS DMS 文件
- AWS SCT 文件
- Amazon RDS 定價

教學課程和影片

- AWS DMS 入門
- Amazon RDS 入門
- AWS DMS （影片）
- Amazon RDS （影片）

使用 Oracle 旁觀者和 AWS DMS 將內部部署 Oracle 資料庫遷移至 Amazon RDS for PostgreSQL

Cady Motyka，Amazon Web Services

總結

此模式說明如何將現場部署 Oracle 資料庫遷移至下列任一 PostgreSQL 相容 AWS 資料庫服務，並將停機時間降至最低：

- PostgreSQL 的 Amazon Relational Database Service (Amazon RDS)
- Amazon Aurora PostgreSQL-Compatible Edition

解決方案使用 AWS Database Migration Service (AWS DMS) 來遷移資料、AWS Schema Conversion Tool (AWS SCT) 來轉換資料庫結構描述，以及 Oracle bystander 資料庫來協助管理遷移。在此實作中，停機時間僅限於建立或驗證資料庫上所有外部金鑰所需的時間。

解決方案也使用 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體搭配 Oracle 旁觀者資料庫，以協助透過 AWS DMS 控制資料串流。您可以暫停從現場部署 Oracle 資料庫到 Oracle 旁觀者的串流複寫，以啟用 AWS DMS 來跟上資料驗證的進度，或使用其他資料驗證工具。當 AWS DMS 完成遷移目前的變更時，Amazon RDS for PostgreSQL 資料庫執行個體或 Aurora PostgreSQL 相容資料庫執行個體和旁觀者資料庫將具有相同的資料。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 已設定 Active Data Guard 待命資料庫的現場部署資料中心中的來源 Oracle 資料庫
- 在內部部署資料中心和 AWS Secrets Manager 之間設定 AWS Direct Connect，以存放資料庫秘密
- AWS SCT 連接器的 Java Database Connectivity (JDBC) 驅動程式，安裝在本機電腦或安裝 AWS SCT 的 EC2 執行個體上
- 熟悉使用 Oracle 資料庫做為 AWS DMS 來源
- 熟悉使用 PostgreSQL 資料庫做為 AWS DMS 的目標

限制

- 資料庫大小限制：64 TB

產品版本

- AWS DMS 支援 10.2 版和更新版本（適用於 10.x 版）、11g 和最高 12.2、18c 和 19c 版的所有 Oracle 資料庫版本。如需支援版本的最新清單，請參閱使用 Oracle 資料庫做為 AWS DMS 的來源。我們建議您使用最新版本的 AWS DMS，以獲得最全面的版本和功能支援。如需 AWS SCT 支援的 Oracle 資料庫版本的相關資訊，請參閱 AWS SCT 文件。
- AWS DMS 支援 PostgreSQL 9.4 版和更新版本（適用於 9.x 版）、10.x、11.x、12.x 和 13.x 版。如需最新資訊，請參閱 AWS 文件中的使用 PostgreSQL 資料庫做為 AWS DMS 的目標。

Architecture

來源技術堆疊

- 內部部署 Oracle 資料庫
- 保留 Oracle 資料庫旁觀者的 AnEC2 執行個體

目標技術堆疊

- Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 執行個體、PostgreSQL 9.3 及更新版本

目標架構

下圖顯示使用 AWS DMS 和 Oracle 旁觀者，將 Oracle 資料庫遷移至 PostgreSQL 相容 AWS 資料庫的範例工作流程：

工具

- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。
- AWS Schema Conversion Tool (AWS SCT) 會自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，以支援異質資料庫遷移。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。

史詩

將 Oracle 資料庫結構描述轉換為 PostgreSQL

任務	Description	所需的技能
設定 AWS SCT。	建立新的報告，並連接至 Oracle 做為來源，而 PostgreSQL 做為目標。在專案設定中，前往 SQL 指令碼索引標籤。將目標 SQL 指令碼變更為多個檔案。這些檔案將在稍後使用，並命名如下： • create_database.sql • create_sequence.sql • create_table.sql • create_view.sql • create_function.sql	DBA
轉換 Oracle 資料庫結構描述。	在動作索引標籤中，選擇產生報告。然後，選擇轉換結構描述，然後選擇另存為 SQL。	DBA
修改指令碼。	例如，如果來源結構描述中的數字已在 PostgreSQL 中轉換為數值格式，但您想要改用 BIGINT 以獲得更好的效能，您可能想要修改指令碼。	DBA

建立和設定 Amazon RDS 資料庫執行個體

任務	Description	所需的技能
建立 Amazon RDS 資料庫執行個體。	在正確的 AWS 區域中，建立新的 PostgreSQL 資料庫執行個體。如需詳細資訊，請參閱	AWS SysAdmin、DBA

任務	Description	所需的技能
	Amazon RDS 文件中的建立 PostgreSQL 資料庫執行個體和連線至 PostgreSQL 資料庫執行個體上的資料庫。	
設定資料庫執行個體規格。	指定資料庫引擎版本、資料庫執行個體類別、異地同步備份部署、儲存類型和配置的儲存。輸入資料庫執行個體識別符、主要使用者名稱和主要密碼。	AWS SysAdmin、DBA
設定網路和安全性。	指定虛擬私有雲端 (VPC)、子網路群組、公有可存取性、可用區域偏好設定和安全群組。	DBA、SysAdmin
設定資料庫選項。	指定資料庫名稱、連接埠、參數群組、加密和 KMS 金鑰。	AWS SysAdmin、DBA
設定備份。	指定備份保留期、備份時段、開始時間、持續時間，以及是否要將標籤複製到快照。	AWS SysAdmin、DBA
設定監控選項。	啟用或停用增強型監控和效能洞察。	AWS SysAdmin、DBA
設定維護選項。	指定自動次要版本升級、維護時段，以及開始日期、時間和持續時間。	AWS SysAdmin、DBA

任務	Description	所需的技能
從 AWS SCT 執行預遷移指令碼。	在 Amazon RDS 執行個體上，執行 AWS SCT 產生的下列指令碼： • create_database.sql • create_sequence.sql • create_table.sql • create_view.sql • create_function.sql	AWS SysAdmin、DBA

在 Amazon EC2 中設定 Oracle 旁觀者

任務	Description	所需的技能
設定 Amazon EC2 的網路。	建立新的 VPC、子網路、網際網路閘道、路由表和安全群組。	AWS SysAdmin
建立 EC2 執行個體。	在適當的 AWS 區域中，建立新的 EC2 執行個體。選取 Amazon Machine Image (AMI)，選擇執行個體大小，並設定執行個體詳細資訊：執行個體數量 (1)、您在先前任務中建立的 VPC 和子網路、自動指派公有 IP 和其他選項。新增儲存、設定安全群組和啟動。出現提示時，請建立並儲存下一個步驟的金鑰對。	AWS SysAdmin
將 Oracle 來源資料庫連接至 EC2 執行個體。	將 IPv4 公有 IP 地址和 DNS 複製到文字檔案，並使用 SSH 連線，如下所示：ssh -i "your_file.pem" ec2-user@	AWS SysAdmin

任務	Description	所需的技能
	<your-IP-address-or-public-DNS>。	
在 Amazon EC2 中為旁觀者設定初始主機。	設定 SSH 金鑰、Bash 設定檔、ORATAB 和符號連結。建立 Oracle 目錄。	AWS SysAdmin、Linux Admin
在 Amazon EC2 中設定旁觀者的資料庫副本	使用 RMAN 建立資料庫複本、啟用補充記錄，以及建立待命控制檔案。複製完成後，將資料庫置於復原模式。	AWS SysAdmin、DBA
設定 Oracle Data Guard。	修改您的 listener.ora 檔案並啟動接聽程式。設定新的封存目的地。將旁觀者置於復原模式、取代暫存檔案以避免未來損毀、視需要安裝 crontab 以防止封存目錄空間不足，以及編輯來源和待命的 manage-trclog-files-oracle.cfg 檔案。	AWS SysAdmin、DBA
準備 Oracle 資料庫以同步運送。	新增待命日誌檔案並變更復原模式。變更來源主要和來源待命上傳送至 SYNC AFFIRM 的日誌。在主要 上切換日誌，透過 Amazon EC2 旁觀者警示日誌確認您正在使用待命日誌檔案，並確認重做串流正在 SYNC 中流動。	AWS SysAdmin、DBA

使用 AWS DMS 遷移資料

任務	Description	所需的技能
在 AWS DMS 中建立複寫執行個體。	完成名稱、執行個體類別、VPC（與 Amazon EC2 執行個體相同）、異地同步備份和公有可存取性的欄位。在進階下，指定配置的儲存、子網路群組、可用區域、VPC 安全群組和 AWS Key Management Service (AWS KMS) 金鑰。	AWS SysAdmin、DBA
建立來源資料庫端點。	指定端點名稱、類型、來源引擎 (Oracle)、伺服器名稱 (Amazon EC2 私有 DNS 名稱）、連接埠、SSL 模式、使用者名稱、密碼、SID、VPC（指定具有複寫執行個體的 VPC) 和複寫執行個體。若要測試連線，請選擇執行測試，然後建立端點。您也可以設定下列進階設定：maxFileSize 和 numberDataTypeScale。	AWS SysAdmin、DBA
將 AWS DMS 連線至 Amazon RDS for PostgreSQL。	建立跨 VPCs 連線的遷移安全群組。	AWS SysAdmin、DBA
建立目標資料庫端點。	指定端點名稱、類型、來源引擎 (PostgreSQL)、伺服器名稱 (Amazon RDS 端點）、連接埠、SSL 模式、使用者名稱、密碼、資料庫名稱、VPC（指定具有複寫執行個體的 VPC) 和複寫執行個體。若要測試連線，請選擇執行測試，然後建立端點。您也可以設定下	AWS SysAdmin、DBA

任務	Description	所需的技能
	列進階設定：maxFileSize 和 numberDataTypeScale。	
建立 AWS DMS 複寫任務。	指定任務名稱、複寫執行個體、來源和目標端點，以及複寫執行個體。針對遷移類型，選擇遷移現有資料並複寫持續變更。清除建立時啟動任務核取方塊。	AWS SysAdmin、DBA
設定 AWS DMS 複寫任務設定。	針對目標資料表準備模式，選擇不執行任何動作。完全載入完成後停止任務（建立主索引鍵）。指定有限或完整 LOB 模式，並啟用控制資料表。或者，您可以設定 CommitRate 進階設定。	DBA
設定資料表映射。	在資料表映射區段中，為遷移中包含的所有結構描述中的所有資料表建立包含規則，然後建立排除規則。新增三個轉換規則，將結構描述、資料表和資料欄名稱轉換為小寫，並新增此特定遷移所需的任何其他規則。	DBA
啟動 任務。	啟動複寫任務。確定完全載入正在執行中。在主要 Oracle 資料庫上執行 ALTER SYSTEM SWITCH LOGFILE，以啟動任務。	DBA

任務	Description	所需的技能
從 AWS SCT 執行中遷移指令碼。	在 Amazon RDS for PostgreSQL 中，執行 AWS SCT 產生的下列指令碼： • create_index.sql • create_constraint.sql	DBA
重新啟動任務以繼續變更資料擷取 (CDC)。	在 Amazon RDS for PostgreSQL 資料庫執行個體上執行 VACUUM，然後重新啟動 AWS DMS 任務以套用快取的 CDC 變更。	DBA

切換到 PostgreSQL 資料庫

任務	Description	所需的技能
檢閱 AWS DMS 日誌和驗證資料表是否有任何錯誤。	檢查並修正任何複寫或驗證錯誤。	DBA
停止所有 Oracle 相依性。	停止所有 Oracle 相依性、關閉 Oracle 資料庫上的接聽程式，並執行 ALTER SYSTEM SWITCH LOGFILE。在未顯示活動時停止 AWS DMS 任務。	DBA
從 AWS SCT 執行遷移後指令碼。	在 Amazon RDS for PostgreSQL 中，執行 AWS SCT 產生的下列指令碼： • create_foreign_key_constraint.sql • create_triggers.sql	DBA

任務	Description	所需的技能
完成其他 Amazon RDS for PostgreSQL 步驟。	視需要遞增序列以符合 Oracle，執行 VACUUM 和 ANALYZE，並拍攝快照以符合合規。	DBA
開啟 Amazon RDS for PostgreSQL 的連線。	從 Amazon RDS for PostgreSQL 移除 AWS DMS 安全群組、新增生產安全群組，並將您的應用程式指向新的資料庫。	DBA
清除 AWS DMS 物件。	移除端點、複寫任務、複寫執行個體和 EC2 執行個體。	SysAdmin、DBA

相關資源

- AWS DMS 文件
- AWS SCT 文件
- Amazon RDS for PostgreSQL 定價

使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫遷移至 Amazon Redshift

Piyush Goyal 和 Brian motzer，Amazon Web Services

總結

此模式提供使用 AWS Database Migration Service (AWS DMS) 和 AWS Schema Conversion Tool (AWS SCT)，將 Oracle 資料庫遷移至 Amazon Web Services (AWS) 雲端中 Amazon Redshift 雲端資料倉儲的指引。模式涵蓋現場部署或安裝在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的來源 Oracle 資料庫。它還涵蓋 Oracle 資料庫的 Amazon Relational Database Service (Amazon RDS)。

先決條件和限制

先決條件

- 在內部部署資料中心或 AWS 雲端中執行的 Oracle 資料庫
- 作用中的 AWS 帳戶
- 熟悉使用 Oracle 資料庫做為 AWS DMS 來源
- 熟悉使用 Amazon Redshift 資料庫做為 AWS DMS 的目標
- 了解 Amazon RDS、Amazon Redshift、適用的資料庫技術和 SQL
- 適用於 AWS SCT 連接器的 Java Database Connectivity (JDBC) 驅動程式，其中已安裝 AWS SCT

產品版本

- 對於自我管理的 Oracle 資料庫，AWS DMS 支援 10.2 版和更新版本 （適用於 10.x 版）、11g 和最高 12.2、18c 和 19c 版的所有 Oracle 資料庫版本。對於 AWS 管理的 Amazon RDS for Oracle 資料庫，AWS DMS 支援 11g 版 (11.2.0.4 版和更新版本） 和最多 12.2、18c 和 19c 版的所有 Oracle 資料庫版本。我們建議您使用最新版本的 AWS DMS，以獲得最全面的版本和功能支援。

Architecture

來源技術堆疊

下列其中一項：

- 內部部署 Oracle 資料庫
- EC2 執行個體上的 Oracle 資料庫

- Amazon RDS for Oracle 資料庫執行個體

目標技術堆疊

- Amazon Redshift

目標架構

從 AWS 雲端中執行的 Oracle 資料庫到 Amazon Redshift：

從現場部署資料中心中執行的 Oracle 資料庫到 Amazon Redshift：

工具

- AWS DMS - AWS Data Migration Service (AWS DMS) 可協助您快速安全地將資料庫遷移至 AWS。來源資料庫在遷移期間保持完全運作，將依賴資料庫的應用程式停機時間降至最低。AWS DMS 可以在最廣泛使用的商業和開放原始碼資料庫之間遷移您的資料。
- AWS SCT - AWS Schema Conversion Tool (AWS SCT) 可用來將現有的資料庫結構描述從一個資料庫引擎轉換為另一個資料庫引擎。它支援各種資料庫引擎做為來源，包括 Oracle、SQL Server 和 PostgresSQL。

史詩

準備遷移

任務	Description	所需的技能
驗證資料庫版本。	驗證來源和目標資料庫版本，並確認 AWS DMS 支援這些版本。如需有關支援的 Oracle 資料庫版本的資訊，請參閱使用 Oracle 資料庫做為 AWS DMS 的來源。如需使用 Amazon Redshift 做為目標的資訊，請參閱使用 Amazon Redshift 資料庫做為 AWS DMS 的目標。	DBA

任務	Description	所需的技能
建立 VPC 和安全群組。	在您的 AWS 帳戶中，如果虛擬私有雲端 (VPC) 不存在，請建立它。為來源和目標資料庫的傳出流量建立安全群組。如需詳細資訊，請參閱 Amazon Virtual Private Cloud (Amazon VPC) 文件。	系統管理員
安裝 AWS SCT。	下載並安裝最新版本的 AWS SCT 及其對應的驅動程式。如需詳細資訊，請參閱安裝、驗證和更新 AWS SCT。	DBA
為 AWS DMS 任務建立使用者。	在來源資料庫中建立 AWS DMS 使用者，並授予其讀取權限。AWS SCT 和 AWS DMS 都會使用此使用者。	DBA
測試資料庫連線。	測試 Oracle 資料庫執行個體的連線。	DBA
在 AWS SCT 中建立新專案。	開啟 AWS SCT 工具並建立新的專案。	DBA
分析要遷移的 Oracle 結構描述。	使用 AWS SCT 分析要遷移的結構描述，並產生資料庫遷移評估報告。如需詳細資訊，請參閱 AWS SCT 文件中的建立資料庫遷移評估報告。	DBA
檢閱評估報告。	檢閱報告是否有遷移可行性。有些資料庫物件可能需要手動轉換。如需報告的詳細資訊，請參閱 AWS SCT 文件中的檢視評估報告。	DBA

準備目標資料庫

任務	Description	所需的技能
建立 Amazon Redshift 叢集。	在您先前建立的 VPC 內建立 Amazon Redshift 叢集。如需詳細資訊，請參閱 Amazon Redshift 文件中的 Amazon Redshift 叢集。	DBA
建立資料庫使用者。	從 Oracle 來源資料庫擷取使用者、角色和授權的清單。在目標 Amazon Redshift 資料庫中建立使用者，並套用上一個步驟中的角色。	DBA
評估資料庫參數。	檢閱 Oracle 來源資料庫中的資料庫選項、參數、網路檔案和資料庫連結，並評估其對目標的適用性。	DBA
將任何相關設定套用至目標。	如需此步驟的詳細資訊，請參閱 Amazon Redshift 文件中的組態參考。	DBA

在目標資料庫中建立物件

任務	Description	所需的技能
在目標資料庫中建立 AWS DMS 使用者。	在目標資料庫中建立 AWS DMS 使用者，並授予讀取和寫入權限。驗證來自 AWS SCT 的連線。	DBA
轉換結構描述、檢閱 SQL 報告，並儲存任何錯誤或警告。	如需詳細資訊，請參閱 AWS SCT 文件中的使用 AWS SCT 轉換資料庫結構描述。	DBA

任務	Description	所需的技能
將結構描述變更套用至目標資料庫，或將其儲存為 .sql 檔案。	如需說明，請參閱 AWS SCT 文件中的在 AWS SCT 中儲存和套用轉換後的結構描述。	DBA
驗證目標資料庫中的物件。	驗證在目標資料庫中上一個步驟中建立的物件。重寫或重新設計任何未成功轉換的物件。	DBA
停用外部索引鍵和觸發條件。	停用任何外部索引鍵和觸發條件。這些可能會在執行 AWS DMS 的完全載入程序期間造成資料載入問題。	DBA

使用 AWS DMS 遷移資料

任務	Description	所需的技能
建立 AWS DMS 複寫執行個體。	登入 AWS 管理主控台，然後開啟 AWS DMS 主控台。在導覽窗格中，選擇複寫執行個體、建立複寫執行個體。如需詳細說明，請參閱 AWS DMS 文件中的 AWS DMS 入門中的步驟 1。	DBA
建立來源和目標端點。	建立來源和目標端點，測試從複寫執行個體到來源和目標端點的連線。如需詳細說明，請參閱 AWS DMS 文件中的 AWS DMS 入門中的步驟 2。	DBA
建立複寫任務。	建立複寫任務，然後選取適當的遷移方法。如需詳細說明，請參閱 AWS DMS 文件中的 AWS DMS 入門中的步驟 3。	DBA

任務	Description	所需的技能
啟動資料複寫。	啟動複寫任務並監控日誌是否有任何錯誤。	DBA

遷移您的應用程式

任務	Description	所需的技能
建立應用程式伺服器。	在 AWS 上建立新的應用程式伺服器。	應用程式擁有者
遷移應用程式程式碼。	將應用程式碼遷移至新的伺服器。	應用程式擁有者
設定應用程式伺服器。	設定目標資料庫和驅動程式的應用程式伺服器。	應用程式擁有者
最佳化應用程式程式碼。	最佳化目標引擎的應用程式碼。	應用程式擁有者

切換到目標資料庫

任務	Description	所需的技能
驗證使用者。	在目標 Amazon Redshift 資料庫中，驗證使用者並授予他們角色和權限。	DBA
驗證應用程式是否已鎖定。	請確定應用程式已鎖定，以防止進一步變更。	應用程式擁有者
驗證資料。	驗證目標 Amazon Redshift 資料庫中的資料。	DBA

任務	Description	所需的技能
啟用外部索引鍵和觸發條件。	在目標 Amazon Redshift 資料庫中啟用外部金鑰和觸發條件。	DBA
連線至新的資料庫。	設定應用程式以連線至新的 Amazon Redshift 資料庫。	應用程式擁有者
執行最終檢查。	在上線之前執行最終的全面系統檢查。	DBA，應用程式擁有者
上線。	使用目標 Amazon Redshift 資料庫上線。	DBA

關閉遷移專案

任務	Description	所需的技能
關閉臨時 AWS 資源。	關閉臨時 AWS 資源，例如 AWS DMS 複寫執行個體和用於 AWS SCT 的 EC2 執行個體。	DBA，系統管理員
檢閱文件。	檢閱並驗證遷移專案文件。	DBA，系統管理員
收集指標。	收集遷移專案的相關資訊，例如遷移時間、手動與工具任務的百分比，以及節省總成本。	DBA，系統管理員
關閉專案。	關閉專案並提供意見回饋。	DBA，系統管理員

相關資源

參考

- AWS DMS 使用者指南
- AWS SCT 使用者指南

- Amazon Redshift 入門指南

教學課程和影片

- 深入探討 AWS SCT 和 AWS DMS（來自 AWS re：Invent 2019 的簡報）
- AWS Database Migration Service 入門

使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫遷移至 Aurora PostgreSQL

Senthil Ramasamy，Amazon Web Services

總結

此模式說明如何使用 AWS Data Migration Service (AWS DMS) 和 AWS Schema Conversion Tool (AWS SCT)，將 Oracle 資料庫遷移至 Amazon Aurora PostgreSQL 相容版本。

此模式涵蓋現場部署的來源 Oracle 資料庫、安裝在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的 Oracle 資料庫，以及 Oracle 資料庫的 Amazon Relational Database Service (Amazon RDS)。模式會將這些資料庫轉換為 Aurora PostgreSQL 相容。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署資料中心或 AWS 雲端中的 Oracle 資料庫。
- SQL 用戶端安裝在本機電腦或 EC2 執行個體上。
- AWS SCT 連接器的 Java Database Connectivity (JDBC) 驅動程式，安裝在本機電腦或安裝 AWS SCT 的 EC2 執行個體上。

限制

- 資料庫大小限制：128 TB
- 如果來源資料庫支援商用off-the-shelf(COTS) 應用程式或特定於廠商，您可能無法將其轉換為另一個資料庫引擎。使用此模式之前，請確認應用程式支援 Aurora PostgreSQL 相容。

產品版本

- 對於自我管理的 Oracle 資料庫，AWS DMS 支援 10.2 版和更新版本 （適用於 10.x 版）、11g 版，以及高達 12.2、18c 和 19c 版的所有 Oracle 資料庫版本。如需支援 Oracle 資料庫版本的最新清單 （包括自我管理和 Amazon RDS for Oracle)，請參閱使用 Oracle 資料庫做為 AWS DMS 的來源和使用 PostgreSQL 資料庫做為 AWS DMS 的目標。
- 我們建議您使用最新版本的 AWS DMS，以獲得最全面的版本和功能支援。如需 AWS SCT 支援的 Oracle 資料庫版本的相關資訊，請參閱 AWS SCT 文件。

- Aurora 支援 Amazon Aurora PostgreSQL 版本和引擎版本中列出的 PostgreSQL 版本。PostgreSQL

Architecture

來源技術堆疊

下列其中一項：

- 內部部署 Oracle 資料庫
- EC2 執行個體上的 Oracle 資料庫
- Amazon RDS for Oracle 資料庫執行個體

目標技術堆疊

- Aurora PostgreSQL 相容

目標架構

資料遷移架構

- 從 AWS 雲端中執行的 Oracle 資料庫

- 從內部部署資料中心執行的 Oracle 資料庫

工具

- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。
- AWS Schema Conversion Tool (AWS SCT) 會自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，以支援異質資料庫遷移。

史詩

準備遷移

任務	Description	所需的技能
準備來源資料庫。	若要準備來源資料庫，請參閱 AWS SCT 文件中的使用 Oracle 資料庫做為 AWS SCT 的來源。	DBA
為 AWS SCT 建立 EC2 執行個體。	視需要建立和設定 AWS SCT 的 EC2 執行個體。	DBA
下載 AWS SCT。	下載最新版本的 AWS SCT 和相關聯的驅動程式。如需詳細資訊，請參閱 AWS SCT 文件中的安裝、驗證和更新 AWS SCT。	DBA
新增使用者和許可。	在來源資料庫中新增和驗證先決條件使用者和許可。	DBA
建立 AWS SCT 專案。	為工作負載建立 AWS SCT 專案，並連線至來源資料庫。如需說明，請參閱 AWS SCT 文件中的建立 AWS SCT 專案和新增資料庫伺服器。	DBA
評估可行性。	產生評估報告，摘要無法自動轉換之結構描述的動作項目，並提供手動轉換工作的預估值。如需詳細資訊，請參閱 AWS SCT 文件中的建立和檢閱資料庫遷移評估報告。	DBA

準備目標資料庫

任務	Description	所需的技能
建立目標 Amazon RDS 資料庫執行個體。	使用 Amazon Aurora 做為資料庫引擎，建立目標 Amazon RDS 資料庫執行個體。如需說明，請參閱《Amazon RDS 文件》中的建立 Amazon RDS 資料庫執行個體。	DBA
擷取使用者、角色和許可。	從來源資料庫擷取使用者、角色和許可的清單。	DBA
映射使用者。	將現有的資料庫使用者映射至新的資料庫使用者。	應用程式擁有者
建立使用者。	在目標資料庫中建立使用者。	DBA、應用程式擁有者
套用角色。	將上一個步驟的角色套用至目標資料庫。	DBA
檢查選項、參數、網路檔案和資料庫連結。	檢閱來源資料庫是否有選項、參數、網路檔案和資料庫連結，然後評估其對目標資料庫的適用性。	DBA
套用設定。	將任何相關設定套用至目標資料庫。	DBA

傳輸物件

任務	Description	所需的技能
設定 AWS SCT 連線。	設定目標資料庫的 AWS SCT 連線。	DBA

任務	Description	所需的技能
使用 AWS SCT 轉換結構描述。	AWS SCT 會自動將來源資料庫結構描述和大多數自訂程式碼轉換為與目標資料庫相容的格式。工具無法自動轉換的任何程式碼都會清楚標示，讓您可以手動轉換。	DBA
檢閱報告。	檢閱產生的 SQL 報告，並儲存任何錯誤和警告。	DBA
套用自動化結構描述變更。	將自動化結構描述變更套用至目標資料庫，或將其儲存為 .sql 檔案。	DBA
驗證物件。	驗證 AWS SCT 是否已在目標上建立物件。	DBA
處理未轉換的項目。	手動重寫、拒絕或重新設計任何無法自動轉換的項目。	DBA、應用程式擁有者
套用角色和使用者許可。	套用產生的角色和使用者許可，並檢閱任何例外狀況。	DBA

遷移資料

任務	Description	所需的技能
決定 方法。	決定遷移資料的方法。	DBA
建立複寫執行個體。	從 AWS DMS 主控台建立複寫執行個體。如需詳細資訊，請參閱 AWS DMS 文件中的使用 AWS DMS 複寫執行個體。	DBA

任務	Description	所需的技能
建立來源和目標端點。	若要建立端點，請遵循 AWS DMS 文件中的建立來源和目標端點中的指示。	DBA
建立複寫任務。	若要建立任務，請參閱 AWS DMS 文件中的使用 AWS DMS 任務。	DBA
啟動複寫任務並監控日誌。	如需此步驟的詳細資訊，請參閱 AWS DMS 文件中的監控 AWS DMS 任務。	DBA

遷移應用程式

任務	Description	所需的技能
分析和轉換應用程式程式碼中的 SQL 項目。	使用 AWS SCT 來分析和轉換應用程式程式碼中的 SQL 項目。當您將資料庫結構描述從一個引擎轉換到另一個引擎，您也需更新應用程式中的 SQL 程式碼，以便與新的資料庫引擎互動，取代舊引擎。您可以檢視、分析、編輯和儲存轉換後的 SQL 程式碼。	應用程式擁有者
建立應用程式伺服器。	在 AWS 上建立新的應用程式伺服器。	應用程式擁有者
遷移應用程式程式碼。	將應用程式碼遷移至新的伺服器。	應用程式擁有者
設定應用程式伺服器。	設定目標資料庫和驅動程式的應用程式伺服器。	應用程式擁有者

任務	Description	所需的技能
修正程式碼。	修正應用程式中來源資料庫引擎特有的任何程式碼。	應用程式擁有者
最佳化程式碼。	最佳化目標資料庫引擎的應用程式程式碼。	應用程式擁有者

剪下

任務	Description	所需的技能
切換到目標資料庫。	執行切換到新資料庫。	DBA
鎖定應用程式。	鎖定應用程式，避免任何進一步的變更。	應用程式擁有者
驗證變更。	驗證所有變更是否已傳播到目標資料庫。	DBA
重新導向至目標資料庫。	將新的應用程式伺服器指向目標資料庫。	應用程式擁有者
檢查所有項目。	執行最終、全面的系統檢查。	應用程式擁有者
上線。	完成最終切換任務。	應用程式擁有者

關閉專案

任務	Description	所需的技能
關閉臨時資源。	關閉臨時 AWS 資源，例如 AWS DMS 複寫執行個體和用於 AWS SCT 的 EC2 執行個體。	DBA、應用程式擁有者

任務	Description	所需的技能
更新意見回饋。	更新內部團隊的 AWS DMS 程序意見回饋。	DBA、應用程式擁有者
修訂程序和範本。	修訂 AWS DMS 程序並視需要改善範本。	DBA、應用程式擁有者
驗證文件。	檢閱並驗證專案文件。	DBA、應用程式擁有者
收集指標。	收集指標以評估遷移時間、手動與工具成本節省的百分比等。	DBA、應用程式擁有者
關閉專案。	關閉遷移專案，並向利益相關者提供意見回饋。	DBA、應用程式擁有者

相關資源

參考

- 使用 Oracle 資料庫做為 AWS DMS 的來源
- 使用 PostgreSQL 資料庫做為 AWS Database Migration Service 的目標
- Oracle Database 11g/12c 至具有 PostgreSQL 相容性 (9.6.x) 遷移手冊的 Amazon Aurora
- 具有 PostgreSQL 相容性 (12.4) 遷移手冊的 Oracle Database 19c 到 Amazon Aurora
- 將 Amazon RDS for Oracle 資料庫遷移至 Amazon Aurora PostgreSQL 相容版本
- AWS Data Migration Service
- AWS Schema Conversion Tool
- 從 Oracle 遷移至 Amazon Aurora
- Amazon RDS 定價

教學課程和影片

- 資料庫遷移Step-by-Step演練
- AWS DMS 入門
- Amazon RDS 入門

- AWS Data Migration Service （影片）
- 將 Oracle 資料庫遷移至 PostgreSQL （影片）

其他資訊

.

將資料從現場部署 Oracle 資料庫遷移至 Aurora PostgreSQL

Michelle Deng 和 Shunan Xiang，Amazon Web Services

總結

此模式提供從現場部署 Oracle 資料庫到 Amazon Aurora PostgreSQL 相容版本的資料遷移指導。它針對包含具有高資料處理語言 (DML) 活動的大型資料表的多 TB Oracle 資料庫，以停機時間最短的線上資料遷移策略為目標。Oracle Active Data Guard 待命資料庫用作從主要資料庫卸載資料遷移的來源。可在完全載入期間暫停從 Oracle 主要資料庫到待命的複寫，以避免 ORA-01555 錯誤。

具有資料類型 NUMBER 的主索引鍵 PKs) 或外部索引鍵 (FKs) 中的資料表資料欄通常用於在 Oracle 中存放整數。我們建議您在 PostgreSQL 中將這些轉換為 INT 或 BIGINT，以獲得更好的效能。您可以使用 AWS Schema Conversion Tool (AWS SCT) 來變更 PK 和 FK 資料欄的預設資料類型映射。（如需詳細資訊，請參閱 AWS 部落格文章將 NUMBER 資料類型從 Oracle 轉換為 PostgreSQL。）此模式中的資料遷移使用 AWS Database Migration Service (AWS DMS) 進行完全載入和變更資料擷取 (CDC)。

您也可以使用此模式將內部部署 Oracle 資料庫遷移至 PostgreSQL 的 Amazon Relational Database Service (Amazon RDS)，或在 Amazon Elastic Compute Cloud (Amazon EC2) 上託管的 Oracle 資料庫遷移至 Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 相容。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 已設定 Active Data Guard 待命的現場部署資料中心中的 Oracle 來源資料庫
- 在內部部署資料中心和 AWS 雲端之間設定的 AWS Direct Connect
- 熟悉使用 Oracle 資料庫做為 AWS DMS 的來源
- 熟悉使用 PostgreSQL 資料庫做為 AWS DMS 的目標

限制

- Amazon Aurora 資料庫叢集最多可建立 128 TiB 的儲存體。Amazon RDS for PostgreSQL 資料庫執行個體最多可建立 64 TiB 的儲存體。如需最新的儲存資訊，請參閱 AWS 文件中的Amazon Aurora 儲存和可靠性以及Amazon RDS 資料庫執行個體儲存。

產品版本

- AWS DMS 支援 10.2 版和更新版本（適用於 10.x 版）、11g 和最高 12.2、18c 和 19c 版的所有 Oracle 資料庫版本。如需支援版本的最新清單，請參閱 AWS 文件中的使用 Oracle 資料庫做為 AWS DMS 的來源。

Architecture

來源技術堆疊

- 已設定 Oracle Active Data Guard 待命的現場部署 Oracle 資料庫

目標技術堆疊

- Aurora PostgreSQL 相容

資料遷移架構

工具

- AWS DMS - AWS Database Migration Service (AWS DMS) 支援數個來源和目標資料庫。如需支援的 Oracle 來源和目標資料庫版本和版本的清單，請參閱 AWS DMS 文件中的使用 Oracle 資料庫做為 AWS DMS 的來源。如果 AWS DMS 不支援來源資料庫，您必須選取另一種方法來遷移階段 6 中的資料 (在範例區段中）。重要注意事項：由於這是異質遷移，因此您必須先檢查資料庫是否支援商用off-the-shelf(COTS) 應用程式。如果應用程式是 COTS，請先諮詢廠商，確認支援 Aurora PostgreSQL 相容，然後再繼續。如需詳細資訊，請參閱 AWS 文件中的 AWS DMS Step-by-Step遷移演練。
- AWS SCT - AWS Schema Conversion Tool (AWS SCT) 會自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，以促進異質資料庫遷移。工具轉換的自訂程式碼包含檢視、預存程序和函數。工具無法自動轉換的任何程式碼都會清楚標示，讓您可以自行轉換。

史詩

規劃遷移

任務	Description	所需的技能
驗證來源和目標資料庫版本。		DBA

任務	Description	所需的技能
安裝 AWS SCT 和驅動程式。		DBA
新增並驗證 AWS SCT 先決條件使用者和 grant-source 資料庫。		DBA
為工作負載建立 AWS SCT 專案，並連線至來源資料庫。		DBA
產生評估報告並評估可行性。		DBA、應用程式擁有者

準備目標資料庫

任務	Description	所需的技能
建立 Aurora PostgreSQL 相容目標資料庫。		DBA
從來源資料庫擷取使用者、角色和授予清單。		DBA
將現有的資料庫使用者映射至新的資料庫使用者。		應用程式擁有者
在目標資料庫中建立使用者。		DBA
將上一個步驟的角色套用至目標 Aurora PostgreSQL 相容資料庫。		DBA
從來源資料庫檢閱資料庫選項、參數、網路檔案和資料庫連結，並評估其對目標資料庫的適用性。		DBA、應用程式擁有者

任務	Description	所需的技能
將任何相關設定套用至目標資料庫。		DBA

準備進行資料庫物件程式碼轉換

任務	Description	所需的技能
設定目標資料庫的 AWS SCT 連線。		DBA
在 AWS SCT 中轉換結構描述，並將轉換後的程式碼儲存為 .sql 檔案。		DBA、應用程式擁有者
手動轉換任何無法自動轉換的資料庫物件。		DBA、應用程式擁有者
最佳化資料庫程式碼轉換。		DBA、應用程式擁有者
根據物件類型，將 .sql 檔案分成多個 .sql 檔案。		DBA、應用程式擁有者
驗證目標資料庫中的 SQL 指令碼。		DBA、應用程式擁有者

準備資料遷移

任務	Description	所需的技能
建立 AWS DMS 複寫執行個體。		DBA
建立來源和目標端點。	如果 PKs和 FKs的資料類型從 Oracle 中的 NUMBER 轉換為 PostgreSQL 中的	DBA

任務	Description	所需的技能
	BIGINT，請考慮在建立來源端點`numberDataTypeScale=-2` 時指定連線屬性。	

遷移資料 – 完全載入

任務	Description	所需的技能
在目標資料庫中建立結構描述和資料表。		DBA
透過分組資料表或根據資料表大小分割大型資料表來建立 AWS DMS 完全載入任務。		DBA
短暫停止來源 Oracle 資料庫上的應用程式。		應用程式擁有者
確認 Oracle 待命資料庫與主要資料庫同步，並停止從主要資料庫到待命資料庫的複寫。		DBA、應用程式擁有者
在來源 Oracle 資料庫上啟動應用程式。		應用程式擁有者
從 Oracle 待命資料庫平行啟動 AWS DMS 完全載入任務至 Aurora PostgreSQL 相容資料庫。		DBA
在完全載入完成後建立 PKs 和次要索引。		DBA
驗證資料。		DBA

遷移資料 – CDC

任務	Description	所需的技能
在 Oracle 待命與主要資料庫同步時，以及在先前任務中重新啟動應用程式之前，透過指定自訂 CDC 開始時間或系統變更號碼 (SCN) 來建立 AWS DMS 持續複寫任務。		DBA
平行啟動 AWS DMS 任務，將 Oracle 待命資料庫的持續變更複寫至 Aurora PostgreSQL 相容資料庫。		DBA
重新建立從 Oracle 主要資料庫到待命資料庫的複寫。		DBA
監控日誌，並在目標 Aurora PostgreSQL 相容資料庫與來源 Oracle 資料庫幾乎同步時停止 Oracle 資料庫上的應用程式。		DBA、應用程式擁有者
當目標與來源 Oracle 資料庫完全同步時，停止 AWS DMS 任務。		DBA
建立 FKs並驗證目標資料庫中的資料。		DBA
在目標資料庫中建立函數、檢視、觸發程序、序列和其他物件類型。		DBA
在目標資料庫中套用角色授予。		DBA

遷移應用程式

任務	Description	所需的技能
使用 AWS SCT 來分析和轉換應用程式程式碼內的 SQL 陳述式。		應用程式擁有者
在 AWS 上建立新的應用程式伺服器。		應用程式擁有者
將應用程式碼遷移至新的伺服器。		應用程式擁有者
設定目標資料庫和驅動程式的應用程式伺服器。		應用程式擁有者
修正應用程式中來源資料庫引擎特有的任何程式碼。		應用程式擁有者
最佳化目標資料庫的應用程式碼。		應用程式擁有者

剪下

任務	Description	所需的技能
將新的應用程式伺服器指向目標資料庫。		DBA、應用程式擁有者
執行健全度檢查。		DBA、應用程式擁有者
上線。		DBA、應用程式擁有者

關閉專案

任務	Description	所需的技能
關閉臨時 AWS 資源。		DBA，系統管理員
檢閱並驗證專案文件。		DBA、應用程式擁有者
收集遷移時間、手動與工具使用的百分比、節省成本和類似資料的指標。		DBA、應用程式擁有者
關閉專案並提供意見回饋。		DBA、應用程式擁有者

相關資源

參考

- 與 Aurora PostgreSQL 相容之 Oracle 資料庫：遷移手冊
- 將 Amazon RDS for Oracle Database遷移至 Amazon Aurora MySQL
- AWS DMS 網站
- AWS DMS 文件
- AWS SCT 網站
- AWS SCT 文件
- 從 Oracle 遷移至 Amazon Aurora

教學課程

- AWS DMS 入門
- Amazon RDS 入門
- AWS Database Migration Service Step-by-Step演練

使用 AWS DMS 從 SAP ASE 遷移至 Amazon RDS for SQL Server

Amit Kumar，Amazon Web Services

總結

此模式提供將 SAP Adaptive Server Enterprise (ASE) 資料庫遷移至執行 Microsoft SQL Server 之 Amazon Relational Database Service (Amazon RDS) 資料庫執行個體的指引。來源資料庫可以位於內部部署資料中心或 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。模式使用 AWS Database Migration Service (AWS DMS) 遷移資料，以及 （ 選用 ） 電腦輔助軟體工程 (CASE) 工具來轉換資料庫結構描述。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署資料中心或 EC2 執行個體上的 SAP ASE 資料庫
- 啟動並執行的目標 Amazon RDS for SQL Server 資料庫

限制

- 資料庫大小限制：64 TB

產品版本

- 僅限 SAP ASE 15.7 或 16.x 版。如需最新資訊，請參閱使用 SAP 資料庫做為 AWS DMS 的來源。
- 對於 Amazon RDS 目標資料庫，AWS DMS 支援 Amazon RDS 上的 Microsoft SQL Server 版本，適用於 Enterprise、Standard、Web 和 Express 版本。如需支援版本的最新資訊，請參閱 AWS DMS 文件。我們建議您使用最新版本的 AWS DMS，以獲得最全面的版本和功能支援。

Architecture

來源技術堆疊

- 內部部署或 Amazon EC2 執行個體上的 SAP ASE 資料庫

目標技術堆疊

- Amazon RDS for SQL Server 資料庫執行個體

來源和目標架構

從 Amazon EC2 上的 SAP ASE 資料庫到 Amazon RDS for SQL Server 資料庫執行個體：

從內部部署 SAP ASE 資料庫到 Amazon RDS for SQL Server 資料庫執行個體：

工具

- AWS Database Migration Service (AWS DMS) 是一種 Web 服務，可用來將資料從內部部署、Amazon RDS 資料庫執行個體或 EC2 執行個體上的資料庫中遷移到 AWS 服務上的資料庫，例如 Amazon RDS for SQL Server 或 EC2 執行個體。您也可以將資料庫從 AWS 服務遷移至內部部署資料庫。您可以在異質或同質資料庫引擎之間遷移資料。
- 對於結構描述轉換，您可以選擇使用 erwin Data Modeler 或 SAP PowerDesigner。

史詩

規劃遷移

任務	Description	所需的技能
驗證來源和目標資料庫版本。		DBA
識別儲存需求（儲存類型和容量）。		DBA、SysAdmin
根據容量、儲存功能和網路功能選擇適當的執行個體類型。		DBA、SysAdmin
識別來源和目標資料庫的網路存取安全需求。		DBA、SysAdmin
識別應用程式遷移策略。		DBA、SysAdmin、應用程式擁有者

設定基礎設施

任務	Description	所需的技能
建立虛擬私有雲端 (VPC) 和子網路。		SysAdmin
建立安全群組和網路存取控制清單 ACLs)。		SysAdmin
設定和啟動 Amazon RDS 資料庫執行個體。		SysAdmin

遷移資料 - 選項 1

任務	Description	所需的技能
手動遷移資料庫結構描述，或使用 CASE 工具，例如 erwin Data Modeler 或 SAP PowerDesigner。		DBA

遷移資料 - 選項 2

任務	Description	所需的技能
使用 AWS DMS 遷移資料。		DBA

遷移應用程式

任務	Description	所需的技能
遵循應用程式遷移策略。		DBA、SysAdmin、應用程式擁有者

剪下

任務	Description	所需的技能
將應用程式用戶端切換到新的基礎設施。		DBA、SysAdmin、應用程式擁有者

關閉專案

任務	Description	所需的技能
關閉臨時 AWS 資源。		DBA、SysAdmin
檢閱並驗證專案文件。		DBA、SysAdmin、應用程式擁有者
收集指標，例如遷移時間、手動與自動任務的百分比，以及節省成本。		DBA、SysAdmin、應用程式擁有者
關閉專案並提供意見回饋。		DBA、SysAdmin、應用程式擁有者

相關資源

參考

- AWS DMS 網站
- Amazon RDS 定價
- 使用 SAP ASE 資料庫做為 AWS DMS 的來源
- RDS Custom for SQL Server 的限制

教學課程和影片

- AWS DMS 入門
- Amazon RDS 入門

- AWS DMS （影片）
- Amazon RDS （影片）

使用 AWS DMS 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon Redshift

Marcelo Fernandes，Amazon Web Services

總結

此模式提供使用 AWS Data Migration Service (AWS DMS) 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon Redshift 的指引。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署資料中心中的來源 Microsoft SQL Server 資料庫
- 使用 Amazon Redshift 資料庫做為 AWS DMS 目標的已完成先決條件，如 AWS DMS 文件所述

產品版本

- SQL Server 2005-2019、Enterprise、Standard、Workgroup、Developer 和 Web 版本。如需支援版本的最新清單，請參閱 AWS 文件中的使用 Microsoft SQL Server 資料庫做為 AWS DMS 的來源。

Architecture

來源技術堆疊

- 內部部署 Microsoft SQL Server 資料庫

目標技術堆疊

- Amazon Redshift

資料遷移架構

工具

- AWS DMS 是一種資料遷移服務，支援多種類型的來源和目標資料庫。如需有關支援與 AWS DMS 搭配使用的 Microsoft SQL Server 資料庫版本和版本的資訊，請參閱 AWS DMS 文件中的使用 Microsoft SQL Server 資料庫做為 AWS DMS 的來源。如果 AWS DMS 不支援您的來源資料庫，您必須選取用於資料遷移的替代方法。

史詩

規劃遷移

任務	Description	所需的技能
驗證來源和目標資料庫版本和引擎。		DBA
識別目標伺服器執行個體的硬體需求。		DBA，系統管理員
識別儲存需求（儲存類型和容量）。		DBA，系統管理員
根據容量、儲存功能和網路功能選擇適當的執行個體類型。		DBA，系統管理員
識別來源和目標資料庫的網路存取安全需求。		DBA，系統管理員
識別應用程式遷移策略。		DBA、應用程式擁有者、系統管理員

設定基礎設施

任務	Description	所需的技能
建立 Virtual Private Cloud (VPC)	如需詳細資訊，請參閱 AWS 文件中的在 VPC 中使用資料庫執行個體。	系統管理員

任務	Description	所需的技能
建立安全群組。		系統管理員
設定和啟動 Amazon Redshift 叢集。	如需詳細資訊，請參閱《Amazon Redshift 文件》中的建立範例 Amazon Redshift 叢集。	DBA，系統管理員

遷移資料

任務	Description	所需的技能
使用 AWS DMS 從 Microsoft SQL Server 資料庫遷移資料。		DBA

遷移應用程式

任務	Description	所需的技能
遵循應用程式遷移策略。		DBA、應用程式擁有者、系統管理員

剪下

任務	Description	所需的技能
將應用程式用戶端切換到新的基礎設施。		DBA、應用程式擁有者、系統管理員

關閉專案

任務	Description	所需的技能
關閉臨時資源。		DBA，系統管理員
檢閱並驗證專案文件。		DBA、應用程式擁有者、系統管理員
收集指標，例如遷移時間、手動與自動任務的百分比，以及節省成本。		DBA、應用程式擁有者、系統管理員
關閉專案並提供意見回饋。		DBA、應用程式擁有者、系統管理員

相關資源

參考

- AWS DMS 文件
- Amazon Redshift 文件
- Amazon Redshift 定價

教學課程和影片

- AWS DMS 入門
- 開始使用 Amazon RedShift
- 使用 Amazon Redshift 資料庫做為 AWS Database Migration Service 的目標
- AWS DMS （影片）

使用 AWS SCT 資料擷取代理程式將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon Redshift

Neha Thakur，Amazon Web Services

總結

此模式概述使用 AWS Schema Conversion Tool (AWS SCT) 資料擷取代理程式，將內部部署 Microsoft SQL Server 來源資料庫遷移至 Amazon Redshift 目標資料庫的步驟。代理程式是與 AWS SCT 整合的外部程式，但在別處執行資料轉換，並代表您與其他 AWS 服務互動。

先決條件和限制

先決條件

- 用於內部部署資料中心內資料倉儲工作負載的 Microsoft SQL Server 來源資料庫
- 作用中的 AWS 帳戶

產品版本

- Microsoft SQL Server 2008 版或更新版本。如需支援版本的最新清單，請參閱 AWS SCT 文件。

Architecture

技術堆疊來源

- 內部部署 Microsoft SQL Server 資料庫

技術堆疊目標

- Amazon Redshift

資料遷移架構

工具

- AWS Schema Conversion Tool (AWS SCT) 會自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，以處理異質資料庫遷移。當來源和目標資料庫非常不同時，您可以使用

AWS SCT 代理程式來執行額外的資料轉換。如需詳細資訊，請參閱 AWS 文件中的將資料從現場部署資料倉儲遷移至 Amazon Redshift。

最佳實務

- AWS SCT 的最佳實務
- Amazon Redshift 的最佳實務

史詩

準備遷移

任務	Description	所需的技能
驗證來源和目標資料庫版本和引擎。		DBA
識別目標伺服器執行個體的硬體需求。		DBA、SysAdmin
識別儲存需求（儲存類型和容量）。		DBA、SysAdmin
選擇適當的執行個體類型（容量、儲存功能、網路功能）。		DBA、SysAdmin
識別來源和目標資料庫的網路存取安全需求。		DBA、SysAdmin
選擇應用程式遷移策略。		DBA、SysAdmin、應用程式擁有者

設定基礎設施

任務	Description	所需的技能
建立虛擬私有雲端 (VPC) 和子網路。		SysAdmin
建立安全群組。		SysAdmin
設定和啟動 Amazon Redshift 叢集。		SysAdmin

遷移資料

任務	Description	所需的技能
使用 AWS SCT 資料擷取代理程式遷移資料。		DBA

遷移應用程式

任務	Description	所需的技能
遵循所選的應用程式遷移策略。		DBA、SysAdmin、應用程式擁有者

切換到目標資料庫

任務	Description	所需的技能
將應用程式用戶端切換到新的基礎設施。		DBA、SysAdmin、應用程式擁有者

關閉專案

任務	Description	所需的技能
關閉臨時 AWS 資源。		DBA、SysAdmin
檢閱並驗證專案文件。		DBA、SysAdmin、應用程式擁有者
收集指標，例如遷移時間、手動與自動任務的百分比，以及節省成本。		DBA、SysAdmin、應用程式擁有者
關閉專案並提供任何意見回饋。		DBA、SysAdmin、應用程式擁有者

相關資源

參考

- AWS SCT 使用者指南
- 使用資料擷取代理程式
- Amazon Redshift 定價

教學課程和影片

- AWS Schema Conversion Tool入門
- 開始使用 Amazon RedShift

將舊版應用程式從 Oracle Pro*C 遷移至 ECPG

Sai Parthasaradhi 和 Mahesh Balumuri，Amazon Web Services

摘要

大多數具有內嵌 SQL 程式碼的舊版應用程式會使用 Oracle Pro*C 前置編譯器來存取資料庫。當您將這些 Oracle 資料庫遷移至 PostgreSQL 的 Amazon Relational Database Service (Amazon RDS) 或 Amazon Aurora PostgreSQL 相容版本時，您必須將應用程式程式碼轉換為與 PostgreSQL 中預編譯器相容的格式，稱為 ECPG。此模式說明如何將 Oracle Pro*C 程式碼轉換為 PostgreSQL ECPG 中的同等程式碼。

如需 Pro*C 的詳細資訊，請參閱 Oracle 文件。如需 ECPG 的簡介，請參閱其他資訊一節。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 相容資料庫
- 在內部部署執行的 Oracle 資料庫

工具

- 下一節中列出的 PostgreSQL 套件。
- AWS CLI – AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可透過命令列 shell 中的命令與 AWS 服務互動。透過最少的組態，您可以從命令提示中執行 AWS CLI 命令，該命令會實作與瀏覽器型 AWS 管理主控台所提供功能相同的功能。

史詩

在 CentOS 或 RHEL 上設定建置環境

任務	Description	所需的技能
安裝 PostgreSQL 套件。	使用以下命令安裝所需的 PostgreSQL 套件。 `yum update -y`	應用程式開發人員、DevOps 工程師

任務	Description	所需的技能

	`yum install -y yum-utils`
	`rpm -ivh https://download.postgresql.org/pub/repos/yum/reporpms/EL-8-x86_64/pgdg-redhat-repo-latest.noarch.rpm`
	`dnf -qy module disable postgresql`	

安裝標頭檔案和程式庫。	使用以下命令安裝包含標頭檔案和程式庫的 `postgresql12-devel` 套件。在開發和執行階段環境中安裝 套件，以避免執行階段環境中發生錯誤。
	`dnf -y install postgresql12-devel`
	`yum install ncompress zip ghostscript jq unzip wget git -y`
	僅針對開發環境，也請執行下列命令。
	`yum install zlib-devel make -y`
	`ln -s /usr/pgsql-12/bin/ecpg /usr/bin/`
		應用程式開發人員、DevOps 工程師

設定環境路徑變數。	設定 PostgreSQL 用戶端程式庫的環境路徑。 `export PATH=$PATH:/usr/pgsql-12/bin`	應用程式開發人員、DevOps 工程師

任務	Description	所需的技能

視需要安裝其他軟體。	如有必要，請在 Oracle 中安裝 pgLoader 作為 SQL*Loader 的替代。
	
	`wget -O /etc/yum.repos.d/pgloader-ccl.repo https://dl.packager.io/srv/opf/pgloader-ccl/master/installer/el/7.repo`
	`yum install pgloader-ccl -y`
	`ln -s /opt/pgloader-ccl/bin/pgloader /usr/bin/`
	
	如果您從 Pro*C 模組呼叫任何 Java 應用程式，請安裝 Java。
	
	`yum install java -y`
	
	安裝 ant 以編譯 Java 程式碼。
	
	`yum install ant -y`
		應用程式開發人員、DevOps 工程師

任務	Description	所需的技能

安裝 AWS CLI。	安裝 AWS CLI 來執行命令，以從您的應用程式與 AWS Secrets Manager 和 Amazon Simple Storage Service (Amazon S3) 等 AWS 服務互動。Amazon S3
	`cd /tmp/`
	`curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"`
	`unzip awscliv2.zip`
	`./aws/install -i /usr/local/aws-cli -b /usr/local/bin --update`
		應用程式開發人員、DevOps 工程師

識別要轉換的程式。	識別您要從 Pro*C 轉換為 ECPG 的應用程式。	應用程式開發人員、應用程式擁有者

將 Pro*C 程式碼轉換為 ECPG

任務	Description	所需的技能
移除不需要的標頭。	移除 PostgreSQL 中不需要的`include` 標頭，例如 `oci.h`、 `oratypes`和 `sqlda`。	應用程式擁有者、應用程式開發人員
更新變數宣告。	為用作主機變數的所有變數宣告新增`EXEC SQL`陳述式。 從您的應用程式中移除宣告，`EXEC SQL VAR`例如以下內容。	應用程式開發人員、應用程式擁有者

任務	Description	所需的技能
	`EXEC SQL VAR query IS STRING(2048);`	

任務	Description	所需的技能

更新 ROWNUM 功能。	PostgreSQL 中無法使用 ROWNUM函數。將此取代為 SQL 查詢中的ROW_NUMBER 視窗函數。
	
	Pro*C 程式碼：
	
	`SELECT SUBSTR(RTRIM(FILE_NAME,'.txt'),12) INTO :gcpclFileseq FROM (SELECT FILE_NAME FROM DEMO_FILES_TABLE WHERE FILE_NAME LIKE '%POC%' ORDER BY FILE_NAME DESC) FL2 WHERE ROWNUM <=1 ORDER BY ROWNUM;`
	
	ECPG 程式碼：
	
	`SELECT SUBSTR(RTRIM(FILE_NAME,'.txt'),12) INTO :gcpclFileseq FROM (SELECT FILE_NAME , ROW_NUMBER() OVER (ORDER BY FILE_NAME DESC) AS ROWNUM FROM demo_schema.DEMO_FILES_TABLE WHERE FILE_NAME LIKE '%POC%' ORDER BY FILE_NAME DESC) FL2`
		應用程式開發人員、應用程式擁有者

任務	Description	所需的技能
	`WHERE ROWNUM <=1 ORDER BY ROWNUM;`	

更新函數參數以使用別名變數。	在 PostgreSQL 中，函數參數無法用作主機變數。使用別名變數覆寫它們。
	
	Pro*C 程式碼：
	
	`int processData(int referenceId){ EXEC SQL char col_val[100]; EXEC SQL select column_name INTO :col_val from table_name where col=:referenceId; }`
	
	ECPG 程式碼：
	
	`int processData(int referenceIdParam){ EXEC SQL int referenceId = referenceIdParam; EXEC SQL char col_val[100]; EXEC SQL select column_name INTO :col_val from table_name where col=:referenceId; }`
		應用程式開發人員、應用程式擁有者

任務	Description	所需的技能
更新結構類型。	typedef 如果struct類型變數用作主機變數，請使用定義 EXEC SQL BEGIN和END區塊中的struct類型。如果struct類型是在標頭 (.h) 檔案中定義，請包含EXEC SQL包含陳述式的檔案。 Pro*C 程式碼： 標頭檔案 (demo.h) `struct s_partitio on_ranges {  char   sc_table_ group[31];  char   sc_table_ name[31];  char   sc_range_ value[10]; }; struct s_partitio on_ranges_ind {   short    ss_table_ group;   short    ss_table_ name;   short    ss_range_ value; };` ECPG 程式碼： 標頭檔案 (demo.h) `EXEC SQL BEGIN DECLARE  SECTION;`	應用程式開發人員、應用程式擁有者

任務	Description	所需的技能

```
typedef struct
{
  char   sc_table_
group[31];
  char   sc_table_
name[31];
  char   sc_range_
value[10];
} s_partition_ranges;
typedef struct
{
  short    ss_table_
group;
  short    ss_table_
name;
  short    ss_range_
value;
} s_partition_ranges
_ind;
EXEC SQL END DECLARE
 SECTION;
```

Pro*C 檔案 (demo.pc)

```
#include "demo.h"
struct s_partiti
on_ranges gc_partit
ion_data[MAX_PART_
TABLE] ;
struct s_partiti
on_ranges_ind
 gc_partition_data_
ind[MAX_PART_TABLE] ;
```

ECPG 檔案 (demo.pc)

```
exec sql include
 "demo.h"
EXEC SQL BEGIN DECLARE
 SECTION;
```

任務	Description	所需的技能
	`s_partition_ranges gc_partition_data[ MAX_PART_TABLE] ; s_partition_ranges_ind gc_partition_data_ ind[MAX_PART_TABLE] ; EXEC SQL END DECLARE SECTION;`	

修改邏輯以從游標擷取。	若要使用陣列變數從游標擷取多個資料列，請將程式碼變更為使用 FETCH FORWARD。
	
	Pro*C 程式碼：
	
	`EXEC SQL char aPoeFiles [MAX_FILES][FILENA ME_LENGTH]; EXEC SQL FETCH filename_ cursor into :aPoeFile s;`
	
	ECPG 程式碼：
	
	`EXEC SQL char aPoeFiles [MAX_FILES][FILENA ME_LENGTH]; EXEC SQL int fetchSize = MAX_FILES; EXEC SQL FETCH FORWARD :fetchSiz e filename_cursor into :aPoeFiles;`
		應用程式開發人員、應用程式擁有者

任務	Description	所需的技能

修改沒有傳回值的套件呼叫。	沒有傳回值的 Oracle 套件函數應該使用 指標變數呼叫。如果您的應用程式包含多個具有相同名稱的函數，或不明類型的函數產生執行時間錯誤，請輸入將值傳送到資料類型。
	
	Pro*C 程式碼：
	
	`void ProcessData (char *data , int id) { EXEC SQL EXECUTE BEGIN pkg_demo.process_data (:data, :id); END; END-EXEC; }`
	
	ECPG 程式碼：
	
	`void ProcessData (char *dataParam, int idParam ) { EXEC SQL char *data = dataParam; EXEC SQL int id = idParam; EXEC SQL short rowInd; EXEC SQL short rowInd = 0;`
		應用程式開發人員、應用程式擁有者

任務	Description	所需的技能
	EXEC SQL SELECT pkg_demo.process_data (inp_data => :data::text, inp_id => :id) INTO :rowInd; }	

任務	Description	所需的技能

重寫 SQL_CURSOR 變數。	重寫SQL_CURSOR 變數及其實作。
	
	Pro*C 程式碼：
	
	`/* SQL Cursor */`
	`SQL_CURSOR`
	` demo_cursor;`
	`EXEC SQL`
	` ALLOCATE :demo_cursor;`
	`EXEC SQL EXECUTE`
	`  BEGIN`
	`      pkg_demo.`
	`get_cursor(`
	`        demo_cur=`
	`>:demo_cursor`
	`      );`
	`  END;`
	`END-EXEC;`
	
	ECPG 程式碼：
	
	`EXEC SQL DECLARE`
	` demo_cursor CURSOR FOR`
	` SELECT`
	`         * from`
	`    pkg_demo.open_file`
	`name_rc(`
	`            demo_cur=`
	`>refcursor`
	`          ) ;`
	`EXEC SQL char open_file`
	`name_rcInd[100];`
	`# As the below function`
	` returns cursor_name`
	` as`
	`# return we need to`
	` use char[] type as`
	` indicator.`
		應用程式開發人員、應用程式擁有者

任務	Description	所需的技能
	`EXEC SQL SELECT pkg_demo.get_cursor ( demo_cur=>'demo_cursor' ) INTO :open_filename_rcInd;`	
套用常見的遷移模式。	• 變更 SQL 查詢，使其與 PostgreSQL 相容。 • 當 ECPG 不支援匿名區塊時，將其移至資料庫。 • 移除 PostgreSQL 不支援的`dbms_application_info` 邏輯。 • 在游標關閉後移動`EXEC SQL COMMIT`陳述式。如果您在迴圈中遞交查詢以從游標擷取記錄，則會關閉游標，並顯示游標不存在錯誤。 • 如需有關在 ECPG 和錯誤代碼中處理例外狀況的資訊，請參閱 PostgreSQL 文件中的錯誤處理。	應用程式開發人員、應用程式擁有者
視需要啟用偵錯。	若要在偵錯模式下執行 ECPG 程式，請在主要函數區塊中新增下列命令。 `ECPGdebug(1, stderr);`	應用程式開發人員、應用程式擁有者

編譯 ECPG 程式

任務	Description	所需的技能

建立 ECPG 的可執行檔。	如果您有名為 的內嵌 SQL C 來源檔案`prog1.pgc` ，您可以使用下列命令序列來建立可執行程式。
	`ecpg prog1.pgc`
	`cc -I/usr/local/pgsql/include -c prog1.c`
	`cc -o prog1 prog1.o -L/usr/local/pgsql/lib -lecpg`
		應用程式開發人員、應用程式擁有者

建立要編譯的 make 檔案。	建立 make 檔案以編譯 ECPG 程式，如下列範例檔案所示。
	`CFLAGS ::= $(CFLAGS) -I/usr/pgsql-12/include -g -Wall`
	`LDFLAGS ::= $(LDFLAGS) -L/usr/pgsql-12/lib -Wl,-rpath,/usr/pgsql-12/lib`
	`LDLIBS ::= $(LDLIBS) -lecpg`
	`PROGRAMS = test`
	`.PHONY: all clean`
	`%.c: %.pgc`
	`  ecpg $<`
	`all: $(PROGRAMS)`
	`clean:`
	`  rm -f $(PROGRAMS) $(PROGRAMS:%=%.c) $(PROGRAMS:%=%.o)`
		應用程式開發人員、應用程式擁有者

測試應用程式。

任務	Description	所需的技能
測試代碼。	測試轉換後的應用程式程式碼，以確保其正常運作。	應用程式開發人員、應用程式擁有者、測試工程師

相關資源

- ECPG - C 中的內嵌 SQL (PostgreSQL 文件）
- 錯誤處理 (PostgreSQL 文件）
- 為什麼要使用 Oracle Pro*C/C++ 前置編譯器 (Oracle 文件）

其他資訊

PostgreSQL 具有內嵌 SQL 前置編譯器 ECPG，相當於 Oracle Pro*C 前置編譯器。ECPG 會將內嵌 SQL 陳述式的 C 程式轉換為標準 C 程式碼，方法是將 SQL 呼叫取代為特殊函數呼叫。然後，可以使用任何 C 編譯器工具鏈處理輸出檔案。

輸入和輸出檔案

ECPG 會將您在命令列指定的每個輸入檔案轉換為對應的 C 輸出檔案。如果輸入檔案名稱沒有副檔名，則會假設 .pgc。檔案的副檔名由 取代`.c`，以建構輸出檔案名稱。不過，您可以使用 `-o`選項覆寫預設輸出檔案名稱。

如果您使用破折號 (-) 做為輸入檔案名稱， ECPG 會從標準輸入 讀取程式，並寫入標準輸出，除非您使用 `-o`選項覆寫該程式。

標頭檔案

當 PostgreSQL 編譯器編譯預先處理的 C 程式碼檔案時，它會在 PostgreSQL `include`目錄中尋找 ECPG 標頭檔案。因此，您可能必須使用 `-I`選項，將編譯器指向正確的目錄 （ 例如，`-I/usr/local/pgsql/include`)。

Libraries (程式庫)

使用 C 程式碼搭配內嵌 SQL `libecpg`的程式必須連結到程式庫。例如，您可以使用連結器選項 `-L/usr/local/pgsql/lib -lecpg`。

轉換後的 ECPG 應用程式會透過內嵌 SQL 程式庫 (ecpglib) 呼叫libpq程式庫中的函數，並使用標準前端/後端通訊協定與 PostgreSQL 伺服器通訊。

將虛擬產生的資料欄從 Oracle 遷移至 PostgreSQL

Veeranjaneyulu Grandhi、Rajesh Madiwale 和 Ramesh Pathuri，Amazon Web Services

總結

在 11 版和更早版本中，PostgreSQL 不提供直接等同於 Oracle 虛擬資料欄的功能。在從 Oracle 資料庫遷移至 PostgreSQL 第 11 版或更早版本時，處理虛擬產生的資料欄很困難，原因有兩個：

- 遷移期間看不到虛擬資料欄。
- PostgreSQL 不支援版本 12 之前的`generate`表達式。

不過，有模擬類似功能的解決方法。當您使用 AWS Database Migration Service (AWS DMS) 將資料從 Oracle Database 遷移至 PostgreSQL 第 11 版及更早版本時，您可以使用觸發函數在虛擬產生的資料欄中填入值。此模式提供 Oracle 資料庫和 PostgreSQL 程式碼的範例，您可以用於此目的。在 AWS 上，您可以將 Amazon Relational Database Service (Amazon RDS) 用於 PostgreSQL，或將 Amazon Aurora PostgreSQL 相容版本用於 PostgreSQL 資料庫。

從 PostgreSQL 第 12 版開始，支援產生的資料欄。產生的資料欄可以即時從其他資料欄值計算，也可以計算和儲存。PostgreSQL 產生的資料欄類似於 Oracle 虛擬資料欄。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 來源 Oracle 資料庫
- Target PostgreSQL 資料庫 (Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 相容）
- PL/pgSQL 編碼專業知識

限制

- 僅適用於第 12 版之前的 PostgreSQL 版本。
- 適用於 Oracle 資料庫 11g 版或更新版本。
- 資料遷移工具不支援虛擬資料欄。
- 僅適用於相同資料表中定義的資料欄。
- 如果虛擬產生的資料欄是指確定性的使用者定義函數，則不能用作分割索引鍵資料欄。

- 表達式的輸出必須是純量值。它無法傳回 Oracle 提供的資料類型、使用者定義的類型LOB、 或 LONG RAW。
- 針對虛擬資料欄定義的索引等同於 PostgreSQL 中的函數型索引。
- 必須收集資料表統計資料。

工具

- pgAdmin 4 是 PostgreSQL 的開放原始碼管理工具。此工具提供圖形界面，可簡化資料庫物件的建立、維護和使用。
- Oracle SQL Developer 是免費的整合開發環境，可在傳統和雲端部署中使用 Oracle 資料庫中的 SQL。

史詩

建立來源和目標資料庫資料表

任務	Description	所需的技能
建立來源 Oracle 資料庫資料表。	在 Oracle Database 中，使用下列陳述式建立具有虛擬產生資料欄的資料表。 `CREATE TABLE test.generated_column ( CODE NUMBER, STATUS VARCHAR2(12) DEFAULT 'PreOpen', FLAG CHAR(1) GENERATED ALWAYS AS (CASE UPPER(STATUS) WHEN 'OPEN' THEN 'N' ELSE 'Y' END) VIRTUAL VISIBLE );` 在此來源資料表中，資料STATUS欄中的資料會透過 AWS DMS 遷移至目標資料	DBA，應用程式開發人員

任務	Description	所需的技能
	庫。不過，資料FLAG欄會使用 generate by功能填入，因此 AWS DMS 在遷移期間看不到此資料欄。若要實作 的功能generated by，您必須使用目標資料庫中的觸發條件和函數來填入資料FLAG欄中的值，如下圖所示。	
在 AWS 上建立目標 PostgreSQL 資料表。	使用下列陳述式在 AWS 上建立 PostgreSQL 資料表。 `CREATE TABLE test.generated_column ( code integer not null, status character varying(12) not null , flag character(1) );` 在此資料表中，資料status欄是標準資料欄。flag 資料欄將根據資料欄中的資料產生資料status欄。	DBA，應用程式開發人員

建立觸發函數來處理 PostgreSQL 中的虛擬資料欄

任務	Description	所需的技能
建立 PostgreSQL 觸發。	在 PostgreSQL 中，建立觸發條件。 `CREATE TRIGGER tgr_gen_column`	DBA，應用程式開發人員

任務	Description	所需的技能
	`AFTER INSERT OR UPDATE OF status ON test.generated_column FOR EACH ROW EXECUTE FUNCTION test.tgf_gen_column();`	

任務	Description	所需的技能
建立 PostgreSQL 觸發函數。	在 PostgreSQL 中，為觸發建立 函數。此函數會填入由應用程式或 AWS DMS 插入或更新的虛擬資料欄，並驗證資料。 `CREATE OR REPLACE FUNCTION test.tgf_gen_column() RETURNS trigger AS $VIRTUAL_COL$ BEGIN IF (TG_OP = 'INSERT') THEN IF (NEW.flag IS NOT NULL) THEN RAISE EXCEPTION 'ERROR: cannot insert into column "flag"' USING DETAIL = 'Column "flag" is a generated column.'; END IF; END IF; IF (TG_OP = 'UPDATE') THEN IF (NEW.flag::VARCHAR != OLD.flag::varchar) THEN RAISE EXCEPTION 'ERROR: cannot update column "flag"' USING DETAIL = 'Column "flag" is a generated column.'; END IF; END IF; IF TG_OP IN ('INSERT','UPDATE') THEN IF (old.flag is NULL) OR (coalesce(old.stat`	DBA，應用程式開發人員

任務	Description	所需的技能
	us,'') != coalesce(new.status,'')) THEN UPDATE test.gene rated_column SET flag = (CASE UPPER(status) WHEN 'OPEN' THEN 'N' ELSE 'Y' END) WHERE code = new.code; END IF; END IF; RETURN NEW; END $VIRTUAL_COL$ LANGUAGE plpgsql;	

使用 AWS DMS 測試資料遷移

任務	Description	所需的技能
建立複寫執行個體。	若要建立複寫執行個體，請遵循 AWS DMS 文件中的指示。複寫執行個體應與來源和目標資料庫位於相同的虛擬私有雲端 (VPC) 中。	DBA，應用程式開發人員
建立來源和目標端點。	若要建立端點，請遵循 AWS DMS 文件中的指示。	DBA，應用程式開發人員
測試端點連線。	您可以透過指定 VPC 和複寫執行個體，然後選擇執行測試來測試端點連線。	DBA，應用程式開發人員
建立並啟動完整載入任務。	如需說明，請參閱 AWS DMS 文件中的建立任務和完全載入任務設定。	DBA，應用程式開發人員

任務	Description	所需的技能
驗證虛擬資料欄的資料。	比較來源和目標資料庫中虛擬資料欄中的資料。您可以手動驗證資料或撰寫此步驟的指令碼。	DBA，應用程式開發人員

相關資源

- AWS Database Migration Service 入門 (AWS DMS 文件）
- 使用 Oracle 資料庫做為 AWS DMS 的來源 (AWS DMS 文件）
- 使用 PostgreSQL 資料庫做為 AWS DMS 的目標 (AWS DMS 文件）
- PostgreSQL 中產生的資料欄 (PostgreSQL 文件）
- 觸發函數 (PostgreSQL 文件）
- Oracle 資料庫中的虛擬資料欄 (Oracle 文件）

在 Aurora PostgreSQL 相容上設定 Oracle UTL_FILE 功能

Rakesh Raghav 和 anuradha chintha，Amazon Web Services

總結

在從 Oracle 遷移到 Amazon Web Services (AWS) 雲端上 Amazon Aurora PostgreSQL 相容版本的過程中，您可能會遇到多個挑戰。例如，遷移依賴 Oracle UTL_FILE公用程式的程式碼一律是一項挑戰。在 Oracle PL/SQL 中，UTL_FILE套件會搭配基礎作業系統用於檔案操作，例如讀取和寫入。UTL_FILE 公用程式適用於伺服器和用戶端機器系統。

Amazon Aurora PostgreSQL 相容是受管資料庫產品。因此，無法存取資料庫伺服器上的檔案。此模式會逐步引導您整合 Amazon Simple Storage Service (Amazon S3) 和 Amazon Aurora PostgreSQL 相容，以實現UTL_FILE功能子集。使用此整合，我們可以建立和使用檔案，而無需使用第三方擷取、轉換和載入 (ETL) 工具或服務。

或者，您可以設定 Amazon CloudWatch 監控和 Amazon SNS 通知。

我們建議在生產環境中實作此解決方案之前，先徹底測試此解決方案。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- AWS Database Migration Service (AWS DMS) 專業知識
- PL/pgSQL 編碼的專業知識
- Amazon Aurora PostgreSQL 相容叢集
- S3 儲存貯體

限制

此模式不提供可取代 Oracle UTL_FILE公用程式的功能。不過，您可以進一步增強步驟和範本程式碼，以實現資料庫現代化目標。

產品版本

- Amazon Aurora PostgreSQL 相容版本 11.9

Architecture

目標技術堆疊

- Amazon Aurora PostgreSQL 相容
- Amazon CloudWatch
- Amazon Simple Notification Service (Amazon SNS)
- Amazon S3

目標架構

下圖顯示解決方案的高階表示法。

1. 檔案會從應用程式上傳到 S3 儲存貯體。
2. aws_s3 延伸模組會使用 PL/pgSQL 存取資料，並將資料上傳至 Aurora PostgreSQL 相容。

工具

- Amazon Aurora PostgreSQL 相容 – Amazon Aurora PostgreSQL 相容版本是全受管、PostgreSQL 相容和 ACID 相容關聯式資料庫引擎。它結合了高階商業資料庫的速度和可靠性，以及開放原始碼資料庫的成本效益。
- AWS CLI – AWS Command Line Interface (AWS CLI) 是管理 AWS 服務的統一工具。只需下載和設定一個工具，您就可以從命令列控制多個 AWS 服務，並透過指令碼將其自動化。
- Amazon CloudWatch – Amazon CloudWatch 會監控 Amazon S3 資源和使用方式。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是網際網路的儲存體。在此模式中，Amazon S3 提供儲存層來接收和存放檔案，以供取用和往返 Aurora PostgreSQL 相容叢集傳輸。
- aws_s3 – aws_s3延伸模組整合 Amazon S3 和 Aurora PostgreSQL 相容。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 會協調和管理發佈者和用戶端之間的訊息傳遞或傳送。在此模式中，Amazon SNS 用於傳送通知。
- pgAdmin – pgAdmin 是 Postgres 的開放原始碼管理工具。pgAdmin 4 提供圖形界面，用於建立、維護和使用資料庫物件。

Code

為了實現所需的功能， 模式會使用類似於 的命名來建立多個函數UTL_FILE。其他資訊區段包含這些函數的程式碼基底。

在程式碼中，將 取代testaurorabucket為測試 S3 儲存貯體的名稱。us-east-1 將 取代為測試 S3 儲存貯體所在的 AWS 區域。

史詩

整合 Amazon S3 和 Aurora PostgreSQL 相容

任務	Description	所需的技能
設定 IAM 政策。	建立 AWS Identity and Access Management (IAM) 政策，以授予 S3 儲存貯體和其中物件的存取權。如需程式碼，請參閱其他資訊一節。	AWS 管理員，DBA
將 Amazon S3 存取角色新增至 Aurora PostgreSQL。	建立兩個 IAM 角色：一個角色用於讀取，一個角色用於寫入存取 Amazon S3。將兩個角色連接到 Aurora PostgreSQL 相容叢集： • S3Export 功能的一個角色 • S3Import 功能的一個角色 如需詳細資訊，請參閱 Aurora PostgreSQL 相容文件，了解如何將資料匯入和匯出至 Amazon S3。	AWS 管理員，DBA

在 Aurora PostgreSQL 相容中設定擴充功能

任務	Description	所需的技能
建立 aws_commons 延伸模組。	`aws_commons` 延伸模組是`aws_s3`延伸模組的相依性。	DBA、開發人員
建立 aws_s3 延伸模組。	`aws_s3` 延伸模組會與 Amazon S3 互動。	DBA、開發人員

驗證 Amazon S3 和 Aurora PostgreSQL 相容整合

任務	Description	所需的技能
測試將檔案從 Amazon S3 匯入 Aurora PostgreSQL。	若要測試將檔案匯入 Aurora PostgreSQL 相容，請建立範例 CSV 檔案並將其上傳至 S3 儲存貯體。根據 CSV 檔案建立資料表定義，並使用 `aws_s3.table_import_from_s3` 函數將檔案載入資料表。	DBA、開發人員
測試將檔案從 Aurora PostgreSQL 匯出至 Amazon S3。	若要測試從 Aurora PostgreSQL 相容匯出檔案，請建立測試資料表、填入資料，然後使用 `aws_s3.query_export_to_s3` 函數匯出資料。	DBA、開發人員

若要模擬 UTL_FILE 公用程式，請建立包裝函式

任務	Description	所需的技能
建立 utl_file_utility 結構描述。	結構描述會將包裝函式放在一起。若要建立結構描述，請執行下列命令。	DBA、開發人員

任務	Description	所需的技能
	`CREATE SCHEMA utl_file_utility;`	
建立 file_type 類型。	若要建立 `file_type` 類型，請使用下列程式碼。 `CREATE TYPE utl_file_utility.file_type AS ( p_path character varying(30), p_file_name character varying );`	DBA/開發人員
建立初始化函數。	`init` 函數會初始化常見的變數，例如 `bucket`或 `region`。如需程式碼，請參閱其他資訊一節。	DBA/開發人員
建立包裝函式。	建立包裝函式 `fopen`、`put_line`和 `fclose`。如需程式碼，請參閱其他資訊一節。	DBA、開發人員

測試包裝函式

任務	Description	所需的技能
在寫入模式下測試包裝函式。	若要在寫入模式下測試包裝函式，請使用其他資訊區段中提供的程式碼。	DBA、開發人員
在附加模式下測試包裝函式。	若要在附加模式下測試包裝函式，請使用其他資訊區段中提供的程式碼。	DBA、開發人員

相關資源

- Amazon S3 整合
- Amazon S3
- Aurora
- Amazon CloudWatch
- Amazon SNS

其他資訊

設定 IAM 政策

建立下列政策。

政策名稱	JSON

S3IntRead	{
	"Version": "2012-10-17",
	"Statement": [
	{
	"Sid": "S3integrationtest
	",
	"Effect": "Allow",
	"Action": [
	"s3:GetObject",
	"s3:ListBucket"
	],
	"Resource": [
	"arn:aws:s3:::testaurorabuc
	ket/*",
	"arn:aws:s3:::testaurorabuc
	ket"
	]
	}
	]
	}

S3IntWrite	{ "Version": "2012-10-17", "Statement": [

```
        {
            "Sid": "S3integrationtest
",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",

                "s3:ListBucket"
            ],
            "Resource": [
    "arn:aws:s3:::testaurorabucket/
*",                "arn:aws:s3:::test
aurorabucket"
            ]
        }
    ]
}
```

建立初始化函數

若要初始化常見變數，例如 bucket或 region，請使用下列程式碼建立 init函數。

```
CREATE OR REPLACE FUNCTION utl_file_utility.init(
    )
    RETURNS void
    LANGUAGE 'plpgsql'

    COST 100
    VOLATILE
AS $BODY$
BEGIN
      perform set_config
      ( format( '%s.%s','UTL_FILE_UTILITY', 'region' )
      , 'us-east-1'::text
      , false );

      perform set_config
      ( format( '%s.%s','UTL_FILE_UTILITY', 's3bucket' )
      , 'testaurorabucket'::text
      , false );
END;
$BODY$;
```

建立包裝函式

建立 fopen、 put_line和 fclose 包裝函式。

fopen

```
CREATE OR REPLACE FUNCTION utl_file_utility.fopen(
    p_file_name character varying,
    p_path character varying,
    p_mode character DEFAULT 'W'::bpchar,
    OUT p_file_type utl_file_utility.file_type)
    RETURNS utl_file_utility.file_type
    LANGUAGE 'plpgsql'

    COST 100
    VOLATILE
AS $BODY$
declare
    v_sql character varying;
    v_cnt_stat integer;
    v_cnt integer;
    v_tabname character varying;
    v_filewithpath character varying;
    v_region character varying;
    v_bucket character varying;

BEGIN
    /*initialize common variable */
    PERFORM utl_file_utility.init();
    v_region := current_setting( format( '%s.%s', 'UTL_FILE_UTILITY', 'region' ) );
    v_bucket :=  current_setting( format( '%s.%s', 'UTL_FILE_UTILITY', 's3bucket' ) );

    /* set tabname*/
    v_tabname := substring(p_file_name,1,case when strpos(p_file_name,'.') = 0 then
 length(p_file_name) else strpos(p_file_name,'.') - 1 end );
    v_filewithpath := case when NULLif(p_path,'') is null then p_file_name else
 concat_ws('/',p_path,p_file_name) end ;
    raise notice 'v_bucket %, v_filewithpath % , v_region %', v_bucket,v_filewithpath,
 v_region;

    /* APPEND MODE HANDLING; RETURN EXISTING FILE DETAILS IF PRESENT ELSE CREATE AN
 EMPTY FILE */
    IF p_mode = 'A' THEN
```

```
       v_sql := concat_ws('','create temp table if not exists ', v_tabname,' (col1
text)');
       execute v_sql;

       begin
       PERFORM aws_s3.table_import_from_s3
           ( v_tabname,
           '',
           'DELIMITER AS ''#''',
           aws_commons.create_s3_uri
           (     v_bucket,
               v_filewithpath ,
               v_region)
           );
       exception
           when others then
            raise notice 'File load issue ,%',sqlerrm;
            raise;
       end;
       execute concat_ws('','select count(*) from ',v_tabname) into v_cnt;

       IF v_cnt > 0
       then
           p_file_type.p_path := p_path;
           p_file_type.p_file_name := p_file_name;
       else
           PERFORM aws_s3.query_export_to_s3('select ''''',
                           aws_commons.create_s3_uri(v_bucket, v_filewithpath,
v_region)
                             );

           p_file_type.p_path := p_path;
           p_file_type.p_file_name := p_file_name;
       end if;
       v_sql := concat_ws('','drop table ', v_tabname);
       execute v_sql;
   ELSEIF p_mode = 'W' THEN
           PERFORM aws_s3.query_export_to_s3('select ''''',
                           aws_commons.create_s3_uri(v_bucket, v_filewithpath,
v_region)
                             );
           p_file_type.p_path := p_path;
           p_file_type.p_file_name := p_file_name;
   END IF;
```

```
EXCEPTION
        when others then
            p_file_type.p_path := p_path;
            p_file_type.p_file_name := p_file_name;
            raise notice 'fopenerror,%',sqlerrm;
            raise;
END;
$BODY$;
```

put_line

```
CREATE OR REPLACE FUNCTION utl_file_utility.put_line(
    p_file_name character varying,
    p_path character varying,
    p_line text,
    p_flag character DEFAULT 'W'::bpchar)
    RETURNS boolean
    LANGUAGE 'plpgsql'

    COST 100
    VOLATILE
AS $BODY$
/*****************************************************************************
* Write line, p_line in windows format to file, p_fp - with carriage return
* added before new line.
*****************************************************************************/
declare
    v_sql varchar;
    v_ins_sql varchar;
    v_cnt INTEGER;
    v_filewithpath character varying;
    v_tabname  character varying;
    v_bucket character varying;
    v_region character varying;

BEGIN
 PERFORM utl_file_utility.init();

/* check if temp table already exist */

 v_tabname := substring(p_file_name,1,case when strpos(p_file_name,'.') = 0 then
 length(p_file_name) else strpos(p_file_name,'.') - 1 end );
```

```
 v_sql := concat_ws('','select count(1) FROM pg_catalog.pg_class c LEFT JOIN
 pg_catalog.pg_namespace n ON n.oid = c.relnamespace where n.nspname like ''pg_temp_
%'''
                         ,' AND pg_catalog.pg_table_is_visible(c.oid) AND
 Upper(relname) = Upper( '''
                         ,  v_tabname ,''' ) ');

 execute v_sql into v_cnt;

  IF v_cnt = 0 THEN
         v_sql := concat_ws('','create temp table ',v_tabname,' (col text)');
        execute v_sql;
        /* CHECK IF APPEND MODE */
        IF upper(p_flag) = 'A' THEN
            PERFORM utl_file_utility.init();
            v_region := current_setting( format( '%s.%s', 'UTL_FILE_UTILITY',
 'region' ) );
            v_bucket :=  current_setting( format( '%s.%s', 'UTL_FILE_UTILITY',
 's3bucket' ) );

            /* set tabname*/
            v_filewithpath := case when NULLif(p_path,'') is null then p_file_name else
 concat_ws('/',p_path,p_file_name) end ;

            begin
               PERFORM aws_s3.table_import_from_s3
                     ( v_tabname,
                          '',
                       'DELIMITER AS ''#''',
                        aws_commons.create_s3_uri
                           ( v_bucket,
                               v_filewithpath,
                               v_region    )
                     );
            exception
                when others then
                    raise notice  'Error Message : %',sqlerrm;
                    raise;
            end;
        END IF;
    END IF;
    /* INSERT INTO TEMP TABLE */
    v_ins_sql := concat_ws('','insert into ',v_tabname,' values(''',p_line,''')');
```

```
    execute v_ins_sql;
    RETURN TRUE;
    exception
            when others then
                raise notice  'Error Message : %',sqlerrm;
                raise;
END;
$BODY$;
```

關閉

```
CREATE OR REPLACE FUNCTION utl_file_utility.fclose(
    p_file_name character varying,
    p_path character varying)
    RETURNS boolean
    LANGUAGE 'plpgsql'

    COST 100
    VOLATILE
AS $BODY$
DECLARE
    v_filewithpath character varying;
    v_bucket character varying;
    v_region character varying;
    v_tabname character varying;
    v_sql character varying;
BEGIN
      PERFORM utl_file_utility.init();

    v_region := current_setting( format( '%s.%s', 'UTL_FILE_UTILITY', 'region' ) );
    v_bucket :=  current_setting( format( '%s.%s', 'UTL_FILE_UTILITY', 's3bucket' ) );

    v_tabname := substring(p_file_name,1,case when strpos(p_file_name,'.') = 0 then
 length(p_file_name) else strpos(p_file_name,'.') - 1 end );
    v_filewithpath := case when NULLif(p_path,'') is null then p_file_name else
 concat_ws('/',p_path,p_file_name) end ;

    raise notice 'v_bucket %, v_filewithpath % , v_region %', v_bucket,v_filewithpath,
 v_region ;

    /* exporting to s3 */
    perform aws_s3.query_export_to_s3
        (concat_ws('','select * from ',v_tabname,'  order by ctid asc'),
```

```
            aws_commons.create_s3_uri(v_bucket, v_filewithpath, v_region)
        );
    v_sql := concat_ws('','drop table ', v_tabname);
    execute v_sql;
    RETURN TRUE;
EXCEPTION
       when others then
     raise notice 'error fclose %',sqlerrm;
     RAISE;
END;
$BODY$;
```

測試您的設定和包裝函式

使用以下匿名程式碼區塊來測試您的設定。

測試寫入模式

下列程式碼會在 S3 儲存貯s3inttest體中寫入名為 的檔案。

```
do $$
declare
l_file_name varchar := 's3inttest' ;
l_path varchar := 'integration_test' ;
l_mode char(1) := 'W';
l_fs utl_file_utility.file_type ;
l_status boolean;

begin
select * from
utl_file_utility.fopen( l_file_name, l_path , l_mode ) into l_fs ;
raise notice 'fopen : l_fs : %', l_fs;

select * from
utl_file_utility.put_line( l_file_name, l_path ,'this is test file:in s3bucket: for
 test purpose', l_mode ) into l_status ;
raise notice 'put_line : l_status %', l_status;

select * from utl_file_utility.fclose( l_file_name , l_path ) into l_status ;
raise notice 'fclose : l_status %', l_status;

end;
$$
```

測試附加模式

下列程式碼會將行附加至先前測試中建立的 s3inttest 檔案。

```
do $$
declare
l_file_name varchar := 's3inttest' ;
l_path varchar := 'integration_test' ;
l_mode char(1) := 'A';
l_fs utl_file_utility.file_type ;
l_status boolean;

begin
select * from
utl_file_utility.fopen( l_file_name, l_path , l_mode ) into l_fs ;
raise notice 'fopen : l_fs : %', l_fs;


select * from
utl_file_utility.put_line( l_file_name, l_path ,'this is test file:in s3bucket: for
 test purpose : append 1', l_mode ) into l_status ;
raise notice 'put_line : l_status %', l_status;

select * from
utl_file_utility.put_line( l_file_name, l_path ,'this is test file:in s3bucket : for
 test purpose : append 2', l_mode ) into l_status ;
raise notice 'put_line : l_status %', l_status;

select * from utl_file_utility.fclose( l_file_name , l_path ) into l_status ;
raise notice 'fclose : l_status %', l_status;

end;
$$
```

Amazon SNS 通知

或者，您可以在 S3 儲存貯體上設定 Amazon CloudWatch 監控和 Amazon SNS 通知。如需詳細資訊，請參閱監控 Amazon S3 和設定 Amazon SNS 通知。

從 Oracle 遷移到 Amazon Aurora PostgreSQL 後驗證資料庫物件

Venkatramana Chintha 和 Eduardo Valentim，Amazon Web Services

總結

此模式描述將 step-by-step方法。 PostgreSQL

此模式概述資料庫物件驗證的使用案例和步驟；如需更多詳細資訊，請參閱 AWS 資料庫部落格上的使用 AWS SCT 和 AWS DMS 在遷移後驗證資料庫物件。 https://aws.amazon.com/blogs/

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 已遷移至 Aurora PostgreSQL 相容資料庫的現場部署 Oracle 資料庫。
- 已套用 Aurora PostgreSQL 相容資料庫 AmazonRDSDataFullAccess 政策的登入憑證。
- 此模式使用 Aurora Serverless 資料庫叢集的查詢編輯器，可在 Amazon Relational Database Service (Amazon RDS) 主控台中使用。不過，您可以將此模式與任何其他查詢編輯器搭配使用。

限制

- Oracle SYNONYM 物件無法在 PostgreSQL 中使用，但可以透過檢視或 SET search_path 查詢進行部分驗證。
- Amazon RDS 查詢編輯器僅適用於特定 AWS 區域和特定 MySQL 和 PostgreSQL 版本。

Architecture

工具

工具

- Amazon Aurora PostgreSQL 相容版本 – Aurora PostgreSQL 相容是全受管、PostgreSQL 相容且 ACID 相容的關聯式資料庫引擎，結合了高階商業資料庫的速度和可靠性，以及開放原始碼資料庫的簡單性和成本效益。

- Amazon RDS – Amazon Relational Database Service (Amazon RDS) 可讓您更輕鬆地在 AWS 雲端中設定、操作和擴展關聯式資料庫。其能為產業標準的關聯式資料庫提供具成本效益、可調整大小的容量，並管理常見的資料庫管理任務。
- Aurora Severless 的查詢編輯器 – 查詢編輯器可協助您在 Amazon RDS 主控台中執行 SQL 查詢。您可以在 Aurora Serverless 資料庫叢集上執行任何有效的 SQL 陳述式，包括資料處理和資料定義陳述式。

若要驗證物件，請使用「附件」區段中「物件驗證指令碼」檔案中的完整指令碼。使用下表做為參考。

Oracle 物件	要使用的指令碼
套件	查詢 1
表格	查詢 3
檢視	查詢 5
序列	查詢 7
觸發	查詢 9
主索引鍵	查詢 11
索引	查詢 13
檢查限制	查詢 15
外部索引鍵	查詢 17

PostgreSQL 物件	要使用的指令碼
套件	查詢 2
表格	查詢 4
檢視	查詢 6
序列	查詢 8

觸發	查詢 10
主索引鍵	查詢 12
索引	查詢 14
檢查限制	查詢 16
外部索引鍵	查詢 18

史詩

驗證來源 Oracle 資料庫中的物件

任務	Description	所需的技能
在來源 Oracle 資料庫中執行「套件」驗證查詢。	從「附件」區段下載並開啟「物件驗證指令碼」檔案。透過用戶端程式連線至來源 Oracle 資料庫。從「物件驗證指令碼」檔案執行「查詢 1」驗證指令碼。重要：在查詢中輸入您的 Oracle 使用者名稱，而不是「your_schema」。請務必記錄查詢結果。	開發人員，DBA
執行「資料表」驗證查詢。	從「物件驗證指令碼」檔案執行「查詢 3」指令碼。請務必記錄查詢結果。	開發人員，DBA
執行「檢視」驗證查詢。	從「物件驗證指令碼」檔案執行「查詢 5」指令碼。請務必記錄查詢結果。	開發人員，DBA
執行「序列」計數驗證。	從「物件驗證指令碼」檔案執行「查詢 7」指令碼。請務必記錄查詢結果。	開發人員，DBA

任務	Description	所需的技能
執行「觸發器」驗證查詢。	從「物件驗證指令碼」檔案執行「查詢 9」指令碼。請務必記錄查詢結果。	開發人員，DBA
執行「主索引鍵」驗證查詢。	從「物件驗證指令碼」檔案執行「查詢 11」指令碼。請務必記錄查詢結果。	開發人員，DBA
執行「索引」驗證查詢。	從「物件驗證指令碼」檔案執行「查詢 13」驗證指令碼。請務必記錄查詢結果。	開發人員，DBA
執行「檢查限制條件」驗證查詢。	從「物件驗證指令碼」檔案執行「查詢 15」指令碼。請務必記錄查詢結果。	開發人員，DBA
執行「外部金鑰」驗證查詢。	從「物件驗證指令碼」檔案執行「查詢 17」驗證指令碼。請務必記錄查詢結果。	開發人員，DBA

驗證目標 Aurora PostgreSQL 相容資料庫中的物件

任務	Description	所需的技能
使用查詢編輯器連線至目標 Aurora PostgreSQL 相容資料庫。	登入 AWS 管理主控台並開啟 Amazon RDS 主控台。在右上角，選擇您建立 Aurora PostgreSQL 相容資料庫的 AWS 區域。在導覽窗格中，選擇「資料庫」，然後選擇目標 Aurora PostgreSQL 相容資料庫。在「動作」中，選擇「查詢」。重要：如果您之前尚未連線到資料庫，則「連線至資料庫」頁面會開啟。然後，您	開發人員，DBA

任務	Description	所需的技能
	需要輸入資料庫資訊，例如使用者名稱和密碼。	
執行「套件」驗證查詢。	從「附件」區段中的「物件驗證指令碼」檔案執行「查詢 2」指令碼。請務必記錄查詢結果。	開發人員，DBA
執行「資料表」驗證查詢。	返回 Aurora PostgreSQL 相容資料庫的查詢編輯器，並從「物件驗證指令碼」檔案執行「查詢 4」指令碼。請務必記錄查詢結果。	開發人員，DBA
執行「檢視」驗證查詢。	返回 Aurora PostgreSQL 相容資料庫的查詢編輯器，並從「物件驗證指令碼」檔案執行「查詢 6」指令碼。請務必記錄查詢結果。	開發人員，DBA
執行「序列」計數驗證。	返回 Aurora PostgreSQL 相容資料庫的查詢編輯器，並從「物件驗證指令碼」檔案執行「查詢 8」指令碼。請務必記錄查詢結果。	開發人員，DBA
執行「觸發器」驗證查詢。	返回 Aurora PostgreSQL 相容資料庫的查詢編輯器，並從「物件驗證指令碼」檔案執行「查詢 10」指令碼。請務必記錄查詢結果。	開發人員，DBA

任務	Description	所需的技能
執行「主索引鍵」驗證查詢。	返回 Aurora PostgreSQL 相容資料庫的查詢編輯器，並從「物件驗證指令碼」檔案執行「查詢 12」指令碼。請務必記錄查詢結果。	開發人員，DBA
執行「索引」驗證查詢。	返回 Aurora PostgreSQL 相容資料庫的查詢編輯器，並從「物件驗證指令碼」檔案執行「查詢 14」指令碼。請務必記錄查詢結果。	開發人員，DBA
執行「檢查限制條件」驗證查詢。	從「物件驗證指令碼」檔案執行「查詢 16」指令碼。請務必記錄查詢結果。	開發人員，DBA
執行「外部金鑰」驗證查詢。	從「物件驗證指令碼」檔案執行「查詢 18」驗證指令碼。請務必記錄查詢結果。	開發人員，DBA

比較來源和目標資料庫驗證記錄

任務	Description	所需的技能
比較並驗證兩個查詢結果。	比較 Oracle 和 Aurora PostgreSQL 相容資料庫的查詢結果，以驗證所有物件。如果它們都相符，則所有物件都已成功驗證。	開發人員，DBA

相關資源

- 使用 AWS SCT 和 AWS DMS 驗證遷移後的資料庫物件
- Amazon Aurora 功能：PostgreSQL 相容版本

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

重新託管

主題

- 加速探索 Microsoft 工作負載並將其遷移至 AWS
- 在重新託管遷移至 期間建立防火牆請求的核准程序 AWS
- 將 EC2 Windows 執行個體擷取並遷移至 AWS Managed Services 帳戶
- 將 Couchbase Server 資料庫遷移至 Amazon EC2
- 使用日誌運送將 LUW 的 Db2 遷移至 Amazon EC2，以減少中斷時間
- 將 LUW 的 Db2 遷移至具有高可用性災難復原的 Amazon EC2
- 使用 appcmd.exe 將 IIS 託管應用程式遷移至 Amazon EC2
- 使用 Application Migration Service 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon EC2
- 在 上將 F5 BIG-IP 工作負載遷移至 F5 BIG-IP VE AWS 雲端
- 使用二進位方法將內部部署 Go Web 應用程式遷移至 AWS Elastic Beanstalk
- AWS 使用 將內部部署 SFTP 伺服器遷移至 AWS Transfer for SFTP
- 使用 AWS Application Migration Service 將內部部署 VM 遷移至 Amazon EC2
- 使用 將小型資料集從內部部署遷移至 Amazon S3 AWS SFTP
- 將內部部署 Oracle 資料庫遷移至 Amazon EC2 上的 Oracle
- 使用 Oracle Data Pump 將內部部署 Oracle 資料庫遷移至 Amazon EC2
- 使用 AWS MGN 將 RHEL BYOL 系統遷移至包含 AWS 授權的執行個體
- 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon EC2
- 在 AWS 雲端中重新託管內部部署工作負載：遷移檢查清單
- 使用 Amazon FSx 設定 SQL Server Always On FCI 的異地同步備份基礎設施
- 使用 BMC Discovery 查詢來擷取遷移資料以進行遷移規劃

加速探索 Microsoft 工作負載並將其遷移至 AWS

Ali Alzand，Amazon Web Services

總結

此模式說明如何使用 Migration Validator Toolkit PowerShell 模組來探索 Microsoft 工作負載並將其遷移至 AWS。此模組的運作方式是針對與任何 Microsoft 工作負載相關聯的一般任務，執行多次檢查和驗證。例如，模組會檢查是否有可能連接多個磁碟的執行個體，或使用多個 IP 地址的執行個體。如需模組可執行之檢查的完整清單，請參閱模組 GitHub 頁面上的檢查一節。

您的組織可透過 Migration Validator Toolkit PowerShell 模組，減少用來探索 Microsoft 工作負載上有哪些應用程式和服務正在運作的時間和心力。此模組也可助您識別工作負載的組態，以便瞭解 AWS 是否支援您的組態。此模組也會提供後續步驟和緩解動作的建議，以避免在遷移之前、期間或之後出現任何組態錯誤。

先決條件和限制

先決條件

- 本機管理員帳戶
- PowerShell 4.0

限制

- 僅適用於 Microsoft Windows Server 2012 R2 或更新版本

工具

工具

- PowerShell 4.0

程式碼儲存庫

此模式的 Migration Validator Toolkit PowerShell 模組可在 GitHubmigration-validator-toolkit-for-microsoft-workloads 中取得。

史詩

在單一目標上執行 Migration Validator Toolkit PowerShell 模組

任務	Description	所需的技能

下載、擷取、匯入和叫用模組。	選擇下列其中一種方法來下載和部署模組：
	• 執行 PowerShell 指令碼
	• 下載並解壓縮 .zip 檔案
	• 複製 GitHub 儲存庫
	
	執行 PowerShell 指令碼
	
	在 PowerShell 中，執行下列範例程式碼：
	
	`#MigrationValidatorToolkit`
	`$uri = 'https://github.com/aws-samples/migration-validator-toolkit-for-microsoft-workloads/archive/refs/heads/main.zip'`
	`$destination = (Get-Location).Path`
	`if ((Test-Path -Path "$destination\MigrationValidatorToolkit.zip" -PathType Leaf) -or (Test-Path -Path "$destination\MigrationValidatorToolkit")) {`
	`write-host "File $destination\MigrationValidatorToolkit.zip or folder`
		系統管理員

任務	Description	所需的技能
	$destination\Migra tionValidatorToolkit found, exiting" }else { Write-host "Enable TLS 1.2 for this PowerShell session only." [Net.ServicePointM anager]::SecurityP rotocol = [Net.Secu rityProtocolType]: :Tls12 $webClient = New-Object System.Ne t.WebClient Write-host "Downloading Migration ValidatorToolkit.zip" $webClient.Downloa dFile($uri, "$destina tion\MigrationVali datorToolkit.zip") Write-host "MigrationValidato rToolkit.zip download successfully" Add-Type -Assembly "system.io.compres sion.filesystem" [System.IO.Compres sion.ZipFile]::Ext ractToDirectory("$ destination\Migrat ionValidatorToolki t.zip","$destinati on\MigrationValida torToolkit") Write-host "Extracting Migration ValidatorToolkit.zip complete successfully"	

任務	Description	所需的技能

	<pre>Import-Module "$destination\MigrationValidatorToolkit\migration-validator-toolkit-for-microsoft-workloads-main\MigrationValidatorToolkit.psm1"; Invoke-MigrationValidatorToolkit
	}</pre>
	此程式碼會從 .zip 檔案下載模組。然後，程式碼會擷取、匯入和叫用模組。
	
	下載並解壓縮 .zip 檔案
	
	1. 下載 .zip 檔案 （下載）。
	2. 解壓縮 .zip 檔案。
	3. 請遵循本指南手動叫用模組案例中的步驟。
	
	複製 GitHub 儲存庫
	
	1. 若要複製 GitHubmigration-validator-toolkit-for-microsoft-workloads，請在終端機視窗中執行下列 Git 命令：
	
	<pre>git clone https://github.com/aws-samples/migration-validator-toolkit-for-microsoft-workloads.git</pre>	

任務	Description	所需的技能
	2. 請遵循本指南手動叫用模組案例中的步驟。	

任務	Description	所需的技能

手動叫用模組。	1. 前往存放下載模組的目錄。
	2. 若要產生您選擇的輸出，請在 PowerShell 中以管理員身分執行下列其中一個命令：
	
	Format-Tableformat：
	`Import-Module .\MigrationValidatorToolkit.psm1;Invoke-MigrationValidatorToolkit`
	
	Format-Listformat：
	`Import-Module .\MigrationValidatorToolkit.psm1;Invoke-MigrationValidatorToolkit -List`
	
	Out-GridViewformat：
	`Import-Module .\MigrationValidatorToolkit.psm1;Invoke-MigrationValidatorToolkit -GridView`
	
	ConvertTo-Csvformat：
	`Import-Module .\MigrationValidatorToolkit.psm1;Invoke-MigrationValidatorToolkit -csv`
		系統管理員

在多個目標上執行 Migration Validator Toolkit PowerShell 模組

任務	Description	所需的技能
下載 .zip 檔案或複製 GitHub 儲存庫。	請選擇下列其中一個選項： • 下載zip 檔案。（下載）。 • 若要複製 GitHubmigration-validator-toolkit-for-microsoft-workloads，請在終端機視窗中執行下列 Git 命令： `git clone https://github.com/aws-samples/migration-validator-toolkit-for-microsoft-workloads.git`	系統管理員
更新 server.csv 清單。	如果您下載了 .zip 檔案，請遵循下列步驟： 1. 解壓縮 .zip 檔案。 2. 前往 `Migration ValidatorToolkit\Inputs\` 目錄。 3. `serverlist.csv` 使用目標電腦的主機名稱進行更新。	系統管理員
叫用 模組。	您可以使用網域內任何使用具有目標電腦管理員存取權的網域使用者的電腦。 1. 下載原始程式碼做為 .zip 檔案並解壓縮檔案。	系統管理員

任務	Description	所需的技能
	2. 身為 PowerShell 中的管理員，請執行下列命令： `Import-Module .\MigrationValidatorToolkit.psm1;Invoke-DomainComputers` 輸出 .csv 檔案會以字首名稱儲存在 中`MigrationValidatorToolkit\Outputs\folder DomainComputers_MigrationAutomations_YYYY-MM-DDTHH-MM-SS` 。	

疑難排解

問題	解決方案
`MigrationValidatorToolkit` 會將有關執行、命令和錯誤的資訊寫入執行中主機上的日誌檔案。	您可以在下列位置手動檢視日誌檔案： 1. 前往 `MigrationValidatorToolkit\logs\` 目錄。 2. 找到日誌檔案。日誌檔案名稱的格式為：`ComputerName_MigrationValidatorToolkit_YYYY-MM-SSTHH-MM-SS.log`

相關資源

- 將 Microsoft 工作負載遷移至 AWS 的選項、工具和最佳實務 (AWS 規範性指導)

- Microsoft 遷移模式 (AWS 方案指引）
- AWS 免費雲端遷移服務 (AWS 文件）
- 預先定義的啟動後動作 （應用程式行銷文件）

其他資訊

常見問答集

我可以在哪裡執行 Migration Validator Toolkit PowerShell 模組？

您可以在 Microsoft Windows Server 2012 R2 或更新版本上執行模組。

何時執行此模組？

我們建議您在遷移旅程的評估階段執行模組。

模組是否會修改我現有的伺服器？

否。此模組中的所有動作皆為唯讀。

執行模組需要多長時間？

執行模組通常需要 1-5 分鐘，但取決於您伺服器的資源配置。

模組需要執行哪些許可？

您必須從本機管理員帳戶執行模組。

我可以在實體伺服器上執行模組嗎？

是，只要作業系統是 Microsoft Windows Server 2012 R2 或更新版本即可。

如何為多部伺服器大規模執行模組？

若要大規模在多個加入網域的電腦上執行模組，請遵循本指南多個目標的執行遷移驗證工具組 PowerShell 模組中的步驟。對於未加入網域的電腦，請使用遠端調用或在本機執行模組，方法是遵循本指南單一目標上執行遷移驗證工具組 PowerShell 模組中的步驟。

在重新託管遷移至 期間建立防火牆請求的核准程序 AWS

Srikanth Rangavajhala，Amazon Web Services

摘要

如果您想要在 上使用 AWS Application Migration Service或 Cloud Migration Factory AWS 來重新託管遷移至 AWS 雲端，其中一個先決條件是您必須保持 TCP 連接埠 443 和 1500 開啟。一般而言，開啟這些防火牆連接埠需要您的資訊安全 (InfoSec) 團隊核准。

此模式概述在重新託管遷移至 期間，從 InfoSec 團隊取得防火牆請求核准的程序 AWS 雲端。您可以使用此程序來避免 InfoSec 團隊拒絕您的防火牆請求，這會變得昂貴且耗時。防火牆請求程序在 AWS 遷移顧問和主管之間有兩個審核和核准步驟，這些人員會與您的 InfoSec 和應用程式團隊合作以開啟防火牆連接埠。

此模式假設您正在規劃與 AWS 顧問或組織中的遷移專家進行重新託管遷移。如果您的組織沒有防火牆核准程序或防火牆請求空白核准表單，您可以使用此模式。如需詳細資訊，請參閱此模式的限制一節。如需 Application Migration Service 網路需求的詳細資訊，請參閱 Application Migration Service 文件中的網路需求。

先決條件和限制

先決條件

- 與您組織的 AWS 顧問或遷移專家進行規劃的重新託管遷移
- 遷移堆疊所需的連接埠和 IP 資訊
- 現有和未來的狀態架構圖
- 有關內部部署和目的地基礎設施、連接埠和zone-to-zone流量流程的防火牆資訊
- 防火牆請求檢閱檢查清單 （ 已連接 ）
- 防火牆請求文件，根據您組織的需求進行設定
- 防火牆檢閱者和核准者的聯絡人清單，包括下列角色：
  - 防火牆請求提交者 – AWS 遷移專家或顧問。防火牆請求提交者也可以是您組織的遷移專家。
  - 防火牆請求檢閱者 – 一般而言，這是來自 的單一聯絡點 (SPOC) AWS。
  - 防火牆請求核准者 – InfoSec 團隊成員。

限制

- 此模式說明一般防火牆請求核准程序。個別組織的需求可能有所不同。

- 請務必追蹤防火牆請求文件的變更。

下表顯示此模式的使用案例。

您的組織是否具有現有的防火牆核准程序？	您的組織是否有現有的防火牆請求表單？	建議動作
是	是	與 AWS 顧問或遷移專家合作，以實作組織的程序。
否	是	使用此模式的防火牆核准程序。使用您組織的 AWS 顧問或遷移專家來提交防火牆請求括號核准表單。
否	否	使用此模式的防火牆核准程序。使用您組織的 AWS 顧問或遷移專家來提交防火牆請求括號核准表單。

Architecture

下圖顯示防火牆請求核准程序的步驟。

工具

您可以使用 Palo Alto Networks 或 SolarWinds 等掃描器工具來分析和驗證防火牆和 IP 地址。

史詩

分析防火牆請求

任務	Description	所需的技能
分析連接埠和 IP 地址。	防火牆請求提交者完成初始分析，以了解所需的防火牆連接埠和 IP 地址。完成後，他們會	AWS 雲端工程師、遷移專家

任務	Description	所需的技能
	請求您的 InfoSec 團隊開啟所需的連接埠並映射 IP 地址。	

驗證防火牆請求

任務	Description	所需的技能
驗證防火牆資訊。	AWS 雲端 工程師會與您的 InfoSec 團隊安排會議。在此會議期間，工程師會檢查並驗證防火牆請求資訊。 一般而言，防火牆請求提交者與防火牆請求者是同一個人。如果觀察到或建議任何項目，此驗證階段可能會根據核准者提供的意見回饋而變得反覆。	AWS 雲端工程師、遷移專家
更新防火牆請求文件。	在 InfoSec 團隊分享其意見回饋後，防火牆請求文件會編輯、儲存和重新上傳。本文件會在每次反覆運算後更新。 我們建議您將此文件存放在版本控制的儲存資料夾中。這表示會追蹤並正確套用所有變更。	AWS 雲端工程師、遷移專家

提交防火牆請求

任務	Description	所需的技能
提交防火牆請求。	在防火牆請求核准者核准防火牆空白核准請求之後， AWS 雲端 工程師會提交防火牆請	AWS 雲端工程師、遷移專家

任務	Description	所需的技能
	求。請求會指定必須開放的連接埠，以及映射和更新 所需的 IP 地址 AWS 帳戶。 您可以在提交防火牆請求後提出建議或提供意見回饋。我們建議您自動化此意見回饋程序，並透過定義的工作流程機制傳送任何編輯。	

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

將 EC2 Windows 執行個體擷取並遷移至 AWS Managed Services 帳戶

Anil Kunapareddy 和 Venkatramana Chintha，Amazon Web Services

總結

此模式說明將 Amazon Elastic Compute Cloud (Amazon EC2) Windows 執行個體遷移和擷取至 Amazon Web Services (AWS) Managed Services (AMS) 帳戶的step-by-step程序。AMS 可協助您更有效率且安全地管理執行個體。AMS 提供營運彈性、增強安全性和合規性，並協助您最佳化容量並降低成本。

此模式從您已遷移至 AMS 帳戶中臨時子網路的 EC2 Windows 執行個體開始。您可以使用各種遷移服務和工具來執行此任務，例如 AWS Application Migration Service。

若要變更 AMS 受管環境，您可以針對特定操作或動作建立並提交變更請求 (RFC)。使用 AMS 工作負載擷取 (WIGS) RFC，您可以將執行個體擷取至 AMS 帳戶並建立自訂 Amazon Machine Image (AMI)。然後，您可以透過提交另一個 RFC 來建立 EC2 堆疊來建立 AMS 受管 EC2 執行個體。如需詳細資訊，請參閱 AMS 文件中的 AMS 工作負載擷取。

先決條件和限制

先決條件

- 作用中的 AMS 受管 AWS 帳戶
- 現有的登陸區域
- 在 AMS 受管 VPC 中進行變更的許可
- 您 AMS 帳戶中預備子網路中的 Amazon EC2 Windows 執行個體
- 完成使用 AMS WIGS 遷移工作負載的一般先決條件
- 完成使用 AMS WIGS 遷移工作負載的 Windows 先決條件

限制

- 此模式適用於操作 Windows Server 的 EC2 執行個體。此模式不適用於執行其他作業系統的執行個體，例如 Linux。

Architecture

來源技術堆疊

AMS 帳戶中預備子網路中的 Amazon EC2 Windows 執行個體

目標技術堆疊

AWS Managed Services (AMS) 管理的 Amazon EC2 Windows 執行個體

目標架構

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以使用 Amazon EC2 視需要啟動任意數量或任意數量的虛擬伺服器，也可以向外擴展或向內擴展。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Managed Services (AMS) 提供 AWS 基礎設施的持續管理，包括 AWS 工作負載的監控、事件管理、安全指導、修補程式支援和備份，協助您更有效率且安全地操作。

其他服務

- PowerShell 是在 Windows、Linux 和 macOS 上執行的 Microsoft 自動化和組態管理程式。

史詩

在執行個體上設定設定

任務	Description	所需的技能
變更 DNS 用戶端設定。	1. 在來源 EC2 執行個體上，以管理員身分開啟命令提示字元，輸入 `gpedit.msc` ，然後按 Enter 鍵。 2. 在本機群組政策編輯器中，導覽至電腦組態、管理範本、網路、DNS 用戶端。 3. 針對主要 DNS 尾碼，選擇未設定。	遷移工程師

任務	Description	所需的技能
	4. 針對主要 DNS 尾碼轉移，選擇未設定。	
變更 Windows Update 設定。	1. 在本機群組政策編輯器中，導覽至電腦組態、管理範本、Windows 元件、Windows Update。 2. 針對指定內部網路 Microsoft 更新服務位置，選擇未設定。 3. 針對設定自動更新，選擇未設定。 4. 針對自動更新偵測頻率，選擇未設定。 5. 關閉本機群組政策編輯器。	遷移工程師
啟用防火牆。	1. 在來源 EC2 執行個體上，以管理員身分開啟命令提示字元，輸入 `services.msc` ，然後按 Enter 鍵。 2. 在 Windows Services 中，啟用防火牆。 3. 關閉 Windows 服務。	遷移工程師

準備 AMS WIGS 的執行個體

任務	Description	所需的技能
清除並準備執行個體。	1. 使用堡壘主機和本機登入資料，在預備子網路中建立 EC2 執行個體的遠端桌面通訊協定 (RDP) 連線。	遷移工程師

任務	Description	所需的技能
	2. 移除 AMS 中不需要的所有舊版軟體、防毒軟體和備份解決方案。	

修復 sppnp.dll 檔案。	1. 前往 `C:\Windows\System32\sppnp.dll` 。
	2. 重新命名`sppnp.dll` 為 `sppnp_old.dll` 。
	3. 使用 PowerShell 和管理員登入資料，輸入下列命令：
	`dism /online /cleanup-image /restorehealth`
	`sfc /scannnow`
	4. 重新啟動 EC2 Windows 執行個體。
		遷移工程師

執行預先 WIG 驗證指令碼。	1. 從遷移工作負載下載 Windows WIGS 擷取前驗證 zip 檔案 (`windows-prewings-valication.zip`)：AMS 文件中的 Windows 擷取前驗證。https://docs.aws.amazon.com/managedservices/latest/appguide/ex-migrate-instance-win-validation.html 2. 執行 Windows 預先 WIG 驗證指令碼並驗證結果。 3. 如果驗證失敗，請修正問題，並重新執行驗證指令碼，直到驗證成功為止。	遷移工程師

任務	Description	所需的技能
建立故障安全 AMI。	在預先 WIG 驗證通過之後，建立擷取前 AMI，如下所示： 1. 選擇部署、進階堆疊元件、AMI、建立。 2. 在建立期間，新增標籤 `Key=Name, Value=APPLICATION-ID_IngestReady` 。 3. 請等待 AMI 建立後再繼續。 如需詳細資訊，請參閱 AMS 文件中的 AMI | Create。	遷移工程師

擷取和驗證執行個體

任務	Description	所需的技能
提交 RFC 以建立工作負載擷取堆疊。	提交變更請求 (RFC) 以啟動 AMS WIGS。如需說明，請參閱 AMS 文件中的工作負載擷取堆疊：建立。這會啟動工作負載擷取，並安裝 AMS 所需的所有軟體，包括備份工具、Amazon EC2 管理軟體和防毒軟體。	遷移工程師
驗證成功遷移。	工作負載擷取完成後，您可以看到 AMS 受管執行個體和 AMS 擷取的 AMI。 1. 使用網域登入資料登入 AMS 受管執行個體。	遷移工程師

任務	Description	所需的技能
	2. 驗證加入的網域，如下所示： a. 在 Windows Explorer 中，用滑鼠右鍵按一下此 PC，然後選擇屬性。 b. 在裝置規格區段中，確認網域出現在完整裝置名稱中。 3. 驗證來源和目標磁碟機。	

在目標 AMS 帳戶中啟動執行個體

任務	Description	所需的技能
提交 RFC 以建立 EC2 堆疊。	1. 使用 Windows 執行個體的 AMS 擷取 AMI，根據 AMS 文件中建立 EC2 堆疊執行個體的指示，為 EC2 堆疊準備 RFC。 EC2 在 EC2 堆疊 RFC 中，提供所有參數，包括伺服器名稱、標籤、目標 VPC、目標子網路、執行個體類型、目標安全群組、擷取 AMI 和角色。 2. 提交 EC2 堆疊的 RFC，然後等待執行個體成功建立。	遷移工程師

相關資源

AWS 方案指引

- 在 Windows 上自動化 AWS Managed Services 的工作負載前擷取活動
- 使用 Python 在 AMS 中自動建立 RFC

AMS 文件

- AMS 工作負載擷取
- 遷移如何變更您的資源
- 遷移工作負載：標準程序

行銷資源

- AWS Managed Services
- AWS Managed Services FAQs
- AWS Managed Services 資源
- AWS Managed Services功能

將 Couchbase Server 資料庫遷移至 Amazon EC2

Subhani Shaik，Amazon Web Services

總結

此模式說明如何將 Couchbase Server 從內部部署環境遷移至 Amazon Elastic Compute Cloud (Amazon EC2) AWS。

Couchbase Server 是分散式 NoSQL (JSON 文件） 資料庫，可提供關聯式資料庫功能。將 Couchbase Server 資料庫遷移到 AWS 可以提供更高的可擴展性、改善的效能、成本效益、增強的安全性、簡化的管理和全域覆蓋，這可以使需要高可用性和低延遲資料存取的應用程式受益。您也可以透過 AWS 受管服務存取進階功能。

上的 Couchbase Server AWS 提供下列主要功能：

- 記憶體優先架構
- 高可用性、災難復原和負載平衡
- 多主機、多區域部署，提供最佳效能

如需主要優點的詳細資訊，請參閱其他資訊一節和 Couchbase 網站。

先決條件和限制

先決條件

- AWS 帳戶 使用虛擬私有雲端 (VPC)、兩個可用區域、私有子網路和安全群組的作用中 。如需說明，請參閱《Amazon Virtual Private Cloud (Amazon VPC) 文件》中的建立 VPC。
- 在來源和目標環境之間啟用連線。如需 Couchbase Server 使用的 TCX 連接埠相關資訊，請參閱 Couchbase 文件。

Architecture

下圖顯示將 Couchbase Server 遷移至 的高階架構 AWS。

從內部部署 Couchbase 叢集，資料會使用 透過客戶閘道移動AWS Direct Connect。資料會通過路由器和 Direct Connect 路由，並透過 AWS Virtual Private Network (Site-to-Site VPN) 閘道到達

VPC。VPC 包含執行 Couchbase Server 的 EC2 執行個體。 AWS 基礎設施也包含用於存取控制的 AWS Identity and Access Management (IAM)、用於資料加密的 AWS Key Management Service (AWS KMS)、用於區塊儲存的 Amazon Elastic Block Store (Amazon EBS)，以及用於資料儲存的 Amazon Simple Storage Service (Amazon S3)。

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- AWS Direct Connect 會透過標準乙太網路光纖纜線將您的內部網路連結至某個 Direct Connect 位置。透過此連線，您可以在網路路徑中略過網際網路服務提供者 AWS 服務 時，直接建立虛擬介面至公有。

最佳實務

- 在不同的操作平台上安裝和設定 Couchbase
- 在 上部署 Couchbase Server 的最佳實務 AWS
- 建立 Couchbase 叢集
- Couchbase 應用程式的效能最佳實務
- Couchbase Server 的安全最佳實務
- Couchbase Server 資料庫的儲存最佳實務

史詩

部署適用於 Couchbase Server 的 Amazon EC2 執行個體

任務	Description	所需的技能
開啟 Amazon EC2 主控台。	登入 AWS 管理主控台 並開啟 Amazon EC2 主控台。	DevOps 工程師、Couchbase 管理員
部署 Amazon EC2 執行個體。	啟動符合內部部署 Couchbase Server 組態的 EC2 執行個體。如需如何部署 EC2 執行個體的詳細資訊，請參閱	DevOps 工程師、Couchbase 管理員

任務	Description	所需的技能
	Amazon EC2 文件中的啟動 Amazon EC2 執行個體。Amazon EC2	

在 Amazon EC2 上安裝和設定 Couchbase Server

任務	Description	所需的技能
安裝 Couchbase 叢集。	在 Amazon EC2 上安裝 Couchbase Server 之前，請檢閱 Couchbase Server 部署準則。 若要安裝 Couchbase Server，請參閱 Couchbase Server 文件	Couchbase 管理員
設定叢集。	若要設定叢集，請參閱 Couchbase 文件中的叢集組態選項。	Couchbase 管理員

新增節點並重新平衡 Couchbase 叢集

任務	Description	所需的技能
新增 EC2 執行個體的節點。	將已安裝 Couchbase 的新部署 EC2 執行個體新增至現有的現場部署叢集。如需說明，請參閱 Couchbase Server 文件中的新增節點並重新平衡。	Couchbase 管理員
重新平衡叢集。	重新平衡程序可讓新增的節點與 EC2 執行個體成為 Couchbase 叢集的作用中成員。如需說明，請參閱	Couchbase 管理員

任務	Description	所需的技能
	Couchbase Server 文件中的新增節點並重新平衡	

重新設定連線

任務	Description	所需的技能
移除內部部署節點並重新平衡。	您現在可以從叢集中移除內部部署節點。移除節點後，請遵循重新平衡程序，在叢集中的可用節點之間重新分配資料、索引、事件處理和查詢處理。如需說明，請參閱 Couchbase Server 文件中的移除節點並重新平衡。	Couchbase 管理員
更新連線參數。	更新應用程式的連線參數以使用新的 Amazon EC2 IP 地址，讓您的應用程式可以連線到新的節點。	Couchbase 應用程式開發人員

相關資源

- Couchbase 伺服器服務
- 使用 部署 Couchbase 伺服器 AWS Marketplace
- 連線至 Couchbase Server
- 管理儲存貯體
- 跨資料中心複寫 (XDCR)
- Couchbase Inc. 授權合約

其他資訊

主要優點

遷移 Couchbase 資料庫 AWS 以提供下列優點：

延展性。 您可以根據需求向上或向下擴展 Couchbase 叢集，而無需管理實體硬體，因此您可以輕鬆容納波動的資料磁碟區和應用程式用量。 AWS 提供：

- 垂直和水平擴展選項
- 全域部署功能
- 跨 的負載平衡 AWS 區域
- 資料庫擴展解決方案
- 內容交付最佳化

效能最佳化。 AWS 提供高效能網路基礎設施和最佳化執行個體類型，以確保 Couchbase 資料庫的快速資料存取和低延遲。

- 高效能運算 (HPC) 選項
- 透過 Amazon CloudFront 提供全域內容
- 多個儲存選項
- 進階資料庫服務，包括 Amazon Relational Database Service (Amazon RDS) 和 Amazon DynamoDB
- 使用 的低延遲連線 Direct Connect

成本最佳化。選取適當的執行個體類型和組態，以根據您的工作負載平衡效能和成本。只需為您使用的資源付費。這可以透過消除管理內部部署硬體和利用規模 AWS 雲端 經濟的需求來降低您的營運成本。

- 預留執行個體可協助您提前規劃，並在使用 Couchbase 時大幅降低成本 AWS。
- 自動擴展可防止過度佈建，並協助您最佳化使用率和成本效率。

增強安全性。受益於 上的強大安全功能 AWS，例如資料加密、存取控制和安全群組，以協助保護您存放在 Couchbase 中的敏感資料。其他優點：

- AWS 共同責任模型可清楚區分雲端安全性 （責任）AWS 和雲端安全性 （客戶責任）。
- AWS 合規支援主要安全標準。
- AWS 提供進階加密選項。
- AWS Identity and Access Management (IAM) 可協助您管理 資源的安全存取。

簡化的 management. AWS provides 受管服務適用於 Couchbase，因此您可以專注於應用程式開發，而不是管理基礎基礎設施。

全球觸角。您可以將 Couchbase 叢集部署到多個 AWS 區域 ，為全球使用者實現低延遲。您可以將資料庫完全部署在雲端或混合環境中。您可以使用內建的企業級安全性，以及從邊緣到雲端的資料快速、高效的雙向同步來保護資料。同時，您可以使用一致的程式設計模型來簡化開發，以建置 Web 和行動應用程式。

業務持續性：

- 資料備份和復原。如果發生問題，您可以使用 AWS Backup來確保資料彈性和輕鬆復原。如需災難復原選項，請參閱 AWS Well-Architected Framework 文件。
- Couchbase 多區域部署：若要在多區域 AWS 環境中部署 Couchbase 資料庫，您可以在 中訂閱 Couchbase ServerAWS Marketplace，使用 AWS CloudFormation範本在每個區域中建立個別的 Couchbase 叢集，然後設定跨區域複寫以同步跨區域的資料。此組態可確保跨多個區域的高可用性和地理備援。如需詳細資訊，請參閱 Couchbase 文件中的使用 部署 Couchbase 伺服器 AWS Marketplace。

基礎設施敏捷性：

- 快速資源佈建和取消佈建
- 全球基礎設施觸角
- 根據需求自動擴展
- 用於一致部署的基礎設施即程式碼 (IaC)
- 針對不同工作負載進行最佳化的多個執行個體類型

創新啟用：

- 存取最新技術，包括 AI/ML、IoT 和分析
- 受管服務，可降低營運開銷
- 現代應用程式開發實務
- 無伺服器運算選項

卓越營運：

- 集中式監控和記錄

- 自動化資源管理
- 預測性維護功能
- 增強對資源用量的可見性
- 簡化的部署程序

現代化機會：

- Microservices 架構
- DevOps 實務實作
- 雲端原生應用程式開發
- 傳統應用程式現代化

競爭優勢：

- 更快的上市時間
- 改善客戶體驗
- 資料驅動型決策
- 增強的商業智慧

使用日誌運送將 LUW 的 Db2 遷移至 Amazon EC2，以減少中斷時間

Feng Cai、Ambarish Satarkar 和 Saurabh Sharma，Amazon Web Services

總結

當客戶將 IBM Db2 for LUW (Linux、UNIX 和 Windows) 工作負載遷移至 Amazon Web Services (AWS) 時，使用 Amazon Elastic Compute Cloud (Amazon EC2) 搭配自帶授權 (BYOL) 模型是最快的方式。不過，將大量資料從內部部署 Db2 遷移到 AWS 可能是一項挑戰，特別是當中斷時段很短時。許多客戶嘗試將中斷時段設定為少於 30 分鐘，這對於資料庫本身來說幾乎沒有時間。

此模式涵蓋如何使用交易日誌運送，以短暫中斷時段完成 Db2 遷移。此方法適用於小端 Linux 平台上的 Db2。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在 ECDb2EC2 執行個體，符合內部部署檔案系統配置
- EC2 執行個體可存取的 Amazon Simple Storage Service (Amazon S3) 儲存貯體
- AWS Identity and Access Management (IAM) 政策和角色，用於對 Amazon S3 進行程式設計呼叫
- Amazon EC2 和內部部署伺服器上的同步時區和系統時鐘
- 透過 AWS Site-to-Site VPN 或 AWS Direct Connect 連線至 AWS 的內部部署網路

限制

- Db2 內部部署執行個體和 Amazon EC2 必須位於相同的平台系列。
- 必須記錄 Db2 內部部署工作負載。若要封鎖任何未記錄的交易，請在資料庫組態`blocknonlogged=yes`中設定 。

產品版本

- Db2 for LUW 11.5.9 版及更新版本

Architecture

來源技術堆疊

- Linux x86_64 上的 Db2

目標技術堆疊

- Amazon EBS
- Amazon EC2
- AWS Identity and Access Management (IAM)
- Amazon S3
- AWS Site-to-Site或 Direct Connect

目標架構

下圖顯示一個在內部部署執行的 Db2 執行個體，具有與 Amazon EC2 上的 Db2 的虛擬私有網路 (VPN) 連線。虛線代表資料中心和 AWS 雲端之間的 VPN 通道。

工具

AWS 服務

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- AWS Direct Connect 透過標準乙太網路光纖纜線，將您的內部網路連結至 Direct Connect 位置。透過此連線，您可以直接建立與公有 AWS 服務的虛擬介面，同時略過網路路徑中的網際網路服務供應商。
- Amazon Elastic Block Store (Amazon EBS) 提供區塊層級儲存磁碟區，可與 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體搭配使用。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Site-to-Site VPN 可協助您在 AWS 上啟動的執行個體與您自己的遠端網路之間傳遞流量。

其他工具

- db2cli 是 Db2 互動式 CLI 命令。

最佳實務

- 在目標資料庫上，使用 Amazon S3 的閘道端點來存取 Amazon S3 中的資料庫備份映像和日誌檔案。
- 在來源資料庫上，使用適用於 Amazon S3 的 AWS PrivateLink 將資料庫備份映像和日誌檔案傳送至 Amazon S3。

史詩

設定環境變數

任務	Description	所需的技能
設定環境變數。	此模式使用以下名稱： • 執行個體名稱：`db2inst1` • 資料庫名稱：`SAMPLE` 您可以變更它們以符合您的環境。	DBA

設定內部部署 Db2 伺服器

任務	Description	所需的技能

設定 AWS CLI。	若要下載並安裝最新版本的 AWS CLI，請執行下列命令：
	`$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"`
	`unzip awscliv2.zip`
		Linux 管理員

任務	Description	所需的技能
	`sudo ./aws/install`	

設定 Db2 封存日誌的本機目的地。	若要讓 Amazon EC2 上的目標資料庫與內部部署來源資料庫保持同步，需要從來源擷取最新的交易日誌。
	
	在此設定中，`/db2logs`會在來源`LOGARCHMETH2` 上由 設定為預備區域。此目錄中的封存日誌會同步至 Amazon S3，並由 Amazon EC2 上的 Db2 存取。模式使用 ，`LOGARCHMETH2` 因為 `LOGARCHMETH1` 可能已設定為使用 AWS CLI 命令無法存取的第三方廠商工具。若要擷取日誌，請執行下列命令：
	
	`db2 connect to sample`
	`db2 update db cfg for SAMPLE using LOGARCHMETH2 disk:/db2logs`
		DBA

執行線上資料庫備份。	執行線上資料庫備份，並將其儲存至本機備份檔案系統： `db2 backup db sample online to /backup`	DBA

設定 S3 儲存貯體和 IAM 政策

任務	Description	所需的技能
建立 S3 儲存貯體。	為現場部署伺服器建立 S3 儲存貯體，以在 AWS 上將備份 Db2 映像和日誌檔案傳送至 。Amazon EC2 也會存取儲存貯體： `aws s3api create-bucket --bucket logshipmig-db2 --region us-east-1`	AWS 系統管理員
建立 IAM 政策。	`db2bucket.json` 檔案包含存取 Amazon S3 儲存貯體的 IAM 政策： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "kms:GenerateDataKey", "kms:Decrypt", "s3:PutObject", "s3:GetObject", "s3:AbortMultipartUpload", "s3:ListBucket",`	AWS 管理員、AWS 系統管理員

任務	Description	所需的技能

	<pre>"s3:DeleteObject",
	
	"s3:GetObjectVersion",
	
	"s3:ListMultipartUploadParts"
	],
	"Resource":
	[
	
	"arn:aws:s3:::logshipmig-db2/*",
	
	"arn:aws:s3:::logshipmig-db2"
	]
	}
	]
	}</pre>
	若要建立政策，請使用下列 AWS CLI 命令：
	
	<pre>aws iam create-policy \
	--policy-name
	db2s3policy \
	--policy-document
	file://db2bucket.json</pre>
	JSON 輸出會顯示政策的 Amazon Resource Name (ARN)，其中 `aws_account_id` 代表您的帳戶 ID：	

任務	Description	所需的技能
	`"Arn": "arn:aws:iam::aws_account_id:policy/db2s3policy"`	
將 IAM 政策連接至 EC2 執行個體所使用的 IAM 角色。	在大多數 AWS 環境中，執行中的 EC2 執行個體具有由系統管理員設定的 IAM 角色。如果未設定 IAM 角色，請建立角色，然後選擇 EC2 主控台上的修改 IAM 角色，將角色與託管 Db2 資料庫的 EC2 執行個體建立關聯。使用政策 ARN 將 IAM 政策連接至 IAM 角色： `aws iam attach-role-policy \ --policy-arn "arn:aws:iam::aws_account_id:policy/db2s3policy" \ --role-name db2s3role` 連接政策後，與 IAM 角色相關聯的任何 EC2 執行個體都可以存取 S3 儲存貯體。	AWS 管理員、AWS 系統管理員

將來源資料庫備份映像和日誌檔案傳送至 Amazon S3

任務	Description	所需的技能
在內部部署 Db2 伺服器上設定 AWS CLI。	使用 設定 AWS CLI，`Access Key ID`並在先前步驟中`Secret Access Key`產生：	AWS 管理員、AWS 系統管理員

任務	Description	所需的技能

	`$ aws configure`
	`AWS Access Key ID`
	` [None]: **********`
	`AWS Secret Access Key`
	` [None]: *********`
	`**********`
	`Default region name`
	` [None]: us-east-1`
	`Default output format`
	` [None]: json`	

將備份映像傳送至 Amazon S3。	稍早，線上資料庫備份已儲存至/backup本機目錄。若要將備份映像傳送至 S3 儲存貯體，請執行下列命令：
	`aws s3 sync /backup`
	` s3://logshipmig-db2/`
	`SAMPLE_backup`
		AWS 管理員、遷移工程師

將 Db2 封存日誌傳送至 Amazon S3。	將內部部署 Db2 封存日誌與 Amazon EC2 上目標 Db2 執行個體可存取的 S3 儲存貯體同步：
	`aws s3 sync /db2logs`
	` s3://logshipmig-db2/`
	`SAMPLE_LOG`
	使用 cron 或其他排程工具定期執行此命令。頻率取決於來源資料庫封存交易日誌檔案的頻率。
		AWS 管理員、遷移工程師

將 Amazon EC2 上的 Db2 連線至 Amazon S3 並啟動資料庫同步 Amazon EC2

任務	Description	所需的技能

建立 PKCS12 金鑰存放區。	Db2 使用公有金鑰密碼編譯標準 (PKCS) 加密金鑰存放區來保護 AWS 存取金鑰的安全。建立金鑰存放區並設定來源 Db2 執行個體以使用它：
	`gsk8capicmd_64 -keydb -create -db "/home/db2inst1/.keystore/db2s3.p12" -pw "<password>" -type pkcs12 -stash`
	`db2 "update dbm cfg using keystore_location /home/db2inst1/.keystore/db2s3.p12 keystore_type pkcs12"`
		DBA

建立 Db2 儲存體存取別名。	若要建立儲存存取別名，請使用下列指令碼語法：
	`db2 "catalog storage access alias <alias_name> vendor S3 server <S3 endpoint> container '<bucket_name>'"`
	例如，您的指令碼可能如下所示：
	`db2 "catalog storage access alias DB2AWSS3 vendor S3 server`
		DBA

任務	Description	所需的技能
	`s3.us-east-1.amazonaws.com container 'logshipmig-db2'"`	

任務	Description	所需的技能

設定預備區域。	根據預設，Db2 會使用 DB2_OBJECT_STORAGE_LOCAL_STAGING_PATH 做為臨時區域，在 Amazon S3 之間上傳和下載檔案。預設路徑位於執行個體主目錄sqllib/tmp/RemoteStorage.xxxx 下，並xxxx參照 Db2 分割區編號。請注意，預備區域必須有足夠的容量來存放備份映像和日誌檔案。您可以使用登錄檔，將暫存區域指向不同的目錄。
	
	我們也建議使用 DB2_ENABLE_COS_SDK=ON 、DB2_OBJECT_STORAGE_SETTINGS=EnableStreamingRestore 和程式awssdk庫的連結，略過資料庫備份和還原的 Amazon S3 預備區域：
	
	`#By root:`
	`cp -rp /home/db2inst1/sqllib/lib64/awssdk/RHEL/7.6/* /home/db2inst1/sqllib/lib64/`
	
	`#By db2 instance owner:`
	`db2set DB2_OBJECT_STORAGE_LOCAL_STAGING_PATH=/db2stage`
	`db2set DB2_ENABLE_COS_SDK=ON`
		DBA

任務	Description	所需的技能

	`Db2set DB2_OBJECT_STORAGE_SETTINGS=EnableStreamingRestore`
	`db2stop`
	`db2start`	

從備份映像還原資料庫。	從 S3 儲存貯體中的備份映像還原 Amazon EC2 上的目標資料庫： `db2 restore db sample from DB2REMOTE://DB2AWSS3/logshipmig-db2/SAMPLE_backup replace existing`	DBA

任務	Description	所需的技能

向前滾動資料庫。	還原完成後，目標資料庫將進入向前滾動擱置狀態。設定 LOGARCHMETH1 和 ，LOGARCHMETH2 以便 Db2 知道在何處取得交易日誌檔案：
	`db2 update db cfg for SAMPLE using LOGARCHMETH1 'DB2REMOTE://DB2AWSS3//SAMPLE_LOGS/'`
	`db2 update db cfg for SAMPLE using LOGARCHMETH2 OFF`
	開始資料庫向前滾動：
	`db2 ROLLFORWARD DATABASE sample to END OF LOGS`
	此命令會處理已傳輸至 S3 儲存貯體的所有日誌檔案。根據現場部署 Db2 伺服器上s3 sync命令的頻率定期執行。例如，如果 每小時s3 sync執行一次，且同步所有日誌檔案需要 10 分鐘，請將 命令設定為每小時 10 分鐘後執行一次。
		DBA

在切換時段將 Db2 帶入 Amazon EC2

任務	Description	所需的技能

讓目標資料庫上線。	在切換時段期間，執行下列其中一項操作：
	• 在 中放置內部部署資料庫ADMIN MODE，然後執行s3 sync命令強制封存最後一個交易日誌。
	• 關閉資料庫。
	
	將最後一個交易日誌同步至 Amazon S3 後，請執行ROLLFORWARD 命令最後一次：
	
	`db2 rollforward DB sample to END OF LOGS`
	`db2 rollforward DB sample complete`
	
	`Rollforward Status`
	`....`
	`Rollforward status = not pending`
	`....`
	`DB20000I The ROLLFORWARD command completed successfully.`
	
	`db2 activate db sample`
	`DB20000I The ACTIVATE DATABASE command completed successfully.`
		DBA

任務	Description	所需的技能
	將目標資料庫上線，並將應用程式連線指向 Amazon EC2 上的 Db2。 Amazon EC2	

疑難排解

問題	解決方案
如果多個資料庫在不同的主機 (DEV、QA、PROD) 上具有相同的執行個體名稱和資料庫名稱，則備份和日誌可能會移至相同的子目錄。	針對 DEV、QA 和 PROD 使用不同的 S3 儲存貯體，並將主機名稱新增為子目錄字首，以避免混淆。
如果同一個位置有多個備份映像，當您還原時，會收到下列錯誤： `SQL2522N More than one backup file matches the time stamp value provided for the backed up database image.`	在 `restore`命令中，新增備份的時間戳記： `db2 restore db sample from DB2REMOTE://DB2AWSS3/logshipmig-db2/SAMPLE_backup taken at 20230628164042 replace existing`

相關資源

- 不同作業系統和硬體平台之間的 Db2 備份和還原操作
- 設定 Db2 STORAGE ACCESS ALIAS 和 DB2REMOTE
- Db2 ROLLFORWARD 命令
- Db2 次要日誌封存方法

將 LUW 的 Db2 遷移至具有高可用性災難復原的 Amazon EC2

Feng Cai、Aruna Gangireddy 和 Venkatesan Govindan，Amazon Web Services

摘要

當客戶將 IBM Db2 LUW (Linux、UNIX 和 Windows) 工作負載遷移至 Amazon Web Services (AWS) 時，使用 Amazon Elastic Compute Cloud (Amazon EC2) 搭配自帶授權 (BYOL) 模型是最快的方式。不過，將大量資料從內部部署 Db2 遷移到 AWS 可能是一項挑戰，特別是當中斷時段很短時。許多客戶嘗試將中斷時段設定為少於 30 分鐘，這對於資料庫本身來說幾乎沒有時間。

此模式涵蓋如何使用 Db2 高可用性災難復原 (HADR)，以短暫的中斷時段完成 Db2 遷移。此方法適用於在小端 Linux 平台上且未使用資料分割功能 (DPF) 的 Db2 資料庫。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在符合內部部署檔案系統配置的 Amazon EC2 執行個體上執行的 Db2 執行個體 Amazon EC2
- EC2 執行個體可存取的 Amazon Simple Storage Service (Amazon S3) 儲存貯體
- AWS Identity and Access Management (IAM) 政策和角色，用於對 Amazon S3 進行程式設計呼叫
- Amazon EC2 和內部部署伺服器上的同步時區和系統時鐘
- 透過 AWS Site-to-Site VPN 或 AWS Direct Connect 連線至 AWS 的內部部署網路
- HADR 連接埠上的內部部署伺服器與 Amazon EC2 之間的通訊

限制

- Db2 內部部署執行個體和 Amazon EC2 必須位於相同的平台系列。
- 分割的資料庫環境中不支援 HADR。
- HADR 不支援對資料庫日誌檔案使用原始 I/O （直接磁碟存取）。
- HADR 不支援無限記錄。
- `LOGINDEXBUILD` 必須設定為 `YES`，這會增加重建索引的日誌用量。
- 必須記錄 Db2 內部部署工作負載。在資料庫組態`blocknonlogged=yes`中設定 ，以封鎖任何未記錄的交易。

產品版本

- Db2 for LUW 11.5.9 版及更新版本

Architecture

來源技術堆疊

- Linux x86_64 上的 Db2

目標技術堆疊

- Amazon EC2
- AWS Identity and Access Management (IAM)
- Amazon S3
- AWS Site-to-Site VPN

目標架構

在下圖中，現場部署的 Db2 正在 上執行，`db2-server1`做為主要節點。它有兩個 HADR 待命目標。一個待命目標在內部部署，並且是選用的。另一個待命目標 `db2-ec2`位於 Amazon EC2。在將資料庫切換到 AWS 之後， 會`db2-ec2`成為主要資料庫。

1. 日誌會從主要現場部署資料庫串流到待命現場部署資料庫。
2. 使用 Db2 HADR，日誌會透過 Site-to-Site VPN 從主要現場部署資料庫串流至 Amazon EC2 上的 Db2。 Amazon EC2
3. Db2 備份和封存日誌會從主要現場部署資料庫傳送至 AWS 上的 S3 儲存貯體。

工具

AWS 服務

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- AWS Direct Connect 透過標準乙太網路光纖纜線，將您的內部網路連結至 Direct Connect 位置。透過此連線，您可以直接建立與公有 AWS 服務的虛擬介面，同時略過網路路徑中的網際網路服務供應商。

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Site-to-Site VPN 可協助您在 AWS 上啟動的執行個體與您自己的遠端網路之間傳遞流量。

其他工具

- db2cli 是 Db2 互動式 CLI 命令。

最佳實務

- 在目標資料庫上，使用 Amazon S3 的閘道端點來存取 Amazon S3 中的資料庫備份映像和日誌檔案。
- 在來源資料庫上，使用適用於 Amazon S3 的 AWS PrivateLink 將資料庫備份映像和日誌檔案傳送至 Amazon S3。

史詩

設定環境變數

任務	Description	所需的技能
設定環境變數。	此模式使用以下名稱和連接埠： 1. Db2 內部部署主機名稱：`db2-server1` 2. HADR 待命主機名稱：`db2-server2` （如果 HADR 目前正在內部部署執行） 3. Amazon EC2 主機名稱：`db2-ec2`	DBA

任務	Description	所需的技能
	4. 執行個體名稱：db2inst1 5. 資料庫名稱：SAMPLE 6. HADR 連接埠： • db2-server1: 50010 • db2-server2: 50011 • db2-ec2: 50012 您可以變更它們以符合您的環境。	

設定內部部署 Db2 伺服器

任務	Description	所需的技能

設定 AWS CLI。	若要下載並安裝最新版本的 AWS CLI，請執行下列命令：
	
	`$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"`
	`unzip awscliv2.zip`
	`sudo ./aws/install`
		Linux 管理員

設定 Db2 封存日誌的本機目的地。	大量更新批次任務和網路變慢等條件可能會導致 HADR 待命伺服器延遲。若要趕上進度，待命伺服器需要來自主要伺服器的交易日誌。請求日誌的位置順序如下： • 主要伺服器上的作用中日誌目錄	DBA

任務	Description	所需的技能

	• 待命伺服器上的 LOGARCHMETH1 或 LOGARCHMETH2 位置
	• 主伺服器上的 LOGARCHMETH1 或 LOGARCHMETH2 位置
	
	在此設定中，/db2logs會在來源LOGARCHMETH2 上由 設定為預備區域。此目錄中的封存日誌會同步至 Amazon S3，並由 Amazon EC2 上的 Db2 存取。模式使用 ，LOGARCHMETH2 因為 LOGARCHMETH1 可能已設定為使用 AWS CLI 命令無法存取的第三方廠商工具：
	
	`db2 connect to sample`
	`db2 update db cfg for SAMPLE using LOGARCHMETH2 disk:/db2logs`	

執行線上資料庫備份。	執行線上資料庫備份，並將其儲存至本機備份檔案系統： `db2 backup db sample online to /backup`	DBA

設定 S3 儲存貯體和 IAM 政策

任務	Description	所需的技能
建立 S3 儲存貯體。	為現場部署伺服器建立 S3 儲存貯體，以在 AWS 上將備份 Db2 映像和日誌檔案傳送至 。儲存貯體將由 Amazon EC2 存取： `aws s3api create-bucket --bucket hadrmig-db2 --region us-east-1`	AWS 管理員
建立 IAM 政策。	`db2bucket.json` 檔案包含用於存取 S3 儲存貯體的 IAM 政策： `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "kms:GenerateDataKey", "kms:Decrypt", "s3:PutObject", "s3:GetObject", "s3:AbortMultipartUpload", "s3:ListBucket",`	AWS 管理員、AWS 系統管理員

任務	Description	所需的技能

	<pre>"s3:DeleteObject",
	
	"s3:GetObjectVersion",
	
	"s3:ListMultipartUploadParts"
	],
	"Resource":
	[
	
	"arn:aws:s3:::hadrmig-db2/*",
	
	"arn:aws:s3:::hadrmig-db2"
	]
	}
	]
	}</pre>
	若要建立政策，請使用下列 AWS CLI 命令：
	<pre>aws iam create-policy \
	--policy-name
	db2s3hapolicy \
	--policy-document
	file://db2bucket.json</pre>
	JSON 輸出會顯示政策的 Amazon Resource Name (ARN)，其中 `aws_account_id` 代表您的帳戶 ID：
	<pre>"Arn": "arn:aws:iam::aws_account_i</pre>	

任務	Description	所需的技能
	`d:policy/db2s3hapo licy"`	
將 IAM 政策連接至 IAM 角色。	通常，執行 Db2 的 EC2 執行個體會有系統管理員指派的 IAM 角色。 Db2 如果未指派 IAM 角色，您可以在 Amazon EC2 主控台上選擇修改 IAM 角色。 將 IAM 政策連接至與 EC2 執行個體相關聯的 IAM 角色。連接政策後，EC2 執行個體可以存取 S3 儲存貯體： `aws iam attach-role-policy --policy-arn "arn:aws:iam::aws_account_id:policy/db2s3hapolicy" --role-name db2s3harole`	

將來源資料庫備份映像和日誌檔案傳送至 Amazon S3

任務	Description	所需的技能

在內部部署 Db2 伺服器上設定 AWS CLI。	使用`Secret Access Key`您先前產生的 `Access Key ID`和 設定 AWS CLI：
	
	`$ aws configure`
	`AWS Access Key ID [None]: **********`
	`AWS Secret Access Key [None]: ********** **********`
		AWS 管理員、AWS 系統管理員

任務	Description	所需的技能
	`Default region name [None]: us-east-1` `Default output format [None]: json`	
將備份映像傳送至 Amazon S3。	稍早，線上資料庫備份已儲存至/backup本機目錄。若要將備份映像傳送至 S3 儲存貯體，請執行下列命令： `aws s3 sync /backup s3://hadrmig-db2/SAMPLE_backup`	AWS 管理員、AWS 系統管理員
將 Db2 封存日誌傳送至 Amazon S3。	將內部部署 Db2 封存日誌與 Amazon EC2 上的目標 Db2 執行個體可存取的 Amazon S3 儲存貯體同步： Db2 Amazon EC2 `aws s3 sync /db2logs s3://hadrmig-db2/SAMPLE_LOGS` 使用 cron 或其他排程工具定期執行此命令。頻率取決於來源資料庫封存交易日誌檔案的頻率。	

將 Amazon EC2 上的 Db2 連線至 Amazon S3，並啟動初始資料庫同步 Amazon EC2

任務	Description	所需的技能
建立 PKCS12 金鑰存放區。	Db2 使用公有金鑰密碼編譯標準 (PKCS) 加密金鑰存放區來	DBA

任務	Description	所需的技能

	保護 AWS 存取金鑰的安全。建立金鑰存放區，並設定來源 Db2 以使用它：
	
	`gsk8capicmd_64 -keydb -create -db "/home/db2inst1/.keystore/db2s3.p12" -pw "<password>" -type pkcs12 -stash`
	
	`db2 "update dbm cfg using keystore_location /home/db2inst1/.keystore/db2s3.p12 keystore_type pkcs12"`	

任務	Description	所需的技能

建立 Db2 儲存體存取別名。	Db2 使用儲存存取別名，透過 INGEST、BACKUP DATABASE、 LOAD或 RESTORE DATABASE命令直接存取 Amazon S3。
	
	由於您已將 IAM 角色指派給 EC2 執行個體，USERPASSWORD因此不需要：
	
	`db2 "catalog storage access alias <alias_name> vendor S3 server <S3 endpoint> container '<bucket_name>'"`
	
	例如，您的指令碼可能如下所示：
	
	`db2 "catalog storage access alias DB2AWSS3 vendor S3 server s3.us-east-1.amazonaws.com container 'hadrmig-db2'"`
		DBA

任務	Description	所需的技能
設定預備區域。	建議使用 DB2_ENABLE_COS_SDK=ON 、DB2_OBJECT_STORAGE_SETTINGS=EnableStreamingRestore 和程式awssdk庫的連結，略過資料庫備份和還原的 Amazon S3 預備區域： #By root: cp -rp /home/db2inst1/sqllib/lib64/awssdk/RHEL/7.6/* /home/db2inst1/sqllib/lib64/ #By db2 instance owner: db2set DB2_OBJECT_STORAGE_LOCAL_STAGING_PATH=/db2stage db2set DB2_ENABLE_COS_SDK=ON db2set DB2_OBJECT_STORAGE_LOCAL_STAGING_PATH=/db2stage db2stop db2start	DBA

任務	Description	所需的技能

從備份映像還原資料庫。	從 S3 儲存貯體中的備份映像還原 Amazon EC2 上的目標資料庫：
	
	`db2 create db sample on /data1`
	`db2 restore db sample from DB2REMOTE://DB2AWSS3/hadrmig-db2/SAMPLE_backup replace existing`
		DBA

在內部部署中設定沒有 HADR 的 HADR

任務	Description	所需的技能
將內部部署 Db2 伺服器設定為主要伺服器。	將 HADR on db2-server1 （內部部署來源）的資料庫組態設定更新為主要伺服器。HADR_SYNCMODE 設定為 SUPERASYNC 模式，其交易回應時間最短： `db2 update db cfg for sample using HADR_LOCAL_HOST db2-server1 HADR_LOCAL_SVC 50010 HADR_REMOTE_HOST db2-ec2 HADR_REMOTE_SVC 50012 HADR_REMOTE_INST db2inst1 HADR_SYNCMODE SUPERASYNC DB20000I The UPDATE DATABASE CONFIGURATION command`	DBA

任務	Description	所需的技能
	completed successfu lly 現場部署資料中心和 AWS 之間預計會發生一些網路延遲。（您可以根據網路可靠性設定不同的HADR_SYNCMODE 值。如需詳細資訊，請參閱相關資源一節）。	
變更目標資料庫日誌封存目的地。	變更目標資料庫日誌封存目的地以符合 Amazon EC2 環境： db2 update db cfg for SAMPLE using LOGARCHME TH1 'DB2REMOTE://DB2AW SS3//SAMPLE_LOGS/' LOGARCHMETH2 OFF DB20000I The UPDATE DATABASE CONFIGURA TION command completed successfully	DBA

任務	Description	所需的技能
在 Amazon EC2 伺服器上設定 Db2 的 HADR。 Amazon EC2	將 上的 HADR 資料庫組態更新db2-ec2為待命： `db2 update db cfg for sample using HADR_LOCAL_HOST db2-ec2 HADR_LOCAL_SVC 50012 HADR_REMOTE_HOST db2-server1 HADR_REMOTE_SVC 50010 HADR_REMOTE_INST db2inst1 HADR_SYNCMODE SUPERASYNC DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully`	DBA

任務	Description	所需的技能

驗證 HADR 設定。	驗證來源和目標 Db2 伺服器上的 HADR 參數。
	
	若要驗證 上的設定db2-server1 ，請執行下列命令：
	
	`db2 get db cfg for sample|grep HADR`
	`HADR database role = PRIMARY`
	`HADR local host name (HADR_LOCAL_HOST) = db2-server1`
	`HADR local service name (HADR_LOCAL_SVC) = 50010`
	`HADR remote host name (HADR_REMOTE_HOST) = db2-ec2`
	`HADR remote service name (HADR_REMOTE_SVC) = 50012`
	`HADR instance name of remote server (HADR_REMOTE_INST) = db2inst1`
	`HADR timeout value (HADR_TIMEOUT) = 120`
	`HADR target list (HADR_TARGET_LIST) =`
	`HADR log write synchronization mode (HADR_SYNCMODE) = NEARSYNC`
	`HADR spool log data limit (4KB)`
		DBA

任務	Description	所需的技能
	(HADR_SPOOL_LIMIT) = AUTOMATIC(52000) HADR log replay delay (seconds) (HADR_REPLAY_DELAY) = 0 HADR peer window duration (seconds) (HADR_PEER_WINDOW) = 0 HADR SSL certificate label (HADR_SSL_LABEL) = HADR SSL Hostname Validation (HADR_SSL_HOST_VAL) = OFF 若要驗證 上的設定db2-ec2，請執行下列命令： db2 get db cfg for sample|grep HADR HADR database role = STANDBY HADR local host name (HADR_LOCAL_HOST) = db2-ec2 HADR local service name (HADR_LOCAL_SVC) = 50012 HADR remote host name (HADR_REMOTE_HOST) = db2-server1 HADR remote service name (HADR_REMOTE_SVC) = 50010 HADR instance name of remote server	

任務	Description	所需的技能
	(HADR_REMOTE_INST) = db2inst1 HADR timeout value (HADR_TIMEOUT) = 120 HADR target list (HADR_TARGET_LIST) = HADR log write synchronization mode (HADR_SYNCMODE) = SUPERASYNC HADR spool log data limit (4KB) (HADR_SPOOL_LIMIT) = AUTOMATIC(52000) HADR log replay delay (seconds) (HADR_REPLAY_DELAY) = 0 HADR peer window duration (seconds) (HADR_PEER_WINDOW) = 0 HADR SSL certificate label (HADR_SSL_LABEL) = HADR SSL Hostname Validation (HADR_SSL_HOST_VAL) = OFF HADR_LOCAL_HOST 、HADR_REMOTE_HOST 、 HADR_LOCAL_SVC 和 HADR_REMOTE_SVC 參數指出一個主要和一個待命 HADR 設定。	

任務	Description	所需的技能

啟動 Db2 HADR 執行個體。	首先在待命伺服器上啟動 Db2 HADR 執行個體db2-ec2：
	`db2 start hadr on db sample as standby`
	`DB20000I The START HADR ON DATABASE command completed successfully.`
	在主要（來源）伺服器 上啟動 Db2 HADRdb2-server1 ：
	`db2 start hadr on db sample as primary`
	`DB20000I The START HADR ON DATABASE command completed successfully.`
	Db2 on 內部部署和 Amazon EC2 之間的 HADR 連線現已成功建立。Db2 主要伺服器db2-ec2會即時db2-server1 開始將交易日誌記錄串流至 。
		DBA

當內部部署存在 HADR 時設定 HADR

任務	Description	所需的技能
在 Amazon EC2 上新增 Db2 作為輔助待命。 Amazon EC2	如果 HADR 在內部部署 Db2 執行個體上執行，您可以在 上執行下列命令HADR_TARGET_LIST ，使用 新增 Amazon EC2 上的 Db2 做為	DBA

任務	Description	所需的技能
	輔助待命db2-ec2： Amazon EC2 db2 update db cfg for sample using HADR_LOCAL_HOST db2-ec2 HADR_LOCAL_SVC 50012 HADR_REMOTE_HOST db2-server1 HADR_REMOTE_SVC 50010 HADR_REMOTE_INST db2inst1 HADR_SYNCMODE SUPERASYNC DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully. db2 update db cfg for sample using HADR_TARGET_LIST "db2-server1:50010|db2-server2:50011" DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.	

任務	Description	所需的技能
將輔助待命資訊新增至內部部署伺服器。	HADR_TARGET_LIST 在兩個內部部署伺服器上更新（主要和待命）。 在 db2-server1 上執行下列程式碼： `db2 update db cfg for sample using HADR_TARGET_LIST "db2-server2:50011|db2-ec2:50012" DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully. SQL1363W One or more of the parameters submitted for immediate modification were not changed dynamically. For these configuration parameters, the database must be shutdown and reactivated before the configuration parameter changes become effective.`  在 db2-server2 上執行下列程式碼：  `db2 update db cfg for sample using HADR_TARGET_LIST "db2-serv`	DBA

任務	Description	所需的技能
	er1:50010|db2-ec2:50012" DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully. SQL1363W One or more of the parameters submitted for immediate modification were not changed dynamically. For these configuration parameters, the database must be shutdown and reactivated before the configuration parameter changes become effective.	

任務	Description	所需的技能
驗證 HADR 設定。	驗證來源和目標 Db2 伺服器上的 HADR 參數。 在 db2-server1 上執行下列程式碼： `db2 get db cfg for sample|grep HADR HADR database role = PRIMARY HADR local host name (HADR_LOCAL_HOST) = db2-server1 HADR local service name (HADR_LOCAL_SVC) = 50010 HADR remote host name (HADR_REMOTE_HOST) = db2-server2 HADR remote service name (HADR_REMOTE_SVC) = 50011 HADR instance name of remote server (HADR_REMOTE_INST) = db2inst1 HADR timeout value (HADR_TIMEOUT) = 120 HADR target list (HADR_TARGET_LIST) = db2-server2:50011|db2-ec2:50012 HADR log write synchronization mode`	

任務	Description	所需的技能
	(HADR_SYNCMODE) = NEARSYNC HADR spool log data limit (4KB) (HADR_SPOOL_LIMIT) = AUTOMATIC(52000) HADR log replay delay (seconds) (HADR_REPLAY_DELAY) = 0 HADR peer window duration (seconds) (HADR_PEER_WINDOW) = 0 HADR SSL certificate label (HADR_SSL_LABEL) = HADR SSL Hostname Validation (HADR_SSL_HOST_VAL) = OFF 在 db2-server2 上執行下列程式碼： db2 get db cfg for sample|grep HADR HADR database role = STANDBY HADR local host name (HADR_LOCAL_HOST) = db2-server2 HADR local service name (HADR_LOCAL_SVC) = 50011 HADR remote host name (HADR_REMOTE_HOST) = db2-server1 HADR remote service name	

任務	Description	所需的技能
	(HADR_REMOTE_SVC) = 50010 HADR instance name of remote server (HADR_REMOTE_INST) = db2inst1 HADR timeout value (HADR_TIMEOUT) = 120 HADR target list (HADR_TARGET_LIST) = db2-server1:50010|db2-ec2:50012 HADR log write synchronization mode (HADR_SYNCMODE) = NEARSYNC HADR spool log data limit (4KB) (HADR_SPOOL_LIMIT) = AUTOMATIC(52000) HADR log replay delay (seconds) (HADR_REPLAY_DELAY) = 0 HADR peer window duration (seconds) (HADR_PEER_WINDOW) = 0 HADR SSL certificate label (HADR_SSL_LABEL) = HADR SSL Hostname Validation (HADR_SSL_HOST_VAL) = OFF 在 db2-ec2上執行下列程式碼： db2 get db cfg for sample|grep HADR	

任務	Description	所需的技能
	HADR database role = STANDBY HADR local host name (HADR_LOCAL_HOST) = db2-ec2 HADR local service name (HADR_LOCAL_SVC) = 50012 HADR remote host name (HADR_REMOTE_HOST) = db2-server1 HADR remote service name (HADR_REMOTE_SVC) = 50010 HADR instance name of remote server (HADR_REMOTE_INST) = db2inst1 HADR timeout value (HADR_TIMEOUT) = 120 HADR target list (HADR_TARGET_LIST) = db2-server1:50010|db2-server2:50011 HADR log write synchronization mode (HADR_SYNCMODE) = SUPERASYNC HADR spool log data limit (4KB) (HADR_SPOOL_LIMIT) = AUTOMATIC(52000) HADR log replay delay (seconds) (HADR_REPLAY_DELAY) = 0	

任務	Description	所需的技能
	HADR peer window duration (seconds) (HADR_PEER_WINDOW) = 0 HADR SSL certificate label (HADR_SSL_LABEL) = HADR SSL Hostname Validation (HADR_SSL_HOST_VAL) = OFF HADR_LOCAL_HOST 、HADR_LOCAL_SVC 、HADR_REMOTE_HOST 、 HADR_REMOTE_SVC 和 HADR_TARGET_LIST 參數指出一個主要和兩個待命 HADR 設定。	

任務	Description	所需的技能

停止和啟動 Db2 HADR。	HADR_TARGET_LIST 現在已在所有三個伺服器上設定。每個 Db2 伺服器都知道另外兩個伺服器。停止並重新啟動 HADR（短暫中斷），以利用新的組態。
	
	在 db2-server1 上執行下列命令：
	
	`db2 stop hadr on db sample`
	`db2 deactivate db sample`
	`db2 activate db sample`
	
	在 db2-server2 上執行下列命令：
	
	`db2 deactivate db sample`
	`db2 start hadr on db sample as standby`
	`SQL1766W The command completed successfully`
	
	在 db2-ec2上執行下列命令：
	
	`db2 start hadr on db sample as standby`
	`SQL1766W The command completed successfully`
	
	在 db2-server1 上執行下列命令：
		DBA

任務	Description	所需的技能

	`db2 start hadr on db sample as primary`
	`SQL1766W The command completed successfully`
	
	Db2 on 內部部署和 Amazon EC2 之間的 HADR 連線現已成功建立。Db2 主要伺服器db2-server1 會開始即時串流交易日誌記錄到 db2-server2 和 db2-ec2 。	

在切換時段將 Amazon EC2 上的 Db2 設為主要 Amazon EC2

任務	Description	所需的技能

確定待命伺服器上沒有 HADR 延遲。	從主要伺服器 檢查 HADR 狀態db2-server1 。當 HADR_STATE 處於 REMOTE_CATCHUP 狀態時，請勿發出警示，當 HADR_SYNCMODE 設為 時，這是正常的SUPERASYNC 。PRIMARY_LOG_TIME 和 STANDBY_REPLAY_LOG_TIME 顯示它們處於同步狀態：
	
	`db2pd -hadr -db sample`
	
	`HADR_ROLE = PRIMARY`
	
	`REPLAY_TYPE = PHYSICAL`
		DBA

任務	Description	所需的技能
	HADR_SYNCMODE = SUPERASYNC STANDBY_ID = 2 LOG_STREAM_ID = 0 HADR_STATE = REMOTE_CATCHUP ..... PRIMARY_LOG_TIME = 10/26/2022 02:11:32.000000 (1666750292) STANDBY_LOG_TIME = 10/26/2022 02:11:32.000000 (1666750292) STANDBY_REPLAY_LOG_TIME = 10/26/2022 02:11:32.000000 (1666750292)	

任務	Description	所需的技能

執行 HADR 接管。	若要完成遷移，請執行 HADR 接管命令來建立db2-ec2主要資料庫。使用 命令db2pd來驗證HADR_ROLE 值：
	
	`db2 TAKEOVER HADR ON DATABASE sample`
	`DB20000I The TAKEOVER HADR ON DATABASE command completed successfully.`
	
	`db2pd -hadr -db sample`
	`Database Member 0 -- Database SAMPLE -- Active -- Up 0 days 00:03:25 -- Date 2022-10-26-02.46.45.048988`
	
	`HADR_ROLE = PRIMARY`
	`REPLAY_TYPE = PHYSICAL`
	
	若要完成遷移至 AWS，請將應用程式連線指向 Amazon EC2 上的 Db2。 Amazon EC2	

疑難排解

問題	解決方案
如果您因防火牆和安全原因而使用 NAT，主機可以有兩個 IP 地址（一個內部和一個外部），	若要在 NAT 環境中支援 HADR，您可以使用內部和外部地址HADR_LOCAL_HOST 來設定 。例如，如果 Db2 伺服器具有內部名稱 host1和

問題	解決方案
這可能會導致 HADR IP 地址檢查失敗。START HADR ON DATABASE 命令將傳回下列訊息： HADR_LOCAL_HOST:HADR_LOCAL_SVC (-xx-xx-xx-xx.:50011 (xx.xx.xx.xx:50011)) on remote database is different from HADR_REMOTE_HOST:HADR_REMOTE_SVC (xx-xx-xx-xx.:50011 (x.x.x.x:50011)) on local database.	外部名稱 host1E，則 HADR_LOCAL_HOST 可以是 HADR_LOCAL_HOST: "host1 | host1E"。

相關資源

- 不同作業系統和硬體平台之間的 Db2 備份和還原操作
- 設定 Db2 STORAGE ACCESS ALIAS 和 DB2REMOTE
- Db2 高可用性災難復原
- hadr_syncmode - 用於對等狀態組態參數中日誌寫入的 HADR 同步模式

使用 appcmd.exe 將 IIS 託管應用程式遷移至 Amazon EC2

Deepak Kumar，Amazon Web Services

總結

當您將網際網路資訊服務 (IIS) 託管的應用程式遷移至 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體時，您需要解決幾個身分驗證挑戰。這些挑戰包括重新輸入應用程式集區身分的網域登入資料，以及可能為適當的網站功能重新產生機器金鑰。您可以使用 AWS Directory Service 與內部部署 Active Directory 建立信任關係，或在其中建立新的受管 Active Directory AWS。此模式說明在 Amazon EC2 執行個體上使用 IIS 備份和還原功能的乾淨遷移方法。此方法使用 appcmd.exe 在目標 EC2 執行個體上解除安裝和重新安裝 IIS，以便成功遷移 IIS 託管的網站、應用程式集區身分和機器金鑰。

先決條件和限制

先決條件

- AWS 帳戶 目標伺服器的作用中 。
- 具有託管網站的功能來源 IIS 伺服器。
- 了解 IIS 工作原則，例如管理和組態。
- 來源和目標伺服器上的系統管理員存取權。
- 已完成將來源 IIS 伺服器遷移至目標 AWS 帳戶。您可以使用遷移工具 AWS Application Migration Service，例如 Amazon Machine Image (AMI) 快照型方法或其他遷移工具。

限制

- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

- IIS 8.5 或 IIS 10.0

Architecture

來源技術堆疊

- 安裝 IIS 8.5 或 IIS 10.0 的 Windows Server

目標技術堆疊

- 安裝 IIS 8.5 或 IIS 10.0 的 Windows Server
- 應用程式遷移服務

目標架構

下圖顯示此模式的工作流程和架構元件。

解決方案包含下列步驟：

1. 在公司資料中心的來源 IIS 伺服器上安裝和設定 AWS 複寫代理程式。此代理程式會啟動複寫程序，並管理 的資料傳輸 AWS。
2. AWS 複寫代理程式會建立與 Application Migration Service 的安全連線，並開始複寫來源伺服器資料，包括 IIS 組態、網站和應用程式檔案。
3. Application Migration Service 會使用複寫的資料在應用程式子網路中啟動 EC2 執行個體。目標 EC2 執行個體會執行 IIS，並包含已遷移的應用程式及其相關聯的 Amazon Elastic Block Store (Amazon EBS) 磁碟區。在初始複寫之後，Application Migration Service 會繼續同步變更，直到您準備好切換到新環境為止。

工具

AWS 服務

- AWS Application Migration Service 可協助您將應用程式重新託管 (提升和轉移) 到 ， AWS 雲端 而無須變更且停機時間最短。
- Amazon Elastic Block Store (Amazon EBS) 提供區塊層級儲存體磁碟區，可搭配使用 Amazon EC2 執行個體。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。

其他工具

- 適用於 Windows Server 的網際網路資訊服務 (IIS) 是 Web 伺服器，具有可擴展且開放的架構，可在 Web 上託管任何內容。IIS 提供一組管理工具，包括管理和命令列工具 （ 例如 appcmd.exe)、受管程式碼和指令碼 APIs，以及 Windows PowerShell 支援。

史詩

在遷移前於來源備份 IIS

任務	Description	所需的技能
建立 IIS 託管網站、組態金鑰和WAS金鑰的備份。	若要為 IIS 託管網站、組態金鑰 (iisConfigurationKey) 和WAS金鑰 (iisWasKey) 建立備份，請在來源伺服器上使用 appcmd.exe。使用下列步驟： 1. 開啟命令提示字元並以管理員身分執行。 2. 執行下列命令：`c:\windows\system32\inetsrv` 3. 執行下列命令，將 取代<backupname> 為您的值： `appcmd.exe add backup "<backupname>"` 若要匯出組態金鑰和WAS金鑰，請執行下列動作： 1. 執行下列命令：`cd c:\windows\Microsoft.NET\Framework\v2.0.50727 5. aspnet_regiis.exe -px "iisConfigurationKey"` 2. 執行下列命令：`C:\temp\iisConfKey.xml -pri`	IIS 管理員

任務	Description	所需的技能
	3. 執行下列命令：`aspnet_regiis.exe -px "iisWasKey" C:\temp\iisWasKey.xml -pri`	

在目標伺服器上解除安裝和重新安裝 IIS

任務	Description	所需的技能
在目標伺服器上解除安裝 IIS。	若要解除安裝目標伺服器上的 IIS，請使用下列步驟： 1. 使用管理員權限登入伺服器。 2. 前往伺服器管理員。 3. 選擇管理、移除角色和功能、下一步和下一步。 4. 清除 Web IIS 核取方塊，然後選擇下一步、下一步。 5. 選取自動重新啟動目的地伺服器核取方塊。解除安裝完成時，系統會重新啟動。 6. 前往 `C:\Windows\System32\inetsrv` 並確認此資料夾為空，以確認 IIS 已成功解除安裝。	IIS 管理員
在目標伺服器上安裝 IIS。	若要在目標伺服器上安裝 IIS，請使用下列步驟： 1. 使用管理員權限登入伺服器。 2. 前往伺服器管理員。	IIS 管理員

任務	Description	所需的技能
	3. 選擇新增角色和功能、下一步、下一步、下一步和下一步。 4. 選取 Webserver IIS 核取方塊。出現提示時，選擇新增功能。 5. 選擇應用程式開發、新增功能、選取所有功能中顯示的所有核取方塊。您可以在確認之前選取精細層級的功能。 6. 選擇 Install (安裝)。	

從備份還原 IIS 網站和組態

任務	Description	所需的技能
還原 IIS 網站和組態。	若要還原您在目標伺服器上從來源伺服器建立的 IIS 備份，請使用下列步驟： 1. 在 backup資料夾中建立名為 的C:\windows\system32\inetsrv 資料夾，並將來源備份資料夾放在其中。 2. 開啟命令提示字元並以管理員身分執行。 3. 執行下列命令：cd C:\windows\system32\inetsrv 4. 執行下列命令：C:\windows\system32\inetsrv	IIS 管理員

任務	Description	所需的技能
	\appcmd.exe list backup 5. 執行下列命令：C:\windows\system32\inetsrv\appcmd.exe restore backup <sourcebackup> 6. 執行下列命令：cd c:\windows\Microsoft.NET\Framework64\v2.0.50727\ 7. 執行下列命令：aspnet_regiis.exe -pi "iisConfigurationKey" C:\temp\iisConfKey.xml -exp 8. 執行下列命令：aspnet_regiis.exe -pi "iisWasKey" C:\temp\iisWasKey.xml -exp 9. 重設 IIS，然後前往應用程式集區身分和網站進階設定以重新輸入密碼。	

相關資源

AWS 文件

- 安裝複寫代理程式 AWS(AWS Application Migration Service 文件)

AWS 方案指引

- 使用 將內部部署 VM 遷移至 Amazon EC2 AWS Application Migration Service

- 使用 AMIs 或 Amazon EBS 快照進行備份

Microsoft 資源

- 應用程式集區身分
- IIS 文件
- IIS 8 appcmd.exe 文件
- IIS 10 appcmd.exe 文件
- 強大的管理員工具

使用 Application Migration Service 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon EC2

Senthil Ramasamy，Amazon Web Services

總結

此模式說明將 Microsoft SQL Server 資料庫從內部部署資料中心遷移至 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的步驟。它使用 AWS Application Migration Service (AWS MGN) 來重新託管資料庫，方法是使用自動lift-and-shift。 AWS MGN 會執行來源資料庫伺服器的區塊層級複寫。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 內部部署資料中心中的來源 Microsoft SQL Server 資料庫

限制

- 您的網路頻寬可能會在內部部署資料中心與 之間受到限制 AWS。
- AWS MGN 僅限於託管在具有專用儲存的獨立伺服器上的資料庫。它不支援遷移叢集資料庫系統和變更速率超過網路輸送量的資料庫系統。
- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

產品版本

- Microsoft SQL Server 資料庫的所有版本
- 支援 AWS MGN 的 Windows 和 Linux 作業系統

Architecture

來源技術堆疊

內部部署 Microsoft SQL Server 資料庫

目標技術堆疊

Amazon EC2 執行個體上的 Microsoft SQL Server 資料庫

目標架構

此架構使用 AWS MGN 將資料從內部部署企業資料中心複寫到 AWS。圖表顯示資料複寫程序、API 通訊，以及測試和切換階段。

1. 資料複寫：
   - AWS MGN 會將內部部署企業資料中心的資料複寫到 ， AWS 並啟動持續複寫的變更。
   - 預備子網路中的複寫伺服器會接收和處理資料。
2. API 通訊：
   - 複寫伺服器會透過 TCP 連接埠 443 連線至 AWS MGN、Amazon EC2 和 Amazon Simple Storage Service (Amazon S3) API 端點。Amazon S3
   - AWS MGN 會管理遷移。
   - Amazon EC2 會管理執行個體操作。
3. 測試和切換：
   - 使用複寫資料在操作子網路中測試執行個體啟動。
   - 測試成功後， AWS MGN 會為最終遷移建立切換執行個體。

工具

- AWS Application Migration Service (AWS MGN) 可協助您將應用程式重新託管 (提升和轉移) 到 ， AWS 雲端 無需變更且停機時間最短。
- Direct Connect 透過標準乙太網路光纖纜線，將您的內部網路連結至 Direct Connect 位置。透過此連線，您可以直接建立與公有 AWS 服務的虛擬介面，同時略過網路路徑中的網際網路服務供應商。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

最佳實務

- 在虛擬私有雲端 (VPC) 中為 AWS MGN、Amazon EC2 和 Amazon S3 設定 API 區域端點，以禁止從網際網路公開存取。

- 設定 AWS MGN 啟動設定以啟動私有子網路中的目標資料庫伺服器。
- 僅允許資料庫安全群組中的必要連接埠。
- 遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。

史詩

設定 AWS MGN

任務	Description	所需的技能
設定 AWS MGN。	在 AWS Application Migration Service 中搜尋 AWS 管理主控台，然後啟動設定程序。這會建立複寫範本，並將您重新導向至 MGN 主控台來源伺服器頁面。當您設定 MGN 服務時，請從產生的清單中選擇服務角色。	DBA，遷移工程師
新增來源伺服器。	新增現場部署來源資料庫伺服器的詳細資訊，然後新增伺服器。	DBA，遷移工程師
在來源伺服器上安裝 AWS MGN 代理程式。	下載 AWS MGN 代理程式安裝程式到您的本機系統，並將安裝程式轉移到您的來源資料庫伺服器。若要驗證安裝程式雜湊，請參閱驗證AWS 下載的 Windows 2012 複寫代理程式安裝程式。	DBA，遷移工程師

在來源機器上安裝 AWS MGN 代理程式

任務	Description	所需的技能
產生用戶端 IAM 登入資料。	安裝 AWS MGN 代理程式之前，請建立具有適當許可的新 IAM 使用者來產生 AWS 憑證。 如需詳細資訊，請參閱適用於 的 AWS 受管政策 AWS Application Migration Service，以及產生所需的 AWS 登入資料。	DBA，遷移工程師
在來源伺服器上安裝 代理程式。	在託管 Microsoft SQL Server 資料庫的來源機器上安裝代理程式。如需詳細資訊，請參閱在 Windows 伺服器上安裝 AWS 複寫代理程式。 提供下列 AWS 登入資料： • AWS 區域 • AWS 存取金鑰 ID • AWS 私密存取金鑰 您的唯一 AWS 登入資料可讓 AWS MGN 代理程式驗證和執行遷移任務。	應用程式擁有者、DBA、遷移工程師
選擇要複寫的磁碟。	輸入您的 AWS 登入資料後，安裝程式會驗證您的伺服器是否符合代理程式安裝的最低需求（例如，伺服器是否有足夠的磁碟空間安裝 AWS MGN 代理程式）。安裝程式會顯示磁碟區標籤和儲存詳細資訊。	DBA，遷移工程師

任務	Description	所需的技能
	若要使用 AWS MGN 服務複寫資料庫，請選取來源伺服器上適用的磁碟。輸入每個磁碟的路徑，以逗號分隔。如果您想要複寫所有磁碟，請將路徑保留空白。確認選取的磁碟後，安裝會繼續進行。	
監控同步進度。	AWS 複寫代理程式會先擷取所選磁碟的快照，然後複寫資料，以啟動同步程序。 您可以從 AWS MGN 主控台的來源伺服器頁面監控同步進度。如需詳細資訊，請參閱在遷移生命週期中監控伺服器。	DBA，遷移工程師

使用 AWS MGN 複寫

任務	Description	所需的技能
管理複寫進度。	開始初始同步後，您的來源伺服器會出現在 AWS MGN 主控台中，您可以在其中管理和監控遷移。主控台會顯示完成複寫的預估時間，這取決於所選磁碟的總大小和可用的網路頻寬。	DBA，遷移工程師
驗證同步。	來源伺服器上的磁碟完全同步後，請確認所有選取的磁碟都列為完全同步，而且主控台中不會回報任何錯誤。 AWS MGN 主控台接著會自動將遷移生命週期狀態轉換為就	應用程式擁有者、DBA、遷移工程師

任務	Description	所需的技能
	緒進行測試，表示 中的複寫環境 AWS 已準備好進行效能和功能測試。	

測試和切換

任務	Description	所需的技能
設定啟動設定。	在 AWS MGN 主控台中選擇來源伺服器，並更新目標測試執行個體的啟動設定。從來源伺服器詳細資訊頁面，導覽至啟動設定索引標籤以設定測試執行個體。 選擇經濟實惠的執行個體類型和 Amazon Elastic Block Store (Amazon EBS) 磁碟區類型，然後設定安全群組和網路需求。如需詳細資訊，請參閱啟動設定。	DBA，遷移工程師
啟動目標測試執行個體。	導覽至同步來源機器的 AWS MGN 主控台，然後選擇測試並切換，然後啟動測試執行個體，以啟動目標測試執行個體。 這會建立使用您設定的 設定部署測試執行個體的啟動任務。執行個體會在 中啟動， AWS 雲端 並複寫來源資料庫伺服器的環境。從啟動歷史記錄頁面監控啟動進度，您可以在其中	DBA，遷移工程師

任務	Description	所需的技能
	追蹤執行個體建立並解決任何問題。	
驗證目標測試執行個體。	驗證 Amazon EC2 資料庫伺服器： 1. 確保 AWS MGN 主控台顯示測試執行個體已成功執行。 2. 使用 RDP 用戶端登入。 3. 從開始功能表中，開啟 SQL Server Configuration Manager。 4. 驗證 SQL Server 服務的狀態，包括 SQL Server (MSSQLSERVER) 和 SQL Server Agent，確保其完整且設定為正確的啟動類型。 5. 比較測試執行個體與來源資料庫伺服器之間的磁碟設定和組態，確認磁碟機代號、磁碟區、磁碟配置和必要的目錄已正確映射。 6. 連線至測試 Amazon EC2 執行個體上的 SQL Server，並確認所有來源資料庫都已遷移並出現在資料庫清單中。 執行驗證測試，以確保資料庫如預期般運作。	DBA，遷移工程師

任務	Description	所需的技能

重新命名伺服器。	AWS MGN 遷移涉及現場部署來源伺服器的儲存層級複本。您的 SQL Server EC2 執行個體在其二進位檔中僅包含原始來源伺服器的詳細資訊，因此請更新二進位資訊以反映新伺服器的名稱。
	1. 使用 SQL Server Management Studio (SSMS) 連線到 SQL Server EC2 執行個體。
	2. 檢查伺服器的名稱：
	`SELECT @@SERVERNAME AS 'Current Server Name';`
	3. 將 取代NEW_SERVER_NAME 為您的伺服器名稱，以重新命名 SQL Server 執行個體：
	`EXEC sp_dropserver 'OLD_SERVER_NAME'`
	`EXEC sp_addserver 'NEW_SERVER_NAME', 'local';`
	4. 驗證伺服器名稱是否正確：
	`SELECT @@SERVERNAME AS 'Updated Server Name';`
	5. 重新啟動 SQL Server 執行個體。
		DBA，遷移工程師

任務	Description	所需的技能
啟動切換執行個體。	在 AWS MGN 主控台的來源伺服器頁面上，確認伺服器的遷移生命週期狀態已準備好進行切換。設定切換執行個體的啟動設定，確保設定反映您的現場部署環境。 啟動切換之前，請關閉您的現場部署資料庫，以確保下列事項： • 所有進行中的交易都已完成。 • 在切換過程中不會發生新的交易。 • 來源和目標磁碟之間的資料同步已完成。 在 AWS MGN 主控台中啟動切換執行個體。當切換執行個體運作時，請登入執行個體並執行下列測試： 1. 確定 SQL Server 正確啟動且可存取資料庫。 2. 驗證您的資料是否完整且與來源伺服器一致。 3. 執行任何應用程式測試，以確認它們如預期般執行。 4. 在 AWS MGN 主控台中，將遷移狀態設定為完成切換。 5. 開始將流量路由到 EC2 執行個體。	應用程式擁有者、DBA、遷移工程師、遷移負責人

疑難排解

問題	解決方案
初始同步會在身分驗證步驟失敗。	這是網路連線問題。複寫伺服器無法連線至 AWS MGN。

相關資源

AWS 文件

- 入門 AWS Application Migration Service
- 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon EC2
- 什麼是 Amazon EC2 上的 Microsoft SQL Server？

影片

- 使用 執行提升和轉移遷移 AWS Application Migration Service （影片）

在 上將 F5 BIG-IP 工作負載遷移至 F5 BIG-IP VE AWS 雲端

Suresh Vee Mirrori 和 Deepak Kumar，Amazon Web Services

總結

組織希望遷移到 AWS 雲端 ，以提高敏捷性和彈性。將 F5 BIG-IP 安全和流量管理解決方案遷移至 後 AWS 雲端，您可以專注於在整個企業架構中採用高價值操作模型的敏捷性和採用性。

此模式說明如何將 F5 BIG-IP 工作負載遷移至 上的 F5 BIG-IP Virtual Edition (VE) 工作負載 AWS 雲端。工作負載將透過重新託管現有環境並部署轉換的各個層面來進行遷移，例如服務探索和 API 整合。 AWS CloudFormation 範本可加速工作負載遷移至 AWS 雲端。

此模式適用於技術工程和架構團隊，這些團隊正在遷移 F5 安全和流量管理解決方案，並隨附 AWS 規範性指引網站上的指南從 F5 BIG-IP 遷移至 F5 BIG-IP VE AWS 雲端。

先決條件和限制

先決條件

- 現有的內部部署 F5 BIG-IP 工作負載。
- BIG-IP VE 版本的現有 F5 授權。
- 作用中 AWS 帳戶。
- 現有的虛擬私有雲端 (VPC)，透過 NAT 閘道或彈性 IP 地址設定輸出，並設定存取下列端點：Amazon Simple Storage Service (Amazon S3)、Amazon Elastic Compute Cloud (Amazon EC2)、AWS Security Token Service (AWS STS) 和 Amazon CloudWatch。您也可以將模組化和可擴展的 VPC 架構 Quick Start 修改為部署的建置區塊。
- 一或兩個現有的可用區域，視您的需求而定。
- 每個可用區域中有三個現有的私有子網路。
- AWS CloudFormation 範本，可在 F5 GitHub 儲存庫中使用。

在遷移期間，視您的需求而定，您也可以使用下列項目：

- 管理彈性 IP 地址映射、次要 IP 映射和路由表變更的 F5 雲端容錯移轉延伸。
- 如果您使用多個可用區域，則需要使用 F5 雲端容錯移轉延伸來處理彈性 IP 映射到虛擬伺服器。
- 您應該考慮使用 F5 Application Services 3 (AS3)、F5 Application Services Templates (FAST) 或其他基礎設施做為程式碼 (IaC) 模型來管理組態。在 IaC 模型中準備組態並使用程式碼儲存庫，將有助於遷移和持續的管理工作。

專業知識

- 此模式需要熟悉如何將一或多個 VPCs 連接到現有資料中心。如需詳細資訊，請參閱 Network-to-Amazon Amazon VPC 連線選項。
- F5 產品和模組也需要熟悉，包括流量管理作業系統 (TMOS)、本機流量管理員 (LTM)、全域流量管理員 (GTM)、存取政策管理員 (APM)、應用程式安全管理員 (ASM)、進階防火牆管理員 (AFM) 和 BIG-IQ。

產品版本

- 我們建議您使用 F5 BIG-IP 13.1 版或更新版本，雖然模式支援 F5 BIG-IP 12.1 版或更新版本。

Architecture

來源技術堆疊

- F5 BIG-IP 工作負載

目標技術堆疊

- Amazon CloudFront
- CloudWatch
- Amazon EC2
- Amazon S3
- Amazon VPC
- AWS Global Accelerator
- AWS STS
- AWS Transit Gateway
- F5 BIG-IP VE

目標架構

工具

- CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和 生命週期中管理資源 AWS 區域。
- Amazon CloudFront 透過全球資料中心網路提供 Web 內容，進而降低延遲並改善效能，進而加快 Web 內容的發佈速度。
- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您在 上執行的應用程式。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 中提供可擴展的運算容量 AWS 雲端 。您可以根據需要啟動任意數量的虛擬伺服器，並快速擴展或縮減它們。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Security Token Service (AWS STS) 可協助您為使用者請求暫時、有限權限的登入資料。
- AWS Transit Gateway 是連接虛擬私有雲端 (VPC) 和內部部署網路的中央中樞。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。

史詩

探索和評估

任務	Description	所需的技能
評估 F5 BIG-IP 的效能。	收集並記錄虛擬伺服器上應用程式的效能指標，以及將遷移的系統指標。這將有助於正確調整目標 AWS 基礎設施的大小，以獲得更好的成本最佳化。	F5 架構師、工程師和網路架構師、工程師
評估 F5 BIG-IP 作業系統和組態。	評估要遷移哪些物件，以及是否需要維護網路結構，例如 VLANs。	F5 架構師、工程師

任務	Description	所需的技能
評估 F5 授權選項。	評估您需要的授權和耗用量模型。此評估應以您對 F5 BIG-IP 作業系統和組態的評估為基礎。	F5 架構師、工程師
評估公有應用程式。	決定哪些應用程式需要公有 IP 地址。將這些應用程式與所需的執行個體和叢集保持一致，以滿足效能和服務層級協議 (SLA) 要求。	F5 架構師、雲端架構師、網路架構師、工程師、應用程式團隊
評估內部應用程式。	評估內部使用者將使用哪些應用程式。請確定您知道這些內部使用者在組織中的位置，以及這些環境如何連線到 AWS 雲端。您也應該確保這些應用程式可以使用網域名稱系統 (DNS) 做為預設網域的一部分。	F5 架構師、雲端架構師、網路架構師、工程師、應用程式團隊
完成 AMI。	並非所有 F5 BIG-IP 版本都會建立為 Amazon Machine Image (AMIs)。如果您有特定的必要快速修正工程 (QFE) 版本，您可以使用 F5 BIG-IP Image Generator Tool。如需此工具的詳細資訊，請參閱「相關資源」一節。	F5 架構師、雲端架構師、工程師
完成執行個體類型和架構。	決定執行個體類型、VPC 架構和互連架構。	F5 架構師、雲端架構師、網路架構師、工程師

完整的安全與合規相關活動

任務	Description	所需的技能
記錄現有的 F5 安全政策。	收集並記錄現有的 F5 安全政策。請務必在安全程式碼儲存庫中建立其複本。	F5 架構師、工程師
加密 AMI。	（選用）您的組織可能需要靜態資料加密。如需建立自訂自帶授權 (BYOL) 映像的詳細資訊，請參閱「相關資源」一節。	F5 架構師、工程師雲端架構師、工程師
強化裝置。	這將有助於防止潛在的漏洞。	F5 架構師、工程師

設定您的新 AWS 環境

任務	Description	所需的技能
建立邊緣和安全帳戶。	登入 AWS 管理主控台 並建立 AWS 帳戶 ，以提供和操作邊緣和安全服務。這些帳戶可能與為共用服務和應用程式操作 VPCs的帳戶不同。此步驟可作為登陸區域的一部分完成。	雲端架構師、工程師
部署邊緣和安全 VPCs。	設定交付邊緣和安全服務所需的 VPCs。	雲端架構師、工程師
連線至來源資料中心。	連線至託管 F5 BIG-IP 工作負載的來源資料中心。	雲端架構師、網路架構師、工程師
部署 VPC 連線。	將邊緣和安全服務 VPCs連接到應用程式 VPCs。	Network Architect，工程師

任務	Description	所需的技能
部署執行個體。	使用「相關資源」區段中的 CloudFormation 範本來部署執行個體。	F5 架構師、工程師
測試和設定執行個體容錯移轉。	請確定已設定 AWS 進階 HA iAPP 範本或 F5 雲端容錯移轉延伸模組並正常運作。	F5 架構師、工程師

設定聯網

任務	Description	所需的技能
準備 VPC 拓撲。	開啟 Amazon VPC 主控台，並確保您的 VPC 具有 F5 BIG-IP VE 部署所需的所有子網路和保護。	網路架構師、F5 架構師、雲端架構師、工程師
準備您的 VPC 端點。	準備 Amazon EC2、Amazon S3 的 VPC 端點，以及 AWS STS F5 BIG-IP 工作負載是否無法存取 TMM 界面上的 NAT 閘道或彈性 IP 地址。	雲端架構師、工程師

遷移資料

任務	Description	所需的技能
遷移組態。	在 上將 F5 BIG-IP 組態遷移至 F5 BIG-IP VE AWS 雲端。	F5 架構師、工程師
關聯次要 IPs。	虛擬伺服器 IP 地址與指派給執行個體的次要 IP 地址有關聯。指派次要 IP 地址，並確認	F5 架構師、工程師

任務	Description	所需的技能
	已選取「允許重新對應/重新指派」。	

測試組態

任務	Description	所需的技能
驗證虛擬伺服器組態。	測試虛擬伺服器。	F5 架構師、應用程式團隊

完成操作

任務	Description	所需的技能
建立備份策略。	必須關閉系統才能建立完整快照。如需詳細資訊，請參閱「相關資源」一節中的「更新 F5 BIG-IP 虛擬機器」。	F5 架構師、雲端架構師、工程師
建立叢集容錯移轉 Runbook。	確定容錯移轉 Runbook 程序已完成。	F5 架構師、工程師
設定和驗證記錄。	設定 F5 遙測串流，將日誌傳送至所需的目的地。	F5 架構師、工程師

完成切換

任務	Description	所需的技能
切換到新的部署。		F5 架構師、雲端架構師、網路架構師、工程師、AppTeams

相關資源

遷移指南

- 在 上從 F5 BIG-IP 遷移至 F5 BIG-IP VE AWS 雲端

F5 資源

- CloudFormation F5 GitHub 儲存庫中的 範本
- 中的 F5 AWS Marketplace
- F5 BIG-IP VE 概觀
- 範例 Quickstart - BIG-IP Virtual Edition 搭配 WAF (LTM + ASM)
- F5 應用程式服務開啟 AWS：概觀 （ 影片 ）
- F5 Application Services 3 延伸模組使用者指南
- F5 雲端文件
- F5 iControl REST wiki
- F5 單一組態檔案概觀 (11.x - 15.x)
- F5 白皮書
- F5 BIG-IP 影像產生器工具
- 更新 F5 BIG-IP VE 虛擬機器
- UCS 封存 "platform-migrate" 選項概觀

使用二進位方法將內部部署 Go Web 應用程式遷移至 AWS Elastic Beanstalk

Suhas Basavaraj 和 Shumaz Mukhtar Kazi，Amazon Web Services

總結

此模式說明如何將內部部署 Go Web 應用程式遷移至 AWS Elastic Beanstalk。應用程式遷移後，Elastic Beanstalk 會建置來源套件的二進位檔，並將其部署至 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。

作為重新託管遷移策略，此模式的方法很快，不需要變更程式碼，這表示測試和遷移時間更少。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署 Go Web 應用程式。
- 包含 Go 應用程式原始碼的 GitHub 儲存庫。如果您不使用 GitHub，還有其他方法可以為 Elastic Beanstalk 建立應用程式原始碼套件。

產品版本

- Elastic Beanstalk 支援的最新 Go 版本。如需詳細資訊，請參閱 Elastic Beanstalk 文件。

Architecture

來源技術堆疊

- 內部部署 Go Web 應用程式

目標技術堆疊

- AWS Elastic Beanstalk
- Amazon CloudWatch

目標架構

工具

- AWS Elastic Beanstalk 可在 AWS 雲端中快速部署和管理應用程式，而使用者不必了解執行這些應用程式的基礎設施。Elastic Beanstalk 可降低管理複雜性而不會限制選擇或控制。
- GitHub 是一種開放原始碼分散式版本控制系統。

史詩

建立 Go Web 應用程式原始碼套件 .zip 檔案

任務	Description	所需的技能
建立 Go 應用程式的原始碼套件。	開啟包含 Go 應用程式原始碼的 GitHub 儲存庫，並準備原始碼套件。原始碼套件包含根目錄中的`application.go` 原始碼檔案，該檔案託管 Go 應用程式的主要套件。如果您不使用 GitHub，請參閱此模式稍早的先決條件一節，了解建立應用程式原始碼套件的其他方式。	系統管理員、應用程式開發人員
建立一個程式組態檔案。	在原始碼套件中建立 `.ebextensions` 資料夾，然後在此資料夾中建立 `options.config` 檔案。如需詳細資訊，請參閱 Elastic Beanstalk 文件。	系統管理員、應用程式開發人員
建立原始碼套件 .zip 檔案。	執行下列命令。 `git archive -o ../godemoapp.zip HEAD` 這會建立原始碼套件 .zip 檔案。下載 .zip 檔案並將其儲存為本機檔案。	系統管理員、應用程式開發人員

任務	Description	所需的技能
	Important .zip 檔案不能超過 512 MB，也不能包含父資料夾或頂層目錄。	

將 Go Web 應用程式遷移至 Elastic Beanstalk

任務	Description	所需的技能
選擇 Elastic Beanstalk 應用程式。	1. 登入 AWS 管理主控台並開啟 Elastic Beanstalk 主控台。 2. 從區域清單中，選擇您的 AWS 區域。 3. 在導覽窗格中，選擇應用程式，然後選擇現有的 Elastic Beanstalk 應用程式或建立一個應用程式。 如需如何建立 Elastic Beanstalk 應用程式的指示，請參閱 Elastic Beanstalk 文件。	系統管理員、應用程式開發人員
啟動 Elastic Beanstalk Web 伺服器環境。	1. 在應用程式概觀頁面上，選擇建立新環境，然後選擇 Web 伺服器環境。 2. 完成環境名稱和網域名稱欄位。 3. 選擇平台版本，然後選取 Go 做為您的平台。	系統管理員、應用程式開發人員

任務	Description	所需的技能
將原始碼套件 .zip 檔案上傳至 Elastic Beanstalk。	1. 在應用程式程式碼中，選擇上傳程式碼，然後選擇本機檔案。 2. 選擇包含原始碼套件的 .zip 檔案。 3. 在版本標籤中，為檔案提供唯一的名稱，然後選擇建立環境。	系統管理員、應用程式開發人員
測試部署的 Go Web 應用程式。	系統會將您重新導向至 Elastic Beanstalk 應用程式的概觀頁面。在概觀頂端的環境 ID 旁，選擇結尾為 的 URL `elasticbeanstalk.com` 以導覽至您的應用程式。您的應用程式必須在其組態檔案中使用此名稱做為環境變數，並在網頁上顯示它。	系統管理員、應用程式開發人員

疑難排解

問題	解決方案
無法透過 Application Load Balancer 存取應用程式。	檢查包含 Elastic Beanstalk 應用程式的目標群組。如果運作狀態不佳，請登入您的 Elastic Beanstalk 執行個體並檢查`nginx.conf` 檔案組態，以確認其路由至正確的運作狀態 URL。您可能需要變更目標群組運作狀態檢查 URL。

相關資源

- Elastic Beanstalk 支援的 Go 平台版本
- 搭配 Elastic Beanstalk 使用組態檔案

- 在 Elastic Beanstalk 中建立範例應用程式

AWS 使用 將內部部署 SFTP 伺服器遷移至 AWS Transfer for SFTP

Akash Kumar，Amazon Web Services

總結

此模式說明如何 AWS 雲端 使用 AWS Transfer for SFTP 服務，將使用 Secure Shell (SSH) 檔案傳輸通訊協定 (SFTP) 的內部部署檔案傳輸解決方案遷移至 。使用者通常會透過其網域名稱或固定 IP 連線到 SFTP 伺服器。此模式涵蓋這兩種情況。

AWS Transfer for SFTP 是 的成員 AWS Transfer Family。這是一項安全傳輸服務，可讓您透過 SFTP 將檔案傳入和傳出 AWS 儲存服務。您可以 AWS Transfer for SFTP 搭配 Amazon Simple Storage Service (Amazon S3) 或 Amazon Elastic File System (Amazon EFS) 使用 。此模式使用 Amazon S3 進行儲存。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 現有的 SFTP 網域名稱或固定的 SFTP IP。

限制

- 您可以在一個請求中傳輸的最大物件目前為 5 GiB。對於大於 100 MiB 的檔案，請考慮使用 Amazon S3 分段上傳。

Architecture

來源技術堆疊

- 內部部署平面檔案或資料庫傾印檔案。

目標技術堆疊

- AWS Transfer for SFTP
- Amazon S3
- Amazon Virtual Private Cloud (Amazon VPC)
- AWS Identity and Access Management (IAM) 角色和政策

- 彈性 IP 位址
- Security groups (安全群組)
- Amazon CloudWatch Logs （選用）

目標架構

自動化和擴展

若要自動化此模式的目標架構，請使用連接的 CloudFormation 範本：

- `amazon-vpc-subnets.yml` 會佈建具有兩個公有和兩個私有子網路的虛擬私有雲端 (VPC)。
- `amazon-sftp-server.yml` 會佈建 SFTP 伺服器。
- `amazon-sftp-customer.yml` 新增 使用者。

工具

AWS 服務

- Amazon CloudWatch Logs 可協助您集中所有系統、應用程式的日誌， AWS 服務 以便您可以監控日誌並將其安全地存檔。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。此模式使用 Amazon S3 做為檔案傳輸的儲存系統。
- AWS Transfer for SFTP 可協助您透過 SFTP 通訊協定將檔案傳入和傳出 AWS 儲存服務。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。

史詩

建立 VPC

任務	Description	所需的技能
建立具有子網路的 VPC。	開啟 Amazon VPC 主控台。建立具有兩個公有子網路的虛擬	開發人員、系統管理員

任務	Description	所需的技能
	私有雲端 (VPC)。（第二個子網路提供高可用性。） —或— 您可以在 CloudFormation 主控台中部署連接的 CloudFormation 範本 `amazon-vpc-subnets.yml` ，以自動化此史詩中的任務。	
新增網際網路閘道。	佈建網際網路閘道並將其連接到 VPC。	開發人員、系統管理員
遷移現有的 IP。	將現有的 IP 連接到彈性 IP 地址。您可以從地址集區建立彈性 IP 地址並使用它。	開發人員、系統管理員

佈建 SFTP 伺服器

任務	Description	所需的技能
建立 SFTP 伺服器。	開啟 AWS Transfer Family 主控台。請遵循 AWS Transfer Family 文件中為伺服器建立面向網際網路的端點中的指示，使用面向網際網路的端點建立 SFTP 伺服器。針對端點類型，選擇託管的 VPC。針對存取，選擇網際網路面向。針對 VPC，選擇您在上一個史詩中建立的 VPC。 —或—	開發人員、系統管理員

任務	Description	所需的技能
	您可以在 CloudFormation 主控台中部署連接的 CloudFormation 範本 `amazon-sftp-server.yml` ，以自動化此史詩中的任務。	
遷移網域名稱。	將現有的網域名稱連接到自訂主機名稱。如果您使用新的網域名稱，請使用 Amazon Route 53 DNS 別名。針對現有的網域名稱，選擇其他 DNS。如需詳細資訊，請參閱 AWS Transfer Family 文件中的使用自訂主機名稱。	開發人員、系統管理員
新增 CloudWatch 記錄角色。	（選用）如果您想要啟用 CloudWatch 記錄，請使用 CloudWatch Logs API 操作 `logs:CreateLogGroup` 、 `logs:CreateLogStream` `logs:DescribeLogStreams` 和 建立`Transfer`角色`logs:PutLogEvents` 。如需詳細資訊，請參閱 AWS Transfer Family 文件中的使用 CloudWatch 記錄活動。	開發人員、系統管理員
儲存並提交。	選擇儲存。針對動作，選擇開始並等待 SFTP 伺服器建立為線上狀態。	開發人員、系統管理員

將彈性 IP 地址映射至 SFTP 伺服器

任務	Description	所需的技能
停止伺服器，以便您可以修改設定。	在 AWS Transfer Family 主控台上，選擇伺服器，然後選取您建立的 SFTP 伺服器。針對 Actions (動作)，選擇 Stop (停止)。當伺服器離線時，選擇編輯以修改其設定。	開發人員、系統管理員
選擇可用區域和子網路。	在可用區域區段中，選擇 VPC 的可用區域和子網路。	開發人員、系統管理員
新增彈性 IP 地址。	針對 IPv4 地址，為每個子網路選擇彈性 IP 地址，然後選擇儲存。	開發人員、系統管理員

新增使用者

任務	Description	所需的技能
建立 IAM 角色，讓使用者存取 S3 儲存貯體。	建立 `Transfer`和 新增的 IAM 角色 `s3:ListBucket` `s3:GetBucketLocation` ，`s3:PutObject` 並使用 S3 儲存貯體名稱做為資源。如需詳細資訊，請參閱 AWS Transfer Family 文件中的建立 IAM 角色和政策。 —或— 您可以在 CloudFormation 主控台中部署連接的 CloudFormation 範本 `amazon-sftp-`	開發人員、系統管理員

任務	Description	所需的技能
	`customer.yml` ，以自動化此史詩中的任務。	
建立 S3 儲存貯體。	為應用程式建立 S3 儲存貯體。	開發人員、系統管理員
建立選用資料夾。	（選用）如果您想要分別存放使用者的檔案，請在特定的 Amazon S3 資料夾中，視需要新增資料夾。	開發人員、系統管理員
建立 SSH 公有金鑰。	若要建立 SSH 金鑰對，請參閱 AWS Transfer Family 文件中的產生 SSH 金鑰。	開發人員、系統管理員
新增使用者。	在AWS Transfer Family 主控台上，選擇伺服器，選擇您建立的 SFTP 伺服器，然後選擇新增使用者。針對主目錄，選擇您建立的 S3 儲存貯體。針對 SSH 公有金鑰，指定 SSH 金鑰對的公有金鑰部分。新增 SFTP 伺服器的使用者，然後選擇新增。	開發人員、系統管理員

測試 SFTP 伺服器

任務	Description	所需的技能
更新安全群組。	在 SFTP 伺服器的安全群組區段中，新增測試機器的 IP 以取得 SFTP 存取權。	開發人員
使用 SFTP 用戶端公用程式來測試伺服器。	使用任何 SFTP 用戶端公用程式測試檔案傳輸。如需用戶端和說明的清單，請參閱 AWS	開發人員

任務	Description	所需的技能
	Transfer Family 文件中的使用用戶端傳輸檔案。	

相關資源

- AWS Transfer Family 使用者指南
- Amazon S3 使用者指南
- Amazon EC2 文件中的彈性 IP 地址

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS Application Migration Service 將內部部署 VM 遷移至 Amazon EC2

Thanh Nguyen，Amazon Web Services

總結

進行應用程式遷移時，組織可以採取不同的方法來將應用程式伺服器從內部部署環境重新託管（提升和轉移）到 Amazon Web Services (AWS) 雲端。其中一種方法是佈建新的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，然後從頭開始安裝和設定應用程式。另一種方法是使用第三方或 AWS 原生遷移服務同時遷移多個伺服器。

此模式概述使用 AWS Application Migration Service 將支援的虛擬機器 (VM) 遷移至 AWS 雲端上的 Amazon EC2 執行個體的步驟。您可以在此模式中使用 方法，手動逐一遷移一或多個虛擬機器，或根據概述的步驟建立適當的自動化指令碼來自動遷移。

先決條件和限制

先決條件

- 支援 Application Migration Service 的其中一個 AWS 區域中的作用中 AWS 帳戶
- 使用 AWS Direct Connect 或虛擬私有網路 (VPN)，或透過網際網路，透過私有網路在來源伺服器與目標 EC2 伺服器之間建立網路連線

限制

- 如需支援區域的最新清單，請參閱支援的 AWS 區域。
- 如需支援的作業系統清單，請參閱支援的作業系統和 Amazon EC2 FAQs的一般區段。

Architecture

來源技術堆疊

- 實體、虛擬或雲端託管伺服器，執行 Amazon EC2 支援的作業系統

目標技術堆疊

- 執行與來源 VM 相同作業系統的 Amazon EC2 執行個體
- Amazon Elastic Block Store (Amazon EBS)

來源和目標架構

下圖顯示解決方案的高階架構和主要元件。在內部部署資料中心中，有虛擬機器具有本機磁碟。在 AWS 上，有一個具有複寫伺服器的預備區域，以及具有 EC2 執行個體用於測試和切換的遷移資源區域。兩個子網路都包含 EBS 磁碟區。

1. 初始化 AWS Application Migration Service。
2. 設定預備區域伺服器組態和報告，包括預備區域資源。
3. 在來源伺服器上安裝代理程式，並使用連續的區塊層級資料複寫（壓縮和加密）。
4. 自動化協同運作和系統轉換，以縮短切換時段。

網路架構

下圖顯示從聯網角度而言，解決方案的高階架構和主要元件，包括內部部署資料中心和 AWS 中主要元件之間通訊所需的通訊協定和連接埠。

工具

- AWS Application Migration Service 可協助您將應用程式重新託管 (提升和轉移) 至 AWS 雲端，無需變更且停機時間最短。

最佳實務

- 在目標 EC2 執行個體的切換完成之前，請勿讓來源伺服器離線或執行重新啟動。
- 提供充分的機會，讓使用者在目標伺服器上執行使用者接受度測試 (UAT)，以識別和解決任何問題。理想情況下，此測試應在切換前至少兩週開始。
- 經常監控 Application Migration Service 主控台上的伺服器複寫狀態，以及早識別問題。
- 使用臨時 AWS Identity and Access Management (IAM) 登入資料進行代理程式安裝，而非永久的 IAM 使用者登入資料。

史詩

產生 AWS 登入資料

任務	Description	所需的技能
建立 AWS 複寫代理程式 IAM 角色。	使用 AWS 帳戶的管理許可登入。 在 AWS Identity and Access Management (IAM) 主控台上，建立 IAM 角色： 1. 在 IAM 主控台上，選擇角色。 2. 選擇建立角色。 3. 在選取受信任實體頁面的受信任實體類型區段 中，選取 AWS 帳戶。 4. 在 AWS 帳戶區段中，選取此帳戶 (<account-id>)。 5. 選擇下一步。 6. 在新增許可頁面上，搜尋AWSApplicationMigrationAgentInstallationPolicy 政策，選取政策名稱旁的核取方塊。 7. 選擇下一步。 8. 在角色詳細資訊頁面上，輸入 MGN_Agent_Installation_Role 做為角色名稱。 9. 驗證欄位是否正確，然後選擇建立角色。	AWS 管理員、遷移工程師
產生臨時安全登入資料。	在已安裝 AWS Command Line Interface (AWS CLI) 的機器	AWS 管理員、遷移工程師

任務	Description	所需的技能
	上，使用管理許可登入。或者（在支援的 AWS 區域內），在 AWS 管理主控台上，使用 AWS 帳戶的管理許可登入，然後開啟 AWS CloudShell。 使用下列命令產生臨時登入資料，<account-id> 以 AWS 帳戶 ID 取代。 `aws sts assume-role --role-arn arn:aws:iam::<account-id>:role/MGN_Agent_Installation_Role --role-session-name mgn_installation_session_role` 從 命令的輸出中，複製 AccessKeyId 、SecretAccessKey 和 的值 SessionToken 。將它們存放在安全的位置，以供日後使用。 **Important** 這些臨時登入資料將在一小時後過期。如果您在一小時後需要登入資料，請重複上述步驟。	

初始化 Application Migration Service 並建立複寫設定範本

任務	Description	所需的技能
初始化服務。	在 主控台上，使用 AWS 帳戶的管理許可登入。 選擇 Application Migration Service，然後選擇開始使用。	AWS 管理員、遷移工程師
建立和設定複寫設定範本。	1. 提供下列組態詳細資訊： a. 選取預備區域子網路。 b. 選取複寫伺服器執行個體類型 (`t3.small`預設為)。 c. 選取 EBS 磁碟區類型（預設為 gp3)。 d. 選取 EBS 加密選項。 e. 確定已選取永遠使用 Application Migration Service 安全群組核取方塊。 f. 如果您使用內部部署環境和 AWS 之間的私有網路連線，請選取使用私有 IP 進行資料複寫 (VPN、DirectConnect、VPC 對等互連）核取方塊。 g. 如果您想要限制 Application Migration Service 的網路頻寬，請選取調節網路頻寬（每個伺服器 - 以 Mbps 為單位）核取方塊。 2. 選擇建立範本。	AWS 管理員、遷移工程師

任務	Description	所需的技能
	Application Migration Service 將自動建立促進資料複寫和啟動遷移伺服器所需的所有 IAM 角色。	

在來源機器上安裝 AWS 複寫代理程式

任務	Description	所需的技能
準備好必要的 AWS 登入資料。	當您在來源伺服器上執行安裝程式檔案時，您將需要輸入先前產生的臨時登入資料，包括 `AccessKeyId` 、`SecretAccessKey` 和 `SessionToken` 。	遷移工程師、AWS 管理員
對於 Linux 伺服器，請安裝 代理程式。	複製安裝程式命令、登入您的來源伺服器，然後執行安裝程式。如需詳細說明，請參閱 AWS 文件。	AWS 管理員、遷移工程師
對於 Windows 伺服器，請安裝代理程式。	將安裝程式檔案下載到每個伺服器，然後執行安裝程式命令。如需詳細說明，請參閱 AWS 文件。	AWS 管理員、遷移工程師
等待初始資料複寫完成。	安裝代理程式後，來源伺服器會顯示在 Application Migration Service 主控台的來源伺服器區段中。等待伺服器進行初始資料複寫。	AWS 管理員、遷移工程師

設定啟動設定

任務	Description	所需的技能
指定伺服器詳細資訊。	在 Application Migration Service 主控台上，選擇來源伺服器區段，然後從清單中選擇伺服器名稱以存取伺服器詳細資訊。	AWS 管理員、遷移工程師
設定啟動設定。	選擇啟動設定索引標籤。您可以設定各種設定，包括一般啟動設定和 EC2 啟動範本設定。如需詳細說明，請參閱 AWS 文件。	AWS 管理員、遷移工程師

執行測試

任務	Description	所需的技能
測試來源伺服器。	1. 在 Application Migration Service 主控台的來源伺服器區段中，確保來源伺服器的遷移生命週期已準備好進行測試，且資料複寫狀態良好。 2. 選取每個來源伺服器左側的核取方塊。 3. 選擇測試和切換，然後選擇啟動測試執行個體。 4. 出現提示時，選擇啟動。 將啟動伺服器。	AWS 管理員、遷移工程師

任務	Description	所需的技能
驗證測試是否成功完成。	測試伺服器完全啟動後，頁面上的提醒狀態會顯示每個伺服器已啟動。	AWS 管理員、遷移工程師
測試伺服器。	針對測試伺服器執行測試，以確保其如預期般運作。	AWS 管理員、遷移工程師

排程和執行切換

任務	Description	所需的技能
排程切換時段。	與相關團隊安排適當的切換時間範圍。	AWS 管理員、遷移工程師
執行切換。	1. 在應用程式遷移主控台的來源伺服器頁面上，選取每個來源伺服器左側的核取方塊。 2. 選擇測試和切換，然後選取標記為「準備切換」。 3. 確認每個來源伺服器的遷移生命週期已準備好進行切換。 4. 選擇測試和切換，然後選擇啟動切換執行個體。 5. 出現提示時，選擇啟動。將啟動伺服器。 來源伺服器的遷移生命週期將變更為進行中的切換。	AWS 管理員、遷移工程師

任務	Description	所需的技能
驗證切換是否成功完成。	切換伺服器完全啟動後，來源伺服器頁面上的提醒狀態會顯示每個伺服器已啟動。	AWS 管理員、遷移工程師
測試伺服器。	針對切換伺服器執行測試，以確保其如預期般運作。	AWS 管理員、遷移工程師
完成切換。	選擇測試和切換，然後選擇完成切換以完成遷移程序。	AWS 管理員、遷移工程師

相關資源

- AWS Application Migration Service
- AWS Application Migration Service 使用者指南

使用 將小型資料集從內部部署遷移至 Amazon S3 AWS SFTP

Charles Gibson 和 Sergiy Shevchenko，Amazon Web Services

總結

此模式說明如何使用 AWS Transfer for SFTP () 將小型資料集 (5 TB 或更少） 從內部部署資料中心遷移至 Amazon Simple Storage Service (Amazon S3)AWS SFTP。資料可以是資料庫傾印或一般檔案。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 您的資料中心與 之間建立 AWS Direct Connect 的連結 AWS

限制

- 資料檔案必須小於 5 TB。對於超過 5 TB 的檔案，您可以執行分段上傳至 Amazon S3 或選擇其他資料傳輸方法。

Architecture

來源技術堆疊

- 內部部署平面檔案或資料庫傾印

目標技術堆疊

- Amazon S3

來源和目標架構

工具

- AWS SFTP – 啟用使用安全檔案傳輸通訊協定 (SFTP) 將檔案直接傳入和傳出 Amazon S3。
- AWS Direct Connect – 從現場部署資料中心建立專用網路連線 AWS。

- VPC 端點 – 可讓您在沒有網際網路閘道、網路位址轉譯 (NAT) 裝置、VPN 連線或 Direct Connect 連線 AWS PrivateLink 的情況下，將 VPC 私下連線至支援的 AWS 服務 和採用 技術的 VPC 端點服務。VPC 中的執行個體不需要公有 IP 地址，即可與 服務中的資源通訊。

史詩

準備遷移

任務	Description	所需的技能
記錄目前的 SFTP 要求。		應用程式擁有者、SA
識別身分驗證需求。	要求可能包括金鑰型身分驗證、使用者名稱或密碼，或身分提供者 (IdP)。	應用程式擁有者、SA
識別應用程式整合需求。		應用程式擁有者
識別需要 服務的使用者。		應用程式擁有者
判斷 SFTP 伺服器端點的 DNS 名稱。		聯網
決定備份策略。		SA、DBA （如果傳輸資料）
識別應用程式遷移或切換策略。		應用程式擁有者、SA、DBA

設定基礎設施

任務	Description	所需的技能
在 中建立一或多個虛擬私有雲端 (VPCs) 和子網路 AWS 帳戶。		應用程式擁有者、AMS
建立安全群組和網路存取控制清單 (ACL)。		安全性、聯網、AMS

任務	Description	所需的技能
建立 Amazon S3 儲存貯體。		應用程式擁有者、AMS
建立 AWS Identity and Access Management (IAM) 角色。	建立 IAM 政策，其中包含允許 AWS SFTP 存取 Amazon S3 儲存貯體的許可。此 IAM 政策決定您提供 SFTP 使用者的存取層級。建立另一個 IAM 政策以與 建立信任關係 AWS SFTP。	安全性、AMS
關聯已註冊的網域（選用）。	如果您有自己的註冊網域，您可以將其與 SFTP 伺服器建立關聯。您可以將 SFTP 流量從網域或子網域路由到 SFTP 伺服器端點。	網路、AMS
建立 SFTP 伺服器。	指定 服務用來驗證使用者的身分提供者類型。	應用程式擁有者、AMS
開啟 SFTP 用戶端。	開啟 SFTP 用戶端，並將連線設定為使用 SFTP 端點主機。 AWS SFTP 支援任何標準 SFTP 用戶端。常用的 SFTP 用戶端包括 OpenSSH、WinSCP、Cyberduck 和 FileZilla。您可以從 AWS SFTP 主控台取得 SFTP 伺服器主機名稱。	應用程式擁有者、AMS

規劃和測試

任務	Description	所需的技能
規劃應用程式遷移。	規劃所需的任何應用程式組態變更、設定遷移日期，以及判斷測試排程。	應用程式擁有者、AMS

任務	Description	所需的技能
測試基礎設施。	在非生產環境中進行測試。	應用程式擁有者、AMS

相關資源

參考

- AWS Transfer for SFTP 使用者指南
- AWS Direct Connect resources
- VPC 端點

教學課程和影片

- AWS Transfer for SFTP （影片）
- AWS Transfer for SFTP 使用者指南
- AWS SA 白板 - Direct Connect （影片）

將內部部署 Oracle 資料庫遷移至 Amazon EC2 上的 Oracle

Baji Shaik 和 Pankaj Choudhary，Amazon Web Services

總結

此模式會逐步解說在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上將內部部署 Oracle 資料庫遷移至 Oracle 的步驟。它描述了遷移的兩個選項：使用 AWS Data Migration Service (AWS DMS) 或使用 RMAN、Data Pump 匯入/匯出、可傳輸資料表空間和 Oracle GoldenGate 等原生 Oracle 工具。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署資料中心中的來源 Oracle 資料庫

限制

- Amazon EC2 必須支援目標作業系統 (OS)。如需支援系統的完整清單，請參閱 Amazon EC2 FAQs。

產品版本

- 適用於 Enterprise、Standard、Standard One 和 Standard Two 等版本的 Oracle 10.2 版和更新版本 (適用於 10.x 版)、11g 版且最高可達 12.2 版，以及 18c 版。如需 AWS DMS 支援的最新版本清單，請參閱 AWS DMS 文件中的資料遷移來源中的「內部部署和 Amazon EC2 執行個體資料庫」。

Architecture

來源技術堆疊

- 內部部署 Oracle 資料庫

目標技術堆疊

- Amazon EC2 上的 Oracle 資料庫執行個體

目標架構

資料遷移架構

使用 AWS DMS：

使用原生 Oracle 工具：

工具

- AWS DMS -AWS Database Migration Services(AWS DMS) 支援多種類型的來源和目標資料庫。如需有關支援的資料庫版本和版本的資訊，請參閱使用 Oracle 資料庫做為 AWS DMS 的來源。我們建議您使用最新版本的 AWS DMS，以獲得最全面的版本和功能支援。
- 原生 Oracle 工具 -RMAN、Data Pump 匯入/匯出、可傳輸資料表空間、Oracle GoldenGate

史詩

規劃遷移

任務	Description	所需的技能
驗證來源和目標資料庫的版本。		DBA
識別目標作業系統的版本。		DBA、SysAdmin
根據 Oracle 相容性清單和容量需求，識別目標伺服器執行個體的硬體需求。		DBA、SysAdmin
識別儲存需求（儲存類型和容量）。		DBA、SysAdmin
識別網路需求（延遲和頻寬）。		DBA、SysAdmin

任務	Description	所需的技能
根據容量、儲存功能和網路功能選擇適當的執行個體類型。		DBA、SysAdmin
識別來源和目標資料庫的網路/主機存取安全需求。		DBA、SysAdmin
識別 Oracle 軟體安裝所需的作業系統使用者清單。		DBA、SysAdmin
下載 AWS Schema Conversion Tool (AWS SCT) 和驅動程式。		DBA
為工作負載建立 AWS SCT 專案，並連線至來源資料庫。		DBA
產生 SQL 檔案以建立物件（資料表、索引、序列等）。		DBA
決定備份策略。		DBA、SysAdmin
判斷可用性需求。		DBA
識別應用程式遷移/切換策略。		DBA、SysAdmin、應用程式擁有者

設定基礎設施

任務	Description	所需的技能
在您的 AWS 帳戶中建立虛擬私有雲端 (VPC) 和子網路。		SysAdmin
建立安全群組和網路存取控制清單 ACLs)。		SysAdmin
設定和啟動 EC2 執行個體。		SysAdmin

安裝 Oracle 軟體

任務	Description	所需的技能
建立 Oracle 軟體所需的作業系統使用者和群組。		DBA、SysAdmin
下載必要的 Oracle 軟體版本。		
在 EC2 執行個體上安裝 Oracle 軟體。		DBA、SysAdmin
使用 AWS SCT 產生的指令碼建立物件，例如資料表、主索引鍵、檢視和序列。		DBA

遷移資料 - 選項 1

任務	Description	所需的技能
使用原生 Oracle 工具或第三方工具來遷移資料庫物件和資料。	Oracle 工具包括 Data Pump 匯入/匯出、RMAN、可傳輸資料表空間和 GoldenGate。	DBA

遷移資料 - 選項 2

任務	Description	所需的技能
決定遷移方法。		DBA
在 AWS DMS 主控台中建立複寫執行個體。		DBA
建立來源和目標端點。		DBA
建立複寫任務。		DBA

任務	Description	所需的技能
啟用變更資料擷取 (CDC) 以擷取連續複寫的變更。		DBA
執行複寫任務並監控日誌。		DBA
完成完全載入時，建立次要物件，例如索引和外部索引鍵。		DBA

遷移應用程式

任務	Description	所需的技能
遵循應用程式遷移策略。		DBA、SysAdmin、應用程式擁有者

剪下

任務	Description	所需的技能
遵循應用程式切換/切換策略。		DBA、SysAdmin、應用程式擁有者

關閉專案

任務	Description	所需的技能
關閉臨時 AWS Secrets Manager 資源。		DBA、SysAdmin
檢閱並驗證專案文件。		DBA、SysAdmin、應用程式擁有者
收集遷移時間的指標、手動與工具的 %、節省成本等。		DBA、SysAdmin、應用程式擁有者

任務	Description	所需的技能
關閉專案並提供意見回饋。		

相關資源

參考

- 將 Oracle 資料庫遷移至 AWS 的策略
- 將 Oracle 資料庫遷移至 AWS 雲端
- Amazon EC2 網站
- AWS DMS 網站
- AWS DMS 部落格文章
- Amazon EC2 定價
- 在雲端運算環境中授權 Oracle 軟體

教學課程和影片

- Amazon EC2 入門
- AWS DMS 入門
- Amazon EC2 簡介 - Elastic Cloud Server & Hosting with AWS （影片）

使用 Oracle Data Pump 將內部部署 Oracle 資料庫遷移至 Amazon EC2

Navakanth Talluri，Amazon Web Services

總結

遷移資料庫時，您必須考慮來源和目標資料庫引擎和版本、遷移工具和服務，以及可接受的停機時間期間等因素。如果您要將現場部署 Oracle 資料庫遷移至 Amazon Elastic Compute Cloud (Amazon EC2)，您可以使用 Oracle 工具，例如 Oracle Data Pump 和 Oracle Recovery Manager (RMAN)。如需策略的詳細資訊，請參閱將 Oracle 資料庫遷移至 AWS 雲端。

Oracle Data Pump 可協助您擷取資料庫的邏輯一致備份，並將其還原至目標 EC2 執行個體。此模式說明如何使用 Oracle Data Pump 和 NETWORK_LINK 參數，將現場部署 Oracle 資料庫遷移至 EC2 執行個體，並將停機時間降至最低。NETWORK_LINK 參數會透過資料庫連結開始匯入。目標 EC2 執行個體上的 Oracle Data Pump Import (impdp) 用戶端會連線至來源資料庫、從中擷取資料，以及將資料直接寫入目標執行個體上的資料庫。此解決方案中沒有使用的備份或傾印檔案。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署 Oracle 資料庫：
  - 不是 Oracle Real Application Clusters (RAC) 資料庫
  - 不是 Oracle Automatic Storage Management (Oracle ASM) 資料庫
  - 處於讀寫模式。
- 您已在內部部署資料中心和 AWS 之間建立 AWS Direct Connect 連結。如需詳細資訊，請參閱建立連線 (Direct Connect 文件）。

產品版本

- Oracle 資料庫 10g 版本 1 (10.1) 和更新版本

Architecture

來源技術堆疊

- 內部部署資料中心中的獨立 （非 ASM 和非 ASM) Oracle 資料庫伺服器

目標技術堆疊

- 在 Amazon EC2 上執行的 Oracle 資料庫

目標架構

AWS Well-Architected Framework 的可靠性支柱建議建立資料備份，以協助提供高可用性和彈性。如需詳細資訊，請參閱在 AWS 上執行 Oracle 資料庫的最佳實務中的高可用性架構。此模式會使用 Oracle Active Data Guard 在 EC2 執行個體上設定主要和待命資料庫。為了獲得高可用性，EC2 執行個體應該位於不同的可用區域。不過，可用區域可以位於相同的 AWS 區域或不同的 AWS 區域。

Active Data Guard 提供實體待命資料庫的唯讀存取權，並持續從主要資料庫套用重做變更。根據您的復原點目標 (RPO) 和復原時間目標 (RTO)，您可以選擇同步和非同步重做傳輸選項。

如果主要和待命 EC2 執行個體位於不同的 AWS 區域，下圖會顯示目標架構。

資料遷移架構

完成目標架構的設定後，您可以使用 Oracle Data Pump 將內部部署資料和結構描述遷移至主要 EC2 執行個體。在切換期間，應用程式無法存取現場部署資料庫或目標資料庫。您可以關閉這些應用程式，直到它們可以連接到主要 EC2 執行個體上的新目標資料庫為止。

下圖顯示資料遷移期間的架構。在此範例架構中，主要和待命 EC2 執行個體位於不同的 AWS 區域。

工具

AWS 服務

- AWS Direct Connect 透過標準乙太網路光纖纜線，將您的內部網路連結至 Direct Connect 位置。透過此連線，您可以直接建立與公有 AWS 服務的虛擬介面，同時略過網路路徑中的網際網路服務供應商。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。

其他工具和服務

- Oracle Active Data Guard 可協助您建立、維護、管理和監控待命資料庫。
- Oracle Data Pump 可協助您以高速將資料和中繼資料從一個資料庫移至另一個資料庫。

最佳實務

- 在 AWS 上執行 Oracle Database 的最佳實務
- 使用 NETWORK_LINK 匯入資料

史詩

在 AWS 上設定 EC2 執行個體

任務	Description	所需的技能
識別現場部署主機的來源硬體組態和核心參數。	驗證內部部署組態，包括儲存大小、每秒輸入/輸出操作 (IOPS) 和 CPU。這對以 CPU 核心為基礎的 Oracle 授權很重要。	DBA、SysAdmin
在 AWS 上建立基礎設施。	建立虛擬私有雲端 (VPCs)、私有子網路、安全群組、網路存取控制清單 ACLs)、路由表和網際網路閘道。如需詳細資訊，請參閱下列內容： • VPCs和子網路 • 教學課程：建立 VPC 以搭配資料庫執行個體使用	DBA、AWS 系統管理員
使用 Active Data Guard 設定 EC2 執行個體。	使用 Active Data Guard 組態來設定 AWS EC2 執行個體，如 AWS Well-Architected Framework 中所述。EC2 執行個體上的 Oracle 資料庫版本可能與內部部署版本不同，因為此模式使用邏輯備份。注意下列事項： • 將目標資料庫置於讀寫模式。	DBA、AWS 系統管理員

任務	Description	所需的技能
	• 在目標資料庫上，提供來源資料庫的透明網路子結構 (TNS) 詳細資訊。 如需詳細資訊，請參閱： • 啟動資料庫 (Oracle 文件） • 建立和設定 Oracle 資料庫 (Oracle 文件）	

將資料庫遷移至 Amazon EC2

任務	Description	所需的技能
從 EC2 執行個體建立內部部署資料庫的 dblink。	在 EC2 執行個體上的 Oracle 資料庫與內部部署 Oracle 資料庫之間建立資料庫連結 (dblink)。如需詳細資訊，請參閱使用網路連結匯入移動資料 (Oracle 文件）。	DBA
驗證 EC2 執行個體與內部部署主機之間的連線。	使用 dblink 確認 EC2 執行個體與內部部署資料庫之間的連線正常運作。如需說明，請參閱 CREATE DATABASE LINK (Oracle 文件）。	DBA
停止所有連接到現場部署資料庫的應用程式。	核准資料庫停機時間後，請關閉任何包含到現場部署資料庫的應用程式和相依任務。您可以直接從應用程式或使用 cron 從資料庫執行此操作。如需詳細資訊，請參閱使用 Crontab	DBA，應用程式開發人員

任務	Description	所需的技能
	公用程式在 Oracle Linux 上排程任務。	
排程資料遷移任務。	在目標主機上，使用 命令impdb來排程 Data Pump 匯入。這會將目標資料庫連接到現場部署主機，並啟動資料遷移。如需詳細資訊，請參閱 Data Pump Import 和 NETWORK_LINK (Oracle 文件）。	DBA
驗證資料遷移。	資料驗證是重要的步驟。對於資料驗證，您可以使用自訂工具或 Oracle 工具，例如 dblink 和 SQL 查詢的組合。	DBA

剪下

任務	Description	所需的技能
將來源資料庫置於唯讀模式。	確認應用程式已關閉，且未對來源資料庫進行任何變更。以唯讀模式開啟來源資料庫。這可協助您避免任何開啟的交易。如需詳細資訊，請參閱 SQL 陳述式ALTER DATABASE中的 (Oracle 文件）。	DBA、DevOps 工程師、應用程式開發人員
驗證物件計數和資料。	若要驗證資料和物件，請使用自訂工具或 Oracle 工具，例如 dblink 和 SQL 查詢的組合。	DBA，應用程式開發人員

任務	Description	所需的技能
將應用程式連接至主要 EC2 執行個體上的資料庫。	變更應用程式的連線屬性，以指向您在主要 EC2 執行個體上建立的新資料庫。	DBA，應用程式開發人員
驗證應用程式效能。	啟動應用程式。使用自動化工作負載儲存庫 (Oracle 文件) 驗證應用程式的功能和效能。	應用程式開發人員、DevOps 工程師、DBA

相關資源

AWS 參考

- 將 Oracle 資料庫遷移至 AWS 雲端
- Amazon EC2 for Oracle
- 將大量 Oracle 資料庫遷移至跨平台環境的 AWS
- VPCs和子網路
- 教學課程：建立 VPC 以與資料庫執行個體搭配使用

Oracle 參考

- Oracle Data Guard 組態
- 資料幫浦匯入

使用 AWS MGN 將 RHEL BYOL 系統遷移至包含 AWS 授權的執行個體

Mike Kuznetsov，Amazon Web Services

摘要

當您使用 AWS Application Migration Service (AWS MGN) 將工作負載遷移至 AWS 時，您可能需要在遷移期間將 Red Hat Enterprise Linux (RHEL) 執行個體提升和轉移 （恢復），並將授權從預設自帶授權 (BYOL) 模型變更為 AWS License Included (LI) 模型。AWS MGN 支援使用 Amazon Machine Image (AMI) IDs可擴展方法。此模式說明如何在大規模重新託管遷移期間，在 RHEL 伺服器上完成授權變更。它還說明如何變更已在 Amazon Elastic Compute Cloud (Amazon EC2) 上執行的 RHEL 系統授權。

先決條件和限制

先決條件

- 存取目標 AWS 帳戶
- 在目標 AWS 帳戶和區域中初始化的 AWS MGN 以進行遷移 （如果您已經從現場部署系統遷移至 AWS，則不需要）
- 具有有效 RHEL 授權的來源 RHEL 伺服器

Architecture

此模式涵蓋兩個案例：

- 使用 AWS MGN 將系統從內部部署直接遷移到 AWS LI 執行個體。在此案例中，請遵循第一個 epic (遷移至 LI 執行個體 - 選項 1) 和第三個 epic 中的指示。
- 針對已在 Amazon EC2 上執行的先前遷移 RHEL 系統，將授權模型從 BYOL 變更為 LI。在此案例中，請遵循第二個 epic (遷移至 LI 執行個體 - 選項 2) 和第三個 epic 中的指示。

> **Note**
>
> 第三個史詩涉及重新設定新的 RHEL 執行個體，以使用 AWS 提供的 Red Hat Update Infrastructure (RHUI) 伺服器。這兩個案例的此程序都相同。

工具

AWS 服務

- AWS Application Migration Service (AWS MGN) 可協助您將應用程式重新託管（提升和轉移）至 AWS 雲端，無需變更且停機時間最短。

史詩

遷移至 LI 執行個體 - 選項 1（適用於內部部署 RHEL 系統）

任務	Description	所需的技能
尋找目標區域中 RHEL AWS LI 執行個體的 AMI ID。	請造訪 AWS Marketplace 或使用 Amazon EC2 主控台尋找符合 RHEL 來源系統版本的 RHEL AMI ID（例如 RHEL-7.7），然後寫下 AMI ID。在 Amazon EC2 主控台上，您可以使用下列其中一個搜尋詞彙來篩選 AMIs： • 描述 = 由 Red Hat， Inc. 提供 • AMI 名稱 = RHEL-7.7	雲端管理員
設定 AWS MGN 啟動設定。	1. 在 AWS MGN 主控台上，依照 AWS MGN 文件中的指示，新增來源 RHEL 系統：安裝 AWS 複寫代理程式並新增來源伺服器。 2. 在來源伺服器頁面上，選擇來源 RHEL 系統，然後選擇啟動設定索引標籤。 3. 在一般啟動設定區段中，選擇編輯。若要停用自動選取並手動指定目標執行個體類型，請將執行個體類型	雲端管理員

任務	Description	所需的技能
	大小變更為無，然後選擇儲存設定。這可讓您使用在 Amazon EC2 啟動範本中設定的執行個體類型。如需詳細資訊，請參閱 AWS MGN 文件。 4. 在 EC2 啟動範本區段中，選擇修改。在關於修改 EC2 啟動範本對話方塊中，再次選擇修改。這會開啟 Amazon EC2 主控台，讓您可以變更此執行個體的範本。 5. 檢閱 AWS MGN 文件中的主要考量事項。 **Note** 您可以忽略警告，不要選擇您自己的 AMI。 6. 在 Amazon EC2 主控台的新啟動範本中，修改下列項目： • 對於 AMI，請指定您先前識別的 AMI ID，或搜尋 RHEL-x 並指定您需要的版本（例如 RHEL-7.7)。 • 針對執行個體類型，設定所需的目標執行個體類型。 • 將下列區段保持不變：金鑰對（登入）、網路設定（除非您想要指定目標	

任務	Description	所需的技能
	子網路和安全群組）、儲存、資源標籤（除非您想要新增或修改任何標籤）。 • （選用）在進階詳細資訊區段中，視需要指定 IAM 執行個體設定檔角色，以供 AWS Systems Manager 未來管理。 7. 選擇建立範本版本，然後選擇成功訊息中的連結以檢視啟動範本。 8. 選擇動作、設定預設版本。對於範本版本，選取最新版本（新系統的版本 2），然後選擇設定為預設版本。 AWS MGN 現在將使用此版本的啟動範本來啟動測試或切換執行個體。如需詳細資訊，請參閱 AWS MGN 文件。	

任務	Description	所需的技能
驗證設定。	1. 在 AWS MGN 主控台的來源伺服器頁面上，選擇來源伺服器，然後選擇啟動設定索引標籤。 2. 在 EC2 啟動範本區段中，確認執行個體類型、子網路和安全群組參數設定正確。 **Note** 本節不會顯示您選取的 AMI ID。若要查看 ID，您可以開啟 Amazon EC2 主控台、啟動範本檢視，並搜尋本節中顯示的範本 ID。	雲端管理員

任務	Description	所需的技能
啟動新的 LI 執行個體。	1. 當初始同步完成時，AWS MGN 主控台來源伺服器頁面上伺服器的遷移生命週期資料欄會變更為準備好進行測試。若要啟動新的測試執行個體，請選擇來源伺服器，開啟測試和切換功能表，然後選擇啟動測試執行個體。選擇檢視任務詳細資訊以監控啟動任務的狀態。如需詳細資訊，請參閱 AWS MGN 文件。 2. 等待啟動任務完成，然後開啟啟動的 EC2 執行個體詳細資訊頁面。選擇詳細資訊索引標籤，並確認執行個體詳細資訊區段包含下列項目： • 平台詳細資訊："Red Hat Enterprise Linux" • AMI 名稱：您在 EC2 啟動範本中指定的 AMI 名稱 3. 遵循 AWS MGN 文件中的指示，切換到新的 LI 執行個體。 4. 依照最後一個範例中的步驟，將新執行個體重新設定為使用 AWS 提供的 RHUI 伺服器。	雲端管理員

遷移至 LI 執行個體 - 選項 2（適用於 RHEL BYOL EC2 執行個體）

任務	Description	所需的技能
將 RHEL BYOL EC2 執行個體遷移至 AWS LI 執行個體。	您可以透過移動 RHEL 系統磁碟 (Amazon Elastic Block Store 磁碟區） 並將其連接至新的 LI 執行個體，將先前以 BYOL 形式遷移至 AWS 的 RHEL 系統切換至 AWS LI 執行個體。若要進行此切換，請依照下列步驟進行： 1. 從 RHEL LI AMI 啟動新的目標 RHEL 執行個體。請確定您選取的 AMI： • 使用與目前 RHEL 執行個體相同的 RHEL 版本。 • 具有與您目前的 RHEL 執行個體相同的開機程序 (BIOS 或 UEFI)。例如，如果來源伺服器是以 BIOS 為基礎，請使用也是以 BIOS 為基礎的 AWS Marketplace RHEL AMI；對於以 UEFI 為基礎的系統，請選擇以 UEFI 為基礎的 AMI。 2. 停止這兩個執行個體：新的 LI 執行個體和原始來源執行個體。 3. 從新的 LI 執行個體分離所有 EBS 磁碟區（包括根磁碟），並將其刪除。 4. 從舊來源執行個體分離所有 EBS 磁碟區（包括根磁	雲端管理員

任務	Description	所需的技能
	碟），並將其連接至新的 LI 執行個體。將磁碟區與裝置保持相同的映射。（例如，先前連接至/dev/sda磁碟機的 EBS 磁碟區必須/dev/sda連接至新的執行個體。） 5. 刪除來源（現在為無磁碟）執行個體。 6. 啟動新的 LI 執行個體。登入執行個體，然後依照下一個史詩中的步驟，將其重新設定為使用 AWS 提供的 RHUI 伺服器。	

重新設定 RHEL 作業系統以使用 AWS 提供的 RHUI – 這兩個選項

任務	Description	所需的技能
從 Red Hat 訂閱和授權取消註冊作業系統。	在遷移和成功切換後，必須從 Red Hat 訂閱中移除 RHEL 系統，以停止耗用 Red Hat 授權並避免重複計費。 若要從 Red Hat 訂閱中移除 RHEL 作業系統，請遵循 Red Hat 訂閱管理 (RHSM) 文件中所述的程序。使用 CLI 命令： `subscription-manager unregister` 您也可以停用訂閱管理員外掛程式，以停止檢查每次 yum	Linux 或系統管理員

任務	Description	所需的技能
	呼叫的訂閱狀態。若要這樣做，請編輯組態檔案並將 參數`/etc/yum/pluginconf.d/subscription-manager.conf` 變更為 `enabled=1` `enabled=0` 。	

任務	Description	所需的技能

使用 AWS 提供的 RHUI 取代舊的更新組態 (RHUI、Red Hat Satellite 網路、yum 儲存庫）。	您必須重新設定遷移的 RHEL 系統，才能使用 AWS 提供的 RHUI 伺服器。這可讓您存取 AWS 區域內的 RHUI 伺服器，而不需要外部更新基礎設施。變更包含下列程序：
	1. 備份現有的 yum 組態。
	2. 移除舊的 RHUI (yum 儲存庫） 組態和套件。
	3. 新增 AWS 提供的 RHUI 組態和憑證 套件。您必須從 AWS 上的另一個 RHEL 執行個體擷取這些組態套件，因為這些組態套件只能在 AWS 提供的 RHUI 伺服器上使用。
	
	以下是詳細的步驟和命令：
	1. 將所有 和 `/etc/pki/*` 資料夾複製到備份位置，以備份現有的 yum 組態`/etc/yum*` 和憑證。例如：
	`mkdir yum-backup`
	`cp -ra /etc/yum* /etc/pki ./yum-backup`
	`tar czf yum-backup.tgz ./yum-backup`
	2. 移除舊的 RHUI 組態和套件：
	a. 尋找所有已安裝的 RHUI 套件：
		Linux 或系統管理員

任務	Description	所需的技能

	`sudo rpm -qa | grep rhui`
	b. 刪除這些套件：
	`sudo yum remove $(rpm -qa | grep rhui)`
	c. 如果`/etc/yum/vars/releasever` 檔案存在，請將其移除。
	3. 新增 AWS 提供的 RHUI 和憑證套件。您必須從 AWS 上的另一個 RHEL 執行個體擷取這些執行個體。有幾種方式可以執行此作業。例如，您可以遵循 Red Hat 知識庫文章中提供的指示：
	a. 從 AWS Marketplace 啟動另一個 RHEL (RHEL-EC2) 執行個體。
	b. 從此執行個體下載兩個套件：最新的 RHUI 用戶端組態套件和憑證授權機構 (CA) 憑證。例如，從您的桌面執行此命令：
	`ssh RHEL-EC2 "sudo yumdownloader ca-certificates rh-amazon-rhui-client"`	

任務	Description	所需的技能

	c. 將套件從 RHEL-EC2 執行個體複製到新的遷移系統。例如：
	`scp RHEL-EC2:rh-amazon-rhui-client\* RHEL-EC2:ca-certificates\* . ssh <migrated-instance> "mkdir /tmp/amazon" scp rh-amazon-rhui-client* ca-certificates* <migrated-instance>:/tmp/amazon`
	d. 在遷移的執行個體上安裝新的 RHUI 和 CA 組態套件：
	`ssh <migrated-instance> "sudo rpm -Uhv /tmp/amazon/*"`	

驗證組態。	在目標遷移執行個體上，驗證新組態是否正確： `sudo yum clean all sudo yum repolist`	Linux 或系統管理員

相關資源

- AWS Application Migration Service (AWS MGN) 使用者指南
- 取得支援 IMDSv2 的 AWS RHUI 用戶端套件 (Red Hat 知識庫文章)
- Amazon EC2 啟動範本 (Amazon EC2 文件)

將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon EC2

Senthil Ramasamy，Amazon Web Services

總結

此模式說明如何將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的 Microsoft SQL Server。它涵蓋兩個遷移選項：使用 AWS Database Migration Service (AWS DMS) 或使用原生 Microsoft SQL Server 工具，例如備份和還原、複製資料庫精靈，或複製和連接資料庫。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- Amazon EC2 支援的作業系統（如需支援作業系統版本的完整清單，請參閱 Amazon EC2 FAQs)
- 內部部署資料中心中的 Microsoft SQL Server 來源資料庫

產品版本

- 對於內部部署和 Amazon EC2 執行個體資料庫， AWS DMS 支援：
  - SQL Server 2005、2008、2008R2、2012、2014、2016、2017 和 2019 版
  - 企業、標準、工作群組、開發人員和 Web 版本
- 如需支援版本的最新清單，請參閱使用 Microsoft SQL Server 資料庫做為目標 AWS DMS。

Architecture

來源技術堆疊

- 內部部署 Microsoft SQL Server 資料庫

目標技術堆疊

- EC2 執行個體上的 Microsoft SQL Server 資料庫

目標架構

資料遷移架構

- 使用 AWS DMS

- 使用原生 SQL Server 工具

工具

- AWS Database Migration Service (AWS DMS) 可協助您將資料遷移至廣泛使用的商業和開放原始碼資料庫，包括 Oracle、SQL Server、MySQL 和 PostgreSQL。您可以使用 AWS DMS 將資料遷移到 AWS 雲端、內部部署執行個體之間（透過 AWS 雲端 設定），或雲端和內部部署設定的組合之間。
- AWS Schema Conversion Tool (AWS SCT) 透過自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，支援異質資料庫遷移。
- 原生 Microsoft SQL Server 工具包括備份和還原、複製資料庫精靈，以及複製和連接資料庫。

史詩

規劃遷移

任務	Description	所需的技能
驗證來源和目標資料庫版本。		DBA
識別目標作業系統版本。		DBA，系統管理員
根據 Microsoft SQL Server 相容性清單和容量需求，識別目標伺服器執行個體的硬體需求。		DBA，系統管理員
識別 類型和容量的儲存需求。		DBA，系統管理員

任務	Description	所需的技能
識別網路需求，包括延遲和頻寬。		DBA，系統管理員
根據容量、儲存功能和網路功能選擇 EC2 執行個體類型。		DBA，系統管理員
識別來源和目標資料庫的網路和主機存取安全需求。		DBA，系統管理員
識別 Microsoft SQL Server 軟體安裝所需的使用者清單。		DBA，系統管理員
決定備份策略。		DBA
判斷可用性需求。		DBA
識別應用程式遷移和切換策略。		DBA，系統管理員

設定基礎設施

任務	Description	所需的技能
建立虛擬私有雲端 (VPC) 和子網路。		系統管理員
建立安全群組和網路存取控制清單 (ACL)。		系統管理員
設定和啟動 EC2 執行個體。		系統管理員

安裝軟體

任務	Description	所需的技能
建立 Microsoft SQL Server 軟體所需的使用者和群組。		DBA，系統管理員
下載 Microsoft SQL Server 軟體。		DBA，系統管理員
在 EC2 執行個體上安裝 Microsoft SQL Server 軟體並設定伺服器。		DBA，系統管理員

遷移資料 - 選項 1

任務	Description	所需的技能
使用原生 Microsoft SQL Server 工具或第三方工具來遷移資料庫物件和資料。	工具包括備份和還原、複製資料庫精靈，以及複製和連接資料庫。如需詳細資訊，請參閱將 Microsoft SQL Server 資料庫遷移至 AWS 雲端指南。	DBA

遷移資料 - 選項 2

任務	Description	所需的技能
使用 AWS DMS 遷移資料。	如需使用 的詳細資訊 AWS DMS，請參閱相關資源區段中的連結。	DBA

遷移應用程式

任務	Description	所需的技能
遵循應用程式遷移策略。	使用 AWS Schema Conversion Tool (AWS SCT) 來分析和修改內嵌在應用程式原始碼中的 SQL 程式碼。	DBA、應用程式擁有者

剪下

任務	Description	所需的技能
遵循應用程式切換策略。		DBA、應用程式擁有者、系統管理員

關閉專案

任務	Description	所需的技能
關閉所有臨時 AWS 資源。	暫時資源包括 的 AWS DMS 複寫執行個體和 EC2 執行個體 AWS SCT。	DBA，系統管理員
檢閱並驗證專案文件。		DBA、應用程式擁有者、系統管理員
收集遷移時間、手動與工具成本節省百分比等指標。		DBA、應用程式擁有者、系統管理員
關閉專案並提供意見回饋。		DBA、應用程式擁有者、系統管理員

相關資源

參考

- 將 Microsoft SQL Server 資料庫遷移至 AWS 雲端
- Amazon EC2
- Amazon EC2 FAQs
- Amazon EC2 定價
- AWS Database Migration Service
- 上的 Microsoft 產品 AWS
- 上的 Microsoft 授權 AWS
- 上的 Microsoft SQL Server AWS

教學課程和影片

- Amazon EC2 入門
- 開始使用 AWS Database Migration Service
- 將 Amazon EC2 執行個體加入您的 Simple AD Active Directory
- 將 Amazon EC2 執行個體加入您的 AWS Managed Microsoft AD Active Directory
- AWS Database Migration Service（影片）
- Amazon EC2 簡介 – Elastic Cloud Server & Hosting with AWS （影片）

在 AWS 雲端中重新託管內部部署工作負載：遷移檢查清單

Srikanth Rangavajhala，Amazon Web Services

總結

在 Amazon Web Services (AWS) 雲端中重新託管內部部署工作負載涉及下列遷移階段：規劃、探索前、探索、建置、測試和切換。此模式概述階段及其相關任務。這些任務是以高層級描述，並支援大約 75% 的所有應用程式工作負載。您可以在敏捷的衝刺週期中，在兩到三週內實作這些任務。

您應該與您的遷移團隊和顧問一起檢閱和審核這些任務。檢閱後，您可以收集輸入、視需要消除或重新評估任務，以符合您的需求，並修改其他任務以支援您產品組合中至少 75% 的應用程式工作負載。然後，您可以使用 Atlassian Jira 或 Rally Software 等敏捷的專案管理工具來匯入任務、將任務指派給資源，以及追蹤遷移活動。

模式假設您使用 AWS Cloud Migration Factory 來重新託管工作負載，但您可以使用您選擇的遷移工具。

Amazon Macie 可協助識別知識庫中的敏感資料，並將其儲存為 Amazon Simple Storage Service (Amazon S3) 儲存貯體中的資料來源、模型調用日誌和提示存放區。如需詳細資訊，請參閱 Macie 文件。

先決條件和限制

先決條件

- 用於追蹤遷移任務的專案管理工具（例如，Atlassian Jira 或 Rally Software)
- 在 AWS 上重新託管工作負載的遷移工具（例如，Cloud Migration Factory)

Architecture

來源平台

- 內部部署來源堆疊（包括技術、應用程式、資料庫和基礎設施）

目標平台

- AWS 雲端目標堆疊（包括技術、應用程式、資料庫和基礎設施）

架構

下圖說明使用 Cloud Migration Factory 和 AWS Application Migration Service 重新託管（從內部部署來源環境探索並將伺服器遷移至 AWS)。

工具

- 您可以使用您選擇的遷移和專案管理工具。

史詩

規劃階段

任務	Description	所需的技能
清理探索前待處理項目。	與部門主管和應用程式擁有者一起執行探索前待處理項目清理工作工作階段。	專案經理，敏捷 scrum 領導者
執行衝刺規劃工作工作階段。	作為範圍界定練習，分發您要在衝刺和波浪之間遷移的應用程式。	專案經理，敏捷 scrum 領導者

探索前階段

任務	Description	所需的技能
確認應用程式知識。	確認並記錄應用程式擁有者及其對應用程式的知識。判斷是否有另一個指標人員處理技術問題。	遷移專家（採訪者）
確定應用程式合規要求。	向應用程式擁有者確認應用程式不需要符合支付卡產業資料安全標準 (PCI DSS)、沙賓法案 (SOX)、個人身分識別資訊 (PII) 或其他標準的要求。如果存在合規要求，團隊必須在	遷移專家（採訪者）

任務	Description	所需的技能
	要遷移的伺服器上完成合規檢查。	
確認生產版本需求。	與應用程式擁有者或技術聯絡人確認將遷移的應用程式發佈至生產環境（例如發行日期和停機時間）的需求。	遷移專家（採訪者）
取得伺服器清單。	取得與目標應用程式相關聯的伺服器清單。	遷移專家（採訪者）
取得顯示目前狀態的邏輯圖表。	從企業架構師或應用程式擁有者取得應用程式的目前狀態圖表。	遷移專家（採訪者）
建立顯示目標狀態的邏輯圖表。	建立應用程式邏輯圖，顯示 AWS 上的目標架構。此圖表應說明伺服器、連線能力和映射因素。	企業架構師、企業擁有者
取得伺服器資訊。	收集與應用程式相關聯的伺服器相關資訊，包括其組態詳細資訊。	遷移專家（採訪者）
將伺服器資訊新增至探索範本。	將詳細的伺服器資訊新增至應用程式探索範本（請參閱此模式附件`mobilize-application-questionnaire.xlsx` 中的）。此範本包含所有應用程式相關的安全性、基礎設施、作業系統和聯網詳細資訊。	遷移專家（採訪者）
發佈應用程式探索範本。	與應用程式擁有者和遷移團隊共用應用程式探索範本，以供常見存取和使用。	遷移專家（採訪者）

探索階段

任務	Description	所需的技能
確認伺服器清單。	與應用程式擁有者或技術主管確認伺服器清單，以及每個伺服器的目的。	遷移專家
識別並新增伺服器群組。	識別伺服器群組，例如 Web 伺服器或應用程式伺服器，並將此資訊新增至應用程式探索範本。選取每個伺服器應所屬的應用程式層 (Web、應用程式、資料庫）。	遷移專家
填寫應用程式探索範本。	在遷移團隊、應用程式團隊和 AWS 的協助下，完成應用程式探索範本的詳細資訊。	遷移專家
新增遺失的伺服器詳細資訊（中介軟體和作業系統團隊）。	要求中介軟體和作業系統 (OS) 團隊檢閱應用程式探索範本，並新增任何遺失的伺服器詳細資訊，包括資料庫資訊。	遷移專家
取得傳入/傳出流量規則（網路團隊）。	要求網路團隊取得來源和目的地伺服器的傳入/傳出流量規則。網路團隊也應新增現有的防火牆規則、將這些規則匯出為安全群組格式，並將現有的負載平衡器新增至應用程式探索範本。	遷移專家
識別必要的標記。	判斷應用程式的標記需求。	遷移專家
建立防火牆請求詳細資訊。	擷取和篩選與應用程式通訊所需的防火牆規則。	遷移專家、解決方案架構師、網路領導
更新 EC2 執行個體類型。	根據基礎設施和伺服器需求，更新要在目標環境中使用的	遷移專家、解決方案架構師、網路領導

任務	Description	所需的技能
	Amazon Elastic Compute Cloud (Amazon EC2) 執行個體類型。	
識別目前的狀態圖表。	識別或建立顯示應用程式目前狀態的圖表。此圖表將用於資訊安全 (InfoSec) 請求。	解決方案架構師遷移專家
完成未來狀態圖表。	完成顯示應用程式未來（目標）狀態的圖表。此圖表也會用於 InfoSec 請求。	解決方案架構師遷移專家
建立防火牆或安全群組服務請求。	建立防火牆或安全群組服務請求（用於開發/QA、生產前和生產）。如果您使用的是 Cloud Migration Factory，請包含尚未開啟的複寫特定連接埠。	遷移專家、解決方案架構師、網路領導
檢閱防火牆或安全群組請求 (InfoSec 團隊）。	在此步驟中，InfoSec 團隊會檢閱並核准上一個步驟中建立的防火牆或安全群組請求。	InfoSec 工程師，遷移專家
實作防火牆安全群組請求（網路團隊）。	InfoSec 團隊核准防火牆請求後，網路團隊會實作必要的傳入/傳出防火牆規則。	遷移專家、解決方案架構師、網路領導

建置階段（重複用於開發/QA、生產前和生產環境）

任務	Description	所需的技能
匯入應用程式和伺服器資料。	1. 確認您以網域使用者身分登入遷移執行伺服器，並在範圍內來源伺服器上具有本機管理員許可。	遷移專家，雲端管理員

任務	Description	所需的技能
	2. 使用遷移接收表單來匯入範圍內來源伺服器的屬性。如需詳細資訊，請參閱 Cloud Migration Factory 實作指南。 如果您不是使用 Cloud Migration Factory，請遵循設定遷移工具的指示。	
檢查來源伺服器的先決條件。	與範圍內來源伺服器連線，以驗證先決條件，例如 TCP 連接埠 1500、TCP 連接埠 443、根磁碟區可用空間、.NET Framework 版本和其他參數。這些是複寫的必要項目。如需詳細資訊，請參閱 Cloud Migration Factory 實作指南。	遷移專家，雲端管理員
建立服務請求以安裝複寫代理程式。	建立服務請求，以在範圍內伺服器上安裝複寫代理程式，以進行開發/QA、生產前或生產。	遷移專家，雲端管理員
安裝複寫代理程式。	在開發/QA、生產前或生產機器上的範圍內來源伺服器上安裝複寫代理程式。如需詳細資訊，請參閱 Cloud Migration Factory 實作指南。	遷移專家，雲端管理員

任務	Description	所需的技能
推送啟動後指令碼。	Application Migration Service 支援啟動後指令碼，協助您自動化作業系統層級活動，例如在啟動目標執行個體後安裝或解除安裝軟體。此步驟會將啟動後指令碼推送至 Windows 或 Linux 機器，具體取決於為遷移而識別的伺服器。如需說明，請參閱 Cloud Migration Factory 實作指南。	遷移專家，雲端管理員
驗證複寫狀態。	使用提供的指令碼，自動確認範圍內來源伺服器的複寫狀態。指令碼每五分鐘重複一次，直到指定波動中所有來源伺服器的狀態變更為正常為止。如需說明，請參閱 Cloud Migration Factory 實作指南。	遷移專家，雲端管理員
建立管理員使用者。	在從範圍內來源伺服器遷移到 AWS 之後，可能需要來源機器上的本機管理員或 sudo 使用者來疑難排解任何問題。當身分驗證伺服器（例如 DC 或 LDAP 伺服器）無法連線時，遷移團隊會使用此使用者登入目標伺服器。如需此步驟的說明，請參閱 雲端遷移工廠實作指南。	遷移專家，雲端管理員
驗證啟動範本。	驗證伺服器中繼資料，以確保其成功運作且沒有無效資料。此步驟會驗證測試和切換中繼資料。如需說明，請參閱Cloud Migration Factory 實作指南。	遷移專家，雲端管理員

測試階段（重複用於開發/QA、生產前和生產環境）

任務	Description	所需的技能
建立服務請求。	為基礎設施團隊和其他團隊建立服務請求，以執行應用程式切換到開發/QA、生產前或生產執行個體。	遷移專家，雲端管理員
設定負載平衡器（選用）。	設定必要的負載平衡器，例如 Application Load Balancer 或具有 iRules 的 F5 負載平衡器。	遷移專家，雲端管理員
啟動執行個體進行測試。	在測試模式下，在 Application Migration Service 中啟動指定波的所有目標機器。如需詳細資訊，請參閱Cloud Migration Factory 實作指南。	遷移專家，雲端管理員
驗證目標執行個體狀態。	檢查相同波次中所有範圍內來源伺服器的開機程序，以驗證目標執行個體的狀態。最多可能需要 30 分鐘才能啟動目標執行個體。您可以手動檢查狀態，方法是登入 Amazon EC2 主控台、搜尋來源伺服器名稱，以及檢閱狀態檢查欄。通過的狀態 2/2 檢查表示從基礎設施的角度來看，執行個體運作狀態良好。	遷移專家，雲端管理員
修改 DNS 項目。	修改網域名稱系統 (DNS) 項目。(`host.conf` 針對 Microsoft Windows 環境使用 `resolv.conf` 或 。) 設定每個 EC2 執行個體以指向此主機的新 IP 地址。	遷移專家，雲端管理員

任務	Description	所需的技能
	Note 確定現場部署和 AWS 雲端伺服器之間沒有 DNS 衝突。此步驟和下列步驟是選用的，取決於託管伺服器的環境。	
從 EC2 執行個體測試後端主機的連線能力。	使用遷移伺服器的網域登入資料來檢查登入。	遷移專家，雲端管理員
更新 DNS A 記錄。	更新每個主機的 DNS A 記錄，以指向新的 Amazon EC2 私有 IP 地址。	遷移專家，雲端管理員
更新 DNS CNAME 記錄。	更新虛擬 IPs（負載平衡器名稱）的 DNS CNAME 記錄，以指向 Web 和應用程式伺服器的叢集。	遷移專家，雲端管理員
在適用的環境中測試應用程式。	登入新的 EC2 執行個體，並在開發/QA、生產前和生產環境中測試應用程式。	遷移專家，雲端管理員
標記為準備好進行切換。	測試完成時，請變更來源伺服器的狀態以表示其已準備好進行切換，讓使用者可以啟動切換執行個體。如需說明，請參閱 雲端遷移工廠實作指南。	遷移專家，雲端管理員

切換階段

任務	Description	所需的技能
建立生產部署計畫。	建立生產部署計畫（包括備份計畫）。	遷移專家，雲端管理員
將停機時間通知營運團隊。	通知 操作團隊伺服器的停機時間排程。有些團隊可能需要此通知的變更請求或服務請求 (CR/SR) 票證。	遷移專家，雲端管理員
複寫生產機器。	使用 Application Migration Service 或其他遷移工具複寫生產機器。	遷移專家，雲端管理員
關閉範圍內來源伺服器。	驗證來源伺服器的複寫狀態後，您可以關閉來源伺服器，以停止從用戶端應用程式到伺服器的交易。您可以在切換視窗中關閉來源伺服器。如需詳細資訊，請參閱 雲端遷移工廠實作指南。	雲端管理員
啟動切換的執行個體。	在 Application Migration Service 中以切換模式啟動指定波的所有目標機器。如需詳細資訊，請參閱 雲端遷移工廠實作指南。	遷移專家，雲端管理員
擷取目標執行個體 IPs。	擷取目標執行個體IPs。如果 DNS 更新是環境中的手動程序，您將需要取得所有目標執行個體的新 IP 地址。如需詳細資訊，請參閱 雲端遷移工廠實作指南。	遷移專家，雲端管理員

任務	Description	所需的技能
驗證目標伺服器連線。	更新 DNS 記錄後，請使用主機名稱連線到目標執行個體，以驗證連線。如需詳細資訊，請參閱 雲端遷移工廠實作指南。	遷移專家，雲端管理員

相關資源

- 如何遷移
- AWS Cloud Migration Factory 實作指南
- 使用 Cloud Migration Factory 自動化大規模伺服器遷移
- AWS Application Migration Service 使用者指南
- Migration Acceleration Program (MAP)

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Amazon FSx 設定 SQL Server Always On FCI 的異地同步備份基礎設施

Manish Garg、T.V.R.L.Phani Kumar Dadi、Nishad Mankar 和 RAJNEESH TYAGI，Amazon Web Services

總結

如果您需要快速遷移大量 Microsoft SQL Server Always On 容錯移轉叢集執行個體 (FCIs)，此模式可協助您將佈建時間降至最低。透過使用自動化和 Amazon FSx for Windows File Server，可減少手動作業、人為錯誤，以及部署大量叢集所需的時間。

此模式會在 Amazon Web Services (AWS) 的多可用區域（多可用區域）部署中設定 SQL Server FCIs 的基礎設施。此基礎設施所需的 AWS 服務佈建是使用 AWS CloudFormation 範本自動化的。Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的 SQL Server 安裝和叢集節點建立是使用 PowerShell 命令執行。

此解決方案使用高度可用的多可用區域 Amazon FSx for Windows 檔案系統做為存放 SQL Server 資料庫檔案的共用見證。託管 SQL Server 的 Amazon FSx 檔案系統和 EC2 Windows 執行個體會加入相同的 AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD) 網域。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 具有足夠許可的 AWS 使用者，可使用 AWS CloudFormation 範本佈建資源
- 適用於 Microsoft Active Directory 的 AWS Directory Service
- AWS Secrets Manager 中要驗證金鑰/值對中 AWS Managed Microsoft AD 的登入資料：
  - `ADDomainName`：<網域名稱>
  - `ADDomainJoinUserName`：<Domain Username>
  - `ADDomainJoinPassword`：<網域使用者密碼>
  - `TargetOU`：<Target OU 值>

> **Note**
>
> 您將在 AWS Managed Microsoft AD 聯結活動的 AWS Systems Manager 自動化中使用相同的金鑰名稱。

- SQL Server 安裝和 Windows 服務或網域帳戶建立的 SQL Server 媒體檔案，將在叢集建立期間使用
- 虛擬私有雲端 (VPC)，在個別可用區域中有兩個公有子網路、可用區域中有兩個私有子網路、網際網路閘道、NAT 閘道、路由表關聯和跳接伺服器

產品版本

- Windows Server 2012 R2 和 Microsoft SQL Server 2016

Architecture

來源技術堆疊

- 使用共用磁碟機搭配 FCIs 的內部部署 SQL Server

目標技術堆疊

- AWS EC2 執行個體
- Amazon FSx for Windows File Server
- AWS Systems Manager Automation Runbook
- 網路組態 (VPC、子網路、網際網路閘道、NAT 閘道、跳躍伺服器、安全群組)
- AWS Secrets Manager
- AWS 受管 Microsoft AD
- Amazon EventBridge
- AWS Identity and Access Management (IAM)

目標架構

下圖顯示單一 AWS 區域中的 AWS 帳戶，其中包含兩個可用區域的 VPC、兩個具有 NAT 閘道的公有子網路、第一個公有子網路中的跳轉伺服器、兩個私有子網路，每個子網路都有節點安全群組中 SQL Server 節點的 EC2 執行個體，以及連線至每個 SQL Server 節點的 Amazon FSx 檔案系統。也包含 AWS Directory Service、Amazon EventBridge、AWS Secrets Manager 和 AWS Systems Manager。

自動化和擴展

- 您可以使用 AWS Systems Manager 加入 AWS Managed Microsoft AD 並執行 SQL Server 安裝。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速一致地佈建資源，以及在整個 AWS 帳戶和區域的生命週期中管理這些資源。
- AWS Directory Service 提供多種方式來搭配其他 AWS 服務使用 Microsoft Active Directory (AD)，例如 Amazon Elastic Compute Cloud (Amazon EC2)、Amazon Relational Database Service (Amazon RDS) for SQL Server 和 Amazon FSx for Windows File Server。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如，AWS Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他 AWS 帳戶中的事件匯流排。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Secrets Manager 可協助您以 API 呼叫 Secrets Manager，以程式設計方式擷取秘密，取代程式碼中的硬式編碼登入資料，包括密碼。
- AWS Systems Manager 可協助您管理在 AWS 雲端中執行的應用程式和基礎設施。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。

其他工具

- PowerShell 是在 Windows、Linux 和 macOS 上執行的 Microsoft 自動化和組態管理程式。此模式使用 PowerShell 指令碼。

程式碼儲存庫

此模式的程式碼可在 GitHub aws-windows-failover-cluster-automation 儲存庫中使用。

最佳實務

- 用於部署此解決方案的 IAM 角色應遵循最低權限原則。如需詳細資訊，請參閱 IAM 文件。
- 遵循 AWS CloudFormation 最佳實務。

史詩

部署基礎設施

任務	Description	所需的技能
部署 Systems Manager CloudFormation 堆疊。	1. 登入您的 AWS 帳戶，然後開啟 AWS 管理主控台。 2. 導覽至 CloudFormation 主控台，並透過上傳 `ssm.yaml`範本來建立 Systems Manager CloudFormation 堆疊。提供下列參數的值： • StateUnJoinAssociationLoggingBucketName – 提供範本為記錄目的而建立的 S3 儲存貯體名稱。 • SSMAssociationADUnjoinName – 提供`AWS::SSM::Association` 資源的名稱。 • SSMAutomationDocumentName – 提供 Systems Manager Automation Runbook 的名稱。 • EventBridgeName – 提供 EventBridge 事件匯流排的名稱。 3. 啟動 CloudFormation 範本以部署 Systems Manager `ssm.yaml` CloudFormationstack。範本會建立 Systems Manager	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	Atomation runboook，並在具有 標籤的新 EC2 執行個體`ADJoined: FSXADD`啟動時啟動。Automation Runbook 會將執行個體新增至 AWS Managed Microsoft AD 目錄。	

任務	Description	所需的技能
部署基礎設施堆疊。	成功部署 Systems Manager 堆疊後，請建立`infra`堆疊，其中包含 EC2 執行個體節點、安全群組、Amazon FSx for Windows File Server 檔案系統和 IAM 角色。 1. 導覽至 CloudFormation 主控台並啟動`infra-cf.yaml` 範本。若要部署此堆疊，需要下列參數： • `ActiveDirectoryId` – AWS Managed Microsoft AD 的 ID • `ADDnsIpAddresses1` – AWS Managed Microsoft AD 的主要 DNS IP 地址 • `ADDnsIpAddresses2` – AWS Managed Microsoft AD 的次要 DNS IP 地址 • `FSxSecurityGroupName` – Amazon FSx 安全群組的名稱 • `FSxWindowsFileSystemName` – Amazon FSx 磁碟機的名稱 • `ImageID` – 用來建立 SQL Server 執行個體節點的基本 Windows 2012 R2 映像或 Amazon Machine Image (AMI) 的 ID	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	• KeyPairName – 要連接到 EC2 執行個體節點以進行存取的金鑰值對 • Node1SecurityGroupName – 第一個節點安全群組的名稱 • Node2SecurityGroupName – 第二個節點安全群組的名稱 • OUSecretName – 包含 AWS Managed Microsoft AD 資訊的秘密名稱 • PrivateSubnet1 – 第一個私有子網路的 ID • PrivateSubnet2 – 第二個私有子網路的 ID • SqlFSxFCIName – 套用至主要和次要節點和 Amazon FSx 的標籤名稱。 • SqlFSxServerNetBIOSName1 – 主要 EC2 執行個體節點的名稱（最多 15 個字元） • SqlFSxServerNetBIOSName2 – 次要 EC2 執行個體節點的名稱（最多 15 個字元） • VPC – VPC ID • WorkloadInstanceType – EC2 執行個體的類型	

任務	Description	所需的技能
	部署infra堆疊。堆疊會建立設定 Windows SQL Server FCI 所需的所有基礎設施元件。 2. 啟動 EC2 執行個體節點之後，會叫用 Systems Manager 自動化文件，將這些執行個體加入 AWS Managed Microsoft AD。您可以在 Systems Manager 主控台自動化頁面上追蹤進度。	

設定 Windows SQL Server Always On FCI

任務	Description	所需的技能

安裝 Windows 工具。	1. 登入主要 EC2 執行個體，即節點 1。若要安裝 Windows 功能 (Active Directory 和 FCI 工具），請執行下列 PowerShell 指令碼。
	`Install-WindowsFeature -Name RSAT-AD-Powershell,Failover-Clustering -IncludeManagementTools`
	`Install-WindowsFeature -Name RSAT-Clustering,RSAT-ADDS-Tools,RSAT-AD-Powershell,RSAT-DHCP,RSAT-DNS-Server`
		AWS DevOps、DevOps 工程師、DBA

任務	Description	所需的技能
	2. 登入次要 EC2 執行個體，即節點 2，並執行相同的指令碼以啟用節點 2 上的功能。	
在 Active Directory Domain Services 中預先準備叢集電腦物件。	若要預先準備 Active Directory Domain Services (AD DS) 中的叢集名稱物件 (CNO)，並預先準備叢集角色的虛擬電腦物件 (VCO)，請遵循 Windows Server 文件中的指示。	AWS DevOps、DBA、DevOps 工程師

任務	Description	所需的技能
建立 WSFC。	若要建立 Windows Server 容錯移轉叢集 (WSFC) 叢集，請執行下列動作： 1. 登入主要 EC2 執行個體，即節點 1。若要建立 Amazon FSx 檔案共享並授予所列 AD 服務帳戶的完整存取權，請執行下列程式碼。 `Invoke-Command -ComputerName "<FSx Windows Remote PowerShell Endpoint>" -ConfigurationName FSxRemoteAdmin -scriptblock { New-FSxSmbShare -Name "SQLDB" -Path "D:\share" -Description "SQL Databases Share" -ContinuouslyAvailable $true -FolderEnumerationMode AccessBased -EncryptData $true grant-fsxsmbshareaccess -name SQLDB -AccountName "<domain\user>" -accessRight Full }` 此命令也會建立持續可用的 (CA) 檔案共用，並針對 Microsoft SQL Server 的使用進行最佳化。	AWS DevOps、DBA、DevOps 工程師

任務	Description	所需的技能
	2. 若要在主要執行個體 （節點 1) 上建立容錯移轉叢集，請執行下列命令。 `New-Cluster -Name <CNO Name> -Node <Node1 Name>, <Node2 Name> -StaticAddress <Node1 Secondary Private IP>, <Node2 Secondary Private IP>` 命令需要下列參數： • Name – 叢集的名稱 (CNO) • Node – 主要和次要節點的名稱 • StaticAddress – 主要和次要節點的次要 IP 地址，分別是 **Important** 網域管理員或一般使用者必須擁有兩個節點的管理員許可，才能建立 Windows Server 容錯移轉叢集 (WSFC) 叢集。否則，上一個命令將會失敗並傳回訊息 `You do not have administrator privilege on servers`。	

任務	Description	所需的技能
	3. 建立叢集之後，請執行下列命令來連接檔案共享見證。 `Set-ClusterQuorum -FileShareWitness \ \<FSx Windows Remote PowerShell Endpoint> \share\witness`	

任務	Description	所需的技能
安裝 SQL Server 容錯移轉叢集。	設定 WSFC 叢集後，在主要執行個體 (node1) 上安裝 SQL Server 叢集。 1. 在兩個節點上的 T 磁碟機中，建立 `tempdb` 和 `log` 資料夾。這些資料夾用於 PowerShell 命令。 2. 在兩個節點上複製 SQL Server 安裝的 SQL Server 媒體檔案之後，請在節點 1 上執行下列 PowerShell 命令，以在節點 1 上安裝 SQL Server。 `D:\setup.exe /Q ` ` /ACTION=InstallFailoverCluster ` /IACCEPTSQLSERVERLICENSETERMS ` /FEATURES="SQL,IS,BC,Conn" ` /INSTALLSHAREDDIR="C:\Program Files\Microsoft SQL Server" ` /INSTALLSHAREDWOWDIR="C:\Program Files (x86)\Microsoft SQL Server" ` /RSINSTALLMODE="FilesOnlyMode" ` /INSTANCEID="MSSQLSERVER" ` /INSTANCENAME="MSSQLSERVER" ` /FAILOVERCLUSTERGROUP="SQL Server (MSSQLSERVER)" `	AWS DevOps、DBA、DevOps 工程師

任務	Description	所需的技能
	/FAILOVERCLUSTER IPADDRESSES="IPv4; <2nd Sec Private Ip node1>;Cluster Network 1;<subnet mask>" ` /FAILOVERCLUSTER NETWORKNAME="<Fail over cluster Network Name>" ` /INSTANCEDIR="C: \Program Files\Mic rosoft SQL Server" ` /ENU="True" ` /ERRORREPORTING=0 ` /SQMREPORTING=0 ` /SAPWD="<Domain User password>" ` /SQLCOLLATION="S QL_Latin1_General_ CP1_CI_AS" ` /SQLSYSADMINACCO UNTS="<domain\user name>" ` /SQLSVCACCOUNT=" <domain\username>" /SQLSVCPASSWORD="< Domain User password>" ` /AGTSVCACCOUNT=" <domain\username>" /AGTSVCPASSWORD="< Domain User password>" ` /ISSVCACCOUNT="<domain \username>" /ISSVCPAS SWORD="<Domain User password>" ` /FTSVCACCOUNT="NT Service\MSSQLFDLau ncher" ` /INSTALLSQLDATADIR="\ \<FSX DNS name>\sha	

任務	Description	所需的技能
	re\Program Files\Mic rosoft SQL Server" ` /SQLUSERDBDIR="\\<FSX DNS name>\share\data" ` /SQLUSERDBLOGDIR="\ \<FSX DNS name>\share \log" ` /SQLTEMPDBDIR="T: \tempdb" ` /SQLTEMPDBLOGDIR="T: \log" ` /SQLBACKUPDIR="\\<FSX DNS name>\share\SQLBac kup" ` /SkipRules=Clust er_VerifyForErrors ` /INDICATEPROGRESS	

任務	Description	所需的技能
將次要節點新增至叢集。	若要將 SQL Server 新增至次要節點（節點 2），請執行下列 PowerShell 命令。 `D:\setup.exe /Q ` `/ACTION=AddNode ` `/IACCEPTSQLSERVERLICENSETERMS ` `/INSTANCENAME="MSSQLSERVER" ` `/FAILOVERCLUSTERGROUP="SQL Server (MSSQLSERVER)" ` `/FAILOVERCLUSTERIPADDRESSES="IPv4;<2nd Sec Private Ip node2>;Cluster Network 2;<subnet mask>" ` `/FAILOVERCLUSTERNETWORKNAME="<Fail over cluster Network Name>" ` `/CONFIRMIPDEPENDENCYCHANGE=1 ` `/SQLSVCACCOUNT="<domain\username>" /SQLSVCPASSWORD="<Domain User password>" ` `/AGTSVCACCOUNT="domain\username>" /AGTSVCPASSWORD="<Domain User password>" ` `/FTSVCACCOUNT="NT Service\MSSQLFDLauncher" ` `/SkipRules=Cluster_VerifyForErrors ` `/INDICATEPROGRESS`	AWS DevOps、DBA、DevOps 工程師

任務	Description	所需的技能
測試 SQL Server FCI。	1. 在其中一個節點的 Windows 執行個體上，在管理工具中啟動容錯移轉叢集管理員。 2. 導覽至節點，並確認節點狀態為執行中狀態。 3. 選取角色，開啟 SQL Server (MSSQLSERVER) 的內容（按一下滑鼠右鍵）選單，然後選取移動和選取節點。 4. 選取節點之後，SQL Server 應該在其他節點上執行。	DBA，DevOps 工程師

清除資源

任務	Description	所需的技能
清除資源。	若要清除資源，請使用 AWS CloudFormation 堆疊刪除程序： 1. 開啟 AWS CloudFormation 主控台。 2. 在堆疊頁面上，選取`infra`堆疊。此堆疊目前必須正在執行。 3. 在 stack details (堆疊詳細資訊) 窗格中，選擇 Delete (刪除)。 4. 當系統提示時，選取 Delete stack (刪除堆疊)。 5. 針對`ssm`堆疊重複步驟 2-4。	AWS DevOps、DBA、DevOps 工程師

任務	Description	所需的技能
	堆疊刪除完成後，堆疊將處於 DELETE_COMPLETE 狀態。根據預設，DELETE_COMPLETE 處於狀態的堆疊不會顯示在 CloudFormation 主控台中。若要顯示已刪除的堆疊，您必須變更堆疊檢視篩選條件，如在 AWS CloudFormation 主控台上檢視已刪除的堆疊所述。 如果刪除失敗，堆疊將處於 DELETE_FAILED 狀態。如需解決方案，請參閱 CloudFormation 文件中的刪除堆疊失敗。	

疑難排解

問題	解決方案
AWS CloudFormation 範本失敗	如果 CloudFormation 範本在部署期間失敗，請執行下列動作： 1. 開啟 AWS CloudFormation 主控台。 2. 在 CloudFormation 主控台的堆疊頁面上，選取堆疊。 3. 選擇事件，並檢查堆疊狀態。
AWS Managed Microsoft AD 聯結失敗	若要疑難排解聯結問題，請依照下列步驟進行： 1. 開啟 Systems Manager 主控台。 2. 選取部署區域。 3. 在左側窗格中，選擇自動化，然後尋找失敗的 Automation Runbook。

問題	解決方案
	4. 開啟 Automation Runbook，並檢查執行狀態和執行步驟。 5. 調查失敗步驟的詳細資訊，以查看確切的錯誤或失敗。

相關資源

- 使用 Amazon FSx for Windows File Server 簡化 Microsoft SQL Server 高可用性部署
- 將 FSx for Windows File Server 與 Microsoft SQL Server 搭配使用

使用 BMC Discovery 查詢來擷取遷移資料以進行遷移規劃

Ben Tailor-Hamblin、Emma Baldry、Simon Cunningham 和 Shabnam Khan，Amazon Web Services

總結

本指南提供查詢範例和步驟，協助您使用 BMC Discovery 從內部部署基礎設施和應用程式擷取資料。模式說明如何使用 BMC Discovery 查詢來掃描您的基礎設施，並擷取軟體、服務和相依性資訊。評估和調動大規模遷移至 Amazon Web Services (AWS) 雲端的階段需要擷取的資料。您可以使用此資料，針對遷移計畫中要一起遷移的應用程式做出關鍵決策。

先決條件和限制

先決條件

- BMC Helix Discovery 的 BMC Discovery （ 先前稱為 BMC ADDM) 授權或軟體即服務 (SaaS) 版本
- 現場部署或 SaaS 版本的 BMC Discovery，已安裝

> **Note**
>
> 對於內部部署版本的 BMC Discovery，您必須在用戶端網路上安裝應用程式，該用戶端網路可存取跨多個資料中心遷移範圍內的所有聯網和伺服器裝置。必須根據應用程式安裝指示提供用戶端網路的存取權。如果需要掃描 Windows Server 資訊，則必須在網路中設定 Windows Proxy Manager 裝置。

- 如果您使用 BMC Helix Discovery，允許應用程式跨資料中心掃描裝置的聯網存取

產品版本

- BMC Discovery 22.2 (12.5)
- BMC Discovery 22.1 (12.4)
- BMC Discovery 21.3 (12.3)
- BMC Discovery 21.05 (12.2)
- BMC Discovery 20.08 (12.1)
- BMC Discovery 20.02 (12.0)
- BMC Discovery 11.3
- BMC Discovery 11.2
- BMC Discovery 11.1

- BMC Discovery 11.0
- BMC Atrium Discovery 10.2
- BMC Atrium Discovery 10.1
- BMC Atrium Discovery 10.0

Architecture

下圖顯示資產管理員如何使用 BMC Discovery 查詢來掃描 SaaS 和內部部署環境中的 BMC 模型應用程式。

下圖顯示下列工作流程：資產管理員使用「BMC Discovery」或「BMC Helix Discovery」掃描在多個實體伺服器上託管的虛擬伺服器上執行的資料庫和軟體執行個體。此工具可以使用橫跨多個虛擬和實體伺服器的元件來建立應用程式模型。

技術堆疊

- BMC 探索
- BMC Helix 探索

工具

- BMC Discovery 是一種資料中心探索工具，可協助您自動探索資料中心。
- BMC Helix Discovery 是一種以 SaaS 為基礎的探索和相依性建模系統，可協助您動態建立資料資產及其相依性的模型。

最佳實務

當您遷移至雲端時，最佳實務是映射應用程式、相依性和基礎設施資料。映射可協助您了解目前環境的複雜性，以及各種元件之間的相依性。

這些查詢提供的資產資訊很重要，原因有幾個：

1. 規劃 – 了解元件之間的相依性可協助您更有效地規劃遷移程序。例如，您可能需要先遷移某些元件，以確保其他元件可以成功遷移。
2. 風險評估 – 映射元件之間的相依性可協助您識別遷移過程中可能出現的任何潛在風險或問題。例如，您可能會發現某些元件依賴過時或不支援的技術，這些技術可能會導致雲端出現問題。

3. 雲端架構 – 映射您的應用程式和基礎設施資料也可以協助您設計符合您組織需求的合適雲端架構。例如，您可能需要設計多層架構，以支援高可用性或可擴展性需求。

整體而言，映射應用程式、相依性和基礎設施資料是雲端遷移程序中的關鍵步驟。映射練習可協助您更了解目前的環境、識別任何潛在問題或風險，以及設計適當的雲端架構。

史詩

識別和評估探索工具

任務	Description	所需的技能
識別 ITSM 擁有者。	識別 IT Service Management (ITSM) 擁有者（通常是透過聯絡營運支援團隊）。	遷移潛在客戶
檢查 CMDB。	識別包含資產資訊的組態管理資料庫 (CMDBs) 數目，然後識別該資訊的來源。	遷移潛在客戶
識別探索工具並檢查是否使用 BMC Discovery。	如果您的組織使用 BMC Discovery 將環境相關資料傳送至 CMDB 工具，請檢查其掃描的範圍和涵蓋範圍。例如，檢查 BMC Discovery 是否正在掃描所有資料中心，以及存取伺服器是否位於周邊區域。	遷移潛在客戶
檢查應用程式建模的層級。	檢查應用程式是否在 BMC Discovery 中建模。如果沒有，建議使用 BMC Discovery 工具來建立哪些執行中的軟體執行個體提供應用程式和商業服務的模型。	遷移工程師，遷移負責人

擷取基礎設施資料

任務	Description	所需的技能
擷取實體和虛擬伺服器上的資料。	若要擷取由 BMC Discovery 掃描的實體和虛擬伺服器上的資料，請使用 Query Builder 執行下列查詢： `search Host show key as 'Serverid', virtual, name as 'HOSTNAME', os_type as 'osName', os_version as 'OS Version', num_logical_processors as 'Logical Processor Counts', cores_per_processor as 'Cores per Processor', logical_ram as 'Logical RAM', #Consumer:StorageUse:Provider:DiskDrive.size as 'Size'` **Note** 您可以使用擷取的資料來判斷適當的執行個體大小以進行遷移。	遷移工程師，遷移負責人
擷取建模應用程式上的資料。	如果您的應用程式在 BMC Discovery 中建立模型，您可以擷取執行應用程式軟體之伺服器的資料。若要取得伺服器名稱，請使用查詢建置器來執行下列查詢：	BMC Discovery 應用程式擁有者

任務	Description	所需的技能
	`search SoftwareInstance show key as 'ApplicationID', #RunningSoftware:HostedSoftware:Host:Host.key as 'ReferenceID', type, name` **Note** 應用程式透過執行中軟體執行個體的集合在 BMC Discovery 中建模。應用程式取決於執行應用程式軟體的所有伺服器。	

任務	Description	所需的技能
擷取資料庫上的資料。	若要取得所有掃描資料庫的清單，以及這些資料庫正在執行的伺服器，請使用 Query Builder 執行下列查詢： `search Database show key as 'Key', name, type as 'Source Engine Type', #Detail:Detail:ElementWithDetail:SoftwareInstance.name as 'Software Instance', #Detail:Detail:ElementWithDetail:SoftwareInstance.product_version as 'Product Version', #Detail:Detail:ElementWithDetail:SoftwareInstance.edition as 'Edition', #Detail:Detail:ElementWithDetail:SoftwareInstance.#RunningSoftware:HostedSoftware:Host:Host.key as 'ServerID'`	應用程式擁有者

任務	Description	所需的技能
在伺服器通訊上擷取資料。	若要從歷史網路通訊日誌取得由 BMC Discovery 收集之伺服器間所有網路通訊的資訊，請使用查詢建置器來執行下列查詢： `search Host TRAVERSE InferredElement:Inference:Associate:DiscoveryAccess TRAVERSE DiscoveryAccess:DiscoveryAccessResult:DiscoveryResult:NetworkConnectionList TRAVERSE List:List:Member:DiscoveredNetworkConnection PROCESS WITH networkConnectionInfo`	BMC Discovery 應用程式擁有者
擷取應用程式探索的資料。	若要取得應用程式相依性的相關資訊，請使用查詢建置器來執行下列查詢： `search SoftwareInstance show key as 'SRC App ID', #Dependant:Dependency:DependedUpon:SoftwareInstance.key as 'DEST App ID'`	BMC Discovery 應用程式擁有者

任務	Description	所需的技能
擷取商業服務上的資料。	若要擷取主機提供之商業服務的資料，請使用查詢建置器來執行下列查詢： `search Host show name, #Host:HostedSoftware:AggregateSoftware:BusinessService.name as 'Name'`	BMC Discovery 應用程式擁有者

疑難排解

問題	解決方案
查詢無法執行或包含未填入的資料欄。	檢閱 BMC Discovery 中的資產記錄，並判斷您需要哪些欄位。然後，使用查詢建置器取代查詢中的這些欄位。
不會填入相依資產的詳細資訊。	這可能是由於存取許可或網路連線。探索工具可能沒有存取特定資產的必要許可，特別是當它們位於不同的網路或不同環境中時。 我們建議您與探索主題專家緊密合作，以確保識別所有相關資產。

相關資源

參考

- BMC Discovery 授權權利 (BMC 文件）
- BMC Discovery 功能和元件 (BMC 文件）
- BMC Discovery 使用者指南 (BMC 文件）
- 搜尋資料（在 BMC Discovery 上）(BMC 文件）
- 用於遷移的產品組合探索和分析 (AWS 規範性指導）

教學課程和影片

- BMC 探索：網路研討會 - 報告查詢最佳實務 （ 第 1 部分 ） (YouTube)

重新定位

主題

- 將 Amazon RDS for Oracle 資料庫遷移至另一個資料庫 AWS 帳戶 ，AWS 區域 並使用 AWS DMS 進行持續複寫
- 將 Amazon RDS 資料庫執行個體遷移至另一個 VPC 或帳戶
- 將 Amazon Redshift 叢集遷移至中國的 AWS 區域
- 使用 pg_transport 在兩個 Amazon RDS 資料庫執行個體之間傳輸 PostgreSQL 資料庫

將 Amazon RDS for Oracle 資料庫遷移至另一個資料庫 AWS 帳戶 ， AWS 區域 並使用 AWS DMS 進行持續複寫

Durga Prasad Cheepuri 和 Eduardo Valentim，Amazon Web Services

總結

警告：IAM 使用者具有長期登入資料，這會造成安全風險。為了協助降低此風險，建議您只為這些使用者提供執行任務所需的許可，並在不再需要這些使用者時將其移除。

此模式會逐步引導您將適用於 Oracle 的 Amazon Relational Database Service (Amazon RDS) 來源資料庫遷移至不同 AWS 帳戶 和 的步驟 AWS 區域。模式使用資料庫快照進行一次性完整資料載入，並啟用 AWS Database Migration Service (AWS DMS) 進行持續複寫。

先決條件和限制

先決條件

- 作用中 AWS 帳戶 ，其中包含來源 Amazon RDS for Oracle 資料庫，該資料庫已使用非default AWS Key Management Service (AWS KMS) 金鑰加密
- AWS 帳戶 與來源資料庫 AWS 區域 不同的作用中 ，用於目標 Amazon RDS for Oracle 資料庫
- 來源和目標 VPCs 之間的虛擬私有雲端 (VPC) 對等互連
- 熟悉使用 Oracle 資料庫做為 的來源 AWS DMS
- 熟悉使用 Oracle 資料庫做為 的目標 AWS DMS

產品版本

- Oracle 版本 11g （版本 11.2.0.3.v1 和更新版本） 和最高 12.2 和 18c。如需支援版本的最新清單，請參閱文件中使用 Oracle 資料庫做為 的來源 AWS DMS，以及使用 Oracle 資料庫做為 文件的目標 AWS DMS。 AWS 如需 Amazon RDS 支援的 Oracle 版本，請參閱 Amazon RDS 上的 Oracle。

Architecture

來源和目標技術堆疊

- Amazon RDS for Oracle 資料庫執行個體

持續複寫架構

工具

用於一次性完整資料載入的工具

- Amazon Relational Database Service (Amazon RDS) 會建立資料庫執行個體的儲存磁碟區快照，備份整個資料庫執行個體，而不只是個別資料庫。建立資料庫快照時，您必須找出要進行備份的資料庫執行個體，並為該資料庫快照命名，使得您稍後可透過它進行還原。建立快照所需的時間量因資料庫的大小而異。由於快照包括整個儲存體磁碟區，檔案大小，例如暫存檔案，也會影響建立快照所需的時間量。如需使用資料庫快照的詳細資訊，請參閱 Amazon RDS 文件中的建立資料庫快照。
- AWS Key Management Service (AWS KMS) 會建立 Amazon RDS 加密的 金鑰。當您建立加密的資料庫執行個體時，您也可以提供加密AWS KMS金鑰的金鑰識別符。如果您未指定AWS KMS金鑰識別符，Amazon RDS 會將您的預設加密金鑰用於新的資料庫執行個體。 會為您的 AWS KMS建立預設加密金鑰 AWS 帳戶。每個 AWS 帳戶 都有不同的預設加密金鑰 AWS 區域。對於此模式，應使用非預設AWS KMS金鑰加密 Amazon RDS 資料庫執行個體。如需使用 Amazon RDS 加密AWS KMS金鑰的詳細資訊，請參閱 Amazon RDS 文件中的加密 Amazon RDS 資源。

用於持續複寫的工具

- AWS Database Migration Service (AWS DMS) 用於複寫正在進行的變更，並保持來源和目標資料庫的同步。如需使用 AWS DMS 進行持續複寫的詳細資訊，請參閱 AWS DMS 文件中的使用 AWS DMS 複寫執行個體。

史詩

設定您的來源 AWS 帳戶

任務	Description	所需的技能
準備來源 Oracle 資料庫執行個體。	讓 Amazon RDS for Oracle 資料庫執行個體以 ARCHIVELOG 模式執行，並設定保留期間。如需詳細資訊，請參閱使	DBA

任務	Description	所需的技能
	用 AWS 受管 Oracle 資料庫做為 的來源 AWS DMS。	
設定來源 Oracle 資料庫執行個體的補充記錄。	設定 Amazon RDS for Oracle 資料庫執行個體的資料庫層級和資料表層級補充記錄。如需詳細資訊，請參閱使用 AWS 受管 Oracle 資料庫做為 的來源 AWS DMS。	DBA
更新來源帳戶中的 AWS KMS 金鑰政策。	更新來源中的 AWS KMS 金鑰政策 AWS 帳戶 ， AWS 帳戶 以允許目標使用加密的 Amazon RDS AWS KMS 金鑰。如需詳細資訊，請參閱 AWS KMS 文件。	SysAdmin
建立來源資料庫執行個體的手動 Amazon RDS 資料庫快照。		AWS IAM 使用者
與目標共用手動加密的 Amazon RDS 快照 AWS 帳戶。	如需詳細資訊，請參閱共用資料庫快照。	AWS IAM 使用者

設定您的目標 AWS 帳戶

任務	Description	所需的技能
連接政策。	在目標中 AWS 帳戶，將 AWS Identity and Access Management (IAM) 政策連接至根 IAM 使用者，以允許 IAM 使用者使用共用 AWS KMS 金鑰複製加密的資料庫快照。	SysAdmin

任務	Description	所需的技能
切換到來源 AWS 區域。		AWS IAM 使用者
複製共用快照。	在 Amazon RDS 主控台的快照窗格中，選擇與我共用，然後選取共用快照。使用來源資料庫所用 AWS KMS 金鑰的 Amazon Resource Name (ARN)，將快照複製到與來源資料庫 AWS 區域 相同的 。如需詳細資訊，請參閱複製資料庫快照。	AWS IAM 使用者
切換到目標 AWS 區域，並建立新的 AWS KMS 金鑰。		AWS IAM 使用者
複製快照。	切換到來源 AWS 區域。在 Amazon RDS 主控台的快照窗格中，選擇由我擁有，然後選取複製的快照。 AWS 區域使用新目標的 AWS KMS 金鑰，將快照複製到目標 AWS 區域。	AWS IAM 使用者
還原快照。	切換到目標 AWS 區域。在 Amazon RDS 主控台的快照窗格中，選擇由我擁有。選取複製的快照，並將其還原至 Amazon RDS for Oracle 資料庫執行個體。如需詳細資訊，請參閱從資料庫快照還原。	AWS IAM 使用者

準備您的來源資料庫以進行持續複寫

任務	Description	所需的技能
建立具有適當許可的 Oracle 使用者。	建立具有 Oracle 所需權限的 Oracle 使用者做為來源 AWS DMS。如需詳細資訊，請參閱 AWS DMS 文件。	DBA
設定 Oracle LogMiner 或 Oracle Binary Reader 的來源資料庫。		DBA

準備您的目標資料庫以進行持續複寫

任務	Description	所需的技能
建立具有適當許可的 Oracle 使用者。	建立具有 Oracle 所需權限的 Oracle 使用者做為目標 AWS DMS。如需詳細資訊，請參閱 AWS DMS 文件。	DBA

建立 AWS DMS 元件

任務	Description	所需的技能
在目標中建立複寫執行個體 AWS 區域。	在目標的 VPC 中建立複寫執行個體 AWS 區域。如需詳細資訊，請參閱 AWS DMS 文件。	AWS IAM 使用者
使用必要的加密和測試連線來建立來源和目標端點。	如需詳細資訊，請參閱 AWS DMS 文件。	DBA
建立複寫任務。	1. 針對遷移類型，選擇持續複寫。	IAM 使用者

任務	Description	所需的技能
	2. 對於變更資料擷取 (CDC) 起點，請在拍攝 Amazon RDS 快照進行完全載入時使用 Oracle 系統變更編號 (SCN)，或在拍攝完全載入時使用時間戳記。 3. 針對 `TargetTablePrepMode` ，選擇 DO_NOTHING。如果任務具有大型二進位物件 (LOB) 資料表，請選擇有限 LOB 模式，並將最大 LOB 大小設定為資料表中 LOB 資料的大小上限。 4. 啟用記錄。 5. 將透過金鑰相關的資料表分組為單一任務。如果資料表具有大量 LOB 資料，且資料表與其他資料表沒有任何關係，請使用上述的 LOB 設定為其建立單獨的任務。 如需詳細資訊，請參閱 AWS DMS 文件。	
啟動任務並監控它們。	如需詳細資訊，請參閱 AWS DMS 文件。	AWS IAM 使用者
視需要對任務啟用驗證。	請注意，啟用驗證確實會對複寫產生效能影響。如需詳細資訊，請參閱 AWS DMS 文件。	AWS IAM 使用者

相關資源

- 變更金鑰政策
- 建立手動 Amazon RDS 資料庫快照
- 共用手動 Amazon RDS 資料庫快照
- 複製快照
- 從 Amazon RDS 資料庫快照還原
- 入門 AWS DMS
- 使用 Oracle 資料庫做為 的來源 AWS DMS
- 使用 Oracle 資料庫做為 的目標 AWS DMS
- AWS DMS 使用 VPC 對等互連進行設定
- 如何與另一個 共用手動 Amazon RDS 資料庫快照或資料庫叢集快照 AWS 帳戶？ (AWS 知識中心文章)

將 Amazon RDS 資料庫執行個體遷移至另一個 VPC 或帳戶

Dhrubajyoti Mukherjee，Amazon Web Services

總結

此模式提供將 Amazon Relational Database Service (Amazon RDS) 資料庫執行個體從一個虛擬私有雲端 (VPC) 遷移至相同 AWS 帳戶中另一個，或從一個 AWS 帳戶遷移至另一個 AWS 帳戶的指引。

如果您想要將 Amazon RDS 資料庫執行個體遷移到另一個 VPC 或帳戶，基於分離或安全原因 （ 例如，當您想要將應用程式堆疊和資料庫放置在不同的 VPCs時），此模式非常有用。

將資料庫執行個體遷移至另一個 AWS 帳戶需要採取一些步驟，例如手動快照、共用快照，以及還原目標帳戶中的快照。視資料庫變更和交易費率而定，此程序可能非常耗時。這也會導致資料庫停機，因此請事先規劃遷移。請考慮藍/綠部署策略，將停機時間降至最低。或者，您可以評估 AWS Data Migration Service (AWS DMS)，將變更的停機時間降至最低。不過，此模式不會涵蓋此選項。若要進一步了解，請參閱 AWS DMS 文件。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- VPC、子網路和 Amazon RDS 主控台所需的 AWS Identity and Access Management (IAM) 許可

限制

- 變更 VPC 會導致資料庫重新啟動，導致應用程式中斷。建議您在低尖峰時間進行遷移。
- 將 Amazon RDS 遷移至另一個 VPC 的限制：
  - 您要遷移的資料庫執行個體必須是沒有待命的單一執行個體。它不能是叢集的成員。
  - Amazon RDS 不得位於多個可用區域。
  - Amazon RDS 不得有任何僅供讀取複本。
  - 在目標 VPC 中建立的子網路群組必須具有來源資料庫執行所在可用區域的子網路。
- 將 Amazon RDS 遷移至另一個 AWS 帳戶時的限制：
  - 目前不支援共用使用 Amazon RDS 預設服務金鑰加密的快照。

Architecture

遷移至相同 AWS 帳戶中的 VPC

下圖顯示將 Amazon RDS 資料庫執行個體遷移至相同 AWS 帳戶中不同 VPC 的工作流程。

步驟包含下列項目。如需詳細說明，請參閱 Epics 一節。

1. 在目標 VPC 中建立資料庫子網路群組。資料庫子網路群組是子網路的集合，您可以在建立資料庫執行個體時用來指定特定的 VPC。
2. 在來源 VPC 中設定 Amazon RDS 資料庫執行個體，以使用新的資料庫子網路群組。
3. 套用變更，將 Amazon RDS 資料庫遷移至目標 VPC。

遷移至不同的 AWS 帳戶

下圖顯示將 Amazon RDS 資料庫執行個體遷移至不同 AWS 帳戶的工作流程。

步驟包含下列項目。如需詳細說明，請參閱 Epics 一節。

1. 存取來源 AWS 帳戶中的 Amazon RDS 資料庫執行個體。
2. 在來源 AWS 帳戶中建立 Amazon RDS 快照。
3. 與目標 AWS 帳戶共用 Amazon RDS 快照。
4. 存取目標 AWS 帳戶中的 Amazon RDS 快照。
5. 在目標 AWS 帳戶中建立 Amazon RDS 資料庫執行個體。

工具

AWS 服務

- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路類似於您在自己的資料中心內操作的傳統網路，具有使用可擴展的 AWS 基礎設施的優勢。

最佳實務

- 如果將 Amazon RDS 資料庫執行個體遷移至另一個 帳戶時需要考慮資料庫停機時間，我們建議您使用 AWS DMS。此服務提供資料複寫，這會導致不到五分鐘的中斷時間。

史詩

遷移至相同 AWS 帳戶中的不同 VPC

任務	Description	所需的技能
建立新 VPC	在 Amazon VPC 主控台上，建立具有所需屬性和 IP 地址範圍的新 VPC 和子網路。如需詳細說明，請參閱 Amazon VPC 文件。	管理員
建立資料庫子網路群組。	在 Amazon RDS 主控台上： 1. 選擇子網路群組、建立資料庫子網路群組。 2. 輸入子網路群組名稱、描述和 VPC ID。 3. 新增屬於子網路群組的子網路。新增子網路以涵蓋至少兩個可用區域。 4. 選擇建立。 如需詳細資訊，請參閱 Amazon RDS 文件。	管理員
修改 Amazon RDS 資料庫執行個體以使用新的子網路群組。	在 Amazon RDS 主控台上： 1. 在導覽窗格中，選擇資料庫，然後選擇要遷移的 Amazon RDS 資料庫執行個體。	管理員

任務	Description	所需的技能
	2. 在連線區段中，選擇與目標 VPC 相關聯的子網路群組。 3. 在排程修改區段中，選擇立即套用。 當遷移至目標 VPC 完成時，目標 VPC 的預設安全群組會指派給 Amazon RDS 資料庫執行個體。您可以使用資料庫執行個體所需的傳入和傳出規則，為該 VPC 設定新的安全群組。 或者，使用 AWS Command Line Interface (AWS CLI)，透過明確提供新的 VPC 安全群組 ID 來執行目標 VPC 的遷移。例如： `aws rds modify-db-instance \ --db-instance-identifier testrds \ --db-subnet-group-name new-vpc-subnet-group \ --vpc-security-group-ids sg-idxxxx \ --apply-immediately`	

遷移至不同的 AWS 帳戶

任務	Description	所需的技能
在目標 AWS 帳戶中建立新的 VPC 和子網路群組。	1. 在 Amazon VPC 主控台上，建立具有所需屬性和 IP 地址範圍的新 VPC。如需詳細說明，請參閱 Amazon VPC 文件。 2. 遵循 Amazon VPC 文件中的指示，為新 VPC 建立子網路。 3. 在 Amazon RDS 主控台上，建立資料庫子網路群組。如需說明，請參閱 Amazon RDS 文件。	管理員
共用資料庫的手動快照，並與目標帳戶共用。	1. 遵循 Amazon RDS 文件中的指示，手動拍攝來源資料庫的快照。 2. 提供目標帳戶 ID，與目標 AWS 帳戶共用快照。如需說明，請參閱 re：Post 有關與其他 帳戶共用資料庫快照的文章。	管理員
啟動新的 Amazon RDS 資料庫執行個體。	從目標 AWS 帳戶中的共用快照啟動新的 Amazon RDS 資料庫執行個體。如需說明，請參閱 Amazon RDS 文件。	管理員

相關資源

- Amazon VPC 文件
- Amazon RDS 文件
- 如何變更 RDS 資料庫執行個體的 VPC？ (AWS re：Post 文章)

- 如何將 Amazon RDS 資源的所有權轉移到不同的 AWS 帳戶？ (AWS re：Post 文章）
- 如何與其他 AWS 帳戶共用手動 Amazon RDS 資料庫快照或 Aurora 資料庫叢集快照？ (AWS re：Post 文章）
- AWS DMS 文件

將 Amazon Redshift 叢集遷移至中國的 AWS 區域

Jing Yan，Amazon Web Services

總結

此模式提供step-by-step方法，將 Amazon Redshift 叢集從另一個 AWS 區域遷移至中國的 AWS 區域。

此模式使用 SQL 命令重新建立所有資料庫物件，並使用 UNLOAD 命令將此資料從 Amazon Redshift 移至來源區域中的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。然後，資料會遷移至中國 AWS 區域中的 S3 儲存貯體。COPY 命令用於從 S3 儲存貯體載入資料，並將其傳輸至目標 Amazon Redshift 叢集。

Amazon Redshift 目前不支援跨區域功能，例如快照複製到中國的 AWS 區域。此模式提供了解決該限制的方法。您也可以反轉此模式中的步驟，將資料從中國的 AWS 區域遷移到另一個 AWS 區域。

先決條件和限制

先決條件

- 中國區域和中國以外 AWS 區域的作用中 AWS 帳戶
- 中國區域和中國以外 AWS 區域中現有的 Amazon Redshift 叢集

限制

- 這是離線遷移，這表示來源 Amazon Redshift 叢集無法在遷移期間執行寫入操作。

Architecture

來源技術堆疊

- 中國以外 AWS 區域中的 Amazon Redshift 叢集

目標技術堆疊

- 中國 AWS 區域中的 Amazon Redshift 叢集

目標架構

工具

工具

- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種物件儲存服務，可提供可擴展性、資料可用性、安全性和效能。您可以使用 Amazon S3 從 Amazon Redshift 存放資料，也可以將資料從 S3 儲存貯體複製到 Amazon Redshift。
- Amazon Redshift – Amazon Redshift 是雲端中全受管的 PB 級資料倉儲服務。
- psql – psql 是 PostgreSQL 的終端型前端。

史詩

準備在來源區域中遷移

任務	Description	所需的技能
在來源區域中啟動和設定 EC2 執行個體。	登入 AWS 管理主控台並開啟 Amazon Elastic Compute Cloud (Amazon EC2) 主控台。您目前的區域會顯示在畫面頂端的導覽列中。此區域不能是中國的 AWS 區域。從 Amazon EC2 主控台儀表板中，選擇「啟動執行個體」，然後建立和設定 EC2 執行個體。重要：確保您的傳入規則的 EC2 安全群組允許從來源機器無限制存取 TCP 連接埠 22。如需如何啟動和設定 EC2 執行個體的說明，請參閱「相關資源」一節。	DBA、開發人員
安裝 psql 工具。	下載並安裝 PostgreSQL。Amazon Redshift 不提供 psql 工具，它會與 PostgreSQL 一起安裝。如需使用 psql 和安裝 PostgreSQL 工具的詳細	DBA

任務	Description	所需的技能
	資訊，請參閱「相關資源」一節。	
記錄 Amazon Redshift 叢集詳細資訊。	開啟 Amazon Redshift 主控台，然後在導覽窗格中選擇「叢集」。然後從清單中選擇 Amazon Redshift 叢集名稱。在「屬性」索引標籤的「資料庫組態」區段中，記錄「資料庫名稱」和「連接埠」。開啟「連線詳細資訊」區段，並記錄「端點」，其格式為「端點：<連接埠>/<資料庫名稱>」。重要：確保您的傳入規則的 Amazon Redshift 安全群組允許從 EC2 執行個體無限制存取 TCP 連接埠 5439。	DBA
將 psql 連接至 Amazon Redshift 叢集。	在命令提示字元中，執行「psql -h <endpoint> -U <userid> -d <databasename> -p <port>」命令來指定連線資訊。在 psql 密碼提示中，輸入 "<userid>" 使用者的密碼。您接著會連線至 Amazon Redshift 叢集，並以互動方式輸入命令。	DBA
建立 S3 儲存貯體。	開啟 Amazon S3 主控台，並建立 S3 儲存貯體以存放從 Amazon Redshift 匯出的檔案。如需如何建立 S3 儲存貯體的說明，請參閱「相關資源」一節。	DBA、AWS General

任務	Description	所需的技能
建立支援卸載資料的 IAM 政策。	開啟 AWS Identity and Access Management (IAM) 主控台，然後選擇「政策」。選擇「建立政策」，然後選擇「JSON」索引標籤。複製並貼上從「其他資訊」區段卸載資料的 IAM 政策。重要：將 "s3_bucket_name" 取代為您的 S3 儲存貯體名稱。選擇「檢閱政策」，然後輸入政策的名稱和描述。選擇「建立政策」。	DBA
建立 IAM 角色以允許 Amazon Redshift 的 UNLOAD 操作。	開啟 IAM 主控台並選擇「角色」。選擇「建立角色」，然後在「選取信任實體類型」中選擇「AWS 服務」。為服務選擇 "Redshift"，選擇 "Redshift – Customizable"，然後選擇 "Next"。選擇您先前建立的「卸載」政策，然後選擇「下一步」。輸入「角色名稱」，然後選擇「建立角色」。	DBA
將 IAM 角色與 Amazon Redshift 叢集建立關聯。	開啟 Amazon Redshift 主控台，然後選擇「管理 IAM 角色」。從下拉式選單中選擇「可用角色」，然後選擇您先前建立的角色。選擇「套用變更」。當「管理 IAM 角色」上 IAM 角色的「狀態」顯示為「同步中」時，您可以執行 UNLOAD 命令。	DBA

任務	Description	所需的技能
停止對 Amazon Redshift 叢集的寫入操作。	您必須記得停止對來源 Amazon Redshift 叢集的所有寫入操作，直到遷移完成為止。	DBA

準備在目標區域中遷移

任務	Description	所需的技能
在目標區域中啟動並設定 EC2 執行個體。	登入中國區域的 AWS 管理主控台，可以是北京或寧夏。從 Amazon EC2 主控台中，選擇「啟動執行個體」，然後建立和設定 EC2 執行個體。重要：確定傳入規則的 Amazon EC2 安全群組允許從來源機器無限制存取 TCP 連接埠 22。如需如何啟動和設定 EC2 執行個體的進一步指示，請參閱「相關資源」一節。	DBA
記錄 Amazon Redshift 叢集詳細資訊。	開啟 Amazon Redshift 主控台，然後在導覽窗格中選擇「叢集」。然後從清單中選擇 Amazon Redshift 叢集名稱。在「屬性」索引標籤的「資料庫組態」區段中，記錄「資料庫名稱」和「連接埠」。開啟「連線詳細資訊」區段，並記錄「端點」，其格式為「端點：<連接埠>/<資料庫名稱>」。重要：請確定傳入規則的 Amazon Redshift 安全群組	DBA

任務	Description	所需的技能
	允許從 EC2 執行個體無限制存取 TCP 連接埠 5439。	
將 psql 連接至 Amazon Redshift 叢集。	在命令提示字元中，執行「psql -h <endpoint> -U <userid> -d <databasename> -p <port>」命令來指定連線資訊。在 psql 密碼提示中，輸入 "<userid>" 使用者的密碼。您接著會連線至 Amazon Redshift 叢集，並以互動方式輸入命令。	DBA
建立 S3 儲存貯體。	開啟 Amazon S3 主控台，並建立 S3 儲存貯體以保留從 Amazon Redshift 匯出的檔案。如需此案例和其他案例的協助，請參閱「相關資源」一節。	DBA
建立支援複製資料的 IAM 政策。	開啟 IAM 主控台並選擇「政策」。選擇「建立政策」，然後選擇「JSON」索引標籤。複製並貼上從「其他資訊」區段複製資料的 IAM 政策。重要：使用 S3 儲存貯體的名稱取代 "s3_bucket_name"。選擇「檢閱政策」，輸入政策的名稱和描述。選擇「建立政策」。	DBA

任務	Description	所需的技能
建立 IAM 角色以允許 Amazon Redshift 的 COPY 操作。	開啟 IAM 主控台並選擇「角色」。選擇「建立角色」，然後在「選取信任實體類型」中選擇「AWS 服務」。為服務選擇「Redshift」，選擇「Redshift – 可自訂」，然後選擇「下一步」。選擇您先前建立的「複製」政策，然後選擇「下一步」。輸入「角色名稱」，然後選擇「建立角色」。	DBA
將 IAM 角色與 Amazon Redshift 叢集建立關聯。	開啟 Amazon Redshift 主控台，然後選擇「管理 IAM 角色」。從下拉式選單中選擇「可用角色」，然後選擇您先前建立的角色。選擇「套用變更」。當「管理 IAM 角色」上 IAM 角色的「狀態」顯示為「 In-sync」時，您可以執行「COPY」命令。	DBA

在開始遷移之前驗證來源資料和物件資訊

任務	Description	所需的技能
驗證來源 Amazon Redshift 資料表中的資料列。	使用「其他資訊」區段中的指令碼來驗證和記錄來源 Amazon Redshift 資料表中的資料列數。請記得平均分割 UNLOAD 和 COPY 指令碼的資料。這將改善資料卸載和載入效率，因為每個指令碼涵蓋的資料數量將會平衡。	DBA

任務	Description	所需的技能
驗證來源 Amazon Redshift 叢集中的資料庫物件數量。	使用「其他資訊」區段中的指令碼來驗證和記錄來源 Amazon Redshift 叢集中的資料庫、使用者、結構描述、資料表、檢視和使用者定義函數 (UDFs) 的數量。	DBA
在遷移之前驗證 SQL 陳述式結果。	某些用於資料驗證的 SQL 陳述式應根據實際業務和資料情況進行排序。這是為了驗證匯入的資料，以確保其一致且正確顯示。	DBA

將資料和物件遷移至目標區域

任務	Description	所需的技能
產生 Amazon Redshift DDL 指令碼。	使用「SQL 陳述式」區段中的連結來查詢 Amazon Redshift，以產生資料定義語言 (DDL) 指令碼。這些 DDL 指令碼應包含 "create user"、"create schema"、"privileges on schema to user"、"create table/view"、"privileges on object to user" 和 "create function" 查詢。	DBA
在目標區域的 Amazon Redshift 叢集中建立物件。	在中國的 AWS 區域中，使用 AWS 命令列界面 (AWS CLI) 執行 DDL 指令碼。這些指令碼會在目標區域的 Amazon Redshift 叢集中建立物件。	DBA

任務	Description	所需的技能
將來源 Amazon Redshift 叢集資料卸載至 S3 儲存貯體。	執行 UNLOAD 命令，將資料從來源區域中的 Amazon Redshift 叢集卸載至 S3 儲存貯體。	DBA、開發人員
將來源區域 S3 儲存貯體資料傳輸至目標區域 S3 儲存貯體。	將資料從來源區域 S3 儲存貯體傳輸到目標 S3 儲存貯體。由於無法使用「$ aws s3 同步」命令，請務必使用「相關資源」區段中的「將 Amazon S3 資料從 AWS 區域轉移到中國的 AWS 區域」文章中概述的程序。	開發人員
將資料載入目標 Amazon Redshift 叢集。	在目標區域的 psql 工具中，執行 COPY 命令，將資料從 S3 儲存貯體載入目標 Amazon Redshift 叢集。	DBA

在遷移後驗證來源和目標區域中的資料

任務	Description	所需的技能
驗證並比較來源和目標資料表中的資料列數。	驗證並比較來源和目標區域中的資料表資料列數目，以確保全部都已遷移。	DBA
驗證並比較來源和目標資料庫物件的數量。	驗證並比較來源和目標區域中的所有資料庫物件，以確保全部都已遷移。	DBA
驗證和比較來源和目標區域中的 SQL 指令碼結果。	執行遷移之前準備的 SQL 指令碼。驗證並比較資料，以確保 SQL 結果正確。	DBA

任務	Description	所需的技能
重設目標 Amazon Redshift 叢集中所有使用者的密碼。	遷移完成且所有資料都經過驗證後，您應該重設中國 AWS 區域中 Amazon Redshift 叢集的所有使用者密碼。	DBA

相關資源

- 將 Amazon S3 資料從 AWS 區域傳輸到中國的 AWS 區域
- 建立 S3 儲存貯體
- 重設 Amazon Redshift 使用者密碼
- psql 文件

其他資訊

用於卸載資料的 IAM 政策

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::s3_bucket_name"]
    },
    {
      "Effect": "Allow",
      "Action": ["s3:GetObject", "s3:DeleteObject"],
      "Resource": ["arn:aws:s3:::s3_bucket_name/*"]
    }
  ]
}
```

用於複製資料的 IAM 政策

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::s3_bucket_name"]
    },
    {
      "Effect": "Allow",
      "Action": ["s3:GetObject"],
      "Resource": ["arn:aws:s3:::s3_bucket_name/*"]
    }
  ]
}
```

查詢 Amazon Redshift 的 SQL 陳述式

```
##Database

select * from pg_database where datdba>1;

##User

select * from pg_user where usesysid>1;

##Schema

SELECT n.nspname AS "Name",

  pg_catalog.pg_get_userbyid(n.nspowner) AS "Owner"

FROM pg_catalog.pg_namespace n

WHERE n.nspname !~ '^pg_' AND n.nspname <> 'information_schema'

ORDER BY 1;

##Table

select count(*) from pg_tables where schemaname not in
 ('pg_catalog','information_schema');

select schemaname,count(*) from pg_tables where schemaname not in
 ('pg_catalog','information_schema') group by schemaname order by 1;
```

```
##View

SELECT

    n.nspname AS schemaname,c.relname AS
 viewname,pg_catalog.pg_get_userbyid(c.relowner) as "Owner"

FROM

    pg_catalog.pg_class AS c

INNER JOIN

    pg_catalog.pg_namespace AS n

    ON c.relnamespace = n.oid

WHERE relkind = 'v' and n.nspname not in ('information_schema','pg_catalog');

##UDF

SELECT

   n.nspname AS schemaname,

   p.proname AS proname,

   pg_catalog.pg_get_userbyid(p.proowner) as "Owner"

FROM pg_proc p

LEFT JOIN pg_namespace n on n.oid = p.pronamespace

WHERE p.proowner != 1;
```

產生 DDL 陳述式的 SQL 指令碼

- Get_schema_priv_by_user 指令碼
- Generate_tbl_ddl 指令碼
- Generate_view_ddl
- Generate_user_grant_revoke_ddl
- Generate_udf_ddl

使用 pg_transport 在兩個 Amazon RDS 資料庫執行個體之間傳輸 PostgreSQL 資料庫

Raunak Rishabh 和 Jitender Kumar，Amazon Web Services

總結

此模式說明使用 pg_transport 擴充功能，在兩個 Amazon Relational Database Service (Amazon RDS) for PostgreSQL 資料庫執行個體之間遷移極大型資料庫的步驟。此擴充套件提供實體的傳輸機制來移動每個資料庫。透過以最少的處理方式串流資料庫檔案，它提供非常快速的方法，可在資料庫執行個體之間遷移大型資料庫，並將停機時間降至最低。此延伸模組使用提取模型，其中目標資料庫執行個體會從來源資料庫執行個體匯入資料庫。

先決條件和限制

先決條件

- 兩個資料庫執行個體都必須執行相同的 PostgreSQL 主要版本。
- 資料庫不得存在於目標上。否則，傳輸會失敗。
- 來源資料庫中不得啟用 pg_transport 以外的擴充功能。
- 所有來源資料庫物件都必須位於預設 pg_default 資料表空間中。
- 來源資料庫執行個體的安全群組應允許來自目標資料庫執行個體的流量。
- 安裝 PostgreSQL 用戶端，例如 psql 或 PgAdmin，以使用 Amazon RDS PostgreSQL 資料庫執行個體。您可以在本機系統中安裝用戶端，或使用 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。在此模式中，我們在 EC2 執行個體上使用 psql。

限制

- 您無法在 Amazon RDS for PostgreSQL 的不同主要版本之間傳輸資料庫。
- 來源資料庫的存取權限和擁有權不會傳輸至目標資料庫。
- 您無法在僅供讀取複本或僅供讀取複本的父執行個體上傳輸資料庫。
- 您無法在計劃使用此方法傳輸的任何資料庫資料表中使用 reg 資料類型。
- 在資料庫執行個體上，您最多可以同時執行總共 32 個傳輸 （包括匯入和匯出）。
- 您無法重新命名或包含/排除資料表。一切都會照原樣遷移。

注意

- 在移除擴充功能之前進行備份，因為移除擴充功能也會移除相依物件和一些對資料庫操作至關重要的資料。
- 當您判斷 pg_transport 的工作者數量和work_mem值時，請考慮在來源執行個體的其他資料庫上執行的執行個體類別和程序。
- 傳輸開始時，來源資料庫上的所有連線都會結束，且資料庫會進入唯讀模式。

> **Note**
>
> 當傳輸在一個資料庫上執行時，不會影響相同伺服器上的其他資料庫。

產品版本

- Amazon RDS for PostgreSQL 10.10 及更新版本，以及 Amazon RDS for PostgreSQL 11.5 及更新版本。如需最新版本的資訊，請參閱 Amazon RDS 文件中的在資料庫執行個體之間傳輸 PostgreSQL 資料庫。

Architecture

工具

- pg_transport 提供實體傳輸機制來移動每個資料庫。透過以最少的處理方式串流資料庫檔案，實體傳輸會比傳統傾印和載入程序更快地移動資料，並且需要最少的停機時間。PostgreSQL 可傳輸的資料庫使用提取模式，也就是目的地的資料庫執行個體從來源資料庫執行個體輸入資料庫。當您準備來源和目標環境時，您可以在資料庫執行個體上安裝此延伸模組，如此模式所述。
- psql 可讓您連線至 PostgreSQL 資料庫執行個體並加以使用。若要在您的系統上安裝 psql，請參閱 PostgreSQL 下載頁面。

史詩

建立目標參數群組

任務	Description	所需的技能
建立目標系統的參數群組。	指定可識別為目標參數群組的群組名稱，例如 pgtarget-	DBA

任務	Description	所需的技能
	`param-group` 。如需說明，請參閱 Amazon RDS 文件。	

任務	Description	所需的技能
修改參數群組的參數。	設定下列參數： 1. 將 pg_transport 新增至 shared_preload_libraries 參數。 ``` shared_preload_libraries = pg_stat_statements, pg_transport ``` 2. 設定 pg_transport.num_workers 參數。選擇您要用來執行傳輸的工作者數量。您設定的值決定要在來源中建立的transport.send_file 工作者數量。 3. 將 的值增加max_worker_processes 到 值的三倍以上pg_transport.num_workers 。例如，如果您將 值設定為 pg_transport.num_workers 4，則max_worker_processes 值應至少為 13。如果失敗，pg_transport 會建議最小值。 4. pg_transport.timing 設定為 1。此設定可啟用傳輸期間的時間資訊報告。 5. 設定 pg_transport.work_mem 參數。此參數指定要分配給每個工	DBA

任務	Description	所需的技能
	作者的最大記憶體。預設值為 128 MB。 如需這些參數的詳細資訊，請參閱 Amazon RDS 文件。	

建立來源參數群組

任務	Description	所需的技能
建立來源系統的參數群組。	指定群組名稱，將其識別為來源參數群組；例如 `pgsource-param-group` 。如需說明，請參閱 Amazon RDS 文件。	DBA
修改參數群組的參數。	設定下列參數： 1. 將 `pg_transport` 新增至 `shared_preload_libraries` 參數。 ``` shared_preload_libraries = pg_stat_statements, pg_transport ``` 2. 設定 `pg_transport.num_workers` 參數。目標中定義的此參數值會決定要使用的`transport.send_file` 工作者數量。如果您在此執行個體上執行匯入，請增加此值，	DBA

任務	Description	所需的技能
	但請考慮已執行的工作者數量。 3. 將 的值增加max_worker_processes 到目標pg_transport.num_workers 上 值的三倍以上。例如，如果您在目標上將 值設定為 pg_transport.num_workers 4，則來源上的max_worker_processes 值應至少為 13。如果失敗，pg_transport 會建議最小值。 4. 設定 pg_transport.work_mem 參數。此參數指定要分配給每個工作者的最大記憶體。預設值為 128 MB。 如需這些參數的詳細資訊，請參閱 Amazon RDS 文件。	

準備目標環境

任務	Description	所需的技能
建立新的 Amazon RDS for PostgreSQL 資料庫執行個體以傳輸來源資料庫。	根據您的業務需求決定執行個體類別和 PostgreSQL 版本。	DBA、系統管理員、資料庫架構師
修改目標的安全群組，以允許來自 EC2 執行個體的資料庫執行個體連接埠連線。	根據預設，PostgreSQL 執行個體的連接埠為 5432。如果您使用另一個連接埠，則必須為	DBA，系統管理員

任務	Description	所需的技能
	EC2 執行個體開啟該連接埠的連線。	
修改執行個體，並指派新的目標參數群組。	例如 `pgtarget-param-group` 。	DBA
重新啟動目標 Amazon RDS 資料庫執行個體。	參數 `shared_preload_libraries` 和 `max_worker_processes` 是靜態參數，需要重新啟動執行個體。	DBA，系統管理員
使用 psql 從 EC2 執行個體連線至資料庫。	使用 命令： `psql -h <rds_end_point> -p PORT -U username -d database -W`	DBA
建立 pg_transport 擴充功能。	以具有 `rds_superuser` 角色的使用者身分執行下列查詢： `create extension pg_transport;`	DBA

準備來源環境

任務	Description	所需的技能
修改來源的安全群組，以允許來自 Amazon EC2 執行個體和目標資料庫執行個體的資料庫執行個體連接埠連線	根據預設，PostgreSQL 執行個體的連接埠為 5432。如果您使用另一個連接埠，則必須為 EC2 執行個體開啟該連接埠的連線。	DBA，系統管理員
修改執行個體並指派新的來源參數群組。	例如 `pgsource-param-group` 。	DBA

任務	Description	所需的技能
重新啟動來源 Amazon RDS 資料庫執行個體。	參數 `shared_preload_libraries` 和 `max_worker_processes` 是靜態參數，需要重新啟動執行個體。	DBA
使用 psql 從 EC2 執行個體連線至資料庫。	使用 命令： `psql -h <rds_end_point> -p PORT -U username -d database -W`	DBA
建立 pg_transport 擴充功能，並從要傳輸的資料庫移除所有其他擴充功能。	如果來源資料庫上安裝 pg_transport 以外的任何延伸項目，則傳輸會失敗。此命令必須由具有 `rds_superuser` 角色的使用者執行。	DBA

執行傳輸

任務	Description	所需的技能
執行試轉。	使用 `transport.import_from_server` 函數先執行試轉： `SELECT transport.import_from_server( 'source-db-instance-endpoint', source-db-instance-port, 'source-db-instance-user', 'source-user-password', 'source-database-name', 'destination-user-password', 'true');`	DBA

任務	Description	所需的技能
	此函數的最後一個參數（設定為 `true`) 會定義試轉。 此函數會顯示您在執行主要傳輸時會看到的任何錯誤。在執行主要傳輸之前解決錯誤。	
如果試轉成功，請啟動資料庫傳輸。	執行 `transport.import_from_server` 函數以執行傳輸。它連接到來源並匯入資料。 `SELECT transport.import_from_server( 'source-db-instance-endpoint', source-db-instance-port, 'source-db-instance-user', 'source-user-password', 'source-database-name', 'destination-user-password', false);` 此函數的最後一個參數（設定為 `false`) 表示這不是試轉。	DBA
執行傳輸後步驟。	資料庫傳輸完成後： • 驗證目標環境中的資料。 • 將所有角色和許可新增至目標。 • 如有需要，請啟用目標和來源中所有必要的延伸模組。 • 還原 `max_worker_processes` 參數的值。	DBA

相關資源

- Amazon RDS 文件
- pg_transport 文件
- 使用 RDS PostgreSQL 可傳輸資料庫遷移資料庫（部落格文章）
- PostgreSQL 下載
- psql 公用程式
- 建立資料庫參數群組
- 修改資料庫參數群組中的參數
- PostgreSQL 下載

平台重建

主題

- 使用 AWS DMS 將 Microsoft SQL Server 資料庫匯出至 Amazon S3
- 將 Oracle CLOB 值遷移至 AWS 上的 PostgreSQL 中的個別資料列
- 透過資料庫連結使用直接 Oracle Data Pump Import，將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 將 Oracle PeopleSoft 遷移至 Amazon RDS Custom
- 將 Oracle ROWID 功能遷移至 AWS 上的 PostgreSQL
- 將 Oracle 資料庫錯誤代碼遷移至與 Amazon Aurora PostgreSQL 相容的資料庫
- 使用 AWS SCT 和 AWS DMS 將 Amazon EC2 上的 SAP ASE 遷移至與 Amazon Aurora PostgreSQL 相容
- 使用 ACM 將 Windows SSL 憑證遷移至 Application Load Balancer
- 將訊息佇列從 Microsoft Azure Service Bus 遷移至 Amazon SQS
- 使用 Oracle Data Pump 和 AWS DMS 將 Oracle JD Edwards EnterpriseOne 資料庫遷移至 AWS
- 使用 AWS DMS 將 Oracle PeopleSoft 資料庫遷移至 AWS
- 將內部部署 MySQL 資料庫遷移至 Amazon RDS for MySQL
- 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon RDS for SQL Server
- 使用 Rclone 將資料從 Microsoft Azure Blob 遷移至 Amazon S3
- 從 Couchbase Server 遷移至 AWS 上的 Couchbase Capella
- 從 IBM WebSphere Application Server 遷移至 Amazon EC2 上的 Apache Tomcat
- 使用 Auto Scaling 從 IBM WebSphere Application Server 遷移至 Amazon EC2 上的 Apache Tomcat
- 將 .NET 應用程式從 Microsoft Azure App Service 遷移至 AWS Elastic Beanstalk
- 在 Amazon ECS 上從 Oracle WebLogic 遷移至 Apache Tomcat (TomEE)
- 使用 AWS DMS 將 Oracle 資料庫從 Amazon EC2 遷移至 Amazon RDS for Oracle
- 使用 Logstash 將內部部署 Oracle 資料庫遷移至 Amazon OpenSearch Service
- 將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 使用 Oracle Data Pump 將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 使用 pglogical 從 Amazon EC2 上的 PostgreSQL 遷移至 Amazon RDS for PostgreSQL Amazon EC2

- 將內部部署 PostgreSQL 資料庫遷移至 Aurora PostgreSQL
- 將內部部署 Microsoft SQL Server 資料庫遷移至執行 Linux 的 Amazon EC2 上的 Microsoft SQL Server
- 使用連結的伺服器將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon RDS for SQL Server
- 使用原生備份和還原方法，將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon RDS for SQL Server
- 使用 AWS DMS 和 AWS SCT 將 Microsoft SQL Server 資料庫遷移至 Aurora MySQL
- 使用原生工具將內部部署 MariaDB 資料庫遷移至 Amazon RDS for MariaDB
- 將內部部署 MySQL 資料庫遷移至 Aurora MySQL
- 使用 Percona XtraBackup、Amazon EFS 和 Amazon S3 將內部部署 MySQL 資料庫遷移至 Aurora MySQL
- 使用 AWS App2Container 將內部部署 Java 應用程式遷移至 AWS
- 在 AWS 大型遷移中遷移共用檔案系統
- 使用 Oracle GoldenGate 平面檔案轉接器，將 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 變更 Python 和 Perl 應用程式，以支援從 Microsoft SQL Server 到 Amazon Aurora PostgreSQL 相容版本的資料庫遷移

使用 AWS DMS 將 Microsoft SQL Server 資料庫匯出至 Amazon S3

Sweta Krishna，Amazon Web Services

總結

組織通常需要將資料庫複製到 Amazon Simple Storage Service (Amazon S3)，以進行資料庫遷移、備份和還原、資料封存和資料分析。此模式說明如何將 Microsoft SQL Server 資料庫匯出至 Amazon S3。來源資料庫可以在內部部署或 Amazon Elastic Compute Cloud (Amazon EC2) 或 Amazon Web Services (AWS) Cloud 上 Microsoft SQL Server 的 Amazon Relational Database Service (Amazon RDS) 上託管。

使用 AWS Database Migration Service (AWS DMS) 匯出資料。根據預設，AWS DMS 會以逗號分隔值 (.csv) 格式寫入完全載入和變更資料擷取 (CDC) 資料。對於更精簡的儲存和更快的查詢選項，此模式使用 Apache Parquet (.parquet) 格式選項。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 帳戶 AWS Identity and Access Management (IAM) 角色，具有目標 S3 儲存貯體的寫入、刪除和標記存取權，而 AWS DMS (`dms.amazonaws.com`) 已新增為此 IAM 角色的信任實體
- 內部部署 Microsoft SQL Server 資料庫（或 EC2 執行個體或 Amazon RDS for SQL Server 資料庫上的 Microsoft SQL Server)
- AWS 上的虛擬私有雲端 (VPC) 與 AWS Direct Connect 或虛擬私有網路 (VPN) 提供的內部部署網路之間的網路連線

限制

- AWS DMS 3.4.7 之前的版本目前不支援啟用 VPC（閘道 VPC) S3 儲存貯體。
- 不支援在完全載入階段變更來源資料表結構。
- 不支援 AWS DMS 完整大型二進位物件 (LOB) 模式。

產品版本

- 適用於 Enterprise、Standard、Workgroup 和 Developer 版本的 Microsoft SQL Server 2005 版或更新版本。

- AWS DMS 3.3.2 版及更新版本提供 Microsoft SQL Server 2019 版做為來源的支援。

Architecture

來源技術堆疊

- 內部部署 Microsoft SQL Server 資料庫 （ 或 EC2 執行個體或 Amazon RDS for SQL Server 資料庫上的 Microsoft SQL Server)

目標技術堆疊

- AWS Direct Connect
- AWS DMS
- Amazon S3

目標架構

工具

- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。
- AWS Direct Connect 透過標準乙太網路光纖纜線，將您的內部網路連結至 Direct Connect 位置。透過此連線，您可以直接建立與公有 AWS 服務的虛擬介面，同時略過網路路徑中的網際網路服務供應商。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

史詩

準備遷移

任務	Description	所需的技能
驗證資料庫版本。	驗證來源資料庫版本，並確認 AWS DMS 支援該版本。如	DBA

任務	Description	所需的技能
	需有關支援的 SQL Server 資料庫版本的資訊，請參閱使用 Microsoft SQL Server 資料庫做為 AWS DMS 的來源。	
建立 VPC 和安全群組。	在您的 AWS 帳戶中，建立 VPC 和安全群組。如需詳細資訊，請參閱 Amazon VPC 文件。	系統管理員
為 AWS DMS 任務建立使用者。	在來源資料庫中建立 AWS DMS 使用者，並授予其讀取許可。AWS DMS 將使用此使用者。	DBA
測試資料庫連線。	從 AWS DMS 使用者測試 SQL Server 資料庫執行個體的連線。	DBA
建立 S3 儲存貯體。	建立目標 S3 儲存貯體。此儲存貯體將保留遷移的資料表資料。	系統管理員
建立 IAM 政策和角色。	1. 若要建立具有儲存貯體許可的 IAM 政策，請使用其他資訊區段中的程式碼。 2. 建立 AWS DMS 的角色，並將政策連接至角色。	系統管理員

使用 AWS DMS 遷移資料

任務	Description	所需的技能
建立 AWS DMS 複寫執行個體。	登入 AWS 管理主控台，然後開啟 AWS DMS 主控台。在導覽窗格中，選擇複寫執行個	DBA

任務	Description	所需的技能
	體、建立複寫執行個體。如需說明，請參閱 AWS DMS 文件中的步驟 1。	
建立來源和目標端點。	建立來源和目標端點。測試從複寫執行個體到來源和目標端點的連線。如需說明，請參閱 AWS DMS 文件中的步驟 2。	DBA
建立複寫任務。	建立複寫任務，然後選取具有變更資料擷取 (CDC) 的完全載入或完全載入，將資料從 SQL Server 遷移至 S3 儲存貯體。如需說明，請參閱 AWS DMS 文件中的步驟 3。	DBA
啟動資料複寫。	啟動複寫任務，並監控日誌是否有任何錯誤。	DBA

驗證資料

任務	Description	所需的技能
驗證遷移的資料。	在 主控台上，導覽至您的目標 S3 儲存貯體。開啟與來源資料庫同名的子資料夾。確認資料夾包含從來源資料庫遷移的所有資料表。	DBA

清除資源

任務	Description	所需的技能
關閉並刪除臨時 AWS 資源。	關閉您為資料遷移建立的臨時 AWS 資源，例如 AWS DMS	DBA

任務	Description	所需的技能
	複寫執行個體，並在驗證匯出後將其刪除。	

相關資源

- AWS Database Migration Service 使用者指南
- 使用 Microsoft SQL Server 資料庫做為 AWS DMS 的來源
- 使用 Amazon S3 做為 AWS Database Migration Service 的目標
- 使用 S3 儲存貯體做為 AWS DMS 目標 (AWS re：Post)

其他資訊

使用下列程式碼，為 AWS DMS 角色新增具有 S3 儲存貯體許可的 IAM 政策。用您的儲存貯體名稱取代 `bucketname`。

```
{
     "Version": "2012-10-17",
     "Statement": [
         {
             "Effect": "Allow",
             "Action": [
                 "s3:PutObject",
                 "s3:DeleteObject"
             ],
             "Resource": [
                 "arn:aws:s3:::bucketname*"
             ]
         },
         {
             "Effect": "Allow",
             "Action": [
                 "s3:ListBucket"
             ],
             "Resource": [
                 "arn:aws:s3:::bucketname*"
             ]
         }
     ]
```

```
}
```

將 Oracle CLOB 值遷移至 AWS 上的 PostgreSQL 中的個別資料列

Sai Krishna Namburu 和 Sindhusha Paturu，Amazon Web Services

總結

此模式說明如何將 Oracle 字元大型物件 (CLOB) 值分割為適用於 PostgreSQL 的 Amazon Aurora PostgreSQL 相容版本和 Amazon Relational Database Service (Amazon RDS) PostgreSQL 中的個別資料列。PostgreSQL 不支援 CLOB 資料類型。

具有間隔分割區的資料表會在來源 Oracle 資料庫中識別，而資料表名稱、分割區類型、分割區的間隔，以及其他中繼資料會擷取並載入目標資料庫。您可以使用 AWS Database Migration Service (AWS DMS) 將大小小於 1 GB 的 CLOB 資料以文字形式載入目標資料表，也可以匯出 CSV 格式的資料、將其載入 Amazon Simple Storage Service (Amazon S3) 儲存貯體，然後將其遷移至目標 PostgreSQL 資料庫。

遷移後，您可以使用此模式隨附的自訂 PostgreSQL 程式碼，根據新的行字元識別符 (`CHR(10)`) 將 CLOB 資料分割為個別資料列，並填入目標資料表。

先決條件和限制

先決條件

- 具有間隔分割區和 CLOB 資料類型記錄的 Oracle 資料庫資料表。
- Aurora PostgreSQL 相容或 Amazon RDS for PostgreSQL 資料庫，其資料表結構類似於來源資料表（相同的資料欄和資料類型）。

限制

- CLOB 值不能超過 1 GB。
- 目標資料表中的每一列都必須有新的行字元識別符。

產品版本

- Oracle 12c
- Aurora Postgres 11.6

Architecture

下圖顯示具有 CLOB 資料的來源 Oracle 資料表，以及 Aurora PostgreSQL 相容 11.6 版中的同等 PostgreSQL 資料表。

工具

AWS 服務

- Amazon Aurora PostgreSQL 相容版本是完全受管且符合 ACID 規範的關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。
- 適用於 PostgreSQL 的 Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展 PostgreSQL 關聯式資料庫。
- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他工具

您可以使用下列用戶端工具來連接、存取和管理 Aurora PostgreSQL 相容和 Amazon RDS for PostgreSQL 資料庫。（這些工具不會在此模式中使用。）

- pgAdmin 是 PostgreSQL 的開放原始碼管理工具。它提供圖形界面，可協助您建立、維護和使用資料庫物件。
- DBeaver 是開發人員和資料庫管理員的開放原始碼資料庫工具。您可以使用 工具來操作、監控、分析、管理和遷移資料。

最佳實務

如需將資料庫從 Oracle 遷移至 PostgreSQL 的最佳實務，請參閱 AWS 部落格文章將 Oracle 資料庫遷移至 Amazon RDS PostgreSQL 或 Amazon Aurora PostgreSQL 的最佳實務：遷移程序和基礎設施考量。

如需設定 AWS DMS 任務以遷移大型二進位物件的最佳實務，請參閱 AWS DMS 文件中的遷移大型二進位物件 (LOBs)。

史詩

識別 CLOB 資料

任務	Description	所需的技能
分析 CLOB 資料。	在來源 Oracle 資料庫中，分析 CLOB 資料，以查看它是否包含資料欄標頭，讓您可以判斷將資料載入目標資料表的方法。 若要分析輸入資料，請使用下列查詢。 `SELECT * FROM clobdata_or;`	開發人員
將 CLOB 資料載入目標資料庫。	將具有 CLOB 資料的資料表遷移至 Aurora 或 Amazon RDS 目標資料庫中的臨時（預備）資料表。您可以使用 AWS DMS 或將資料作為 CSV 檔案上傳至 Amazon S3 儲存貯體。 如需針對此任務使用 AWS DMS 的詳細資訊，請參閱 AWS DMS 文件中的使用 Oracle 資料庫做為來源和使用 PostgreSQL 資料庫做為目標。 如需針對此任務使用 Amazon S3 的詳細資訊，請參閱 AWS DMS 文件中的使用 Amazon S3 做為目標。	遷移工程師，DBA

任務	Description	所需的技能

驗證目標 PostgreSQL 資料表。	使用目標資料庫中的下列查詢，針對來源資料驗證目標資料，包括標頭。
	`SELECT * FROM clobdata_pg;`
	`SELECT * FROM clobdatatarget;`
	將結果與來源資料庫的查詢結果進行比較（從第一個步驟）。
		開發人員

將 CLOB 資料分割成不同的資料列。	執行其他資訊區段中提供的自訂 PostgreSQL 程式碼，以分割 CLOB 資料，並將其插入目標 PostgreSQL 資料表中的個別資料列。	開發人員

驗證資料。

任務	Description	所需的技能

驗證目標資料表中的資料。	使用以下查詢驗證插入目標資料表的資料。
	`SELECT * FROM clobdata_pg;`
	`SELECT * FROM clobdatatarget;`
		開發人員

相關資源

- CLOB 資料類型 (Oracle 文件)

- 資料類型 (PostgreSQL 文件)

其他資訊

用於分割 CLOB 資料的 PostgreSQL 函數

```
do
$$
declare
totalstr varchar;
str1 varchar;
str2 varchar;
pos1 integer := 1;
pos2 integer ;
len integer;


begin
        select rawdata||chr(10) into totalstr from clobdata_pg;
        len :=  length(totalstr) ;
        raise notice 'Total length : %',len;
        raise notice 'totalstr : %',totalstr;
        raise notice 'Before while loop';


        while pos1 < len  loop


                  select position (chr(10) in totalstr) into pos2;
                 raise notice '1st position of new line : %',pos2;


                str1 := substring (totalstr,pos1,pos2-1);
                raise notice 'str1 : %',str1;


                   insert into clobdatatarget(data) values (str1);
                   totalstr := substring(totalstr,pos2+1,len);
                   raise notice 'new totalstr :%',totalstr;
                len :=  length(totalstr) ;
```

```
        end loop;
end
$$
LANGUAGE 'plpgsql' ;
```

輸入和輸出範例

您可以使用下列範例，在遷移資料之前嘗試 PostgreSQL 程式碼。

建立具有三個輸入列的 Oracle 資料庫。

```
CREATE TABLE clobdata_or (
id INTEGER GENERATED ALWAYS AS IDENTITY,
rawdata clob  );


insert into clobdata_or(rawdata) values (to_clob('test line 1') || chr(10) ||
 to_clob('test line 2') || chr(10) || to_clob('test line 3') || chr(10));
COMMIT;


SELECT * FROM clobdata_or;
```

這會顯示下列輸出。

id	原始資料
1	測試列 1 測試列 2 測試列 3

將來源資料載入 PostgreSQL 預備資料表 (clobdata_pg) 進行處理。

```
SELECT * FROM clobdata_pg;

CREATE TEMP TABLE clobdatatarget (id1 SERIAL,data VARCHAR );

<Run the code in the additional information section.>

SELECT * FROM clobdatatarget;
```

這會顯示下列輸出。

id1	data
1	測試列 1
2	測試列 2
3	測試列 3

透過資料庫連結使用直接 Oracle Data Pump Import，將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle

Rizwan Wangde，Amazon Web Services

總結

許多模式涵蓋使用 Oracle Data Pump 將內部部署 Oracle 資料庫遷移至 Amazon Relational Database Service (Amazon RDS) for Oracle，這是遷移大型 Oracle 工作負載的首選方式。這些模式通常涉及將應用程式結構描述或資料表匯出至傾印檔案、將傾印檔案傳輸至 Amazon RDS for Oracle 上的資料庫目錄，然後從傾印檔案匯入應用程式結構描述和資料。

使用該方法，根據資料的大小和將傾印檔案傳輸至 Amazon RDS 執行個體所需的時間，遷移可能需要更長的時間。此外，傾印檔案位於 Amazon RDS 執行個體的 Amazon Elastic Block Store (Amazon EBS) 磁碟區上，其大小必須足以容納資料庫和傾印檔案。匯入後刪除傾印檔案時，無法擷取空白空間，因此您繼續支付未使用的空間。

此模式可透過透過資料庫連結使用 Oracle Data Pump API (DBMS_DATAPUMP)，在 Amazon RDS 執行個體上執行直接匯入來緩解這些問題。模式會在來源和目標資料庫之間啟動同時匯出和匯入管道。此模式不需要調整傾印檔案的 EBS 磁碟區大小，因為磁碟區上不會建立或存放傾印檔案。此方法 可節省未使用的磁碟空間的每月成本。

先決條件和限制

先決條件

- 作用中的 Amazon Web Services (AWS) 帳戶。
- 在至少兩個可用區域中設定私有子網路的虛擬私有雲端 (VPC)，以提供 Amazon RDS 執行個體的網路基礎設施。
- 內部部署資料中心中的 Oracle 資料庫，或在 Amazon Elastic Compute Cloud (Amazon EC2) 上自我管理。
- 單一可用區域中現有的 Amazon RDS for Oracle 執行個體。使用單一可用區域可改善遷移期間的寫入效能。異地同步備份部署可以在切換前 24-48 小時啟用。

  此解決方案也可以使用 Amazon RDS Custom for Oracle 做為目標。
- AWS Direct Connect （建議用於大型資料庫）。
- 內部部署的網路連線和防火牆規則，設定為允許從 Amazon RDS 執行個體到內部部署 Oracle 資料庫的傳入連線。

限制

- 截至 2022 年 12 月，Amazon RDS for Oracle 的資料庫大小限制為 64 TB (TiB)。
- Amazon RDS for Oracle 資料庫執行個體上單一檔案的大小上限為 16 TiB。這很重要，因為您可能需要將資料表分散到多個資料表空間。

產品版本

- 來源資料庫：Oracle Database 10g 版本 1 和更新版本。
- 目標資料庫：如需 Amazon RDS 上支援版本的最新清單，請參閱 AWS 文件中的 Amazon RDS for Oracle。

Architecture

來源技術堆疊

- 內部部署或雲端中的自我管理 Oracle 資料庫

目標技術堆疊

- Amazon RDS for Oracle 或 Amazon RDS Custom for Oracle

目標架構

下圖顯示從內部部署 Oracle 資料庫遷移到單一可用區環境中 Amazon RDS for Oracle 的架構。箭頭方向描述架構中的資料流程。圖表不會顯示哪個元件正在啟動連線。

1. Amazon RDS for Oracle 執行個體會連線至內部部署來源 Oracle 資料庫，以透過資料庫連結執行完全載入遷移。
2. AWS Database Migration Service (AWS DMS) 會連線至內部部署來源 Oracle 資料庫，以使用變更資料擷取 (CDC) 執行持續複寫。
3. CDC 變更會套用至 Amazon RDS for Oracle 資料庫。

工具

AWS 服務

- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移到 AWS 雲端 或在雲端和內部部署設定的組合之間遷移。此模式使用 CDC，且僅複寫資料變更設定。
- AWS Direct Connect 會透過標準乙太網路光纖纜線將您的內部網路連結至某個 Direct Connect 位置。透過此連線，您可以在網路路徑中略過網際網路服務提供者 AWS 服務 時，直接建立虛擬介面至公有。
- Amazon Relational Database Service 可協助您在 AWS 雲端中設定、操作和擴展 Oracle 關聯式資料庫。

其他工具

- Oracle Data Pump 可協助您以高速將資料和中繼資料從一個資料庫移至另一個資料庫。
- Oracle Instant Client 或 SQL Developer 等用戶端工具用於在資料庫上連接和執行 SQL 查詢。

最佳實務

雖然 Direct Connect 使用內部部署網路與 之間的專用私有網路連線 AWS，但請考慮下列選項，以取得傳輸中資料的額外安全性和資料加密：

- 使用 從內部部署網路到網路的虛擬私有網路 (VPN) AWS Site-to-Site VPN 或 IPsec VPN 連線 AWS
- 在內部部署 Oracle 資料庫上設定的 Oracle 資料庫原生網路加密
- 使用 TLS 加密

史詩

準備內部部署來源 Oracle 資料庫

任務	Description	所需的技能
設定從目標資料庫到來源資料庫的網路連線。	設定內部部署網路和防火牆，以允許從目標 Amazon RDS 執行個體傳入連線至內部部署來源 Oracle 資料庫。	網路管理員、安全工程師
建立具有適當權限的資料庫使用者。	在內部部署來源 Oracle 資料庫中建立資料庫使用者，具有使用 Oracle Data Pump 在來源和目標之間遷移資料的權限：	DBA

任務	Description	所需的技能

	`GRANT CONNECT to <migration_user>;`
	`GRANT DATAPUMP_EXP_FULL_DATABASE to <migration_user>;`
	`GRANT SELECT ANY TABLE to <migration_user>;`	

準備內部部署來源資料庫以進行 CDC AWS DMS 遷移。	（選用）在完成 Oracle Data Pump Full Load 之後，準備內部部署來源 Oracle 資料庫以進行 AWS DMS CDC 遷移：
	1. 設定在 Oracle Data Pump 遷移期間管理 FLASHBACK 所需的其他權限：
	`GRANT FLASHBACK ANY TABLE to <migration_user>;`
	`GRANT FLASHBACK ARCHIVE ADMINISTER to <migration_user>;`
	2. 若要設定自我管理 Oracle 來源所需的使用者帳戶權限 AWS DMS，請參閱 AWS DMS 文件。
	3. 若要使用 為 CDC 準備 Oracle 自我管理來源資料庫 AWS DMS，請參閱 AWS DMS 文件。
		DBA

安裝和設定 SQL Developer。	安裝並設定 SQL Developer 以連接和執行來源和目標資料庫上的 SQL 查詢。	DBA，遷移工程師

任務	Description	所需的技能
產生指令碼以建立資料表空間。	使用下列範例 SQL 查詢在來源資料庫上產生指令碼： `SELECT 'CREATE TABLESPACE ' tablespace_name ' DATAFILE SIZE 1G AUTOEXTEND ON MAXSIZE UNLIMITED;' from dba_tablespaces where tablespace_name not in ('SYSTEM', 'SYSAUX','TEMP','UNDOTBS1') order by 1;` 指令碼將套用至目標資料庫。	DBA
產生指令碼以建立使用者、設定檔、角色和權限。	若要產生指令碼來建立資料庫使用者、設定檔、角色和權限，請使用 Oracle 支援文件中的指令碼 如何使用 dbms_metadata.get_ddl (Doc ID 2739952.1) 擷取使用者的 DDL，包括權限和角色（需要 Oracle 帳戶）。 指令碼將套用至目標資料庫。	DBA

準備目標 Amazon RDS for Oracle 執行個體

任務	Description	所需的技能
建立來源資料庫的資料庫連結並驗證連線。	若要建立內部部署來源資料庫的資料庫連結，您可以使用下列範例命令：	DBA

任務	Description	所需的技能

	`CREATE DATABASE LINK link2src CONNECT TO <migration_user_account> IDENTIFIED BY <password> USING '(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=<dns or ip address of remote db>) (PORT=<listener port>))(CONNECT_DATA=(SID=<remote SID>)))';`
	
	若要驗證連線能力，請執行下列 SQL 命令：
	
	`select * from dual@link2src;`
	
	如果回應為 ，連線會成功X。	

執行指令碼以準備目標執行個體。	執行先前產生的指令碼以準備目標 Amazon RDS for Oracle 執行個體： 1. 資料表空間 2. 個人檔案 3. 角色 這有助於確保 Oracle Data Pump 遷移可以建立結構描述及其物件。	DBA，遷移工程師

透過資料庫連結使用 Oracle Data Pump Import 執行完全載入遷移

任務	Description	所需的技能
遷移所需的結構描述。	若要將所需結構描述從來源現場部署資料庫遷移至目標 Amazon RDS 執行個體，請使用其他資訊區段中的程式碼： • 若要遷移單一結構描述，請從其他資訊區段執行程式碼 1。 • 若要遷移多個結構描述，請從其他資訊區段執行程式碼 2。 若要調整遷移的效能，您可以執行下列命令來調整平行程序的數量： `DBMS_DATAPUMP.SET_ PARALLEL (handle => v_hdnl, degree => 4);`	DBA
收集結構描述統計資料以改善效能。	收集結構描述統計資料命令會傳回針對資料庫物件收集的 Oracle 查詢最佳化工具統計資料。透過使用此資訊，最佳化工具可以針對這些物件選取任何查詢的最佳執行計畫： `EXECUTE DBMS_STAT S.GATHER_SCHEMA_ST ATS(ownname => '<schema_name>');`	DBA

使用 Oracle Data Pump 和 執行完全載入遷移和 CDC 複寫 AWS DMS

任務	Description	所需的技能
在來源現場部署 Oracle 資料庫上擷取 SCN。	擷取來源內部部署 Oracle 資料庫上的系統變更編號 (SCN)。您將使用 SCN 進行完全載入匯入，並將 做為 CDC 複寫的起點。 若要在來源資料庫上產生目前的 SCN，請執行下列 SQL 陳述式： `SELECT current_scn FROM V$DATABASE;`	DBA
執行結構描述的完全載入遷移。	若要將所需的結構描述 (FULL LOAD) 從來源現場部署資料庫遷移至目標 Amazon RDS 執行個體，請執行下列動作： • 若要遷移單一結構描述，請從其他資訊區段執行程式碼 3。 • 若要遷移多個結構描述，請從其他資訊區段執行程式碼 4。 在程式碼中，將 <CURRENT_SCN_VALUE_IN_SOURCE_DATABASE> 取代為您從來源資料庫擷取的 SCN： `DBMS_DATAPUMP.SET_PARAMETER (handle => v_hdnl, name => 'FLASHBACK_SCN', value`	DBA

任務	Description	所需的技能
	=> <CURRENT_SCN_VALUE _IN_SOURCE_DATABAS E>); 若要調整遷移的效能，您可以調整平行程序的數量： DBMS_DATAPUMP.SET_ PARALLEL (handle => v_hdnl, degree => 4);	
停用遷移結構描述下的觸發條件。	開始僅限 AWS DMS CDC 任務之前，請在遷移的結構描述TRIGGERS下停用 。	DBA
收集結構描述統計資料以改善效能。	收集結構描述統計資料命令會傳回針對資料庫物件收集的 Oracle 查詢最佳化工具統計資料： EXECUTE DBMS_STAT S.GATHER_SCHEMA_ST ATS(ownname => '<schema_name>'); 透過使用此資訊，最佳化工具可以為針對這些物件的任何查詢選取最佳執行計畫。	DBA

任務	Description	所需的技能
使用 AWS DMS 執行從來源到目標的持續複寫。	使用 AWS DMS 執行從來源 Oracle 資料庫到目標 Amazon RDS for Oracle 執行個體的持續複寫。 如需詳細資訊，請參閱使用建立持續複寫的任務 AWS DMS，以及部落格文章如何使用 中的原生 CDC 支援 AWS DMS。	DBA，遷移工程師

切換到 Amazon RDS for Oracle

任務	Description	所需的技能
在切換前 48 小時啟用執行個體上的異地同步備份。	如果這是生產執行個體，建議您在 Amazon RDS 執行個體上啟用異地同步備份部署，以提供高可用性 (HA) 和災難復原 (DR) 的優點。	DBA，遷移工程師
停止僅限 AWS DMS CDC 的任務 （ 如果已開啟 CDC)。	1. 確保 AWS DMS 任務的 Amazon CloudWatch 指標上的來源延遲和目標延遲顯示 0 秒。 2. 停止僅限 AWS DMS CDC 的任務。	DBA
啟用觸發條件。	啟用您在建立 `TRIGGERS` CDC 任務之前停用的 。	DBA

相關資源

AWS

- 使用 為 CDC 準備 Oracle 自我管理來源資料庫 AWS DMS
- 使用 建立持續複寫的任務 AWS DMS
- 高可用性的異地同步備份部署
- 如何在 中使用原生 CDC 支援 AWS DMS （部落格文章）

Oracle 文件

- DBMS_DATAPUMP

其他資訊

程式碼 1：僅限完全載入遷移，單一應用程式結構描述

```
DECLARE
    v_hdnl NUMBER;
BEGIN
    v_hdnl := DBMS_DATAPUMP.OPEN(operation => 'IMPORT', job_mode  => 'SCHEMA',
 remote_link => '<DB LINK Name to Source Database>', job_name  => null);
    DBMS_DATAPUMP.ADD_FILE( handle => v_hdnl, filename  => 'import_01.log',  directory
 => 'DATA_PUMP_DIR', filetype  => dbms_datapump.ku$_file_type_log_file);
    DBMS_DATAPUMP.METADATA_FILTER(v_hdnl,'SCHEMA_EXPR','IN (''<schema_name>'')');  --
 To migrate one selected schema
    DBMS_DATAPUMP.METADATA_FILTER (hdnl, 'EXCLUDE_PATH_EXPR','IN (''STATISTICS'')'); --
 To prevent gathering Statistics during the import
    DBMS_DATAPUMP.SET_PARALLEL (handle => v_hdnl, degree => 4);  -- Number of parallel
 processes performing export and import
    DBMS_DATAPUMP.START_JOB(v_hdnl);
END;
/
```

程式碼 2：僅限完全載入遷移，多個應用程式結構描述

```
DECLARE
    v_hdnl NUMBER;
BEGIN
    v_hdnl := DBMS_DATAPUMP.OPEN(operation => 'IMPORT', job_mode  => 'SCHEMA',
 remote_link => '<DB LINK Name to Source Database>', job_name  => null);
    DBMS_DATAPUMP.ADD_FILE( handle => v_hdnl, filename  => 'import_01.log',  directory
 => 'DATA_PUMP_DIR', filetype  => dbms_datapump.ku$_file_type_log_file);
```

```
    DBMS_DATAPUMP.METADATA_FILTER (v_hdnl, 'SCHEMA_LIST',
 '''<SCHEMA_1>'',''<SCHEMA_2>'', ''<SCHEMA_3>'''); -- To migrate multiple schemas
    DBMS_DATAPUMP.METADATA_FILTER (v_hdnl, 'EXCLUDE_PATH_EXPR','IN (''STATISTICS'')');
 -- To prevent gathering Statistics during the import
    DBMS_DATAPUMP.SET_PARALLEL (handle => v_hdnl, degree => 4);  -- Number of parallel
 processes performing export and import
    DBMS_DATAPUMP.START_JOB(v_hdnl);
END;
/
```

程式碼 3：僅在 CDC 任務之前進行完全載入遷移，單一應用程式結構描述

```
DECLARE
    v_hdnl NUMBER;
BEGIN
    v_hdnl := DBMS_DATAPUMP.OPEN(operation => 'IMPORT', job_mode  => 'SCHEMA',
 remote_link => '<DB LINK Name to Source Database>', job_name  => null);
    DBMS_DATAPUMP.ADD_FILE( handle => v_hdnl, filename  => 'import_01.log',  directory
 => 'DATA_PUMP_DIR', filetype  => dbms_datapump.ku$_file_type_log_file);
    DBMS_DATAPUMP.METADATA_FILTER(v_hdnl,'SCHEMA_EXPR','IN (''<schema_name>'')');  --
 To migrate one selected schema
    DBMS_DATAPUMP.METADATA_FILTER (v_hdnl, 'EXCLUDE_PATH_EXPR','IN (''STATISTICS'')');
 -- To prevent gathering Statistics during the import
    DBMS_DATAPUMP.SET_PARAMETER (handle => v_hdnl, name => 'FLASHBACK_SCN', value =>
 <CURRENT_SCN_VALUE_IN_SOURCE_DATABASE>); -- SCN required for AWS DMS CDC only task.
    DBMS_DATAPUMP.SET_PARALLEL (handle => v_hdnl, degree => 4);  -- Number of parallel
 processes performing export and import
    DBMS_DATAPUMP.START_JOB(v_hdnl);
END;
/
```

程式碼 4：僅限 CDC 任務之前的完全載入遷移，多個應用程式結構描述

```
DECLARE
    v_hdnl NUMBER;
BEGIN
    v_hdnl := DBMS_DATAPUMP.OPEN (operation => 'IMPORT', job_mode  => 'SCHEMA',
 remote_link => '<DB LINK Name to Source Database>', job_name  => null);
    DBMS_DATAPUMP.ADD_FILE (handle => v_hdnl, filename  => 'import_01.log',  directory
 => 'DATA_PUMP_DIR', filetype  => dbms_datapump.ku$_file_type_log_file);
    DBMS_DATAPUMP.METADATA_FILTER (v_hdnl, 'SCHEMA_LIST',
 '''<SCHEMA_1>'',''<SCHEMA_2>'', ''<SCHEMA_3>'''); -- To migrate multiple schemas
```

```
    DBMS_DATAPUMP.METADATA_FILTER (v_hdnl, 'EXCLUDE_PATH_EXPR','IN (''STATISTICS'')');
 -- To prevent gathering Statistics during the import
    DBMS_DATAPUMP.SET_PARAMETER (handle => v_hdnl, name => 'FLASHBACK_SCN', value =>
 <CURRENT_SCN_VALUE_IN_SOURCE_DATABASE>); -- SCN required for AWS DMS CDC only task.
    DBMS_DATAPUMP.SET_PARALLEL (handle => v_hdnl, degree => 4);  -- Number of parallel
 processes performing export and import
    DBMS_DATAPUMP.START_JOB(v_hdnl);
END;
/
```

混合遷移方法可以更好地運作的案例

在來源資料庫包含數百萬資料列和非常大型 LOBSEGMENT 資料欄的資料表極少數情況下，此模式會減慢遷移速度。Oracle 透過網路連結逐一遷移 LOBSEGMENTs。它會從來源資料表擷取單一資料列（以及 LOB 資料欄資料），並將資料列插入目標資料表，重複此程序，直到遷移所有資料列為止。透過資料庫連結的 Oracle Data Pump 不支援 LOBSEGMENTs的大量載入或直接路徑載入機制。

在這種情況下，我們建議下列事項：

- 透過新增下列中繼資料篩選條件，在 Oracle Data Pump 遷移期間略過已識別的資料表：

```
dbms_datapump.metadata_filter(handle =>h1, name=>'NAME_EXPR', value => 'NOT IN
 (''TABLE_1'',''TABLE_2'')');
```

- 使用 AWS DMS 任務（完整載入遷移，視需要搭配 CDC 複寫）來遷移已識別的資料表。 AWS DMS 會從來源 Oracle 資料庫擷取多個資料列，並將其批次插入目標 Amazon RDS 執行個體，以改善效能。

將 Oracle PeopleSoft 遷移至 Amazon RDS Custom

Gaurav Gupta，Amazon Web Services

摘要

Oracle PeopleSoft 是適用於整個企業程序的企業資源規劃 (ERP) 解決方案。PeopleSoft 具有三層架構：用戶端、應用程式和資料庫。PeopleSoft 可以在 Amazon Relational Database Service (Amazon RDS) 上執行。現在，您也可以在 Amazon RDS Custom 上執行 PeopleSoft，這可讓您存取基礎作業系統。

Amazon RDS Custom for Oracle 是一種受管資料庫服務，適用於需要存取基礎作業系統和資料庫環境的舊版、自訂和封裝應用程式。當您將 Oracle 資料庫遷移至 Amazon RDS Custom 時，Amazon Web Services (AWS) 可以管理備份任務和高可用性，同時可以專注於維護 PeopleSoft 應用程式和功能。如需考慮遷移的關鍵因素，請參閱 AWS 方案指引中的 Oracle 資料庫遷移策略。

此模式著重於使用 Oracle Recovery Manager (RMAN) 備份，將 Amazon Elastic Compute Cloud (Amazon EC2) 上的 PeopleSoft 資料庫遷移至 Amazon RDS Custom 的步驟。它在 EC2 執行個體和 Amazon RDS Custom 之間使用 Amazon Elastic File System (Amazon EFS) 共用檔案系統，但您也可以使用 Amazon FSx 或任何共用磁碟機。模式使用 RMAN 完整備份 （有時稱為層級 0 備份）。

先決條件和限制

先決條件

- 使用 Oracle Linux 7、Oracle Linux 8、Red Hat Enterprise Linux (RHEL) 7 或 RHEL 8 在 Amazon EC2 上執行的 Oracle 19C 版來源資料庫。在此模式範例中，來源資料庫名稱為 `FSDM092`，但這並非必要項目。

> **Note**
>
> 您也可以將此模式與現場部署 Oracle 來源資料庫搭配使用。您必須擁有內部部署網路與虛擬私有雲端 (VPC) 之間的適當網路連線。

- PeopleSoft 9.2 示範執行個體。
- 單一 PeopleSoft 應用程式層。不過，您可以調整此模式以使用多個應用程式層。
- Amazon RDS Custom 已設定至少 8 GB 的交換空間。

限制

此模式不支援下列組態：

- 將資料庫ARCHIVE_LAG_TARGET參數設定為 60–7200 範圍以外的值
- 停用資料庫執行個體日誌模式 (NOARCHIVELOG)
- 關閉 EC2 執行個體的 Amazon Elastic Block Store (Amazon EBS) 最佳化屬性
- 修改連接至 EC2 執行個體的原始 EBS 磁碟區
- 新增 EBS 磁碟區，或將磁碟區類型從 gp2 變更為 gp3
- 變更 LOG_ARCHIVE_FORMAT 參數的延伸格式（需要 *.arc)
- 多工或變更控制檔案位置和名稱（必須是 /rdsdbdata/db/*DBNAME*/controlfile/control-01.ctl)

如需有關這些和其他不支援組態的其他資訊，請參閱 Amazon RDS 文件。

產品版本

對於 Amazon RDS Custom 支援的 Oracle 資料庫版本和執行個體類別，請參閱 Amazon RDS Custom for Oracle 的需求和限制。

Architecture

目標技術堆疊

- Application Load Balancer
- Amazon EFS
- Amazon RDS Custom for Oracle
- AWS Secrets Manager
- Amazon Simple Storage Service (Amazon S3)

目標架構

下列架構圖代表在 AWS 上單一可用區域中執行的 PeopleSoft 系統。應用程式層是透過 Application Load Balancer 存取。應用程式和資料庫都位於私有子網路中，Amazon RDS Custom 和 Amazon EC2 資料庫執行個體會使用 Amazon EFS 共用檔案系統來存放和存取 RMAN 備份檔案。Amazon S3 用於建立自訂 RDS Oracle 引擎和儲存重做日誌中繼資料。

工具

工具

AWS 服務

- Amazon RDS Custom for Oracle 是一種受管資料庫服務，適用於需要存取基礎作業系統和資料庫環境的舊版、自訂和封裝應用程式。它可自動化資料庫管理任務，例如備份和高可用性。
- Amazon Elastic File System (Amazon EFS) 可協助您在 AWS 雲端中建立和設定共用檔案系統。此模式使用 Amazon EFS 共用檔案系統來存放和存取 RMAN 備份檔案。
- AWS Secrets Manager 可協助您使用以程式設計方式擷取秘密的 API 呼叫取代程式碼中的硬式編碼登入資料，包括密碼。在此模式中，您會從 Secrets Manager 擷取資料庫使用者密碼，以建立 `RDSADMIN`和 `ADMIN`使用者，以及變更 `sys`和 `system`密碼。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Elastic Load Balancing (ELB) 會將傳入的應用程式或網路流量分散到多個目標。例如，您可以將流量分散到一或多個可用區域中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、容器和 IP 地址。此模式使用 Application Load Balancer。

其他工具

- Oracle Recovery Manager (RMAN) 提供 Oracle 資料庫的備份和復原支援。此模式使用 RMAN 在 Amazon EC2 上執行在 Amazon RDS Custom 上還原的來源 Oracle 資料庫熱備份。

最佳實務

- 對於資料庫初始化參數，自訂 Amazon RDS Custom 資料庫執行個體為 PeopleSoft 提供的標準 pfile，而不是使用 Oracle 來源資料庫中的 spfile。這是因為在 Amazon RDS Custom 中建立僅供讀取複本時，空格和註解會導致問題。如需資料庫初始化參數的詳細資訊，請參閱 Oracle Support Note 1100831.1（需要 Oracle Support 帳戶）。
- 根據預設，Amazon RDS Custom 會使用 Oracle 自動記憶體管理。如果您想要使用 Hugemem 核心，您可以將 Amazon RDS Custom 設定為改用自動共用記憶體管理。
- 預設為啟用 `memory_max_target` 參數。框架會在背景使用此項目來建立僅供讀取複本。
- 啟用 Oracle Flashback 資料庫。此功能在容錯移轉（非切換）測試案例中恢復待命時非常有用。

史詩

設定資料庫執行個體和檔案系統

任務	Description	所需的技能
建立資料庫執行個體。	在 Amazon RDS 主控台中，使用稱為 FSDMO92（或您的來源資料庫名稱）的資料庫名稱建立 Amazon RDS Custom for Oracle 資料庫執行個體。 如需說明，請參閱 AWS 文件中的使用 Amazon RDS Custom 和資料庫環境部落格文章中的 Amazon RDS Custom for Oracle – 新控制功能。這可確保資料庫名稱設定為與來源資料庫相同的名稱。（如果保持空白，EC2 執行個體和資料庫名稱將設定為 `ORCL`。）	DBA

執行來源 Amazon EC2 資料庫的 RMAN 完整備份

任務	Description	所需的技能

建立備份指令碼。	建立 RMAN 備份指令碼，將資料庫備份到您掛載的 Amazon EFS 檔案系統 (`/efs`在下列範例中）。您可以使用範例程式碼或執行其中一個現有的 RMAN 指令碼。
	
	`#!/bin/bash`
	`Dt=`date +'%Y%m%d-%H%M'`
		DBA

任務	Description	所需的技能
	BACKUP_LOG="rman-${O RACLE_SID}-$Dt" export TAGDATE=`date + %Y%m%d%H%M`; LOGPATH=/u01/scripts/l ogs rman target / >> $LOGPATH/rman-${OR ACLE_SID}-$Dt << EOF SQL "ALTER SYSTEM SWITCH LOGFILE"; SQL "ALTER SESSION SET NLS_DATE_FORMAT="D D.MM.YYYY HH24:MI:S S""; RUN { ALLOCATE CHANNEL ch11 TYPE DISK MAXPIECESIZE 5G; ALLOCATE CHANNEL ch12 TYPE DISK MAXPIECESIZE 5G; BACKUP AS COMPRESSED BACKUPSET FULL DATABASE FORMAT '/efs/rma n_backup/FSCM/%d_%T_ %s_%p_FULL' ; SQL "ALTER SYSTEM ARCHIVE LOG CURRENT"; BACKUP FORMAT '/efs/ rman_backup/FSCM/%d_ %T_%s_%p_ARCHIVE' ARCHIVELOG ALL DELETE ALL INPUT ; BACKUP CURRENT CONTROLFILE FORMAT '/ efs/rman_backup/FSCM/ %d_%T_%s_%p_CONTROL'; } EXIT; EOF	

任務	Description	所需的技能

執行備份指令碼。	若要執行 RMAN 備份指令碼，請以 Oracle Home User 身分登入，然後執行指令碼。
	
	`$  chmod a+x rman_backup.sh`
	`$ ./rman_backup.sh &`
		DBA

任務	Description	所需的技能

檢查是否有錯誤，並記下備份檔案的名稱。	檢查 RMAN 日誌檔案是否有錯誤。如果一切正常，請執行下列命令列出控制檔案的備份。
	`RMAN> list backup of controlfile;`
	`using target database control file instead of recovery catalog`
	請記下輸出檔案的名稱。
	`List of Backup Sets`
	`==========`
	`BS Key Type LV Size Device Type Elapsed Time Completion Time`
	`12 Full 21.58M DISK 00:00:01 13-JUL-22`
	`BP Key: 12 Status: AVAILABLE Compressed: NO Tag: TAG20220713T150155`
	`Piece Name: /efs/rman_backup/FSCM/FSDMO92_20220713_12_1_CONTROL`
	`Control File Included: Ckp SCN: 16559159985898 Ckp time: 13-JUL-22`
		DBA

任務	Description	所需的技能
	當您在 Amazon RDS Custom 上還原資料庫`/efs/rman_backup/FSCM/FSDMO92_20220713_12_1_CONTROL` 時，將使用備份控制檔案。	

關閉來源應用程式層

任務	Description	所需的技能

關閉應用程式。	若要關閉來源應用程式層，請使用 `psadmin`公用程式或`psadmin`命令列公用程式。
	1. 若要關閉 Web 伺服器，請執行下列命令。
	`psadmin -w shutdown -d "webserver domain name"`
	2. 若要關閉應用程式伺服器，請執行下列命令。
	`psadmin -c shutdown -d "application server domain name"`
	3. 若要關閉程序排程器，請執行下列命令。
	`psadmin -p stop -d "process scheduler domain name"`
		DBA，PeopleSoft 管理員

設定目標 Amazon RDS Custom 資料庫

任務	Description	所需的技能
安裝 nfs-utils rpm 套件。	若要安裝`nfs-utils rpm`套件，請執行下列命令。 `$ yum install -y nfs-utils`	DBA
掛載 EFS 儲存體。	從 Amazon EFS 主控台頁面取得 Amazon EFS 掛載命令。使用網路檔案系統 (NFS) 用戶端在 Amazon RDS 執行個體上掛載 EFS 檔案系統。 `sudo mount -t nfs4 -o nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport fs-xxxxxxxxxx.efs.eu-west-1.amazonaws.com:/ /efs` `sudo mount -t nfs4 -o nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport fs-xxxxxxxxxx.efs.eu-west-1.amazonaws.com:/ /efs`	DBA

捨棄入門資料庫並建立目錄以存放資料庫檔案

任務	Description	所需的技能
暫停自動化模式。	您必須先暫停 Amazon RDS Custom 資料庫執行個體上的自動化模式，才能繼續後續步驟，以確保自動化不會干擾 RMAN 還原活動。 您可以使用 AWS 主控台或 AWS 命令列界面 (AWS CLI) 命令 （請確定您已先設定 AWS CLI) 來暫停自動化。 `aws rds modify-db-instance \ --db-instance-id entifier peoplesoft-fscm-92 \ --automation-mode all-paused \ --resume-full-au tomation-mode-minute 360 \ --region eu-west-1` 當您指定暫停的持續時間時，請確定您有足夠的時間進行 RMAN 還原。這取決於來源資料庫的大小，因此請相應地修改 360 值。 此外，請確定暫停自動化的總時間不會與資料庫的備份或維護時段重疊。	DBA
建立和修改 PeopleSoft 的參數檔案	若要建立和修改 PeopleSoft 的 pfile， 請使用使用 Amazon	DBA

任務	Description	所需的技能

	RDS Custom 資料庫執行個體建立的標準 pfile。新增 PeopleSoft 所需的參數。
	
	1. 執行下列命令`rds user rdsdb`以切換到 。
	
	``
	$ sudo su - rdsdb
	``
	
	2. 登入入門資料庫上的 SQL*Plus，並執行下列命令來建立 pfile。
	
	``
	SQL> create pfile
	 from spfile;
	``
	
	這會在 中建立 pfile`$ORACLE_HOME/dbs`。
	3. 建立此 pfile 的備份。
	4. 編輯 pfile 以新增或更新 PeopleSoft 參數。
	
	``
	*._gby_hash_aggreg
	ation_enabled=false
	
	*._unnest_subqu
	ery=false
	
	*.nls_language='
	AMERICAN'
	
	*.nls_length_sem
	antics='CHAR'
	
	*.nls_territ
	ory='AMERICA'
	``	

任務	Description	所需的技能
	*.open_cursors=1000 *.db_files=1200 *.undo_tablespace='UNDOTBS1' 您可以在 Oracle Support Note 1100831.1 中找到 PeopleSoft 相關參數。 https://support.oracle.com/ 5. 從 pfile 移除 spfile 參考。 *.spfile='/rdsdbbin/oracle/dbs/spfileFSDMO92.ora'	
捨棄入門資料庫。	若要捨棄現有的 Amazon RDS Custom 資料庫，請使用下列程式碼。 $ sqlplus / as sysdba SQL> shutdown immediate; SQL> startup mount exclusive restrict; SQL> drop database; SQL> exit	

任務	Description	所需的技能
從備份還原 Amazon RDS Custom 資料庫。	使用下列指令碼還原資料庫。指令碼會先還原控制檔案，然後從存放在 EFS 掛載上的備份片段還原整個資料庫。 `#!/bin/bash` `Dt=`date +'%Y%m%d-%H%M'` `BACKUP_LOG="rman-${ORACLE_SID}-$Dt"` `export TAGDATE=`date +%Y%m%d%H%M`;` `LOGPATH=/rdsdbdata/scripts/logs` `rman target / >> $LOGPATH/rman-${ORACLE_SID}-$Dt << EOF` `restore controlfile from "/efs/rman_backup/FSCM/FSDMO92_20220713_12_1_CONTROL";` `alter database mount;` `run` `{` `set newname for database to '/rdsdbdata/db/FSDMO92_A/datafile/%f_%b';` `SET NEWNAME FOR TEMPFILE 1 TO '/rdsdbdata/db/FSDMO92_A/datafile/%f_%b';` `RESTORE DATABASE;` `SWITCH DATAFILE ALL;` `SWITCH TEMPFILE ALL;` `RECOVER DATABASE;` `}` `EOF`	DBA

任務	Description	所需的技能
	sqlplus / as sysdba >> $LOGPATH/rman-${ORACLE_SID}-$Dt<<-EOF ALTER DATABASE RENAME FILE '/u01/psoft/db/oradata/FSDMO92/redo01.log' TO '/rdsdbdata/db/FSDMO92_A/onlinelog/redo01.log'; ALTER DATABASE RENAME FILE '/u01/psoft/db/oradata/FSDMO92/redo02.log' TO '/rdsdbdata/db/FSDMO92_A/onlinelog/redo02.log'; ALTER DATABASE RENAME FILE '/u01/psoft/db/oradata/FSDMO92/redo03.log' TO '/rdsdbdata/db/FSDMO92_A/onlinelog/redo03.log'; alter database clear unarchived logfile group 1; alter database clear unarchived logfile group 2; alter database clear unarchived logfile group 3; alter database open resetlogs; EXIT EOF	

從 Secrets Manager 擷取密碼、建立使用者和變更密碼

任務	Description	所需的技能
從 Secrets Manager 擷取密碼。	您可以使用 AWS 主控台或 AWS CLI 來執行此步驟。下列步驟顯示 主控台的指示。 1. 登入 AWS 管理主控台並開啟 Amazon RDS 主控台。 2. 在導覽窗格中，選擇資料庫，然後選取 Amazon RDS 資料庫。 3. 選擇組態索引標籤，並記下執行個體的資源 ID。其格式為 db-<ID> (例如 db-73GJNHLGDNZNDOXNWXSECUW6LE)。 4. 開啟 Secrets Manager 主控台。 5. 選擇與 同名的秘密do-not-delete-custom-<resource_id> ，其中 resource-id 是指您在步驟 3 中記下的資源 ID。 6. 選擇 Retrieve secret value (擷取秘密值)。 sys、、systemrdsadmin和admin使用者的此密碼將相同。	DBA
建立 RDSADMIN 使用者。	RDSADMIN 是用於監控和協調 Amazon RDS Custom 資料庫執行個體的資料庫使用者。由於啟動者資料庫已捨棄，且目	DBA

任務	Description	所需的技能

	標資料庫已使用 RMAN 從來源還原，因此您必須在還原操作後重新建立此使用者，以確保 Amazon RDS Custom 監控如預期般運作。您也必須為RDSADMIN使用者建立單獨的設定檔和資料表空間。
	
	1. 在 SQL 提示中輸入下列命令。
	
	`SQL> set echo on feedback on serverout on`
	`SQL> @?/rdbms/admin/utlpwdmg.sql`
	`SQL> ALTER PROFILE DEFAULT LIMIT FAILED_LOGIN_ ATTEMPTS UNLIMITED PASSWORD_LIFE_TIME UNLIMITED PASSWORD_VERIFY_F UNCTION NULL;`
	
	2. 建立設定檔 RDSADMIN。
	
	`SQL> set echo on feedback on serverout on`
	`SQL> alter session set "_oracle_script"=t rue;`
	`SQL> CREATE PROFILE RDSADMIN LIMIT COMPOSITE_LIMIT UNLIMITED`	

任務	Description	所需的技能
	SESSIONS_PER_USER UNLIMITED CPU_PER_SESSION UNLIMITED CPU_PER_CALL UNLIMITED LOGICAL_READS_PER _SESSION UNLIMITED LOGICAL_READS_PER _CALL UNLIMITED IDLE_TIME UNLIMITED CONNECT_TIME UNLIMITED PRIVATE_SGA UNLIMITED FAILED_LOGIN_ATTE MPTS 10 PASSWORD_LIFE_TIME UNLIMITED PASSWORD_REUSE_TIME UNLIMITED PASSWORD_REUSE_MAX UNLIMITED PASSWORD_VERIFY_F UNCTION NULL PASSWORD_LOCK_TIME 86400/86400 PASSWORD_GRACE_TIME 604800/86400; 3. 建立RDSADMIN資料表空間。 SQL> CREATE BIGFILE TABLESPACE rdsadmin '/rdsdbdata/db/FSD MO92_A/datafile/rd sadmin.dbf' DATAFILE SIZE 7M AUTOEXTEND ON NEXT 1m	

任務	Description	所需的技能
	LOGGING ONLINE PERMANENT BLOCKSIZE 8192 EXTENT MANAGEMEN T LOCAL AUTOALLOCATE DEFAULT NOCOMPRES S SEGMENT SPACE MANAGEMENT AUTO; 4. 建立RDSADMIN使用者。將RDSADMIN密碼取代為您先前從 Secrets Manager 取得的密碼。 SQL> CREATE USER rdsadmin IDENTIFIED BY xxxxxxxxxx DEFAULT TABLESPACE rdsadmin TEMPORARY TABLESPACE TEMP profile rdsadmin ; 5. 將權限授予 RDSADMIN。 SQL> GRANT "CONNECT" TO RDSADMIN WITH ADMIN OPTION; SQL> GRANT "RESOURCE " TO RDSADMIN WITH ADMIN OPTION; SQL> GRANT "DBA" TO RDSADMIN; SQL> GRANT "SELECT_C ATALOG_ROLE" TO RDSADMIN WITH ADMIN OPTION; SQL> GRANT ALTER SYSTEM TO RDSADMIN; SQL> GRANT UNLIMITED TABLESPACE TO RDSADMIN;	

任務	Description	所需的技能
	SQL> GRANT SELECT ANY TABLE TO RDSADMIN; SQL> GRANT ALTER DATABASE TO RDSADMIN; SQL> GRANT ADMINISTER DATABASE TRIGGER TO RDSADMIN; SQL> GRANT ANY OBJECT PRIVILEGE TO RDSADMIN WITH ADMIN OPTION; SQL> GRANT INHERIT ANY PRIVILEGES TO RDSADMIN; SQL> ALTER USER RDSADMIN DEFAULT ROLE ALL; 6. Set the SYS, SYSTEM, and DBSNMP user profiles to RDSADMIN. SQL> set echo on feedback on serverout on SQL> alter user SYS profile RDSADMIN; SQL> alter user SYSTEM profile RDSADMIN; SQL> alter user DBSNMP profile RDSADMIN;	

任務	Description	所需的技能

建立主要使用者。	由於啟動者資料庫已捨棄，且目標資料庫已使用 RMAN 從來源還原，因此您必須重新建立主要使用者。在此範例中，主要使用者名稱為 admin。
	
	`SQL> create user admin identified by <password>;`
	`SQL> grant dba to admin`
		DBA

變更系統密碼。	使用您從 Secrets Manager 擷取的密碼來變更系統密碼。
	
	`SQL> alter user sys identified by xxxxxxxxxxx;`
	`SQL> alter user system identified by xxxxxxxxxx;`
	
	如果您不變更這些密碼，Amazon RDS Custom 會顯示錯誤訊息「資料庫監控使用者或使用者登入資料已變更」。
		DBA

設定 Amazon RDS Custom 和 PeopleSoft 的 TNS 項目

任務	Description	所需的技能
設定 tnsnames 檔案。	若要從應用程式層連線至資料庫，請設定 tnsnames.ora 檔案，以便從應用程式層連線至資料庫。在下列範例中，您	DBA

任務	Description	所需的技能

	可以看到 tnsnames.ora 檔案有軟連結，但檔案預設為空白。
	
	`$ cd /rdsdbbin/oracle/network/admin`
	`$ ls -ltr`
	`-rw-r--r-- 1 rdsdb database 1536 Feb 14 2018 shrept.lst`
	`lrwxrwxrwx 1 rdsdb database 30 Apr 5 13:19 listener.ora -> /rdsdbdata/config/listener.ora`
	`lrwxrwxrwx 1 rdsdb database 28 Apr 5 13:19 sqlnet.ora -> /rdsdbdata/config/sqlnet.ora`
	`lrwxrwxrwx 1 rdsdb database 30 Apr 5 13:19 tnsnames.ora -> /rdsdbdata/config/tnsnames.ora`
	
	1. 建立 tsnames.ora 項目。由於 Amazon RDS 自動化剖析檔案的方式，您必須確保項目不包含任何空格、註解或額外的行。否則，您可能會在使用一些 APIs時遇到問題，例如 create-db-instance-read-replica。
	2. 根據您的 PeopleSoft 資料庫需求取代連接埠、主機和 SID。使用下列程式碼做為範例。	

任務	Description	所需的技能

	<pre>$ vi tnsnames.ora
	
	FSDMO92=(DESCRIP
	TION = (ADDRESS_
	LIST = (ADDRESS =
	(PROTOCOL = TCP)(HOST
	= x.x.x.x)(PORT =
	1521))) (CONNECT_
	DATA = (SERVER =
	DEDICATED) (SID =
	FSDMO92)))</pre>	

	3. 若要確認可以到達 PeopleSoft 資料庫，請執行下列命令。
	<pre>$ tnsping FSDMO92
	
	TNS Ping Utility
	for Linux: Version
	19.0.0.0.0 -
	Production on 14-
	JUL-2022 10:16:45
	
	Copyright (c) 1997,
	2021, Oracle. All
	rights reserved.
	
	Used parameter files:
	/rdsdbbin/oracle/net
	work/admin/sqlnet.
	ora
	
	Used TNSNAMES adapter
	to resolve the alias
	Attempting to
	contact (DESCRIPT
	ION = (ADDRESS_
	LIST = (ADDRESS =
	(PROTOCOL = TCP)(HOST</pre>	

任務	Description	所需的技能
	= x.x.x.x)(PORT = 1521))) (CONNECT_DATA = (SERVER = DEDICATED) (SID = FSDMO92))) OK (0 msec)	

建立 spfile softlink

任務	Description	所需的技能

建立 spfile softlink。	1. 若要在位置 中建立 spfile/rdsdbdata/admin/FSDMO92/pfile ，請執行下列命令。
	`SQL> create spfile='/rdsdbdata/admin/FSDMO92/pfile/spfileFSDMO92.ora' from pfile;`
	2. 導覽至 $ORACLE_HOME/dbs，並建立 spfile 的軟連結。
	`ln -s '/rdsdbdata/admin/FSDMO92/pfile/spfileFSDMO92.ora' spfileFSDMO92.ora`
	3. 建立此檔案之後，您可以使用 spfile 關閉並啟動資料庫。
		DBA

執行遷移後步驟

任務	Description	所需的技能
驗證結構描述、連線和維護任務。	若要完成遷移，請執行下列任務。 • 收集結構描述統計資料。 • 確保 PeopleSoft 應用程式層可以連接到新的 Amazon RDS Custom 資料庫。 • 設定您的備份和維護排程。	DBA

相關資源

- 使用 Amazon RDS Custom
- Amazon RDS Custom for Oracle – 資料庫環境中的新控制功能（部落格文章）
- 將 Amazon RDS Custom for Oracle 與 Amazon EFS 整合（部落格文章）
- 將 Amazon RDS 設定為 Oracle PeopleSoft 資料庫 (AWS 白皮書）

將 Oracle ROWID 功能遷移至 AWS 上的 PostgreSQL

Rakesh Raghav 和 Ramesh Pathuri，Amazon Web Services

總結

此模式說明將 Oracle 資料庫中的ROWID虛擬資料欄功能遷移至 PostgreSQL (Amazon Relational Database Service RDS) for PostgreSQL、Amazon Aurora PostgreSQL 相容版本或 Amazon Elastic Compute Cloud (Amazon EC2) 中 PostgreSQL 資料庫的選項。

在 Oracle 資料庫中，ROWID虛擬資料欄是資料表中資料列的實體地址。即使資料表上沒有主索引鍵，此虛擬資料欄也會用來唯一識別資料列。PostgreSQL 具有稱為 的類似虛擬資料欄ctid，但無法用作ROWID。如 PostgreSQL 文件所述，如果更新或在每次VACUUM程序之後， ctid可能會變更。

您可以透過三種方式在 PostgreSQL ROWID 中建立虛擬資料欄功能：

- 使用主索引鍵欄而非 ROWID來識別資料表中的資料列。
- 在 資料表中使用邏輯主要/唯一金鑰 （ 可能是複合金鑰 ）。
- 新增具有自動產生值的資料欄，使其成為要模擬的主要/唯一金鑰ROWID。

此模式會逐步解說這三個實作，並說明每個選項的優點和缺點。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 程序語言/PostgreSQL (PL/pgSQL) 編碼專業知識
- 來源 Oracle 資料庫
- Amazon RDS for PostgreSQL 或 Aurora PostgreSQL 相容叢集，或託管 PostgreSQL 資料庫的 EC2 執行個體

限制

- 此模式提供 ROWID功能的解決方法。PostgreSQL 在 ROWID Oracle 資料庫中不提供 的同等 。

產品版本

- PostgreSQL 11.9 或更新版本

Architecture

來源技術堆疊

- Oracle Database

目標技術堆疊

- Aurora PostgreSQL 相容、Amazon RDS for PostgreSQL 或具有 PostgreSQL 資料庫的 EC2 執行個體

實作選項

根據您的資料表是否具有主索引鍵或唯一索引、邏輯主索引鍵或身分屬性，有三種選項可以解決 PostgreSQL 中缺乏ROWID支援的問題。您的選擇取決於您的專案時間表、目前的遷移階段，以及應用程式和資料庫程式碼的相依性。

選項	Description	優點	缺點
主索引鍵或唯一索引	如果您的 Oracle 資料表具有主索引鍵，您可以使用此索引鍵的屬性來唯一識別資料列。	• 不依賴專屬資料庫功能。 • 對效能的影響最小，因為主索引鍵欄位會編製索引。	• 需要變更依賴ROWID切換到主索引鍵欄位的應用程式和資料庫程式碼。
邏輯主要/唯一金鑰	如果您的 Oracle 資料表具有邏輯主索引鍵，您可以使用此索引鍵的屬性來唯一識別資料列。邏輯主索引鍵由屬性或一組屬性組成，這些屬性可以唯一識別資料列，但不會透過限制在資料庫上強制執行。	• 不依賴專屬資料庫功能。	• 需要變更依賴ROWID切換到主索引鍵欄位的應用程式和資料庫程式碼。 • 如果邏輯主索引鍵的屬性未編製索引，會對效能造成重大影響。不過，

			您可以新增唯一的索引以防止效能問題。
身分屬性	如果您的 Oracle 資料表沒有主索引鍵，您可以將其他欄位建立為 `GENERATED ALWAYS AS IDENTITY`。此屬性會在資料插入資料表時產生唯一值，因此可用於唯一識別資料控制語言 (DML) 操作的資料列。	• 不依賴專屬資料庫功能。 • PostgreSQL 資料庫會填入 屬性並維護其唯一性。	• 需要變更依賴 `ROWID`切換到身分屬性的應用程式和資料庫程式碼。 • 如果其他欄位未編製索引，則對效能有重大影響。不過，您可以新增索引以防止效能問題。

工具

- 適用於 PostgreSQL 的 Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展 PostgreSQL 關聯式資料庫。
- Amazon Aurora PostgreSQL 相容版本是完全受管的 ACID 相容關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。在此模式中，您可以使用 AWS CLI 透過 pgAdmin 執行 SQL 命令。
- pgAdmin 是 PostgreSQL 的開放原始碼管理工具。它提供圖形界面，可協助您建立、維護和使用資料庫物件。
- AWS Schema Conversion Tool (AWS SCT) 會自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，以支援異質資料庫遷移。

史詩

識別來源資料表

任務	Description	所需的技能
識別使用 `ROWID` 屬性的 Oracle 資料表。	使用 AWS Schema Conversion Tool (AWS SCT) 來識別具有	DBA 或開發人員

任務	Description	所需的技能
	ROWID功能的 Oracle 資料表。如需詳細資訊，請參閱 AWS SCT 文件。 —或— 在 Oracle 中，使用 DBA_TAB_COLUMNS 檢視來識別具有 ROWID 屬性的資料表。這些欄位可用於存放英數 10 位元組字元。判斷用量，並適時將其轉換為VARCHAR欄位。	
識別參考這些資料表的程式碼。	使用 AWS SCT 產生遷移評估報告，以識別受 影響的程序ROWID。如需詳細資訊，請參閱 AWS SCT 文件。 —或— 在來源 Oracle 資料庫中，使用dba_source 資料表的文字欄位來識別使用 ROWID功能的物件。	DBA 或開發人員

判斷主索引鍵用量

任務	Description	所需的技能
識別沒有主索引鍵的資料表。	在來源 Oracle 資料庫中，使用 DBA_CONSTRAINTS 來識別沒有主索引鍵的資料表。此資訊將協助您判斷每個資料表的策略。例如：	DBA 或開發人員

任務	Description	所需的技能
	`select dt.* from dba_tables dt where not exists (select 1 from all_constraints ct where ct.owner = Dt.owner and ct.table_name = Dt.table_name and ct.constraint_type = 'P' ) and dt.owner = '{schema}'`	

識別並套用解決方案

任務	Description	所需的技能
針對具有已定義或邏輯主索引鍵的資料表套用變更。	進行其他資訊區段中顯示的應用程式和資料庫程式碼變更，以使用唯一的主索引鍵或邏輯主索引鍵來識別資料表中的資料列。	DBA 或開發人員
將其他欄位新增至沒有已定義或邏輯主索引鍵的資料表。	新增 類型的屬性`GENERATED ALWAYS AS IDENTITY`。進行其他資訊區段中顯示的應用程式和資料庫程式碼變更。	DBA 或開發人員
視需要新增索引。	將索引新增至其他欄位或邏輯主索引鍵，以改善 SQL 效能。	DBA 或開發人員

相關資源

- PostgreSQL CTID (PostgreSQL 文件)
- 產生的資料欄 (PostgreSQL 文件)
- ROWID 虛擬資料欄 (Oracle 文件)

其他資訊

下列各節提供 Oracle 和 PostgreSQL 程式碼範例，以說明這三種方法。

案例 1：使用主要唯一金鑰

在下列範例中，您會使用 建立`testrowid_s1`資料表`emp_id`做為主索引鍵。

Oracle 程式碼：

```
create table testrowid_s1 (emp_id integer, name varchar2(10), CONSTRAINT testrowid_pk
 PRIMARY KEY (emp_id));
INSERT INTO testrowid_s1(emp_id,name) values (1,'empname1');
INSERT INTO testrowid_s1(emp_id,name) values (2,'empname2');
INSERT INTO testrowid_s1(emp_id,name) values (3,'empname3');
INSERT INTO testrowid_s1(emp_id,name) values (4,'empname4');
commit;

SELECT rowid,emp_id,name FROM testrowid_s1;
ROWID                  EMP_ID NAME
------------------ ---------- ----------
AAAF3pAAAAAAAMOAAA          1 empname1
AAAF3pAAAAAAAMOAAB          2 empname2
AAAF3pAAAAAAAMOAAC          3 empname3
AAAF3pAAAAAAAMOAAD          4 empname4

UPDATE testrowid_s1 SET name = 'Ramesh' WHERE rowid = 'AAAF3pAAAAAAAMOAAB' ;
commit;

SELECT rowid,emp_id,name FROM testrowid_s1;
ROWID                  EMP_ID NAME
------------------ ---------- ----------
AAAF3pAAAAAAAMOAAA          1 empname1
AAAF3pAAAAAAAMOAAB          2 Ramesh
AAAF3pAAAAAAAMOAAC          3 empname3
AAAF3pAAAAAAAMOAAD          4 empname4
```

PostgreSQL 程式碼：

```
CREATE TABLE public.testrowid_s1
(
    emp_id integer,
    name character varying,
    primary key (emp_id)
);

insert into public.testrowid_s1 (emp_id,name) values
(1,'empname1'),(2,'empname2'),(3,'empname3'),(4,'empname4');

select emp_id,name from testrowid_s1;
 emp_id |   name
--------+----------
      1 | empname1
      2 | empname2
      3 | empname3
      4 | empname4

update testrowid_s1 set name = 'Ramesh' where emp_id = 2 ;

select emp_id,name from testrowid_s1;
 emp_id |   name
--------+----------
      1 | empname1
      3 | empname3
      4 | empname4
      2 | Ramesh
```

案例 2：使用邏輯主索引鍵

在下列範例中，您會使用 建立資料表`testrowid_s2``emp_id`做為邏輯主索引鍵。

Oracle 程式碼：

```
create table testrowid_s2 (emp_id integer, name varchar2(10) );
INSERT INTO testrowid_s2(emp_id,name) values (1,'empname1');
INSERT INTO testrowid_s2(emp_id,name) values (2,'empname2');
INSERT INTO testrowid_s2(emp_id,name) values (3,'empname3');
INSERT INTO testrowid_s2(emp_id,name) values (4,'empname4');
commit;
```

```
SELECT rowid,emp_id,name FROM testrowid_s2;
ROWID                  EMP_ID NAME
------------------ ---------- ----------
AAAF3rAAAAAAAMeAAA          1 empname1
AAAF3rAAAAAAAMeAAB          2 empname2
AAAF3rAAAAAAAMeAAC          3 empname3
AAAF3rAAAAAAAMeAAD          4 empname4

UPDATE testrowid_s2 SET name = 'Ramesh' WHERE rowid = 'AAAF3rAAAAAAAMeAAB' ;
commit;

SELECT rowid,emp_id,name FROM testrowid_s2;
ROWID                  EMP_ID NAME
------------------ ---------- ----------
AAAF3rAAAAAAAMeAAA          1 empname1
AAAF3rAAAAAAAMeAAB          2 Ramesh
AAAF3rAAAAAAAMeAAC          3 empname3
AAAF3rAAAAAAAMeAAD          4 empname4
```

PostgreSQL 程式碼：

```
CREATE TABLE public.testrowid_s2
(
    emp_id integer,
    name character varying
);

insert into public.testrowid_s2 (emp_id,name) values
(1,'empname1'),(2,'empname2'),(3,'empname3'),(4,'empname4');

select emp_id,name from testrowid_s2;
 emp_id |   name
--------+----------
      1 | empname1
      2 | empname2
      3 | empname3
      4 | empname4

update testrowid_s2 set name = 'Ramesh' where emp_id = 2 ;

select emp_id,name from testrowid_s2;
 emp_id |   name
--------+----------
```

```
      1 | empname1
      3 | empname3
      4 | empname4
      2 | Ramesh
```

案例 3：使用身分屬性

在下列範例中，您使用身分屬性建立不含主索引鍵的資料表testrowid_s3。

Oracle 程式碼：

```
create table testrowid_s3 (name varchar2(10));
INSERT INTO testrowid_s3(name) values ('empname1');
INSERT INTO testrowid_s3(name) values ('empname2');
INSERT INTO testrowid_s3(name) values ('empname3');
INSERT INTO testrowid_s3(name) values ('empname4');
commit;

SELECT rowid,name FROM testrowid_s3;
ROWID              NAME
------------------ ----------
AAAF3sAAAAAAAMmAAA empname1
AAAF3sAAAAAAAMmAAB empname2
AAAF3sAAAAAAAMmAAC empname3
AAAF3sAAAAAAAMmAAD empname4

UPDATE testrowid_s3 SET name = 'Ramesh' WHERE rowid = 'AAAF3sAAAAAAAMmAAB' ;
commit;

SELECT rowid,name FROM testrowid_s3;
ROWID              NAME
------------------ ----------
AAAF3sAAAAAAAMmAAA empname1
AAAF3sAAAAAAAMmAAB Ramesh
AAAF3sAAAAAAAMmAAC empname3
AAAF3sAAAAAAAMmAAD empname4
```

PostgreSQL 程式碼：

```
CREATE TABLE public.testrowid_s3
(
    rowid_seq bigint generated always as identity,
    name character varying
```

```
);

insert into public.testrowid_s3 (name) values
('empname1'),('empname2'),('empname3'),('empname4');

select rowid_seq,name from testrowid_s3;
 rowid_seq |   name
-----------+----------
         1 | empname1
         2 | empname2
         3 | empname3
         4 | empname4

update testrowid_s3 set name = 'Ramesh' where rowid_seq = 2 ;

select rowid_seq,name from testrowid_s3;
 rowid_seq |   name
-----------+----------
         1 | empname1
         3 | empname3
         4 | empname4
         2 | Ramesh
```

將 Oracle 資料庫錯誤代碼遷移至與 Amazon Aurora PostgreSQL 相容的資料庫

Sai Parthasaradhi 和 Veeranjaneyulu Grandhi，Amazon Web Services

總結

此模式說明如何使用預先定義的中繼資料表，將 Oracle 資料庫錯誤代碼遷移至 Amazon Aurora PostgreSQL 相容版本資料庫。

Oracle 資料庫錯誤代碼不一定有對應的 PostgreSQL 錯誤代碼。這種錯誤代碼的差異可能會使得設定目標 PostgreSQL 架構中程序或函數的處理邏輯變得困難。

您可以將對 PL/pgSQL 程式有意義的來源和目標資料庫錯誤代碼儲存在中繼資料表中，以簡化程序。然後，將資料表設定為標記有效的 Oracle 資料庫錯誤代碼，並將其映射至其 PostgreSQL 對等項目，然後再繼續剩餘的程序邏輯。如果 Oracle 資料庫錯誤碼不在中繼資料資料表中，則程序會因例外狀況而結束。然後，如果您的程式需要，您可以手動檢閱錯誤詳細資訊，並將新的錯誤代碼新增至資料表。

透過使用此組態，Amazon Aurora PostgreSQL 相容資料庫可以像來源 Oracle 資料庫一樣處理錯誤。

> **Note**
>
> 設定 PostgreSQL 資料庫以正確處理 Oracle 資料庫錯誤碼通常需要變更資料庫和應用程式碼。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 啟動和執行執行個體和接聽程式服務的來源 Oracle 資料庫
- 已啟動並執行的 Amazon Aurora PostgreSQL 相容叢集
- 熟悉 Oracle 資料庫
- 熟悉 PostgreSQL 資料庫

Architecture

下圖顯示資料錯誤碼驗證和處理的 Amazon Aurora PostgreSQL 相容資料庫工作流程範例：

該圖顯示以下工作流程：

1. 資料表包含 Oracle 資料庫錯誤代碼和分類，以及其同等 PostgreSQL 錯誤代碼和分類。資料表包含 valid_error 資料欄，可分類特定預先定義的錯誤代碼是否有效。
2. 當 PL/pgSQL 函數 (func_processdata) 擲回例外狀況時，它會叫用第二個 PL/pgSQL 函數 (error_validation)。
3. error_validation 函數接受 Oracle 資料庫錯誤代碼作為輸入引數。然後，函數會對照資料表檢查傳入的錯誤代碼，以查看錯誤是否包含在資料表中。
4. 如果資料表中包含 Oracle 資料庫錯誤代碼，則 error_validation 函數會傳回 TRUE 值，且程序邏輯會繼續。如果錯誤碼不包含在資料表中，則函數會傳回 FALSE 值，且程序邏輯會因例外狀況而結束。
5. 當函數傳回 FALSE 值時，應用程式的功能主管會手動檢閱錯誤詳細資訊，以判斷其有效性。
6. 然後，新的錯誤代碼要不是手動新增到資料表。如果錯誤代碼有效並新增至資料表，則 error_validation 函數會在下次發生例外狀況時傳回 TRUE 值。如果錯誤碼無效，且程序必須在發生例外狀況時失敗，則錯誤碼不會新增至資料表。

技術堆疊

- Amazon Aurora PostgreSQL
- pgAdmin
- Oracle SQL Developer

工具

- Amazon Aurora PostgreSQL 相容版本是完全受管的 ACID 相容關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。
- pgAdmin 是 PostgreSQL 的開放原始碼管理和開發工具。它提供圖形界面，可簡化資料庫物件的建立、維護和使用。
- Oracle SQL Developer 是免費的整合開發環境，可簡化傳統和雲端部署中 Oracle 資料庫的開發和管理。

史詩

將 Oracle 資料庫錯誤代碼遷移至與 Amazon Aurora PostgreSQL 相容的資料庫

任務	Description	所需的技能
在 Amazon Aurora PostgreSQL 相容資料庫中建立資料表。	執行下列 PostgreSQL CREATE TABLE 命令： `( source_error_code numeric NOT NULL, target_error_code character varying NOT NULL, valid_error character varying(1) NOT NULL );`	PostgreSQL 開發人員、Oracle、RDS/Aurora for PostgreSQL
將 PostgreSQL 錯誤代碼及其對應的 Oracle 資料庫錯誤代碼新增至資料表。	執行 PostgreSQL INSERT 命令，將所需的錯誤代碼值新增至 error_codes 資料表。 PostgreSQL 錯誤代碼必須使用不同字元的資料類型 (SQLSTATE 值）。Oracle 錯誤代碼必須使用數值資料類型 (SQLCODE 值）。 插入陳述式範例： `insert into error_codes values (-1817,'22007','Y');`	PostgreSQL 開發人員、Oracle、RDS/Aurora for PostgreSQL

任務	Description	所需的技能

	`insert into error_codes values (-1816,'22007','Y');`
	`insert into error_codes values (-3114,'08006','N');`
	
	Note
	如果您發現 Oracle 特定的 Java 資料庫連線 (JDBC) 例外狀況，您必須以一般跨資料庫例外狀況取代這些例外狀況，或切換到 PostgreSQL 特定的例外狀況。	

建立 PL/pgSQL 函數以驗證錯誤代碼。	執行 PostgreSQL CREATE FUNCTION 命令來建立 PL/pgSQL https://www.postgresql.org/docs/current/sql-createfunction.html函數。PostgreSQL 請確定函數執行下列動作： • 接受程式擲回的 Oracle 錯誤代碼。 • 檢查 error_codes 資料表中是否存在錯誤代碼。 • 根據錯誤碼是否存在於中繼資料表中，傳回 TRUE 或 FALSE 值。	PostgreSQL 開發人員、Oracle、RDS/Aurora for PostgreSQL

任務	Description	所需的技能
手動檢閱由 PL/pgSQL 函數記錄的新錯誤代碼。	手動檢閱新的錯誤代碼。 如果新的錯誤代碼對您的使用案例有效，請執行 PostgreSQL INSERT 命令將其新增至 error_codes 資料表。 -或- 如果新的錯誤碼對您的使用案例無效，請勿將其新增至資料表。發生錯誤時，程序邏輯將繼續失敗並結束，但有例外。	PostgreSQL 開發人員、Oracle、RDS/Aurora for PostgreSQL

相關資源

附錄 A. PostgreSQL 錯誤代碼 (PostgreSQL 文件）

資料庫錯誤訊息 (Oracle 資料庫文件）

使用 AWS SCT 和 AWS DMS 將 Amazon EC2 上的 SAP ASE 遷移至與 Amazon Aurora PostgreSQL 相容

Amit Kumar 和 Ankit Gupta，Amazon Web Services

總結

此模式說明如何使用 AWS Schema Conversion Tool (AWS SCT) 和 AWS Database Migration Service (AWS DMS)，將託管在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的 SAP Adaptive Server Enterprise (SAP ASE) 資料庫遷移至 Amazon Aurora PostgreSQL 相容版本。 AWS Database Migration Service 模式著重於儲存物件和資料遷移的資料定義語言 (DDL) 轉換。

Aurora PostgreSQL 相容支援線上交易處理 (OLTP) 工作負載。此受管服務提供可自動隨需擴展的組態。它可以根據您的應用程式需求自動啟動、關閉、擴展或縮減資料庫。您可以在雲端中執行資料庫，而無需管理任何資料庫執行個體。Aurora PostgreSQL 相容為不常見、間歇性或無法預測的工作負載提供經濟實惠的選項。

遷移程序包含兩個主要階段：

- 使用 AWS SCT 轉換資料庫結構描述
- 使用 AWS DMS 遷移資料

這兩個階段的詳細指示會在 Epics 區段中提供。如需有關將 AWS DMS 與 SAP ASE 資料庫搭配使用的特定問題疑難排解的資訊，請參閱 AWS DMS 文件中的 SAP ASE 問題疑難排解。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- EC2 執行個體上的來源 SAP ASE 資料庫，可啟動和執行伺服器、資料庫和接聽程式服務
- 目標 Aurora PostgreSQL 相容資料庫

限制

- 連線的連接埠號碼必須為 5432。
- huge_pages 功能預設為開啟，但可以修改。
- Point-in-time復原 (PITR) 精細程度為 5 分鐘。

- 跨區域複寫目前無法使用。
- Aurora 資料庫的儲存大小上限為 128 TiB。
- 您最多可以建立 15 個僅供讀取複本。
- 資料表大小限制僅受限於 Aurora 叢集磁碟區的大小，因此 Aurora PostgreSQL 相容資料庫叢集的資料表大小上限為 32 TiB。建議您遵循資料表設計的最佳實務，例如分割大型資料表。

產品版本

- 來源資料庫：AWS DMS 目前支援 SAP ASE 15、15.5、15.7 和 16.x。如需 SAP ASE 版本支援的最新資訊，請參閱 AWS DMS 使用者指南。
- 目標資料庫：PostgreSQL 9.4 和更新版本（適用於 9.x 版）、10.x、11.x、12.x、13.x 和 14.x。如需最新支援的 PostgreSQL 版本，請參閱 AWS DMS 使用者指南。
- Amazon Aurora 1.x 或更新版本。如需最新資訊，請參閱 Aurora 文件中的 Aurora PostgreSQL 相容版本和引擎版本。

Architecture

來源技術堆疊

- 在 Amazon EC2 上執行的 SAP ASE 資料庫

目標技術堆疊

- Aurora PostgreSQL 相容資料庫

遷移架構

工具

- Amazon Aurora PostgreSQL 相容版本是完全受管的 ACID 相容關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。
- AWS Schema Conversion Tool (AWS SCT) 會自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，以支援異質資料庫遷移。
- AWS DMS 支援數個不同的來源和目標資料庫。如需詳細資訊，請參閱 AWS DMS 文件中的資料遷移來源和資料遷移目標。如需最全面的版本和功能支援，建議您使用最新版本的 AWS DMS。

史詩

設定環境

任務	Description	所需的技能
在來源 EC2 執行個體中設定網路存取。	在託管來源 SAP ASE 資料庫的 EC2 執行個體中設定安全群組。 如需說明，請參閱 Amazon EC2 文件中的 Linux 執行個體的 Amazon EC2 安全群組。Amazon EC2	系統管理員
建立您的目標 Aurora PostgreSQL 相容資料庫叢集。	為您的目標資料庫安裝、設定和啟動 Aurora PostgreSQL 相容叢集。 如需詳細資訊，請參閱 Aurora 文件中的建立 Amazon Aurora 資料庫叢集。	DBA
設定目標資料庫叢集的授權。	設定目標資料庫的安全群組和防火牆。 如需說明，請參閱 Aurora 文件中的建立 Amazon Aurora 資料庫叢集。	DBA，系統管理員

使用 AWS SCT 轉換資料庫結構描述

任務	Description	所需的技能
啟動 AWS SCT。	依照 AWS SCT 文件中的指示啟動 AWS SCT。 AWS SCT 提供專案型使用者介面，可將 SAP ASE 來	DBA

任務	Description	所需的技能
	源資料庫的資料庫結構描述自動轉換為與您目標 Aurora PostgreSQL 相容資料庫執行個體相容的格式。	
建立 AWS SCT 端點。	建立來源 SAP ASE 和目標 PostgreSQL 資料庫的端點。 如需說明，請參閱 AWS SCT 文件。	DBA
建立評估報告。	建立資料庫遷移評估報告，以評估遷移並偵測任何不相容的物件和函數。 如需說明，請參閱 AWS SCT 文件。	DBA
轉換結構描述。	遵循 AWS SCT 文件中的指示轉換資料庫結構描述。	DBA
驗證資料庫物件。	如果 AWS SCT 無法轉換資料庫物件，它會識別其名稱和其他詳細資訊。您必須手動轉換這些物件。 若要識別這些不相符項目，請依照 AWS 部落格文章中的指示，在從 SAP ASE 遷移至 Amazon RDS for PostgreSQL 或 Amazon Aurora PostgreSQL 之後驗證資料庫物件。	DBA

分析 AWS DMS 遷移

任務	Description	所需的技能
驗證來源和目標資料庫版本。	檢查 SAP ASE 資料庫版本與 AWS DMS 的相容性。 如需詳細資訊，請參閱 AWS DMS 文件中的 AWS DMS 的來源和 AWS DMS 的目標。	DBA
識別儲存類型和容量的需求。	根據來源資料庫的大小，為目標資料庫選擇適當的儲存容量。	DBA，系統管理員
選擇複寫執行個體的執行個體類型、容量和其他功能。	選擇符合您需求的執行個體類型、容量、儲存功能和網路功能。 如需指引，請參閱 AWS DMS 文件中的為您的遷移選擇正確的 AWS DMS 複寫執行個體。	DBA，系統管理員
識別網路存取安全需求。	識別來源和目標資料庫的網路存取安全需求。 遵循 AWS DMS 文件中設定複寫執行個體網路的指引。	DBA，系統管理員

遷移資料

任務	Description	所需的技能
在 AWS DMS 中建立遷移任務來遷移資料。	若要遷移資料，請建立任務並遵循 AWS DMS 文件中的指示。	DBA

任務	Description	所需的技能
	我們建議您使用最新版本的 AWS DMS，以獲得最全面的版本和功能支援。	
驗證資料。	若要驗證您的資料是否已正確從來源資料庫遷移至目標資料庫，請遵循 AWS DMS 文件中提供的資料驗證準則。	DBA

遷移應用程式

任務	Description	所需的技能
識別應用程式遷移策略。	選擇七種策略之一 (7R) 將應用程式遷移至雲端。	DBA、應用程式擁有者、系統管理員
遵循應用程式遷移策略。	完成應用程式團隊識別的資料庫任務，包括更新目標資料庫的 DNS 連線詳細資訊，以及更新動態查詢。	DBA、應用程式擁有者、系統管理員

切換到目標資料庫

任務	Description	所需的技能
將應用程式用戶端切換到新的基礎設施。	將連線從來源資料庫切換到目標資料庫。 如需詳細資訊，請參閱關聯式資料庫遷移策略的剪下區段。	DBA、應用程式擁有者、系統管理員

關閉專案

任務	Description	所需的技能
關閉臨時 AWS 資源。	終止所有遷移任務、複寫執行個體、端點和其他 AWS SCT 和 AWS DMS 資源。 如需詳細資訊，請參閱 AWS DMS 文件。	DBA，系統管理員
檢閱並驗證專案文件。	驗證專案文件中的所有步驟，以確保所有任務都已成功完成。	DBA、應用程式擁有者、系統管理員
關閉專案。	關閉遷移專案並提供任何意見回饋。	DBA、應用程式擁有者、系統管理員

相關資源

參考

- 在 Amazon RDS 中啟用 PostgreSQL 資料庫執行個體的加密連線 (AWS 規範性指導）
- 使用 pg_transport 在兩個 Amazon RDS 資料庫執行個體之間傳輸 PostgreSQL 資料庫 (AWS 規範性指導）
- Amazon Aurora 定價
- Amazon Aurora PostgreSQL 相容版本的最佳實務 (Amazon Aurora 文件）
- AWS SCT 文件
- AWS DMS 文件
- 使用 SAP ASE 資料庫做為 AWS DMS 的來源

教學課程和影片

- AWS Database Migration Service 入門
- AWS Database Migration Service （影片）

使用 ACM 將 Windows SSL 憑證遷移至 Application Load Balancer

Chandra Sekhar Yaratha 和 Igor Kovalchuk，Amazon Web Services

總結

模式提供使用 AWS Certificate Manager (ACM) 將現有 Secure Sockets Layer (SSL) 憑證從託管在內部部署伺服器或 Microsoft Internet Information Services (IIS) 上的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的網站遷移的指引。然後，SSL 憑證可以與 AWS 上的 Elastic Load Balancing 搭配使用。

SSL 可保護您的資料、確認您的身分、提供更好的搜尋引擎排名、協助滿足支付卡產業資料安全標準 (PCI DSS) 要求，並改善客戶信任。管理這些工作負載的開發人員和 IT 團隊希望其 Web 應用程式和基礎設施，包括 IIS 伺服器和 Windows Server，保持符合其基準政策。

此模式涵蓋從 Microsoft IIS 手動匯出現有的 SSL 憑證、將它們從個人資訊交換 (PFX) 格式轉換為 ACM 支援的 Private Enhanced Mail (PEM) 格式，然後將它們匯入 AWS 帳戶中的 ACM。它也說明如何為您的應用程式建立 Application Load Balancer，並將 Application Load Balancer 設定為使用匯入的憑證。然後，在 Application Load Balancer 上終止 HTTPS 連線，您不需要在 Web 伺服器上進一步設定額外負荷。如需詳細資訊，請參閱建立 Application Load Balancer 的 HTTPS 接聽程式。

Windows 伺服器使用 .pfx 或 .p12 檔案來包含公有金鑰檔案 (SSL 憑證） 及其唯一的私有金鑰檔案。Certificate Authority (CA) 為您提供公有金鑰檔案。您可以使用伺服器來產生建立憑證簽署請求 (CSR) 的相關聯私有金鑰檔案。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- AWS 上的虛擬私有雲端 (VPC)，在目標使用的每個可用區域中至少有一個私有和一個公有子網路
- 在 Windows Server 2012 或更新版本上執行的 IIS 8.0 版或更新版本
- 在 IIS 上執行的 Web 應用程式
- IIS 伺服器的管理員存取權

Architecture

來源技術堆疊

- 使用 SSL 實作 IIS Web 伺服器，以確保在加密連線 (HTTPS) 中安全地傳輸資料

來源架構

目標技術堆疊

- 您 AWS 帳戶中的 ACM 憑證
- 設定為使用匯入憑證的 Application Load Balancer
- 私有子網路中的 Windows Server 執行個體

目標架構

工具

- AWS Certificate Manager (ACM) 可協助您建立、存放和續約公有和私有 SSL/TLS X.509 憑證和金鑰，以保護 AWS 網站和應用程式。
- Elastic Load Balancing (ELB) 會將傳入的應用程式或網路流量分配到多個目標。例如，您可以在一或多個可用區域中跨 EC2 執行個體、容器和 IP 地址分配流量。

最佳實務

- 強制流量從 HTTP 重新導向至 HTTPS。
- 為您的 Application Load Balancer 正確設定安全群組，以僅允許特定連接埠的傳入流量。
- 在不同的可用區域中啟動 EC2 執行個體，以確保高可用性。
- 將應用程式的網域設定為指向 Application Load Balancer 的 DNS 名稱，而不是其 IP 地址。
- 確定 Application Load Balancer 已設定應用程式層運作狀態檢查。
- 設定運作狀態檢查的閾值。
- 使用 Amazon CloudWatch 監控 Application Load Balancer。

史詩

匯出 .pfx 檔案

任務	Description	所需的技能
從 Windows Server 匯出 .pfx 檔案。	若要從 Windows Server 中的現場部署 IIS 管理員將 SSL 憑證匯出為 .pfx 檔案： 1. 選擇開始、管理、網際網路資訊服務 (IIS) 管理員。 2. 選取伺服器名稱，然後在安全性下按兩下伺服器憑證。 3. 選擇您要匯出的憑證，然後選擇匯出。 4. 在匯出憑證方塊中，選擇 .pfx 檔案的位置、路徑和名稱。 5. 指定並確認 .pfx 檔案的密碼。 Note 安裝 .pfx 檔案時，您需要此密碼。 6. 選擇確定。 您的 .pfx 檔案現在應儲存至您指定的位置和路徑。	系統管理員

將 PFX 編碼憑證轉換為 PEM 格式

任務	Description	所需的技能
下載並安裝 OpenSSL 工具組。	1. 從 Shining Light Productions 網站下載並安裝 Win32/Win64 OpenSSL。 2. 將 OpenSSL 二進位檔的位置新增至您的系統PATH變數，以便二進位檔可供命令列使用。	系統管理員
將 PFX 編碼憑證轉換為 PEM 格式。	下列步驟會將 PFX 編碼的簽章憑證檔案轉換為三個 PEM 格式的檔案： • `cert-file.pem` 包含 資源的 SSL/TLS 憑證。 • `privatekey.pem` 包含憑證的私有金鑰，沒有密碼保護。 • `ca-chain.pem` 包含 CA 的根憑證。 若要轉換 PFX 編碼憑證： 1. 執行 Windows PowerShell。 2. 使用下列命令，從 PFX 檔案擷取憑證的私有金鑰。出現提示時，輸入憑證密碼。 `openssl pkcs12 -in <filename>.pfx -nocerts -out withpw-privatekey.pem`	系統管理員

任務	Description	所需的技能

	命令會產生名為 的 PEM 編碼私有金鑰檔案`privatekey.pem` 。輸入密碼短語，以在出現提示時保護私有金鑰檔案。
	3. 執行下列命令來移除密碼短語。出現提示時，請提供您在步驟 2 中建立的密碼短語。
	`openssl rsa -in withpw-privatekey.pem -out privatekey.pem`
	如果命令成功，則會顯示「寫入 RSA 金鑰」訊息。
	4. 使用下列命令，將憑證從 PFX 檔案傳輸到 PEM 檔案。
	`openssl pkcs12 -in <file_name>.pfx -clcerts -nokeys -out cert-file.pem`
	這會建立名為 的 PEM 編碼憑證檔案`cert-file.pem` 。如果命令成功，則會顯示「MAC 驗證確定」訊息。
	5. 從 PFX 檔案建立 CA 鏈結檔案。下列命令會建立名為	

任務	Description	所需的技能
	的 CA 鏈結檔案`ca-chain.pem` 。 `openssl pkcs12 -in <file_name>.pfx -cacerts -nokeys -chain -out ca-chain.pem` 如果命令成功，則會顯示「MAC 驗證確定」訊息。	

將憑證匯入 ACM

任務	Description	所需的技能
準備匯入憑證。	在 ACM 主控台上，選擇匯入憑證。	雲端管理員
提供憑證內文。	針對憑證內文，貼上您要匯入的 PEM 編碼憑證。 如需此範例和其他任務中所述命令和步驟的詳細資訊，請參閱 ACM 文件中的匯入憑證。	雲端管理員
提供憑證私有金鑰。	對於 Certificate private key (憑證私有金鑰)，貼上與憑證公有金鑰相符的 PEM 編碼、未加密私有金鑰。	雲端管理員
提供憑證鏈。	對於憑證鏈，請貼上存放在`CertificateChain.pem` 檔案中的 PEM 編碼憑證鏈。	雲端管理員

任務	Description	所需的技能
匯入憑證。	選擇 Review and import (檢閱和匯入)。確認憑證的相關資訊正確無誤，然後選擇匯入。	雲端管理員

建立 Application Load Balancer

任務	Description	所需的技能
建立和設定負載平衡器和接聽程式。	遵循 Elastic Load Balancing 文件中的指示來設定目標群組、註冊目標，以及建立 Application Load Balancer 和接聽程式。新增連接埠 443 的第二個接聽程式 (HTTPS)。	雲端管理員

疑難排解

問題	解決方案
Windows PowerShell 無法辨識 OpenSSL 命令，即使您將其新增至系統路徑。	檢查`$env:path` 以確保它包含 OpenSSL 二進位檔的位置。 如果沒有，請在 PowerShell 中執行下列命令： `$env:path = $env:path + ";C:\OpenSSL-Win64\bin"`

相關資源

將憑證匯入 ACM

- ACM 主控台
- 用於匯入的憑證和金鑰格式

- 匯入憑證
- AWS Certificate Manager 使用者指南

建立 Application Load Balancer

- 建立 Application Load Balancer
- Application Load Balancer 使用者指南

將訊息佇列從 Microsoft Azure Service Bus 遷移至 Amazon SQS

Nisha Gambhir，Amazon Web Services

總結

此模式說明如何將 .NET Framework 或 .NET Core Web 或主控台應用程式從使用 Microsoft Azure Service 匯流排佇列傳訊平台遷移至 Amazon Simple Queue Service (Amazon SQS)。

應用程式使用簡訊服務將資料傳送至其他應用程式，以及從其他應用程式接收資料。這些服務有助於在雲端中建置解耦、高度可擴展的微服務、分散式系統和無伺服器應用程式。

Azure 服務匯流排佇列是更廣泛的 Azure 訊息基礎設施的一部分，支援佇列和發佈/訂閱訊息。

Amazon SQS 是一種全受管訊息佇列服務，可讓您解耦和擴展微服務、分散式系統和無伺服器應用程式。Amazon SQS 消除了與管理和操作訊息導向中介軟體相關的複雜性和額外負荷，並使開發人員能夠專注於區分工作。使用 Amazon SQS，您可以在任何磁碟區中傳送、存放和接收軟體元件之間的訊息，而不會遺失訊息或需要其他 服務。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 使用 Azure Service Bus 佇列的 .NET Framework 或 .NET Core Web 或主控台應用程式 （附加範例程式碼）

產品版本

- .NET Framework 3.5 或更新版本，或 .NET Core 1.0.1、2.0.0 或更新版本

Architecture

來源技術堆疊

- 使用 Azure 服務匯流排佇列傳送訊息的 .NET （核心或架構） Web 或主控台應用程式

目標技術堆疊

- Amazon SQS

工具

工具

- Microsoft Visual Studio

Code

若要為 Amazon SQS 建立 AWS Identity and Access Management (IAM) 政策：

1. 登入 AWS 管理主控台，然後前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。

2. 在左邊的導覽窗格中，選擇 Policies (政策)，然後選擇 Create policy (建立政策)。

3. 選擇 JSON 索引標籤，然後貼上下列程式碼：

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "VisualEditor0",
         "Effect": "Allow",
         "Action": [
            "sqs:DeleteMessage",
            "sqs:GetQueueUrl",
            "sqs:ChangeMessageVisibility",
            "sqs:SendMessageBatch",
            "sqs:ReceiveMessage",
            "sqs:SendMessage",
            "sqs:GetQueueAttributes",
            "sqs:ListQueueTags",
            "sqs:ListDeadLetterSourceQueues",
            "sqs:DeleteMessageBatch",
            "sqs:PurgeQueue",
            "sqs:DeleteQueue",
            "sqs:CreateQueue",
            "sqs:ChangeMessageVisibilityBatch",
            "sqs:SetQueueAttributes"
         ],
         "Resource": "arn:aws:sqs:*:<AccountId>:*"
```

```
      },
      {
         "Sid": "VisualEditor1",
         "Effect": "Allow",
         "Action": "sqs:ListQueues",
         "Resource": "*"
      }
   ]
}
```

4. 選擇檢閱政策，輸入名稱，然後選擇建立政策。

5. 將新建立的政策連接到現有的 IAM 角色或建立新的角色。

史詩

在 AWS 中設定 Amazon SQS

任務	Description	所需的技能
為 Amazon SQS 建立 IAM 政策。	建立提供 Amazon SQS 存取權的 IAM 政策。如需範例政策，請參閱程式碼一節。	系統工程師
建立 AWS 設定檔。	透過執行 AWS Tools for PowerShell 命令 Set-AWSCredential 建立新的設定檔。此命令會將您的存取金鑰和私密金鑰存放在您指定的設定檔名稱下的預設登入資料檔案中。連結您先前使用此帳戶建立的 Amazon SQS 政策。保留 AWS 存取金鑰 ID 和私密存取金鑰。這些是後續步驟的必要項目。	系統工程師
建立 SQS 佇列。	您可以建立標準佇列或先進先出 (FIFO) 佇列。如需說明，請參閱參考區段中的連結。	系統工程師

修改您的 .NET 應用程式程式碼

任務	Description	所需的技能
安裝 AWS Toolkit for Visual Studio。	此工具組是 Microsoft Visual Studio 的延伸，可讓您更輕鬆地在 AWS 中建置和部署 .NET 應用程式。如需安裝和使用說明，請參閱參考章節中的連結。	應用程式開發人員
安裝 AWSSDK.SQS NuGet 套件。	您可以在 Visual Studio AWSSDK 中選擇「管理 NuGet 套件」，或執行命令「Install-Package AWSSDK.SQS」來安裝 AWSSDK.SQS。	應用程式開發人員
在 .NET 應用程式中建立 AWSCredentials 物件。	附件中的範例應用程式示範如何建立從 AWSCredentials 繼承的 BasicAWSCredentials 物件。您可以使用舊版的存取金鑰 ID 和私密存取金鑰，或讓物件從 .aws 資料夾挑選這些金鑰，做為執行時間使用者設定檔的一部分。	應用程式開發人員
建立 SQS 用戶端物件。	為 .NET Framework 建立 SQS 用戶端物件 (AmazonSQSClient)。這是 Amazon.SQS 命名空間的一部分。此物件是必要物件，而非屬於 Microsoft .Azure.ServiceBus 命名空間的 IQueueClient。	應用程式開發人員
呼叫 SendMessageAsync 方法，將訊息傳送至 SQS 佇列。	變更將訊息傳送至佇列的程式碼，以使用 amazonSqsClient.SendMessageAsync 方	應用程式開發人員

任務	Description	所需的技能
	法。如需詳細資訊，請參閱連接的程式碼範例。	
呼叫 ReceiveMessageAsync 方法來接收來自 SQS 佇列的訊息。	變更接收訊息的程式碼，以使用 amazonSqsClient.ReceiveMessageAsync 方法。如需詳細資訊，請參閱連接的程式碼範例。	應用程式開發人員
呼叫 DeleteMessageAsync 方法，從 SQS 佇列刪除訊息。	若要刪除訊息，請將 queueClient.CompleteAsync 方法的程式碼變更為 amazonSqsClient.DeleteMessageAsync 方法。如需詳細資訊，請參閱連接的程式碼範例。	應用程式開發人員

相關資源

- 適用於 .NET 開發人員的 AWS 開發套件
- 使用 Amazon SQS 傳送訊息
- 使用適用於 .NET 的 AWS 開發套件建立和使用 Amazon SQS 佇列
- 傳送 Amazon SQS 訊息
- 從 Amazon SQS 佇列接收訊息
- 從 Amazon SQS 佇列刪除訊息
- AWS Toolkit for Visual Studio

其他資訊

此模式包含兩個範例應用程式（請參閱附件區段）：

- AzureSbTestApp 包含使用 Azure 服務匯流排佇列的程式碼。
- AmazonSqsTestApp 使用 Amazon SQS。這是使用 .NET Core 2.2 的主控台應用程式，其中包含傳送和接收訊息的範例。

備註：

- queueClient 是 IQueueClient 的物件，屬於 Microsoft.Azure.ServiceBus 命名空間（包含在 Microsoft.Azure.ServiceBus NuGet 套件中）。
- amazonSqsClient 是 AmazonSQSClient 的物件，屬於 Amazon.SQS 命名空間（包含在 AWSSDK.SQS NuGet 套件中）。
- 根據程式碼的執行位置，假設其是否在 EC2 上執行，角色需要有寫入 SQS 佇列的許可。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Oracle Data Pump 和 AWS DMS 將 Oracle JD Edwards EnterpriseOne 資料庫遷移至 AWS

Thanigaivel Thirumalai，Amazon Web Services

摘要

您可以在 Amazon Relational Database Service (Amazon RDS) 上遷移和執行 JD Edwards EnterpriseOne 資料庫。 Amazon Relational Database Service 當您將資料庫遷移至 Amazon RDS 時，AWS 可以處理備份任務和高可用性設定，因此您可以專注於維護 EnterpriseOne 應用程式及其功能。如需遷移程序期間要考量之關鍵因素的完整清單，請參閱 AWS 方案指引中的 Oracle 資料庫遷移策略。

有多種方法可以遷移 EnterpriseOne 資料庫，包括：

- 使用 Oracle Universal Batch Engine (UBE) R98403 建立結構描述和資料表，並使用 AWS Database Migration Service (AWS DMS) 進行遷移
- 使用資料庫原生工具建立結構描述和資料表，並使用 AWS DMS 進行遷移
- 使用資料庫原生工具遷移現有資料（完全載入），並使用 AWS DMS 進行變更資料擷取 (CDC) 任務

此模式涵蓋第三個選項。它說明如何使用 Oracle Data Pump 搭配 AWS DMS 及其 CDC 功能，將內部部署 EnterpriseOne 資料庫遷移至 Amazon RDS for Oracle。

Oracle JD Edwards EnterpriseOne 是企業資源規劃 (ERP) 解決方案，適用於生產、建構、分發、服務或管理產品或實體資產的組織。JD Edwards EnterpriseOne 支援各種硬體、作業系統和資料庫平台。

當您遷移 JD Edwards EnterpriseOne 等關鍵的 ERP 應用程式時，將停機時間降至最低是關鍵。AWS DMS 支援從來源資料庫到目標資料庫的完全載入和持續複寫，將停機時間降至最低。AWS DMS 也為遷移提供即時監控和記錄，這可協助您識別和解決可能導致停機的任何問題。

當您使用 AWS DMS 複寫變更時，您必須指定時間或系統變更號碼 (SCN) 做為從資料庫日誌讀取變更的起點。請務必在伺服器上保留一段指定的時間（建議 15 天）來存取這些日誌，以確保 AWS DMS 可以存取這些變更。

先決條件和限制

先決條件

- 在您的 AWS 雲端環境中佈建為目標資料庫的 Amazon RDS for Oracle 資料庫。如需說明，請參閱 Amazon RDS 文件。

- 在內部部署或 AWS 上的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上執行的 EnterpriseOne 資料庫。

> **Note**
>
> 此模式專為從現場部署遷移到 AWS 而設計，但在 EC2 執行個體上使用 EnterpriseOne 資料庫進行測試。如果您計劃從內部部署環境遷移，則必須設定適當的網路連線。

- 結構描述詳細資訊。識別您計劃為 EnterpriseOne 遷移的 Oracle 資料庫結構描述 （ 例如 DV920)。開始遷移程序之前，請收集下列有關結構描述的詳細資訊：
  - 結構描述大小
  - 每個物件類型的物件數量
  - 無效物件的數量

限制

- 您必須在目標 Amazon RDS for Oracle 資料庫上建立您想要的任何結構描述，AWS DMS 不會為您建立這些結構描述。(Epics 區段說明如何使用 Data Pump 匯出和匯入結構描述。) 目標 Oracle 資料庫的結構描述名稱必須已存在。來源結構描述中的資料表會匯入使用者或結構描述，而 AWS DMS 會使用管理員或系統帳戶來連線至目標執行個體。若要遷移多個結構描述，您可以建立多個複寫任務。您也可以將資料遷移到目標執行個體上的不同結構描述。若要這樣做，請在 AWS DMS 資料表映射上使用結構描述轉換規則。
- 此模式已使用示範資料集進行測試。我們建議您驗證資料集和自訂的相容性。
- 此模式使用在 Microsoft Windows 上執行的 EnterpriseOne 資料庫。不過，您可以將相同的程序與 AWS DMS 支援的其他作業系統搭配使用。

Architecture

下圖顯示在 Oracle 資料庫上執行 EnterpriseOne 做為來源資料庫的系統，以及做為目標資料庫的 Amazon RDS for Oracle 資料庫。資料會從來源 Oracle 資料庫匯出，並使用 Oracle Data Pump 匯入目標 Amazon RDS for Oracle 資料庫，並使用 AWS DMS 複寫以進行 CDC 更新。

1. Oracle Data Pump 從來源資料庫擷取資料，並將資料傳送至 Amazon RDS for Oracle 資料庫目標。
2. CDC 資料會從來源資料庫傳送至 AWS DMS 中的來源端點。
3. 從來源端點，資料會傳送至執行複寫任務的 AWS DMS 複寫執行個體。

4. 複寫任務完成後，資料會傳送至 AWS DMS 中的目標端點。
5. 從目標端點，資料會傳送至 Amazon RDS for Oracle 資料庫執行個體。

工具

AWS 服務

- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。
- 適用於 Oracle 的 Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展 Oracle 關聯式資料庫。

其他服務

- Oracle Data Pump 可協助您以高速將資料和中繼資料從一個資料庫移至另一個資料庫。

最佳實務

遷移 LOBs

如果您的來源資料庫包含需要遷移至目標資料庫的大型二進位物件 (LOBs)，AWS DMS 會提供下列選項：

- 完整 LOB 模式 – AWS DMS 會將來源的所有 LOBs 遷移至目標資料庫，無論其大小為何。雖然遷移速度比其他模式慢，但優點是資料不會被截斷。為了獲得更好的效能，您可以在新的複寫執行個體上建立單獨的任務，以遷移 LOBs 大於幾個 MB 的資料表。
- 有限 LOB 模式 – 您可以指定 LOB 資料欄資料的大小上限，這可讓 AWS DMS 預先配置資源並大量套用 LOBs。如果 LOB 資料欄的大小超過任務中指定的大小，AWS DMS 會截斷資料，並將警告傳送至 AWS DMS 日誌檔案。如果您的 LOB 資料大小在有限的 LOB 大小內，您可以使用有限的 LOB 模式來改善效能。
- 內嵌 LOB 模式 – 您可以透過複寫小型和大型 LOBs 來遷移 LOBs而無需截斷資料或降低任務的效能。首先，指定 `InlineLobMaxSize` 參數的值，只有在完整 LOB 模式設定為 時才能使用`true`。AWS DMS 任務會內嵌傳輸小型 LOBs，這更有效率。然後，AWS DMS 會透過從來源資料表執行查詢來遷移大型 LOBs。不過，內嵌 LOB 模式僅適用於完全載入階段。

產生序列值

在 AWS DMS CDC 程序期間，不會從來源資料庫複寫增量序號。為了避免序列值的差異，您必須從所有序列的來源產生最新的序列值，並將其套用至目標 Amazon RDS for Oracle 資料庫。

AWS Secrets Manager

為了協助管理您的登入資料，建議您遵循部落格文章中的指示， 使用 AWS Secrets Manager 管理您的 AWS DMS 端點登入資料。

效能

- 複寫執行個體 – 如需選擇最佳執行個體大小的指引，請參閱 AWS DMS 文件中的為複寫執行個體選取最佳大小。
- 連線選項 – 為了避免延遲問題，我們建議您選擇正確的連線選項。AWS Direct Connect 提供 AWS 資源的最短路徑，因為它是公司資料中心與 AWS 之間的專用連線。傳輸時，您的網路流量會保留在 AWS 全球網路上，永遠不會透過網際網路。相較於使用 VPN 或公有網際網路，這可降低遇到瓶頸或意外增加延遲的機會。
- 網路頻寬 – 若要最佳化效能，請確認您的網路輸送量快速。如果您在內部部署來源資料庫和 AWS DMS 之間使用 VPN 通道，請確定頻寬足以滿足您的工作負載。
- 任務平行處理 – 您可以在完全載入期間平行載入多個資料表，以加速資料複寫。此模式使用 RDBMS 端點，因此此選項僅適用於完全載入程序。任務平行處理是由 `MaxFullLoadSubTasks` 參數控制，這會決定平行執行多少個完全載入子任務。根據預設，此參數設定為 8，這表示在完整模式期間，八個資料表（如果在資料表映射中選取）會一起載入。您可以在任務 JSON 指令碼的完全載入任務設定區段中調整此參數。
- 資料表平行處理 – AWS DMS 也可讓您使用多個平行執行緒載入單一大型資料表。這對於擁有數十億筆記錄以及多個分割區和子分割區的 Oracle 來源資料表特別有用。如果來源資料表未分割，您可以使用平行載入的資料欄邊界。
- 分割負載 – 當您將負載分割到多個任務或 AWS DMS 執行個體時，請在擷取變更時記住交易界限。

史詩

使用 Oracle Data Pump 匯出 EnterpriseOne 結構描述

任務	Description	所需的技能
產生 SCN。	當來源資料庫處於作用中狀態且由 EnterpriseOne 應用程式使用時，請使用 Oracle Data	DBA

任務	Description	所需的技能

	Pump 啟動資料匯出。您必須先從來源資料庫產生系統變更編號 (SCN)，以便在使用 Oracle Data Pump 匯出期間達到資料一致性，並做為 AWS DMS 中 CDC 的起點。
	
	若要從來源資料庫產生目前的 SCN，請使用下列 SQL 陳述式：
	
	`SQL> select current_scn`
	`from v$database;`
	
	`CURRENT_SCN`
	`-----------`
	`30009727`
	
	儲存產生的 SCN。當您匯出資料和建立 AWS DMS 複寫任務時，將使用 SCN。	

任務	Description	所需的技能

建立 參數檔案。	若要建立參數檔案以匯出結構描述，您可以使用下列程式碼。
	
	`directory=DMS_DATA_PUMP_DIR logfile=export_dms.log dumpfile=export_dms_data.dmp schemas=<schema name> flashback_scn=<SCN from previous command>`
	
	Note
	您也可以根據您的需求DATA_PUMP_DIR ，使用以下命令來定義自己的命令。
	
	`SQL> CREATE OR REPLACE DIRECTORY DMS_DATA_PUMP_DIR AS '<Directory for dump>'; Directory created.`
	
	`SQL> GRANT READ, WRITE ON DIRECTORY DMS_DATA_PUMP_DIR TO SYSTEM; Grant succeeded.`
		DBA

任務	Description	所需的技能

匯出結構描述。	若要執行匯出，請使用expdp公用程式，如下所示：
	
	`C:\Users\Administrator>expdp system/********@<DB Name> PARFILE='<Path to PAR file create above>'`
	
	`Export: Release 19.0.0.0.0 - Production on *** *** ** **:**:** **** Version 19.3.0.0.0`
	
	`Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved.`
	
	`Connected to: Oracle Database 19c Standard Edition 2 Release 19.0.0.0.0 - Production`
	`Starting "SYSTEM". "SYS_EXPORT_SCHEMA_02": system/******@<DB Name>PARFILE='E:\exp_dms_datapump.par'`
	`Processing object type SCHEMA_EXPORT/TABLE/TABLE_DATA`
	`Processing object type SCHEMA_EXPORT/TABLE/INDEX/STATISTICS/INDEX_STATISTICS`
	`Processing object type SCHEMA_EXPORT/TABL`
		DBA

任務	Description	所需的技能
	E/STATISTICS/TABLE _STATISTICS Processing object type SCHEMA_EXPORT/STAT ISTICS/MARKER Processing object type SCHEMA_EXPORT/USER Processing object type SCHEMA_EXPORT/ROLE _GRANT Processing object type SCHEMA_EXPORT/DEFA ULT_ROLE Processing object type SCHEMA_EXPORT/TABL ESPACE_QUOTA Processing object type SCHEMA_EXPORT/PRE_ SCHEMA/PROCACT_SCHEMA Processing object type SCHEMA_EXPORT/TABLE/ TABLE Processing object type SCHEMA_EXPORT/TABL E/GRANT/OWNER_GRANT/ OBJECT_GRANT Processing object type SCHEMA_EXPORT/TABLE/ INDEX/INDEX Processing object type SCHEMA_EXPORT/TABLE/ CONSTRAINT/CONSTRAINT . . exported "<Schema Name>"."<Table Name>" 228.9 MB 496397 rows Master table "SYSTEM". "SYS_EXPORT_SCHEMA _02" successfully loaded/unloaded	

任務	Description	所需的技能
	``` ********** ********** ********** ********** **** Dump file set for SYSTEM.SYS_EXPORT_ SCHEMA_02 is: E:\DMSDUMP\EXPORT_ DMS_DATA.DMP Job "SYSTEM"."SYS_EXPO RT_SCHEMA_02" successfully completed at *** *** ** **:**:** **** elapsed 0 00:01:57 ```	

使用 Oracle Data Pump 匯入 EnterpriseOne 結構描述

任務	Description	所需的技能
將傾印檔案傳輸至目標執行個體。	若要使用 DBMS_FILE_TRANSFER 公用程式傳輸檔案，您需要建立從來源資料庫到 Amazon RDS for Oracle 執行個體的資料庫連結。建立連結後，您可以使用 公用程式將 Data Pump 檔案直接傳輸到 Amazon RDS 執行個體。 或者，您可以將 Data Pump 檔案傳輸至 Amazon Simple Storage Service (Amazon S3)，然後將其匯入 Amazon RDS for Oracle 執行個體。如需此選項的詳細資訊，請參閱其他資訊一節。	DBA

任務	Description	所需的技能

	若要在目標資料庫執行個體建立ORARDSDB連線至 Amazon RDS 主要使用者的資料庫連結，請在來源資料庫上執行下列命令：
	
	`sqlplus / as sysdba`
	
	`SQL*Plus: Release 19.0.0.0.0 on *** *** ** **:**:** **** Version 19.3.0.0.0`
	
	`Copyright (c) 1982, 2019, Oracle. All rights reserved.`
	
	`Connected to: Oracle Database 19c Standard Edition 2 Release 19.0.0.0.0 Version 19.3.0.0.0`
	
	`SQL> create database link orardsdb connect to admin identifie d by "******" using '(DESCRIPTION = (ADDRESS = (PROTOCOL = TCP)(HOST = orcl.******.us-east-1.rds.amazonaws.com)(PORT = 1521))(CONNECT_DATA = (SERVER = DEDICATED ) (SERVICE_NAME = orcl)))';`
	
	`Database link created.`
	
	`SQL>`	

任務	Description	所需的技能

測試資料庫連結。	測試資料庫連結，以確保您可以使用 連線至 Amazon RDS for Oracle 目標資料庫`sqlplus`。
	
	`SQL> select name from v$database@orardsdb;`
	
	`NAME`
	`---------`
	`ORCL`
		DBA

任務	Description	所需的技能

將傾印檔案傳輸到目標資料庫。	若要將傾印檔案複製到 Amazon RDS for Oracle 資料庫，您可以使用預設DATA_PUMP_DIR 目錄，或使用下列程式碼建立自己的目錄，該程式碼必須在目標 Amazon RDS 執行個體上執行：
	`exec rdsadmin.rdsadmin_util.create_directory(p_directory_name => 'DMS_TARGET_PUMP_DIR');`
	`PL/SQL procedure successfully completed.`
	下列指令碼會使用名為 的資料庫連結，將名為 的傾印檔案EXPORT_DMS_DATA.DMP 從來源執行個體複製到目標 Amazon RDS for Oracle 資料庫orardsdb。您必須在來源資料庫執行個體上執行指令碼。
	`BEGIN DBMS_FILE_TRANSFER.PUT_FILE( source_directory_object => 'DMS_DATA_PUMP_DIR', source_file_name => 'EXPORT_DMS_DATA.DMP',`
		DBA

任務	Description	所需的技能
	destination_directory_object => 'DMS_TARGET_PUMP_DIR', destination_file_name => 'EXPORT_DMS_DATA.DMP', destination_database => 'orardsdb'); END; PL/SQL procedure successfully completed.	
列出目標資料庫中的傾印檔案。	PL/SQL 程序完成後，您可以使用下列程式碼，在 Amazon RDS for Oracle 資料庫中列出資料傾印檔案： select * from table (rdsadmin.rds_file_util.listdir(p_directory => 'DMS_TARGET_PUMP_DIR'));	DBA

任務	Description	所需的技能

在目標執行個體中建立 JDE 特定使用者。	在目標執行個體中使用這些命令來建立 JD Edwards 描述檔和角色：
	
	`SQL> CREATE PROFILE "JDEPROFILE" LIMIT IDLE_TIME 15;`
	`Profile created.`
	
	`SQL> CREATE ROLE "JDE_ROLE";`
	`Role created.`
	
	`SQL> CREATE ROLE "JDEADMIN";`
	`CREATE ROLE "JDEUSER";`
	`Role created.`
	`Role created.`
	
	將必要的許可授予角色：
	
	`SQL> GRANT CREATE ANY SEQUENCE TO JDE_ROLE;`
	`GRANT DROP ANY SEQUENCE TO JDE_ROLE;`
	`GRANT CREATE ANY TRIGGER TO JDE_ROLE;`
	`GRANT DROP ANY TRIGGER TO JDE_ROLE;`
		DBA、JDE CNC

任務	Description	所需的技能

在目標執行個體中建立資料表空間。	針對涉及此遷移的結構描述，使用下列命令，在目標執行個體中建立所需的資料表空間：
	
	`SQL> CREATE TABLESPACE <Tablespace Name for Tables>;`
	`Tablespace created.`
	
	`SQL> CREATE TABLESPACE <Tablespace Name for Indexes>;`
	`Tablespace created.`
		DBA、JDE CNC

任務	Description	所需的技能
在目標資料庫上啟動匯入。	開始匯入程序之前，請使用資料傾印檔案，在目標 Amazon RDS for Oracle 資料庫上設定角色、結構描述和資料表空間。 若要執行匯入，請使用 Amazon RDS 主要使用者帳戶存取目標資料庫，並使用 `tnsnames.ora` 檔案中的連線字串名稱，其中包含 Amazon RDS for Oracle Database`tns-entry` 。如有必要，您可以包含重新映射選項，以將資料傾印檔案匯入不同的資料表空間，或在不同的結構描述名稱下匯入。 若要開始匯入，請使用下列程式碼： `impdp admin@orardsdb directory=DMS_TARGET_PUMP_DIR logfile=import.log dumpfile=EXPORT_DMS_DATA.DMP` 為了確保成功匯入，請檢查匯入日誌檔案是否有任何錯誤，並檢閱物件計數、資料列計數和無效物件等詳細資訊。如果有任何無效的物件，請重新編譯。此外，比較來源和目標資料庫物件，以確認它們相符。	DBA

使用來源和目標端點佈建 AWS DMS 複寫執行個體

任務	Description	所需的技能
下載 範本。	下載 AWS CloudFormation DMS_instance.yaml 範本，以佈建 AWS DMS 複寫執行個體及其來源和目標端點。	雲端管理員，DBA
開始建立堆疊。	1. 登入 AWS 管理主控台，然後開啟位於 https：//https://console.aws.amazon.com/cloudformation 的 AWS CloudFormation 主控台。 2. 選擇建立堆疊。 3. 對於 Specify template (指定範本)，選擇 Upload a template file (上傳範本檔案)。 4. 選擇選擇檔案。 5. 選擇 `DMS_instance.yaml` 檔案。 6. 選擇下一步。	雲端管理員，DBA
指定參數。	1. 針對堆疊名稱，輸入您的堆疊名稱。 2. 針對 AWS DMS 執行個體參數，輸入下列參數： • DMSInstanceType – 根據您的業務需求，選擇 AWS DMS 複寫執行個體所需的執行個體。 • DMSStorageSize – 根據您的遷移大小，輸入 AWS DMS 執行個體的儲存體大小。	雲端管理員，DBA

任務	Description	所需的技能
	3. 針對來源 Oracle 資料庫組態，輸入下列參數： • SourceOracleEndpointID – 來源 Oracle 資料庫伺服器名稱 • SourceOracleDatabaseName – 適用時的來源資料庫服務名稱或工作階段 ID (SID) • SourceOracleUserName – 來源資料庫使用者名稱（預設為 `system`） • SourceOracleDBPassword – 來源資料庫使用者名稱的密碼 • SourceOracleDBPort – 來源資料庫連接埠 4. 針對目標 RDS for Oracle 資料庫組態，輸入下列參數： • TargetRDSOracleEndpointID – 目標 RDS 資料庫端點 • TargetRDSOracleDatabaseName – 目標 RDS 資料庫名稱 • TargetRSOracleUserName – 目標 RDS 使用者名稱 • TargetRDSOracleDBPassword – 目標 RDS 密碼 • TargetOracleDBPort – 目標 RDS 資料庫連接埠	

任務	Description	所需的技能
	5. 針對 VPC、子網路和安全群組組態，輸入下列參數： • VPCID – 複寫執行個體的 VPC • VPCSecurityGroupId – 複寫執行個體的 VPC 安全群組 • DMSSubnet1 – 可用區域 1 的子網路 • DMSSubnet2 – 可用區域 2 的子網路 6. 選擇下一步。	
建立堆疊。	1. 在設定堆疊選項頁面上，針對標籤輸入任何選用值。 2. 選擇下一步。 3. 在檢閱頁面上，驗證詳細資訊，然後選擇提交。 佈建應該會在大約 5-10 分鐘內完成。當 AWS CloudFormation Stacks 頁面顯示 CREATE_COMPLETE 時即完成。	雲端管理員，DBA

任務	Description	所需的技能
設定端點。	1. 開啟位於 https：//https://console.aws.amazon.com/dms/v2/ 的 AWS DMS 主控台。 2. 針對資源管理，選擇複寫執行個體，然後檢閱複寫執行個體。 3. 針對資源管理，選擇端點，然後檢閱端點。	雲端管理員，DBA
測試連線能力。	來源和目標端點將狀態顯示為作用中之後，請測試連線。針對每個端點（來源和目標）選擇執行測試，以確保狀態顯示為成功。	雲端管理員，DBA

建立即時複寫的 AWS DMS 複寫任務

任務	Description	所需的技能
建立複寫任務。	使用下列步驟建立 AWS DMS 複寫任務： 1. 開啟位於 https：//https://console.aws.amazon.com/dms/v2/ 的 AWS DMS 主控台。 2. 在導覽窗格的遷移資料下，選擇資料庫遷移任務。 3. 在任務組態方塊中，針對任務識別符，輸入您的任務識別符。	雲端管理員，DBA

任務	Description	所需的技能
	4. 針對複寫執行個體，選擇您建立的 DMS 複寫執行個體。 5. 針對來源資料庫端點，選擇您的來源端點。 6. 針對目標資料庫端點，選擇您的目標 Amazon RDS for Oracle 資料庫。 7. 針對遷移類型，選擇僅複寫資料變更。如果您收到需要開啟補充記錄的訊息，請遵循故障診斷一節中的指示。 8. 在任務設定方塊中，選擇指定日誌序號。 9. 針對系統變更號碼，輸入您從來源 Oracle 資料庫產生的 Oracle 資料庫 SCN。 10選擇啟用驗證。 11選擇啟用 CloudWatch Logs。 透過啟用此功能，您可以驗證資料和 Amazon CloudWatch 日誌，以檢閱 AWS DMS 複寫執行個體日誌。 12.在選取規則下，完成下列操作： • 針對結構描述，選擇輸入結構描述。 • 針對結構描述名稱，輸入 JDE 結構描述名稱（例如：DV920)。	

任務	Description	所需的技能
	• 針對資料表名稱，輸入 %。 • 針對動作，選擇包含。 13選擇 Create task (建立任務)。 建立任務後，AWS DMS 會從您在 CDC 啟動模式下提供的 SCN 遷移持續變更至 Amazon RDS for Oracle 資料庫執行個體。您也可以檢閱 CloudWatch 日誌來驗證遷移。	
重複複寫任務。	重複上述步驟，為屬於遷移一部分的其他 JD Edwards 結構描述建立複寫任務。	雲端管理員、DBA、JDE CNC 管理員

驗證目標 Amazon RDS for Oracle 資料庫上的資料庫結構描述

任務	Description	所需的技能
驗證資料傳輸。	AWS DMS 任務啟動後，您可以查看任務頁面上的資料表統計資料索引標籤，以查看對資料所做的變更。 您可以在資料庫遷移任務頁面的 主控台中監控進行中複寫的狀態。 如需詳細資訊，請參閱 AWS DMS 資料驗證。	雲端管理員，DBA

切換

任務	Description	所需的技能
停止複寫。	停止複寫程序並停止來源應用程式服務。	雲端管理員，DBA
啟動 JD Edwards 應用程式。	在 AWS 上啟動目標 JD Edwards 簡報和邏輯層應用程式，並將其導向至 Amazon RDS for Oracle 資料庫。 當您存取應用程式時，應該會注意到所有連線現在都已透過 Amazon RDS for Oracle 資料庫建立。	DBA、JDE CNC 管理員
關閉來源資料庫。	確認沒有更多連線後，您可以關閉來源資料庫。	DBA

疑難排解

問題	解決方案

您會收到警告訊息，以在來源資料庫中啟用補充記錄以進行持續複寫	輸入這些命令以啟用補充記錄：
	
	`SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;`
	`SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS;`
	`SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (UNIQUE) COLUMNS;`
	`SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (FOREIGN KEY) COLUMNS;`
	`SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS;`
	`SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (UNIQUE) COLUMNS;`

問題	解決方案
AWS DMS 已關閉補充記錄。	AWS DMS 中的補充記錄預設為關閉。若要為來源 Oracle 端點開啟它： 1. 登入 AWS 管理主控台，並在 https：//https://console.aws.amazon.com/dms/v2/ 開啟 AWS DMS 主控台。 2. 選擇端點。 3. 選擇要新增補充記錄的 Oracle 來源端點。 4. 選擇 Modify (修改)。 5. 選擇進階，然後將下列程式碼新增至額外連線屬性文字方塊： `addSupplementalLogging=Y` 6. 選擇 Modify (修改)。

CDB 層級未啟用補充記錄。	1. 輸入此命令：
	`SQL> alter session set container = CDB$ROOT;`
	`Session altered.`
	2. 重複這些步驟以啟用補充記錄。

您收到錯誤訊息：「測試端點失敗：Application -Status：1020912、Application-Message：Oracle PDB 環境端點初始化失敗，不支援 LogMiner。」	如果您遇到此錯誤訊息，您可以使用 Binary Reader 而非 LogMiner。 在端點設定下，將此行新增至來源資料庫的額外連線屬性： `useLogMinerReader=N;useBfile=Y;`

相關資源

- AWS Database Migration Service 入門

- AWS Database Migration Service 的最佳實務
- 將 Oracle 資料庫遷移至 AWS 雲端
- AWS CloudFormation 的 AWS Database Migration Service 資源類型參考 AWS CloudFormation
- 使用 AWS Secrets Manager 管理您的 AWS DMS 端點憑證
- 對 AWS Database Migration Service 中的遷移任務進行故障診斷
- AWS Database Migration Service 的最佳實務

其他資訊

使用 Amazon S3 傳輸檔案

若要將檔案傳輸至 Amazon S3，您可以使用 AWS CLI 或 Amazon S3 主控台。將檔案傳輸至 Amazon S3 之後，您可以使用 Amazon RDS for Oracle 執行個體從 Amazon S3 匯入 Data Pump 檔案。

如果您選擇使用 Amazon S3 整合做為替代方法傳輸傾印檔案，請執行下列步驟：

1. 建立 S3 儲存貯體。
2. 使用 Oracle Data Pump 從來源資料庫匯出資料。
3. 將 Data Pump 檔案上傳至 S3 儲存貯體。
4. 將 Data Pump 檔案從 S3 儲存貯體下載至目標 Amazon RDS for Oracle 資料庫。
5. 使用 Data Pump 檔案執行匯入。

> **Note**
>
> 若要在 S3 和 RDS 執行個體之間傳輸大型資料檔案，建議您使用 Amazon S3 Transfer Acceleration 功能。

使用 AWS DMS 將 Oracle PeopleSoft 資料庫遷移至 AWS

sampath kathirvel，Amazon Web Services

總結

Oracle PeopleSoft 是適用於整個企業程序的企業資源規劃 (ERP) 解決方案。PeopleSoft 具有三層架構：用戶端、應用程式和資料庫。PeopleSoft 可以在 Amazon Relational Database Service (Amazon RDS) 上執行。

如果您將 Oracle 資料庫遷移至 Amazon RDS，Amazon Web Services (AWS) 可以處理備份任務和高可用性，讓您可以專心維護 PeopleSoft 應用程式及其功能。如需遷移程序期間要考量之關鍵因素的完整清單，請參閱 AWS 方案指引中的 Oracle 資料庫遷移策略。

此模式提供使用 Oracle Data Pump 搭配 AWS Database Migration Service (AWS DMS) 及其變更資料擷取 (CDC) 功能，將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle 的解決方案。

遷移 Oracle PeopleSoft 等重要 ERP 應用程式時，將停機時間降至最低是關鍵。AWS DMS 支援完全載入和持續複寫，將停機時間降至最低。 從來源資料庫到目標資料庫。AWS DMS 也提供即時監控和記錄遷移，這可協助您識別和解決可能導致停機的任何問題。

使用 AWS DMS 複寫變更時，您必須指定時間或系統變更號碼 (SCN) 做為起點，AWS DMS 才能從資料庫日誌讀取變更。請務必在伺服器上保留一段指定的時間來存取這些日誌，以確保 AWS DMS 可以存取這些變更。

先決條件和限制

先決條件

- 在您 AWS 雲端環境中佈建的 Amazon RDS for Oracle 資料庫做為目標資料庫。
- 在 AWS 雲端內部部署或 Amazon Elastic Compute Cloud (Amazon EC2) 上執行的 Oracle PeopleSoft 資料庫。

> **Note**
>
> 此模式專為從現場部署遷移到 AWS 而設計，但在 Amazon EC2 執行個體上使用 Oracle Database 進行測試。若要從內部部署遷移，您需要設定適當的網路連線。

- 結構描述詳細資訊。將 Oracle PeopleSoft 應用程式遷移至 Amazon RDS for Oracle 時，必須識別要遷移的 Oracle 資料庫結構描述 （ 例如 SYSADM)。開始遷移程序之前，請收集下列有關結構描述的詳細資訊：

- 大小
- 每個物件類型的物件數量
- 無效物件的數量。

此資訊將有助於遷移程序。

限制

- 此案例僅使用 PeopleSoft DEMO 資料庫進行測試。它尚未使用大型資料集進行測試。

Architecture

下圖顯示執行 Oracle 資料庫做為來源資料庫的執行個體，以及執行 Amazon RDS for Oracle 資料庫做為目標資料庫的執行個體。資料會使用 Oracle Data Pump 從來源 Oracle 資料庫匯出和匯入至目標 Amazon RDS for Oracle 資料庫，並使用 AWS DMS 複寫 CDC 變更。

1. 初始步驟涉及使用 Oracle Data Pump 從來源資料庫擷取資料，然後傳送至 Amazon RDS for Oracle 資料庫目標。
2. 資料會從來源資料庫傳送至 AWS DMS 中的來源端點。
3. 從來源端點，資料會傳送至執行複寫任務的 AWS DMS 複寫執行個體。
4. 複寫任務完成後，資料會傳送至 AWS DMS 中的目標端點。
5. 從目標端點，資料會傳送至 Amazon RDS for Oracle 資料庫執行個體。

工具

AWS 服務

- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。
- 適用於 Oracle 的 Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展 Oracle 關聯式資料庫。

其他服務

- Oracle Data Pump 可協助您以高速將資料和中繼資料從一個資料庫移至另一個資料庫。

最佳實務

遷移 LOBs

如果您的來源資料庫包含需要遷移至目標資料庫的大型二進位物件 (LOBs)，AWS DMS 會提供下列選項：

- 完整 LOB 模式 – AWS DMS 會將來源的所有 LOBs 遷移至目標資料庫，無論其大小為何。雖然遷移速度較慢，但優點是資料不會被截斷。為了獲得更好的效能，您可以在新的複寫執行個體上建立單獨的任務，以遷移 LOBs 大於數個 MB 的資料表。
- 有限 LOB 模式 – 您可以指定 LOB 資料欄資料的大小上限，這可讓 AWS DMS 預先配置資源並大量套用 LOBs。如果 LOB 資料欄的大小超過任務中指定的大小，AWS DMS 會截斷資料，並將警告傳送至 AWS DMS 日誌檔案。如果您的 LOB 資料大小在有限 LOB 大小內，您可以使用有限 LOB 模式來改善效能。
- 內嵌 LOB 模式 – 您可以透過複寫小型和大型 LOBs 來遷移 LOBs而無需截斷資料或降低任務的效能。首先，指定 InlineLobMaxSize 參數的值，只有在完整 LOB 模式設定為 true 時才能使用。AWS DMS 任務會內嵌傳輸小型 LOBs，這更有效率。然後，AWS DMS 會透過從來源資料表執行查詢來遷移大型 LOBs。不過，內嵌 LOB 模式僅適用於完全載入階段。

產生序列值

請記住，在 AWS DMS 的變更資料擷取過程中，不會從來源資料庫複寫增量序號。為了避免序列值的差異，您必須從所有序列的來源產生最新的序列值，並將其套用至目標 Amazon RDS for Oracle 資料庫。

登入資料管理

為了協助保護您的 AWS 資源，建議您遵循 AWS Identity and Access Management (IAM) 的最佳實務。

史詩

使用來源和目標端點佈建 AWS DMS 複寫執行個體

任務	Description	所需的技能
下載 範本。	下載 DMS_instance.yaml AWS CloudFormation 範本，以佈建	雲端管理員，DBA

任務	Description	所需的技能
	AWS DMS 複寫執行個體及其來源和目標端點。	
開始建立堆疊。	1. 在 AWS 管理主控台上，選擇 CloudFormation。 2. 選擇建立堆疊。 3. 對於 Specify template (指定範本)，選擇 Upload a template file (上傳範本檔案)。 4. 選擇選擇檔案。 5. 選擇 `DMS_instance.yaml` 檔案。 6. 選擇下一步。	雲端管理員，DBA

任務	Description	所需的技能
指定參數。	1. 針對堆疊名稱，輸入您的堆疊名稱。 2. 在 AWS DMS 執行個體參數下，輸入下列參數： • DMSInstanceType – 根據您的業務需求，選擇 AWS DMS 複寫執行個體所需的執行個體。 • DMSStorageSize – 根據您的遷移大小，輸入 AWS DMS 執行個體的儲存體大小。 3. 在來源 Oracle 資料庫組態下，輸入下列參數： • SourceOracleEndpointID – 來源 Oracle 資料庫伺服器名稱 • SourceOracleDatabaseName – 適用時的來源資料庫服務名稱或工作階段 ID (SID) • SourceOracleUserName – 來源資料庫使用者名稱（預設為系統） • SourceOracleDBPassword – 來源資料庫使用者名稱的密碼 • SourceOracleDBPort – 來源資料庫連接埠 4. 在目標 RDS for Oracle 資料庫組態下，輸入下列參數：	雲端管理員，DBA

任務	Description	所需的技能
	• TargetRDSOracleEndpointID – 目標 RDS 資料庫端點 • TargetRDSOracleDatabaseName – 目標 RDS 資料庫名稱 • TargetRSOracleUserName – 目標 RDS 使用者名稱 • TargetRDSOracleDBPassword – 目標 RDS 密碼 • TargetOracleDBPort – 目標 RDS 資料庫連接埠 5. 在 VPC、子網路和安全群組組態下，輸入下列參數： • VPCID – 複寫執行個體的 VPC • VPCSecurityGroupId – 複寫執行個體的 VPC 安全群組 • DMSSubnet1 – 可用區域 1 的子網路 • DMSSubnet2 – 可用區域 2 的子網路 6. 選擇下一步。	

任務	Description	所需的技能
建立堆疊。	1. 在設定堆疊選項頁面上，針對標籤輸入任何選用值。 2. 選擇下一步。 3. 在檢閱頁面上，驗證詳細資訊，然後選擇提交。 佈建應該會在大約 5-10 分鐘內完成。當 AWS CloudFormation Stacks 頁面顯示 CREATE_COMPLETE 時即完成。	雲端管理員，DBA
設定端點。	1. 從 AWS 管理主控台中，選擇資料庫遷移服務。 2. 在資源管理下，選擇複寫執行個體。 3. 在資源管理下，選擇端點。	雲端管理員，DBA
測試連線能力。	來源和目標端點將狀態顯示為作用中之後，請測試連線能力。針對每個端點（來源和目標）選擇執行測試，以確保狀態顯示為成功。	雲端管理員，DBA

使用 Oracle Data Pump 從現場部署 Oracle 資料庫匯出 PeopleSoft 結構描述

任務	Description	所需的技能
產生 SCN。	當來源資料庫處於作用中狀態且應用程式正在使用中時，請使用 Oracle Data Pump 啟動資料匯出。您必須先從來源資料庫產生系統變更編號	DBA

任務	Description	所需的技能

	(SCN)，以便在使用 Oracle Data Pump 匯出期間達到資料一致性，並做為 AWS DMS 中擷取變更資料的起點。
	
	若要從來源資料庫產生目前的 SCN，請輸入下列 SQL 陳述式。
	
	`SQL> select name from v$database;`
	`SQL> select name from v$database;`
	`NAME`
	`---------`
	`PSFTDMO`
	`SQL> SELECT current_scn FROM v$database;`
	`CURRENT_SCN`
	`-----------`
	`23792008`
	
	儲存產生的 SCN，以便在匯出資料和建立 AWS DMS 複寫任務時使用。	

任務	Description	所需的技能

建立 參數檔案。	若要建立參數檔案以匯出結構描述，您可以使用下列程式碼。
	
	`$ cat exp_datapmp.par`
	`userid=system/*******`
	`directory=DATA_P`
	`UMP_DIR`
	`logfile=export_dms_`
	`sample_user.log`
	`dumpfile=export_dms_`
	`sample_data_%U.dmp`
	`schemas=SYSADM`
	`flashback_scn=237920`
	`08`
	
	Note
	您也可以根據您的需求DATA_PUMP_DIR ，使用以下命令來定義自己的命令。
	
	`SQL> CREATE OR REPLACE`
	` DIRECTORY DATA_PUMP`
	`_DIR AS '/opt/oracle/`
	`product/19c/dbhome_1/`
	`dmsdump/';`
	`Directory created.`
	`SQL> GRANT READ, WRITE`
	` ON DIRECTORY DATA_PUMP`
	`_DIR TO system;`
	`Grant succeeded.`
	`SQL>`
	`SQL> SELECT owner,`
	` directory_name,`
	` directory_path FROM`
		DBA

任務	Description	所需的技能
	dba_directories WHERE directory_name='DA TA_PUMP_DIR'; OWNER DIRECTORY_NAME DIRECTORY_PATH ---------- ---------- ---------- ---------- ---------- ---------- SYS DATA_PUMP_DIR /opt/ oracle/product/19c/dbh ome_1/dmsdump/	

任務	Description	所需的技能

匯出結構描述。	若要執行匯出，請使用expdp公用程式。
	
	`$ expdp parfile=exp_datapmp.par`
	`.....`
	`Transferring the dump file with DBMS_FILE_TRANSFER to Target:`
	`. . exported "SYSADM"."PS_XML_TEMPLT_LNG" 6.320 KB 0 rows`
	`. . exported "SYSADM"."PS_XML_TEMPLT_LNK" 6.328 KB 0 rows`
	`. . exported "SYSADM"."PS_XML_XLATDEF_LNG" 6.320 KB 0 rows`
	`. . exported "SYSADM"."PS_XML_XLATITM_LNG" 7.171 KB 0 rows`
	`. . exported "SYSADM"."PS_XPQRYRUNCNTL" 7.601 KB 0 rows`
	`. . exported "SYSADM"."PS_XPQRYRUNPARM" 7.210 KB 0 rows`
	`. . exported "SYSADM"."PS_YE_AMOUNTS" 9.351 KB 0 rows`
	`. . exported "SYSADM"."PS_YE_DATA" 16.58 KB 0 rows`
	`. . exported "SYSADM"."PS_YE_EE" 6.75 KB 0 rows`
	`. . exported "SYSADM"."PS_YE_W2CP_AMOUNTS" 9.414 KB 0 rows`
		DBA

任務	Description	所需的技能
	. . exported "SYSADM". "PS_YE_W2CP_DATA" 20.94 KB 0 rows . . exported "SYSADM". "PS_YE_W2C_AMOUNTS" 10.27 KB 0 rows . . exported "SYSADM". "PS_YE_W2C_DATA" 20.95 KB 0 rows . . exported "SYSADM". "PS_ZBD_JOBCODE_TBL" 14.60 KB 0 rows . . exported "SYSADM". "PTGRANTTBL" 5.468 KB 0 rows Master table "SYSTEM". "SYS_EXPORT_SCHEMA _01" successfully loaded/unloaded ** Dump file set for SYSTEM.SYS_EXPORT_ SCHEMA_01 is: /opt/oracle/pr oduct/19c/dbhome_1 /dmsdump/export_dm s_sample_data_01.dmp Job "SYSTEM"."SYS_EXPO RT_SCHEMA_01" successfully completed at Mon Dec 19 20:13:57 2022 elapsed 0 00:38:22	

使用 Oracle Data Pump 將 PeopleSoft 結構描述匯入 Amazon RDS for Oracle 資料庫

任務	Description	所需的技能
將傾印檔案傳輸至目標執行個體。	若要使用 傳輸檔案DBMS_FILE_TRANSFER ，您需要建立	DBA

任務	Description	所需的技能

	從來源資料庫到 Amazon RDS for Oracle 執行個體的資料庫連結。建立連結後，您可以使用公用程式直接將 Data Pump 檔案傳輸到 RDS 執行個體。
	
	或者，您可以將 Data Pump 檔案傳輸至 Amazon Simple Storage Service (Amazon S3)，然後將其匯入 Amazon RDS for Oracle 執行個體。如需此選項的詳細資訊，請參閱其他資訊一節。
	
	若要在目標資料庫執行個體建立ORARDSDB連線至 Amazon RDS 主要使用者的資料庫連結，請在來源資料庫上執行下列命令。
	
	`$sqlplus / as sysdba`
	`$ SQL> create database link orardsdb connect to admin identified by "*****" using '(DESCRIPTION = (ADDRESS = (PROTOCOL = TCP)(HOST = testpsft.*******.us-west-2.rds.amazonaws.com)(PORT = 1521))(CONNECT_DATA = (SERVER = DEDICATED) (SERVICE_NAME = orcl)))';`
	`Database link created.`	

任務	Description	所需的技能
測試資料庫連結。	測試資料庫連結，以確保您可以使用 sqlplus 連線到 Amazon RDS for Oracle 目標資料庫。 `SQL> SQL> select name from v$database@orardsdb; NAME --------- ORCL SQL>`	DBA

任務	Description	所需的技能

將傾印檔案傳輸至目標資料庫。	若要將傾印檔案複製到 Amazon RDS for Oracle 資料庫，您可以使用預設DATA_PUMP_DIR 目錄，或使用下列程式碼建立自己的目錄。
	
	`exec rdsadmin.rdsadmin_util.create_directory(p_directory_name => 'TARGET_PUMP_DIR') ;`
	
	下列指令碼會使用名為 的資料庫連結，將名為 的傾印檔案export_dms_sample_data_01.dmp 從來源執行個體複製到目標 Amazon RDS for Oracle 資料庫orardsdb。
	
	`$ sqlplus / as sysdba SQL> BEGIN DBMS_FILE_TRANSFER.PUT_FILE( source_directory_object => 'DATA_PUMP_DIR', source_file_name => 'export_dms_sample_data_01.dmp', destination_directory_object => 'TARGET_PUMP_DIR'', destination_file_name => 'export_dms_sample_data_01.dmp',`
		DBA

任務	Description	所需的技能

	`destination_database => 'orardsdb' );`
	`END;`
	`/`
	`PL/SQL procedure successfully completed.`	

列出目標資料庫中的傾印檔案。	PL/SQL 程序完成後，您可以使用下列程式碼，在 Amazon RDS for Oracle 資料庫中列出資料傾印檔案。 `SQL> select * from table (rdsadmin.rds_file_util.listdir(p_directory => 'TARGET_PUMP_DIR'));`	DBA

任務	Description	所需的技能
在目標資料庫上啟動匯入。	開始匯入程序之前，請使用資料傾印檔案，在目標 Amazon RDS for Oracle 資料庫上設定角色、結構描述和資料表空間。 若要執行匯入，請使用 Amazon RDS 主要使用者帳戶存取目標資料庫，並使用 `tnsnames.ora` 檔案中的連線字串名稱，其中包含 Amazon RDS for Oracle Database `tns-entry` 。如有必要，您可以包含重新映射選項，將資料傾印檔案匯入不同的資料表空間或不同的結構描述名稱。 若要開始匯入，請使用下列程式碼。 `impdp admin@orardsdb directory=TARGET_PUMP_DIR logfile=import.log dumpfile=export_dms_sample_data_01.dmp` 為了確保成功匯入，請檢查匯入日誌檔案是否有任何錯誤，並檢閱物件計數、資料列計數和無效物件等詳細資訊。如果有任何無效的物件，請重新編譯。此外，比較來源和目標資料庫物件，以確認它們相符。	DBA

使用 CDC 建立 AWS DMS 複寫任務來執行即時複寫

任務	Description	所需的技能
建立複寫任務。	使用下列步驟建立 AWS DMS 複寫任務： 1. 在 AWS DMS 主控台的轉換和遷移下，選擇資料庫遷移任務。 2. 在任務組態下，針對任務識別符，輸入您的任務識別符。 3. 針對複寫執行個體，選擇您建立的 DMS 複寫執行個體。 4. 針對來源資料庫端點，選擇您的來源端點。 5. 針對目標資料庫端點，選擇您的目標 Amazon RDS for Oracle 資料庫。 6. 針對遷移類型，選擇僅複寫資料變更。如果您收到需要開啟補充記錄的訊息，請遵循其他資訊一節中的指示。 7. 在任務設定下，選取指定日誌序號。 8. 針對系統變更號碼，輸入您從來源 Oracle 資料庫產生的 Oracle 資料庫 SCN。 9. 選擇啟用驗證。 10選擇啟用 CloudWatch Logs。	雲端管理員，DBA

任務	Description	所需的技能
	透過啟用此功能，您可以驗證資料和 Amazon CloudWatch 日誌，以檢閱 AWS DMS 複寫執行個體日誌。 11.在選取規則下，完成下列操作： • 針對結構描述，選擇輸入結構描述。 • 針對結構描述名稱，輸入 SYSADM。 • 針對資料表名稱，輸入 %。 • 針對動作，選擇包含。 12.在轉換規則下，完成下列操作： • 針對目標，選擇資料表。 • 針對結構描述名稱，選擇輸入結構描述。 • 針對結構描述名稱，輸入 SYSADM。 • 針對動作，選擇重新命名。 13.選擇 Create task (建立任務)。 建立任務之後，它會從您在 CDC 啟動模式下提供的 SCN 將 CDC 遷移至 Amazon RDS for Oracle 資料庫執行個體。您	

任務	Description	所需的技能
	也可以檢閱 CloudWatch 日誌來驗證 。	

驗證目標 Amazon RDS for Oracle 資料庫上的資料庫結構描述

任務	Description	所需的技能
驗證資料傳輸。	AWS DMS 任務啟動後，您可以查看任務頁面上的資料表統計資料索引標籤，以查看對資料所做的變更。 您可以在資料庫遷移任務頁面的 主控台中監控進行中複寫的狀態。 如需詳細資訊，請參閱 AWS DMS 資料驗證。	雲端管理員，DBA

切換

任務	Description	所需的技能
停止複寫。	停止複寫程序並停止來源應用程式服務。	雲端管理員，DBA
啟動 PeopleSoft 中間層。	在 AWS 中啟動目標 PeopleSoft 中層應用程式，並將其導向至最近遷移的 Amazon RDS for Oracle 資料庫。 當您存取應用程式時，您應該會注意到所有應用程式連線	DBA，PeopleSoft 管理員

任務	Description	所需的技能
	現在都已與 Amazon RDS for Oracle 資料庫建立。	
關閉來源資料庫。	確認不再有來源資料庫的連線後，即可將其關閉。	DBA

相關資源

- AWS Database Migration Service 入門
- AWS Database Migration Service 的最佳實務
- 將 Oracle 資料庫遷移至 AWS 雲端

其他資訊

使用 Amazon S3 傳輸檔案

若要將檔案傳輸至 Amazon S3，您可以使用 AWS CLI 或 Amazon S3 主控台。將檔案傳輸至 Amazon S3 之後，您可以使用 Amazon RDS for Oracle 執行個體從 Amazon S3 匯入 Data Pump 檔案。

如果您選擇使用 Amazon S3 整合做為替代方法傳輸傾印檔案，請執行下列步驟：

1. 建立 S3 儲存貯體。
2. 使用 Oracle Data Pump 從來源資料庫匯出資料。
3. 將 Data Pump 檔案上傳至 S3 儲存貯體。
4. 將 Data Pump 檔案從 S3 儲存貯體下載至目標 Amazon RDS for Oracle 資料庫。
5. 使用 Data Pump 檔案執行匯入。

> **Note**
>
> 若要在 S3 和 RDS 執行個體之間傳輸大型資料檔案，建議使用 Amazon S3 Transfer Acceleration 功能。

啟用補充記錄

如果您收到警告訊息，在來源資料庫中啟用補充記錄以進行持續複寫，請使用下列步驟。

```
SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;
SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS;
SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (UNIQUE) COLUMNS;
SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (FOREIGN KEY) COLUMNS;
SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS
SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (UNIQUE) COLUMNS;
```

將內部部署 MySQL 資料庫遷移至 Amazon RDS for MySQL

Lorenzo Mota，Amazon Web Services

摘要

此模式提供將內部部署 MySQL 資料庫遷移至 MySQL 的 Amazon Relational Database Service (Amazon RDS) MySQL 的指引。模式討論使用 AWS Database Migration Service (AWS DMS) 或原生 MySQL 工具進行完整的資料庫遷移，例如 mysqldump。此模式主要用於DBAs解決方案架構師。它可用於小型或大型專案做為測試程序（我們建議至少有一個測試週期）或做為最終遷移程序。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 內部部署資料中心中的 MySQL 來源資料庫

限制

- 資料庫大小限制：64 TB

產品版本

- MySQL 5.5、5.6、5.7、8.0 版。如需支援版本的最新清單，請參閱 AWS 文件中的 Amazon RDS 上的 MySQL。如果您使用的是 AWS DMS，另請參閱 使用 MySQL 相容資料庫作為目前支援的 for MySQL 版本的目標 AWS DMS AWS DMS。 MySQL

Architecture

來源技術堆疊

- 內部部署 MySQL 資料庫

目標技術堆疊

- 執行 MySQL 的 Amazon RDS 資料庫執行個體

目標架構

下圖顯示遷移後的目標 Amazon RDS for MySQL 實作。

AWS 資料遷移架構

使用 AWS DMS：

下圖顯示使用 AWS DMS 傳送完整和增量變更時的資料遷移架構，直到切換為止。從內部部署到 的網路連線 AWS 取決於您的需求，且超出此模式的範圍。

使用原生 MySQL 工具：

下圖顯示當您使用原生 MySQL 工具時的資料遷移架構。匯出傾印檔案會複製到 Amazon Simple Storage Service (Amazon S3)，並在切換 AWS 之前匯入 中的 Amazon RDS for MySQL 資料庫。從內部部署到 的網路連線 AWS 取決於您的需求，且超出此模式的範圍。

備註：

- 根據停機時間需求和資料庫的大小，使用 AWS DMS 或變更資料擷取 (CDC) 工具可將切換時間降至最低。 AWS DMS 可協助將新目標的切換時間縮短到最短 （通常為 分鐘）。如果資料庫和網路延遲的大小允許短時間，則 mysqldump 的 離線策略就足夠。（我們建議進行測試以取得大約的時間。）
- 通常， 等 CDC 策略 AWS DMS 比離線選項需要更多的監控和複雜性。

工具

- AWS 服務：AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端 或在雲端和內部部署設定的組合之間遷移。如需 支援的 MySQL 來源和 目標資料庫的相關資訊 AWS DMS，請參閱 將 MySQL 相容資料庫遷移至 AWS。如果您的來源資料庫不受 支援 AWS DMS，您必須選擇另一種方法來遷移資料。
- 原生 MySQL 工具：mysqldump
- 第三方工具：Percona XtraBackup

史詩

規劃遷移

任務	Description	所需的技能
驗證資料庫版本。	驗證來源和目標資料庫版本。	DBA
識別硬體需求。	識別目標伺服器的硬體需求。	DBA，系統管理員
識別儲存需求。	識別目標資料庫的儲存需求（例如儲存類型和容量）。	DBA，系統管理員
選擇執行個體類型。	根據容量、儲存功能和聯網功能選擇目標執行個體類型。	DBA，系統管理員
識別網路存取需求。	識別來源和目標資料庫網路存取的安全需求。	DBA，系統管理員
識別不支援的物件。	識別不支援的物件（如果有的話）並判斷遷移工作。	DBA
識別相依性。	識別遠端資料庫上的任何相依性。	DBA
決定應用程式遷移策略。	決定遷移用戶端應用程式的策略。	DBA、應用程式擁有者、系統管理員

設定基礎設施

任務	Description	所需的技能
建立 Virtual Private Cloud (VPC)	設定路由表、網際網路閘道、NAT 閘道和子網路。如需詳細資訊，請參閱 Amazon RDS 文件中的 VPCs 和 Amazon RDS。	系統管理員

任務	Description	所需的技能
建立安全群組。	根據您的需求設定連接埠和 CIDR 範圍或特定 IPs。MySQL 的預設連接埠為 3306。如需詳細資訊，請參閱 Amazon RDS 文件中的使用安全群組控制存取。	系統管理員
設定和啟動 Amazon RDS for MySQL 資料庫執行個體。	如需說明，請參閱《Amazon RDS 文件》中的建立 Amazon RDS 資料庫執行個體。檢查支援的版本。	系統管理員

遷移資料 – 選項 1（使用原生工具）

任務	Description	所需的技能
使用原生 MySQL 工具或第三方工具來遷移資料庫物件和資料。	如需說明，請參閱 MySQL 工具的文件，例如 mysqldump 和 Percona XtraBackup （適用於實體遷移）。 如需選項的詳細資訊，請參閱 MySQL 遷移至 Amazon RDS for MySQL 或 Amazon Aurora MySQL 的部落格文章。	DBA

遷移資料 – 選項 2（使用 AWS DMS)

任務	Description	所需的技能
使用 遷移資料 AWS DMS。	如需說明，請參閱 AWS DMS 文件。	DBA

在切換之前執行初步任務

任務	Description	所需的技能
修正物件計數差異。	從來源資料庫和新目標資料庫收集物件計數。修正目標資料庫中的差異。	DBA
檢查相依性。	檢查往返其他資料庫的相依性（連結）是否有效並如預期運作。	DBA
執行測試。	如果這是一個測試週期，請執行查詢測試、收集指標並修正問題。	DBA

切換

任務	Description	所需的技能
切換到目標資料庫。	將用戶端應用程式切換到新的基礎設施。	DBA、應用程式擁有者、系統管理員
提供測試支援。	提供功能應用程式測試的支援。	DBA

關閉專案

任務	Description	所需的技能
關閉資源。	關閉您為遷移建立的臨時 AWS 資源。	DBA，系統管理員
驗證專案文件。	檢閱並驗證專案文件。	DBA、應用程式擁有者、系統管理員

任務	Description	所需的技能
收集指標。	收集遷移時間、手動與自動化工作的百分比、節省成本等指標。	DBA、應用程式擁有者、系統管理員
關閉專案。	關閉專案並提供意見回饋。	DBA、應用程式擁有者、系統管理員
停用來源資料庫。	當所有遷移和切換任務完成時，請停用現場部署資料庫。	DBA，系統管理員

相關資源

參考

- 關聯式資料庫的遷移策略
- AWS DMS website
- AWS DMS 文件
- Amazon RDS 文件
- Amazon RDS 定價
- Amazon VPC 和 Amazon RDS
- Amazon RDS 異地同步備份部署
- 使用 Percona XtraBackup、Amazon EFS 和 Amazon S3 將內部部署 MySQL 資料庫遷移至 Aurora MySQL
- Amazon RDS 資料庫執行個體儲存體

教學課程

- 入門 AWS DMS
- Amazon RDS 入門

將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon RDS for SQL Server

Henrique Lobao、Jonathan Pereira Cruz 和 Amazon Web Services Vishal Singh

總結

此模式提供從內部部署 Microsoft SQL Server 資料庫遷移至 SQL Server 的 Amazon Relational Database Service (Amazon RDS) 的指引。其中說明兩種遷移選項：使用 AWS Data Migration Service (AWS DMS) 或使用 Copy Database Wizard 等原生 Microsoft SQL Server 工具。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署資料中心中的來源 Microsoft SQL Server 資料庫

限制

- 資料庫大小限制：16 TB

產品版本

- 如需支援版本和功能的最新清單，請參閱 AWS 文件中的 Amazon RDS 上的 Microsoft SQL Server。如果您使用的是 AWS DMS，另請參閱使用 Microsoft SQL Server 資料庫做為 AWS DMS for SQL Server 版本支援的目標。

Architecture

來源技術堆疊

- 內部部署 Microsoft SQL Server 資料庫

目標技術堆疊

- Amazon RDS for SQL Server 資料庫執行個體

來源和目標架構

使用 AWS DMS：

使用原生 SQL Server 工具：

工具

- AWS DMS 支援多種類型的來源和目標資料庫。如需詳細資訊，請參閱 AWS DMS Step-by-Step演練。如果 AWS DMS 不支援來源資料庫，請選取另一個方法來遷移資料。
- 原生 Microsoft SQL Server 工具包括備份和還原、複製資料庫精靈、複製和連接資料庫。

史詩

規劃遷移

任務	Description	所需的技能
驗證來源和目標資料庫版本和引擎。		DBA
識別目標伺服器執行個體的硬體需求。		DBA，系統管理員
識別儲存需求（儲存類型和容量）。		DBA，系統管理員
根據容量、儲存功能和網路功能選擇適當的執行個體類型。		DBA，系統管理員
識別來源和目標資料庫的網路存取安全需求。		DBA，系統管理員
識別應用程式遷移策略。		DBA，系統管理員

設定基礎設施

任務	Description	所需的技能
建立 Virtual Private Cloud (VPC)		系統管理員
建立安全群組。		系統管理員
設定和啟動 Amazon RDS 資料庫執行個體。		DBA，系統管理員

遷移資料 - 選項 1

任務	Description	所需的技能
使用原生 SQL Server 工具或第三方工具來遷移資料庫物件和資料。		DBA

遷移資料 - 選項 2

任務	Description	所需的技能
使用 AWS DMS 遷移資料。		DBA

遷移應用程式

任務	Description	所需的技能
遵循應用程式遷移策略。		DBA、應用程式擁有者、系統管理員

剪下

任務	Description	所需的技能
將應用程式用戶端切換到新的基礎設施。		DBA、應用程式擁有者、系統管理員

關閉專案

任務	Description	所需的技能
關閉臨時 AWS 資源。		DBA，系統管理員
檢閱並驗證專案文件。		DBA、應用程式擁有者、系統管理員
收集指標，例如遷移時間、手動與自動任務的百分比，以及節省成本。		DBA、應用程式擁有者、系統管理員
關閉專案並提供意見回饋。		DBA、應用程式擁有者、系統管理員

相關資源

參考

- 在 Amazon Web Services 上部署 Microsoft SQL Server
- AWS DMS 網站
- Amazon RDS 定價
- AWS 上的 Microsoft 產品
- AWS 上的 Microsoft 授權
- AWS 上的 Microsoft SQL Server
- 搭配 Microsoft SQL Server 資料庫執行個體使用 Windows 身分驗證
- Amazon RDS 異地同步備份部署

教學課程和影片

- AWS DMS 入門
- Amazon RDS 入門
- AWS DMS （影片）
- Amazon RDS （影片）

使用 Rclone 將資料從 Microsoft Azure Blob 遷移至 Amazon S3

Suhas Basavaraj、Acidian Keane 和 Corey Lane，Amazon Web Services

總結

此模式說明如何使用 Rclone 將資料從 Microsoft Azure Blob 物件儲存遷移至 Amazon Simple Storage Service (Amazon S3) 儲存貯體。您可以使用此模式來執行一次性遷移或持續同步資料。Rclone 是以 Go 編寫的命令列程式，用於從雲端提供者跨各種儲存技術移動資料。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 存放在 Azure Blob 容器服務中的資料

Architecture

來源技術堆疊

- Azure Blob 儲存容器

目標技術堆疊

- Amazon S3 儲存貯體
- Amazon Elastic Compute Cloud (Amazon EC2) Linux 執行個體

架構

工具

- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Rclone 是以 rsync 為設計來源的開放原始碼命令列程式。它用於管理多個雲端儲存平台的檔案。

最佳實務

當您將資料從 Azure 遷移至 Amazon S3 時，請注意這些考量，以避免不必要的成本或緩慢的傳輸速度：

- 在與 Azure 儲存帳戶和 Blob 容器相同的地理區域中建立 AWS 基礎設施，例如 AWS 區域 `us-east-1`（維吉尼亞北部）和 Azure 區域 `East US`。
- 如果可能，請避免使用 NAT Gateway，因為它會同時產生輸入和輸出頻寬的資料傳輸費用。
- 使用 Amazon S3 的 VPC 閘道端點來提高效能。
- 考慮使用 AWS Graviton2 (ARM) 處理器型 EC2 執行個體，相較於 Intel x86 執行個體，成本更低且效能更高。Rclone 是高度跨編譯的，並提供預先編譯的 ARM 二進位檔。

史詩

準備 AWS 和 Azure 雲端資源

任務	Description	所需的技能
準備目的地 S3 儲存貯體。	在適當的 AWS 區域中建立新的 S3 儲存貯體，或選擇現有的儲存貯體做為您要遷移資料的目的地。	AWS 管理員
為 Amazon EC2 建立 IAM 執行個體角色。	為 Amazon EC2 建立新的 AWS Identity and Access Management (IAM) 角色。此角色可讓 EC2 執行個體寫入目的地 S3 儲存貯體。	AWS 管理員
將政策連接至 IAM 執行個體角色。	使用 IAM 主控台或 AWS Command Line Interface (AWS CLI) 為 EC2 執行個體角色建立內嵌政策，允許對目的地 S3 儲存貯體的寫入存取許可。如需範例政策，請參閱其他資訊一節。	AWS 管理員

任務	Description	所需的技能
啟動 EC2 執行個體。	啟動設定為使用新建立 IAM 服務角色的 Amazon Linux EC2 執行個體。此執行個體也需要透過網際網路存取 Azure 公有 API 端點。 **Note** 請考慮使用 AWS Graviton 型 EC2 執行個體來降低成本。Rclone 提供 ARM 編譯的二進位檔。	AWS 管理員
建立 Azure AD 服務主體。	使用 Azure CLI 建立對來源 Azure Blob 儲存容器具有唯讀存取權的 Azure Active Directory (Azure AD) 服務主體。如需說明，請參閱其他資訊一節。將這些登入資料存放在您的 EC2 執行個體上至位置 `~/azure-principal.json` 。	雲端管理員，Azure

安裝和設定 Rclone

任務	Description	所需的技能
下載並安裝 Rclone。	下載並安裝 Rclone 命令列程式。如需安裝說明，請參閱 Rclone 安裝文件。	一般 AWS、雲端管理員
設定 Rclone。	複製下列`rclone.conf` 範例檔案。`AZStorageAccount`	一般 AWS、雲端管理員

任務	Description	所需的技能
	將 取代為您的 Azure Storage 帳戶名稱，並將 us-east-1 取代為您的 S3 儲存貯體所在的 AWS 區域。將此檔案儲存至 EC2 執行個體~/.config/rclone/rclone.conf 上的 位置。 [AZStorageAccount] type = azureblob account = AZStorageAccount service_principal_file = azure-principal.json [s3] type = s3 provider = AWS env_auth = true region = us-east-1	

任務	Description	所需的技能

驗證 Rclone 組態。	若要確認 Rclone 已設定且許可正常運作，請確認 Rclone 可以剖析您的組態檔案，以及可存取 Azure Blob 容器和 S3 儲存貯體中的物件。如需驗證命令範例，請參閱下列內容。
	• 在組態檔案中列出設定的遠端。這將確保正確剖析您的組態檔案。檢閱輸出，確認其與您的`rclone.conf` 檔案相符。
	`rclone listremotes`
	`AZStorageAccount:`
	`s3:`
	• 列出已設定帳戶中的 Azure Blob 容器。`AZStorage Account` 將 取代為您在 `rclone.conf` 檔案中使用的儲存體帳戶名稱。
	`rclone lsd AZStorage`
	`Account:`
	`2020-04-29 08:29:26`
	`docs`
	• 列出 Azure Blob 容器中的檔案。將此命令中的文件取代為 Azure 儲存帳戶中的實際 Blob 容器名稱。
	`rclone ls AZStorage`
	`Account:docs`
	`824884 administr`
	`ator-en.a4.pdf`
		一般 AWS、雲端管理員

任務	Description	所需的技能

	• 列出 AWS 帳戶中的儲存貯體。
	
	`[root@ip-10-0-20-157 ~]# rclone lsd s3:`
	`2022-03-07 01:44:40 amzn-s3-demo-bucket1`
	`2022-03-07 01:45:16 amzn-s3-demo-bucket2`
	`2022-03-07 02:12:07 amzn-s3-demo-bucket3`
	
	• 列出 S3 儲存貯體中的檔案。
	
	`[root@ip-10-0-20-157 ~]# rclone ls s3:amzn-s3-demo-bucket1`
	`template0.yaml`
	`template1.yaml`	

使用 Rclone 遷移資料

任務	Description	所需的技能
從容器遷移資料。	執行 Rclone 複製或同步命令。 範例：複製 此命令會將來源 Azure Blob 容器的資料複製到目的地 S3 儲存貯體。	一般 AWS、雲端管理員

任務	Description	所需的技能

	`rclone copy AZStorageAccount:blob-container s3:amzn-s3-demo-bucket1`
	
	範例：同步
	
	此命令會同步來源 Azure Blob 容器與目的地 S3 儲存貯體之間的資料。
	
	`rclone sync AZStorageAccount:blob-container s3:amzn-s3-demo-bucket1`
	
	Important
	當您使用同步命令時，來源容器中不存在的資料將從目的地 S3 儲存貯體中刪除。	

同步您的容器。	初始複製完成後，請執行 Rclone 同步命令以進行持續遷移，以便僅複製目的地 S3 儲存貯體中遺失的新檔案。	一般 AWS、雲端管理員
確認資料已成功遷移。	若要檢查資料是否已成功複製到目的地 S3 儲存貯體，請執行 Rclone lsd 和 ls 命令。	一般 AWS、雲端管理員

相關資源

- Amazon S3 使用者指南 (AWS 文件)

- Amazon EC2 的 IAM 角色 (AWS 文件)
- 建立 Microsoft Azure Blob 容器 (Microsoft Azure 文件)
- Rclone 命令 (Rclone 文件)

其他資訊

EC2 執行個體的角色政策範例

此政策可讓 EC2 執行個體讀取和寫入存取您帳戶中的特定儲存貯體。如果您的儲存貯體使用客戶受管金鑰進行伺服器端加密，則政策可能需要額外存取 AWS Key Management Service (AWS KMS)。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:DeleteObject",
                "s3:GetObject",
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": [
              "arn:aws:s3:::amzn-s3-demo-bucket/*",
              "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "arn:aws:s3:::*"
        }
    ]
}
```

建立唯讀 Azure AD 服務主體

Azure 服務主體是客戶應用程式、服務和自動化工具用來存取特定 Azure 資源的安全身分。將其視為具有特定角色的使用者身分 （登入和密碼或憑證），以及存取 資源的嚴格控制許可。若要建立唯讀服務主體以遵循最低權限許可，並保護 Azure 中的資料免於意外刪除，請遵循下列步驟：

1. 登入您的 Microsoft Azure 雲端帳戶入口網站，並在 PowerShell 中啟動 Cloud Shell，或在工作站上使用 Azure Command-Line Interface (CLI)。
2. 建立服務主體，並使用 Azure Blob 儲存體帳戶的唯讀存取權進行設定。將此命令的 JSON 輸出儲存至名為 的本機檔案`azure-principal.json`。檔案將上傳至您的 EC2 執行個體。使用 Azure 訂閱 ID、資源群組名稱和儲存體帳戶名稱取代以括號 ({ 和 }) 顯示的預留位置變數。

```
az ad sp create-for-rbac `
--name AWS-Rclone-Reader `
--role "Storage Blob Data Reader" `
--scopes /subscriptions/{Subscription ID}/resourceGroups/{Resource Group Name}/
providers/Microsoft.Storage/storageAccounts/{Storage Account Name}
```

從 Couchbase Server 遷移至 AWS 上的 Couchbase Capella

Battulga Purevragchaa 和 Saurabh Shanbhag，Amazon Web Services

Mark Gamble，無

總結

Couchbase Capella 是完全受管的 NoSQL 資料庫即服務 (DBaaS)，適用於關鍵任務應用程式（例如，使用者設定檔或線上目錄和庫存管理）。Couchbase Capella 會在 Couchbase 管理的 Amazon Web Services (AWS) 帳戶中管理您的 DBaaS 工作負載。Capella 可讓您在單一界面中輕鬆執行和管理多叢集、多 AWS 區域、多雲端和混合雲端複寫。

Couchbase Capella 可協助您立即擴展 Couchbase Server 應用程式，協助您在幾分鐘內建立多節點叢集。Couchbase Capella 支援所有 Couchbase Server 功能，包括SQL++、全文搜尋、事件服務和分析服務。它也不需要管理安裝、升級、備份和一般資料庫維護。

此模式說明將自我管理 Couchbase Server 環境遷移至 AWS 雲端的步驟和最佳實務。模式提供可重複的程序，將資料和索引從內部部署或雲端執行的 Couchbase Server 叢集遷移至 Couchbase Capella。使用這些步驟可協助您避免在遷移期間發生問題，並加速整體遷移程序。

此模式提供下列兩個遷移選項：

- 如果您的遷移索引少於 50 個，則選項 1 是適當的。
- 如果您有超過 50 個要遷移的索引，則選項 2 是適當的。

您也可以在自我管理的 Couchbase 伺服器上設定範例資料，以遵循遷移指南。

如果您選擇遷移選項 2，或者您使用的是預設值以外的範圍或集合，則必須使用範例組態檔案，其位於其他資訊區段中。

先決條件和限制

先決條件

- 現有的 Couchbase Capella 付費帳戶。您也可以在 AWS 上建立 Couchbase Capella 帳戶，並使用 Couchbase Capella 免費試用版，然後升級到付費帳戶來設定叢集以進行遷移。若要開始使用試用版，請遵循 Couchbase Capella 入門中的指示。

- 在內部部署或部署在雲端服務提供者上的現有自我管理 Couchbase Server 環境。
- 對於遷移選項 2、Couchbase Shell 和組態檔案。若要建立組態檔案，您可以使用其他資訊區段中的範例檔案。
- 熟悉管理 Couchbase Server 和 Couchbase Capella。
- 熟悉在命令列界面 (CLI) 中開啟 TCP 連接埠和執行命令。

遷移程序也需要下表所述的角色和專業知識。

Role	專業知識	責任
Couchbase 管理員	• 熟悉 Couchbase Server 和 Couchbase Capella • 基本命令列知識很有幫助，但並非必要	• Couchbase Server 和 Capella 特定任務
系統管理員、IT 管理員	• 熟悉自我管理的 Couchbase Server 系統環境和管理	• 在自我管理的 Couchbase Server 叢集節點上開啟連接埠並判斷 IP 地址

限制

- 此模式用於將資料、索引和 Couchbase 全文搜尋索引從 Couchbase 伺服器遷移至 AWS 上的 Couchbase Capella。此模式不適用於遷移 Couchbase Eventing Service 或 Couchbase Analytics。
- Couchbase Capella 可在多個 AWS 區域中使用。如需 Capella 支援區域up-to-date，請參閱 Couchbase 文件中的 Amazon Web Services。

產品版本

- Couchbase Server (Community 或 Enterprise) Edition 5.x 版或更新版本

Architecture

來源技術堆疊

- Couchbase 伺服器

目標技術堆疊

- Couchbase Capella

目標架構

1. 您可以使用 Capella 控制平面存取 Couchbase Capella。您可以使用 Capella 控制平面來執行下列動作：
   - 控制和監控您的帳戶。
   - 管理叢集和資料、索引、使用者和群組、存取許可、監控和事件。
2. 叢集已建立。
3. Capella Data Plane 位於 Couchbase 管理的 AWS 帳戶中。建立新叢集之後，Couchbase Capella 會將叢集部署到所選 AWS 區域中的多個可用區域。
4. 您可以在 AWS 帳戶中的 VPC 中開發和部署 Couchbase 應用程式。一般而言，此 VPC 會透過 VPC 對等互連存取 Capella 資料平面。

工具

- Couchbase 跨資料中心複寫 (XDCR) 有助於跨位於不同雲端提供者和不同資料中心的叢集複寫資料。它用於將資料從自我管理的 Couchbase Server 叢集遷移到 Couchbase Capella。

> **Note**
>
> XDCR 無法與 Couchbase Server Community Edition 搭配使用，以遷移至 Couchbase Capella。反之，您可以使用 cbexport。如需詳細資訊，請參閱從 Community Edition Epic 遷移資料。

- Couchbase Shell 是 Couchbase Server 和 Couchbase Capella 存取本機和遠端 Couchbase 叢集的命令列 Shell。在此模式中，Couchbase Shell 用於遷移索引。
- cbexport 是用於從 Couchbase 叢集匯出資料的 Couchbase 公用程式。包含在 Couchbase Server CLI 工具中。

史詩

準備遷移

任務	Description	所需的技能
評估自我管理 Couchbase Server 叢集的大小。	登入 Couchbase Server 的 Couchbase Web 主控台，並評估自我管理叢集的節點和儲存貯體。 1. 若要顯示叢集節點清單，請選擇導覽列中的伺服器索引標籤。 2. 記錄節點數量，然後選擇清單中的每個節點以顯示其屬性。 3. 記錄每個個別節點的記憶體和儲存體。 4. 選擇導覽列中的儲存貯體索引標籤，然後選擇清單中的每個儲存貯體以顯示其屬性。記錄每個儲存貯體的 RAM 配額和衝突解決設定。 您將使用自我管理的 Couchbase Server 叢集組態作為在 Couchbase Capella 上調整和設定目標叢集的一般指南。 如需更詳細 Couchbase Capella 大小練習的說明，請聯絡 Couchbase。	Couchbase 管理員

任務	Description	所需的技能
在自我管理的 Couchbase Server 叢集上記錄 Couchbase Service 分佈。	1. 在 Couchbase Web 主控台上，選擇伺服器索引標籤以顯示叢集節點清單。 2. 選擇每個節點以顯示其屬性，然後記錄每個節點的 Couchbase Service 分佈 (資料服務、查詢服務、索引服務、搜尋服務、分析服務和事件服務)。	Couchbase 管理員
記錄自我管理 Couchbase Server 叢集節點的 IP 地址。	（如果您使用的是 Community Edition，請忽略此步驟。）記錄叢集中每個節點的 IP 地址。它們稍後會新增至 Couchbase Capella 叢集上的允許清單。	Couchbase 管理員、系統管理員

在 Couchbase Capella 上部署和設定資源

任務	Description	所需的技能
選擇一個範本。	1. 登入您的 Couchbase Capella 控制平面，選擇主導覽中的儀表板索引標籤或叢集索引標籤，然後選擇建立叢集。 2. 使用您從自我管理 Couchbase Server 叢集評估中記錄的資訊，選擇符合組態需求的叢集範本。如果您沒有找到適當的範本，請在叢集大小編輯器中選擇自訂範本。	Couchbase 管理員

任務	Description	所需的技能
選擇並設定節點。	選擇並設定節點以符合自我管理的 Couchbase Server 叢集環境，包括節點數量、服務分佈、運算或 RAM，以及儲存體。 Couchbase Capella 使用多維度擴展最佳實務。只能根據部署最佳實務來選擇服務和節點。這可能表示您無法完全符合自我管理的 Couchbase Server 叢集組態。	Couchbase 管理員

任務	Description	所需的技能
部署叢集。	選擇支援區域和支援套件，然後部署叢集。如需詳細步驟和說明，請參閱 Couchbase 文件中的建立叢集。 **Important** 如果您使用的是 Couchbase Capella 免費試用，您必須在開始遷移之前將其轉換為付費帳戶。若要轉換您的帳戶，請開啟 Couchbase Capella 控制平面的帳單區段，然後選擇新增啟用 ID。在您與 Couchbase Sales 完成購買協議後，或透過 AWS Marketplace 進行購買後，啟用 ID 會傳送到您的帳單聯絡人電子郵件地址。	Couchbase 管理員

任務	Description	所需的技能
建立資料庫登入資料使用者。	資料庫登入資料使用者專屬於叢集，由使用者名稱、密碼和一組儲存貯體權限組成。建立儲存貯體和存取儲存貯體資料時，需要此使用者。 在 Couchbase Capella 控制平面中，遵循 Couchbase Capella 文件中的設定資料庫憑證中的指示，為新叢集建立資料庫憑證。 **Note** 如果組織使用者想要從遠端或透過 Couchbase Capella UI 存取特定叢集上的儲存貯體資料，則需要指派給他們的組織角色憑證。這與資料庫登入資料不同，通常由應用程式和整合使用。建立組織使用者可讓您在 Couchbase Capella 叢集上建立和管理目標儲存貯體。	Couchbase 管理員

任務	Description	所需的技能
如果使用遷移選項 2，請安裝 Couchbase Shell。	您可以在可存取自我管理 Couchbase Server 和 Couchbase Capella 叢集的任何系統上安裝 Couchbase Shell。如需詳細資訊，請參閱 Couchbase Shell 文件中的 Install Couchbase Shell 1.0.0-beta.5 版。 在命令列終端機中測試與自我管理叢集的連線，以確認已安裝 Couchbase Shell。	Couchbase 管理員、系統管理員

任務	Description	所需的技能
允許 IP 地址。	1. 在 Couchbase Capella 控制平面中，選擇叢集，然後選擇您的目標叢集。 2. 選擇叢集的連線索引標籤，並記錄在管理允許 IP 下叢集的連線端點。 3. 若要將安裝 Couchbase Shell 的系統 IP 地址和自我管理 Couchbase Server 叢集執行個體的 IP 地址新增為允許的 IP 地址，請執行下列動作： a. 在廣域網路下，選擇管理允許的 IP。 b. 選擇新增允許 IP，輸入您安裝 Couchbase Shell 之系統的 IP 地址，然後選擇新增 IP。 c. 重複上一個步驟，新增自我管理 Couchbase Server 叢集執行個體的 IP 地址。 如需允許 IP 地址的詳細資訊，請參閱 Couchbase 文件中的設定允許的 IP 地址。	Couchbase 管理員、系統管理員

任務	Description	所需的技能
設定憑證。	1. 若要下載叢集的根憑證，請在根憑證下選擇下載。 2. 使用 .pem 副檔名將根憑證儲存在系統上執行 Couchbase Shell 的資料夾中。 3. 接著，登入自我管理的 Couchbase Server Web 主控台，在左側導覽列中選擇安全性，然後選擇憑證索引標籤。 4. 複製自我管理 Couchbase Server 叢集的根憑證，並將其儲存為 .pem 檔案到您儲存 Couchbase Capella 叢集根憑證檔案的相同資料夾。如需根憑證的詳細資訊，請參閱 Couchbase Server 文件中的根憑證。	Couchbase 管理員、系統管理員

任務	Description	所需的技能
建立 Couchbase Shell 的組態檔案。	在 Couchbase Shell 安裝的主目錄中建立組態點檔案（例如，`/<HOME_DIRECTORY>/.cbsh/config`）。如需詳細資訊，請參閱 Couchbase 文件中的Config dotfiles。 將來源和目標叢集的連線屬性新增至組態檔案。您可以使用其他資訊區段中的範例組態檔案，並編輯叢集的設定。 將具有更新設定的組態檔案儲存至 `.cbsh` 資料夾（例如，`/<HOME_DIRECTORY>/.cbsh/config`）。	Couchbase 管理員、系統管理員
建立目標儲存貯體。	對於每個來源儲存貯體，請遵循 Couchbase 文件中的建立儲存貯體中的指示，在 Couchbase Capella 叢集中建立一個目標儲存貯體。 您的目標儲存貯體組態必須符合自我管理 Couchbase Server 叢集中儲存貯體的儲存貯體名稱、記憶體設定和衝突解決設定。	Couchbase 管理員

任務	Description	所需的技能

建立範圍和集合。	每個儲存貯體都包含具有 金鑰空間 的預設範圍和集合_default._default 。如果您針對範圍和集合使用任何其他金鑰空間，您必須在目標 Capella 叢集中建立相同的金鑰空間。
	
	1. 在您安裝 Couchbase Shell 的系統上開啟命令列終端機。
	2. 若要啟動 Couchbase Shell，請執行下列命令。
	
	`./cbsh`
	
	3. 對於您要遷移的每個儲存貯體，請執行下列命令，在 Capella 叢集中建立範圍和集合。請務必將 <BUCKET_NAME> 取代為您要遷移的儲存貯體名稱。
	
	`scopes --clusters "On-Prem-Cluster" --bucket <BUCKET_NAME> | select scope | where scope != "_default" | each { |it| scopes create $it.scope --clusters "Capella-Cluster" } collections --clusters "On-Prem-Cluster" --bucket <BUCKET_NAME> | select scope collection | where $it.scope != "_default`
		Couchbase 管理員

任務	Description	所需的技能
	`" | where $it.colle ction != "_default" | each { |it| collectio ns create $it.colle ction --clusters "Capella-Cluster" -- bucket <BUCKET_NAME> -- scope $it.scope }`	

從 Enterprise Edition 遷移資料

任務	Description	所需的技能
在自我管理的 Couchbase Server 叢集節點上開啟 TCP 連接埠。	確定在自我管理的 Couchbase Server 叢集節點上開啟 XDCR 通訊的適當連接埠。如需詳細資訊，請參閱 Couchbase Server 連接埠文件。	Couchbase 管理員、系統管理員
如果您使用的是 Couchbase Server Enterprise Edition，請設定 Couchbase XDCR。	1. 在 Couchbase Capella 控制平面主要導覽中，選擇叢集，然後選擇要遷移的目標叢集。 2. 在根憑證下，選擇複製。 3. 登入您的自我管理 Couchbase Server Web 主控台，然後在主導覽中選擇 XDCR。然後選擇新增遠端。 4. 輸入以下設定： • 叢集名稱 – Capella 叢集連線的名稱	Couchbase 管理員

任務	Description	所需的技能
	• IP/Hostname – Couchbase Capella 叢集的連線端點 • 遠端叢集的使用者名稱 – Couchbase Capella 叢集的資料庫使用者 • 密碼 – Couchbase Capella 叢集的資料庫使用者密碼 • 啟用安全連線 – 已選取 • 完整 (TLS 加密密碼和資料） – 已選取 5. 貼上您先前複製的 Capella 叢集根憑證，然後選擇儲存。	
啟動 Couchbase XDCR。	1. 在自我管理的 Couchbase Server Web 主控台中，選擇主導覽中的 XDCR，然後選擇新增複寫。 2. 輸入以下設定： • 從儲存貯體複寫 – 選取來源儲存貯體以進行遷移。 • 遠端儲存貯體 – 輸入目標儲存貯體名稱。 • 遠端叢集 – 選取您先前建立的目標叢集。 3. 選擇儲存複寫。複寫程序應該會在幾秒鐘內開始。	Couchbase 管理員

使用選項 1 遷移索引

任務	Description	所需的技能
將自我管理叢集索引遷移至 Couchbase Capella。	**⚠ Important** 如果您有少於 50 個要遷移的索引，建議您執行此程序。如果您有超過 50 個要遷移的索引，建議您使用遷移選項 2。 1. 在 Couchbase Web 主控台上，選擇索引。 2. 在索引清單中，選擇您要遷移的第一個索引。接著會顯示索引定義。 3. 使用 CREATE陳述式複製索引定義，但不複製 `WITH { "defer_build":true }` 。 例如，從下列範例索引定義中，您只能複製 `CREATE INDEX `cityindex` ON `travel-sample`(`city`)` 。 `CREATE INDEX `cityindex` ON `travel-sample`(`city`) WITH { "defer_build":true }`	Couchbase 管理員、系統管理員

任務	Description	所需的技能
	4. 在 Couchbase Capella 控制平面中，選擇叢集，然後選擇目標叢集。 5. 在工具下拉式清單中，選擇查詢工作台。將您先前複製的CREATE陳述式貼入查詢編輯器，然後選擇執行。這會建立並建置索引。 6. 若要確認已建立索引，請從工具下拉式清單中選擇索引。此清單顯示已建立並建置索引。 7. 為每個必須遷移的索引重複此程序。	

使用選項 2 遷移索引

任務	Description	所需的技能
遷移索引定義。	**Important** 如果您有超過 50 個要遷移的索引，建議您執行此程序。如果您的遷移索引少於 50 個，建議您使用遷移選項 1。 1. 在您安裝 Couchbase Shell 的系統上開啟命令列終端機。 2. 若要啟動 Couchbase Shell，請執行下列命令。	Couchbase 管理員、系統管理員

任務	Description	所需的技能

	`./cbsh`
	3. 若要連線至自我管理的 Couchbase Server 叢集，請執行下列命令。
	`cb-env cluster On-Prem-Cluster`
	4. 若要將索引定義從自我管理的 Couchbase Server 叢集遷移至 Couchbase Capella 叢集，請針對您要遷移的每個儲存貯體執行下列命令。請務必<BUCKET_NAME> 將取代為對應至您要遷移之索引的儲存貯體名稱。此遷移選項需要您的目標儲存貯體名稱與來源儲存貯體名稱相同。
	`query indexes --definitions | where bucket =~ <BUCKET_NAME>| get definition | each { |it| query $it --clusters Capella-Cluster }`	

任務	Description	所需的技能

建置索引定義。	1. 若要切換內容至 Couchbase Capella 叢集，請執行下列命令：
	
	`cb-env cluster Capella-Cluster`
	
	2. 若要建置遷移至 Couchbase Capella 叢集的索引定義，請執行下列命令，將 取代<BUCKET_NAME> 為對應至您要建置之索引的儲存貯體名稱。
	
	`query 'SELECT RAW CONCAT("BUILD INDEX ON ", k , "(['", CONCAT2 ("','", inames), "']);") FROM system:indexes AS s LET bid = CONCAT("` ",s.bucket_id, "`"), sid = CONCAT("`", s.scope_id, "`"), kid = CONCAT("` ", s.keyspace_id, "`"), k = NVL2(bid, CONCAT2(".", bid, sid, kid), kid) WHERE s.namespa ce_id = "default" AND s.bucket_id = "" GROUP BY k LETTING inames = ARRAY_AGG (s.name) FILTER (WHERE s.state = 'deferred') HAVING ARRAY_LENGTH(iname`
		Couchbase 管理員、系統管理員

任務	Description	所需的技能
	`s) > 0;' | each { | it| query $it }` 3. 為每個儲存貯體重複此步驟。	

遷移全文搜尋索引

任務	Description	所需的技能
將自我管理叢集全文搜尋索引遷移至 Couchbase Capella。	1. 在 Couchbase Web 主控台中，選擇搜尋。 2. 在全文搜尋 (FTS) 索引清單中，選擇您要遷移的第一個 FTS 索引，選擇顯示索引定義 JSON，然後選擇複製到剪貼簿。記下索引名稱及其所屬的儲存貯體。 3. 在 Couchbase Capella 控制平面中，選擇叢集，然後選擇目標叢集。 4. 在工具下拉式清單中，選擇全文搜尋。 5. 選擇匯入索引，然後貼上 FTS 索引定義。 6. 輸入索引名稱，選取正確的儲存貯體，如自我管理叢集所述，然後選擇建立。 7. 為每個必須遷移的 FTS 索引重複此程序。	Couchbase 管理員

從 Couchbase Community Edition 遷移資料

任務	Description	所需的技能

從自我管理的 Couchbase Server Community Edition 匯出資料。	Couchbase Community Edition 不提供加密的 XDCR。您可以從 Couchbase Community Edition 匯出資料，然後將資料手動匯入 Couchbase Capella。
	
	若要從來源儲存貯體匯出資料，`cbexport`請在命令列使用 。
	
	以下命令提供為範例。
	
	`cbexport json \`
	`--cluster localhost \`
	`--bucket <SOURCE BUCKET NAME> \`
	`--format lines \`
	`--username <USERNAME> \`
	`--password <PASSWORD> \`
	`--include-key cbkey \`
	`--scope-field cbscope \`
	`--collection-field cbcoll \`
	`--output cbexported_data.json`
	
	請注意，`cbkey`、`cbcoll`、`cbscope`和 `cbexported_data.json` 是任意標籤。稍後會在程序中參考它們，因此如果您選擇以不同的方式命名它們，請記下來。
		Couchbase 管理員

任務	Description	所需的技能
將資料匯入 Couchbase Capella。	1. 在 Couchbase Capella 控制平面中，選擇叢集，然後選擇目標叢集。 2. 在工具下拉式清單中，選擇匯入。這將開啟具有下列六個步驟的精靈： a. 儲存貯體 – 選擇目標儲存貯體。 b. 檔案 – 選擇 JSON、選擇行，然後選擇使用您的 Web 瀏覽器。如果您有大量資料，您可以探索手動選項。選取 建立的檔案`cbexport`。 c. 集合 – 選擇自訂集合映射。 如果您的 Community Edition 資料庫不使用範圍或集合，或僅使用 _default，您可以改為選擇選取單一集合選項。 針對集合映射表達式，輸入 `%cbscope%.%cbcoll%` 。若要驗證此表達式是否正常運作，您可以貼上範例資料，如下所示。 `{ "cbscope" :"inventory", "cbcoll":"landmark", "cbkey":" landmark_3991" }`	Couchbase 管理員

任務	Description	所需的技能
	d. 金鑰 – 選擇客戶產生。（如果您不在乎保留要匯入的資料金鑰，您可以改為選取自動產生 UUID，然後繼續進行步驟 5。）針對金鑰名稱產生器表達式，輸入 %cbkey%。若要驗證此表達式是否正常運作，請貼上一些範例資料。 e. 組態 – 選擇忽略欄位，然後輸入 cbscope、cbcoll、cbkey。這些欄位包含暫時性資訊，在匯入後不需要位於目標儲存貯體中。將其他設定保留為各自的預設設定。 f. 匯入 – 檢閱，然後在準備好時選擇匯入。等待上傳和資料匯入。 對於大型檔案，Couchbase Capella 支援使用 cURL 匯入命令列。您可以在 Couchbase Capella 文件的匯入資料中更詳細地探索匯入選項。	

測試並驗證遷移

任務	Description	所需的技能
驗證資料遷移。	1. 在 Couchbase Capella 控制平面中，選擇叢集，然後在叢集清單中選擇目標叢集。 2. 選擇目標叢集的儲存貯體標籤。確認目標儲存貯體中的項目（文件）數量符合來源儲存貯體中的項目數量。 3. 在目標叢集的工具下拉式清單中，選擇文件。確認所有文件都已遷移。 4. （選用）遷移所有資料後，您可以透過刪除複寫來關閉複寫。如需詳細資訊，請參閱 Couchbase 文件中的刪除複寫。	Couchbase 管理員
驗證索引遷移。	在 Couchbase Capella 控制平面中，在目標叢集的工具下拉式清單中，選擇索引。確認索引已遷移並建置。	Couchbase 管理員
驗證查詢結果。	1. 在 Couchbase Capella 控制平面中，在目標叢集的工具下拉式清單中，選擇查詢工作台。 2. 執行範例 N1QL 查詢或應用程式中使用的查詢。請確定您收到的結果與在自我管理的 Couchbase Server 叢集中執行查詢時的結果相同。	Couchbase 管理員

任務	Description	所需的技能
驗證全文搜尋結果（如果您遷移 FTS 索引，則適用）。	1. 在 Couchbase Capella 控制平面中，在目標叢集的工具下拉式清單中，選擇全文搜尋。 2. 選擇名稱以選取 FTS 索引。 3. 選擇 Search (搜尋)。 4. 輸入範例搜尋查詢，然後選擇搜尋。 5. 確認結果與在自我管理叢集上執行搜尋時相同。	Couchbase 管理員

相關資源

準備遷移

- 開始使用 Couchbase Capella 免費試用
- Couchbase Capella 的雲端供應商需求
- Couchbase Capella 調整大小準則

遷移資料和索引

- Couchbase XDCR
- Couchbase Shell 文件

Couchbase Capella SLAs 和支援

- Couchbase Capella 服務層級協議 (SLAs)
- Couchbase Capella Service 支援政策

其他資訊

下列程式碼是 Couchbase Shell 的範例組態檔案。

```
Version = 1

[[clusters]]
identifier = "On-Prem-Cluster"
hostnames = ["<SELF_MANAGED_COUCHBASE_CLUSTER>"]
default-bucket = "travel-sample"
username = "<SELF_MANAGED_ADMIN>"
password = "<SELF_MANAGED_ADMIN_PWD>"
tls-cert-path = "/<ABSOLUTE_PATH_TO_SELF_MANAGED_ROOT_CERT>"
data-timeout = "2500ms"
connect-timeout = "7500ms"
query-timeout = "75s"

[[clusters]]
identifier = "Capella-Cluster"
hostnames = ["<COUCHBASE_CAPELLA_ENDPOINT>"]
default-bucket = "travel-sample"
username = "<CAPELLA_DATABASE_USER>"
password = "<CAPELLA_DATABASE_USER_PWD>"
tls-cert-path = "/<ABSOLUTE_PATH_TO_COUCHBASE_CAPELLA_ROOT_CERT>"
data-timeout = "2500ms"
connect-timeout = "7500ms"
query-timeout = "75s"
```

儲存組態檔案之前，請使用下表來確保您已新增自己的來源和目標叢集資訊。

<SELF_MANAGED_COUCHBASE_CLUSTER>	使用自我管理 Couchbase Server 叢集的 IP 地址。
<SELF_MANAGED_ADMIN>	將管理員使用者用於自我管理的 Couchbase Server 叢集。
<ABSOLUTE_PATH_TO_SELF_MANAGED_ROOT_CERT>	針對自我管理的 Couchbase Server 叢集，使用儲存根憑證檔案的絕對路徑。
<COUCHBASE_CAPELLA_ENDPOINT>	使用 Couchbase Capella 叢集的連線端點。
<CAPELLA_DATABASE_USER>	為您的 Couchbase Capella 叢集使用資料庫使用者。

<CAPELLA_DATABASE_USER_PWD>	為您的 Couchbase Capella 叢集使用資料庫使用者密碼。
<ABSOLUTE_PATH_TO_COUCHBASE_CAPELLA_ROOT_CERT>	為您的 Couchbase Capella 叢集使用儲存根憑證檔案的絕對路徑。

從 IBM WebSphere Application Server 遷移至 Amazon EC2 上的 Apache Tomcat

Neal Ardeljan 和 Afroz Khan，Amazon Web Services

總結

此模式會逐步引導您從執行 IBM WebSphere 應用程式伺服器 (WAS) 的內部部署 Red Hat Enterprise Linux (RHEL) 6.9 或更新版本系統遷移至在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上執行 Apache Tomcat 的 RHEL 8。

模式可套用至下列來源和目標版本：

- WebSphere Application Server 7.x 到 Apache Tomcat 8（使用 Java 7 或更新版本）
- WebSphere Application Server 8.x 到 Apache Tomcat 8（使用 Java 7 或更新版本）
- WebSphere Application Server 8.5.5.x 到 Apache Tomcat 9（使用 Java 8 或更新版本）
- WebSphere Application Server 8.5.5.x 到 Apache Tomcat 10（使用 Java 8 或更新版本）

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 來源 Java 程式碼，假設如下：
  - 使用 Java 開發套件 (JDK) 版的 Java 7 或更新版本
  - 使用 Spring 或 Apache Struts 架構
  - 未使用 Enterprise Java Beans (EJB) 架構或任何其他無法立即提供給 Tomcat 的 WebSphere 伺服器功能
  - 主要使用 servlet 或 Java 伺服器頁面 (JSPs)
  - 使用 Java Database Connectivity (JDBC) 連接器連接到資料庫
- 來源 IBM WebSphere Application Server 7.x 版或更新版本
- 目標 Apache Tomcat 8.5 版或更新版本

Architecture

Source 技術堆疊

- 使用 Apache Struts Model-View-Controller (MVC) 架構建置的 Web 應用程式
- 在 IBM WebSphere Application Server 7.x 或 8.x 版上執行的 Web 應用程式
- 使用輕量型目錄存取通訊協定 (LDAP) 連接器連線至 LDAP 目錄 (iPlanet/eTrust) 的 Web 應用程式
- 使用 IBM Tivoli Access Manager (TAM) 連線更新 TAM 使用者密碼的應用程式（目前實作中，應用程式使用 PD.jar)

內部部署資料庫

- Oracle Database 21c (21.0.0.0)
- Oracle 資料庫 19c (19.0.0.0)
- Oracle 資料庫 12c 版本 2 (12.2.0.1)
- Oracle 資料庫 12c 版本 1 (12.1.0.2)

目標技術堆疊

- 在 EC2 執行個體上的 RHEL 上執行的 Apache Tomcat 第 8 版（或更新版本）
- 適用於 Oracle 的 Amazon Relational Database Service (Amazon RDS)

如需 Amazon RDS 支援的 Oracle 版本的詳細資訊，請參閱 Amazon RDS for Oracle 網站。

目標架構

工具

- 應用程式層：將 Java 應用程式重建為 WAR 檔案。
- 資料庫層：Oracle 原生備份和還原。
- 雅加達 EE 的 Apache Tomcat 遷移工具。此工具採用針對在 Apache Tomcat 9 上執行的 Java EE 8 撰寫的 Web 應用程式，並自動將其轉換為在實作雅加達 EE 9 的 Apache Tomcat 10 上執行。

史詩

規劃遷移

任務	Description	所需的技能
完成應用程式探索、目前狀態足跡和效能基準。		BA，遷移主管
驗證來源和目標資料庫版本。		DBA
識別目標伺服器 EC2 執行個體的硬體需求。		DBA、SysAdmin
識別儲存需求（儲存類型和容量）。		DBA、SysAdmin
根據容量、儲存功能和網路功能，選擇適當的 EC2 執行個體類型。		DBA、SysAdmin
識別來源和目標資料庫的網路存取安全需求。		DBA、SysAdmin
識別應用程式遷移策略和工具。		DBA，遷移主管
完成應用程式的遷移設計和遷移指南。		組建主管、遷移主管
完成應用程式遷移 Runbook。		組建主管、切換主管、測試主管、遷移主管

設定基礎設施

任務	Description	所需的技能
建立 Virtual Private Cloud (VPC)		SysAdmin

任務	Description	所需的技能
建立安全群組。		SysAdmin
設定和啟動 Amazon RDS for Oracle。		DBA、SysAdmin

遷移資料

任務	Description	所需的技能
建立或取得端點的存取權，以擷取資料庫備份檔案。		DBA
使用原生資料庫引擎或第三方工具來遷移資料庫物件和資料。	如需詳細資訊，請參閱其他資訊區段中的「遷移資料庫物件和資料」。	DBA

遷移應用程式

任務	Description	所需的技能
記錄遷移的變更請求 (CR)。		切換潛在客戶
取得遷移的 CR 核准。		切換潛在客戶
遵循應用程式遷移執行手冊中的應用程式遷移策略。	如需詳細資訊，請參閱其他資訊區段中的「設定應用程式層」。	DBA、遷移工程師、應用程式擁有者
升級應用程式（如有必要）。		DBA、遷移工程師、應用程式擁有者
完成功能、非功能、資料驗證、SLA 和效能測試。		測試主管、應用程式擁有者、應用程式使用者

剪下

任務	Description	所需的技能
向應用程式擁有者或企業擁有者取得簽署。		切換潛在客戶
將應用程式用戶端切換到新的基礎設施。		DBA、遷移工程師、應用程式擁有者

關閉專案

任務	Description	所需的技能
關閉臨時 AWS 資源。		DBA、遷移工程師、SysAdmin
檢閱並驗證專案文件。		遷移主管
收集指標，例如遷移時間、手動與自動任務的百分比，以及節省成本。		遷移主管
關閉專案並提供意見回饋。		遷移主管，應用程式擁有者

相關資源

參考

- Apache Tomcat 10.0 文件
- Apache Tomcat 9.0 文件
- Apache Tomcat 8.0 文件
- Apache Tomcat 8.0 安裝指南
- Apache Tomcat JNDI 文件
- Amazon RDS for Oracle 網站
- Amazon RDS 定價
- Oracle 和 Amazon Web Services

- Amazon RDS 上的 Oracle
- Amazon RDS 異地同步備份部署

教學課程和影片

- Amazon RDS 入門

其他資訊

遷移資料庫物件和資料

例如，如果您使用的是原生 Oracle 備份/還原公用程式：

1. 建立資料庫備份檔案的 Amazon Simple Storage Service (Amazon S3) 備份 （選用）。
2. 將 Oracle 資料庫資料備份至網路共用資料夾。
3. 登入遷移預備伺服器以映射網路共用資料夾。
4. 將資料從網路共用資料夾複製到 S3 儲存貯體。
5. 請求 Oracle 的 Amazon RDS 異地同步備份部署。
6. 將內部部署資料庫備份還原至 Amazon RDS for Oracle。

設定應用程式層

1. 從 Apache Tomcat 網站安裝 Tomcat 8 （或 9/10)。
2. 將應用程式和共用程式庫封裝到 WAR 檔案中。
3. 在 Tomcat 中部署 WAR 檔案。
4. 監控 WebSphere `Linux cat`中任何遺失共用程式庫的啟動日誌。
5. 觀看`Linux cat`任何 WebSphere 特定部署描述項延伸項目的開始記錄。
6. 從 WebSphere 伺服器收集任何缺少的相依 Java 程式庫。
7. 使用 Tomcat 相容對等項目修改 WebSphere 特定部署描述項元素。
8. 使用相依的 Java 程式庫和更新的部署描述項重建 WAR 檔案。
9. 更新 LDAP 組態、資料庫組態和測試連線 （請參閱 Apache Tomcat 文件中的Realm Configuration How-TOandJNDI Datasource HOW-TO)。
10. 根據還原的 Amazon RDS for Oracle 資料庫測試已安裝的應用程式。
11. 從 EC2 執行個體建立 Linux 的 Amazon Machine Image (AMI)。

12.使用 Application Load Balancer 和 Auto Scaling 群組啟動完成的架構。

13.更新 URLs（使用 WebSEAL 連接）以指向 Application Load Balancer。

14.更新組態管理資料庫 (CMDB)。

使用 Auto Scaling 從 IBM WebSphere Application Server 遷移至 Amazon EC2 上的 Apache Tomcat

Kevin Yung 和 Afroz Khan，Amazon Web Services

總結

此模式提供在啟用 Amazon EC2 Auto Scaling 的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上，將 Java 應用程式從 IBM WebSphere Application Server 遷移至 Apache Tomcat 的指引。Amazon EC2 Auto Scaling

透過使用此模式，您可以實現：

- 降低 IBM 授權成本
- 使用異地同步備份部署的高可用性
- 使用 Amazon EC2 Auto Scaling 改善應用程式彈性

先決條件和限制

先決條件

- Java 應用程式 (7.x 或 8.x 版） 應該在 LAMP 堆疊中開發。
- 目標狀態是在 Linux 主機上託管 Java 應用程式。此模式已成功在 Red Hat Enterprise Linux (RHEL) 7 環境中實作。其他 Linux 發行版本可以遵循此模式，但應該參考 Apache Tomcat 發行版本的組態。
- 您應該了解 Java 應用程式的相依性。
- 您必須有權存取 Java 應用程式原始碼才能進行變更。

限制和修改變更

- 您應該了解企業封存 (EAR) 元件，並確認所有程式庫都封裝在 Web 元件 WAR 檔案中。您需要設定 Apache Maven WAR 外掛程式並產生 WAR 檔案成品。
- 使用 Apache Tomcat 8 時，servlet-api.jar 和應用程式套件內建 jar 檔案之間存在已知衝突。若要解決此問題，請從應用程式套件中刪除 servlet-api.jar。
- 您必須設定位於 Apache Tomcat 組態 classpath 中的 WEB-INF/resources。根據預設，JAR 程式庫不會載入 目錄中。或者，您可以在 src/main/resources 下部署所有資源。

- 檢查 Java 應用程式中的任何硬式編碼內容根目錄，並更新 Apache Tomcat 的新內容根目錄。
- 若要設定 JVM 執行期選項，您可以在 Apache Tomcat bin 資料夾中建立組態檔案 setenv.sh，例如 JAVA_OPTS、JAVA_HOME 等。
- 驗證是在容器層級設定，並在 Apache Tomcat 組態中設定為領域。會為下列三個領域中的任何一個建立身分驗證：
  - JDBC Database Realm 會在 JDBC 驅動程式存取的關聯式資料庫中查詢使用者。
  - DataSource Database Realm 會在 JNDI 存取的資料庫中查詢使用者。
  - JNDI Directory Realm 會在由 JNDI 供應商存取的輕量型目錄存取通訊協定 (LDAP) 目錄中查詢使用者。查詢需要：
    - LDAP 連線詳細資訊：使用者搜尋基礎、搜尋篩選條件、角色基礎、角色篩選條件
    - 金鑰 JNDI Directory Realm：連線至 LDAP、驗證使用者，以及擷取使用者為成員的所有群組
- 授權：如果容器具有檢查 web.xml 中授權限制的角色型授權，則必須定義 Web 資源，並與限制中定義的角色進行比較。如果 LDAP 沒有群組角色映射，您必須在 web.xml 中設定屬性 <security-role-ref>，以實現群組角色映射。若要查看組態文件的範例，請參閱 Oracle 文件。
- 資料庫連線：使用 Amazon Relational Database Service (Amazon RDS) 端點 URL 和連線詳細資訊，在 Apache Tomcat 中建立資源定義。使用 JNDI 查詢更新應用程式碼以參考 DataSource。在 WebSphere 中定義的現有資料庫連線無法運作，因為它使用 WebSphere 的 JNDI 名稱。您可以使用 JNDI 名稱和 DataSource 類型定義，在 web.xml 中新增 <resource-ref> 項目。若要查看範例組態文件，請參閱 Apache Tomcat 文件。
- 記錄：根據預設，Apache Tomcat 會記錄到主控台或日誌檔案。您可以更新 logging.properties 來啟用領域層級追蹤（請參閱 Tomcat 中的記錄）。如果您使用 Apache Log4j 將日誌附加至檔案，則必須下載 tomcat-juli 並將其新增至 classpath。
- 工作階段管理：如果您為應用程式負載平衡和工作階段管理保留 IBM WebSEAL，則不需要變更。如果您使用 Application Load Balancer 或 AWS 上的 Network Load Balancer 取代 IBM WebSEAL 元件，則必須使用 Amazon ElastiCache 執行個體搭配 Memcached 叢集來設定工作階段管理，並將 Apache Tomcat 設定為使用開放原始碼工作階段管理。
- 如果您使用 IBM WebSEAL 轉送代理，您必須在 AWS 上設定新的 Network Load Balancer。使用 Network Load Balancer for WebSEAL 連接組態提供的 IPs。
- SSL 組態：我們建議您使用 Secure Sockets Layer (SSL) end-to-end通訊。若要在 Apache Tomcat 中設定 SSL 伺服器組態，請遵循 Apache Tomcat 文件中的指示。

Architecture

來源技術堆疊

- IBM WebSphere 應用程式伺服器

目標技術堆疊

- 架構使用 Elastic Load Balancing （第 2 版)。如果您使用 IBM WebSEAL 來識別管理和負載平衡，您可以在 AWS 上選取要與 IBM WebSEAL 反向代理整合的 Network Load Balancer。
- Java 應用程式會部署到 Apache Tomcat 應用程式伺服器，該伺服器在 Amazon EC2 Auto Scaling 群組中的 EC2 執行個體上執行。 Amazon EC2 Auto Scaling 您可以根據 Amazon CloudWatch 指標設定擴展政策，例如 CPU 使用率。
- 如果您要淘汰使用 IBM WebSEAL 進行負載平衡，您可以使用 Amazon ElastiCache for Memcached 進行工作階段管理。
- 對於後端資料庫，您可以為 Amazon RDS 部署高可用性 （多可用區），然後選取資料庫引擎類型。

目標架構

工具

- AWS CloudFormation
- AWS 命令列界面 (AWS CLI)
- Apache Tomcat (7.x 或 8.x 版）
- RHEL 7 或 Centos 7
- Amazon RDS 異地同步備份部署
- Amazon ElastiCache for Memcached （選用）

史詩

設定 VPC

任務	Description	所需的技能
建立 Virtual Private Cloud (VPC)		

任務	Description	所需的技能
建立子網路。		
視需要建立路由表。		
建立網路存取控制清單 ACLs)。		
設定 AWS Direct Connect 或企業 VPN 連線。		

複寫應用程式

任務	Description	所需的技能
重構應用程式建置 Maven 組態以產生 WAR 成品。		
重構 Apache Tomcat 中的應用程式相依性資料來源。		
重構應用程式原始碼，以在 Apache Tomcat 中使用 JNDI 名稱。		
將 WAR 成品部署至 Apache Tomcat。		
完成應用程式驗證和測試。		

設定網路

任務	Description	所需的技能
設定公司防火牆以允許連線至相依性服務。		

任務	Description	所需的技能
設定公司防火牆以允許最終使用者存取 AWS 上的 Elastic Load Balancing。		

建立應用程式基礎設施

任務	Description	所需的技能
在 EC2 執行個體上建立和部署應用程式。		
建立 Amazon ElastiCache for Memcached 叢集以進行工作階段管理。		
為後端資料庫建立 Amazon RDS Multi-AZ 執行個體。		
建立 SSL 憑證並將其匯入 AWS Certificate Manager (ACM)。		
在負載平衡器上安裝 SSL 憑證。		
安裝 Apache Tomcat 伺服器的 SSL 憑證。		
完成應用程式驗證和測試。		

剪下

任務	Description	所需的技能
關閉現有的基礎設施。		

任務	Description	所需的技能
將資料庫從生產還原至 Amazon RDS。		
透過 DNS 變更來切斷應用程式。		

相關資源

參考

- Apache Tomcat 7.0 文件
- Apache Tomcat 7.0 安裝指南
- Apache Tomcat JNDI 文件
- Amazon RDS 異地同步備份部署
- Amazon ElastiCache for Memcached

教學課程和影片

- Amazon RDS 入門

將 .NET 應用程式從 Microsoft Azure App Service 遷移至 AWS Elastic Beanstalk

Raghavender Madamshitti，Amazon Web Services

總結

此模式說明如何將 Microsoft Azure App Service 上託管的 .NET Web 應用程式遷移至 AWS Elastic Beanstalk。有兩種方式可將應用程式遷移至 Elastic Beanstalk：

- 使用 AWS Toolkit for Visual Studio - 此 Microsoft Visual Studio IDE 的外掛程式提供最簡單且最直接的方式，將自訂 .NET 應用程式部署到 AWS。您可以使用此方法將 .NET 程式碼直接部署到 AWS，並直接從 Visual Studio 建立支援資源，例如 SQL Server 資料庫的 Amazon Relational Database Service (Amazon RDS)。
- 上傳和部署至 Elastic Beanstalk - 每個 Azure App Service 都包含名為 Kudu 的背景服務，可用於擷取記憶體傾印和部署日誌、檢視組態參數，以及存取部署套件。您可以使用 Kudu 主控台存取 Azure App Service 內容、擷取部署套件，然後使用 Elastic Beanstalk 主控台中的上傳和部署選項，將套件上傳至 Elastic Beanstalk。

此模式說明第二種方法（透過 Kudu 將您的應用程式上傳至 Elastic Beanstalk)。模式也使用下列 AWS 服務：AWS Elastic Beanstalk、Amazon Virtual Private Cloud (Amazon VPC)、Amazon CloudWatch、Amazon Elastic Compute Cloud (Amazon EC2) Auto Scaling、Amazon Simple Storage Service (Amazon S3) 和 Amazon Route 53。

.NET Web 應用程式會部署到 AWS Elastic Beanstalk，AWS Elastic Beanstalk 會在 Amazon EC2 Auto Scaling 群組中執行。您可以根據 Amazon CloudWatch 指標設定擴展政策，例如 CPU 使用率。對於資料庫，您可以根據您的應用程式和業務需求，在多可用區域環境或 Amazon DynamoDB 中使用 Amazon RDS。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在 Azure App Service 中執行的 .NET Web 應用程式
- 使用 Azure App Service Kudu 主控台的許可

產品版本

- .NET Core (x64) 1.0.1、2.0.0 或更新版本，或 .NET Framework 4.x、3.5（請參閱 Windows Server 平台歷史記錄上的 .NET)
- 在 Windows Server 2012 或更新版本上執行的網際網路資訊服務 (IIS) 8.0 版或更新版本
- .NET 2.0 或 4.0 執行期。

Architecture

來源技術堆疊

- 使用 .NET Framework 3.5 或更新版本或 .NET Core 1.0.1、2.0.0 或更新版本開發，並在 Azure App Service (Web 應用程式或 API 應用程式） 上託管的應用程式

目標技術堆疊

- 在 Amazon EC2 Auto Scaling 群組中執行的 AWS Elastic Beanstalk

遷移架構

部署工作流程

工具

工具

- .NET Core 或 .NET Framework
- C#
- IIS
- Kudu 主控台

AWS 服務和功能

- AWS Elastic Beanstalk – Elastic Beanstalk 是一種easy-to-use用於部署和擴展 .NET Web 應用程式。Elastic Beanstalk 會自動管理容量佈建、負載平衡和自動擴展。
- Amazon EC2 Auto Scaling 群組 – Elastic Beanstalk 包含管理環境中 Amazon EC2 執行個體的 Auto Scaling 群組。在單一執行個體環境中，Auto Scaling 群組可確保隨時都有一個執行個體正在

執行。在負載平衡的環境中，您可以使用要執行的執行個體範圍來設定群組，而 Amazon EC2 Auto Scaling 會根據負載視需要新增或移除執行個體。

- Elastic Load Balancing – 當您在 AWS Elastic Beanstalk 中啟用負載平衡時，它會建立負載平衡器，在環境中的 EC2 執行個體之間分配流量。
- Amazon CloudWatch – Elastic Beanstalk 會自動使用 Amazon CloudWatch 來提供應用程式和環境資源的相關資訊。Amazon CloudWatch 支援標準指標、自訂指標和警示。
- Amazon Route 53 – Amazon Route 53 是高可用性且可擴展的雲端網域名稱系統 (DNS) Web 服務。您可以使用 Route 53 別名記錄，將自訂網域名稱映射至 AWS Elastic Beanstalk 環境。

史詩

設定 VPC

任務	Description	所需的技能
設定虛擬私有雲端 (VPC)。	在您的 AWS 帳戶中，使用必要資訊建立 VPC。	系統管理員
建立子網路。	在 VPC 中建立兩個或多個子網路。	系統管理員
建立路由表。	根據您的需求建立路由表。	系統管理員

設定 Elastic Beanstalk

任務	Description	所需的技能
存取 Azure App Service Kudu 主控台。	導覽至 App Service 儀表板，然後選擇進階工具 Go，透過 Azure 入口網站存取 Kudu。或者，您可以修改 Azure App Service URL，如下所示：`https://<appservicename>.scm.azurewebsites.net` 。	應用程式開發人員、系統管理員

任務	Description	所需的技能
從 Kudu 下載部署套件。	選擇 DebugConsole 選項，導覽至 Windows PowerShell。這會開啟 Kudo 主控台。前往 `wwwroot` 資料夾並下載。這會將 Azure App Service 部署套件下載為 zip 檔案。如需範例，請參閱附件。	應用程式開發人員、系統管理員
為 Elastic Beanstalk 建立套件。	解壓縮您從 Azure App Service 下載的部署套件。建立名為的 JSON 檔案 `aws-windows-deployment-manifest.json` （只有 .NET Core 應用程式需要此檔案）。建立包含 `aws-windows-deployment-manifest.json` 和 Azure App Service 部署套件檔案的 zip 檔案。如需範例，請參閱附件。	應用程式開發人員、系統管理員
建立新的 Elastic Beanstalk 應用程式。	開啟 Elastic Beanstalk 主控台。選擇現有的應用程式或建立新的應用程式。	應用程式開發人員、系統管理員
建立環境。	在 Elastic Beanstalk 主控台動作功能表中，選擇建立環境。選取 Web 伺服器環境和 .NET/IIS 平台。針對應用程式碼，選擇上傳。上傳您為 Elastic Beanstalk 準備的 zip 檔案，然後選擇建立環境。	應用程式開發人員、系統管理員

任務	Description	所需的技能
設定 Amazon CloudWatch。	預設會啟用基本 CloudWatch 監控。如果您想要變更組態，請在 Elastic Beanstalk 精靈中選擇已發佈的應用程式，然後選擇監控。	系統管理員
確認部署套件位於 Amazon S3 中。	建立應用程式環境後，您可以在 S3 儲存貯體中找到部署套件。	應用程式開發人員、系統管理員
測試應用程式。	建立環境後，請使用 Elastic Beanstalk 主控台中提供的 URL 來測試應用程式。	系統管理員

相關資源

- AWS Elastic Beanstack 概念 (Elastic Beanstalk 文件）
- Elastic Beanstalk 上的 .NET 入門 (Elastic Beanstalk 文件）
- Kudu 主控台 (GitHub)
- 使用「Kudu」管理 Azure Web 應用程式 (GS Lab 文章）
- 自訂 ASP.NET Core Elastic Beanstalk 部署 (AWS Toolkit for Visual Studio 使用者指南）
- Elastic Load Balancing 文件
- AWS Elastic Beanstalk 支援的平台 (Elastic Beanstalk 文件）
- 將 Web 應用程式部署至 AWS (C# 轉角文章）
- 擴展 Auto Scaling 群組的大小 (Amazon EC2 文件）
- Amazon RDS 的高可用性（多可用區域） (Amazon RDS 文件）

其他資訊

備註

- 如果您要將內部部署或 Azure SQL Server 資料庫遷移至 Amazon RDS，您也必須更新資料庫連線詳細資訊。

- 為了測試目的，會連接範例示範應用程式。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

在 Amazon ECS 上從 Oracle WebLogic 遷移至 Apache Tomcat (TomEE)

Anya Epishcheva 和 Harshad Gohil，Amazon Web Services

總結

此模式討論透過 Amazon Elastic Container Service (Amazon ECS) 將執行 Oracle WebLogic 的現場部署 Oracle Solaris SPARC 系統遷移至執行 Apache TomEE（新增容器支援的 Apache Tomcat) 的 Docker 容器型安裝的步驟。

如需有關將與您從 Oracle WebLogic 遷移至 Tomcat 之應用程式相關聯的資料庫遷移的資訊，請參閱此目錄中的資料庫遷移模式。

最佳實務

遷移 Java 和 Java Enterprise Edition (Java EE) Web 應用程式的步驟會有所不同，取決於應用程式所使用的容器特定資源數量。以 Spring 為基礎的應用程式通常更容易遷移，因為它們在部署容器上有少量的相依性。相反地，使用企業 JavaBeans (EJBs) 和受管容器資源的 Java EE 應用程式，例如執行緒集區、Java 身分驗證和授權服務 (JAAS) 和容器受管持久性 (CMP)，需要更多努力。

為 Oracle Application Server 開發的應用程式經常使用 Oracle Identity Management 套件。遷移至開放原始碼應用程式伺服器的客戶經常選擇使用以 SAML 為基礎的聯合重新實作身分和存取管理。當從 Oracle Identity Management 套件遷移不是選項時，其他 會使用 Oracle HTTP Server Webgate。

Java 和 Java EE Web 應用程式非常適合部署在以 Docker 為基礎的 AWS 服務上，例如 AWS Fargate 和 Amazon ECS。客戶經常選擇預先安裝最新版本的目標應用程式伺服器（例如 TomEE) 和 Java 開發套件 (JDK) 的 Docker 映像。他們會在基礎 Docker 映像檔上安裝應用程式，將其發佈到 Amazon Elastic Container Registry (Amazon ECR) 登錄檔中，並使用它在 AWS Fargate 或 Amazon ECS 上可擴展地部署其應用程式。

理想情況下，應用程式部署是彈性的；也就是說，應用程式執行個體的數量會根據流量或工作負載縮減或縮減。這表示應用程式執行個體需要上線或終止，才能根據需求調整容量。

將 Java 應用程式移至 AWS 時，請考慮使其無狀態。這是 AWS Well-Architected Framework 的關鍵架構原則，將使用容器化啟用水平擴展。例如，大多數以 Java 為基礎的 Web 應用程式會在本機存放使用者工作階段資訊。為了避免應用程式執行個體因 Amazon Elastic Compute Cloud (Amazon EC2) 中的自動擴展或其他原因而終止，使用者工作階段資訊應全域儲存，以便 Web 應用程式使用者可以繼續無縫且透明地工作，而無需重新連線或重新登入 Web 應用程式。此方法有數種架構選項，包括 Amazon ElastiCache for Redis，或在全域資料庫中儲存工作階段狀態。TomEE 等應用程式伺服器具有外掛程式，可透過 Redis、資料庫和其他全域資料存放區啟用工作階段儲存和管理。

使用與 Amazon CloudWatch 和 AWS X-Ray 輕鬆整合的常見集中式記錄和偵錯工具。遷移提供改善應用程式生命週期功能的機會。例如，您可能想要自動化建置程序，以便使用持續整合和持續交付 (CI/CD) 管道輕鬆進行變更。這可能需要變更應用程式，以便在不停機的情況下進行部署。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 來源 Java 程式碼和 JDK
- 使用 Oracle WebLogic 建置的來源應用程式
- 定義身分和存取管理的解決方案 (SAML 或 Oracle Webgate)
- 定義應用程式工作階段管理的解決方案 (like-for-like或搭配 Amazon ElastiCache 移動，或視需要使應用程式無狀態）
- 了解團隊是否需要重構 J2EE-specific程式庫，以移轉至 Apache TomEE （請參閱 Apache 網站上的 Java EE 7 實作狀態)
- 根據您的安全需求強化 TomEE 映像
- 預先安裝目標 TomEE 的容器映像
- 視需要同意並實作應用程式修復 （例如，記錄偵錯建置、身分驗證）

產品版本

- Oracle WebLogic OC4J、9i、10g
- Tomcat 7 （使用 Java 1.6 或更新版本）

Architecture

來源技術堆疊

- 使用 Oracle WebLogic 建置的 Web 應用程式
- 使用 Oracle Webgate 或 SAML 身分驗證的 Web 應用程式
- 連接至 Oracle Database 10g 版及更新版本的 Web 應用程式

目標技術堆疊

- 在 Amazon ECS 上執行的 TomEE(Apache Tomcat 與新增的容器支援）（另請參閱在 Amazon ECS 上部署 Java Web 應用程式和 Java Microservices)https://aws.amazon.com/blogs/compute/deploying-java-microservices-on-amazon-ec2-container-service/
- 適用於 Oracle 的 Amazon Relational Database Service (Amazon RDS)；適用於 Amazon RDS 支援的 Oracle 版本，請參閱適用於 Oracle 的 Amazon RDS

目標架構

工具

若要在 TomEE 上操作，Java 應用程式必須重建為 .war 檔案。在某些情況下，在 TomEE 上操作應用程式可能需要變更應用程式；您應該檢查以確保正確定義必要的組態選項和環境屬性。

此外，應該正確定義 Java 命名和目錄界面 (JNDI) 查詢和 JavaServer 頁面 (JSP) 命名空間。請考慮檢查應用程式使用的檔案名稱，以避免命名與內建 T 程式庫的衝突。例如，solveence.xml 是 Apache OpenJPA 架構（與 TomEE 中的 OpenEJB 綁定）用於組態用途的檔案名稱。PUI 中的 persistence.xml 檔案包含 Spring 架構 Bean 宣告。

TomEE 版本 7.0.3 和更新版本 (Tomcat 8.5.7 和更新版本）會針對具有特殊字元的原始（未編碼）URLs 傳回 HTTP 400 回應（錯誤請求）。伺服器回應會顯示為最終使用者的空白頁面。TomEE 和 Tomcat 的早期版本允許在 URLs中使用某些未編碼的特殊字元；不過，它被視為不安全，如CVE-2016-6816 網站所述。若要解決 URL 編碼問題，直接透過 JavaScript 傳遞至瀏覽器URLs 必須使用 encodeURI() 方法編碼，而不是用作原始字串。

在 TomEE 中部署 .war 檔案後，請監控是否有任何遺失的共用程式庫和 Oracle 特定延伸模組，以從 Tomcat 程式庫新增缺少的元件。

一般程序

- 在 TomEE 上設定應用程式。
- 識別並重新設定從來源到目標格式的應用程式伺服器特定組態檔案和資源。
- 識別並重新設定 JNDI 資源。
- 將 EJB 命名空間和查閱調整為目標應用程式伺服器所需的格式（如適用）。
- 重新設定 JAAS 應用程式容器特定的安全角色和原則映射（如適用）。
- 將應用程式和共用程式庫封裝到 .war 檔案中。

- 使用提供的 Docker 容器在 TomEE 中部署 .war 檔案。
- 監控啟動日誌以識別任何遺失的共用程式庫和部署描述項延伸。如果找到任何 ，請返回第一個任務。
- 根據還原的 Amazon RDS 資料庫測試已安裝的應用程式。
- 遵循 Deploy Docker Containers 中的指示，以負載平衡器和 Amazon ECS 叢集啟動完整的架構。
- 更新 URLs以指向負載平衡器。
- 更新組態管理資料庫 (CMDB)。

史詩

規劃遷移

任務	Description	所需的技能
執行應用程式探索（目前狀態足跡和效能基準）。		BA，遷移主管
驗證來源和目標資料庫版本和引擎。		DBA
驗證來源和目標應用程式設計（身分和工作階段管理）。		DBA、遷移工程師、應用程式擁有者
識別目標伺服器執行個體的硬體和儲存需求。		DBA、SysAdmin
根據容量、儲存功能和網路功能選擇適當的執行個體類型。		DBA、SysAdmin
識別來源和目標資料庫的網路存取安全需求。		DBA、SysAdmin
識別應用程式遷移策略和工具。		DBA，遷移主管
完成應用程式的遷移設計和遷移指南。		組建主管、遷移主管

任務	Description	所需的技能
完成應用程式遷移 Runbook。		組建主管、切換主管、測試主管、遷移主管

設定基礎設施

任務	Description	所需的技能
建立 Virtual Private Cloud (VPC)		SysAdmin
建立安全群組。		SysAdmin
設定和啟動 Amazon RDS 資料庫執行個體。		DBA、SysAdmin
設定 Amazon ECS 部署。		SysAdmin
將您的應用程式封裝為 Docker 映像。		SysAdmin
將映像推送至 Amazon ECR 登錄檔 （或略過此步驟並將其推送至 Amazon ECS 叢集）。		SysAdmin
設定應用程式和 Amazon ECS 服務選項的任務定義。		SysAdmin
設定叢集、檢閱安全設定，以及設定 AWS Identity and Access Management (IAM) 角色。		SysAdmin
啟動您的設定，並根據您的應用程式遷移 Runbook 執行測試。		SysAdmin

遷移資料

任務	Description	所需的技能
取得您的安全保證團隊將生產資料移至 AWS 的許可。		DBA、遷移工程師、應用程式擁有者
建立並取得端點的存取權，以擷取資料庫備份檔案。		DBA
使用原生資料庫引擎或第三方工具來遷移資料庫物件和資料。		DBA
從應用程式遷移 Runbook 執行必要的測試，以確認資料遷移成功。		DBA、遷移工程師、應用程式擁有者

遷移應用程式

任務	Description	所需的技能
建立遷移的變更請求 (CR)。		切換潛在客戶
取得遷移的 CR 核准。		切換潛在客戶
遵循應用程式遷移 Runbook 中的應用程式遷移策略。		DBA、遷移工程師、應用程式擁有者
升級應用程式（如果需要）。		DBA、遷移工程師、應用程式擁有者
完成功能、非功能、資料驗證、SLA 和效能測試。		測試主管、應用程式擁有者、應用程式使用者

剪下

任務	Description	所需的技能
從應用程式或企業擁有者取得簽署。		切換潛在客戶
執行資料表主題練習，逐步解說切換執行手冊的所有步驟。		DBA、遷移工程師、應用程式擁有者
將應用程式用戶端切換到新的基礎設施。		DBA、遷移工程師、應用程式擁有者

關閉專案

任務	Description	所需的技能
關閉臨時 AWS 資源。		DBA、遷移工程師、SysAdmin
檢閱並驗證專案文件。		遷移主管
收集遷移時間的指標、手動與工具的 %、節省成本等。		遷移主管
關閉專案並提供意見回饋。		遷移主管，應用程式擁有者

相關資源

參考

- Apache Tomcat 7.0 文件
- Apache Tomcat 7.0 安裝指南
- Apache Tomcat JNDI 文件
- Apache TomEE 文件
- Amazon RDS for Oracle
- Amazon RDS 定價

- Oracle 和 AWS
- Oracle on Amazon RDS 文件
- Amazon RDS 異地同步備份部署
- Amazon ECS 入門
- Amazon RDS 入門

教學課程和影片

- 在 Amazon RDS 上執行 Oracle 資料庫的最佳實務 (re：Invent 2018 簡報)

使用 AWS DMS 將 Oracle 資料庫從 Amazon EC2 遷移至 Amazon RDS for Oracle

Chethan Gangadharaiah 和 Brian motzer，Amazon Web Services

總結

此模式說明使用 AWS Database Migration Service (AWS DMS) 將 Amazon Elastic Compute Cloud (Amazon EC2) 上的 Oracle 資料庫遷移至 Amazon Relational Database Service (Amazon RDS) for Oracle 的步驟。模式也會使用 Oracle SQL Developer 或 SQL *Plus 連線到您的 Oracle 資料庫執行個體，並包含可自動化部分任務的 AWS CloudFormation 範本。

遷移至 Amazon RDS for Oracle 可讓您專注於業務和應用程式，同時 Amazon RDS 會處理資料庫管理任務，例如佈建資料庫、備份和復原、安全修補程式、版本升級和儲存體管理。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Amazon EC2 上 Oracle 資料庫的 Amazon Machine Image (AMI)

產品版本

- AWS DMS 支援適用於 Enterprise、Standard、Standard One 和 Standard Two 版本的 Amazon RDS 執行個體資料庫的 Oracle 11g 版 (11.2.0.3.v1 版和更新版本）、12c 和 18c 版。如需支援版本的最新資訊，請參閱 AWS 文件中的使用 Oracle 資料庫做為 AWS DMS 的目標。（連接的 AWS CloudFormation 範本使用 Oracle 12c 版做為來源資料庫。）
- Oracle SQL 開發人員 4.0.3

Architecture

來源架構

- Amazon EC2 上的 Oracle 資料庫

目標架構

- Amazon RDS for Oracle

遷移架構

工具

- AWS DMS – AWS Database Migration Service (AWS DMS) 可協助您快速安全地將資料庫遷移至 AWS。它同時支援同質和異質遷移。如需有關支援的 Oracle 資料庫版本和版本的資訊，請參閱 AWS 文件中的使用 Oracle 資料庫做為 AWS DMS 的來源和使用 Oracle 資料庫做為 AWS DMS 的目標。
- Oracle SQL Developer 或 SQL *Plus – 這些工具可讓您連線至 Amazon RDS for Oracle 資料庫執行個體。

史詩

設定您的目標資料庫

任務	Description	所需的技能
建立 Amazon RDS for Oracle 資料庫執行個體。	登入 AWS 管理主控台，開啟位於 https://console.aws.amazon.com/rds/ 的 Amazon RDS 主控台。透過選取 Oracle 資料庫的適當引擎、範本、資料庫憑證設定、執行個體類型、儲存體、異地同步備份設定、虛擬私有雲端 (VPC) 和組態、登入憑證和其他設定，來建立 Oracle 資料庫執行個體。如需說明，請檢視「相關資源」區段中的連結。或使用附件中的 AWS CloudFormation 範本 (Create_RDS.yaml) 來建立 Amazon RDS for Oracle 資料庫執行個體。	開發人員
連線至 Amazon RDS 並將權限授予 Oracle 使用者。	修改安全群組以開啟要從本機電腦和 AWS DMS 複寫執行個體連線的適當連接埠。當您設	開發人員

任務	Description	所需的技能
	定連線時，請確定已選取「可公開存取」選項，以便您可以從 VPC 外部連線至資料庫。透過 Oracle SQL Developer 或 SQL *Plus 連線至 Amazon RDS，方法是使用登入憑證、建立 AWS DMS 使用者，並提供必要的權限給 AWS DMS 使用者以修改資料庫。	

設定來源 EC2 執行個體的安全群組

任務	Description	所需的技能
檢查 Oracle 資料庫是否已啟動並執行。	使用 Secure Shell (SSH) 連線至 EC2 執行個體，並使用 SQL *Plus 嘗試連線至 Oracle 資料庫。	開發人員
修改安全群組。	修改 EC2 執行個體的安全群組以開啟適當的連接埠，以便您可以從本機電腦和 AWS DMS 複寫執行個體進行連線。	開發人員

設定 AWS DMS

任務	Description	所需的技能
建立 AWS DMS 複寫執行個體。	在 AWS DMS 中，在與 Amazon RDS for Oracle 資料庫執行個體相同的 VPC 中建立複寫執行個體。指定複寫執行個體的名稱和描述、選擇執行個體類別和複寫引擎版本（使	DBA

任務	Description	所需的技能
	用預設值）、選擇您在其中建立 Amazon RDS 資料庫執行個體的 VPC、視需要設定異地同步備份設定、配置儲存、指定可用區域，以及設定其他設定。或者，您可以使用附件中的 AWS CloudFormation 範本 (DMS.yaml) 來實作此步驟。	
連線至來源和目標資料庫端點。	指定端點識別符、引擎、伺服器、連接埠、登入憑證和額外的連線屬性，以建立來源和目標資料庫端點。對於來源伺服器，請使用託管 Oracle 資料庫之 EC2 執行個體的公有 DNS。對於目標伺服器，請使用 Amazon RDS for Oracle 的端點。執行測試以確認來源和目標連線是否正常運作。或者，您可以使用附件中的 AWS CloudFormation 範本 (DMS.yaml) 來實作此步驟。	DBA

任務	Description	所需的技能
建立 AWS DMS 任務。	建立 AWS DMS 任務，將資料從來源端點遷移到目標端點，設定來源和目的地端點之間的複寫，或同時設定兩者。建立 AWS DMS 任務時，請指定複寫執行個體、來源端點、目標端點、遷移類型（僅限資料、僅限複寫或兩者）、資料表映射和篩選條件。執行 AWS DMS 任務、監控任務、檢查資料表統計資料，以及檢查 Amazon CloudWatch 中的日誌。或者，您可以使用附件中的 AWS CloudFormation 範本 (DMS.yaml) 來實作此步驟。	DBA

相關資源

- 建立 Amazon RDS 資料庫執行個體
- 連接至執行 Oracle 資料庫引擎的資料庫執行個體
- AWS DMS 文件
- AWS DMS Step-by-Step演練
- 將 Oracle 資料庫遷移至 AWS 雲端

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Logstash 將內部部署 Oracle 資料庫遷移至 Amazon OpenSearch Service

Aditya Goteti，Amazon Web Services

總結

此模式說明如何使用 Logstash 將資料從現場部署 Oracle 資料庫移至 Amazon OpenSearch Service。它包含架構考量，以及一些必要的技能集和建議。資料可以來自單一資料表，也可以來自需要執行全文搜尋的多個資料表。

OpenSearch Service 可以在虛擬私有雲端 (VPC) 中設定，也可以使用 IP 型限制公開放置。此模式說明在 VPC 中設定 OpenSearch Service 的情況。Logstash 用於從 Oracle 資料庫收集資料、將其剖析為 JSON 格式，然後將資料饋送至 OpenSearch Service。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Java 8 (Logstash 6.4.3 必要）
- 內部部署資料庫伺服器與 VPC 中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體之間的連線，使用 AWS Virtual Private Network (AWS VPN) 建立
- 從資料庫擷取要推送至 OpenSearch Service 所需資料的查詢
- Oracle Java Database Connectivity (JDBC) 驅動程式

限制

- Logstash 無法識別從資料庫硬刪除的記錄

產品版本

- Oracle 資料庫 12c
- OpenSearch Service 6.3
- Logstash 6.4.3

Architecture

來源技術堆疊

- 內部部署 Oracle 資料庫
- 內部部署 AWS VPN

目標技術堆疊

- VPC
- EC2 執行個體
- OpenSearch Service
- Logstash
- NAT Gateway（用於 EC2 執行個體上的作業系統更新，以及安裝 Java 8、Logstash 和外掛程式）

資料遷移架構

工具

- Logstash 6.4.3
- JDBC 輸入外掛程式 (下載和更多資訊)
- Logstash 輸出外掛程式 (logstash-output-amazon_es)
- Oracle JDBC 驅動程式

史詩

規劃遷移

任務	Description	所需的技能
識別來源資料的大小。	來源資料的大小是您用來判斷索引中要設定之碎片數量的參數之一。	DBA，資料庫開發人員
分析每個資料欄的資料類型和對應的資料。	在文件中找到先前看不到的欄位時，OpenSearch Service 會動態映射資料類型。如果有任何特定資料類型或格式（例如日期欄位）需要明確宣告，請	應用程式擁有者、開發人員、資料庫開發人員

任務	Description	所需的技能
	在建立索引期間識別欄位並定義這些欄位的映射。	
判斷是否有任何具有主索引鍵或唯一索引鍵的資料欄。	若要避免在更新或插入期間重複 Amazon OpenSearch Service 中的記錄，您需要在 `amazon_es` 外掛程式的輸出區段中設定 `document_id` 設定（例如，`document_id => "%{customer_id}"` `customer_id` 是主索引鍵）。	應用程式擁有者、開發人員
分析新增記錄的數量和頻率；檢查記錄的刪除頻率。	需要此任務才能了解來源資料的成長速率。如果資料密集讀取且插入很少見，您可以擁有單一索引。如果頻繁插入新記錄且未刪除，碎片大小可以輕鬆超過建議的 50 GB 大小上限。在這種情況下，您可以透過在 Logstash 和程式碼中設定索引模式來動態建立索引，您可以在其中使用別名來存取索引。	應用程式擁有者、開發人員
決定需要多少個複本。		應用程式擁有者、開發人員
決定要在索引上設定的碎片數量。		應用程式擁有者、開發人員
識別專用主節點、資料節點和 EC2 執行個體的執行個體類型。	如需詳細資訊，請參閱相關資源一節。	應用程式擁有者、開發人員
判斷所需的專用主節點和資料節點數量。	如需詳細資訊，請參閱相關資源一節。	

遷移資料

任務	Description	所需的技能
啟動 EC2 執行個體。	在連接 AWS VPN 的 VPC 中啟動 EC2 執行個體。	Amazon VPC 建構、AWS VPN
在 EC2 執行個體上安裝 Logstash。		開發人員
安裝 Logstash 外掛程式。	安裝必要的 Logstash 外掛程式 `jdbc-input` 和 `logstash-output-amazon_es` 。	開發人員
設定 Logstash。	建立 Logstash 金鑰存放區以存放敏感資訊，例如 AWS Secrets Manager 金鑰和資料庫登入資料，然後將參考放在 Logstash 組態檔案中。	開發人員
設定無效字母佇列和持久性佇列。	根據預設，當 Logstash 遇到因為資料包含映射錯誤或其他問題而無法處理的事件時，Logstash 管道會停止或捨棄失敗的事件。為了避免在這種情況下遺失資料，您可以設定 Logstash 將失敗的事件寫入無效字母佇列，而不是捨棄它們。為了避免在異常終止期間遺失資料，Logstash 具有持久性佇列功能，可將訊息佇列存放在磁碟上。持久性佇列在 Logstash 中提供資料耐久性。	開發人員
建立 Amazon OpenSearch Service 網域。	使用不需要使用 AWS Identity and Access Management (IAM) 憑證簽署請求的存取政	開發人員

任務	Description	所需的技能
	策來建立 Amazon OpenSearch Service 網域。Amazon OpenSearch Service 網域必須在相同的 VPC 內建立。您也應該選取執行個體類型，並根據分析設定專用節點和主節點的數量。	
設定所需的 Amazon OpenSearch Service 日誌。	如需詳細資訊，請參閱 OpenSearch Service 文件。	
建立索引。		開發人員
啟動 Logstash。	執行 Logstash 做為背景服務。Logstash 會執行設定的 SQL 查詢、提取資料、將其轉換為 JSON 格式，並將其饋送至 OpenSearch Service。對於初始載入，請勿在 Logstash 組態檔案中設定排程器。	開發人員

任務	Description	所需的技能
檢查文件。	檢查索引上的文件數量，以及來源資料庫中是否存在所有文件。在初始載入期間，它們會新增至索引，並用來停止 Logstash。 變更 Logstash 組態，根據用戶端需求新增以固定間隔執行的排程器，然後重新啟動 Logstash。Logstash 只會挑選上次執行後更新或新增的記錄，而上次執行時間戳記會存放在以 `last_run_metadata_path => "/usr/share/logstash/.logstash_jdbc_last_run"` Logstash 組態檔案中 屬性設定的檔案中。	開發人員

相關資源

- 建議的 CloudWatch 警示
- 專用 Amazon OpenSearch Service 主節點
- 調整 Amazon OpenSearch Service 網域的大小
- Logstash 文件
- JDBC 輸入外掛程式
- Logstash 輸出外掛程式
- Amazon OpenSearch Service 網站

將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle

Baji Shaik 和 Pavan Pusuluri，Amazon Web Services

總結

此模式說明將內部部署 Oracle 資料庫遷移至 Amazon Relational Database Service (Amazon RDS) for Oracle 的步驟。作為遷移程序的一部分，您可以建立遷移計劃，並根據來源資料庫考慮有關目標資料庫基礎設施的重要因素。您可以根據您的業務需求和使用案例，選擇兩個遷移選項之一：

- AWS Database Migration Service (AWS DMS) – 您可以使用 AWS DMS 快速且安全地將資料庫遷移至 AWS 雲端。您的來源資料庫在遷移期間保持完全運作，將依賴資料庫的應用程式停機時間降到最低。您可以使用 AWS DMS 來建立任務，在您透過稱為變更資料擷取 (CDC) 的程序完成初始完全載入遷移後擷取持續變更。
- 原生 Oracle 工具 – 您可以使用原生 Oracle 工具來遷移資料庫，例如 Oracle、Data Pump Export 和 Data Pump Import with Oracle GoldenGate for CDC。您也可以使用原生 Oracle 工具，例如原始匯出公用程式和原始匯入公用程式，以減少完整載入時間。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署 Oracle 資料庫
- Amazon RDS Oracle 資料庫 (DB) 執行個體

限制

- 資料庫大小限制：64 TB

產品版本

- Oracle 版本 11g （版本 11.2.0.3.v1 和更新版本） 以及最多 12.2 和 18c。如需支援版本的最新清單，請參閱 AWS 文件中的Amazon RDS for Oracle。對於 AWS DMS 支援的 Oracle 版本，請參閱 AWS DMS 文件中的使用 Oracle 資料庫做為 AWS DMS 的來源。

Architecture

來源技術堆疊

- 內部部署 Oracle 資料庫

目標技術堆疊

- Amazon RDS for Oracle

來源和目標架構

下圖顯示如何使用 AWS DMS 將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle。

該圖顯示以下工作流程：

1. 建立或使用現有的資料庫使用者，將必要的 AWS DMS 許可授予該使用者，開啟 ARCHIVELOG 模式，然後設定補充記錄。
2. 設定內部部署和 AWS 網路之間的網際網路閘道。
3. 設定 AWS DMS 的來源和目標端點。
4. 設定 AWS DMS 複寫任務，將資料從來源資料庫遷移至目標資料庫。
5. 完成目標資料庫上的遷移後活動。

下圖顯示如何使用原生 Oracle 工具將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle。

該圖顯示以下工作流程：

1. 建立或使用現有的資料庫使用者，並使用 Oracle Export (`exp`) 和 Import (`imp`) 公用程式授予備份 Oracle 資料庫所需的許可。
2. 設定內部部署和 AWS 網路之間的網際網路閘道。
3. 在堡壘主機上設定 Oracle 用戶端以取得備份資料庫。
4. 將備份資料庫上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體。
5. 將資料庫備份從 Amazon S3 還原至 Amazon RDS for Oracle 資料庫。
6. 為 CDC 設定 Oracle GoldenGate。
7. 完成目標資料庫上的遷移後活動。

工具

- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署設定的組合之間遷移。
- 原生 Oracle 工具可協助您執行同質遷移。您可以使用 Oracle Data Pump 在來源和目標資料庫之間遷移資料。此模式使用 Oracle Data Pump 執行從來源資料庫到目標資料庫的完整載入。
- Oracle GoldenGate 可協助您在兩個或多個資料庫之間執行邏輯複寫。此模式使用 GoldenGate，透過 Oracle Data Pump 在初始載入後複寫差異變更。

史詩

規劃遷移

任務	Description	所需的技能
建立專案文件並記錄資料庫詳細資訊。	1. 記錄您的遷移目標、遷移需求、關鍵專案利益相關者、專案里程碑、專案截止日期、關鍵指標、遷移風險和風險緩解計劃。 2. 記錄來源資料庫的重要資訊，包括 RAM、IOPS 和 CPUs。您稍後將使用此資訊來判斷適當的目標資料庫執行個體。 3. 驗證來源和目標資料庫的版本。	DBA
識別儲存需求。	識別並記錄您的儲存需求，包括下列項目： 1. 計算為來源資料庫執行個體配置的儲存體。 2. 從來源資料庫執行個體收集歷史成長指標。 3. 預測目標資料庫執行個體的未來成長。	DBA、SysAdmin

任務	Description	所需的技能
	Note 對於一般用途 (gp2) SSD 磁碟區，每 1 GB 的儲存體可獲得三個 IOPS。透過計算來源資料庫上的讀取和寫入 IOPS 總數來分配儲存體。	
根據運算需求選擇適當的執行個體類型。	1. 判斷目標資料庫執行個體的運算需求。 2. 識別效能問題。 3. 考慮決定適當執行個體類型的因素： • 來源資料庫執行個體的 CPU 使用率 • 來源資料庫執行個體的 IOPS（讀取和寫入） • 來源資料庫執行個體上的記憶體使用量	SysAdmin
識別網路存取安全需求。	1. 識別並記錄來源和目標資料庫的網路存取安全需求。 2. 設定適當的安全群組，讓應用程式能夠與資料庫通訊。	DBA、SysAdmin
識別應用程式遷移策略。	1. 決定並記錄遷移切換策略。 2. 決定並記錄應用程式的復原時間目標 (RTO) 和復原點目標 (RPO)，然後相應地規劃切換。	DBA、SysAdmin、應用程式擁有者

任務	Description	所需的技能
識別遷移風險。	評估資料庫並記錄遷移的特定風險和緩解措施。例如： • 識別無記錄資料表，並強調復原時資料遺失的風險。 • 擷取來源資料庫使用者和權限，並反白顯示與 Amazon RDS 權限的衝突。 • 檢閱提醒日誌是否有任何 Oracle 特定的錯誤和警告。 • 識別目標資料庫執行個體的支援和不支援的功能。 • 檢閱目標資料庫版本引擎的已棄用功能。	DBA

設定基礎設施

任務	Description	所需的技能
建立 VPC。	為目標資料庫執行個體建立新的 Amazon Virtual Private Cloud (Amazon VPC)。	SysAdmin
建立安全群組。	在新的 VPC 中建立安全群組，以允許與資料庫執行個體的傳入連線。	SysAdmin
建立 Amazon RDS for Oracle 資料庫執行個體。	使用新的 VPC 和安全群組建立目標資料庫執行個體，然後啟動執行個體。	SysAdmin

選項 1 - 使用原生 Oracle 或第三方工具來遷移資料

任務	Description	所需的技能
準備來源資料庫。	1. 建立 Data Pump 目錄或使用現有的目錄。 2. 建立遷移使用者並授予執行 Data Pump 擷取的許可。 3. 從來源資料庫擷取角色、使用者和資料表空間做為 SQL 指令碼。 4. 將解壓縮的資料幫浦傾印轉移至目標資料庫執行個體`data pump`目錄。	DBA、SysAdmin
準備目標資料庫。	1. 確認目標 Amazon RDS for Oracle 資料庫執行個體上已安裝或啟用所有資料庫選項（例如文字和 Java)。 2. 建立 Data Pump 目錄或使用現有的目錄。 3. 建立遷移使用者並授予執行 Data Pump 匯入的許可。 4. 在目標資料庫執行個體上建立所需的資料表空間、使用者和角色。 5. 將傳輸的 Data Pump 匯出傾印匯入目標資料庫。 6. 建立匯入或物件建立期間排除的任何索引。 7. 建立匯入期間排除的任何限制條件。 8. 驗證或重新編譯無效的物件。 9. 重建無效的索引。	DBA、SysAdmin

任務	Description	所需的技能
	10驗證來源和目標資料庫之間的資料庫物件計數。 11解決物件計數之間發現的任何差異。	

選項 2 - 使用 AWS DMS 遷移資料

任務	Description	所需的技能
準備資料。	1. 清除來源資料庫中的資料。 2. 建立複寫執行個體。 3. 建立來源端點和目標端點。 4. 識別要遷移的資料表和物件數量。	DBA
遷移資料。	1. 刪除目標資料庫的外部索引鍵限制和觸發。 2. 在目標資料庫上捨棄次要索引。 3. 設定從來源資料庫到目標資料庫的 AWS DMS 完全載入任務設定。 4. 啟用外部金鑰。 5. 啟用 AWS DMS CDC 以複寫進行中的變更。 6. 啟用觸發。 7. 更新序列。 8. 驗證來源和目標資料。	DBA

切換到目標資料庫

任務	Description	所需的技能
將應用程式用戶端切換到新的基礎設施。	1. 停止指向 Oracle 的所有應用程式服務和用戶端連線。 2. 執行 AWS DMS 任務。 3. 設定復原任務（例如，將 CDC 從 Amazon RDS 資料庫反轉至內部部署 Oracle 資料庫）。 4. 驗證資料。 5. 透過將 Amazon Route 53 設定為新的 Amazon RDS for Oracle 資料庫執行個體，在新的目標資料庫上啟動應用程式服務。 6. 將 Amazon CloudWatch 監控新增至新的 Amazon RDS for Oracle 資料庫執行個體。	DBA、SysAdmin、應用程式擁有者
實作您的轉返計劃。	1. 停止指向 Amazon RDS for Oracle 資料庫執行個體的所有應用程式服務。 2. 使用 AWS DMS 任務，將變更轉返至來源現場部署 Oracle 資料庫。 3. 停止從現場部署 Oracle 資料庫執行到 Amazon RDS for Oracle 資料庫的 AWS DMS 任務。 4. 將應用程式設定回來源 Oracle 資料庫。 5. 確認復原部署已完成。	DBA、應用程式擁有者

關閉遷移專案

任務	Description	所需的技能
清除資源。	關閉或移除臨時 AWS 資源，例如 AWS DMS 複寫執行個體和 S3 儲存貯體。	DBA、SysAdmin
檢閱專案文件。	檢閱遷移規劃文件和目標，然後確認您已完成所有必要的遷移步驟。	DBA、SysAdmin、應用程式擁有者
收集指標。	記錄金鑰遷移指標，包括完成遷移所需的時間、手動與工具型任務的百分比、節省成本和其他相關指標。	DBA、SysAdmin、應用程式擁有者
關閉專案。	關閉遷移專案並擷取工作的相關意見回饋。	DBA、SysAdmin、應用程式擁有者

相關資源

參考

- 將 Oracle 資料庫遷移至 AWS 雲端 (AWS 規範指引)
- AWS Database Migration Service (AWS DMS 文件)
- Amazon RDS 定價 (Amazon RDS 文件)

教學課程和影片

- AWS Database Migration Service 入門 (AWS DMS 文件)
- Amazon RDS 資源 (Amazon RDS 文件)
- AWS Database Migration Service (DMS) (YouTube)

使用 Oracle Data Pump 將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle

Mohan Annam 和 Brian motzer，Amazon Web Services

總結

此模式說明如何使用 Oracle Data Pump，將 Oracle 資料庫從內部部署資料中心遷移至 Amazon Relational Database Service (Amazon RDS) for Oracle 資料庫執行個體。

模式涉及從來源資料庫建立資料傾印檔案、將檔案存放在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中，然後將資料還原至 Amazon RDS for Oracle 資料庫執行個體。當您使用 AWS Database Migration Service (AWS DMS) 進行遷移時，此模式非常有用。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在 AWS Identity and Access Management (IAM) 和 Amazon S3 分段上傳中建立角色所需的許可
- 從來源資料庫匯出資料所需的許可
- 安裝https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html並設定 AWS Command Line Interface (AWS CLI)

產品版本

- Oracle Data Pump 僅適用於 Oracle Database 10g 版本 1 (10.1) 和更新版本。

Architecture

來源技術堆疊

- 內部部署 Oracle 資料庫

目標技術堆疊

- Amazon RDS for Oracle
- SQL 用戶端 (Oracle SQL Developer)
- S3 儲存貯體

來源和目標架構

工具

AWS 服務

- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。在此模式中，IAM 用於建立將資料從 Amazon S3 遷移至 Amazon RDS for Oracle 所需的角色和政策。
- Amazon Relational Database Service (Amazon RDS) for Oracle 可協助您在 AWS 雲端中設定、操作和擴展 Oracle 關聯式資料庫。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他工具

- Oracle Data Pump 可協助您以高速將資料和中繼資料從一個資料庫移至另一個資料庫。在此模式中，Oracle Data Pump 會用來將資料傾印 (.dmp) 檔案匯出至 Oracle 伺服器，並將其匯入 Amazon RDS for Oracle。如需詳細資訊，請參閱《Amazon RDS 文件》中的在 Amazon RDS 上將資料匯入 Oracle。
- Oracle SQL Developer 是一種整合的開發環境，可簡化傳統和雲端部署中 Oracle 資料庫的開發和管理。它與現場部署 Oracle 資料庫和 Amazon RDS for Oracle 互動，以執行匯出和匯入資料所需的 SQL 命令。

史詩

建立 S3 儲存貯體

任務	Description	所需的技能
建立儲存貯體。	若要建立 S3 儲存貯體，請遵循 AWS 文件中的指示。	AWS 系統管理員

建立 IAM 角色並指派政策

任務	Description	所需的技能
設定 IAM 許可。	若要設定許可，請遵循 AWS 文件中的指示。	AWS 系統管理員

建立目標 Amazon RDS for Oracle 資料庫執行個體，並關聯 Amazon S3 整合角色

任務	Description	所需的技能
建立目標 Amazon RDS for Oracle 資料庫執行個體。	若要建立 Amazon RDS for Oracle 執行個體，請遵循 AWS 文件中的指示。	AWS 系統管理員
將角色與資料庫執行個體建立關聯。	若要將角色與執行個體建立關聯，請遵循 AWS 文件中的指示。	DBA

在目標資料庫上建立資料庫使用者

任務	Description	所需的技能

建立使用者。	從 Oracle SQL Developer 或 SQL*Plus 連線至目標 Amazon RDS for Oracle 資料庫，並執行下列 SQL 命令來建立要匯入結構描述的使用者。
	`create user SAMPLE_SCHEMA identified by <PASSWORD>;`
	`grant create session, resource to <USER NAME>;`
		DBA

任務	Description	所需的技能
	`alter user <USER NAME> quota 100M on users;`	

從來源 Oracle 資料庫建立匯出檔案

任務	Description	所需的技能
建立資料傾印檔案。	若要在 sample.dmp DATA_PUMP_DIR 目錄中建立名為 的傾印檔案來匯出SAMPLE_SCHEMA 使用者，請使用下列指令碼。 `DECLARE hdnl NUMBER; BEGIN hdnl := dbms_datapump.open(operation => 'EXPORT', job_mode => 'SCHEMA', job_name => NULL); dbms_datapump.add_file( handle => hdnl, filename => 'sample.dmp', directory => 'DATA_PUMP_DIR', filetype => dbms_datapump.ku$_file_type_dump_file);`	DBA

任務	Description	所需的技能

	<pre> dbms_datapump.add_
	file(handle => hdnl,
	
	         filename =>
	 'export.log',
	
	        directory =>
	 'DATA_PUMP_DIR',
	
	         filetype =>
	 dbms_datapump.ku$_
	file_type_log_file);
	
	    dbms_datapump.meta
	data_filter(hdnl,
	 'SCHEMA_EXPR', 'IN
	 (''SAMPLE_SCHEMA''
	)');
	
	    dbms_datapump.star
	t_job(hdnl);
	END;
	/</pre>
	檢閱本機DATA_PUMP_DIR 目錄中export.log 的檔案，以檢閱匯出詳細資訊。	

將傾印檔案上傳至 S3 儲存貯體

任務	Description	所需的技能
將資料傾印檔案從來源上傳至 S3 儲存貯體。	使用 AWS CLI，執行下列命令。	DBA

任務	Description	所需的技能
	`aws s3 cp sample.dmp s3://<bucket_created_epic_1>/`	

從 S3 儲存貯體下載匯出檔案至 RDS 執行個體

任務	Description	所需的技能

將資料傾印檔案下載至 Amazon RDS	若要將傾印檔案`sample.dmp` 從 S3 儲存貯體複製到 Amazon RDS for Oracle 資料庫，請執行下列 SQL 命令。在此範例中，`sample.dmp` 檔案會從 S3 儲存貯體下載`my-s3-integration1` 到 Oracle 目錄 `DATA_PUMP_DIR` 。請確定已將足夠的磁碟空間配置給 RDS 執行個體，以容納資料庫和匯出檔案。
	`-- If you want to download all the files in the S3 bucket remove the p_s3_prefix line.`
	
	`SELECT rdsadmin.rdsadmin_s3_tasks.download_from_s3(`
	`p_bucket_name => 'my-s3-integration',`
	`p_s3_prefix => 'sample.dmp',`
	`p_directory_name => 'DATA_PUMP_DIR')`
	`AS TASK_ID FROM DUAL;`
		AWS 系統管理員

任務	Description	所需的技能

	先前的命令會輸出任務 ID。若要檢閱任務 ID 中的資料來檢閱下載狀態，請執行下列命令。
	`SELECT text FROM table(rdsadmin.rds_file_util.read_text_file('BDUMP','dbtask-<task_id>.log'));`
	若要查看 DATA_PUMP_DIR 目錄中的檔案，請執行下列命令。
	`SELECT filename, type,filesize/1024/1024 size_megs ,to_char(mtime,'DD-MON-YY HH24:MI:SS') timestamp FROM TABLE(rdsadmin.rds_file_util.listdir(p_directory => upper('DATA_PUMP_DIR'))) order by 4;`	

將傾印檔案匯入目標資料庫

任務	Description	所需的技能
將結構描述和資料還原至 Amazon RDS。	若要將傾印檔案匯入 `sample_schema` 資料庫結構描述，請從 SQL Developer 或 SQL*Plus 執行下列 SQL 命令。	DBA

任務	Description	所需的技能

	DECLARE
	hdnl NUMBER;
	BEGIN
	
	hdnl := DBMS_DATA
	PUMP.OPEN(operation
	=> 'IMPORT', job_mode
	=> 'SCHEMA', job_name=
	>null);
	
	DBMS_DATAPUMP.ADD_
	FILE(handle => hdnl,
	filename => 'sample.d
	mp', directory =>
	'DATA_PUMP_DIR',
	filetype => dbms_data
	pump.ku$_file_type
	_dump_file);
	
	DBMS_DATAPUMP.ADD_FILE
	(handle => hdnl,
	filename => 'import.l
	og', directory =>
	'DATA_PUMP_DIR',
	filetype => dbms_data
	pump.ku$_file_type
	_log_file);
	
	DBMS_DATAPUMP.
	METADATA_FILTER(hd
	nl,'SCHEMA_EXPR','
	IN (''SAMPLE_SCHEMA''
	)');
	
	DBMS_DATAPUMP.START_J
	OB(hdnl);
	
	END;
	/	

任務	Description	所需的技能
	若要從匯入查看日誌檔案，請執行下列命令。 `SELECT text FROM table(rdsadmin.rds_file_util.read_text_file('DATA_PUMP_DIR','import.log'));`	

從 DATA_PUMP_DIR 目錄移除傾印檔案

任務	Description	所需的技能

列出並清除匯出檔案。	列出並移除 DATA_PUMP_DIR 目錄中的匯出檔案，執行下列命令。
	
	`-- List the files SELECT filename, type,filesize/1024/1024 size_megs ,to_char(mtime,'DD-MON-YY HH24:MI:SS') timestamp FROM TABLE(rdsadmin.rds_file_util.listdir(p_directory => upper('DATA_PUMP_DIR'))) order by 4;`
	
	`-- Remove the files EXEC UTL_FILE.FREMOVE('DATA_PUMP_DIR','sample.dmp');`
		AWS 系統管理員

任務	Description	所需的技能
	`EXEC UTL_FILE.FREMOVE('DATA_PUMP_DIR','import.log');`	

相關資源

- Amazon S3 整合
- 建立資料庫執行個體
- 在 Amazon RDS 上將資料匯入 Oracle
- Amazon S3 文件
- IAM 文件
- Amazon RDS 文件
- Oracle Data Pump 文件
- Oracle SQL Developer

使用 pglogical 從 Amazon EC2 上的 PostgreSQL 遷移至 Amazon RDS for PostgreSQL Amazon EC2

Rajesh Madiwale，Amazon Web Services

總結

此模式概述使用 PostgreSQL pglogical 延伸模組，將 PostgreSQL 資料庫 (9.5 版及更新版本）從 Amazon Elastic Compute Cloud (Amazon EC2) 遷移至 Amazon Relational Database Service (Amazon RDS) for PostgreSQL 的步驟。Amazon RDS 現在支援 PostgreSQL 第 10 版的 pglogical 擴充功能。

先決條件和限制

先決條件

- 選擇正確的 Amazon RDS 執行個體類型。如需詳細資訊，請參閱Amazon RDS 執行個體類型。
- 請確定 PostgreSQL 的來源和目標版本相同。
- 安裝pglogical 延伸模組並與 PostgreSQLon Amazon EC2 整合。

產品版本

- Amazon RDS 上的 PostgreSQL 10 版和更新版本，具有 Amazon RDS 上支援的功能（請參閱 AWS 文件中的 Amazon RDS 上的 PostgreSQL)。此模式是透過在 Amazon RDS 上將 PostgreSQL 9.5 遷移至 PostgreSQL 10 版來測試，但也適用於 Amazon RDS 上的 PostgreSQL 更新版本。

Architecture

資料遷移架構

工具

- pglogical 延伸模組
- PostgreSQL 原生公用程式：pg_dump 和 pg_restore

史詩

使用 pglogical 延伸模組遷移資料

任務	Description	所需的技能
建立 Amazon RDS PostgreSQL 資料庫執行個體。	在 Amazon RDS 中設定 PostgreSQL 資料庫執行個體。如需說明，請參閱 Amazon RDS for PostgreSQL 文件。	DBA
從來源 PostgreSQL 資料庫取得結構描述傾印，並將其還原至目標 PostgreSQL 資料庫。	1. 使用 pg_dump 公用程式搭配 `-s`選項，從來源資料庫產生結構描述檔案。 2. 使用 psql 公用程式搭配 `-f`選項，將結構描述載入目標資料庫。	DBA
開啟邏輯解碼。	在 Amazon RDS 資料庫參數群組中，將`rds.logical_replication` 靜態參數設定為 1。如需說明，請參閱 Amazon RDS 文件。	DBA
在來源和目標資料庫上建立 pglogical 延伸。	1. 在來源 PostgreSQL 資料庫上建立`pglogical` 擴充功能： `psql -h <amazon-ec2-endpoint> -d target-dbname -U target-dbuser -c "create extension pglogical ;"` 2. 在目標 PostgreSQL 資料庫上建立`pglogical` 擴充功能：	DBA

任務	Description	所需的技能
	`psql -h <amazon-rds-endpoint> -d source-dbname -U source-dbuser -c "create extension pglogical ;"`	
在來源 PostgreSQL 資料庫上建立發佈者。	若要建立發佈者，請執行： `psql -d dbname -p 5432 <<EOF SELECT pglogical .create_n ode( node_name := 'provider1', dsn := 'host=<ec2-endpoin t> port=5432 dbname=so urce-dbname user=sour ce-dbuser' ); EOF`	DBA
建立複寫集、新增資料表和序列。	若要在來源 PostgreSQL 資料庫上建立複寫集，以及將資料表和序列新增至複寫集，請執行： `psql -d dbname -p 5432 <<EOF SELECT pglogical .replication_set_a dd_all_tables('def ault', '{public} '::text[],synchron ize_data := true); EOF`	DBA

任務	Description	所需的技能
建立訂閱者。	若要在目標 PostgreSQL 資料庫上建立訂閱者，請執行： `psql -h <rds-endpoint> -d target-dbname -U target-dbuser <<EOF SELECT pglogical.create_node( node_name := 'subscriber1', dsn := 'host=<rds-endpoint> port=5432 dbname=target-dbname password=postgres user=target-dbuser' ); EOF`	DBA
建立訂閱。	若要在目標 PostgreSQL 資料庫上建立訂閱，請執行： `psql -h <rds-endpoint> -d target -U postgres <<EOF SELECT pglogical.create_subscription( subscription_name := 'subscription1', replication_sets := array['default'], provider_dsn := 'host=<ec2-endpoint> port=5432 dbname=<source-dbname> password=<password> user=source-dbuser' ); EOF`	DBA

驗證您的資料

任務	Description	所需的技能
檢查來源和目標資料庫。	檢查來源和目標資料庫，以確認資料已成功複寫。您可以從來源和目標資料表使用 `select count(1)` 執行基本驗證。	DBA

相關資源

- Amazon RDS
- Amazon RDS 上的 PostgreSQL 邏輯複寫 (Amazon RDS 文件）
- pglogical (GitHub 儲存庫）
- pglogical 的限制 (GitHub 儲存庫 README 檔案）
- 使用邏輯複寫將 PostgreSQL 從內部部署或 Amazon EC2 遷移至 Amazon RDS (AWS 資料庫部落格）

將內部部署 PostgreSQL 資料庫遷移至 Aurora PostgreSQL

Baji Shaik 和 Jitender Kumar，Amazon Web Services

總結

Amazon Aurora PostgreSQL 相容版本結合了高階商業資料庫的效能和可用性，以及開放原始碼資料庫的簡單性和成本效益。Aurora 透過將儲存體擴展到相同 AWS 區域中的三個可用區域，提供這些優勢，並支援最多 15 個僅供讀取複本執行個體，以擴展讀取工作負載，並在單一區域中提供高可用性。透過使用 Aurora 全域資料庫，您可以在最多五個區域中複寫 PostgreSQL 資料庫，以便在發生區域故障時進行遠端讀取存取和災難復原。此模式說明將內部部署 PostgreSQL 來源資料庫遷移至 Aurora PostgreSQL 相容資料庫的步驟。模式包含兩個遷移選項：使用 AWS Data Migration Service (AWS DMS) 或使用原生 PostgreSQL 工具（例如，pg_dump、pg_restore 和psql) 或第三方工具。

此模式中描述的步驟也適用於 Amazon Relational Database Service (Amazon RDS) 和 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的目標 PostgreSQL 資料庫。 Amazon Relational Database Service

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署資料中心中的 PostgreSQL 來源資料庫
- Aurora PostgreSQL 相容資料庫執行個體或 Amazon RDS for PostgreSQL 資料庫執行個體

限制

- Amazon RDS for PostgreSQL 的資料庫大小限制為 64 TB，Aurora PostgreSQL 相容則為 128 TB。
- 如果您使用的是 AWS DMS 遷移選項，請檢閱使用 PostgreSQL 資料庫做為來源的 AWS DMS 限制。

產品版本

- 如需 Amazon RDS 中的 PostgreSQL 主要和次要版本支援，請參閱 Amazon RDS 文件中的 Amazon RDS for PostgreSQL 更新。
- 如需 Aurora 中的 PostgreSQL 支援，請參閱 Aurora 文件中的 Amazon Aurora PostgreSQL 更新。
- 如果您使用的是 AWS DMS 遷移選項，請參閱 AWS DMS 文件中的支援的 PostgreSQL 版本。

Architecture

來源技術堆疊

- 內部部署PostgreSQL 資料庫

目標技術堆疊

- Aurora PostgreSQL 相容資料庫執行個體

來源架構

目標架構

資料遷移架構

使用 AWS DMS

使用原生 PostgreSQL 工具

工具

- AWS Database Migration Service (AWS DMS) 可協助您將資料存放區遷移至 AWS 雲端，或在雲端和內部部署組態的組合之間遷移。此服務支援不同的來源和目標資料庫。如需有關如何驗證支援與 AWS DMS 搭配使用之 PostgreSQL 來源和目標資料庫版本的資訊，請參閱使用 PostgreSQL 資料庫做為 AWS DMS 來源。我們建議您使用最新版本的 AWS DMS，以獲得最全面的版本和功能支援。
- 原生 PostgreSQL 工具包括 pg_dump、pg_restore 和psql。

史詩

分析遷移

任務	Description	所需的技能
驗證來源和目標資料庫版本。	如果您使用的是 AWS DMS，請確定您使用的是支援的 PostgreSQL 版本。	DBA
識別儲存類型和容量需求。	1. 計算為來源資料庫執行個體配置的儲存體。 2. 收集來源資料庫執行個體的歷史成長指標。 3. 預測目標資料庫執行個體的未來成長預測。 4. 透過計算來源資料庫上的讀取和寫入 IOPS 總數來配置儲存體。一般用途 SSD (gp2) 磁碟區為每 1 GB 的儲存提供 3 IOPS。	DBA，系統管理員
選擇適當的執行個體類型、容量、儲存功能和網路功能。	判斷目標資料庫執行個體的運算需求。檢閱可能需要額外注意的已知效能問題。請考慮下列因素，以判斷適當的執行個體類型： • 來源資料庫執行個體的 CPU 使用率 • 來源資料庫執行個體的 IOPS（讀取和寫入操作） • 來源資料庫執行個體上的記憶體使用量	DBA，系統管理員

任務	Description	所需的技能
	如需詳細資訊，請參閱 Aurora 文件中的 Aurora 資料庫執行個體類別。	
識別來源和目標資料庫的網路存取安全需求。	決定適當的安全群組，讓應用程式能夠與資料庫通訊。	DBA，系統管理員
識別應用程式遷移策略。	• 根據應用程式的複雜性決定遷移切換策略。 • 確定應用程式的復原時間目標 (RTO) 和復原點目標 (RPO)，並相應地規劃切換。	DBA、應用程式擁有者、系統管理員

設定基礎設施

任務	Description	所需的技能
建立 VPC。	為目標資料庫執行個體建立新的虛擬私有雲端 (VPC)。	系統管理員
建立安全群組。	在 VPC 內建立安全群組（如上一個圖示中所決定），以允許資料庫執行個體的傳入連線。	系統管理員
設定和啟動 Aurora 資料庫叢集。	使用新的 VPC 和安全群組建立目標資料庫執行個體，並啟動執行個體。	系統管理員

遷移資料 – 選項 1（使用 AWS DMS)

任務	Description	所需的技能
完成預遷移步驟。	1. 清除來源資料庫中的資料。	DBA

任務	Description	所需的技能
	2. 建立複寫執行個體。 3. 建立來源和目標端點。 4. 識別要遷移的可用資料表和物件數量。	
完成遷移步驟。	1. 刪除目標資料庫的外部索引鍵限制和觸發。 2. 在目標資料庫上捨棄次要索引。 3. 使用完全載入任務，將資料從來源遷移至目標資料庫。 4. 啟用外部金鑰。 5. 如果您使用的是 Flash-cut 遷移，且您的應用程式需要最短的停機時間，請啟用變更資料擷取 (CDC) 以複寫持續的變更 6. 啟用觸發。 7. 更新序列。 8. 驗證來源和目標資料。	DBA
驗證資料。	為了確保您的資料從來源準確遷移到目標，請遵循 AWS DMS 文件中的資料驗證步驟。	DBA

遷移資料 – 選項 2 （使用 pg_dump 和 pg_restore)

任務	Description	所需的技能
準備來源資料庫。	1. 建立目錄以存放不存在的 pg_dump 備份。	DBA

任務	Description	所需的技能
	2. 建立具有在資料庫物件上執行 pg_dump 許可的遷移使用者。 3. 連接至 EC2 執行個體並執行 pg_dump 備份。 如需詳細資訊，請參閱 AWS DMS 文件中的 pg_dump 文件和逐步解說。	
準備目標資料庫。	1. 建立具有在資料庫物件上使用 pg_restore 許可的遷移使用者。 2. 使用 pg_restore 匯入資料庫傾印。 如需詳細資訊，請參閱 AWS DMS 文件中的 pg_restore 文件和逐步解說。	DBA
驗證資料。	1. 比較來源和目標資料庫之間的資料庫物件計數。 2. 解決在物件計數之間發現的任何差異。	DBA

遷移應用程式

任務	Description	所需的技能
遵循應用程式遷移策略。	實作您在第一個史詩中建立的應用程式遷移策略。	DBA、應用程式擁有者、系統管理員

切換到目標資料庫

任務	Description	所需的技能
將應用程式用戶端切換到新的基礎設施。	1. 停止指向內部部署 PostgreSQL 資料庫的所有應用程式服務和用戶端連線。 2. 執行 AWS DMS 任務。 3. 視需要設定復原任務（從 Aurora PostgreSQL 相容於內部部署 PostgreSQL 資料庫的反向 CDC)。 4. 驗證資料。 5. 透過將 Amazon Route 53 設定為新的 Aurora PostgreSQL 相容資料庫執行個體，在新目標上啟動應用程式服務。 6. 在新的 Aurora PostgreSQL 相容資料庫執行個體上新增 Amazon CloudWatch 和績效詳情監控。	DBA、應用程式擁有者、系統管理員
如果您需要復原遷移。	1. 停止指向 Aurora PostgreSQL 相容資料庫的所有應用程式服務。 2. 使用您在上一個案例中建立的 AWS DMS 任務，將變更轉返至來源現場部署 PostgreSQL 資料庫。 3. 停止從內部部署 PostgreSQL 資料庫執行到 Aurora PostgreSQL 相容資料庫的 AWS DMS 任務。	DBA、應用程式擁有者

任務	Description	所需的技能
	4. 設定應用程式，使其指向來源內部部署 PostgreSQL 資料庫。 5. 確認所有轉返部署已完成。	

關閉專案

任務	Description	所需的技能
關閉資源。	關閉臨時 AWS 資源。	DBA，系統管理員
驗證文件。	檢閱並驗證專案文件。	DBA、應用程式擁有者、系統管理員
收集指標。	收集遷移時間、手動與工具成本節省百分比等指標。	DBA、應用程式擁有者、系統管理員
關閉專案。	關閉專案並提供任何意見回饋。	DBA、應用程式擁有者、系統管理員

相關資源

參考

- AWS Data Migration Service
- VPCs和 Amazon Aurora
- Amazon Aurora 定價
- 使用 PostgreSQL 資料庫做為 AWS DMS 來源
- 如何建立 AWS DMS 複寫執行個體
- 如何使用 AWS DMS 建立來源和目標端點

其他資源

- AWS DMS 入門

- 資料遷移step-by-step演練
- Amazon Aurora 資源

將內部部署 Microsoft SQL Server 資料庫遷移至執行 Linux 的 Amazon EC2 上的 Microsoft SQL Server

Tirumala Dasari，Amazon Web Services

總結

此模式說明如何使用備份和還原公用程式，從在 Microsoft Windows 上執行的內部部署 Microsoft SQL Server 資料庫遷移至 Amazon Elastic Compute Cloud (Amazon EC2) Linux 執行個體上的 Microsoft SQL Server。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 搭配 Microsoft SQL Server 的 Amazon EC2 Linux AMI (Amazon Machine Image)
- Linux EC2 執行個體上的現場部署 Windows 和 Microsoft SQL Server 之間的 AWS Direct Connect

Architecture

來源技術堆疊

- 內部部署 Microsoft SQL Server 資料庫

目標技術堆疊

- 具有 Microsoft SQL Server 資料庫的 Linux EC2 執行個體

資料庫遷移架構

工具

- WinSCP - 此工具可讓 Windows 使用者輕鬆與 Linux 使用者共用檔案。
- Sqlcmd - 此命令列公用程式可讓您將 T-SQL 陳述式或批次提交至 SQL Server 的本機和遠端執行個體。此公用程式對於批次處理或單元測試等重複性資料庫任務非常有用。

史詩

使用 SQL Server 準備 EC2 Linux 執行個體

任務	Description	所需的技能
選取提供 Linux 作業系統並包含 Microsoft SQL Server 的 AMI。		Sysadmin
設定 AMI 以建立 EC2 執行個體。		Sysadmin
建立安全群組的傳入和傳出規則。		Sysadmin
設定 Microsoft SQL Server 資料庫的 Linux EC2 執行個體。		DBA
建立使用者並提供與來源資料庫中相同的許可。		Appowner、DBA
在 Linux EC2 執行個體上安裝 SQL Server 工具和 sqlcmd 公用程式。		DBA

備份資料庫並將備份檔案移至 Linux EC2 執行個體

任務	Description	所需的技能
備份現場部署 SQL Server 資料庫。		DBA
在 Microsoft SQL Server 上安裝 WinSCP。		DBA

任務	Description	所需的技能
將備份檔案移至執行 Microsoft SQL Server 的 Linux EC2 執行個體。		DBA

在執行 SQL Server 的 Linux EC2 執行個體上還原資料庫

任務	Description	所需的技能
使用 sqlcmd 公用程式從資料庫備份檔案還原資料庫。		DBA
驗證資料庫物件和資料。		開發人員、測試工程師

從 Windows SQL Server 切換到 Linux EC2 執行個體上的 Windows SQL Server

任務	Description	所需的技能
驗證資料庫物件和資料。		開發人員、測試工程師
從內部部署 Microsoft SQL Server 資料庫切換到執行 Microsoft SQL Server 的 Linux EC2 執行個體。		DBA

相關資源

- 如何在 Amazon Linux 和 Ubuntu AMIs 上設定 SQL Server 2017
- 在 Linux 執行個體上安裝 SQL 工具
- 從內部部署 Microsoft SQL Server 資料庫備份和還原至 Linux EC2 執行個體上的 Microsoft SQL Server

使用連結的伺服器將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon RDS for SQL Server

Kevin Yung、Vitakash Adwani 和 Amazon Web Services Vishal Singh

總結

連結的伺服器可讓 Microsoft SQL Server 在資料庫伺服器的其他執行個體上執行 SQL 陳述式。此模式說明如何將內部部署 Microsoft SQL Server 資料庫遷移至 Microsoft SQL Server 的 Amazon Relational Database Service (Amazon RDS)，以實現更低的成本和更高的可用性。目前，Amazon RDS for Microsoft SQL Server 不支援 Amazon Virtual Private Cloud (Amazon VPC) 網路以外的連線。

您可以使用此模式來達成下列目標：

- 將 Microsoft SQL Server 遷移至 Amazon RDS for Microsoft SQL Server，而不會中斷連結的伺服器功能。
- 在不同的波段中排定和遷移連結的 Microsoft SQL Server 的優先順序。

先決條件和限制

先決條件

- 檢查 Amazon RDS 上的 Microsoft SQL Server 是否支援您需要的功能。
- 請確定您可以使用Amazon RDS for Microsoft SQL Server 搭配預設定序或透過資料庫層級設定的定序。

Architecture

來源技術堆疊

- 內部部署資料庫 (Microsoft SQL Server)

目標技術堆疊

- Amazon RDS for SQL Server

來源狀態架構

目標狀態架構

在目標狀態下，您可以使用連結的伺服器，將 Microsoft SQL Server 遷移至 Amazon RDS for Microsoft SQL Server。此架構使用 Network Load Balancer 將流量從 Amazon RDS for Microsoft SQL Server 代理到執行 Microsoft SQL Server 的內部部署伺服器。下圖顯示 Network Load Balancer 的反向代理功能。

工具

- AWS CloudFormation
- Network Load Balancer
- 位於多個可用區域的 Amazon RDS for SQL Server （多AZs區域）
- AWS Database Migration Service (AWS DMS)

史詩

建立登陸區域 VPC

任務	Description	所需的技能
建立 CIDR 配置。		AWS SysAdmin
建立 Virtual Private Cloud (VPC)		AWS SysAdmin
建立 VPC 子網路。		AWS SysAdmin

任務	Description	所需的技能
建立子網路存取控制清單 ACLs)。		AWS SysAdmin
建立子網路路由表。		AWS SysAdmin
建立與 AWS Direct Connect 或 AWS Virtual Private Network (VPN) 的連線。		AWS SysAdmin

將資料庫遷移至 Amazon RDS

任務	Description	所需的技能
建立 Amazon RDS for Microsoft SQL Server 資料庫執行個體。		AWS SysAdmin
建立 AWS DMS 複寫執行個體。		AWS SysAdmin
在 AWS DMS 中建立來源和目標資料庫端點。		AWS SysAdmin
建立遷移任務，並在完全載入後將連續複寫設定為 ON。		AWS SysAdmin
請求防火牆變更，以允許 Amazon RDS for Microsoft SQL Server 存取內部部署 SQL Server 資料庫。		AWS SysAdmin
建立 Network Load Balancer。		AWS SysAdmin
建立目標群組，以資料中心中的資料庫伺服器為目標	我們建議您在目標設定中使用主機名稱來整合資料中心 (DC) 容錯移轉事件。	AWS SysAdmin

任務	Description	所需的技能
針對連結的伺服器設定執行 SQL 陳述式。	針對 Amazon RDS for Microsoft SQL Server 資料庫執行個體，使用 Microsoft SQL 管理工具執行 SQL 陳述式來新增連結的伺服器。在 SQL 陳述式中，將 @datasrc 設定為使用 Network Load Balancer 主機名稱。針對 Amazon RDS for Microsoft SQL Server 資料庫執行個體，使用 Microsoft SQL 管理工具新增連結的伺服器登入憑證。	AWS SysAdmin
測試和驗證 SQL Server 函數。		AWS SysAdmin
建立切換。		AWS SysAdmin

相關資源

- Amazon RDS 上 Microsoft SQL Server 的常見管理任務
- Microsoft SQL Server 的定序和字元集
- Network Load Balancer 文件
- 使用 Amazon RDS for Microsoft SQL Server 實作連結的伺服器 （部落格文章）

使用原生備份和還原方法，將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon RDS for SQL Server

Tirumala Dasari、David Queiroz 和 Amazon Web Services Vishal Singh

總結

此模式說明如何將內部部署 Microsoft SQL Server 資料庫遷移至 SQL Server 資料庫執行個體的 Amazon Relational Database Service (Amazon RDS)（同質遷移）。遷移程序是以原生 SQL Server 備份和還原方法為基礎。它使用 SQL Server Management Studio (SSMS) 來建立資料庫備份檔案，並使用 Amazon Simple Storage Service (Amazon S3) 儲存貯體來存放備份檔案，然後再將其還原至 Amazon RDS for SQL Server。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 存取 S3 儲存貯體和 Amazon RDS for SQL Server 資料庫執行個體的 AWS Identity and Access Management (IAM) 角色政策。

限制

- 此模式中描述的程序只會遷移資料庫。SQL 登入或資料庫使用者，包括任何 SQL Server Agent 任務，都不會遷移，因為它們需要額外的步驟。

產品版本

- SQL Server 2012-2017。如需支援版本和功能的最新清單，請參閱 AWS 文件中的 Amazon RDS 上的 Microsoft SQL Server。

Architecture

來源技術堆疊

- 內部部署 Microsoft SQL Server 資料庫

目標技術堆疊

- Amazon RDS for SQL Server 資料庫執行個體

資料遷移架構

工具

- Microsoft SQL Server Management Studio (SSMS) 是用於管理 SQL Server 基礎設施的整合環境。它提供使用者介面和一組工具，其中包含與 SQL Server 互動的豐富指令碼編輯器。

史詩

建立 Amazon RDS for SQL Server 資料庫執行個體

任務	Description	所需的技能
選取 SQL Server 做為 Amazon RDS for SQL Server 中的資料庫引擎。		DBA
選擇 SQL Server Express Edition。		DBA
指定資料庫詳細資訊。	如需建立資料庫執行個體的詳細資訊，請參閱 Amazon RDS 文件。	DBA、應用程式擁有者

從內部部署 SQL Server 資料庫建立備份檔案

任務	Description	所需的技能
透過 SSMS 連線至內部部署 SQL Server 資料庫。		DBA
建立資料庫的備份。	如需說明，請參閱 SSMS 文件。	DBA、應用程式擁有者

將備份檔案上傳至 Amazon S3

任務	Description	所需的技能
在 Amazon S3 中建立儲存貯體。	如需詳細資訊，請參閱 Amazon S3 說明文件。	DBA
將備份檔案上傳至 S3 儲存貯體。	如需詳細資訊，請參閱 Amazon S3 說明文件。	SysOps 管理員

在 Amazon RDS for SQL Server 中還原資料庫

任務	Description	所需的技能
將選項群組新增至 Amazon RDS。	1. 前往 https://console.aws.amazon.com/rds/，開啟 Amazon RDS 主控台。 2. 在導覽窗格中，選擇選項群組、建立群組。 3. 完成選項群組的資訊，然後選擇建立。 4. 將 SQLSERVER_BACKUP_RESTORE 選項新增至選項群組，然後選擇新增選項。 如需詳細資訊，請參閱 Amazon RDS 文件。	SysOps 管理員
還原資料庫。	1. 透過 SSMS 連線至 Amazon RDS for SQL Server。 2. 呼叫`msdb.dbo.rds_restore_database` 預存程序以還原資料庫。	DBA

驗證目標資料庫

任務	Description	所需的技能
驗證物件和資料。	驗證來源資料庫與 Amazon RDS for SQL Server 之間的物件和資料。 **Note** 此任務只會遷移資料庫。不會遷移登入和任務。	應用程式擁有者、DBA

剪下

任務	Description	所需的技能
重新導向應用程式流量。	驗證後，將應用程式流量重新導向至 Amazon RDS for SQL Server 資料庫執行個體。	應用程式擁有者、DBA

相關資源

- Amazon S3 文件
- Amazon RDS for SQL Server 文件
- Microsoft SQL Server 資料庫引擎的選項

使用 AWS DMS 和 AWS SCT 將 Microsoft SQL Server 資料庫遷移至 Aurora MySQL

Mark Szalkiewicz 和 Pavan Pusuluri，Amazon Web Services

總結

此模式說明如何將內部部署或 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的 Microsoft SQL Server 資料庫遷移至 Amazon Aurora MySQL。模式使用 AWS Database Migration Service (AWS DMS) 和 AWS Schema Conversion Tool (AWS SCT) 進行資料遷移和結構描述轉換。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署資料中心或 EC2 執行個體上的 Microsoft SQL Server 來源資料庫
- AWS SCT 連接器的 Java Database Connectivity (JDBC) 驅動程式，安裝在本機電腦或安裝 AWS SCT 的 EC2 執行個體上

限制

- 資料庫大小限制：64 TB

產品版本

- 適用於 Enterprise、Standard、Workgroup 和 Developer 版本的 Microsoft SQL Server 2008、2008R2、2012、2014、2016 和 2017。AWS DMS 不支援 Web 和 Express 版本。如需支援版本的最新清單，請參閱使用 Microsoft SQL Server 資料庫做為 AWS DMS 的來源。我們建議您使用最新版本的 AWS DMS，以獲得最全面的版本和功能支援。如需 AWS SCT 支援的 Microsoft SQL Server 版本相關資訊，請參閱 AWS SCT 文件。
- MySQL 5.5、5.6 和 5.7 版。如需支援版本的最新清單，請參閱使用 MySQL 相容資料庫做為 AWS DMS 的目標。

Architecture

來源技術堆疊

下列其中一項：

- 內部部署 Microsoft SQL Server 資料庫
- EC2 執行個體上的 Microsoft SQL Server 資料庫

目標技術堆疊

- Aurora MySQL

資料遷移架構

- 從 AWS 雲端中執行的 Microsoft SQL Server 資料庫

- 從內部部署資料中心執行的 Microsoft SQL Server 資料庫

工具

- AWS DMS - AWS Data Migration Service (AWS DMS) 可協助您在廣泛使用的商業和開放原始碼資料庫之間遷移資料，包括 Oracle、SQL Server、MySQL 和 PostgreSQL。您可以使用 AWS DMS 將資料遷移至 AWS 雲端，可在現場部署執行個體 (透過 AWS 雲端設定) 或在雲端和現場部署設定之間進行。
- AWS SCT - AWS Schema Conversion Tool (AWS SCT) 會自動將來源資料庫結構描述和大部分自訂程式碼轉換為與目標資料庫相容的格式，讓異質資料庫遷移變得更容易。

史詩

準備遷移

任務	Description	所需的技能
驗證來源和目標資料庫版本和引擎。		DBA
為來源和目標資料庫建立傳出安全群組。		SysAdmin

任務	Description	所需的技能
視需要建立和設定 AWS SCT 的 EC2 執行個體。		DBA
下載最新版本的 AWS SCT 和相關聯的驅動程式。		DBA
在來源資料庫中新增和驗證先決條件使用者和授權。		DBA
為工作負載建立 AWS SCT 專案，並連線至來源資料庫。		DBA
產生評估報告並評估可行性。		DBA

準備目標資料庫

任務	Description	所需的技能
使用 Amazon Aurora 做為資料庫引擎，建立目標 Amazon RDS 資料庫執行個體。		DBA
從來源擷取使用者、角色和授予的清單。		DBA
將現有的資料庫使用者映射至新的資料庫使用者。		應用程式擁有者
在目標資料庫中建立使用者。		DBA
將上一個步驟的角色套用至目標資料庫。		DBA
檢閱來源資料庫中的資料庫選項、參數、網路檔案和資料庫		DBA

任務	Description	所需的技能
連結，然後評估其對目標資料庫的適用性。		
將任何相關設定套用至目標。		DBA

傳輸物件

任務	Description	所需的技能
設定目標資料庫的 AWS SCT 連線。		DBA
使用 AWS SCT 轉換結構描述。	AWS SCT 會自動將來源資料庫結構描述和大多數自訂程式碼轉換為與目標資料庫相容的格式。工具無法自動轉換的任何程式碼都會清楚標示，讓您可以自行轉換。	DBA
檢閱產生的 SQL 報告，並儲存任何錯誤和警告。		DBA
將自動化結構描述變更套用至目標，或將其儲存為 .sql 檔案。		DBA
驗證 AWS SCT 是否已在目標上建立物件。		DBA
手動重寫、拒絕或重新設計任何無法自動轉換的項目。		DBA
套用產生的角色和使用者授權，並檢閱任何例外狀況。		DBA

遷移資料

任務	Description	所需的技能
決定遷移方法。		DBA
從 AWS DMS 主控台建立複寫執行個體。	如需使用 AWS DMS 的詳細資訊，請參閱「相關資源」一節中的連結。	DBA
建立來源和目標端點。		DBA
建立複寫任務。		DBA
啟動複寫任務並監控日誌。		DBA

遷移應用程式

任務	Description	所需的技能
使用 AWS SCT 來分析和轉換應用程式程式碼中的 SQL 項目。	當您將資料庫結構描述從一個引擎轉換到另一個引擎，您也需更新應用程式中的 SQL 程式碼，以便與新的資料庫引擎互動，取代舊引擎。您可以檢視、分析、編輯和儲存轉換後的 SQL 程式碼。如需使用 AWS SCT 的詳細資訊，請參閱「相關資源」一節中的連結。	應用程式擁有者
在 AWS 上建立新的應用程式伺服器。		應用程式擁有者
將應用程式碼遷移至新的伺服器。		應用程式擁有者

任務	Description	所需的技能
設定目標資料庫和驅動程式的應用程式伺服器。		應用程式擁有者
修正應用程式中來源資料庫引擎特有的任何程式碼。		應用程式擁有者
最佳化目標引擎的應用程式碼。		應用程式擁有者

切換

任務	Description	所需的技能
將任何新使用者、授權和程式碼變更套用至目標。		DBA
鎖定應用程式是否有任何變更。		應用程式擁有者
驗證所有變更是否已傳播到目標資料庫。		DBA
將新的應用程式伺服器指向目標資料庫。		應用程式擁有者
重新檢查所有項目。		應用程式擁有者
上線。		應用程式擁有者

關閉專案

任務	Description	所需的技能
關閉臨時 AWS 資源 (AWS DMS 複寫執行個體和用於		DBA、應用程式擁有者

任務	Description	所需的技能
AWS SCT 的 EC2 執行個體）。		
更新內部團隊的 AWS DMS 程序意見回饋。		DBA、應用程式擁有者
修訂 AWS DMS 程序並視需要改善範本。		DBA、應用程式擁有者
檢閱並驗證專案文件。		DBA、應用程式擁有者
收集遷移時間、手動與工具成本節省百分比等指標。		DBA、應用程式擁有者
關閉專案並提供任何意見回饋。		DBA、應用程式擁有者

相關資源

參考

- AWS DMS 使用者指南
- AWS SCT 使用者指南
- Amazon Aurora 定價

教學課程和影片

- AWS Database Migration Service 入門
- AWS Schema Conversion Tool入門
- Amazon RDS 資源
- AWS DMS Step-by-Step演練

使用原生工具將內部部署 MariaDB 資料庫遷移至 Amazon RDS for MariaDB

Shyam Sunder Rakhecha，Amazon Web Services

總結

此模式提供使用原生工具將內部部署 MariaDB 資料庫遷移至 MariaDB 的 Amazon Relational Database Service (Amazon RDS) 的指引。如果您已安裝 MySQL 工具，則可以使用 mysql 和 mysqldump。如果您已安裝 MariaDB 工具，則可以使用 mariadb 和 mariadb-dump。MySQL 和 MariaDB 工具具有相同的原始伺服器，但在 MariaDB 10.6 版和更新版本中存在細微差異。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 內部部署資料中心中的 MariaDB 來源資料庫

限制

- 資料庫大小限制：64 TB

產品版本

- MariaDB 10.0-10.6 版（如需支援版本的最新清單，請參閱 AWS 文件中的 Amazon RDS 上的 MariaDB)

Architecture

來源技術堆疊

- 內部部署資料中心中的 MariaDB 資料庫

目標技術堆疊

- Amazon RDS for MariaDB 資料庫執行個體

目標架構

資料遷移架構

工具

- 原生 MySQL 工具：mysql 和 mysqldump
- 原生 MariaDB 工具：mariadb 和 mariadb-dump

史詩

規劃遷移

任務	Description	所需的技能
驗證來源和目標資料庫版本和引擎。		DBA
識別目標伺服器執行個體的硬體需求。		DBA，系統管理員
識別儲存需求（儲存類型和容量）。		DBA，系統管理員
根據容量、儲存功能和網路功能選擇適當的執行個體類型。		DBA，系統管理員
識別來源和目標資料庫的網路存取安全需求。		DBA，系統管理員
識別應用程式遷移策略。		DBA、應用程式擁有者、系統管理員

設定基礎設施

任務	Description	所需的技能
建立 Virtual Private Cloud (VPC)		系統管理員

任務	Description	所需的技能
建立安全群組。		系統管理員
設定並啟動執行 MariaDB 的 Amazon RDS 資料庫執行個體。		系統管理員

遷移資料

任務	Description	所需的技能
使用原生工具來遷移資料庫物件和資料。	在來源資料庫中，使用 mysqldump ormariadb-dump 建立包含資料庫物件和資料的輸出檔案。在目標資料庫中，使用 mysql 或 mariadb 還原資料。	DBA
驗證資料。	檢查來源和目標資料庫，以確認資料遷移成功。	DBA

遷移應用程式

任務	Description	所需的技能
遵循應用程式遷移策略。		DBA、應用程式擁有者、系統管理員

剪下

任務	Description	所需的技能
將應用程式用戶端切換到新的基礎設施。		DBA、應用程式擁有者、系統管理員

關閉專案

任務	Description	所需的技能
關閉臨時 AWS 資源。		系統管理員
檢閱並驗證專案文件。		DBA、應用程式擁有者、系統管理員
收集遷移時間、工具提供的成本節省等指標。		DBA、應用程式擁有者、系統管理員
關閉專案並提供意見回饋。		DBA、應用程式擁有者、系統管理員

相關資源

Amazon RDS 參考

- Amazon RDS for MariaDB
- Amazon Virtual Private Cloud VPCs和 Amazon RDS
- Amazon RDS 異地同步備份部署
- Amazon RDS 定價

MySQL 和 MariaDB 參考

- mariadb-dump/mysqldump
- mysql 命令列用戶端

教學課程和影片

- Amazon RDS 入門

將內部部署 MySQL 資料庫遷移至 Aurora MySQL

Igor Obradovic，Amazon Web Services

總結

此模式說明如何將內部部署 MySQL 來源資料庫遷移至 Amazon Aurora MySQL 相容版本。它描述了兩種遷移選項：使用 AWS Database Migration Service (AWS DMS) 或使用 mysqldbcopy 和mysqldump 等原生 MySQL 工具。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 內部部署資料中心中的來源 MySQL 資料庫

限制

- 資料庫大小限制：128 TB

產品版本

- MySQL 8.0 版 (Aurora MySQL 第 3 版） 可在標準支援下使用。
- MySQL 5.7 版 (Aurora MySQL 2 版） 可在擴充支援下使用，需額外付費。

如需支援版本的最新清單，請參閱 AWS 文件中的 Amazon Aurora 版本。如果您使用的是 AWS DMS，另請參閱使用 MySQL 相容資料庫做為 所支援 for MySQL 版本的目標 AWS DMS AWS DMS。MySQL

Architecture

來源技術堆疊

- 內部部署 MySQL 資料庫

目標技術堆疊

- Amazon Aurora MySQL-Compatible Edition

目標架構

Aurora 資料存放在叢集磁碟區中，這是使用固態硬碟 (SSDs的單一虛擬磁碟區。叢集磁碟區包含跨單一 AWS 區域的三個可用區域的資料複本。由於資料會自動跨可用區域複寫，因此非常耐用，且資料遺失的可能性較低。

Aurora 會自動將您的資料庫磁碟區分割為分散在許多磁碟的 10 GB 區段。資料庫磁碟區的每個 10 GB 區塊都會以六種方式跨三個可用區域複寫。下圖說明叢集磁碟區、寫入器資料庫執行個體和 Aurora 資料庫叢集中讀取器資料庫執行個體之間的關係，以及運算容量和儲存體的分離。如需此架構的詳細資訊，請參閱 Aurora 文件和常見問答集。

資料遷移架構

使用 AWS DMS：

下圖說明使用 將內部部署 MySQL 資料庫遷移至 中與 Aurora MySQL 相容的叢集 AWS 雲端 AWS DMS。

使用原生 MySQL 工具：

下圖說明使用 mysqldbcopy 和mysqldump 等原生 MySQL 工具 AWS 雲端，將內部部署 MySQL 資料庫遷移至 中的 Aurora MySQL 相容叢集。

工具

- AWS Database Migration Service (AWS DMS) 支援數個來源和目標資料庫引擎。如需 支援的 MySQL 來源和目標資料庫的相關資訊 AWS DMS，請參閱將 MySQL 相容資料庫遷移至 AWS。我們建議您使用最新版本的 AWS DMS ，以獲得最全面的版本和功能支援。
- mysqldbcopy 是一種 MySQL 公用程式，可在單一伺服器上或在伺服器之間複製 MySQL 資料庫。
- mysqldump 是一種 MySQL 公用程式，可從 MySQL 資料庫建立傾印檔案，以供備份或遷移之用。

史詩

規劃遷移

任務	Description	所需的技能
驗證版本和引擎。	驗證來源和目標資料庫的資料庫版本和引擎。	DBA
識別硬體需求。	識別目標伺服器執行個體的硬體需求。	DBA，系統管理員
識別儲存需求。	識別儲存需求（儲存類型和容量）。	DBA，系統管理員
選擇執行個體類型。	根據您的運算、儲存和網路需求，選擇適當的執行個體類型。	DBA，系統管理員
判斷網路存取安全需求。	識別來源和目標資料庫的網路存取安全需求。	DBA，系統管理員
決定策略。	識別應用程式遷移策略。	DBA、應用程式擁有者、系統管理員

設定基礎設施

任務	Description	所需的技能
建立 Virtual Private Cloud (VPC)	如需說明，請參閱《Amazon Virtual Private Cloud (Amazon VPC) 文件》中的建立 VPC。	系統管理員
建立安全群組。	如需說明，請參閱 Amazon VPC 文件中的為您的 VPC 建立安全群組。	系統管理員

任務	Description	所需的技能
在 中設定和啟動 Aurora MySQL 相容資料庫叢集 AWS 帳戶。	如需說明，請參閱 Aurora 文件中的建立 Amazon Aurora 資料庫叢集。	系統管理員

遷移資料 - 選項 1

任務	Description	所需的技能
使用原生 MySQL 工具或第三方工具來遷移資料庫物件和資料。	如需說明，請參閱 MySQL 工具的文件，例如 mysqldbcopy 和 mysqldump。	DBA

遷移資料 - 選項 2

任務	Description	所需的技能
使用 遷移資料 AWS DMS。	如需說明，請參閱 AWS DMS 文件中的使用 MySQL 相容資料庫做為來源和使用 MySQL 相容資料庫做為目標。	DBA

遷移應用程式

任務	Description	所需的技能
遵循 策略。	遵循應用程式遷移策略。	DBA、應用程式擁有者、系統管理員

剪下

任務	Description	所需的技能
切換應用程式用戶端。	切換應用程式用戶端以連線至新的 Aurora 叢集端點。	DBA、應用程式擁有者、系統管理員

關閉專案

任務	Description	所需的技能
關閉資源。	關閉臨時 AWS 資源。	DBA，系統管理員
檢閱文件。	檢閱並驗證專案文件。	DBA、應用程式擁有者、系統管理員
收集指標。	收集遷移時間、手動步驟與工具用量的百分比、節省成本等指標。	DBA、應用程式擁有者、系統管理員
完成遷移專案。	關閉專案並提供意見回饋。	應用程式擁有者、DBA、系統管理員

相關資源

參考

- 將資料遷移至 Amazon Aurora MySQL 資料庫叢集
- AWS DMS website
- AWS DMS 文件
- Amazon Aurora 定價
- 建立並連線至 Aurora MySQL 資料庫叢集
- Amazon VPC 和 Amazon RDS
- Amazon Aurora 文件

教學課程和影片

- 入門 AWS DMS
- Amazon Aurora 入門

使用 Percona XtraBackup、Amazon EFS 和 Amazon S3 將內部部署 MySQL 資料庫遷移至 Aurora MySQL

Rohan Jamadagni、Udayasimha Theepireddy 和 Sajith menon，Amazon Web Services

總結

此模式說明如何使用 Percona XtraBackup，有效率地將大型內部部署 MySQL 資料庫遷移至 Amazon Aurora MySQL。Percona XtraBackup 是 MySQL 型伺服器的開放原始碼、非封鎖備份公用程式。模式顯示如何使用 Amazon Elastic File System (Amazon EFS) 減少將備份上傳至 Amazon Simple Storage Service (Amazon S3) 的時間，並將備份還原至 Amazon Aurora MySQL。模式也提供如何進行增量 Percona 備份的詳細資訊，以將要套用至目標 Aurora MySQL 資料庫的二進位日誌數目降至最低。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 建立 AWS Identity and Access Management (IAM) 角色和政策的許可
- 內部部署 MySQL 資料庫與 AWS 虛擬私有雲端 (VPC) 之間的網路連線

限制

- 來源伺服器必須是 Linux 系統，可以安裝網路檔案系統 (NFS) 用戶端 (nfs-utils/nfs-common)。
- 用於上傳備份檔案的 S3 儲存貯體僅支援伺服器端加密 (SSE-S3/SSE-KMS)。
- Amazon S3 會將備份檔案的大小限制為 5 TB。如果您的備份檔案超過 5 TB，您可以將其分割成多個較小的檔案。
- 上傳至 S3 儲存貯體的來源檔案數目不能超過一百萬個檔案。
- 模式僅支援 Percona XtraBackup 完整備份和增量備份。它不支援使用 `--tables`、`--tables-exclude`、`--tables-file--databases`、`--databases-exclude`、 或 的部分備份`--databases-file`。
- Aurora 不會從來源 MySQL 資料庫還原使用者、函數、預存程序或時區資訊。

產品版本

- 來源資料庫必須是 MySQL 5.5、5.6 或 5.7 版。
- 對於 MySQL 5.7，您必須使用 Percona XtraBackup 2.4。

- 對於 MySQL 5.6 和 5.6，您必須使用 Percona XtraBackup 2.3 或 2.4。

Architecture

來源技術堆疊

- Linux 作業系統
- MySQL 伺服器
- Percona XtraBackup

目標技術堆疊

- Amazon Aurora
- Amazon S3
- Amazon EFS

目標架構

工具

AWS 服務

- Amazon Aurora 是全受管關聯式資料庫引擎，可讓您輕鬆且符合成本效益地設定、操作和擴展 MySQL 部署。Aurora MySQL 是 MySQL 的下拉式選單。
- Amazon Elastic File System (Amazon EFS) 可協助您在 AWS 雲端中建立和設定共用檔案系統。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他工具

- Percona XtraBackup 是一種開放原始碼公用程式，可執行 MySQL 資料庫的串流、壓縮和增量備份，而不會中斷或封鎖資料庫。

史詩

建立 Amazon EFS 檔案系統

任務	Description	所需的技能
建立安全群組以與 Amazon EFS 掛載目標建立關聯。	在 VPC 中建立安全群組，該安全群組使用透過 AWS Transit Gateway 連接至現場部署資料庫的 VPN 連接進行設定。如需此和其他案例所述命令和步驟的詳細資訊，請參閱此模式結尾的「相關資源」一節中的連結。	AWS DevOps/資料庫管理員
編輯安全群組規則。	新增傳入規則，使用類型 NFS、連接埠 2049 和現場部署資料庫伺服器的 IP 範圍做為來源。根據預設，傳出規則允許所有流量離開。如果不是這種情況 ，請新增傳出規則以開啟 NFS 連接埠的連線。新增另外兩個傳入規則：連接埠 2049（來源：相同安全群組的安全群組 ID) 和連接埠 22（來源：IP 範圍，您將從中連線至 EC2 執行個體）。	AWS DevOps/資料庫管理員
建立檔案系統。	在掛載目標中，使用您在上一個案例中建立的 VPC 和安全群組。根據內部部署資料庫的 I/O 需求，選擇輸送量模式和效能。或者，啟用靜態加密。	AWS DevOps/資料庫管理員

掛載檔案系統

任務	Description	所需的技能
建立要與 EC2 執行個體建立關聯的 IAM 執行個體描述檔角色。	建立具有在 Amazon S3 中上傳和存取物件許可的 IAM 角色。選擇將備份儲存為政策資源的 S3 儲存貯體。	AWS DevOps
建立 EC2 執行個體。	啟動以 Linux 為基礎的 EC2 執行個體，並連接您在上一個步驟中建立的 IAM 執行個體描述檔角色，以及您先前建立的安全群組。	AWS DevOps
安裝 NFS 用戶端。	在內部部署資料庫伺服器和 EC2 執行個體上安裝 NFS 用戶端。如需安裝說明，請參閱「其他資訊」一節。	DevOps
掛載 Amazon EFS 檔案系統。	在內部部署和 EC2 執行個體上掛載 Amazon EFS 檔案系統。在每個伺服器上，建立用於存放備份的目錄，並使用掛載目標端點掛載檔案系統。如需範例，請參閱「其他資訊」一節。	DevOps

進行 MySQL 來源資料庫的備份

任務	Description	所需的技能
安裝 Percona XtraBackup。	在內部部署資料庫伺服器上安裝 Percona XtraBackup 2.3 或 2.4（取決於 MySQL 資料庫的版本）。如需安裝連結，請參閱「相關資源」一節。	資料庫管理員

任務	Description	所需的技能
計算來源資料庫中的結構描述和資料表。	收集並記下來源 MySQL 資料庫中的結構描述和物件數量。遷移後，您將使用這些計數來驗證 Aurora MySQL 資料庫。	資料庫管理員
（選用）記下來源資料庫的最新二進位日誌序列。	如果您想要在來源資料庫和 Aurora MySQL 之間建立二進位日誌複寫，以將停機時間降至最低，請執行此步驟。必須啟用 log-bin，且 server_id 必須是唯一的。請記下來源資料庫中目前的二進位日誌序列，就在啟動備份之前。如果您打算僅使用完整備份，請在完整備份之前執行此步驟。如果您打算在完整備份之後進行增量備份，請在 Aurora MySQL 資料庫執行個體上還原的最終增量備份之前執行此步驟。	資料庫管理員
啟動來源 MySQL 資料庫的完整備份。	使用 Percona XtraBackup 完整備份 MySQL 來源資料庫。如需完整和增量備份的範例命令，請參閱「其他資訊」一節。	資料庫管理員

任務	Description	所需的技能
（選用）使用 Percona XtraBackup 進行增量備份。	增量備份可用來減少您需要套用的二進位日誌數量，以便將來源資料庫與 Aurora MySQL 同步。大型和交易密集型資料庫可能會在備份期間產生大量二進位日誌。透過取得增量備份並將其儲存在共用的 Amazon EFS 檔案系統上，您可以大幅縮短備份和上傳資料庫的時間。如需詳細資訊，請參閱「其他資訊」一節。繼續進行增量備份，直到您準備好開始遷移至 Aurora。	資料庫管理員
準備備份。	在此步驟中，交易日誌會套用至備份期間進行中交易的備份。繼續將交易日誌（僅限 --apply-log-only) 套用至每個增量備份，以合併備份，但上次備份除外。如需範例，請參閱「其他資訊」一節。在此步驟之後，完整的合併備份將以 ~/<efs_mount_name>/fullbackup 顯示。	資料庫管理員
壓縮並分割最終合併的備份。	在您準備最終的合併備份之後，請使用 tar、zip 和 split 命令來建立較小的備份壓縮檔案。如需範例，請參閱「其他資訊」一節。	資料庫管理員

將備份還原至 Aurora MySQL 資料庫叢集

任務	Description	所需的技能
將備份上傳至 Amazon S3。	存放備份檔案的 Amazon EFS 檔案系統會同時掛載在現場部署資料庫和 EC2 執行個體上，因此備份檔案隨時可供 EC2 執行個體使用。使用 Secure Shell (SSH) 連線至 EC2 執行個體，並將壓縮的備份檔案上傳至新的或現有的 S3 儲存貯體；例如：aws s3 sync ~/<efs_mount_name>/fullbackup s3：//<bucket_name>/fullbackup。如需其他詳細資訊，請參閱「相關資源」一節中的連結。	AWS DevOps
為 Aurora 建立服務角色以存取 Amazon S3。	使用信任 "rds.amazonaws.com" 和政策建立 IAM 角色，讓 Aurora 存取存放備份檔案的 S3 儲存貯體。所需的許可為 ListBucket、GetObject 和 GetObjectVersion。	AWS DevOps
建立 Aurora 的聯網組態。	建立具有至少兩個可用區域的叢集資料庫子網路群組，以及允許傳出連線至來源資料庫的子網路路由表組態。建立安全群組，允許對外連線至現場部署資料庫，並允許管理員連線至 Aurora 資料庫叢集。如需詳細資訊，請參閱「相關資源」一節中的連結。	AWS DevOps/資料庫管理員
將備份還原至 Aurora MySQL 資料庫叢集。	從您上傳至 Amazon S3 的備份還原資料。指定來源資料庫	AWS DevOps/資料庫管理員

任務	Description	所需的技能
	的 MySQL 版本、提供上傳備份檔案的 S3 儲存貯體名稱和資料夾路徑字首（例如「其他資訊」區段中的範例為「完整備份」），並提供您建立的 IAM 角色，以授權 Aurora 存取 Amazon S3。	
驗證 Aurora MySQL 資料庫。	根據從來源資料庫取得的計數，驗證還原的 Aurora 資料庫叢集中的結構描述和物件計數。	資料庫管理員
設定 binlog 複寫。	在進行還原至 Aurora 資料庫叢集的最後一個備份之前，請使用您先前記下的二進位日誌序列。在來源資料庫上建立複寫使用者，並遵循「其他資訊」一節中的指示提供適當的權限、在 Aurora 上啟用複寫，並確認複寫是同步的。	AWS DevOps/資料庫管理員

相關資源

建立 Amazon EFS 檔案系統

- 建立安全群組 (Amazon VPC 文件）
- 傳輸閘道 VPN 連接 (Amazon VPC 文件）
- 使用 AWS Transit Gateway 擴展 VPN 輸送量（網路與內容交付部落格）
- 建立 Amazon EFS 檔案系統 (Amazon EFS 文件）
- 建立掛載目標 (Amazon EFS 文件）
- 加密靜態資料 (Amazon EFS 文件）

掛載檔案系統

- Amazon EC2 的 IAM 角色 (Amazon EC2 文件)
- 啟動 Amazon EC2 Linux 執行個體 (Amazon EC2 文件)
- 安裝 NFS 用戶端 (Amazon EFS 文件)
- 在內部部署用戶端上掛載 Amazon EFS 檔案系統 (Amazon EFS 文件)
- 掛載 EFS 檔案系統 (Amazon EFS 文件)

進行 MySQL 來源資料庫的備份

- 安裝 Percona XtraBackup 2.3 (Percona XtraBackup 文件)
- 安裝 Percona XtraBackup 2.4 (Percona XtraBackup 文件)
- 設定複寫主組態 (MySQL 文件)
- 將資料從外部 MySQL 資料庫遷移至 Aurora MySQL 資料庫叢集 (Aurora 文件)
- 增量備份 (Percona XtraBackup 文件)

將備份還原至 Amazon Aurora MySQL

- 建立儲存貯體 (Amazon S3 文件)
- 使用 SSH 連線至 Linux 執行個體 (Amazon Ec2 文件)
- 設定 AWS CLI (AWS CLI 文件)
- sync 命令 (AWS CLI 命令參考)
- 建立 IAM 政策以存取 Amazon S3 資源 (Aurora 文件)
- 資料庫叢集先決條件 (Aurora 文件)
- 使用資料庫子網路群組 (Aurora 文件)
- 為私有資料庫執行個體建立 VPC 安全群組 (Aurora 文件)
- 從 S3 儲存貯體還原 Aurora MySQL 資料庫叢集 (Aurora 文件)
- 使用 MySQL 或其他 Aurora 資料庫叢集設定複寫 (Aurora 文件)
- mysql.rds_set_external_master 程序 (Amazon RDS SQL 上的 MySQL 參考)
- mysql.rds_start_replication 程序 (Amazon RDS SQL 上的 MySQL 參考)

其他參考

- 將資料從外部 MySQL 資料庫遷移至 Aurora MySQL 資料庫叢集 (Aurora 文件)

- MySQL 伺服器下載 (Oracle 網站)

教學課程和影片

- 使用 Amazon S3 將 MySQL 資料遷移至 Aurora MySQL 資料庫叢集 (AWS 知識中心)
- Amazon EFS 設定和掛載 （影片 ）

其他資訊

安裝 NFS 用戶端

- 如果您使用的是 Red Hat 或類似的 Linux 作業系統，請使用 命令：

```
$ sudo yum -y install nfs-utils
```

- 如果您使用的是 Ubuntu 或類似的 Linux 作業系統，請使用 命令：

```
$ sudo apt-get -y install nfs-common
```

如需詳細資訊，請參閱 Amazon EFS 文件中的逐步解說。

掛載 Amazon EFS 檔案系統

使用 命令：

```
mkdir ~/<efs_mount_name>
$ sudo mount -t nfs -o
 nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport mount-
target-IP:/ ~/<efs_mount_name>
```

如需詳細資訊，請參閱 Amazon EFS 文件中的逐步解說和掛載 EFS 檔案系統。 EFS

進行 MySQL 來源資料庫的備份

完整備份

使用類似下列的命令，它會進行備份、壓縮，並將其分割成每個 1 GB 的較小區塊：

```
xtrabackup --backup --user=dbuser --password=<password> --binlog-info=AUTO --stream=tar
 --target-dir=~/<efs_mount_name>/fullbackup | gzip - | split -d --bytes=1024MB - ~/
<efs_mount_name>/fullbackup/backup.tar.gz &
```

如果您打算在完整備份之後進行後續增量備份，請勿壓縮和分割備份。請改用類似以下的命令：

```
xtrabackup --backup --user=dbuser --password=<password> --target-dir=~/
<efs_mount_name>/fullbackup/
```

增量備份

使用 `--incremental-basedir` 參數的完整備份路徑；例如：

```
xtrabackup --backup --user=dbuser --password=<password> --target-dir=~/
<efs_mount_name>/incremental/backupdate --incremental-basedir=~/<efs_mount_name>/
fullbackup
```

其中 basedir 是完整備份和 xtrabackup_checkpoints 檔案的路徑。

如需進行備份的詳細資訊，請參閱 Aurora 文件中的將資料從外部 MySQL 資料庫遷移至 Amazon Aurora MySQL 資料庫叢集。

準備備份

若要準備完整備份：

```
xtrabackup --prepare --apply-log-only --target-dir=~/<efs_mount_name>/fullbackup
```

若要準備增量備份：

```
xtrabackup --prepare --apply-log-only --target-dir=~/<efs_mount_name>/fullbackup --
incremental-dir=~/<efs_mount_name>/incremental/06062020
```

若要準備最終備份：

```
xtrabackup --prepare --target-dir=~/<efs_mount_name>/fullbackup --incremental-dir=~/
<efs_mount_name>/incremental/06072020
```

如需詳細資訊，請參閱 Percona XtraBackup 文件中的增量備份。

壓縮和分割合併的備份

若要壓縮位於 ~/<efs_mount_name>/fullbackup 的合併備份：

```
tar -zcvf <backupfilename.tar.gz> ~/<efs_mount_name>/fullbackup
```

若要分割備份：

```
split -d -b1024M --verbose  <backupfilename.tar.gz> <backupfilename.tar.gz>
```

設定 binlog 複寫

若要在來源資料庫上建立複寫使用者並提供適當的權限：

```
CREATE USER 'repl_user'@'' IDENTIFIED BY ''; GRANT REPLICATION CLIENT, REPLICATION
 SLAVE ON *.* TO 'repl_user'@'';
```

若要透過連線至 Aurora 資料庫叢集在 Aurora 上啟用複寫，請在資料庫叢集參數群組中啟用二進位日誌。Set `binlog_format = mixed`（偏好混合模式）。此變更需要您重新啟動執行個體才能套用更新。

```
CALL mysql.rds_set_external_master ('sourcedbinstanceIP', sourcedbport, 'repl_user',
 '', 'binlog_file_name', binlog_file_position, 0); CALL mysql.rds_start_replication;
```

若要確認複寫是同步的：

```
SHOW Slave Status \G;
```

主欄位後面的秒數會顯示 Aurora 與內部部署資料庫的距離。

使用 AWS App2Container 將內部部署 Java 應用程式遷移至 AWS

Dhananjay Karanjkar，Amazon Web Services

總結

AWS App2Container (A2C) 是一種命令列工具，可協助將虛擬機器中執行的現有應用程式轉換為容器，而不需要任何程式碼變更。A2C 會探索在伺服器上執行的應用程式、識別相依性，並產生相關成品，以便無縫部署至 Amazon Elastic Container Service (Amazon ECS) 和 Amazon Elastic Kubernetes Service (Amazon EKS)。

此模式提供透過工作者機器使用 App2Container 將部署在應用程式伺服器上的內部部署 Java 應用程式遠端遷移至 AWS Fargate 或 Amazon EKS 的步驟。

工作者機器可用於下列使用案例：

- 執行 Java 應用程式的應用程式伺服器上不允許或無法使用 Docker 安裝。
- 您必須管理部署在不同實體或虛擬伺服器上的多個應用程式的遷移。

此模式使用 AWS CodeCommit AWS CodePipeline、 和 AWS CodeBuild。

先決條件和限制

先決條件

- 在 Linux 伺服器上執行 Java 應用程式的應用程式伺服器
- 具有 Linux 作業系統的工作者機器
- 具有至少 20 GB 可用磁碟空間的工作者機器

限制

- 並非所有應用程式都受到支援。如需詳細資訊，請參閱 Linux 支援的應用程式。

Architecture

來源技術堆疊

- 在 Linux 伺服器上執行的 Java 應用程式

目標技術堆疊

- AWS CodeBuild
- AWS CodeCommit
- AWS CodeDeploy
- AWS CodePipeline
- Amazon Elastic Container Registry
- AWS Fargate

目標架構

工具

工具

- AWS App2Container – AWS App2Container (A2C) 是一種命令列工具，可協助您提升和轉移在內部部署資料中心或虛擬機器中執行的應用程式，使其在由 Amazon ECS 或 Amazon EKS 管理的容器中執行。
- AWS CodeBuild – AWS CodeBuild 是雲端中全受管的建置服務。CodeBuild 可編譯原始碼、執行單元測試，並產生可立即部署的成品。
- AWS CodeCommit – AWS CodeCommit 是由 Amazon Web Services 託管的版本控制服務，可用來在雲端中私下存放和管理資產 （例如文件、原始程式碼和二進位檔案）。
- AWS CodePipeline – AWS CodePipeline 是一種持續交付服務，可用來建立模型、視覺化和自動化發行軟體所需的步驟。
- Amazon ECS – Amazon Elastic Container Service (Amazon ECS) 是一種高度可擴展的快速容器管理服務，用於執行、停止和管理叢集上的容器。
- Amazon ECR – Amazon Elastic Container Registry (Amazon ECR) 是一種 AWS 受管容器映像登錄服務，安全、可擴展且可靠。
- Amazon EKS – Amazon Elastic Kubernetes Service (Amazon EKS) 是一項受管服務，可讓您在 AWS 上執行 Kubernetes，而無需安裝、操作和維護您自己的 Kubernetes 控制平面或節點。
- AWS Fargate – AWS Fargate 是一種技術，您可以與 Amazon ECS 搭配使用來執行容器，而無需管理 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的伺服器或叢集。使用 Fargate，就不再需要佈建、設定或擴展虛擬機器的叢集來執行容器。

史詩

設定登入資料

任務	Description	所需的技能
建立秘密以存取應用程式伺服器。	若要從工作者機器遠端存取應用程式伺服器，請在 AWS Secrets Manager 中建立秘密。對於您的秘密，您可以使用 SSH 私有金鑰或憑證和 SSH 私有金鑰。如需詳細資訊，請參閱管理 AWS App2Container 的秘密。	DevOps、開發人員

設定工作者機器

任務	Description	所需的技能
安裝 tar 檔案。	執行 `sudo yum install -y tar`。	DevOps、開發人員
安裝 AWS CLI。	若要安裝 Amazon Command Line Interface (AWS CLI)，請執行 `curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"` 。 解壓縮 `awscliv2.zip` 。 執行 `sudo ./aws/install` 。	DevOps、開發人員
安裝 App2Container。	執行下列命令：	DevOps、開發人員

任務	Description	所需的技能

	`curl -o AWSApp2Container-installer-linux.tar.gz https://app2container-release-us-east-1.s3.us-east-1.amazonaws.com/latest/linux/AWSApp2Container-installer-linux.tar.gz`
	
	`sudo tar xvf AWSApp2Container-installer-linux.tar.gz`
	
	`sudo ./install.sh`	

設定設定檔。	若要設定 AWS 預設設定檔，請執行 `sudo aws configure` 。 若要設定名為 AWS 的預設設定檔，請執行 `sudo aws configure --profile <profile name>`。	DevOps、開發人員

安裝 Docker.	執行下列命令。
	
	`sudo yum install -y docker`
	
	`sudo systemctl enable docker & sudo systemctl restart docker`	

任務	Description	所需的技能
初始化 App2Container。	若要初始化 App2Container，您需要以下資訊： • `workspace` ：存放應用程式容器化成品。我們建議提供至少具有 20 GB 可用磁碟空間的目錄路徑。 • `awsProfile` ：在伺服器上設定的 AWS 設定檔。這是將成品上傳到 Amazon S3、執行 `containerize` 命令，以及產生 AWS 成品以在 Amazon ECS 或 Amazon EKS 上部署的必要項目。 • `s3Bucket`：擷取和存放 AWS 成品。 • `metricsReportPermission` ：收集和存放報告的指標。 • `dockerContentTrust` ：簽署 Docker 映像。 執行 `sudo app2container init`。	DevOps、開發人員

設定工作者機器

任務	Description	所需的技能
將工作者機器設定為遠端連線，並在應用程式伺服器上執行 App2Container 命令。	若要設定工作者機器，需要下列資訊：	DevOps、開發人員

任務	Description	所需的技能
	• `Server FQDN`：應用程式伺服器的完整網域名稱。 • `Server IP address`：應用程式伺服器的 IP 地址。FQDN 或 IP 地址已足夠。 • `SecretARN` ：用來連線至應用程式伺服器的秘密 Amazon Resource Name (ARN)，並存放在 Secrets Manager 中。 • `AuthMethod` ： `key`或`cert`身分驗證方法。 執行 `sudo app2container remote configure` 。	

在工作者機器上探索、分析和擷取應用程式

任務	Description	所需的技能
探索內部部署 Java 應用程式。	若要遠端探索應用程式伺服器上執行的所有應用程式，請執行下列命令。 `sudo app2container remote inventory --target <FQDN/IP of App server>` 此命令會在 中產生已部署應用程式的清單`inventory.json` 。	DevOps 開發人員 DevOps

任務	Description	所需的技能
分析探索到的應用程式。	若要使用庫存階段中取得的應用程式 ID 遠端分析每個應用程式，請執行下列命令。 `sudo app2container remote analyze --application-id <java-app-id> --target <FQDN/IP of App Server>` 這會在工作區位置產生`analysis.json` 檔案。產生此檔案後，您可以根據您的需求修改容器化參數。	DevOps 開發人員 DevOps
擷取已分析的應用程式。	若要為分析的應用程式產生應用程式封存，請遠端執行下列命令，這會在工作區位置產生 tar 套件。 `sudo app2container remote extract --application-id <application id> --target <FQDN/IP of App Server>` 擷取的成品可以在本機工作者機器上產生。	DevOps 開發人員 DevOps

在工作者機器上容器化擷取的成品

任務	Description	所需的技能
容器化擷取的成品。	執行下列命令，將上一個步驟中擷取的成品容器化。 `sudo app2container containerize --input-archive <tar bundle location on worker machine>`	DevOps 開發人員 DevOps
完成目標。	若要完成目標，請開啟 `deployment.json` 在 `containerize` 命令執行時建立的。若要指定 AWS Fargate 做為目標，請將 `createEcsArtifacts` 設定為 `true`。若要將 Amazon EKS 設定為目標，請將 `createEksArtifacts` 設為 `true`。	DevOps 開發人員 DevOps

產生和佈建 AWS 成品

任務	Description	所需的技能
在工作者機器上產生 AWS 部署成品。	若要產生部署成品，請執行下列命令。 `sudo app2container generate app-deployment --application-id <application id>`	DevOps

任務	Description	所需的技能
	這會在工作區中產生 `ecs-master.yml` AWS CloudFormation 範本。	
佈建成品。	若要進一步佈建產生的成品，請執行下列命令來部署 AWS CloudFormation 範本。 `aws cloudformation deploy --template-file <path to ecs-master.yml> --capabilities CAPABILITY_NAMED_IAM --stack-name <application id>-ECS`	DevOps
產生管道。	根據您的需求修改在上一個故事中建立`pipeline.json` 的 。然後執行`generate pipeline`命令來產生管道部署成品。	DevOps

相關資源

- 什麼是 App2Container？
- AWS App2Container 部落格文章
- AWS CLI 組態基本概念
- Amazon ECS 的 Docker 基本概念
- Docker 命令

在 AWS 大型遷移中遷移共用檔案系統

Amit Rudraraju、Sam Apa、Bheemeswararao Balla、Wally Lu 和 Sanjeev Prakasam、Amazon Web Services

總結

遷移 300 個以上的伺服器會被視為大型遷移。大型遷移的目的是將工作負載從現有的現場部署資料中心遷移到 AWS 雲端，這些專案通常專注於應用程式和資料庫工作負載。不過，共用檔案系統需要集中注意力和單獨的遷移計畫。此模式說明共用檔案系統的遷移程序，並提供在大型遷移專案中成功遷移它們的最佳實務。

共用檔案系統 (SFS) 也稱為網路或叢集檔案系統，是掛載到多個伺服器的檔案共用。共用檔案系統可透過網路檔案系統 (NFS)、通用網際網路檔案系統 (CIFS) 或伺服器訊息區塊 (SMB) 等通訊協定存取。

這些系統不會使用標準遷移工具進行遷移，例如， AWS Application Migration Service 因為它們既不專用於要遷移的主機，也不表示為區塊型設備。雖然大多數主機相依性都是透明遷移的，但相依檔案系統的協調和管理必須分別處理。

您會在下列階段遷移共用檔案系統：探索、規劃、準備、切換和驗證。使用此模式和連接的手冊，您可以將共用檔案系統遷移至 AWS 儲存服務，例如 Amazon Elastic File System (Amazon EFS)、Amazon FSx for NetApp ONTAP 或 Amazon FSx for Windows File Server。若要傳輸檔案系統，您可以使用 AWS DataSync 或第三方工具，例如 NetApp SnapMirror。

> **Note**
>
> 此模式是有關大型遷移到 AWS 雲端 AWS 的規範性指引系列的一部分。此模式包含將 SFSs 整合到伺服器的波浪計劃的最佳實務和指示。如果您要在大型遷移專案之外遷移一或多個共用檔案系統，請參閱 Amazon EFS、Amazon FSx for Windows File Server 和 Amazon FSx for NetApp ONTAP AWS 文件中的資料傳輸說明。

先決條件和限制

先決條件

先決條件可能會根據您的來源和目標共用檔案系統和您的使用案例而有所不同。以下是最常見的：

- 作用中 AWS 帳戶。
- 您已完成大型遷移專案的應用程式產品組合探索，並開始開發波動計畫。如需詳細資訊，請參閱AWS 大型遷移的產品組合手冊。

- 虛擬私有雲端 (VPCs) 和安全群組，允許內部部署資料中心與您的 AWS 環境之間的輸入和輸出流量。如需詳細資訊，請參閱網路對 Amazon VPC 連線選項和AWS DataSync 網路需求。
- 建立 AWS CloudFormation 堆疊的許可或建立 Amazon EFS 或 Amazon FSx 資源的許可。如需詳細資訊，請參閱 CloudFormation 文件、Amazon EFS 文件或 Amazon FSx 文件。
- 如果您使用 AWS DataSync 執行遷移，則需要下列許可：
  - AWS DataSync 將日誌傳送至 Amazon CloudWatch Logs 日誌群組的許可。如需詳細資訊，請參閱允許 DataSync 將日誌上傳至 CloudWatch 日誌群組。
  - 存取 CloudWatch Logs 日誌群組的許可。如需詳細資訊，請參閱管理 CloudWatch Logs 資源存取許可的概觀。
  - 在 DataSync 中建立客服人員和任務的許可。如需詳細資訊，請參閱使用 所需的 IAM 許可 AWS DataSync。

限制

- 此模式旨在將 SFSs 遷移為大型遷移專案的一部分。它包含將 SFSs 併入您遷移應用程式的波浪計劃的最佳實務和說明。如果您要在大型遷移專案之外遷移一或多個共用檔案系統，請參閱 Amazon EFS、Amazon FSx for Windows File Server 和 Amazon FSx for NetApp ONTAP AWS 文件中的資料傳輸說明。
- 此模式是以常用的架構、服務和遷移模式為基礎。不過，大型遷移專案和策略可能因組織而異。您可能需要根據您的需求自訂此解決方案或提供的手冊。

Architecture

來源技術堆疊

下列一個或多個：

- Linux (NFS) 檔案伺服器
- Windows (SMB) 檔案伺服器
- NetApp 儲存陣列
- Dell EMC Isilon 儲存陣列

目標技術堆疊

下列一個或多個：

- Amazon Elastic File System
- Amazon FSx for NetApp ONTAP
- Amazon FSx for Windows File Server

目標架構

圖表顯示下列程序：

1. 您可以使用 AWS Direct Connect 或 AWS 服務 等 AWS 雲端 ，在現場部署資料中心與 之間建立連線 AWS Site-to-Site VPN。
2. 您可以在內部部署資料中心安裝 DataSync 代理程式。
3. 根據您的波動計畫，您可以使用 DataSync 將來源共用檔案系統的資料複寫至目標 AWS 檔案共用。

遷移階段

下圖顯示在大型遷移專案中遷移 SFS 的階段和高階步驟。

此模式的 Epics 區段包含有關如何完成遷移和使用連接工作手冊的詳細說明。以下是此分階段方法中步驟的高階概觀。

階段	步驟
探索	1. 使用探索工具，您可以收集共用檔案系統的資料，包括伺服器、掛載點和 IP 地址。 2. 使用組態管理資料庫 (CMDB) 或遷移工具，您可以收集伺服器的詳細資訊，包括遷移波動、環境、應用程式擁有者、IT 服務管理 (ITSM) 服務名稱、組織單位和應用程式 ID 的相關資訊。
計畫	3. 使用 SFSs和伺服器所收集的資訊，建立 SFS 波動計畫。 4. 使用建置工作表中的資訊，為每個 SFS 選擇目標 AWS 服務 和遷移工具。

準備	5. 在 Amazon EFS、Amazon FSx for NetApp ONTAP 或 Amazon FSx for Windows File Server 中設定目標基礎設施。
	6. 設定資料傳輸服務，例如 DataSync，然後啟動初始資料同步。當初始同步完成時，您可以設定重複發生的同步以排程執行，
	7. 使用目標檔案共享的相關資訊更新 SFS 波動計畫，例如 IP 地址或路徑。
剪下	8. 停止主動存取來源 SFS 的應用程式。
	9. 在資料傳輸服務中，執行最終資料同步。
	10. 同步完成時，請檢閱 CloudWatch Logs 中的日誌資料，以驗證是否完全成功。
驗證	11. 在伺服器上，將掛載點變更為新的 SFS 路徑。
	12. 重新啟動並驗證應用程式。

工具

AWS 服務

- Amazon CloudWatch Logs 可協助您集中所有系統、應用程式的日誌， AWS 服務 以便您可以監控日誌並將其安全地存檔。
- AWS DataSync 是一種線上資料傳輸和探索服務，可協助您在 AWS 儲存服務之間來回移動檔案或物件資料。
- Amazon Elastic File System (Amazon EFS) 協助您在 AWS 雲端中建立和設定共用檔案系統。
- Amazon FSx 提供檔案系統，可支援業界標準的連線通訊協定，並提供高可用性和跨系統複寫 AWS 區域。

其他工具

- SnapMirror 是一種 NetApp 資料複寫工具，可將資料分別從指定的來源磁碟區或樹狀目錄複寫到目標磁碟區或樹狀目錄。您可以使用此工具將 NetApp 來源檔案系統遷移至 Amazon FSx for NetApp ONTAP。
- Robocopy 短於強式檔案複製，是 Windows 的命令列目錄和命令。您可以使用此工具將 Windows 來源檔案系統遷移至 Amazon FSx for Windows File Server。

最佳實務

波規劃方法

為您的大型遷移專案規劃波浪時，請考慮延遲和應用程式效能。當 SFS 和相依應用程式在不同位置操作時，例如一個在雲端，另一個在內部部署資料中心，這可能會增加延遲並影響應用程式效能。以下是建立波動計畫時的可用選項：

1. 在同一波內遷移 SFS 和所有相依伺服器 – 此方法可防止效能問題，並將重做降至最低，例如多次重新設定掛載點。當應用程式和 SFS 之間需要非常低的延遲時，建議使用此選項。不過，波動規劃很複雜，目標是從相依性分組中移除變數，而不是新增到它們。此外，如果許多伺服器存取相同的 SFS，則不建議使用此方法，因為它會使波動過大。
2. 在遷移最後一個相依伺服器之後遷移 SFS – 例如，如果多個伺服器存取 SFS，且這些伺服器排程在波 4、6 和 7 中遷移，請排程 SFS 在波 7 中遷移。

   這種方法通常是大型遷移最符合邏輯的方法，建議用於對延遲敏感的應用程式。它可降低與資料傳輸相關的成本。它也可以將 SFS 和更高層級 （例如生產） 應用程式之間的延遲期間降至最低，因為在開發和 QA 應用程式之後，更高層級的應用程式通常會排程為最後遷移。

   不過，這種方法仍然需要探索、規劃和敏捷性。您可能需要在較早的波動中遷移 SFS。確認應用程式可以在第一個相依波與包含 SFS 的波之間的時間內承受額外的延遲。與應用程式擁有者進行探索工作階段，並將應用程式遷移至對延遲最敏感的應用程式。如果在遷移相依應用程式後發現效能問題，請準備好盡快進行樞紐，以盡快遷移 SFS。
3. 在大型遷移專案結束時遷移 SFS – 如果延遲不是一個因素，例如當 SFS 中的資料不常存取或不對應用程式效能至關重要時，建議使用此方法。此方法可簡化遷移，並簡化切換任務。

您可以根據應用程式的延遲敏感度來混合這些方法。例如，您可以使用方法 1 或 2 遷移對延遲敏感 SFSs，然後使用方法 3 遷移其餘SFSs。

選擇 AWS 檔案系統服務

AWS 為檔案儲存提供數種雲端服務。每個 都為效能、擴展、可存取性、整合、合規和成本最佳化提供不同的優點和限制。有一些邏輯預設選項。例如，如果您目前的現場部署檔案系統正在操作 Windows Server，則 Amazon FSx for Windows File Server 是預設選項。或者，如果內部部署檔案系統正在操作 NetApp ONTAP，則 Amazon FSx for NetApp ONTAP 是預設選項。不過，您可以根據您的應用程式需求或實現其他雲端操作優勢，選擇目標服務。如需詳細資訊，請參閱為您的部署選擇正確的 AWS 檔案儲存服務AWS (Summit presentation)。

選擇遷移工具

Amazon EFS 和 Amazon FSx 支援使用 AWS DataSync 將共用檔案系統遷移至 AWS 雲端。如需支援的儲存系統和服務、優點和使用案例的詳細資訊，請參閱什麼是 AWS DataSync 。如需使用 DataSync 傳輸檔案的程序概觀，請參閱AWS DataSync 傳輸的運作方式。

也有數種可用的第三方工具，包括下列項目：

- 如果您選擇 Amazon FSx for NetApp ONTAP，您可以使用 NetApp SnapMirror 將檔案從內部部署資料中心遷移至雲端。SnapMirror 使用區塊層級複寫，可以比 DataSync 更快，並減少資料傳輸程序的持續時間。如需詳細資訊，請參閱使用 NetApp SnapMirror 遷移至 FSx for ONTAP。
- 如果您選擇 Amazon FSx for Windows File Server，您可以使用 Robocopy 將檔案遷移至雲端。如需詳細資訊，請參閱使用 Robocopy 將現有檔案遷移至 FSx for Windows File Server。

史詩

探索

任務	Description	所需的技能
準備 SFS 探索工作手冊。	1. 在此模式的附件區段中下載工作手冊。這包含兩個檔案：SFS-Discovery-Workbook.xlsx 和 SFS-Wave-Plan-Workbook.xlsx。 2. 在 Microsoft Excel 中開啟 SFS-Discovery-Workbook 檔案。 3. 在儀表板工作表上，執行下列動作：	遷移工程師，遷移負責人

任務	Description	所需的技能
	• 在資料欄 A 中，更新環境名稱。 • 在資料欄 B 中，更新環境的順序，將它們按最低 (1) 優先順序排列為最高優先順序。 • 在資料欄 D–E 中，更新波動排程。 • 在資料欄 C 和 K 中，更新 AWS 帳戶名稱。 • 在資料欄 L 中，更新 VPC IDs。 • 在資料欄 M-O 中，更新子網路 IDs。 4. 檢閱手冊範本的其餘部分，並更新組織或使用案例所需的任何其他值。 5. 儲存工作手冊。	

任務	Description	所需的技能
收集來源 SFS 的相關資訊。	1. 使用您偏好的探索工具，識別所有適用儲存裝置、Linux 伺服器和 Windows 伺服器的所有 SFS 掛載。一般而言，您需要收集下列資訊： • 用戶端裝置 • 用戶端 IP 地址 • SFS 詳細資訊 • 掛載點 **Note** 您可以將掛載點詳細資訊新增至遷移執行手冊，以便在遷移後重新掛載 SFS。 2. 開啟 SFS-Discovery-Workbook 檔案。 3. 在 Wave-Sheet 工作表上，執行下列動作： • 在伺服器位置 (D) 欄中的公式中，確認現場部署來源的 CIDR 範圍格式適用於您的範圍。例如，如果您的 CIDR 範圍是 `10.0.0.0/8` ，請輸入 `10.*.*.*`。 • 在 SFS location(E) 欄中，在公式中，確認目標 VPC 的 CIDR 範圍格式適用於您的範圍。例如，	遷移工程師，遷移負責人

任務	Description	所需的技能
	如果您的 CIDR 範圍是 `176.16.0.0/16` ，請輸入 `176.16.*.*` 。 4. 在 SFS-Data 工作表上，執行下列動作： • 在伺服器名稱 (A) 欄中，輸入掛載 SFS 的伺服器名稱。 • 在 SFS 路徑 (B) 欄中，輸入 SFS 的名稱。 • 在 IP address(C) 欄中，輸入伺服器的 IP 地址。 • 新增您在探索期間收集的任何其他相關資訊，例如掛載點和 SFS 大小。您可以稍後使用此資料來修改波動規劃計算。 5. 儲存工作手冊。	

任務	Description	所需的技能
收集伺服器的相關資訊。	1. 使用您的 CMDB 或遷移工具中記錄的資料，識別具有 SFS 掛載之伺服器的下列所有相關資訊： • 伺服器名稱 • IP 位址 • 波浪 • 組織單位 (OU) • 伺服器環境，例如 DEV、QA或 PR0D • Application name (應用程式名稱) • 應用程式擁有者和聯絡資訊 2. 開啟 SFS-Discovery-Workbook 檔案。 3. 在伺服器資料工作表的 columnA–H 中，輸入您收集的來源伺服器相關資訊。注意下列事項： • 在 Wave #(C) 欄中，輸入波動名稱 （例如 Wave1)、out-of-scope(00S) 或 Retire。 • 如果應用程式擁有者聯絡人 (H) 欄，請確認電子郵件地址是否正確。此電子郵件地址會根據您在應用程式擁有者 (G) 欄中提供的名稱自動產生。如有必要，請手動更新值以反映正確的電子郵件地址。	遷移工程師，遷移負責人

任務	Description	所需的技能
	• 請勿修改包含公式的資料欄 I-J。 4. 儲存工作手冊。	

計畫

任務	Description	所需的技能
建置 SFS 波動計畫。	1. 開啟 SFS-Discovery-Workbook 檔案。 2. 確認探索階段中收集的所有資訊都是準確且最新的。 3. 在 Wave-Sheet 工作表上，篩選值 上的 SFS 波 (K) 欄1。這是第一波中所有 SFSs 的清單。 **Note** 0 此欄中的值 表示 SFS 超出遷移範圍。這可能是因為 SFS 已在 上託管，AWS 或因為存取共用的伺服器超出遷移範圍。 4. 確認您想要在此波中遷移這些 SFSs。如需如何將 SFSs 指派給波浪的詳細資訊，請參閱最佳實務區段中的波浪規劃方法。	組建領導、Cutover 領導、遷移工程師、遷移領導

任務	Description	所需的技能
	5. 選取並複製包含篩選值的儲存格。請勿複製包含資料欄標題的標頭列。 6. 開啟您先前下載的 SFS-Wave-Plan-Workbook 檔案。 7. 在Export-from-Discovery工作表上，選取儲存格 A2。 8. 貼上複製的資料。 9. 儲存 SFS-Discovery-Workbook 和 SFS-Wave-Plan-Workbook 檔案。	

任務	Description	所需的技能
選擇目標 AWS 服務 和遷移工具。	1. 在 SFS-Wave-Plan-Workbook 檔案中，從Exported-from-Discovery工作表上，選取並複製舊路徑 (C) 欄中的值。 2. 在 Build-Wave 工作表上，選取 cellA2。 3. 貼上複製的資料。此工作表中的資料欄 B-M 會自動更新，以反映與此路徑相關聯的其他資料。 4. 移除 columnA 中的任何重複值。 如需說明，請參閱移除重複值 (Microsoft Support 網站）。 5. 在目標模式或服務 (F) 欄中，檢閱建議的目標 AWS 服務 並視需要更新。如需詳細資訊，請參閱此模式最佳實務區段中的選擇 AWS 檔案系統服務。 6. 在遷移方法 (G) 欄中，檢閱建議的遷移工具並視需要更新。如需詳細資訊，請參閱此模式最佳實務區段中的選擇遷移工具。 7. 儲存 SFS-Discovery-Workbook 檔案。您已完成為此波動建立波動計畫。 8. 重複這些指示，為每個波動準備波動計畫。由於波動計畫在遷移期間可能會有所變	遷移工程師，遷移負責人

任務	Description	所需的技能
	更，我們建議您事先規劃不超過 5 個波。	

準備

任務	Description	所需的技能
設定目標檔案系統。	根據您的波動計畫中記錄的詳細資訊，在目標、 AWS 帳戶 VPC 和子網路中設定目標檔案系統。如需說明，請參閱下列 AWS 文件： • Amazon EFS • Amazon FSx for NetApp ONTAP • Amazon FSx for Windows File Server	遷移工程師、遷移負責人、AWS 管理員
設定遷移工具和傳輸資料。	1. 如果您使用的是 AWS DataSync，請設定 DataSync 任務的記錄。如需說明，請參閱記錄您的 AWS DataSync 任務活動。 2. 設定遷移工具，並根據所選工具的指示執行初始資料傳輸： • 對於 Amazon EFS，請參閱下列內容： • 使用 將檔案傳輸至 Amazon EFS AWS DataSync	AWS 管理員、雲端管理員、遷移工程師、遷移主管

任務	Description	所需的技能

	• 如需 Amazon FSx for NetApp ONTAP，請參閱下列內容：
	  • 使用 NetApp SnapMirror 遷移至 FSx for ONTAP
	  • 使用 遷移至 FSx for ONTAP AWS DataSync
	• 如需 Amazon FSx for Windows File Server，請參閱下列內容：
	  • 使用 將現有檔案遷移至 FSx for Windows File Server AWS DataSync
	  • 使用 Robocopy 將現有檔案遷移至 FSx for Windows File Server
	3. 來源 SFS 的變更可能會在初始傳輸期間或之後發生。設定來源和目標檔案系統之間的重複資料傳輸，以保持資料同步：
	• 如果您使用的是 DataSync，請參閱排程 AWS DataSync 任務。DataSync 只會傳輸來源 SFS 中修改過的檔案或新檔案。
	• 如果您使用的是第三方工具，請參閱所選工具的文件。	

任務	Description	所需的技能
更新波動計畫。	1. 開啟目前波動的 SFS-Wave-Plan-Workbook 檔案。 2. 在 Build-Wave 工作表的新路徑 IP 地址 (N) 欄中，輸入目標檔案系統的 IP 地址。執行下列其中一項操作來尋找 IP 地址： • 對於 FSx for Windows File Server，在 Amazon FSx 主控台上，選擇檔案系統，選擇您的檔案系統，然後檢視網路與安全區段。 • 對於 FSx for ONTAP，請參閱掛載磁碟區。 • 對於 Amazon EFS，請參閱使用 IP 地址掛載。 3. 在新增路徑 (O) 欄中，輸入新的掛載路徑。掛載路徑是檔案系統的 DNS 名稱。執行下列其中一項來尋找掛載路徑： • 對於 FSx for Windows File Server，在 Amazon FSx 主控台上，選擇檔案系統，選擇您的檔案系統，然後選擇連接。 • 如需 FSx for ONTAP，請參閱檔案系統詳細資訊頁面。如需說明，請參閱掛載磁碟區。 • 對於 Amazon EFS，請參閱收集資訊。	遷移工程師，遷移負責人

任務	Description	所需的技能
	4. 在 Remount-Summary 工作表上，確認新路徑 (C) 和新路徑 IP 地址 (D) 資料欄反映更新的值。 5. 確認您的組織已準備好在切換後重新掛載 Linux 和 Windows 檔案系統的 Runbook。如需一般說明，請參閱下列內容： • 安裝 Amazon EFS 檔案系統 • 存取 FSx for Windows File Server 檔案共享 • 為 ONTAP 磁碟區掛載 FSx 6. 如果此波未包含任何相依伺服器，請將它們記錄在 App-Team-Communication 工作表中。通知各自的應用程式或伺服器擁有者，因為它們可能不會包含在標準波通訊中。 7. 如果 SFSs在完成波動計畫後從波動中移除，請在 Descoped 工作表上追蹤這些 SFS。	

切換

任務	Description	所需的技能
停止應用程式。	如果應用程式或用戶端正在來源 SFS 中主動執行讀取和寫	應用程式擁有者、應用程式開發人員

任務	Description	所需的技能
	入操作，請在執行最終資料同步之前停止它們。如需說明，請參閱應用程式文件或停止讀取和寫入活動的內部程序。例如，請參閱啟動或停止 Web 伺服器 (IIS 8) (Microsoft 文件）或使用 systemctl 管理系統服務 (Red Hat 文件）。	
執行最終資料傳輸。	1. 在遷移工具中，手動執行最終資料傳輸任務，以同步目標檔案系統與來源 SFS。如需說明，請參閱啟動 DataSync 任務或參閱所選第三方遷移工具的文件。 2. 等待資料傳輸任務完成。如需詳細資訊，請參閱使用 Amazon CloudWatch 監控 AWS DataSync 活動和從命令列監控 DataSync 任務。	遷移工程師，遷移負責人

任務	Description	所需的技能
驗證資料傳輸。	如果您使用的是 AWS DataSync，請執行下列動作來驗證已成功完成的最終資料傳輸： 1. 在 AWS DataSync 主控台中，記下任務和執行 ID，例如 `task-0000-exec-1111` 。 2. 導覽至 DataSync 任務的任務記錄區段。 3. 選擇 CloudWatch 日誌群組連結。 4. 在 日誌中，搜尋任務和執行 ID。 5. 請記下任何傳輸錯誤。如需詳細資訊，請參閱 DataSync 文件中的常見錯誤。 6. 驗證下列項目： • 比較來源和目標 SFSs的檔案清單，以確認所有資料都已傳輸 • 比較來源和目標 SFSs之間的檔案存取許可。 如果您使用第三方工具，請參閱所選遷移工具文件中的資料傳輸驗證說明。	遷移工程師，遷移負責人

驗證

任務	Description	所需的技能
重新掛載檔案系統並驗證應用程式函數和效能。	1. 如果相依伺服器在此波次中遷移，請在 SFS-Wave-Plan-Workbook 檔案中的 Remount-Summary 工作表上，在新增伺服器 IP 地址 (F) 欄中輸入伺服器的新 IP 地址。 2. 在所有伺服器上，將檔案系統的掛載點從舊路徑更新為新路徑。使用組織的 Runbook 來重新掛載先前在準備階段中討論的內容。 3. 檢查掛載並確認檔案是否存在，確認檔案系統已正確掛載且可存取。基礎設施團隊通常會執行這些活動。 4. 重新啟動應用程式，並視需要讓應用程式擁有者或 QA 團隊參與，以完成應用程式的功能和效能測試。	AWS 系統管理員、應用程式擁有者

疑難排解

問題	解決方案
Microsoft Excel 中的儲存格值不會更新。	拖曳填充控點，複製範例資料列中的公式。如需詳細資訊，請參閱 Windows 或 Mac 的說明 (Microsoft Support 網站）。

相關資源

AWS 文件

- AWS DataSync 文件
- Amazon EFS 文件
- Amazon FSx 文件
- 大型遷移至 AWS 雲端
  - AWS 大型遷移指南
  - AWS 大型遷移的產品組合手冊

疑難排解

- 故障診斷 AWS DataSync 問題
- Amazon EFS 故障診斷
- Amazon FSx for Windows File Server 故障診斷
- 對 Amazon FSx for NetApp ONTAP 進行故障診斷

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Oracle GoldenGate 平面檔案轉接器，將 Oracle 資料庫遷移至 Amazon RDS for Oracle

Dhairya Jindani 和 Baji Shaik，Amazon Web Services

總結

Oracle GoldenGate 是異質資料庫和 IT 環境的即時資料擷取和複寫服務。不過，此服務目前不支援 Amazon Relational Database Service (Amazon RDS) for Oracle。如需支援的資料庫清單，請參閱 Oracle GoldenGate for Heterogeneous Databases (Oracle 文件）。此模式說明如何使用 Oracle GoldenGate 和 Oracle GoldenGate 平面檔案轉接器從來源 Oracle 資料庫產生平面檔案，該資料庫可以是內部部署或在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上。然後，您可以將這些一般檔案匯入 Amazon RDS for Oracle 資料庫執行個體。

在此模式中，您可以使用 Oracle GoldenGate 從來源 Oracle 資料庫擷取追蹤檔案。資料幫浦會將追蹤檔案複製到整合伺服器，即 Amazon EC2 執行個體。在整合伺服器上，Oracle GoldenGate 會使用平面檔案轉接器，根據線索檔案的傳輸資料擷取產生一系列循序平面檔案。Oracle GoldenGate 會將資料格式化為分隔符號分隔值或長度分隔值。然後，您可以使用 Oracle SQL*Loader 將一般檔案匯入目標 Amazon RDS for Oracle 資料庫執行個體。

目標對象

此模式適用於具有 Oracle GoldenGate 基礎建置區塊經驗和知識的人員。如需詳細資訊，請參閱 Oracle GoldenGate 架構概觀 (Oracle 文件）。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- Oracle GoldenGate 授權。
- Oracle GoldenGate 轉接器的個別授權。
- 來源 Oracle 資料庫，可在內部部署或在 Amazon EC2 執行個體上執行。
- 用作整合伺服器的 Amazon EC2 Linux 執行個體。如需詳細資訊，請參閱開始使用 Amazon EC2 Linux 執行個體 (Amazon EC2 文件）。
- 目標 Amazon RDS for Oracle 資料庫執行個體。如需詳細資訊，請參閱建立 Oracle 資料庫執行個體 (Amazon RDS 文件）。

產品版本

- Oracle Database Enterprise Edition 10g、11g、12c 或更新版本
- Oracle GoldenGate 12.2.0.1.1 版或更新版本

Architecture

來源技術堆疊

Oracle 資料庫 （ 內部部署或 Amazon EC2 執行個體 ）

目標技術堆疊

Amazon RDS for Oracle

來源和目標架構

1. Oracle GoldenGate 從來源資料庫日誌中擷取追蹤。
2. 資料幫浦會擷取線索，並將其遷移至整合伺服器。
3. Oracle GoldenGate 平面檔案轉接器會讀取線索、來源定義和擷取參數。
4. 您結束擷取，這會產生控制檔案和一般資料檔案。
5. 您可以將一般資料檔案遷移至 中的 Amazon RDS for Oracle 資料庫執行個體 AWS 雲端。

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon Relational Database Service (Amazon RDS) for Oracle 可協助您在 中設定、操作和擴展 Oracle 關聯式資料庫 AWS 雲端。

其他服務

- Oracle GoldenGate 是一種服務，可協助您將資料從一個資料庫複寫、篩選和轉換到另一個異質資料庫或另一個目標拓撲，例如一般檔案。
- Oracle GoldenGate 應用程式轉接器可讓 Oracle GoldenGate 從來源資料庫的追蹤檔案中擷取的交易資料產生一系列序列平面檔案和控制檔案。這些轉接器廣泛用於資料倉儲應用程式和專屬或舊版應用程式中的擷取、轉換和載入 (ETL) 操作。Oracle GoldenGate 會執行此擷取，並在異質資料

庫、平台和作業系統之間近乎即時地套用。轉接器支援不同的輸出檔案格式，例如 CSV 或 Apache Parquet。您可以載入這些產生的檔案，以便將資料載入不同的異質資料庫。

史詩

在來源資料庫伺服器上設定 Oracle GoldenGate

任務	Description	所需的技能
下載 Oracle GoldenGate。	在來源資料庫伺服器上，下載 Oracle GoldenGate 12.2.0.1.1 版或更新版本。如需說明，請參閱下載 Oracle GoldenGate (Oracle 文件）。	DBA
安裝 Oracle GoldenGate。	如需說明，請參閱安裝 Oracle GoldenGate (Oracle 文件）。	DBA
設定 Oracle GoldenGate。	如需說明，請參閱準備 Database for Oracle GoldenGate (Oracle 文件）。	DBA

在整合伺服器上設定 Oracle GoldenGate

任務	Description	所需的技能
下載 Oracle GoldenGate。	在整合伺服器上，下載 Oracle GoldenGate 12.2.0.1.1 版或更新版本。如需說明，請參閱下載 Oracle GoldenGate (Oracle 文件）。	DBA
安裝 Oracle GoldenGate。	建立目錄、設定管理員程序，以及為異質環境建立 `defgen` 檔案。如需說明，請參閱安裝 Oracle GoldenGate (Oracle 文件）。	DBA

變更 Oracle GoldenGate 資料擷取組態

任務	Description	所需的技能
準備 Oracle GoldenGate 轉接器。	在整合伺服器上，設定 Oracle GoldenGate 轉接器軟體。請執行下列操作： 1. 從 Oracle 軟體交付雲端下載 ggs_Adapters_Linux_x64.zip。 2. 解壓縮 ggs_Adapters_Linux_x64.zip。 3. 執行下列命令來安裝轉接器： `tar -xvf ggs_Adapters_Linux_x64.tar`	DBA
設定資料幫浦。	在來源伺服器上，設定資料幫浦，將追蹤檔案從來源伺服器傳輸到整合伺服器。建立資料幫浦參數檔案和線索檔案目錄。如需說明，請參閱設定平面檔案轉接器 (Oracle 文件）。	DBA

產生和遷移一般檔案

任務	Description	所需的技能
產生一般檔案。	建立擷取檔案和控制檔案，然後在整合伺服器上啟動擷取程序。這會擷取資料庫變更，並將來源資料庫寫入一般檔案。	DBA

任務	Description	所需的技能
	如需說明，請參閱使用平面檔案轉接器 (Oracle 文件）。	
將一般檔案載入目標資料庫。	將一般檔案載入目標 Amazon RDS for Oracle 資料庫執行個體。如需詳細資訊，請參閱使用 Oracle SQL*Loader 匯入 (Amazon RDS 文件）。	DBA

疑難排解

問題	解決方案
Oracle GoldenGate 平面檔案轉接器會產生錯誤。	如需轉接器錯誤的描述，請參閱尋找錯誤訊息 (Oracle 文件）。如需故障診斷說明，請參閱故障診斷平面檔案轉接器 (Oracle 文件）。

相關資源

- 安裝 Oracle GoldenGate (Oracle 文件）
- 設定 Oracle GoldenGate (Oracle 文件）
- 了解 Oracle GoldenGate 轉接器 (Oracle 文件）
- 設定平面檔案轉接器 (Oracle 文件）

變更 Python 和 Perl 應用程式，以支援從 Microsoft SQL Server 到 Amazon Aurora PostgreSQL 相容版本的資料庫遷移

Dwarika Patra 和 Deepesh Jayaprakash，Amazon Web Services

總結

此模式說明將資料庫從 Microsoft SQL Server 遷移至 Amazon Aurora PostgreSQL 相容版本時可能需要的應用程式儲存庫變更。模式假設這些應用程式是以 Python 為基礎或以 Perl 為基礎，並提供這些指令碼語言的個別指示。

將 SQL Server 資料庫遷移至 Aurora PostgreSQL 相容包含結構描述轉換、資料庫物件轉換、資料遷移和資料載入。由於 PostgreSQL 和 SQL Server 之間的差異（與資料類型、連線物件、語法和邏輯相關），最困難的遷移任務涉及對程式碼基礎進行必要的變更，以便能夠正確搭配 PostgreSQL 使用。

對於 Python 型應用程式，連線物件和類別會分散在整個系統中。此外，Python 程式碼庫可能會使用多個程式庫來連線至資料庫。如果資料庫連線界面變更，執行應用程式內嵌查詢的物件也需要變更。

對於 Perl 型應用程式，變更涉及連線物件、資料庫連線驅動程式、靜態和動態內嵌 SQL 陳述式，以及應用程式如何處理複雜的動態 DML 查詢和結果集。

遷移應用程式時，您也可以考慮 AWS 上可能的增強功能，例如將 FTP 伺服器取代為 Amazon Simple Storage Service (Amazon S3) 存取。

應用程式遷移程序涉及下列挑戰：

- 連線物件。如果連線物件分散在具有多個程式庫和函數呼叫的程式碼中，您可能需要找到一種一般方法來變更它們以支援 PostgreSQL。
- 在記錄擷取或更新期間處理錯誤或例外狀況。如果您在傳回變數、結果集或資料影格的資料庫上有條件式建立、讀取、更新和刪除 (CRUD) 操作，任何錯誤或例外狀況都可能導致具有層疊效果的應用程式錯誤。這些應該透過適當的驗證和儲存點仔細處理。其中一個儲存點是在`BEGIN...EXCEPTION...END`區塊內呼叫大型內嵌 SQL 查詢或資料庫物件。
- 控制交易及其驗證。這些包括手動和自動遞交和轉返。適用於 Perl 的 PostgreSQL 驅動程式需要您一律明確設定自動遞交屬性。
- 處理動態 SQL 查詢。這需要深入了解查詢邏輯和反覆測試，以確保查詢如預期般運作。
- 效能。您應該確保程式碼變更不會導致應用程式效能降低。

此模式會詳細說明轉換程序。

先決條件和限制

先決條件

- Python 和 Perl 語法的工作知識。
- SQL Server 和 PostgreSQL 中的基本技能。
- 了解您現有的應用程式架構。
- 存取您的應用程式碼、SQL Server 資料庫和 PostgreSQL 資料庫。
- 使用用於開發、測試和驗證應用程式變更的登入資料來存取 Windows 或 Linux （或其他 Unix) 開發環境。
- 對於 Python 型應用程式，您的應用程式可能需要的標準 Python 程式庫，例如 Pandas 來處理資料影格，以及 psycopg2 或 SQLAlchemy 用於資料庫連線。
- 對於 Perl 型應用程式，需要具有相依程式庫或模組的 Perl 套件。Comprehensive Perl Archive Network (CPAN) 模組可支援大多數應用程式需求。
- 所有必要的相依自訂程式庫或模組。
- 用於 SQL Server 讀取存取和 Aurora 讀取/寫入存取的資料庫登入資料。
- PostgreSQL 使用 服務和使用者來驗證和偵錯應用程式變更。
- 在應用程式遷移期間存取開發工具，例如 Visual Studio Code、Sublime Text 或 pgAdmin。

限制

- 有些 Python 或 Perl 版本、模組、程式庫和套件與雲端環境不相容。
- 某些用於 SQL Server 的第三方程式庫和架構無法取代，以支援 PostgreSQL 遷移。
- 效能變化可能需要變更您的應用程式、內嵌 Transact-SQL (T-SQL) 查詢、資料庫函數和預存程序。
- PostgreSQL 支援資料表名稱、資料欄名稱和其他資料庫物件的小寫名稱。
- 有些資料類型，例如 UUID 資料欄，只會以小寫儲存。Python 和 Perl 應用程式必須處理此類案例差異。
- 角色編碼差異必須使用 PostgreSQL 資料庫中對應文字資料欄的正確資料類型來處理。

產品版本

- Python 3.6 或更新版本 （使用支援您作業系統的版本）

- Perl 5.8.3 或更新版本 （ 使用支援您作業系統的 版本 ）
- Aurora PostgreSQL 相容版本 4.2 或更新版本 （ 請參閱詳細資訊)

Architecture

來源技術堆疊

- 指令碼 （ 應用程式程式設計 ） 語言：Python 2.7 或更新版本，或 Perl 5.8
- 資料庫：Microsoft SQL Server 第 13 版
- 作業系統：Red Hat Enterprise Linux (RHEL) 7

目標技術堆疊

- 指令碼 （ 應用程式程式設計 ） 語言：Python 3.6 或更新版本，或 Perl 5.8 或更新版本
- 資料庫：Aurora PostgreSQL 相容 4.2
- 作業系統：RHEL 7

遷移架構

工具

AWS 服務和工具

- Aurora PostgreSQL 相容版本是全受管、PostgreSQL 相容且符合 ACID 規範的關聯式資料庫引擎，結合了高階商業資料庫的速度和可靠性，以及開放原始碼資料庫的成本效益。Aurora PostgreSQL 是 PostgreSQL 的下拉式選單，可讓您更輕鬆且更具成本效益地設定、操作和擴展新的和現有的 PostgreSQL 部署。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可讓您在命令列 Shell 中使用命令與 AWS 服務互動。

其他工具

- Python 和 PostgresSQL 資料庫連線程式庫，例如 psycopg2 和 SQLAlchemy
- Perl 及其 DBI 模組
- PostgreSQL 互動式終端機 (psql)

史詩

將您的應用程式儲存庫遷移至 PostgreSQL – 高階步驟

任務	Description	所需的技能
請依照這些程式碼轉換步驟，將您的應用程式遷移至 PostgreSQL。	1. 設定 PostgreSQL 的資料庫特定 ODBC 驅動程式和程式庫。例如，您可以將其中一個 CPAN 模組用於 Perl 和 pyodbc、psycopg2 或 SQLAlchemy for Python。 2. 使用這些程式庫連線至 Aurora PostgreSQL 相容，以轉換資料庫物件。 3. 在現有的應用程式模組中套用程式碼變更，以取得相容的 T-SQL 陳述式。 4. 重寫應用程式程式碼中的資料庫特定函數呼叫和預存程序。 5. 處理應用程式變數及其用於內嵌 SQL 查詢之資料類型的變更。 6. 處理不相容的資料庫特定函數。 7. 完成轉換應用程式程式碼的end-to-end測試，以進行資料庫遷移。 8. 將 Microsoft SQL Server 的結果與您遷移至 PostgreSQL 的應用程式進行比較。 9. 在 Microsoft SQL Server 和 PostgreSQL 之間執行應用程式效能基準測試。	應用程式開發人員

任務	Description	所需的技能
	10.修改預存程序或由應用程式呼叫的內嵌 T-SQL 陳述式，以改善效能。 下列語意提供適用於 Python 和 Perl 應用程式的一些轉換任務的詳細說明。	
針對遷移的每個步驟使用檢查清單。	將以下內容新增至應用程式遷移每個步驟的檢查清單，包括最後一個步驟： • 檢閱 PostgreSQL 文件，以確保您的所有變更都與 PostgreSQL 標準相容。 • 檢查欄的整數和浮點數。 • 識別插入、更新和擷取的資料列數，以及資料欄名稱和日期/時間戳記。您可以使用 diff 公用程式或撰寫指令碼來自動化這些檢查。 • 完成大型內嵌 SQL 陳述式的效能檢查，並檢查應用程式的整體效能。 • 使用多個 try/catch 區塊，檢查資料庫操作和正常程式退出的正確錯誤處理。 • 檢查以確保有適當的記錄程序。	應用程式開發人員

分析和更新您的應用程式 – Python 程式碼基底

任務	Description	所需的技能
分析您現有的 Python 程式碼庫。	您的分析應包含下列項目，以促進應用程式遷移程序： • 識別程式碼中的所有連線物件。 • 識別所有不相容的內嵌 SQL 查詢（例如 T-SQL 陳述式和預存程序），並分析必要的變更。 • 檢閱程式碼的文件，並追蹤控制流程以了解程式碼功能。當您測試應用程式的效能或負載比較時，這會很有幫助。 • 了解應用程式的目的，以便在資料庫轉換後有效地進行測試。大多數適合與資料庫遷移進行轉換的 Python 應用程式都是將資料從其他來源載入資料庫資料表的摘要，或從資料表擷取資料的擷取器，並將其轉換為適合建立報告或進行 API 呼叫以執行驗證的不同輸出格式（例如 CSV、JSON 或一般檔案）。	應用程式開發人員

轉換資料庫連線以支援 PostgreSQL。	大多數 Python 應用程式使用 pyodbc 程式庫來連接 SQL Server 資料庫，如下所示。
	`import pyodbc`
	`....`
		應用程式開發人員

任務	Description	所需的技能

	try:
	conn_string =
	"Driver=ODBC Driver 17
	for SQL
	Server;UID={};PWD=
	{};Server={};Datab
	ase={}".format
	(conn_user, conn_pass
	word,
	conn_server,
	conn_database)
	conn = pyodbc.co
	nnect(conn_string)
	cur = conn.cursor()
	result = cur.execu
	te(query_string)
	for row in result:
	print (row)
	except Exception as e:
	print(str(e))
	
	轉換資料庫連線以支援 PostgreSQL，如下所示。
	
	import pyodbc
	import psycopg2
	....
	try:
	conn_string =
	'postgresql+psycop
	g2://'+
	conn_user+':'+conn
	_password+'@'+conn
	_server+'/'+conn_d
	atabase
	conn = pyodbc.co
	nnect(conn_string,
	connect_args={'opt
	ions':'-csearch_pa
	th=dbo'})
	cur = conn.cursor()	

任務	Description	所需的技能

	<pre> result = cur.execu
	te(query_string)
	    for row in result:
	    print (row)
	except Exception as e:
	    print(str(e))</pre>	

任務	Description	所需的技能

將內嵌 SQL 查詢變更為 PostgreSQL。	將您的內嵌 SQL 查詢轉換為 PostgreSQL 相容格式。例如，下列 SQL Server 查詢會從資料表擷取字串。
	
	`dtype = "type1"`
	`stm = '"SELECT TOP 1 searchcode FROM TypesTable (NOLOCK) WHERE code="' + "'" + str(dtype) + "'"`
	`# For Microsoft SQL Server Database Connection`
	`engine = create_engine('mssql+pyodbc:///?odbc_connect=%s' % urllib.parse.quote_plus(conn_string), connect_args={'connect_timeout':login_timeout})`
	`conn = engine_connect()`
	`rs = conn.execute(stm)`
	`for row in rs:`
	`  print(row)`
	
	轉換後，與 PostgreSQL 相容的內嵌 SQL 查詢如下所示。
	
	`dtype = "type1"`
	`stm = '"SELECT searchcode FROM TypesTable WHERE code="' + "'" + str(dtype) + "' LIMIT 1"`
	`# For PostgreSQL Database Connection`
		應用程式開發人員

任務	Description	所需的技能

	`engine = create_engine(‘postgres+psycopg2://%s’ %conn_string, connect_args={‘connect_timeout’:login_timeout})`
	`conn = engine.connect()`
	`rs = conn.execute(stm)`
	`for row in rs:`
	`  print(row)`	

任務	Description	所需的技能

處理動態 SQL 查詢。	動態 SQL 可以存在於一個指令碼或多個 Python 指令碼中。先前範例示範如何使用 Python 的字串取代函數插入變數來建構動態 SQL 查詢。替代方法是在適用的情況下，使用變數附加查詢字串。
	
	在下列範例中，查詢字串會根據函數傳回的值，以即時方式建構。
	
	`query = '"SELECT id from equity e join issues i on e.permId=i.permId where e.id'"`
	`query += get_id_filter(ids) + " e.id is NOT NULL`
	
	這些類型的動態查詢在應用程式遷移期間非常常見。請依照下列步驟處理動態查詢：
	
	• 檢查整體語法（例如，具有 子句的SELECT 陳述式JOIN語法）。
	• 驗證查詢中使用的所有變數或資料欄名稱，例如 i和 id。
	• 檢查查詢中使用的函數、引數和傳回值（例如，get_id_filter 及其引數ids)。
		應用程式開發人員

任務	Description	所需的技能

處理結果集、變數和資料影格。	對於 Microsoft SQL Server，您可以使用 Python 方法，例如 `fetchall()` `fetchone()` 或 從資料庫擷取結果集。您也可以使用 `fetchmany(size)` 並指定要從結果集傳回的記錄數目。若要這樣做，您可以使用 pyodbc 連線物件，如下列範例所示。
	
	pyodbc (Microsoft SQL Server)
	
	`import pyodbc`
	`server = 'tcp:myse`
	`rver.database.wind`
	`ows.net'`
	`database = 'exampledb'`
	`username = 'exampleu`
	`sername'`
	`password = 'examplep`
	`assword'`
	`conn = pyodbc.co`
	`nnect('DRIVER={ODB`
	`C Driver 17 for SQL`
	` Server};SERVER='+s`
	`erver+';DATABASE='`
	`+database+';UID='+`
	`username+';PWD='+`
	` password)`
	`cursor = conn.cursor()`
	`cursor.execute("SELECT`
	` * FROM ITEMS")`
	`row = cursor.fe`
	`tchone()`
	`while row:`
	`    print(row[0])`
	`    row = cursor.fe`
	`tchone()`
		應用程式開發人員

任務	Description	所需的技能

	在 Aurora 中，若要執行類似任務，例如連線至 PostgreSQL 和擷取結果集，您可以使用 psycopg2 或 SQLAlchemy。這些 Python 程式庫提供連線模組和游標物件，以周遊 PostgreSQL 資料庫記錄，如下列範例所示。
	
	psycopg2 (Aurora PostgreSQL 相容）
	
	`import psycopg2`
	`query = "SELECT * FROM ITEMS;"`
	`//Initialize variables`
	`host=dbname=user=password=port=sslmode=connect_timeout=""`
	`connstring = "host='{host}' dbname='{dbname}' user='{user}' \`
	`password='{password}'port='{port}'".format(host=host,dbname=dbname,\`
	`user=user,password=password,port=port)`
	`conn = psycopg2.connect(connstring)`
	`cursor = conn.cursor()`
	`cursor.execute(query)`
	`column_names = [column[0] for column in cursor.description]`	

任務	Description	所需的技能

	<pre>print("Column Names: ", column_names)
	print("Column values: "
	for row in cursor:
	    print("itemid :", row[0])
	    print("itemdescrpt
	ion :", row[1])
	    print("it
	emprice :", row[3]))</pre>
	SQLAlchemy (Aurora PostgreSQL 相容)
	<pre>from sqlalchemy import create_engine
	from pandas import DataFrame
	conn_string = 'postgres
	ql://core:database
	@localhost:5432/ex
	ampledatabase'
	engine = create_en
	gine(conn_string)
	conn = engine.co
	nnect()
	dataid = 1001
	result = conn.exec
	ute("SELECT * FROM
	ITEMS")
	df = DataFrame
	(result.fetchall())
	df.columns = result.ke
	ys()
	df = pd.DataFrame()
	engine.connect()
	df = pd.read_sql_query(
	sql_query, engine,
	coerce_float=False)
	print("df=", df)</pre>	

任務	Description	所需的技能
在遷移期間和之後測試您的應用程式。	測試遷移的 Python 應用程式是持續進行的程序。由於遷移包含連線物件變更 (psycopg2 或 SQLAlchemy)、錯誤處理、新功能（資料框架）、內嵌 SQL 變更、大量複製功能 (bcp而非 COPY) 和類似變更，因此必須在應用程式遷移期間和之後仔細測試。檢查： • 錯誤條件和處理 • 遷移後的任何記錄不相符 • 記錄更新或刪除 • 執行應用程式所需的時間	應用程式開發人員

分析和更新您的應用程式 – Perl 程式碼庫

任務	Description	所需的技能
分析現有的 Perl 程式碼基底。	您的分析應包含下列項目，以促進應用程式遷移程序。您應該識別： • 任何 INI 或組態型程式碼 • 資料庫特定的標準開放式資料庫連線 (ODBC) Perl 驅動程式或任何自訂驅動程式 • 內嵌和 T-SQL 查詢所需的程式碼變更 • 各種 Perl 模組之間的互動（例如，由多個功能元件呼叫或使用的單一 Perl ODBC 連線物件）	應用程式開發人員

任務	Description	所需的技能
	• 資料集和結果集處理 • 外部相依 Perl 程式庫 • 應用程式中使用的任何 APIs • Perl 版本相容性和與 Aurora PostgreSQL 相容驅動程式相容性	

任務	Description	所需的技能

轉換 Perl 應用程式和 DBI 模組的連線，以支援 PostgreSQL。	Perl 型應用程式通常會使用 Perl DBI 模組，這是 Perl 程式設計語言的標準資料庫存取模組。您可以使用相同的 DBI 模組搭配 SQL Server 和 PostgreSQL 的不同驅動程式。
	
	如需必要 Perl 模組、安裝和其他說明的詳細資訊，請參閱 DBD：：Pg 文件。下列範例會連線至位於 的 Aurora PostgreSQL 相容`examplete st-aurorapg-databa se.cluster-samplec lusture.us-east-.r ds.amazonaws.com` 。
	
	`#!/usr/bin/perl`
	`use DBI;`
	`use strict;`
	`my $driver = "Pg";`
	`my $hostname = "examplet est-aurorapg-datab ase-sampleclusture .us-east.rds.amazo naws.com"`
	`my $dsn = "DBI:$driver: dbname = $hostname ;host = 127.0.0.1;port = 5432";`
	`my $username = "postgres ";`
	`my $password = "pass123" ;`
	`$dbh = DBI->conn ect("dbi:Pg:dbname =$hostname;host=$h`
		應用程式開發人員

任務	Description	所需的技能
	ost;port=$port;opt ions=$options", $username, $password, {AutoCommit => 0, RaiseError => 1, PrintError => 0});	

任務	Description	所需的技能

將內嵌 SQL 查詢變更為 PostgreSQL。	您的應用程式可能有具有 SELECT、UPDATE、DELETE和類似陳述式的內嵌 SQL 查詢，其中包含 PostgreSQL 不支援的查詢子句。例如，PostgreSQL NOLOCK 不支援查詢關鍵字，例如 TOP和 。下列範例示範如何處理 TOP、 NOLOCK和布林值變數。
	
	在 SQL Server 中：
	
	`$sqlStr = $sqlStr . "WHERE a.student_id in (SELECT TOP $numofRecords c_student_id \ FROM active_student_record b WITH (NOLOCK) \ INNER JOIN student_contributor c WITH (NOLOCK) on c.contributor_id = b.c_st)`
	
	對於 PostgreSQL，請轉換為：
	
	`$sqlStr = $sqlStr . "WHERE a.student_id in (SELECT TOP $numofRecords c_student_id \ FROM active_student_record b INNER JOIN student_contributor c \`
		應用程式開發人員

任務	Description	所需的技能
	on c.contributor_id = b.c_student_contr_id WHERE b_current_1 is true \ LIMIT $numofRecords)"	

任務	Description	所需的技能
處理動態 SQL 查詢和 Perl 變數。	動態 SQL 查詢是在應用程式執行時間建置的 SQL 陳述式。這些查詢會在應用程式執行時動態建構，視特定條件而定，因此在執行時間之前，不會知道查詢的全文。例如，財務分析應用程式每天分析前 10 個共享，這些共享每天都會變更。SQL 資料表是根據最佳執行者建立的，在執行時間之前不會知道這些值。 假設此範例的內嵌 SQL 查詢會傳遞至包裝函式，以取得變數中設定的結果，然後變數會使用條件來判斷資料表是否存在： • 如果資料表存在，請勿建立它；請執行一些處理。 • 如果資料表不存在，請建立資料表並執行一些處理。 以下是變數處理的範例，後面接著此使用案例的 SQL Server 和 PostgreSQL 查詢。 `my $tableexists = db_read( arg 1, $sql_qry, undef, 'writer'); my $table_already_exists = $tableexists->[0]{table_exists}; if ($table_already_exists){`	應用程式開發人員

任務	Description	所需的技能

	<pre>```
	# do some thing
	}
	else {
	# do something else
	}
	``</pre>
	SQL Server：
	<pre>```
	my $sql_qry = "SELECT
	 OBJECT_ID('$backen
	dTable', 'U') table_exi
	sts", undef, 'writer')
	";
	``</pre>
	PostgreSQL：
	<pre>```
	my $sql_qry = "SELECT
	 TO_REGCLASS('$back
	endTable', 'U')
	 table_exists", undef,
	 'writer')";
	``</pre>
	下列範例使用內嵌 SQL 中的 Perl 變數，該變數會搭配 執行SELECT陳述式JOIN，以擷取資料表的主索引鍵和索引鍵資料欄的位置。
	SQL Server：
	<pre>```
	my $sql_qry = "SELECT
	 column_name',
	 character_maxi
	 mum_length \
	FROM INFORMATION_SCHEMA
	.COLUMNS \
	WHERE TABLE_SCH
	EMA='$example_sche
	maInfo' \	

任務	Description	所需的技能
	AND TABLE_NAME='$examp le_table' \ AND DATA_TYPE IN ('varchar','nvarch ar');"; PostgreSQL： my $sql_qry = "SELECT c1.column_name, c1.ordinal_position \ FROM information_schema .key_column_usage AS c LEFT \ JOIN information_schema .table_constraints AS t1 \ ON t1.constraint_name = c1.constraint_name \ WHERE t1.table_name = $example_schemaInf o'.'$example_table' \ AND t1.constraint_type = 'PRIMARY KEY' ;";	

對 Perl 型或 Python 型應用程式進行其他變更，以支援 PostgreSQL

任務	Description	所需的技能
將其他 SQL Server 建構轉換為 PostgreSQL。	下列變更適用於所有應用程式，無論程式設計語言為何。 • 限定您的應用程式搭配新的適當結構描述名稱使用的資料庫物件。	應用程式開發人員

任務	Description	所需的技能
	• 使用 PostgreSQL 中的定序功能來處理 LIKE 運算子的區分大小寫比對。 • 處理不支援的資料庫特定函數，例如 DATEDIFF、DATEADD、CONVGETDATE和 CAST運算子。如需同等 PostgreSQL 相容函數，請參閱其他資訊區段中的原生或內建 SQL 函數。 • 在比較陳述式中處理布林值。 • 處理函數的傳回值。這些可以是記錄集、資料影格、變數和布林值。根據您的應用程式需求和支援 PostgreSQL 來處理這些項目。 • 使用使用者定義的新 PostgreSQL 函數來處理匿名區塊 （ 例如 BEGIN TRAN)。 • 轉換資料列的大量插入。從應用程式內部呼叫的 SQL Server 大量複製 (bcp) 公用程式的 PostgreSQL 對等值為 COPY。 • 轉換資料欄串連運算子。SQL Server 使用 + 進行字串串連，但 PostgreSQL 使用 ||。	

改善效能

任務	Description	所需的技能
利用 AWS 服務來增強效能。	當您遷移至 AWS 雲端時，您可以精簡應用程式和資料庫設計，以利用 AWS 服務。例如，如果連接至 Aurora PostgreSQL 相容資料庫伺服器的 Python 應用程式查詢花費的時間比原始 Microsoft SQL Server 查詢更長，您可以考慮直接從 Aurora 伺服器建立歷史資料的摘要至 Amazon Simple Storage Service (Amazon S3) 儲存貯體，並使用 Amazon Athena 型 SQL 查詢來產生使用者儀表板的報告和分析資料查詢。	應用程式開發人員、雲端架構師

相關資源

- Perl
- Perl DBI 模組
- Python
- psycopg2
- SQLAlchemy
- 大量複製 - PostgreSQL
- 大量複製 - Microsoft SQL Server
- PostgreSQL
- 使用 Amazon Aurora PostgreSQL

其他資訊

Microsoft SQL Server 和 Aurora PostgreSQL 相容都是 ANSI SQL 投訴。不過，當您將 Python 或 Perl 應用程式從 SQL Server 遷移至 PostgreSQL 時，仍應注意語法、資料欄資料類型、原生資料庫特定函數、大量插入和區分大小寫方面的任何不相容。

以下各節提供有關可能的不一致的詳細資訊。

資料類型比較

從 SQL Server 到 PostgreSQL 的資料類型變更，可能會導致應用程式操作所產生資料的顯著差異。如需資料類型的比較，請參閱 Sqlines 網站上的 資料表。

原生或內建 SQL 函數

某些函數的行為在 SQL Server 和 PostgreSQL 資料庫之間有所不同。下表提供比較。

Microsoft SQL Server	Description	PostgreSQL
`CAST`	將一個值從某個資料類型轉換至另一個類型。	PostgreSQL `type ::` `operator`
`GETDATE()`	以 `YYYY-MM-DD hh:mm:ss.mmm` 格式傳回目前的資料庫系統日期和時間。	`CLOCK_TIMESTAMP`
`DATEADD`	將時間/日期間隔新增至日期。	`INTERVAL` 表達式
`CONVERT`	將值轉換為特定資料格式。	`TO_CHAR`
`DATEDIFF`	傳回兩個日期之間的差異。	`DATE_PART`
`TOP`	限制SELECT結果集中的資料列數。	`LIMIT/FETCH`

匿名區塊

結構化 SQL 查詢會組織成數個區段，例如宣告、可執行檔和例外狀況處理。下表比較簡單匿名區塊的 Microsoft SQL Server 和 PostgreSQL 版本。對於複雜的匿名區塊，我們建議您在應用程式中呼叫自訂資料庫函數。

Microsoft SQL Server

```
my $sql_qry1=
my $sql_qry2 =
my $sqlqry = "BEGIN TRAN
$sql_qry1 $sql_qry2
if @\@error !=0 ROLLBACK
TRAN
else COMIT TRAN";
```

PostgreSQL

```
my $sql_qry1=
my $sql_qry2 =
my $sql_qry = " DO \$\$
BEGIN
$header_sql $content_sql
END
\$\$";
```

其他差異

- 大量插入資料列：相當於 Microsoft SQL Server bcp 公用程式的 PostgreSQL 是 COPY。 https://docs.microsoft.com/en-us/sql/tools/bcp-utility?view=sql-server-ver15
- 區分大小寫：PostgreSQL 中的資料欄名稱區分大小寫，因此您必須將 SQL Server 資料欄名稱轉換為小寫或大寫。當您擷取或比較資料，或在結果集或變數中放置資料欄名稱時，這會成為一個因素。下列範例會識別值可能以大寫或小寫儲存的資料欄。

```
my $sql_qry = "SELECT $record_id FROM $exampleTable WHERE LOWER($record_name) =
 \'failed transaction\'";
```

- Concatenation：SQL Server 使用 +做為字串串連的運算子，而 PostgreSQL 則使用 ||。
- 驗證：您應該先測試和驗證內嵌 SQL 查詢和函數，再將其用於 PostgreSQL 的應用程式碼。
- ORM 程式庫包含 ：您也可以尋找將現有的資料庫連線程式庫包含或取代為 Python ORM 程式庫，例如 SQLAlchemy 和 PynomoDB。這有助於使用物件導向範式，從資料庫輕鬆查詢和操作資料。

依工作負載的遷移模式

主題

- IBM
- Microsoft
- N/A
- 開放原始碼
- Oracle
- SAP

IBM

- 使用 AWS DMS 將 Db2 資料庫從 Amazon EC2 遷移至 Aurora MySQL 相容
- 使用日誌運送將 LUW 的 Db2 遷移至 Amazon EC2，以減少中斷時間
- 將 LUW 的 Db2 遷移至具有高可用性災難復原的 Amazon EC2
- 使用 AWS DMS 和 AWS SCT 從 Amazon EC2 上的 IBM Db2 遷移至 Aurora PostgreSQL 相容 Amazon EC2
- 從 IBM WebSphere Application Server 遷移至 Amazon EC2 上的 Apache Tomcat

Microsoft

- 加速探索 Microsoft 工作負載並將其遷移至 AWS
- 變更 Python 和 Perl 應用程式，以支援從 Microsoft SQL Server 到 Amazon Aurora PostgreSQL 相容版本的資料庫遷移
- 使用 Microsoft Excel 和 Python 為 AWS DMS 任務建立 AWS CloudFormation 範本
- 使用 AWS DMS 將 Microsoft SQL Server 資料庫匯出至 Amazon S3
- 從 SQL Server 遷移至 PostgreSQL 時，實作 PII 資料的 SHA1 雜湊
- 將 EC2 Windows 執行個體擷取並遷移至 AWS Managed Services 帳戶
- 將訊息佇列從 Microsoft Azure Service Bus 遷移至 Amazon SQS
- 使用 AWS DMS 將 Microsoft SQL Server 資料庫從 Amazon EC2 遷移至 Amazon DocumentDB
- 使用 AWS DMS 和 AWS SCT 將 Microsoft SQL Server 資料庫遷移至 Aurora MySQL
- 將 .NET 應用程式從 Microsoft Azure App Service 遷移至 AWS Elastic Beanstalk
- 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon EC2
- 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon RDS for SQL Server
- 使用連結的伺服器將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon RDS for SQL Server
- 使用原生備份和還原方法，將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon RDS for SQL Server
- 使用 AWS DMS 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon Redshift
- 使用 AWS SCT 資料擷取代理程式將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon Redshift
- 將內部部署 Microsoft SQL Server 資料庫遷移至執行 Linux 的 Amazon EC2 上的 Microsoft SQL Server
- 使用 Rclone 將資料從 Microsoft Azure Blob 遷移至 Amazon S3
- 使用 appcmd.exe 將 IIS 託管應用程式遷移至 Amazon EC2
- 使用 Application Migration Service 將內部部署 Microsoft SQL Server 資料庫遷移至 Amazon EC2
- 使用 ACM 將 Windows SSL 憑證遷移至 Application Load Balancer
- 在 AWS 雲端中重新託管內部部署工作負載：遷移檢查清單
- 解決將 Microsoft SQL Server 遷移至 AWS 雲端後的連線錯誤
- 使用 Amazon FSx 設定 SQL Server Always On FCI 的異地同步備份基礎設施

N/A

- 在重新託管遷移至 期間建立防火牆請求的核准程序 AWS

開放原始碼

- 在 Aurora PostgreSQL 相容中建立應用程式使用者和角色
- 使用 AWS SCT 和 AWS DMS 將 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL AWS CLI CloudFormation
- 使用原生工具將內部部署 MariaDB 資料庫遷移至 Amazon RDS for MariaDB
- 將內部部署 MySQL 資料庫遷移至 Amazon RDS for MySQL
- 將內部部署 MySQL 資料庫遷移至 Aurora MySQL
- 將內部部署 PostgreSQL 資料庫遷移至 Aurora PostgreSQL
- 將 Couchbase Server 資料庫遷移至 Amazon EC2
- 使用 Auto Scaling 從 IBM WebSphere Application Server 遷移至 Amazon EC2 上的 Apache Tomcat
- 使用 pglogical 從 Amazon EC2 上的 PostgreSQL 遷移至 Amazon RDS for PostgreSQL Amazon EC2
- 使用 AWS App2Container 將內部部署 Java 應用程式遷移至 AWS
- 使用 Percona XtraBackup、Amazon EFS 和 Amazon S3 將內部部署 MySQL 資料庫遷移至 Aurora MySQL
- 將 Oracle 外部資料表遷移至 Amazon Aurora PostgreSQL 相容
- 重新啟動 RHEL 來源伺服器後，在不停用 SELinux 的情況下自動重新啟動 AWS 複寫代理程式
- 使用 pg_transport 在兩個 Amazon RDS 資料庫執行個體之間傳輸 PostgreSQL 資料庫

Oracle

- 將 Oracle 的 VARCHAR2(1) 資料類型轉換為 Amazon Aurora PostgreSQL 的布林資料類型
- 使用 PostgreSQL 相容 Aurora 全域資料庫模擬 Oracle DR
- 使用 Oracle SQL Developer 和 AWS SCT，逐步從 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL
- 在 Aurora PostgreSQL 相容中使用檔案編碼將 BLOB 檔案載入 TEXT
- 使用 AWS DMS，以 SSL 模式將 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL
- 將 Amazon RDS for Oracle 資料庫遷移至另一個資料庫 AWS 帳戶 ， AWS 區域 並使用 AWS DMS 進行持續複寫
- 使用 Oracle Data Pump 將內部部署 Oracle 資料庫遷移至 Amazon EC2
- 使用 Logstash 將內部部署 Oracle 資料庫遷移至 Amazon OpenSearch Service
- 使用 AWS DMS 和 AWS SCT 將內部部署 Oracle 資料庫遷移至 Amazon RDS for MySQL
- 將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 透過資料庫連結使用直接 Oracle Data Pump Import，將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 使用 Oracle Data Pump 將內部部署 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 使用 Oracle 旁觀者和 AWS DMS 將內部部署 Oracle 資料庫遷移至 Amazon RDS for PostgreSQL
- 將內部部署 Oracle 資料庫遷移至 Amazon EC2 上的 Oracle
- 使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫從 Amazon EC2 遷移至 Amazon RDS for MariaDB
- 使用 AWS DMS 將 Oracle 資料庫從 Amazon EC2 遷移至 Amazon RDS for Oracle
- 使用 Oracle GoldenGate 平面檔案轉接器，將 Oracle 資料庫遷移至 Amazon RDS for Oracle
- 使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫遷移至 Amazon Redshift
- 使用 AWS DMS 和 AWS SCT 將 Oracle 資料庫遷移至 Aurora PostgreSQL
- 使用 Oracle Data Pump 和 AWS DMS 將 Oracle JD Edwards EnterpriseOne 資料庫遷移至 AWS
- 使用 AWS DMS 將 Oracle 分割的資料表遷移至 PostgreSQL
- 使用 AWS DMS 將 Oracle PeopleSoft 資料庫遷移至 AWS
- 將資料從現場部署 Oracle 資料庫遷移至 Aurora PostgreSQL
- 從 Amazon RDS for Oracle 遷移至 Amazon RDS for MySQL
- 使用具體化視觀表和 AWS DMS，從 Oracle 8i 或 9i 遷移至 Amazon RDS for PostgreSQL
- 使用 SharePlex 和 AWS DMS 從 Oracle 8i 或 9i 遷移至 Amazon RDS for PostgreSQL

- 使用 Oracle GoldenGate 從 Oracle 資料庫遷移至 Amazon RDS for PostgreSQL
- 使用 AWS DMS 和 AWS SCT 從 Oracle on Amazon EC2 遷移至 Amazon RDS for MySQL
- 在 Amazon ECS 上從 Oracle WebLogic 遷移至 Apache Tomcat (TomEE)
- 將函數型索引從 Oracle 遷移至 PostgreSQL
- 將舊版應用程式從 Oracle Pro*C 遷移至 ECPG
- 將 Oracle CLOB 值遷移至 AWS 上的 PostgreSQL 中的個別資料列
- 將 Oracle 資料庫錯誤代碼遷移至與 Amazon Aurora PostgreSQL 相容的資料庫
- 使用延伸模組將 Oracle 原生函數遷移至 PostgreSQL
- 將 Oracle PeopleSoft 遷移至 Amazon RDS Custom
- 將 Oracle ROWID 功能遷移至 AWS 上的 PostgreSQL
- 將 Oracle SERIALLY_REUSABLE pragma 套件遷移至 PostgreSQL
- 將虛擬產生的資料欄從 Oracle 遷移至 PostgreSQL
- 在 Aurora PostgreSQL 相容上設定 Oracle UTL_FILE 功能
- 從 Oracle 遷移到 Amazon Aurora PostgreSQL 後驗證資料庫物件

SAP

- 使用 AWS DMS 從 SAP ASE 遷移至 Amazon RDS for SQL Server
- 使用 AWS SCT 和 AWS DMS 將 Amazon EC2 上的 SAP ASE 遷移至與 Amazon Aurora PostgreSQL 相容

更多模式

- 安裝 以 AWS 服務 從 IBM z/OS 存取 AWS CLI
- 評估將 SQL Server 資料庫遷移至 AWS 上的 MongoDB Atlas 的查詢效能
- 使用 DR Orchestrator Framework 自動化跨區域容錯移轉和容錯回復
- 使用 AWS Lambda 和 任務排程器，在 Amazon EC2 上執行的 SQL Server Express 版本中自動化資料庫任務
- 在 AWS 雲端中建置進階大型主機檔案檢視器
- 透過私有網路連線至 Application Migration Service 資料和控制平面
- 容器化已由 Blu Age 現代化的大型主機工作負載
- 將 JSON Oracle 查詢轉換為 PostgreSQL 資料庫 SQL
- 將 Teradata NORMALIZE 暫時功能轉換為 Amazon Redshift SQL
- 將 Teradata RESET WHEN 功能轉換為 Amazon Redshift SQL
- 使用 跨帳戶複製 Amazon DynamoDB 資料表 AWS Backup
- 使用私有靜態 IPs 在 Amazon EC2 上部署 Cassandra 叢集，以避免重新平衡
- 使用 AWS CDK 搭配 TypeScript 部署多堆疊應用程式
- 使用 Terraform 在 Amazon EC2 和 Amazon FSx 上部署 SQL Server 容錯移轉叢集執行個體
- 在 Amazon Aurora PostgreSQL 和 Amazon RDS for PostgreSQL 中模擬 Oracle PL/SQL 關聯陣列
- 使用 AWR 報告估計 Oracle 資料庫的 Amazon RDS 引擎大小
- 在 Quick Sight 中使用 AWS Mainframe Modernization 和 Amazon Q 產生資料洞見
- 在 Aurora PostgreSQL 中處理動態 SQL 陳述式中的匿名區塊
- 遷移至 Amazon ECR 儲存庫時，自動識別重複的容器映像
- 使用 FSx for Windows File Server 在 Amazon EC2 上設定 Microsoft SQL Server 容錯移轉叢集
- 使用 AWS Glue 將 Apache Cassandra 工作負載遷移至 Amazon Keyspaces
- 將您的容器工作負載從 Azure Red Hat OpenShift (ARO) 遷移至 Red Hat OpenShift Service on AWS (ROSA)
- 使用 SharePlex 和 AWS DMS 從 Oracle 8i 或 9i 遷移至 Amazon RDS for Oracle
- 使用 遷移 Microsoft SQL Server Always On 可用性群組 AWS Application Migration Service
- 將具有超過 100 個引數的 Oracle 函數和程序遷移至 PostgreSQL
- 使用具有相同主機名稱的 SAP HSR 將 SAP HANA 遷移至 AWS
- 使用分散式可用性群組將 SQL Server 遷移至 AWS

- 使用 和 Terraform 現代化 AWS Transform 和部署大型主機應用程式
- 使用 Micro Focus Enterprise Server 和 LRS VPSX/MFI 將 AWS 上的大型主機線上列印工作負載現代化
- AWS 使用 Rocket Enterprise Server 和 LRS PageCenterX 在 上現代化大型主機輸出管理
- 當您從 F5 遷移到 AWS 上的 Application Load Balancer 時修改 HTTP 標頭
- 使用應用程式復原控制器管理 EMR 叢集的多可用區域容錯移轉
- 分析從 Oracle 到 PostgreSQL 的部分資料庫遷移的物件相依性
- 使用 Terraform 設定資料庫遷移的 CI/CD 管道
- 使用 AWS Elastic Disaster Recovery 為 Oracle JD Edwards EnterpriseOne 設定災難復原
- 使用 AWS Private CA 和 AWS RAM 簡化私有憑證管理
- 以 CSV 檔案將大規模 Db2 z/OS 資料傳輸至 Amazon S3

現代化

主題

- 使用 DynamoDB TTL 自動將項目封存至 Amazon S3
- 在 Amazon OpenSearch Service 中建置多租戶無伺服器架構
- 使用 AWS CDK 搭配 TypeScript 部署多堆疊應用程式
- 使用 AWS SAM 自動化巢狀應用程式的部署
- 使用 AWS Lambda 字符販賣機實作 Amazon S3 的 SaaS 租用戶隔離
- 使用 AWS Step Functions 實作無伺服器 saga 模式
- 使用 AWS CDK 設定 Amazon ECS Anywhere 來管理內部部署容器應用程式
- 在 AWS 上現代化 ASP.NET Web Forms 應用程式
- 使用 AWS Fargate 大規模執行事件驅動和排程工作負載
- 使用 C# 和 AWS CDK 在孤立模型的 SaaS 架構中加入租用戶
- 使用 CQRS 和事件來源將整體分解為微服務
- 更多模式

使用 DynamoDB TTL 自動將項目封存至 Amazon S3

Tabby Ward，Amazon Web Services

總結

此模式提供從 Amazon DynamoDB 資料表移除舊資料並將其封存至 Amazon Web Services (AWS) 上 Amazon Simple Storage Service (Amazon S3) 儲存貯體的步驟，而不必管理伺服器機群。

此模式使用 Amazon DynamoDB 存留時間 (TTL) 自動刪除舊項目，並使用 Amazon DynamoDB 串流擷取 TTL 過期項目。然後，它會將 DynamoDB Streams 連線至 AWS Lambda，其會執行程式碼，而無需佈建或管理任何伺服器。

將新項目新增至 DynamoDB 串流時，會啟動 Lambda 函數，並將資料寫入 Amazon Data Firehose 交付串流。Firehose 提供簡單、全受管的解決方案，將資料作為封存載入 Amazon S3。

DynamoDB 通常用於存放時間序列資料，例如網頁點擊串流資料或來自感應器和連線裝置的物聯網 (IoT) 資料。許多客戶不想要刪除較不常存取的項目，而是想要將其封存以供稽核之用。TTL 會根據時間戳記屬性自動刪除項目，簡化此封存。

TTL 刪除的項目可以在 DynamoDB Streams 中識別，該串流會擷取項目層級修改的時間順序，並將序列存放在日誌中長達 24 小時。此資料可供 Lambda 函數使用，並封存在 Amazon S3 儲存貯體中，以降低儲存成本。為了進一步降低成本，可以建立 Amazon S3 生命週期規則，以自動將資料（建立後立即）轉換為成本最低的儲存類別。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在 macOS、Linux 或 Windows 上安裝和設定 AWS Command Line Interface (AWS CLI) 1.7 或更新版本。 macOS
- Python 3.7 或更新版本。
- Boto3，已安裝並設定。如果尚未安裝 Boto3，請執行 `python -m pip install boto3`命令來安裝它。

Architecture

技術堆疊

- Amazon DynamoDB
- Amazon DynamoDB Streams
- Amazon Data Firehose
- AWS Lambda
- Amazon S3

1. TTL 會刪除項目。
2. DynamoDB 串流觸發程序會叫用 Lambda 串流處理器函數。
3. Lambda 函數會以批次格式將記錄放入 Firehose 交付串流中。
4. 資料記錄會封存在 S3 儲存貯體中。

工具

- AWS CLI – AWS Command Line Interface (AWS CLI) 是管理 AWS 服務的統一工具。
- Amazon DynamoDB – Amazon DynamoDB 是一種鍵值和文件資料庫，可在任何規模下提供單一位數毫秒的效能。
- Amazon DynamoDB 存留時間 (TTL) – Amazon DynamoDB TTL 可協助您定義每個項目的時間戳記，以判斷何時不再需要項目。
- Amazon DynamoDB Streams – Amazon DynamoDB Streams 會擷取任何 DynamoDB 資料表中項目層級修改的時間順序，並將此資訊存放在日誌中長達 24 小時。
- Amazon Data Firehose – Amazon Data Firehose 是將串流資料可靠載入資料湖、資料存放區和分析服務的最簡單方法。
- AWS Lambda – AWS Lambda 執行程式碼，無需佈建或管理伺服器。您只需為使用的運算時間支付費用。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種物件儲存服務，可提供業界領先的可擴展性、資料可用性、安全性和效能。

Code

此模式的程式碼可在 GitHub Archive 項目中使用 DynamoDB TTL 儲存庫傳送至 S3。

史詩

設定 DynamoDB 資料表、TTL 和 DynamoDB 串流

任務	Description	所需的技能

建立 DynamoDB 資料表。	使用 AWS CLI 在 DynamoDB 中建立名為 的資料表`Reservation` 。選擇隨機讀取容量單位 (RCU) 和寫入容量單位 (WCU)，並為您的資料表提供兩個屬性： `ReservationID` 和 `ReservationDate` 。
	
	`aws dynamodb create-table \`
	`--table-name Reservation \`
	`--attribute-definitions Attribute Name=ReservationID,AttributeType=S AttributeName=ReservationDate,AttributeType=N \`
	`--key-schema Attribute Name=ReservationID,KeyType=HASH AttributeName=ReservationDate,KeyType=RANGE \`
	`--provisioned-throughput ReadCapacityUnits=100,WriteCapacityUnits=100`
	
	`ReservationDate` 是用來開啟 TTL 的 epoch 時間戳記。
		雲端架構師、應用程式開發人員

任務	Description	所需的技能

開啟 DynamoDB TTL。	使用 AWS CLI 為 `ReservationDate` 屬性開啟 DynamoDB TTL。
	
	`aws dynamodb update-time-to-live \`
	`--table-name Reservation\`
	`  --time-to-live-specification Enabled=true,AttributeName=ReservationDate`
		雲端架構師、應用程式開發人員

任務	Description	所需的技能

開啟 DynamoDB 串流。	使用 AWS CLI，透過使用串流類型來開啟Reservation 資料表的 DynamoDB NEW_AND_OLD_IMAGES 串流。
	
	`aws dynamodb update-table \`
	`--table-name Reservation \`
	`  --stream-specification StreamEnabled=true,StreamViewType=NEW_AND_OLD_IMAGES`
	
	此串流將包含新項目、更新項目、已刪除項目和 TTL 刪除項目的記錄。TTL 刪除的項目記錄包含額外的中繼資料屬性，以區分它們與手動刪除的項目。TTL 刪除userIdentity 的 欄位表示 DynamoDB 服務已執行刪除動作。
	
	在此模式中，只會封存 TTL 刪除的項目，但您只能封存 eventName 為 REMOVE且userIdentity 包含principalId 等於 的記錄dynamodb.amazonaws.com 。
		雲端架構師、應用程式開發人員

建立和設定 S3 儲存貯體

任務	Description	所需的技能
建立 S3 儲存貯體。	使用 AWS CLI 在您的 AWS 區域中建立目的地 S3 儲存貯體，`us-east-1` 將 取代為您的區域，並將 amzn-s3-demo-destination-bucket 取代為您的儲存貯體名稱。 `aws s3api create-bucket \ --bucket amzn-s3-demo-destination-bucket \ --region us-east-1` 請確定 S3 儲存貯體的名稱是全域唯一的，因為命名空間是由所有 AWS 帳戶共用。	雲端架構師、應用程式開發人員
建立 S3 儲存貯體的 30 天生命週期政策。	1. 登入 AWS 管理主控台並開啟 Amazon S3 主控台。 2. 選擇包含來自 Firehose 資料的 S3 儲存貯體。 3. 在 S3 儲存貯體中，選擇管理索引標籤，然後選擇新增生命週期規則。 4. 在生命週期規則對話方塊中輸入規則的名稱，並為儲存貯體設定 30 天的生命週期規則。	雲端架構師、應用程式開發人員

建立 Firehose 交付串流

任務	Description	所需的技能
建立和設定 Firehose 交付串流。	從 GitHub 儲存庫下載和編輯`CreateFireHoseToS3.py` 程式碼範例。 此程式碼是以 Python 撰寫，並說明如何建立 Firehose 交付串流和 AWS Identity and Access Management (IAM) 角色。IAM 角色會有政策，可供 Firehose 用來寫入目的地 S3 儲存貯體。 若要執行指令碼，請使用下列命令和命令列引數。 引數 1 = <Your_S3_bucket_ARN> ，這是您先前建立之儲存貯體的 Amazon Resource Name (ARN) 引數 2 = 您的 Firehose 名稱（此試驗正在使用`firehose_to_s3_stream` 。） 引數 3 = 您的 IAM 角色名稱（此試驗使用 `firehose_to_s3` 。） `python CreateFireHoseToS3.py <Your_S3_Bucket_ARN> firehose_to_s3_stream firehose_to_s3`	雲端架構師、應用程式開發人員

任務	Description	所需的技能
	如果指定的 IAM 角色不存在，指令碼會建立具有信任關係政策的擔任角色，以及授予足夠 Amazon S3 許可的政策。如需這些政策的範例，請參閱其他資訊一節。	
驗證 Firehose 交付串流。	使用 AWS CLI 來驗證已成功建立交付串流，以描述 Firehose 交付串流。 `aws firehose describe-delivery-stream --delivery-stream-name firehose_to_s3_stream`	雲端架構師、應用程式開發人員

建立 Lambda 函數來處理 Firehose 交付串流

任務	Description	所需的技能
建立 Lambda 函數的信任政策。	使用下列資訊建立信任政策檔案。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "Service": "lambda.amazonaws.com" },`	雲端架構師、應用程式開發人員

任務	Description	所需的技能

	<pre>"Action":
	"sts:AssumeRole"
	}
	]
	}</pre>
	這可讓您的函數存取 AWS 資源。	

建立 Lambda 函數的執行角色。	若要建立執行角色，請執行下列程式碼。
	<pre>aws iam create-role
	--role-name lambda-
	ex --assume-role-poli
	cy-document file://Tr
	ustPolicy.json</pre>
		雲端架構師、應用程式開發人員

任務	Description	所需的技能

將許可新增至角色。	若要將許可新增至角色，請使用 `attach-policy-to-role` 命令。
	
	`aws iam attach-role-policy --role-name lambda-ex --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole`
	`aws iam attach-role-policy --role-name lambda-ex --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaDynamoDBExecutionRole`
	`aws iam attach-role-policy --role-name lambda-ex --policy-arn arn:aws:iam::aws:policy/AmazonKinesisFirehoseFullAccess`
	`aws iam attach-role-policy --role-name lambda-ex --policy-arn arn:aws:iam::aws:policy/IAMFullAccess`
		雲端架構師、應用程式開發人員

任務	Description	所需的技能

建立 Lambda 函式。	執行下列命令，從程式碼儲存庫壓縮 `LambdaStreamProcessor.py` 檔案。
	`zip function.zip LambdaStreamProcessor.py`
	當您建立 Lambda 函數時，您將需要 Lambda 執行角色 ARN。若要取得 ARN，請執行下列程式碼。
	`aws iam get-role \ --role-name lambda-ex`
	若要建立 Lambda 函數，請執行下列程式碼。
	`# Review the environment variables and replace them with your values.`
	`aws lambda create-function --function-name LambdaStreamProcessor \ --zip-file fileb://function.zip --handler LambdaStreamProcessor.handler --runtime python3.8 \ --role {Your Lamda Execution Role ARN}\ --environment Variables="{firehose_name=firehose_t`
		雲端架構師、應用程式開發人員

任務	Description	所需的技能
	`o_s3_stream,bucket_arn = <Your_S3_bucket_ARN>,iam_role_name = firehose_to_s3, batch_size=400}"`	
設定 Lambda 函數觸發。	使用 AWS CLI 來設定觸發 (DynamoDB Streams)，這會叫用 Lambda 函數。400 的批次大小是為了避免在 Lambda 並行問題中執行。 `aws lambda create-event-source-mapping --function-name LambdaStreamProcessor \ --batch-size 400 --starting-position LATEST \ --event-source-arn <Your Latest Stream ARN From DynamoDB Console>`	雲端架構師、應用程式開發人員

測試功能

任務	Description	所需的技能
將具有過期時間戳記的項目新增至保留資料表。	若要測試功能，請將具有過期 epoch 時間戳記的項目新增至`Reservation` 資料表。TTL 會根據時間戳記自動刪除項目。 Lambda 函數會在 DynamoDB Stream 活動上啟動，並篩選事件以識別`REMOVE`活動或刪除	雲端架構師

任務	Description	所需的技能
	的項目。然後，它會以批次格式將記錄放入 Firehose 交付串流。 Firehose 交付串流會使用 `firehosetos3example/year=current year/month=current month/day=current day/hour=current hour/` 字首將項目傳輸到目的地 S3 儲存貯體。 **Important** 若要最佳化資料擷取，請使用 `Prefix`和設定 Amazon `S3ErrorOutputPrefix` ，詳細資訊請參閱其他資訊一節。	

清除資源

任務	Description	所需的技能
刪除所有資源。	刪除所有資源，以確保不會針對您未使用的任何服務向您收費。	雲端架構師、應用程式開發人員

相關資源

- 管理儲存生命週期

- Amazon S3 儲存類別
- 適用於 Python 的 AWS 開發套件 (Boto3) 文件

其他資訊

建立和設定 Firehose 交付串流 – 政策範例

Firehose 信任關係政策範例文件

```
firehose_assume_role = {
        'Version': '2012-10-17',
        'Statement': [
            {
                'Sid': '',
                'Effect': 'Allow',
                'Principal': {
                    'Service': 'firehose.amazonaws.com'
                },
                'Action': 'sts:AssumeRole'
            }
        ]
    }
```

S3 許可政策範例

```
s3_access = {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Sid": "",
                "Effect": "Allow",
                "Action": [
                    "s3:AbortMultipartUpload",
                    "s3:GetBucketLocation",
                    "s3:GetObject",
                    "s3:ListBucket",
                    "s3:ListBucketMultipartUploads",
                    "s3:PutObject"
                ],
                "Resource": [
                    "{your s3_bucket ARN}/*",
                    "{Your s3 bucket ARN}"
```

```
                ]
            }
        ]
    }
```

測試功能 – Amazon S3 組態

Amazon S3 組態具有下列特性Prefix，且ErrorOutputPrefix選擇此組態來最佳化資料擷取。

prefix

```
firehosetos3example/year=! {timestamp: yyyy}/month=! {timestamp:MM}/day=!
 {timestamp:dd}/hour=!{timestamp:HH}/
```

Firehose 首先會在 S3 儲存貯體firehosetos3example下直接建立名為 的基本資料夾。然後，它會使用 Java DateTimeFormatter 格式評估表達式 !{timestamp:yyyy}、!{timestamp:dd}、 !{timestamp:MM}和 !{timestamp:HH} 到年、月、日和小時。

例如，Unix epoch 時間中 1604683577 的大致到達時間戳記會評估為 year=2020、day=06、month=11和 hour=05。因此，Amazon S3 中交付資料記錄的位置會評估為 firehosetos3example/year=2020/month=11/day=06/hour=05/。

ErrorOutputPrefix

```
firehosetos3erroroutputbase/!{firehose:random-string}/!{firehose:error-output-type}/!
{timestamp:yyyy/MM/dd}/
```

ErrorOutputPrefix 會產生firehosetos3erroroutputbase直接在 S3 儲存貯體下呼叫的基本資料夾。表達式會!{firehose:random-string}評估為 11 個字元的隨機字串，例如 ztWxkdg3Thg。交付失敗記錄的 Amazon S3 物件位置可以評估為 firehosetos3erroroutputbase/ztWxkdg3Thg/processing-failed/2020/11/06/。

在 Amazon OpenSearch Service 中建置多租戶無伺服器架構

Tabby Ward 和 Nisha Gambhir，Amazon Web Services

總結

Amazon OpenSearch Service 是一種受管服務，可讓您輕鬆部署、操作和擴展 Elasticsearch，這是熱門的開放原始碼搜尋和分析引擎。OpenSearch Service 提供任意文字搜尋，以及近乎即時的串流資料擷取和儀表板，例如日誌和指標。

軟體即服務 (SaaS) 供應商經常使用 OpenSearch Service 來處理各種使用案例，例如以可擴展且安全的方式獲得客戶洞見，同時降低複雜性和停機時間。

在多租戶環境中使用 OpenSearch Service 引入了一系列的考量，會影響 SaaS 解決方案的分割、隔離、部署和管理。SaaS 提供者必須考慮如何使用不斷轉移的工作負載，有效地擴展其 Elasticsearch 叢集。他們還需要考慮分層和嘈雜的鄰國條件如何影響其分割模型。

此模式會檢閱用於使用 Elasticsearch 建構來表示和隔離租戶資料的模型。此外，模式著重於簡單的無伺服器參考架構作為範例，以示範在多租戶環境中使用 OpenSearch Service 進行索引和搜尋。它會實作集區資料分割模型，在所有租用戶之間共用相同的索引，同時維持租用戶的資料隔離。此模式使用下列 AWS 服務：Amazon API Gateway AWS Lambda、Amazon Simple Storage Service (Amazon S3) 和 OpenSearch Service。

如需集區模型和其他資料分割模型的詳細資訊，請參閱其他資訊一節。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- AWS Command Line Interface (AWS CLI) 2.x 版，在 macOS、Linux 或 Windows 上安裝和設定
- Python 3.9 版
- pip3 – Python 原始程式碼以要部署在 Lambda 函數中的 .zip 檔案提供。如果您想要在本機使用或自訂程式碼，請依照下列步驟開發和重新編譯原始程式碼：
  1. 在與 Python 指令碼相同的目錄中執行下列命令來產生`requirements.txt`檔案： `pip3 freeze > requirements.txt`
  2. 安裝相依性： `pip3 install -r requirements.txt`

限制

- 此程式碼在 Python 中執行，目前不支援其他程式設計語言。
- 範例應用程式不包含 AWS 跨區域或災難復原 (DR) 支援。
- 此模式僅供示範之用。它不適用於生產環境。

Architecture

下圖說明此模式的高階架構。架構包含下列項目：

- Lambda 索引和查詢內容
- OpenSearch Service 執行搜尋
- API Gateway 提供與 使用者的 API 互動
- Amazon S3 儲存原始（非索引）資料
- Amazon CloudWatch 監控日誌
- AWS Identity and Access Management (IAM) 以建立租戶角色和政策

自動化和擴展

為了簡化， 模式會使用 AWS CLI 來佈建基礎設施和部署範例程式碼。您可以建立 CloudFormation 範本或 AWS Cloud Development Kit (AWS CDK) 指令碼來自動化模式。

工具

AWS 服務

- AWS CLI 是統一的工具，可讓您在命令列 Shell 中使用命令來管理 AWS 服務 和 資源。
- Lambda 是一種運算服務，可讓您執行程式碼，而無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。
- API Gateway 是 ， AWS 服務 用於建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。
- Amazon S3 是一種物件儲存服務，可讓您隨時從 Web 上的任何位置存放和擷取任意數量的資訊。
- OpenSearch Service 是一項全受管服務，可讓您以經濟實惠的方式大規模部署、保護和執行 Elasticsearch。

Code

附件提供此模式的範例檔案。其中包含：

- `index_lambda_package.zip` – 使用集區模型為 OpenSearch Service 中的資料編製索引的 Lambda 函數。
- `search_lambda_package.zip` – 在 OpenSearch Service 中搜尋資料的 Lambda 函數。
- `Tenant-1-data` – Tenant-1 的原始（非索引）資料範例。
- `Tenant-2-data` – Tenant-2 的原始（非索引）資料範例。

> **Important**
>
> 此模式中的故事包括針對 Unix、Linux 和 macOS 格式化的 AWS CLI 命令範例。用於 Windows 時，請以插入號 (^) 取代每一行結尾處的 Unix 接續字元斜線 (\)。

> **Note**
>
> 在 AWS CLI 命令中，將角括號 (<>) 內的所有值取代為正確的值。

史詩

建立和設定 S3 儲存貯體

任務	Description	所需的技能
建立 S3 儲存貯體。	在 中建立 S3 儲存貯體 AWS 區域。此儲存貯體會保留範例應用程式的非索引租用戶資料。請確定 S3 儲存貯體的名稱是全域唯一的，因為命名空間是由所有 共用 AWS 帳戶。 若要建立 S3 儲存貯體，您可以使用 AWS CLI create-bucket 命令，如下所示：	雲端架構師、雲端管理員

任務	Description	所需的技能
	`aws s3api create-bucket \ --bucket <tenantrawdata> \ --region <your-AWS-Region>` 其中 `tenantrawdata` 是 S3 儲存貯體名稱。（您可以使用遵循儲存貯體命名準則的任何唯一名稱。）	

建立和設定 Elasticsearch 叢集

任務	Description	所需的技能
建立 OpenSearch Service 網域	執行 AWS CLI create-elasticsearch-domain 命令來建立 OpenSearch Service 網域： `aws es create-elasticsearch-domain \ --domain-name vpc-cli-example \ --elasticsearch-version 7.10 \ --elasticsearch-cluster-config InstanceType=t3.medium.elasticsearch,InstanceCount=1 \ --ebs-options EBSEnabled=true,VolumeType=gp2,VolumeSize=10 \`	雲端架構師、雲端管理員

任務	Description	所需的技能
	`--domain-endpoint-options "{\"EnforceHTTPS\": true}" \ --encryption-at-rest-options "{\"Enabled\": true}" \ --node-to-node-encryption-options "{\"Enabled\": true}" \ --advanced-security-options "{\"Enabled\": true, \"InternalUserDatabaseEnabled\": true, \ \"MasterUserOptions\": {\"MasterUserName\": \"KibanaUser\", \ \"MasterUserPassword\": \"NewKibanaPassword@123\"}}" \ --vpc-options "{\"SubnetIds\": [\"<subnet-id>\"], \"SecurityGroupIds\": [\"<sg-id>\"]}" \ --access-policies "{\"Version\": \"2012-10-17\", \"Statement\": [{ \"Effect\": \"Allow\", \ \"Principal\": {\"AWS\": \"*\" }, \"Action\":\"es:*\", \ \"Resource\": \"arn:aws:es:<region>:<account-id>:domain\/vpc-cli-example\/*\" }] }"`	

任務	Description	所需的技能
	執行個體計數設定為 1，因為網域用於測試目的。您需要使用 `advanced-security-options` 參數啟用精細存取控制，因為在建立網域之後無法變更詳細資訊。 此命令會建立主要使用者名稱 (`KibanaUser`) 和密碼，您可以用來登入 Kibana 主控台。 由於網域是虛擬私有雲端 (VPC) 的一部分，因此您必須指定要使用的存取政策，以確保可以連接 Elasticsearch 執行個體。 如需詳細資訊，請參閱 AWS 文件中的在 VPC 內啟動 Amazon OpenSearch Service 網域。	

任務	Description	所需的技能

設定堡壘主機。	將 Amazon Elastic Compute Cloud (Amazon EC2) Windows 執行個體設定為堡壘主機，以存取 Kibana 主控台。Elasticsearch 安全群組必須允許來自 Amazon EC2 安全群組的流量。如需說明，請參閱部落格文章使用堡壘伺服器控制 EC2 執行個體的網路存取。
	
	當堡壘主機已設定，且您有與執行個體相關聯的安全群組可用時，請使用 AWS CLI authorize-security-group-ingress 命令將許可新增至 Elasticsearch 安全群組，以允許來自 Amazon EC2（堡壘主機）安全群組的連接埠 443。
	
	`aws ec2 authorize-security-group-ingress \`
	`  --group-id <SecurityGroupIdfElasticSearch> \`
	`  --protocol tcp \`
	`  --port 443 \`
	`  --source-group <SecurityGroupIdfBashionHostEC2>`
		雲端架構師、雲端管理員

建立和設定 Lambda 索引函數

任務	Description	所需的技能

建立 Lambda 執行角色。	執行 AWS CLI createe-role 命令，以授予 Lambda 索引函數對 AWS 服務 和 資源的存取權：
	`aws iam create-role \ --role-name index-lambda-role \ --assume-role-policy-document file://lambda_assume_role.json`
	其中 `lambda_assume_role.json` 是將`AssumeRole` 許可授予 Lambda 函數的 JSON 文件，如下所示：
	`{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "Service": "lambda.amazonaws.com" }, "Action": "sts:AssumeRole" } ] }`
		雲端架構師、雲端管理員

任務	Description	所需的技能

將受管政策連接至 Lambda 角色。	執行 AWS CLI attach-role-policy 命令，將受管政策連接到上一個步驟中建立的角色。這兩個政策提供角色建立彈性網路界面和將日誌寫入 CloudWatch Logs 的許可。
	
	`aws iam attach-role-policy \`
	`  --role-name index-lambda-role \`
	`  --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole`
	
	`aws iam attach-role-policy \`
	`  --role-name index-lambda-role \`
	`  --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaVPCAccessExecutionRole`
		雲端架構師、雲端管理員

任務	Description	所需的技能

建立政策以授予 Lambda 索引函數讀取 S3 物件的許可。	對 執行 the AWS CLI create-policy 命令，授予 Lambda 索引函數讀取 S3 儲存貯體中物件的`s3:GetObject` 許可：
	`aws iam create-policy \ --policy-name s3-permission-policy \ --policy-document file://s3-policy.json`
	檔案`s3-policy.json` 是如下所示的 JSON 文件，授予`s3:GetObject` 許可以允許對 S3 物件的讀取存取。如果您在建立 S3 儲存貯體時使用不同的名稱，請在 `Resource` 區段中提供正確的儲存貯體名稱：
	`{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "s3:GetObject", "Resource": "arn:aws:s3:::<tenantrawdata>/*" } ] }`
		雲端架構師、雲端管理員

任務	Description	所需的技能

將 Amazon S3 許可政策連接至 Lambda 執行角色。	執行 AWS CLI attach-role-policy 命令，將您在上一個步驟中建立的 Amazon S3 許可政策連接至 Lambda 執行角色：
	`aws iam attach-role-policy \`
	`  --role-name index-lambda-role \`
	`  --policy-arn <PolicyARN>`
	其中 `PolicyARN` 是 Amazon S3 許可政策的 Amazon Resource Name (ARN)。您可以從上一個命令的輸出取得此值。
		雲端架構師、雲端管理員

任務	Description	所需的技能

建立 Lambda 索引函數。	執行 AWS CLI createe-function 命令來建立 Lambda 索引函數，這會存取 OpenSearch Service：
	
	`aws lambda create-function \`
	`  --function-name index-lambda-function \`
	`  --zip-file fileb://index_lambda_package.zip \`
	`  --handler lambda_index.lambda_handler \`
	`  --runtime python3.9 \`
	`  --role "arn:aws:iam::account-id:role/index-lambda-role" \`
	`  --timeout 30 \`
	`  --vpc-config "{\"SubnetIds\": [\"<subnet-id1\>", \"<subnet-id2>\"], \`
	`  \"SecurityGroupIds\": [\"<sg-1>\"]}"`
		雲端架構師、雲端管理員

任務	Description	所需的技能

允許 Amazon S3 呼叫 Lambda 索引函數。	執行 AWS CLI add-permission 命令，給予 Amazon S3 呼叫 Lambda 索引函數的許可：
	
	`aws lambda add-permission \`
	`  --function-name index-lambda-function \`
	`  --statement-id s3-permissions \`
	`  --action lambda:InvokeFunction \`
	`  --principal s3.amazonaws.com \`
	`  --source-arn "arn:aws:s3:::<tenantrawdata>" \`
	`  --source-account "<account-id>"`
		雲端架構師、雲端管理員

任務	Description	所需的技能
為 Amazon S3 事件新增 Lambda 觸發條件。	執行 AWS CLI put-bucket-notification-configuration 命令，以在偵測到 Amazon S3 `ObjectCreated` 事件時傳送通知至 Lambda 索引函數。索引函數會在物件上傳到 S3 儲存貯體時執行。 `aws s3api put-bucket-notification-configuration \ --bucket <tenantrawdata> \ --notification-configuration file://s3-trigger.json` 檔案`s3-trigger.json` 是目前資料夾中的 JSON 文件，會在 Amazon S3 `ObjectCreated` 事件發生時將資源政策新增至 Lambda 函數。	雲端架構師、雲端管理員

建立和設定 Lambda 搜尋函數

任務	Description	所需的技能
建立 Lambda 執行角色。	執行 AWS CLI createe-role 命令，以授予 Lambda 搜尋函數對 AWS 服務 和 資源的存取權： `aws iam create-role \ --role-name search-lambda-role \`	雲端架構師、雲端管理員

任務	Description	所需的技能

	--assume-role-policy-document file://lambda_assume_role.json
	
	其中 lambda_assume_role.json 是目前資料夾中的 JSON 文件，授予 Lambda 函數AssumeRole 許可，如下所示：
	
	{
	"Version": "2012-10-17",
	"Statement": [
	{
	"Effect": "Allow",
	"Principal": {
	"Service": "lambda.amazonaws.com"
	},
	"Action": "sts:AssumeRole"
	}
	]
	}	

任務	Description	所需的技能

將受管政策連接至 Lambda 角色。	執行 AWS CLI attach-role-policy 命令，將受管政策連接至上一個步驟中建立的角色。這兩個政策提供角色建立彈性網路界面和將日誌寫入 CloudWatch Logs 的許可。
	
	`aws iam attach-role-policy \`
	`  --role-name search-lambda-role \`
	`  --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole`
	
	`aws iam attach-role-policy \`
	`  --role-name search-lambda-role \`
	`  --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaVPCAccessExecutionRole`
		雲端架構師、雲端管理員

任務	Description	所需的技能
建立 Lambda 搜尋函數。	執行 AWS CLI createe-function 命令來建立 Lambda 搜尋函數，這會存取 OpenSearch Service： `aws lambda create-function \ --function-name search-lambda-function \ --zip-file fileb://search_lambda_package.zip \ --handler lambda_search.lambda_handler \ --runtime python3.9 \ --role "arn:aws:iam::account-id:role/search-lambda-role" \ --timeout 30 \ --vpc-config "{\"SubnetIds\": [\"<subnet-id1\>", \"<subnet-id2>\"], \ \"SecurityGroupIds\": [\"<sg-1>\"]}"`	雲端架構師、雲端管理員

建立和設定租戶角色

任務	Description	所需的技能
建立租戶 IAM 角色。	執行 the AWS CLI create-role 命令來建立兩個租用戶角色，用於測試搜尋功能： `aws iam create-role \`	雲端架構師、雲端管理員

任務	Description	所需的技能

	<pre> --role-name Tenant-1-role \
	  --assume-role-policy-document file://assume-role-policy.json</pre>
	<pre>aws iam create-role \
	  --role-name Tenant-2-role \
	  --assume-role-policy-document file://assume-role-policy.json</pre>
	檔案assume-role-policy.json 是目前資料夾中的 JSON 文件，可將AssumeRole 許可授予 Lambda 執行角色：
	<pre>{
	    "Version": "2012-10-17",
	    "Statement": [
	        {
	            "Effect": "Allow",
	            "Principal": {
	                "AWS": "<Lambda execution role for index function>",
	                "AWS": "<Lambda execution role for search function>"
	            },
	            "Action": "sts:AssumeRole"
	        }
	    ]</pre>	

任務	Description	所需的技能
	} }	

任務	Description	所需的技能

建立租戶 IAM 政策。	執行 AWS CLI createe-policy 命令來建立租用戶政策，以授予對 Elasticsearch 操作的存取權：
	
	`aws iam create-policy \ --policy-name tenant-policy \ --policy-document file://policy.json`
	
	檔案`policy.json` 是目前資料夾中的 JSON 文件，可授予 Elasticsearch 的許可：
	
	`{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "es:ESHttpDelete", "es:ESHttpGet", "es:ESHttpHead", "es:ESHttpPost", "es:ESHttpPut", "es:ESHttpPatch" ], "Resource": [`
		雲端架構師、雲端管理員

任務	Description	所需的技能

	<pre>"<ARN of Elasticsearch domain created earlier>"
	]
	}
	]
	}</pre>	

將租戶 IAM 政策連接至租戶角色。	執行 AWS CLI attach-role-policy 命令，將租用戶 IAM 政策連接至您在先前步驟中建立的兩個租用戶角色：
	<pre>aws iam attach-role-policy \
	--policy-arn arn:aws:iam::account-id:policy/tenant-policy \
	--role-name Tenant-1-role
	
	aws iam attach-role-policy \
	--policy-arn arn:aws:iam::account-id:policy/tenant-policy \
	--role-name Tenant-2-role</pre>
	政策 ARN 來自上一個步驟的輸出。
		雲端架構師、雲端管理員

任務	Description	所需的技能

建立 IAM 政策，授予 Lambda 擔任角色的許可。	執行 the AWS CLI create-policy 命令，為 Lambda 建立政策以擔任租戶角色：
	
	`aws iam create-policy \ --policy-name assume-tenant-role-policy \ --policy-document file://lambda_policy.json`
	
	檔案`lambda_policy.json` 是目前資料夾中的 JSON 文件，授予 許可`AssumeRole` ：
	
	`{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "sts:AssumeRole", "Resource": "<ARN of tenant role created earlier>" } ] }`
	
	對於 `Resource`，您可以使用萬用字元來避免為每個租用戶建立新的政策。
		雲端架構師、雲端管理員

任務	Description	所需的技能

建立 IAM 政策，授予 Lambda 索引角色存取 Amazon S3 的許可。	執行 AWS CLI createe-policy 命令，授予 Lambda 索引角色存取 S3 儲存貯體中物件的許可：
	`aws iam create-policy \ --policy-name s3-permission-policy \ --policy-document file://s3_lambda_policy.json`
	檔案`s3_lambda_policy.json` 是目前資料夾中的下列 JSON 政策文件：
	`{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "s3:GetObject", "Resource": "arn:aws:s3:::tenantrawdata/*" } ] }`
		雲端架構師、雲端管理員

任務	Description	所需的技能

將政策連接至 Lambda 執行角色。	執行 AWS CLI attach-role-policy 命令，將上一個步驟中建立的政策連接至您先前建立的 Lambda 索引和搜尋執行角色：
	
	`aws iam attach-role-policy \`
	`  --policy-arn arn:aws:iam::account-id:policy/assume-tenant-role-policy \`
	`  --role-name index-lambda-role`
	
	`aws iam attach-role-policy \`
	`  --policy-arn arn:aws:iam::account-id:policy/assume-tenant-role-policy \`
	`  --role-name search-lambda-role`
	
	`aws iam attach-role-policy \`
	`  --policy-arn arn:aws:iam::account-id:policy/s3-permission-policy \`
	`  --role-name index-lambda-role`
	
	政策 ARN 來自上一個步驟的輸出。
		雲端架構師、雲端管理員

建立和設定搜尋 API

任務	Description	所需的技能

在 API Gateway 中建立 REST API。	執行 AWS CLI create-rest-api 命令來建立 REST API 資源：
	
	`aws apigateway create-rest-api \`
	`  --name Test-Api \`
	`  --endpoint-configu ration "{ \"types\": [\"REGIONAL\"] }"`
	
	對於端點組態類型，您可以指定 EDGE，而不是REGIONAL使用節點，而不是特定節點 AWS 區域。
	
	請注意命令輸出中 id 欄位的值。這是您將在後續命令中使用的 API ID。
		雲端架構師、雲端管理員

建立搜尋 API 的資源。	搜尋 API 資源會以資源名稱 啟動 Lambda 搜尋函數search。（您不需要為 Lambda 索引函數建立 API，因為它會在物件上傳至 S3 儲存貯體時自動執行。）
	
	1. 執行 the AWS CLI get-resources 命令以取得根路徑的父系 ID：
	
	`aws apigateway get-resources \`
	`  --rest-api-id <API-ID>`
		雲端架構師、雲端管理員

任務	Description	所需的技能

	請記下 ID 欄位的值。您將在下一個命令中使用此父 ID。
	
	`{ "items": [ { "id": "zpsri964ck", "path": "/" } ] }`
	
	2. 執行 AWS CLI create-resource 命令來建立搜尋 API 的資源。針對 `parent-id` ，指定上一個命令的 ID。
	
	`aws apigateway create-resource \ --rest-api-id <API-ID> \ --parent-id <Parent-ID> \ --path-part search`	

任務	Description	所需的技能

建立搜尋 API 的 GET 方法。	執行 AWS CLI put-method 命令來建立搜尋 API GET 的方法：
	
	`aws apigateway put-method \`
	`  --rest-api-id <API-ID> \`
	`  --resource-id <ID from the previous command output> \`
	`  --http-method GET \`
	`  --authorization-type "NONE" \`
	`  --no-api-key-required`
	
	針對 `resource-id` ，指定來自 `create-resource` 命令輸出的 ID。
		雲端架構師、雲端管理員

任務	Description	所需的技能
建立搜尋 API 的方法回應。	執行 AWS CLI put-method-response 命令，為搜尋 API 新增方法回應： `aws apigateway put-method-response \ --rest-api-id <API-ID> \ --resource-id <ID from the create-resource command output> \ --http-method GET \ --status-code 200 \ --response-models "{\"application/json\": \"Empty\"}"` 針對 `resource-id` ，指定先前`create-resource` 命令輸出的 ID。	雲端架構師、雲端管理員

任務	Description	所需的技能

設定搜尋 API 的代理 Lambda 整合。	執行 AWS CLI put-integration 命令來設定與 Lambda 搜尋函數的整合：
	
	`aws apigateway put-integration \`
	`  --rest-api-id <API-ID> \`
	`  --resource-id  <ID from the create-resource command output> \`
	`  --http-method GET \`
	`  --type AWS_PROXY \`
	`  --integration-http-method GET \`
	`  --uri arn:aws:apigateway:region:lambda:path/2015-03-31/functions/arn:aws:lambda:<region>:<account-id>:function:<function-name>/invocations`
	
	針對 `resource-id` ，指定先前`create-resource` 命令的 ID。
		雲端架構師、雲端管理員

任務	Description	所需的技能
授予 API Gateway 呼叫 Lambda 搜尋函數的許可。	執行 AWS CLI add-permission 命令，給予 API Gateway 使用搜尋函數的許可： `aws lambda add-permission \ --function-name <function-name> \ --statement-id apigateway-get \ --action lambda:InvokeFunction \ --principal apigateway.amazonaws.com \ --source-arn "arn:aws:execute-api:<region>:<account-id>:api-id/*/GET/search"` 如果您使用不同的 API 資源名稱而非 ，請變更`source-arn` 路徑`search`。	雲端架構師、雲端管理員
部署搜尋 API。	執行 the AWS CLI create-deployment 命令來建立名為 的階段資源`dev`： `aws apigateway create-deployment \ --rest-api-id <API-ID> \ --stage-name dev` 如果您更新 API，您可以使用相同的 AWS CLI 命令將其重新部署到相同的階段。	雲端架構師、雲端管理員

建立和設定 Kibana 角色

任務	Description	所需的技能
登入 Kibana 主控台。	1. 在 OpenSearch Service 主控台的網域儀表板上尋找 Kibana 的連結。URL 的格式為：<domain-endpoint>/_plugin/kibana/ 。 2. 使用您在第一個 epic 中設定的堡壘主機來存取 Kibana 主控台。 3. 當您建立 OpenSearch Service 網域時，使用先前步驟的主要使用者名稱和密碼登入 Kibana 主控台。 4. 出現選取租用戶的提示時，請選擇私有。	雲端架構師、雲端管理員
建立和設定 Kibana 角色。	若要提供資料隔離並確保一個租用戶無法擷取另一個租用戶的資料，您需要使用文件安全，這允許租用戶僅存取包含其租用戶 ID 的文件。 1. 在 Kibana 主控台的導覽窗格中，選擇安全性、角色。 2. 建立新的租戶角色。 3. 將叢集許可設定為 indices_all ，這會為 OpenSearch Service 索引提供建立、讀取、更新和刪除 (CRUD) 許可。 4. 限制索引的tenant-data 索引許可。（索引名稱	雲端架構師、雲端管理員

任務	Description	所需的技能
	應與 Lambda 搜尋和索引函數中的名稱相符。) 5. 將索引許可設定為 `indices_all` ，讓使用者能夠執行所有與索引相關的操作。（視您的需求而定，您可以限制更精細的存取操作。) 6. 為了文件層級安全，請使用下列政策依租用戶 ID 篩選文件，為共用索引中的租用戶提供資料隔離： `{ "bool": { "must": { "match": { "TenantId": "Tenant-1" } } } }` 索引名稱、屬性和值區分大小寫。	

任務	Description	所需的技能
將使用者映射至角色。	1. 選擇角色的映射使用者索引標籤，然後選擇映射使用者。 2. 在後端角色區段中，指定您先前建立之 IAM 租用戶角色的 ARN，然後選擇映射。這會將 IAM 租用戶角色映射到 Kibana 角色，以便租用戶特定的搜尋僅傳回該租用戶的資料。例如，如果Tenant-1 的 IAM 角色名稱為 `Tenant-1-Role` ，請在租用戶-1 Kibana 角色的後端角色方塊中指定 `Tenant-1-Role` （從建立和設定租用戶角色 epic) 的 ARN。 Tenant-1 3. 針對Tenant-2。 我們建議您在租戶加入時自動建立租戶和 Kibana 角色。	雲端架構師、雲端管理員

任務	Description	所需的技能
建立租用戶資料索引。	在導覽窗格的管理下，選擇開發工具，然後執行下列命令。此命令會建立tenant-data 索引來定義 TenantId 屬性的映射。 `PUT /tenant-data { "mappings": { "properties": { "TenantId": { "type": "keyword"} } } }`	雲端架構師、雲端管理員

為 Amazon S3 和 建立 VPC 端點 AWS STS

任務	Description	所需的技能
為 Amazon S3 建立 VPC 端點。	執行 AWS CLI create-vpc-endpoint 命令，為 Amazon S3 建立 VPC 端點。端點可讓 VPC 中的 Lambda 索引函數存取 Amazon S3。 `aws ec2 create-vpc-endpoint \ --vpc-id <VPC-ID> \ --service-name com.amazonaws.us-east-1.s3 \ --route-table-ids <route-table-ID>`	雲端架構師、雲端管理員

任務	Description	所需的技能
	針對 `vpc-id`，指定您用於 Lambda 索引函數的 VPC。對於 `service-name` ，請使用 Amazon S3 端點的正確 URL。針對 `route-table-ids`，指定與 VPC 端點相關聯的路由表。	

任務	Description	所需的技能
為 建立 VPC 端點 AWS STS。	執行 AWS CLI create-vpc-endpoint 命令來建立 AWS Security Token Service () 的 VPC 端點AWS STS。端點可讓 VPC 中的 Lambda 索引和搜尋函數存取 AWS STS。函數在擔任 IAM 角色 AWS STS 時使用。 `aws ec2 create-vpc-endpoint \ --vpc-id <VPC-ID> \ --vpc-endpoint-type Interface \ --service-name com.amazonaws.us-east-1.sts \ --subnet-id <subnet-ID> \ --security-group-id <security-group-ID>` 針對 `vpc-id`，指定您用於 Lambda 索引和搜尋函數的 VPC。針對 `subnet-id` ，提供應建立此端點的子網路。針對 `security-group-id` ，指定要與此端點建立關聯的安全群組。（它可以與 Lambda 使用的安全群組相同。）	雲端架構師、雲端管理員

測試多租戶和資料隔離

任務	Description	所需的技能
更新索引和搜尋函數的 Python 檔案。	1. 在 `index_lambda_package.zip` 檔案中，編輯 `lamba_index.py` 檔案以更新 AWS 帳戶 ID AWS 區域和 Elasticsearch 端點資訊。 2. 在 `search_lambda_package.zip` 檔案中，編輯 `lambda_search.py` 檔案以更新 AWS 帳戶 ID AWS 區域和 Elasticsearch 端點資訊。 您可以從 OpenSearch Service 主控台的概觀索引標籤取得 Elasticsearch 端點。 OpenSearch 其格式為 `<AWS-Region>.es.amazonaws.com` 。	雲端架構師、應用程式開發人員

更新 Lambda 程式碼。	使用 AWS CLI update-function-code 命令，使用您對 Python 檔案所做的變更來更新 Lambda 程式碼：
	
	`aws lambda update-function-code \`
	`  --function-name index-lambda-function \`
	`  --zip-file fileb://index_lambda_package.zip`
		雲端架構師、應用程式開發人員

任務	Description	所需的技能
	`aws lambda update-function-code \ --function-name search-lambda-function \ --zip-file fileb://search_lambda_package.zip`	

將原始資料上傳至 S3 儲存貯體。	使用 the AWS CLI cp 命令將 Tenant-1 和Tenant-2 物件的資料上傳至儲存`tenantrawdata` 貯體 （指定您為此目的建立的 S3 儲存貯體名稱）：
	`aws s3 cp tenant-1-data s3://tenantrawdata`
	`aws s3 cp tenant-2-data s3://tenantrawdata`
	S3 儲存貯體設定為在上傳資料時執行 Lambda 索引函數，以便在 Elasticsearch 中編製文件索引。
		雲端架構師、雲端管理員

從 Kibana 主控台搜尋資料。	在 Kibana 主控台上執行下列查詢： `GET tenant-data/_search` 此查詢會顯示在 Elasticsearch 中編製索引的所有文件。在此情況下，您應該會看到兩個單獨的Tenant-1 和Tenant-2 文件。	雲端架構師、雲端管理員

任務	Description	所需的技能

從 API Gateway 測試搜尋 API。	1. 在 API Gateway 主控台中，開啟搜尋 API，在搜尋資源中選擇GET方法，然後選擇測試。
	2. 在測試視窗中，為租戶 ID 提供下列查詢字串（區分大小寫），然後選擇測試。
	`TenantId=Tenant-1`
	Lambda 函數會將查詢傳送至 OpenSearch Service，以根據文件層級的安全性篩選租戶文件。方法會傳回屬於 Tenant-1 的文件。
	3. 將查詢字串變更為：
	`TenantId=Tenant-2`
	此查詢會傳回屬於Tenant-2的文件。
	如需畫面圖例，請參閱其他資訊一節。
		雲端架構師、應用程式開發人員

清除資源。	清除您建立的所有資源，以防止您的 帳戶產生額外費用。	AWS DevOps、雲端架構師、雲端管理員

相關資源

- 適用於 Python (Boto) 的 AWS SDK
- AWS Lambda 文件
- API Gateway 文件

- Amazon S3 文件
- Amazon OpenSearch Service 文件
  - Amazon OpenSearch Service 中的精細存取控制
  - 使用 Amazon OpenSearch Service 建立搜尋應用程式
  - 在 VPC 中啟動 Amazon OpenSearch Service 網域

其他資訊

資料分割模型

多租戶系統中使用三種常見的資料分割模型：孤立、集區和混合。您選擇的模型取決於您環境的合規、雜訊鄰近、操作和隔離需求。

Silo 模型

在孤島模型中，每個租用戶的資料會存放在不同的儲存區域中，其中不會混合租用戶資料。您可以使用兩種方法來使用 OpenSearch Service 實作孤立模型：每個租用戶的網域和每個租用戶的索引。

- 每個租用戶的網域 – 您可以為每個租用戶使用單獨的 OpenSearch Service 網域（與 Elasticsearch 叢集同義）。將每個租用戶放在自己的網域中，可提供與在獨立建構中擁有資料相關聯的所有優點。不過，這種方法帶來了管理和敏捷性的挑戰。其分散式性質使彙整和評估租戶的運作狀態和活動更加困難。這是一個昂貴的選項，要求每個 OpenSearch Service 網域至少擁有三個主節點和兩個資料節點供生產工作負載使用。

- 每個租用戶的索引 – 您可以將租用戶資料放在 OpenSearch Service 叢集中的個別索引中。透過此方法，您可以在建立和命名索引時使用租用戶識別符，方法是在索引名稱前面加上租用戶識別符。每個租用戶的索引方法可協助您實現孤立目標，而無需為每個租用戶引入完全獨立的叢集。不過，如果索引數量增加，您可能會遇到記憶體壓力，因為這種方法需要更多的碎片，主節點必須處理更多的配置和重新平衡。

孤立模型中的隔離 – 在孤立模型中，您可以使用 IAM 政策來隔離存放每個租戶資料的網域或索引。這些政策可防止一個租用戶存取另一個租用戶的資料。若要實作孤立隔離模型，您可以建立以資源為基

礎的政策，以控制對租戶資源的存取。這通常是網域存取政策，指定委託人可以在網域的子資源上執行的動作，包括 Elasticsearch 索引和 APIs。透過 IAM 身分型政策，您可以在 OpenSearch Service 中的網域、索引或 APIs上指定允許或拒絕的動作。IAM 政策的 `Action`元素說明政策允許或拒絕的特定動作，而 `Principal` 元素指定受影響的帳戶、使用者或角色。

下列範例政策僅授予Tenant-1 對`tenant-1`網域上子資源的完整存取權 （ 如 所指定`es:*`)。`Resource`元素`/*`中的結尾表示此政策適用於網域的子資源，而非網域本身。當此政策生效時，租用戶不得在現有網域上建立新的網域或修改設定。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::<aws-account-id>:user/Tenant-1"
         },
         "Action": "es:*",
         "Resource": "arn:aws:es:<Region>:<account-id>:domain/tenant-1/*"
      }
   ]
}
```

若要實作每個索引孤島模型的租用戶，您需要修改此範例政策，透過指定索引名稱，進一步將Tenant-1 限制為指定的索引或索引。下列範例政策會將Tenant-1 限制為`tenant-index-1`索引。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::123456789012:user/Tenant-1"
         },
         "Action": "es:*",
         "Resource": "arn:aws:es:<Region>:<account-id>:domain/test-domain/tenant-
index-1/*"
      }
   ]
}
```

集區模型

在集區模型中，所有租用戶資料都會存放在相同網域內的索引中。租用戶識別符包含在資料（文件）中，並用作分割區索引鍵，因此您可以判斷哪些資料屬於哪個租用戶。此模型可減少管理開銷。操作和管理集區索引比管理多個索引更簡單且更有效率。不過，由於租戶資料在相同索引中混合，您會失去孤立模型提供的自然租戶隔離。這種方法也可能會因為雜訊鄰近效果而降低效能。

集區模型中的租戶隔離 – 一般而言，在集區模型中實作租戶隔離具有挑戰性。與孤立模型搭配使用的 IAM 機制不允許您根據存放在文件中的租用戶 ID 來描述隔離。

另一種方法是使用 Open Distro for Elasticsearch 提供的精細存取控制 (FGAC) 支援。FGAC 可讓您在索引、文件或欄位層級控制許可。在每個請求中，FGAC 會評估使用者登入資料，並對使用者進行身分驗證或拒絕存取。如果 FGAC 驗證使用者，它會擷取對應至該使用者的所有角色，並使用完整的許可集來判斷如何處理請求。

若要在集區模型中實現所需的隔離，您可以使用文件層級安全性，這可讓您將角色限制為索引中的文件子集。下列範例角色會將查詢限制為Tenant-1。透過將此角色套用至Tenant-1，您可以實現必要的隔離。

```
{
   "bool": {
     "must": {
       "match": {
         "tenantId": "Tenant-1"
       }
     }
   }
 }
```

混合模型

混合模型在相同環境中使用孤立和集區模型的組合，為每個租用戶層（例如免費、標準和高級層）提供獨特的體驗。每個層遵循與集區模型中使用的相同安全性描述檔。

混合模型中的租用戶隔離 – 在混合模型中，您遵循與集區模型相同的安全性設定檔，其中在文件層級使用 FGAC 安全模型提供租用戶隔離。雖然此策略可簡化叢集管理並提供敏捷性，但它會使架構的其他層面變得複雜。例如，您的程式碼需要額外的複雜性，才能判斷哪個模型與每個租用戶相關聯。您也必須確保單一租用戶查詢不會讓整個網域飽和，並降低其他租用戶的體驗。

在 API Gateway 中測試

Tenant-1 查詢的測試時段

Tenant-2 查詢的測試時段

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS CDK 搭配 TypeScript 部署多堆疊應用程式

Rahul Sharad Gaikwad 醫生，Amazon Web Services

總結

此模式提供step-by-step方法。 TypeScript 例如， 模式會部署無伺服器即時分析應用程式。

模式會建置和部署巢狀堆疊應用程式。父 AWS CloudFormation 堆疊會呼叫子堆疊或巢狀堆疊。 每個子堆疊都會建置和部署 CloudFormation 堆疊中定義的 AWS 資源。AWS CDK Toolkit 是命令列界面 (CLI) 命令 `cdk`，是 CloudFormation 堆疊的主要界面。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 現有的虛擬私有雲端 (VPC) 和子網路
- 安裝並設定 AWS CDK Toolkit
- 具有管理員許可和一組存取金鑰的使用者。
- Node.js
- AWS 命令列界面 (AWS CLI)

限制

- 由於 AWS CDK 使用 AWS CloudFormation，AWS CDK 應用程式受限於 CloudFormation 服務配額。如需詳細資訊，請參閱 AWS CloudFormation 配額。

產品版本

此模式已使用下列工具和版本建置和測試。

- AWS CDK Toolkit 1.83.0
- Node.js 14.13.0
- npm 7.0.14

模式應適用於任何版本的 AWS CDK 或 npm。請注意，Node.js 13.0.0 到 13.6.0 版與 AWS CDK 不相容。

Architecture

目標技術堆疊

- AWS Amplify 主控台
- Amazon API Gateway
- AWS CDK
- Amazon CloudFront
- Amazon Cognito
- Amazon DynamoDB
- Amazon Data Firehose
- Amazon Kinesis Data Streams
- AWS Lambda
- Amazon Simple Storage Service (Amazon S3)

目標架構

下圖顯示使用 AWS CDK 搭配 TypeScript 的多堆疊應用程式部署。

下圖顯示範例無伺服器即時應用程式的架構。

工具

工具

- AWS Amplify 主控台是 AWS 中完全堆疊 Web 和行動應用程式部署的控制中心。Amplify 主控台託管提供以 git 為基礎的工作流程，用於託管具有持續部署的完整堆疊無伺服器 Web 應用程式。Admin UI 是前端 Web 和行動開發人員的視覺化界面，可在 AWS 主控台外部建立和管理應用程式後端。
- Amazon API Gateway 是一種 AWS 服務，可用於建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。
- AWS 雲端開發套件 (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。

- AWS CDK Toolkit 是命令列雲端開發套件，可協助您與 AWS CDK 應用程式互動。CLI cdk 命令是與您的 AWS CDK 應用程式互動的主要工具。它會執行您的應用程式、查詢您定義的應用程式模型，以及產生和部署由 AWS CDK 產生的 AWS CloudFormation 範本。
- Amazon CloudFront 是一種 Web 服務，可加速靜態和動態 Web 內容的分佈，例如 .html、.css、.js 和映像檔案。CloudFront 透過稱為節點的全球資料中心網路提供內容，以降低延遲並改善效能。
- Amazon Cognito 為您的 Web 和行動應用程式提供身分驗證、授權和使用者管理。您的使用者可以直接或透過第三方登入。
- Amazon DynamoDB 是全受管的 NoSQL 資料庫服務，可提供快速且可預測的效能和無縫的可擴展性。
- Amazon Data Firehose 是一項全受管服務，可將即時串流資料交付至目的地，例如 Amazon S3、Amazon Redshift、Amazon OpenSearch Service、Splunk，以及受支援的第三方服務供應商擁有的任何自訂 HTTP 端點或 HTTP 端點。
- Amazon Kinesis Data Streams 是一項服務，可即時收集和處理大型資料記錄串流。
- AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。只需為使用的運算時間支付費用，一旦未執行程式碼，就會停止計費。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

Code

此模式的程式碼已連接。

史詩

安裝 AWS CDK Toolkit

任務	Description	所需的技能
安裝 AWS CDK Toolkit。	若要全域安裝 AWS CDK Toolkit，請執行下列命令。 `npm install -g aws-cdk`	DevOps
驗證版本。	若要驗證 AWS CDK Toolkit 版本，請執行下列命令。	DevOps

任務	Description	所需的技能
	`cdk --version`	

設定 AWS 登入資料

任務	Description	所需的技能

設定登入資料。	若要設定登入資料，請執行 `aws configure` 命令並依照提示操作。
	
	`$aws configure`
	`AWS Access Key ID`
	`[None]:`
	`AWS Secret Access Key`
	`[None]: your_secr`
	`et_access_key`
	`Default region name`
	`[None]:`
	`Default output format`
	`[None]:`
		DevOps

下載專案程式碼

任務	Description	所需的技能
下載連接的專案程式碼。	如需 目錄和檔案結構的詳細資訊，請參閱其他資訊一節。	DevOps

引導 AWS CDK 環境

任務	Description	所需的技能
引導環境。	若要將 AWS CloudFormation 範本部署到您要使用的帳戶和 AWS 區域，請執行下列命令。 `cdk bootstrap <account>/<Region>` 如需詳細資訊，請參閱 AWS 文件。	DevOps

建置和部署專案

任務	Description	所需的技能
建置專案。	若要建置專案程式碼，請執行`npm run build`命令。	DevOps
部署專案。	若要部署專案程式碼，請執行`cdk deploy`命令。	

驗證輸出

任務	Description	所需的技能
驗證堆疊建立。	在 AWS 管理主控台上，選擇 CloudFormation。在專案的堆疊中，確認已建立父堆疊和兩個子堆疊。	DevOps

測試應用程式。

任務	Description	所需的技能
將資料傳送至 Kinesis Data Streams。	設定您的 AWS 帳戶，使用 Amazon Kinesis Data Generator (KDG) 將資料傳送至 Kinesis Data Streams。如需詳細資訊，請參閱 Amazon Kinesis Data Generator。	DevOps
建立 Amazon Cognito 使用者。	若要建立 Amazon Cognito 使用者，請從 Kinesis Data Generator 說明頁面上的建立 Amazon Cognito 使用者區段下載 cognito-setup.json CloudFormation 範本。啟動範本，然後輸入您的 Amazon Cognito 使用者名稱和密碼。 Outputs 索引標籤會列出 Kinesis Data Generator URL。	DevOps
登入 Kinesis Data Generator	若要登入 KDG，請使用您提供的 Amazon Cognito 登入資料和 Kinesis Data Generator URL。	DevOps
測試應用程式。	在 KDG 的記錄範本範本 1 中，從其他資訊區段貼上測試碼，然後選擇傳送資料。	DevOps
測試 API Gateway。	擷取資料之後，請使用 GET方法來擷取資料，以測試 API Gateway。	DevOps

相關資源

參考

- AWS 雲端開發套件
- GitHub 上的 AWS CDK
- 使用巢狀堆疊
- AWS 範例 - 無伺服器即時分析

其他資訊

目錄和檔案詳細資訊

此模式會設定下列三個堆疊。

- `parent-cdk-stack.ts` – 此堆疊做為父堆疊，並呼叫兩個子應用程式做為巢狀堆疊。
- `real-time-analytics-poc-stack.ts` – 此巢狀堆疊包含基礎設施和應用程式程式碼。
- `real-time-analytics-web-stack.ts` – 此巢狀堆疊僅包含靜態 Web 應用程式程式碼。

重要檔案及其功能

- `bin/real-time-analytics-poc.ts` – AWS CDK 應用程式的進入點。它會載入 下定義的所有堆疊`lib/`。
- `lib/real-time-analytics-poc-stack.ts` – AWS CDK 應用程式堆疊的定義 (`real-time-analytics-poc`)。
- `lib/real-time-analytics-web-stack.ts` – AWS CDK 應用程式堆疊的定義 (`real-time-analytics-web-stack`)。
- `lib/parent-cdk-stack.ts` – AWS CDK 應用程式堆疊的定義 (`parent-cdk`)。
- `package.json` – npm 模組資訊清單，其中包含應用程式名稱、版本和相依性。
- `package-lock.json` – 由 npm 維護。
- `cdk.json` – 用於執行應用程式的 工具組。
- `tsconfig.json` – 專案的 TypeScript 組態。
- `.gitignore` – Git 應從來源控制中排除的檔案清單。
- `node_modules` – 由 npm 維護；包括專案的相依性。

父堆疊中的下一節程式碼會將子應用程式稱為巢狀 AWS CDK 堆疊。

```
import * as cdk from '@aws-cdk/core';
import { Construct, Stack, StackProps } from '@aws-cdk/core';
import { RealTimeAnalyticsPocStack } from './real-time-analytics-poc-stack';
import { RealTimeAnalyticsWebStack } from './real-time-analytics-web-stack';


export class CdkParentStack extends Stack {
  constructor(scope: Construct, id: string, props?: StackProps) {
    super(scope, id, props);


    new RealTimeAnalyticsPocStack(this, 'RealTimeAnalyticsPocStack');
    new RealTimeAnalyticsWebStack(this, 'RealTimeAnalyticsWebStack');
  }
}
```

用於測試的程式碼

```
session={{date.now('YYYYMMDD')}}|sequence={{date.now('x')}}|
reception={{date.now('x')}}|instrument={{random.number(9)}}|
l={{random.number(20)}}|price_0={{random.number({"min":10000,
 "max":30000})}}|price_1={{random.number({"min":10000, "max":30000})}}|
price_2={{random.number({"min":10000, "max":30000})}}|
price_3={{random.number({"min":10000, "max":30000})}}|
price_4={{random.number({"min":10000, "max":30000})}}|
price_5={{random.number({"min":10000, "max":30000})}}|
price_6={{random.number({"min":10000, "max":30000})}}|
price_7={{random.number({"min":10000, "max":30000})}}|
price_8={{random.number({"min":10000, "max":30000})}}|
```

測試 API Gateway

在 API Gateway 主控台上，使用 GET方法測試 API Gateway。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS SAM 自動化巢狀應用程式的部署

Rahul Sharad Gaikwad 醫生、Ishwar Chauthaiwale、Dmitry Gulin 和 Tabby Ward，Amazon Web Services

總結

在 Amazon Web Services (AWS) 上，AWS Serverless Application Model (AWS SAM) 是一種開放原始碼架構，提供速記語法來表達函數、APIs、資料庫和事件來源映射。只要為每個資源幾行，您就可以定義所需的應用程式，並使用 YAML 建立模型。在部署期間，SAM 會將 SAM 語法轉換並擴展為 AWS CloudFormation 語法，您可以用來更快速地建置無伺服器應用程式。

AWS SAM 可簡化 AWS 平台上無伺服器應用程式的開發、部署和管理。它提供標準化架構、更快速的部署、本機測試功能、資源管理、與 開發工具的無縫整合，以及支援社群。這些功能使其成為有效建置無伺服器應用程式的寶貴工具。

此模式使用 AWS SAM 範本來自動化巢狀應用程式的部署。巢狀應用程式是另一個應用程式中的應用程式。父應用程式會呼叫其子應用程式。這些是無伺服器架構鬆散耦合的元件。

使用巢狀應用程式，您可以重複使用獨立撰寫和維護但使用 AWS SAM 和 Serverless Application Repository 組成的服務或元件，快速建置高度複雜的無伺服器架構。巢狀應用程式可協助您建置功能更強大的應用程式，避免重複的工作，並確保團隊和組織的一致性和最佳實務。為了示範巢狀應用程式， 模式會部署範例 AWS 無伺服器購物車應用程式。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 現有的虛擬私有雲端 (VPC) 和子網路
- 整合的開發環境，例如 Visual Studio Code （ 如需詳細資訊，請參閱在 AWS 上建置的工具)
- 如果尚未安裝 Python wheel 程式庫，請使用 pip 安裝 wheel 安裝

限制

- 可在無伺服器應用程式中巢狀化的應用程式數量上限為 200。
- 巢狀應用程式的參數數目上限可以有 60 個。

產品版本

- 此解決方案建置在 AWS SAM 命令列界面 (AWS SAM CLI) 1.21.1 版上，但此架構應與更新的 AWS SAM CLI 版本搭配使用。

Architecture

目標技術堆疊

- Amazon API Gateway
- AWS SAM
- Amazon Cognito
- Amazon DynamoDB
- AWS Lambda
- Amazon Simple Queue Service (Amazon SQS) 佇列

目標架構

下圖顯示使用者如何透過呼叫 APIs 向購物服務提出請求。使用者的請求，包括所有必要的資訊，會傳送至 Amazon API Gateway 和 Amazon Cognito 授權方，該授權方會執行 APIs 的身分驗證和授權機制。

在 DynamoDB 中新增、刪除或更新項目時，會將事件放入 DynamoDB Streams，進而啟動 Lambda 函數。為了避免在同步工作流程中立即刪除舊項目，訊息會放入 SQS 佇列，這會啟動工作者函數來刪除訊息。

在此解決方案設定中，AWS SAM CLI 做為 AWS CloudFormation 堆疊的界面。AWS SAM 範本會自動部署巢狀應用程式。父 SAM 範本會呼叫子範本，而父 CloudFormation 堆疊會部署子堆疊。每個子堆疊都會建置 AWS SAM CloudFormation 範本中定義的 AWS 資源。

1. 建置和部署堆疊。
2. Auth CloudFormation 堆疊包含 Amazon Cognito。
3. 產品 CloudFormation 堆疊包含 Lambda 函數和 Amazon API Gateway
4. 購物 CloudFormation 堆疊包含 Lambda 函數、Amazon API Gateway、SQS 佇列和 Amazon DynamoDB 資料庫。

工具

工具

- Amazon API Gateway 可協助您建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。
- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在 AWS 帳戶和區域的整個生命週期中管理這些資源。
- Amazon Cognito 為 Web 和行動應用程式提供身分驗證、授權和使用者管理。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Serverless Application Model (AWS SAM) 是一種開放原始碼架構，可協助您在 AWS 雲端中建置無伺服器應用程式。
- Amazon Simple Queue Service (Amazon SQS) 提供安全、耐用且可用的託管佇列，可協助您整合和分離分散式軟體系統和元件。

Code

此模式的程式碼可在 GitHub AWS SAM 巢狀堆疊範例儲存庫中使用。

史詩

安裝 AWS SAM CLI

任務	Description	所需的技能
安裝 AWS SAM CLI。	若要安裝 AWS SAM CLI，請參閱 AWS SAM 文件中的說明。	DevOps 工程師
設定 AWS 登入資料。	若要設定 AWS 登入資料，讓 AWS SAM CLI 可以代表您呼叫 AWS 服務，請執行 `aws configure` 命令並依照提示操作。	DevOps 工程師

任務	Description	所需的技能

	`$aws configure`
	`AWS Access Key ID`
	` [None]: <your_acc`
	`ess_key_id>`
	`AWS Secret Access Key`
	` [None]: your_secr`
	`et_access_key`
	`Default region name`
	` [None]:`
	`Default output format`
	` [None]:`
	
	如需設定登入資料的詳細資訊，請參閱身分驗證和存取登入資料。	

初始化 AWS SAM 專案

任務	Description	所需的技能

複製 AWS SAM 程式碼儲存庫。	1. 輸入下列命令，複製此模式的 aws sam 巢狀堆疊範例儲存庫。
	
	`git clone https://g`
	`ithub.com/aws-samp`
	`les/aws-sam-nested-`
	`stack-sample.git`
	
	2. 輸入下列命令以導覽至複製的目錄。
	
	`cd aws-sam-nested-sta`
	`ck-sample`
		DevOps 工程師

部署 範本以初始化專案。	若要初始化專案，請執行 SAM `init`命令。當系統提示您選擇	DevOps 工程師

任務	Description	所需的技能
	範本來源時，請選擇 Custom Template Location。	

編譯和建置 SAM 範本程式碼

任務	Description	所需的技能
檢閱 AWS SAM 應用程式範本。	檢閱巢狀應用程式的範本。此範例使用以下巢狀應用程式範本： • auth.yaml – 此範本會設定身分驗證相關資源，例如 Amazon Cognito 和 AWS Systems Manager 參數存放區。 • product-mock.yaml – 此範本會部署與產品相關的資源，例如 Lambda 函數和 Amazon API Gateway。 • shoppingcart-service.yaml – 此範本會設定購物車相關的資源，例如 AWS Identity and Access Management (IAM)、DynamoDB 資料表和 Lambda 函數。	DevOps 工程師
檢閱父範本。	檢閱將調用巢狀應用程式範本的範本。在此範例中，父範本為 template.yml 。所有個別的應用程式都會巢狀在單一父範本 中template.yml 。	DevOps 工程師

任務	Description	所需的技能
編譯並建置 AWS SAM 範本程式碼。	使用 AWS SAM CLI，執行下列命令。 `sam build`	DevOps 工程師

部署 AWS SAM 範本

任務	Description	所需的技能
部署應用程式。	若要啟動建立巢狀應用程式 CloudFormation 堆疊並在 AWS 環境中部署程式碼的 SAM 範本程式碼，請執行下列命令。 `sam deploy --guided --stack-name shopping-cart-nested-stack --capabilities CAPABILITY_IAM CAPABILITY_AUTO_EXPAND` 命令將提示幾個問題。使用 回答所有問題y。	DevOps 工程師

驗證部署

任務	Description	所需的技能
驗證堆疊。	若要檢閱 AWS CloudFormation 範本中定義的 AWS CloudFormation 堆疊和 AWS 資源，請執行下列動作：	DevOps 工程師

任務	Description	所需的技能
	1. 登入 AWS 管理主控台，然後導覽至 CloudFormation 主控台。 2. 確認已列出父堆疊和子堆疊。 在此範例中，`sam-shopping-cart` 是呼叫巢狀身分驗證、產品和購物堆疊的父堆疊。 產品堆疊提供產品 API Gateway URL 連結做為輸出。	

相關資源

參考

- AWS Serverless 應用程式模型 (AWS SAM)
- GitHub 上的 AWS SAM
- 無伺服器購物車微服務 (AWS 範例應用程式）

教學課程和影片

- 建置無伺服器應用程式
- AWS Online Tech Talks：使用 AWS SAM 進行無伺服器應用程式建置和部署

其他資訊

所有程式碼都就緒後，範例會有下列目錄結構：

- sam_stacks – 此資料夾包含 `shared.py` layer。layer 是檔案封存，其中包含程式庫、自訂執行時間或其他相依性。透過 layer，您可以在函數中使用程式庫，而無需將其包含在部署套件中。
- product-mock-service – 此資料夾包含所有產品相關的 Lambda 函數和檔案。

- shopping-cart-service – 此資料夾包含所有與購物相關的 Lambda 函數和檔案。

使用 AWS Lambda 字符販賣機實作 Amazon S3 的 SaaS 租用戶隔離

Tabby Ward、Thomas Davis 和 Sravan Periyathambi，Amazon Web Services

總結

多租戶 SaaS 應用程式必須實作系統，以確保維持租戶隔離。當您將租戶資料存放在相同的 AWS 資源時，例如當多個租戶將資料存放在相同的 Amazon Simple Storage Service (Amazon S3) 儲存貯體時，您必須確保跨租戶存取不會發生。字符販賣機 (TVMs) 是提供租戶資料隔離的一種方式。這些機器提供一種機制來取得權杖，同時抽象化這些權杖產生方式的複雜性。開發人員可以使用 TVM，而無需詳細了解其如何產生字符。

此模式使用 實作 TVM AWS Lambda。TVM 會產生權杖，其中包含臨時安全權杖服務 (STS) 憑證，以限制對 S3 儲存貯體中單一 SaaS 租用戶資料的存取。

TVMs和此模式提供的程式碼通常用於衍生自 JSON Web Token (JWTs) 的宣告，以將 AWS 資源請求與租用戶範圍 AWS Identity and Access Management (IAM) 政策建立關聯。您可以使用此模式中的程式碼做為實作 SaaS 應用程式的基礎，該應用程式會根據 JWT 權杖中提供的宣告產生範圍廣泛的暫時 STS 憑證。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- AWS Command Line Interface (AWS CLI) 1.19.0 版或更新版本，在 macOS、Linux 或 Windows 上安裝和設定。或者，您可以使用 AWS CLI 2.1 版或更新版本。

限制

- 此程式碼在 Java 中執行，目前不支援其他程式設計語言。
- 範例應用程式不包含 AWS 跨區域或災難復原 (DR) 支援。
- 此模式示範適用於 SaaS 應用程式的 Lambda TVM 如何提供範圍租用戶存取。此模式不適用於生產環境，如果在特定應用程式或使用案例中沒有額外的安全測試。

Architecture

目標技術堆疊

- AWS Lambda
- Amazon S3
- IAM
- AWS Security Token Service (AWS STS)

目標架構

工具

AWS 服務

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- AWS Security Token Service (AWS STS) 可協助您為使用者請求暫時、有限權限的登入資料。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

Code

此模式的原始碼可做為附件使用，並包含下列檔案：

- `s3UploadSample.jar` 提供 Lambda 函數的原始碼，可將 JSON 文件上傳至 S3 儲存貯體。
- `tvm-layer.zip` 提供可重複使用的 Java 程式庫，可為 Lambda 函數提供權杖 (STS 臨時登入資料），以存取 S3 儲存貯體並上傳 JSON 文件。
- `token-vending-machine-sample-app.zip` 提供用來建立這些成品和編譯指示的原始碼。

若要使用這些檔案，請遵循下一節中的指示。

史詩

判斷變數值

任務	Description	所需的技能
判斷變數值。	此模式的實作包含數個必須一致使用的變數名稱。決定應該用於每個變數的值，並在後續步驟中請求時提供該值。 • `<AWS Account ID>` - AWS 帳戶 與您實作此模式的 相關聯的 12 位數帳戶 ID。如需有關如何尋找 AWS 帳戶 ID 的資訊，請參閱 IAM 文件中的為您的 AWS 帳戶 ID 使用別名。 • `<AWS Region>` – 您要在其中實作此模式 AWS 區域的 。如需詳細資訊 AWS 區域，請參閱 AWS 網站上的區域和可用區域。 • `<sample-tenant-name>` – 要在應用程式中使用的租戶名稱。為了簡單起見，建議您在此值中僅使用英數字元，但您可以將任何有效名稱用於 S3 物件金鑰。 • `<sample-tvm-role-name>` - 連接至執行 TVM 和範例應用程式的 Lambda 函數的 IAM 角色名稱。角色名稱是由大寫和小寫英數字元組成的字串，不含空格。您也可以包含下列任何字元：	雲端管理員

任務	Description	所需的技能

	底線 (_)、加號 (+)、等號 (=)、逗號 (,)、句號 (.)、符號 (@) 和連字號 (-)。角色名稱在帳戶內必須是唯一的。
	• `<sample-app-role-name>` - 當 Lambda 函數產生有範圍的暫時 STS 登入資料時，所擔任的 IAM 角色名稱。角色名稱是由大寫和小寫英數字元組成的字串，不含空格。您也可以包含下列任何字元：底線 (_)、加號 (+)、等號 (=)、逗號 (,)、句號 (.)、符號 (@) 和連字號 (-)。角色名稱在帳戶內必須是唯一的。
	• `<sample-app-function-name>` －Lambda 函數的名稱。這是長度最多為 64 個字元的字串。
	• `<sample-app-bucket-name>` － 必須使用範圍限定於特定租用戶的許可存取的 S3 儲存貯體名稱。S3 儲存貯體名稱：
	  • 長度必須介於 3 與 63 個字元之間。
	  • 只能包含小寫字母、數字、句點 (.) 和連字號 (-)。
	  • 必須以字母或數字開頭和結尾。	

任務	Description	所需的技能
	• 不得採用 IP 地址格式 (例如，192.168.5.4)。 • 在分割區中必須是唯一的。分割區是 Regions 的群組。 AWS 目前有三個分割區：aws適用於標準區域、aws-cn適用於中國區域，以及aws-us-gov 適用於 AWS GovCloud (US) Regions。	

建立 S3 儲存貯體

任務	Description	所需的技能
為範例應用程式建立 S3 儲存貯體。	使用下列 AWS CLI 命令來建立 S3 儲存貯體。在程式碼片段中提供 <sample-app-bucket-name> 值： `aws s3api create-bucket --bucket <sample-app-bucket-name>` Lambda 範例應用程式會將 JSON 檔案上傳至此儲存貯體。	雲端管理員

建立 IAM TVM 角色和政策

任務	Description	所需的技能
建立 TVM 角色。	使用下列其中一個 AWS CLI 命令來建立 IAM 角色。在 命令中	雲端管理員

任務	Description	所需的技能
	提供 <sample-tvm-role-name> 值。 對於 macOSor Linux shell： `aws iam create-role \ --role-name <sample-tvm-role-name> \ --assume-role-policy-document '{ "Version": "2012-10-17", "Statement": [{ "Effect": "Allow", "Action": ["sts:AssumeRole"], "Principal": { "Service": ["lambda.amazonaws.com"] }, "Condition": { "StringEquals": { "aws:SourceAccount": "<AWS Account ID>" } } }] }'`	

任務	Description	所需的技能
	對於 Windows 命令列： `aws iam create-role ^ --role-name <sample-tvm-role-name> ^ --assume-role-policy-document "{\"Version\": \"2012-10-17\", \"Statement\": [{\"Effect\": \"Allow\", \"Action\": [\"sts:AssumeRole\"], \"Principal\": {\"Service\": [\"lambda.amazonaws.com\"]}, \"Condition\": {\"StringEquals\": {\"aws:SourceAccount\": \"<AWS Account ID>\"}}}]}"` Lambda 範例應用程式會在叫用應用程式時擔任此角色。使用範圍政策擔任應用程式角色的功能為程式碼提供更廣泛存取 S3 儲存貯體的許可。	

任務	Description	所需的技能

建立內嵌 TVM 角色政策。	使用下列其中一個 AWS CLI 命令來建立 IAM 政策。在命令中提供 <sample-tvm-role-name> <AWS Account ID>、和 <sample-app-role-name> 值。
	
	對於 macOS 或 Linux shell：
	
	`aws iam put-role-policy \ --role-name <sample-tvm-role-name> \ --policy-name assume-app-role \ --policy-document '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "sts:AssumeRole", "Resource": "arn:aws:iam::<AWS Account ID>:role/<sample-app-role-name>" } ]}'`
	
	對於 Windows 命令列：
	
	`aws iam put-role-policy ^ --role-name <sample-tvm-role-name> ^`
		雲端管理員

任務	Description	所需的技能

	`--policy-name assume-app-role ^`
	`--policy-document "{\"Version\": \"2012-10-17\", \"Statement\": [{\"Effect\": \"Allow\", \"Action\": \"sts:AssumeRole\", \"Resource\": \"arn:aws:iam::<AWS Account ID>:role/<sample-app-role-name>\"}]}"`
	
	此政策會連接到 TVM 角色。它為程式碼提供擔任應用程式角色的能力，其具有更廣泛存取 S3 儲存貯體的許可。	

任務	Description	所需的技能

連接 受管 Lambda 政策。	使用下列 AWS CLI 命令來連接AWSLambdaBasicExecutionRole IAM 政策。在命令中提供 <sample-tvm-role-name> 值：
	`aws iam attach-role-policy \ --role-name <sample-tvm-role-name> \ --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole`
	對於 Windows 命令列：
	`aws iam attach-role-policy ^ --role-name <sample-tvm-role-name> ^ --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole`
	此受管政策會連接至 TVM 角色，以允許 Lambda 將日誌傳送至 Amazon CloudWatch。
		雲端管理員

建立 IAM 應用程式角色和政策

任務	Description	所需的技能
建立應用程式角色。	使用下列其中一個 AWS CLI 命令來建立 IAM 角色。在 命令中	雲端管理員

任務	Description	所需的技能

	提供 <AWS Account ID>、<sample-app-role-name> 和 <sample-tvm-role-name> 值。
	
	對於 macOS 或 Linux shell：
	
	`aws iam create-role \ --role-name <sample-app-role-name> \ --assume-role-policy-document '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "AWS": "arn:aws:iam::<AWS Account ID>:role/<sample-tvm-role-name>" }, "Action": "sts:AssumeRole" } ]}'`
	
	對於 Windows 命令列：
	
	`aws iam create-role ^ --role-name <sample-app-role-name> ^ --assume-role-policy-document "{\"Version\": \"2012-10-17\", \"Statement\":`	

任務	Description	所需的技能
	[{\"Effect\": \"Allow\",\"Principal\": {\"AWS\": \"arn:aws:iam::<AWS Account ID>:role/<sample-tvm-role-name>\"},\"Action\": \"sts:AssumeRole\"}]}" Lambda 範例應用程式會使用範圍政策擔任此角色，以取得 S3 儲存貯體的租戶型存取權。	

任務	Description	所需的技能
建立內嵌應用程式角色政策。	使用下列其中一個 AWS CLI mmands 來建立 IAM 政策。在命令中提供 <sample-app-role-name> 和 <sample-app-bucket-name> 值。 對於 macOS 或 Linux shell： `aws iam put-role-policy \ --role-name <sample-app-role-name> \ --policy-name s3-bucket-access \ --policy-document '{ "Version": "2012-10-17", "Statement": [{ "Effect": "Allow", "Action": ["s3:PutObject", "s3:GetObject", "s3:DeleteObject"], "Resource": "arn:aws:s3:::<sample-app-bucket-name>/*" }, { "Effect": "Allow", "Action": ["s3:ListBucket"],`	雲端管理員

任務	Description	所需的技能

	<pre>"Resource
	": "arn:aws:s3:::<sam
	ple-app-bucket-name>"
	}
	]}'</pre>
	對於 Windows 命令列：
	<pre>aws iam put-role-policy
	^
	--role-name <sample-a
	pp-role-name> ^
	--policy-name s3-bucket
	-access ^
	--policy-documen
	t "{\"Version\":
	\"2012-10-17\",
	\"Statement\":
	[{\"Effect\": \"Allow
	\", \"Action\":
	[\"s3:PutObject\",
	\"s3:GetObject\",
	\"s3:DeleteObject\
	"], \"Resource\":
	\"arn:aws:s3:::<sa
	mple-app-bucket-na
	me>/*\"}, {\"Effect\":
	\"Allow\", \"Action\
	": [\"s3:ListBucket
	\"], \"Resource\":
	\"arn:aws:s3:::<sa
	mple-app-bucket-name>
	\"}]}"</pre>
	此政策會連接至應用程式角色。它提供對 S3 儲存貯體中物件的廣泛存取。當範例應用程式擔任角色時，這些許可的	

任務	Description	所需的技能
	範圍會限定為具有 TVM 動態產生政策的特定租用戶。	

使用 TVM 建立 Lambda 範例應用程式

任務	Description	所需的技能
下載編譯的來源檔案。	下載 `s3UploadSample.jar` 和 `tvm-layer.zip` 檔案，其中包含為附件。中提供了用於建立這些成品和編譯函數的原始程式碼`token-vending-machine-sample-app.zip` 。	雲端管理員

建立 Lambda 層。	使用下列 AWS CLI 命令來建立 Lambda 層，讓 Lambda 可存取 TVM。
	Note
	如果您不是從下載的位置執行此命令 `tvm-layer.zip` ，請在 `--zip-file` 參數`tvm-layer.zip` 中提供正確的路徑。
	`aws lambda publish-layer-version \`
	`--layer-name sample-token-vending-machine \`
		雲端管理員、應用程式開發人員

任務	Description	所需的技能

	`--compatible-runtimes java11 \`
	`--zip-file fileb://tvm-layer.zip`
	
	對於 Windows 命令列：
	
	`aws lambda publish-layer-version ^`
	`--layer-name sample-token-vending-machine ^`
	`--compatible-runtimes java11 ^`
	`--zip-file fileb://tvm-layer.zip`
	
	此命令會建立包含可重複使用 TVM 程式庫的 Lambda 層。	

任務	Description	所需的技能
建立 Lambda 函數。	使用下列 AWS CLI 命令來建立 Lambda 函數。在命令中提供 <sample-app-function-name> 、<AWS Account ID>、<sample-app-bucket-name> 、、 <AWS Region> <sample-tvm-role-name> 和 <sample-app-role-name> 值。 **Note** 如果您不是從下載 的位置執行此命令s3UploadSample.jar ，請在 --zip-file 參數s3UploadSample.jar 中提供正確的路徑。 `aws lambda create-function \ --function-name <sample-app-function-name> \ --timeout 30 \ --memory-size 256 \ --runtime java11 \ --role arn:aws:iam::<AWS Account ID>:role/<sample-tvm-role-name> \ --handler com.amazon.aws.s3UploadSample.App \`	雲端管理員、應用程式開發人員

任務	Description	所需的技能
	--zip-file fileb://s3UploadSample.jar \ --layers arn:aws:lambda:<AWS Region>:<AWS Account ID>:layer:sample-token-vending-machine:1 \ --environment "Variables={S3_BUCKET=<sample-app-bucket-name>, ROLE=arn:aws:iam::<AWS Account ID>:role/<sample-app-role-name>}" 對於 Windows 命令列： aws lambda create-function ^ --function-name <sample-app-function-name> ^ --timeout 30 ^ --memory-size 256 ^ --runtime java11 ^ --role arn:aws:iam::<AWS Account ID>:role/<sample-tvm-role-name> ^ --handler com.amazon.aws.s3UploadSample.App ^ --zip-file fileb://s3UploadSample.jar ^ --layers arn:aws:lambda:<AWS Region>:<AWS Account ID>:layer:sample-token-vending-machine:1 ^ --environment "Variables={S3_BUCKET=<samp	

任務	Description	所需的技能
	le-app-bucket-name>,ROLE=arn:aws:iam::<AWS Account ID>:role/<sample-app-role-name>}" 此命令會建立連接範例應用程式程式碼和 TVM layer 的 Lambda 函數。它也會設定兩個環境變數： S3_BUCKET 和 ROLE。範例應用程式使用這些變數來決定要擔任的角色，以及要上傳 JSON 文件的 S3 儲存貯體。	

測試範例應用程式和 TVM

任務	Description	所需的技能
叫用 Lambda 範例應用程式。	使用下列其中一個 AWS CLI 命令，以其預期的承載啟動 Lambda 範例應用程式。在 命令中提供 <sample-app-function-name> 和 <sample-tenant-name> 值。 對於 macOS 和 Linux shell： aws lambda invoke \ --function <sample-app-function-name> \ --invocation-type RequestResponse \ --payload '{"tenant": "<sample-tenant-name>"}' \	雲端管理員、應用程式開發人員

任務	Description	所需的技能

```
--cli-binary-format
 raw-in-base64-out
 response.json
```

對於 Windows 命令列：

```
aws lambda invoke ^
--function <sample-a
pp-function-name> ^
--invocation-type
 RequestResponse ^
--payload "{\"tenant
\": \"<sample-tenant-n
ame>\"}" ^
--cli-binary-format
 raw-in-base64-out
 response.json
```

此命令會呼叫 Lambda 函數，並在`response.json` 文件中傳回結果。在許多以 Unix 為基礎的系統上，您可以將 `response.json` 變更為 `/dev/stdout` ，直接將結果輸出到您的 shell，而無需建立另一個檔案。

> **Note**
>
> 變更此 Lambda 函數後續調用中的`<sample-tenant-name>` 值會變更 JSON 文件的位置，以及字符提供的許可。

任務	Description	所需的技能
檢視 S3 儲存貯體以查看建立的物件。	瀏覽至您先前建立的 S3 儲存貯體 (<sample-app-bucket-name>)。此儲存貯體包含值為 的 S3 物件字首<sample-tenant-name> 。在此字首下，您會找到名為 且具有 UUID 的 JSON 文件。多次叫用範例應用程式會新增更多 JSON 文件。	雲端管理員

任務	Description	所需的技能
在 CloudWatch Logs 中檢視範例應用程式的日誌。	檢視與 CloudWatch Logs `<sample-app-function-name>` 中名為 的 Lambda 函數相關聯的日誌。如需說明，請參閱 Lambda 文件中的將 Lambda 函數日誌傳送至 CloudWatch Logs。您可以在這些日誌中檢視 TVM 產生的租戶範圍政策。此租用戶範圍政策將範例應用程式的許可授予 Amazon S3 PutObject、GetObject、DeleteObject 和 ListBucket APIs，但僅適用於與 相關聯的物件字首`<sample-tenant-name>` 。在範例應用程式的後續調用中，如果您變更 `<sample-tenant-name>` ，TVM 會更新範圍政策，以對應調用承載中提供的租用戶。此動態產生的政策顯示如何在 SaaS 應用程式中使用 TVM 維護租戶範圍存取。 TVM 功能是在 Lambda 層中提供，因此可以連接到應用程式使用的其他 Lambda 函數，而不必複寫程式碼。 如需動態產生政策的圖例，請參閱其他資訊一節。	雲端管理員

相關資源

- 使用動態產生的 IAM 政策隔離租用戶 （部落格文章）

- 在 SaaS 環境中套用動態產生的隔離政策（部落格文章）
- 上的 SaaS AWS

其他資訊

以下日誌顯示此模式中 TVM 程式碼產生的動態產生政策。在此螢幕擷取畫面中，<sample-app-bucket-name>是DOC-EXAMPLE-BUCKET ，而 <sample-tenant-name>是 test-tenant-1。此範圍政策傳回的 STS 登入資料無法在 S3 儲存貯體中的物件上執行任何動作，但與物件金鑰字首 相關聯的物件除外test-tenant-1。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS Step Functions 實作無伺服器 saga 模式

Tabby Ward、Joe Kern 和 Rohan Mehta，Amazon Web Services

摘要

在微服務架構中，主要目標是建置解耦的獨立元件，以提升應用程式的敏捷性、彈性和更快的上市時間。由於解耦，每個微服務元件都有自己的資料持久性層。在分散式架構中，商業交易可以跨越多個微服務。由於這些微服務無法使用單一原子性、一致性、隔離性、耐久性 (ACID) 交易，因此您可能會最終產生部分交易。在此情況下，需要一些控制邏輯才能復原已處理的交易。分散式 saga 模式通常用於此目的。

saga 模式是一種故障管理模式，可協助在分散式應用程式中建立一致性，並協調多個微服務之間的交易，以維持資料一致性。當您使用 saga 模式時，每個執行交易的服務都會發佈事件，觸發後續服務在鏈結中執行下一個交易。這會持續到鏈結中的最後一個交易完成為止。如果商業交易失敗，saga 會協調一系列補償交易，復原先前交易所做的變更。

此模式示範如何使用 AWS Step Functions、AWS Lambda 和 Amazon DynamoDB 等無伺服器技術，自動設定和部署範例應用程式（處理行程保留）。範例應用程式也會使用 Amazon API Gateway 和 Amazon Simple Notification Service (Amazon SNS) 來實作 saga 執行協調器。模式可以使用基礎設施即程式碼 (IaC) 架構進行部署，例如 AWS Cloud Development Kit (AWS CDK)、AWS Serverless Application Model (AWS SAM) 或 Terraform。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 建立 AWS CloudFormation 堆疊的許可。如需詳細資訊，請參閱 CloudFormation 文件中的控制存取。
- 使用 AWS 帳戶設定您選擇的 IaC 架構 (AWS CDK、AWS SAM 或 Terraform)，以便您可以使用架構 CLI 部署應用程式。
- NodeJS，用於建置應用程式並在本機執行。
- 您選擇的程式碼編輯器（例如 Visual Studio Code、Sublime 或 Atom)。

產品版本

- NodeJS 第 14 版

- AWS CDK 2.37.1 版
- AWS SAM 1.71.0 版
- Terraform 1.3.7 版

限制

事件來源是在微服務架構中實作 saga 協同運作模式的一種自然方式，其中所有元件都鬆散耦合，彼此沒有直接知識。如果您的交易涉及少量步驟（三到五個），則 Saga 模式可能非常適合。不過，複雜度會隨著微服務數量和步驟數量而增加。

當您使用此設計時，測試和偵錯可能會變得困難，因為您必須執行所有服務才能模擬交易模式。

Architecture

目標架構

提議的架構使用 AWS Step Functions 建置類似模式來預訂航班、預訂租車，以及處理假期的付款。

下列工作流程圖說明行程保留系統的典型流程。工作流程包含預留航空旅程 ("ReserveFlight")、預留車輛 ("ReserveCarRental")、處理付款 ("ProcessPayment")、確認航班保留 ("ConfirmFlight")，以及確認租車 ("ConfirmCarRental")，隨後在這些步驟完成時會收到成功通知。不過，如果系統在執行任何這些交易時遇到任何錯誤，就會開始向後失敗。例如，付款處理 ("ProcessPayment") 的錯誤會觸發退款 ("RefundPayment")，然後觸發租車和航班 ("CancelRentalReservation" 和 "CancelFlightReservation") 的取消，以失敗訊息結束整個交易。

此模式會針對圖表中反白顯示的每個任務部署個別的 Lambda 函數，以及三個用於航班、租車和付款的 DynamoDB 資料表。每個 Lambda 函數都會建立、更新或刪除個別 DynamoDB 資料表中的資料列，視交易是否確認或復原而定。模式使用 Amazon SNS 傳送文字 (SMS) 訊息給訂閱者，通知他們交易失敗或成功。

自動化和擴展

您可以使用其中一個 IaC 架構來建立此架構的組態。將下列其中一個連結用於您偏好的 IaC。

- 使用 AWS CDK 部署
- 使用 AWS SAM 部署
- 使用 Terraform 部署

工具

AWS 服務

- AWS Step Functions 是一種無伺服器協同運作服務，可讓您結合 AWS Lambda 函數和其他 AWS 服務來建置業務關鍵應用程式。透過 Step Functions 圖形主控台，您會將應用程式的工作流程視為一系列的事件驅動步驟。
- Amazon DynamoDB 是全受管的 NoSQL 資料庫服務，可提供快速且可預測的效能和無縫的可擴展性。您可以使用 DynamoDB 建立資料庫資料表，藉此存放和擷取任意數量的資料，並為任何層級的請求流量提供服務。
- AWS Lambda 是一種運算服務，可讓您執行程式碼，而無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。
- Amazon API Gateway 是一種 AWS 服務，可用於建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。
- Amazon Simple Notification Service (Amazon SNS) 是一種受管服務，可將訊息從發佈者交付給訂閱者。
- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可透過使用 TypeScript、JavaScript、Python、Java 和 C#/ 等熟悉的程式設計語言來定義您的雲端應用程式資源。淨額。
- AWS Serverless Application Model (AWS SAM) 是用於建置無伺服器應用程式的開放原始碼架構。它提供速記語法來表達函數、APIs、資料庫和事件來源映射。

Code

您可以在以下連結中找到示範 saga 模式的範例應用程式的程式碼，包括 IaC 範本 (AWS CDK、AWS SAM 或 Terraform)、Lambda 函數和 DynamoDB 資料表。請遵循第一個 epic 中的指示來安裝這些項目。

- 使用 AWS CDK 部署
- 使用 AWS SAM 部署
- 使用 Terraform 部署

史詩

安裝套件、編譯和建置

任務	Description	所需的技能
安裝 NPM 套件。	建立新的目錄、導覽至終端機中的該目錄，並從此模式稍早的程式碼區段複製您選擇的 GitHub 儲存庫。 在具有 `package.json` 檔案的根資料夾中，執行下列命令來下載並安裝所有 Node Package Manager (NPM) 套件： `npm install`	開發人員、雲端架構師
編譯指令碼。	在根資料夾中，執行下列命令，指示 TypeScript 轉換器建立所有必要的 JavaScript 檔案： `npm run build`	開發人員、雲端架構師
留意變更並重新編譯。	在根資料夾中，在不同的終端機視窗中執行下列命令，以 監看程式碼變更，並在偵測到變更時編譯程式碼： `npm run watch`	開發人員、雲端架構師
執行單位測試 （僅限 AWS CDK)。	如果您使用的是 AWS CDK，請在根資料夾中執行下列命令來執行 Jest 單位測試：	開發人員、雲端架構師

任務	Description	所需的技能
	`npm run test`	

將資源部署到目標 AWS 帳戶

任務	Description	所需的技能

將示範堆疊部署至 AWS。	**Important**
	應用程式與 AWS 區域無關。如果您使用設定檔，則必須在 AWS Command Line Interface (AWS CLI) 設定檔中或透過 AWS CLI 環境變數明確宣告區域。
	
	在根資料夾中，執行下列命令來建立部署組件，並將其部署到預設的 AWS 帳戶和區域。
	
	AWS CDK：
	`cdk bootstrap`
	`cdk deploy`
	
	AWS SAM：
	`sam build`
	`sam deploy --guided`
	
	Terraform：
		開發人員、雲端架構師

任務	Description	所需的技能

	`terraform init`
	`terraform apply`
	
	此步驟可能需要幾分鐘的時間才能完成。此命令使用為 AWS CLI 設定的預設登入資料。
	
	請注意，在部署完成後，主控台上顯示的 API Gateway URL。您需要此資訊來測試 saga 執行流程。	

比較已部署的堆疊與目前狀態。	在根資料夾中，執行下列命令，在變更原始碼之後，將部署的堆疊與目前狀態 進行比較：
	
	AWS CDK：
	
	`cdk diff`
	
	AWS SAM：
	
	`sam deploy`
	
	Terraform：
	
	`terraform plan`
		開發人員、雲端架構師

測試執行流程

任務	Description	所需的技能
測試 saga 執行流程。	當您部署堆疊時，導覽至您在先前步驟中記下的 API	開發人員、雲端架構師

任務	Description	所需的技能
	Gateway URL。此 URL 會觸發狀態機器啟動。如需如何透過傳遞不同的 URL 參數來操作狀態機器流程的詳細資訊，請參閱其他資訊一節。 若要 檢視結果，請登入 AWS 管理主控台，然後導覽至 Step Functions 主控台。在這裡，您可以看到 saga 狀態機器的每個步驟。您也可以檢視 DynamoDB 資料表，以查看插入、更新或刪除的記錄。如果您經常重新整理畫面，您可以觀看交易狀態從 變更為 `pending` `confirmed` 。 您可以訂閱 SNS 主題，方法是使用手機號碼更新 `stateMachine.ts` 檔案中的程式碼，以便在成功或失敗的保留時接收簡訊。如需詳細資訊，請參閱其他資訊區段中的 Amazon SNS。	

清除

任務	Description	所需的技能
清除資源。	若要清除為此應用程式部署的資源，您可以使用下列其中一個命令。 AWS CDK：	應用程式開發人員、雲端架構師

任務	Description	所需的技能

	`cdk destroy`
	AWS SAM：
	`sam delete`
	Terraform：
	`terraform destroy`	

相關資源

技術論文

- 在 AWS 上實作微服務
- 無伺服器應用程式鏡頭

AWS 服務文件

- AWS CDK 入門
- AWS SAM 入門
- AWS Step Functions
- Amazon DynamoDB
- AWS Lambda
- Amazon API Gateway
- Amazon SNS

教學課程

- 無伺服器運算實作研討會

其他資訊

Code

為了測試目的，此模式會部署 API Gateway 和測試 Lambda 函數，以觸發 Step Functions 狀態機器。使用 Step Functions，您可以透過傳遞`run_type`參數來模擬 "ReserveFlight"、"ReserveCarRental"、"ProcessPayment"、"ConfirmFlight" 和 "ConfirmCarRental."

`saga` Lambda 函數 (`sagaLambda.ts`) 會從 API Gateway URL 中的查詢參數取得輸入，建立下列 JSON 物件，並將其傳遞給 Step Functions 執行：

```
let input = {
"trip_id": tripID, //  value taken from query parameter, default is AWS request ID
"depart_city": "Detroit",
"depart_time": "2021-07-07T06:00:00.000Z",
"arrive_city": "Frankfurt",
"arrive_time": "2021-07-09T08:00:00.000Z",
"rental": "BMW",
"rental_from": "2021-07-09T00:00:00.000Z",
"rental_to": "2021-07-17T00:00:00.000Z",
"run_type": runType // value taken from query parameter, default is "success"
};
```

您可以傳遞下列 URL 參數來實驗 Step Functions 狀態機器的不同流程：

- 成功執行 – https：//{api 閘道 url}
- 預留航班失敗 - https：//{api 閘道 url}？runType=failFlightsReservation
- 確認航班失敗 - https：//{api 閘道 url}？runType=failFlightsConfirmation
- 預留租車失敗 - https：//{api 閘道 url}？runType=failCarRentalReservation
- 確認租車失敗 - https：//{api 閘道 url}？runType=failCarRentalConfirmation
- 處理付款失敗 - https：//{api 閘道 url}？runType=failPayment
- 傳遞行程 ID - https：//{api 閘道 url}？tripID={預設情況下，行程 ID 將是 AWS 請求 ID}

IaC 範本

連結的儲存庫包含 IaC 範本，可用來建立整個範例行程保留應用程式。

- 使用 AWS CDK 部署
- 使用 AWS SAM 部署

- 使用 Terraform 部署

DynamoDB 資料表

以下是航班、租車和付款資料表的資料模型。

```
Flight Data Model:
 var params = {
      TableName: process.env.TABLE_NAME,
      Item: {
        'pk' : {S: event.trip_id},
        'sk' : {S: flightReservationID},
        'trip_id' : {S: event.trip_id},
        'id': {S: flightReservationID},
        'depart_city' : {S: event.depart_city},
        'depart_time': {S: event.depart_time},
        'arrive_city': {S: event.arrive_city},
        'arrive_time': {S: event.arrive_time},
        'transaction_status': {S: 'pending'}
      }
    };

Car Rental Data Model:
var params = {
      TableName: process.env.TABLE_NAME,
      Item: {
        'pk' : {S: event.trip_id},
        'sk' : {S: carRentalReservationID},
        'trip_id' : {S: event.trip_id},
        'id': {S: carRentalReservationID},
        'rental': {S: event.rental},
        'rental_from': {S: event.rental_from},
        'rental_to': {S: event.rental_to},
        'transaction_status': {S: 'pending'}
      }
    };

Payment Data Model:
var params = {
      TableName: process.env.TABLE_NAME,
      Item: {
        'pk' : {S: event.trip_id},
        'sk' : {S: paymentID},
```

```
        'trip_id' : {S: event.trip_id},
        'id': {S: paymentID},
        'amount': {S: "750.00"}, // hard coded for simplicity as implementing any
 monetary transaction functionality is beyond the scope of this pattern
        'currency': {S: "USD"},
        'transaction_status': {S: "confirmed"}
      }
    };
```

Lambda 函數

將建立下列函數，以支援 Step Functions 中的狀態機器流程和執行：

- 預留航班：使用 `transaction_status`的 將記錄插入 DynamoDB 航班資料表`pending`，以預訂航班。
- 確認航班：更新 DynamoDB 航班資料表中的記錄，將 `transaction_status`設定為 `confirmed`，以確認航班。
- 取消航班保留：從 DynamoDB 航班資料表刪除記錄，以取消待定航班。
- 預留租車：使用 `transaction_status`的 將記錄插入 DynamoDB CarRentals 資料表`pending`，以預訂租車。
- 確認租車：更新 DynamoDB CarRentals 資料表中的記錄，將 `transaction_status`設定為 `confirmed`，以確認租車。
- 取消租車保留：從 DynamoDB CarRentals 資料表刪除記錄，以取消待定的租車。
- 處理付款：將記錄插入 DynamoDB 付款資料表以進行付款。
- 取消付款：從 DynamoDB 付款資料表中刪除付款的記錄。

Amazon SNS

範例應用程式會建立下列主題和訂閱來傳送簡訊，並通知客戶保留成功或失敗。如果您想要在測試範例應用程式時接收文字訊息，請在狀態機器定義檔案中使用有效的電話號碼更新簡訊訂閱。

AWS CDK 程式碼片段（在下列程式碼的第二行中新增電話號碼）：

```
const topic = new  sns.Topic(this, 'Topic');
topic.addSubscription(new subscriptions.SmsSubscription('+11111111111'));
const snsNotificationFailure = new tasks.SnsPublish(this ,'SendingSMSFailure', {
topic:topic,
integrationPattern: sfn.IntegrationPattern.REQUEST_RESPONSE,
```

```
message: sfn.TaskInput.fromText('Your Travel Reservation Failed'),
});

const snsNotificationSuccess = new tasks.SnsPublish(this ,'SendingSMSSuccess', {
topic:topic,
integrationPattern: sfn.IntegrationPattern.REQUEST_RESPONSE,
message: sfn.TaskInput.fromText('Your Travel Reservation is Successful'),
});
```

AWS SAM 程式碼片段（將+1111111111字串取代為您的有效電話號碼）：

```
  StateMachineTopic11111111111:
    Type: 'AWS::SNS::Subscription'
    Properties:
      Protocol: sms
      TopicArn:
        Ref: StateMachineTopic
      Endpoint: '+11111111111'
    Metadata:
      'aws:sam:path': SamServerlessSagaStack/StateMachine/Topic/+11111111111/Resource
```

Terraform 程式碼片段（將+111111111字串取代為您的有效電話號碼）：

```
resource "aws_sns_topic_subscription" "sms-target" {
  topic_arn = aws_sns_topic.topic.arn
  protocol  = "sms"
  endpoint  = "+11111111111"
}
```

成功的保留

以下流程說明「ReserveFlight」、「ReserveCarRental」和「ProcessPayment」後接「ConfirmFlight」和「ConfirmCarRental." 客戶會透過傳送給 SNS 主題訂閱者的簡訊收到成功預訂的通知。

失敗的保留

此流程是 saga 模式失敗的範例。如果預訂航班和租車後，「ProcessPayment」失敗，步驟會依相反順序取消。 會釋出保留，並透過傳送給 SNS 主題訂閱者的簡訊通知客戶失敗。

使用 AWS CDK 設定 Amazon ECS Anywhere 來管理內部部署容器應用程式

Rahul Sharad Gaikwad 醫生，Amazon Web Services

總結

Amazon ECS Anywhere 是 Amazon Elastic Container Service (Amazon ECS) 的延伸。您可以使用 ECS Anywhere 在內部部署或客戶受管環境中部署原生 Amazon ECS 任務。此功能有助於降低成本，並減少複雜的本機容器協同運作和操作。您可以使用 ECS Anywhere 在內部部署和雲端環境中部署和執行容器應用程式。它消除了您的團隊學習多個網域和技能集，或自行管理複雜軟體的需求。

此模式示範使用 AWS Cloud Development Kit (AWS CDK) 堆疊設定 ECS Anywhere 的步驟。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 安裝並設定 AWS Command Line Interface (AWS CLI)。（請參閱 AWS CLI 文件中的安裝、更新和解除安裝 AWS CLI。）
- AWS CDK Toolkit，已安裝並設定。（請參閱 AWS CDK 文件中的 AWS CDK Toolkit，並依照指示在全球安裝第 2 版。）
- 節點套件管理員 (npm)，已安裝並設定 TypeScript 中的 AWS CDK。（請參閱 npm 文件中的下載和安裝 Node.js 和 npm。）

限制

- 如需限制和考量，請參閱 Amazon ECS Anywhere)。

產品版本

- AWS CDK Toolkit 第 2 版
- npm 7.20.3 版或更新版本
- Node.js 16.6.1 版或更新版本

Architecture

目標技術堆疊

- AWS CloudFormation
- AWS CDK
- Amazon ECS Anywhere
- AWS Identity and Access Management (IAM)

目標架構

下圖說明使用 AWS CDK 搭配 TypeScript 的 ECS Anywhere 設定的高階系統架構，如此模式所實作。

1. 當您部署 AWS CDK 堆疊時，它會在 AWS 上建立 CloudFormation 堆疊。
2. CloudFormation 堆疊會佈建 Amazon ECS 叢集和相關的 AWS 資源。
3. 若要向 Amazon ECS 叢集註冊外部執行個體，您必須在虛擬機器 (VM) 上安裝 AWS Systems Manager Agent (SSM Agent)，並將 VM 註冊為 AWS Systems Manager 受管執行個體。
4. 您還必須在 VM 上安裝 Amazon ECS 容器代理程式和 Docker，以將其註冊為 Amazon ECS 叢集的外部執行個體。
5. 使用 Amazon ECS 叢集註冊和設定外部執行個體時，它可以在您的 VM 上執行多個容器，其已註冊為外部執行個體。

自動化和擴展

此模式隨附的 GitHub 儲存庫會使用 AWS CDK 做為基礎設施做為程式碼 (IaC) 工具，來建立此架構的組態。AWS CDK 可協助您協調資源並設定 ECS Anywhere。

工具

- AWS 雲端開發套件 (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。

Code

此模式的原始碼可在 GitHub 上的 Amazon ECS Anywhere CDK 範例儲存庫中取得。若要複製和使用儲存庫，請遵循下一節中的指示。

史詩

驗證 AWS CDK 組態

任務	Description	所需的技能
驗證 AWS CDK 版本。	執行下列命令來驗證 AWS CDK Toolkit 的版本： `cdk --version` 此模式需要 AWS CDK 第 2 版。如果您有舊版的 AWS CDK，請遵循 AWS CDK 文件中的指示進行更新。	DevOps 工程師

設定 AWS 登入資料。	若要設定登入資料，請執行 `aws configure` 命令並遵循提示：
	`$aws configure`
	`AWS Access Key ID [None]: <your-access-key-ID>`
	`AWS Secret Access Key [None]: <your-secret-access-key>`
	`Default region name [None]: <your-Region-name>`
	`Default output format [None]:`
		DevOps 工程師

引導 AWS CDK 環境

任務	Description	所需的技能
複製 AWS CDK 程式碼儲存庫。	使用 命令複製此模式的 GitHub 程式碼儲存庫： `git clone https://github.com/aws-samples/amazon-ecs-anywhere-cdk-samples.git`	DevOps 工程師
引導環境。	若要將 AWS CloudFormation 範本部署到您要使用的帳戶和 AWS 區域，請執行下列命令： `cdk bootstrap <account-number>/<Region>` 如需詳細資訊，請參閱 AWS CDK 文件中的啟動。	DevOps 工程師

建置和部署專案

任務	Description	所需的技能

安裝套件相依性並編譯 TypeScript 檔案。	安裝套件相依性，並執行下列命令編譯 TypeScript 檔案：
	`$cd amazon-ecs-anywhere-cdk-samples`
	`$npm install`
	`$npm fund`
	這些命令會從範例儲存庫安裝所有套件。
		DevOps 工程師

任務	Description	所需的技能

	Important
	如果您收到有關遺失套件的任何錯誤，請使用下列其中一個命令：
	`$npm ci`
	—或—
	`$npm install -g @aws-cdk/<package_name>`
	如需詳細資訊，請參閱 npm 文件中的 npm ci 和 npm 安裝。	

建置專案。	若要建置專案程式碼，請執行命令： `npm run build` 如需建置和部署專案的詳細資訊，請參閱 AWS CDK 文件中的您的第一個 AWS CDK 應用程式。	DevOps 工程師
部署專案。	若要部署專案程式碼，請執行命令： `cdk deploy`	DevOps 工程師

任務	Description	所需的技能
驗證堆疊建立和輸出。	開啟位於 https：//https://console.aws.amazon.com/cloudformation 的 AWS CloudFormation 主控台，然後選擇`EcsAnywhereStack` 堆疊。Outputs 索引標籤會顯示要在外部 VM 上執行的命令。	DevOps 工程師

設定內部部署機器

任務	Description	所需的技能
使用 Vagrant 設定您的 VM。	基於示範目的，您可以使用 HashiCorp Vagrant 來建立 VM。Vagrant 是一種開放原始碼公用程式，用於建置和維護可攜式虛擬軟體開發環境。從放置 Vagrantfile 的根目錄執行`vagrant up`命令來建立 Vagrant VM。如需詳細資訊，請參閱 Vagrant 文件。	DevOps 工程師
將您的 VM 註冊為外部執行個體。	1. 使用`vagrant ssh` 命令登入 Vagrant VM。如需詳細資訊，請參閱 Vagrant 文件。 2. 建立啟用代碼和 ID，供您用來向 AWS Systems Manager 註冊 VM，以及啟用外部執行個體。此命令的輸出包含`ActivationId` 和`ActivationCode` 值： `aws ssm create-activation --iam-role`	DevOps 工程師

任務	Description	所需的技能
	EcsAnywhereInstanceRole | tee ssm-activation.json 3. 匯出啟用 ID 和程式碼值： export ACTIVATION_ID=<activation-ID> export ACTIVATION_CODE=<activation-code> 4. 將安裝指令碼下載到您的現場部署伺服器或 VM： curl -o "ecs-anywhere-install.sh" "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-install-latest.sh" && sudo chmod +x ecs-anywhere-install.sh 5. 在內部部署伺服器或 VM 上執行安裝指令碼： sudo ./ecs-anywhere-install.sh \ --cluster test-ecs-anywhere \ --activation-id $ACTIVATION_ID \ --activation-code $ACTIVATION_CODE \ --region <Region>	

任務	Description	所需的技能
	如需設定和註冊 VM 的詳細資訊，請參閱 Amazon ECS 文件中的將外部執行個體註冊至叢集。	

驗證 ECS Anywhere 和外部 VM 的狀態。	若要驗證您的虛擬盒是否已連線至 Amazon ECS 控制平面並執行，請使用下列命令：
	`aws ssm describe-instance-information`
	`aws ecs list-container-instances --cluster $CLUSTER_NAME`
		DevOps 工程師

清除

任務	Description	所需的技能
清除和刪除資源。	完成此模式之後，您應該移除您建立的資源，以避免產生任何進一步的費用。若要清除，請執行 命令： `cdk destroy`	DevOps 工程師

相關資源

- Amazon ECS Anywhere 文件
- Amazon ECS Anywhere 示範
- Amazon ECS Anywhere 研討會範例

在 AWS 上現代化 ASP.NET Web Forms 應用程式

Vijai Anand Ramalingam 和 Sreelaxmi Pai，Amazon Web Services

總結

此模式說明透過將舊版 ASP.NET Web Forms 應用程式移植到 AWS 上的 ASP.NET Core 來現代化舊版 Web Forms 應用程式的步驟。

將 ASP.NET Web Forms 應用程式移植到 ASP.NET Core 可協助您利用 Linux 的效能、節省成本和強大的生態系統。不過，這可能會是大量的手動工作。在此模式中，舊版應用程式會使用分階段方法逐步現代化，然後在 AWS 雲端中容器化。

考慮購物車的舊版整體應用程式。假設它是建立為 ASP.NET Web Forms 應用程式，並由具有程式碼後面 (`aspx.cs`) 檔案的 .aspx 頁面組成。現代化程序包含下列步驟：

1. 使用適當的分解模式，將整體分解為微服務。如需詳細資訊，請參閱 AWS 方案指引網站上的將整體分解為微服務指南。
2. 將舊版 ASP.NET Web Forms (.NET Framework) 應用程式移植到 .NET 5 或更新版本中的 ASP.NET Core。在此模式中，您可以使用適用於 .NET 的移植助理來掃描 ASP.NET Web Forms 應用程式，並識別與 ASP.NET Core 的不相容。這可減少手動移植工作。
3. 使用 React 重新開發 Web Forms UI layer。此模式不包含 UI 重新開發。如需說明，請參閱 React 文件中的建立新的 React 應用程式。
4. 將 Web 表單程式碼落後檔案 （ 業務界面 ） 重新開發為 ASP.NET Core Web API。此模式使用 NDepend 報告來協助識別必要的檔案和相依性。
5. 使用適用於 .NET 的移植助理，將舊版應用程式中的共用/常見專案，例如商業邏輯和資料存取，升級至 .NET 5 或更新版本。
6. 新增 AWS 服務以補充您的應用程式。例如，您可以使用 Amazon CloudWatch Logs 來監控、存放和存取應用程式的日誌，以及使用 AWS Systems Manager 來存放應用程式設定。
7. 容器化現代化 ASP.NET Core 應用程式。此模式會建立以 Visual Studio 中的 Linux 為目標的 Docker 檔案，並使用 Docker 桌面在本機進行測試。此步驟假設您的舊版應用程式已在內部部署或 Amazon Elastic Compute Cloud (Amazon EC2) Windows 執行個體上執行。如需詳細資訊，請參閱模式 在 Amazon EC2 Linux 執行個體上執行 ASP.NET Core Web API Docker 容器。
8. 將現代化 ASP.NET 核心應用程式部署至 Amazon Elastic Container Service (Amazon ECS)。此模式不包含部署步驟。如需說明，請參閱 Amazon ECS 研討會。

> **Note**
>
> 此模式不包含 UI 開發、資料庫現代化或容器部署步驟。

先決條件和限制

先決條件

- Visual Studio 或 Visual Studio Code，已下載並安裝。
- 使用 AWS 管理主控台和 AWS Command Line Interface (AWS CLI) 第 2 版存取 AWS 帳戶。（請參閱設定 AWS CLI 的說明。）
- AWS Toolkit for Visual Studio（請參閱設定指示）。
- Docker Desktop，已下載並安裝。
- .NET SDK，已下載並安裝。
- NDepend 工具，下載並安裝。若要安裝 Visual Studio 的 NDepend 延伸模組，請執行 `NDepend.VisualStudioExtension.Installer`(請參閱說明)。根據您的需求，您可以選擇 Visual Studio 2019 或 2022。
- 適用於 .NET 的移植助理，已下載並安裝。

Architecture

現代化購物車應用程式

下圖說明舊版 ASP.NET 購物車應用程式的現代化程序。

目標架構

下圖說明 AWS 上現代化購物車應用程式的架構。ASP.NET Core Web APIs會部署到 Amazon ECS 叢集。記錄和組態服務由 Amazon CloudWatch Logs 和 AWS Systems Manager 提供。

工具

AWS 服務

- Amazon ECS – Amazon Elastic Container Service (Amazon ECS) 是一種高度可擴展的快速容器管理服務，用於執行、停止和管理叢集上的容器。您可以在 AWS Fargate 管理的無伺服器基礎設施上執行任務和服務。或者，若要進一步控制您的基礎設施，您可以在您管理的 EC2 執行個體叢集上執行任務和服務。
- Amazon CloudWatch Logs – Amazon CloudWatch Logs 會集中來自您使用之所有系統、應用程式和 AWS 服務的日誌。您可以檢視和監控日誌、搜尋特定錯誤代碼或模式、根據特定欄位篩選日誌，或安全地封存日誌以供未來分析。
- AWS Systems Manager - AWS Systems Manager 是一種 AWS 服務，可用來檢視和控制 AWS 上的基礎設施。使用 Systems Manager 主控台，您可以檢視來自多個 AWS 服務的操作資料，並自動化 AWS 資源的操作任務。Systems Manager 會掃描受管執行個體，並在偵測到任何政策違規時回報（或採取修正動作），以協助您維護安全性和合規性。

工具

- Visual Studio 或 Visual Studio 程式碼 – 用於建置 .NET 應用程式、Web APIs和其他程式的工具。
- AWS Toolkit for Visual Studio – Visual Studio 的延伸，可協助開發、偵錯和部署使用 AWS 服務的 .NET 應用程式。
- Docker Desktop – 一種工具，可簡化建置和部署容器化應用程式。
- NDepend – 監控 .NET 程式碼是否有相依性、品質問題和程式碼變更的分析器。
- 適用於 .NET 的移植助理 – 掃描 .NET 程式碼以識別與 .NET Core 不相容的分析工具，並估計遷移工作量。

史詩

將舊版應用程式移植到 .NET 5 或更新版本

任務	Description	所需的技能
將 your.NET Framework 舊版應用程式升級至 .NET 5。	您可以使用適用於 .NET 的移植助理，將舊版 ASP.NET Web Forms 應用程式轉換為 .NET 5 或更新版本。遵循適用於 .NET 的移植助理文件中的指示。	應用程式開發人員

任務	Description	所需的技能
產生 NDepend 報告。	當您將 ASP.NET Web Forms 應用程式分解為微服務來現代化時，您可能不需要舊版應用程式中的所有 .cs 檔案。您可以使用 NDepend 來產生任何程式碼後面 (.cs) 檔案的報告，以取得所有來電者和來電者。此報告可協助您識別和僅使用微服務中的必要檔案。 安裝 NDepend 後（請參閱先決條件區段），請在 Visual Studio 中開啟舊版應用程式的解決方案 (.sln 檔案），並遵循下列步驟： 1. 在 Visual Studio 中建置舊版應用程式。 2. 在 Visual Studio 選單列上，選擇 NDepend，將新的 NDepend 專案連接至目前的 VS 解決方案。 3. 選擇分析 .NET 組件。 4. 分析完成後，導覽至 Solution Explorer 中的專案。在您要產生報告的任何程式碼後面檔案（例如 `listproducts.aspx.cs` ）上按一下滑鼠右鍵，然後選擇依相依性圖表顯示。 5. 在導覽列中，選擇來電者和來電者，然後選擇編輯程式碼查詢。	應用程式開發人員

任務	Description	所需的技能
	6. 在查詢和規則編輯窗格中，選擇下載箭頭，然後選擇匯出至 Excel。 此程序會產生代碼後面檔案的報告，列出所有來電者和來電者。如需相依性圖表的詳細資訊，請參閱 NDepend 文件。	

任務	Description	所需的技能
建立新的 .NET 5 解決方案。	若要為現代化 ASP.NET Core Web APIs 建立新的 .NET 5（或更新版本）結構： 1. 開啟 Visual Studio。 2. 建立新的空白解決方案。 3. 根據您的舊版應用程式，建立以 .NET 5（或更新版本）為目標的新專案。如需購物車應用程式的舊版和新專案範例，請參閱其他資訊一節。 4. 使用上一個步驟的 NDepend 報告來識別所有必要檔案。從您先前升級的應用程式複製這些檔案，並將其新增至新的解決方案。 5. 建置解決方案並修正所有問題。 如需建立專案和解決方案的詳細資訊，請參閱 Visual Studio 文件。 **Note** 當您建置解決方案並驗證功能時，除了 NDepend 識別的檔案之外，您還可以識別要新增到解決方案的其他幾個檔案。	應用程式開發人員

更新您的應用程式程式碼

任務	Description	所需的技能
使用 ASP.NET Core 實作 Web APIs。	假設您在舊版單體購物車應用程式中識別的其中一個微服務是產品。您已在上一個史詩中為產品建立新的 ASP.NET Core Web API 專案。在此步驟中，您會識別和現代化與 產品相關的所有 Web 表單 (.aspx 頁面）。假設產品由四個 Web 表單組成，如先前架構一節所示： • 列出產品 • 檢視產品 • 新增/編輯產品 • 刪除產品 您應該分析每個 Web 表單、識別傳送至資料庫以執行一些邏輯的所有請求，以及取得回應。您可以實作每個請求做為 Web API 端點。鑑於其 Web 表單，產品可以有下列可能端點： • `/api/products` • `/api/products/{id}` • `/api/products/add` • `/api/products/update/{id}` • `/api/products/delete/{id}`	應用程式開發人員

任務	Description	所需的技能
	如前所述，您也可以重複使用升級至 .NET 5 的所有其他專案，包括商業邏輯、資料存取和共用/常見專案。	
設定 Amazon CloudWatch Logs。	您可以使用 Amazon CloudWatch Logs 來監控、存放和存取應用程式的日誌。您可以使用 AWS 開發套件將資料記錄到 Amazon CloudWatch Logs。您也可以使用 NLog、LogLog4Net 和 ASP.NET Core 記錄架構等熱門 .NET 記錄架構，將 .NET 應用程式與 CloudWatch Logs 整合。 如需此步驟的詳細資訊，請參閱部落格文章 Amazon CloudWatch Logs 和 .NET Logging Frameworks。	應用程式開發人員

任務	Description	所需的技能
設定 AWS Systems Manager 參數存放區。	您可以使用 AWS Systems Manager 參數存放區，將連線字串與應用程式的程式碼分開存放應用程式設定。NuGet packageAmazon.Extensions.Configuration.SystemsManager 可簡化您的應用程式如何將這些設定從 AWS Systems Manager 參數存放區載入 .NET Core 組態系統。 如需此步驟的詳細資訊，請參閱 AWS Systems Manager 的部落格文章 .NET Core 組態提供者。	應用程式開發人員

新增身分驗證和授權

任務	Description	所需的技能
使用共用 Cookie 進行身分驗證。	將傳統整體應用程式現代化是一種反覆程序，需要整體及其現代化版本才能共存。您可以使用共用 Cookie 來實現兩個版本之間的無縫身分驗證。舊版 ASP.NET 應用程式會繼續驗證使用者登入資料並發出 Cookie，同時現代化 ASP.NET Core 應用程式會驗證 Cookie。 如需說明和範例程式碼，請參閱範例 GitHub 專案。	應用程式開發人員

在本機建置和執行容器

任務	Description	所需的技能
使用 Visual Studio 建立 Docker 映像。	在此步驟中，您會使用 Visual Studio for .NET Core Web API 來建立 Docker 檔案。 1. 開啟 Visual Studio。 2. 在 Solution Explorer 中，從專案的內容（按一下滑鼠右鍵）功能表中，選擇新增、Docker Support。 3. 選取 Linux 做為目標作業系統。 Visual Studio 會為您的專案建立 Docker 檔案。如需 Docker 檔案範例，請參閱 Microsoft 網站上的適用於 Docker 的 Visual Studio 容器工具。	應用程式開發人員
使用 Docker 桌面建置和執行容器。	現在您可以在 Docker 桌面中建置、建立和執行容器。 1. 開啟命令提示視窗。導覽至 Docker 檔案所在的解決方案資料夾。執行下列命令來建立 Docker 映像： `docker build -t aspnetcorewebapiimage -f Dockerfile .` 2. 執行下列命令以檢視所有 Docker 映像：	應用程式開發人員

任務	Description	所需的技能

	`docker images`
	3. 執行下列命令來建立和執行容器：
	`docker run -d -p 8080:80 --name aspnetcorewebapicontainer aspnetcorewebapiimage`
	4. 開啟 Docker 桌面，然後選擇容器/應用程式。您可以看到名為 的新容器`aspnetcorewebapicontainer` 正在執行。	

相關資源

- 在 Amazon EC2 Linux 執行個體上執行 ASP.NET Core Web API Docker 容器 (AWS 方案指引）
- Amazon ECS 研討會
- 使用 AWS CloudFormation (AWS CloudFormation 文件） 透過 CodeDeploy 執行 ECS 藍/綠部署 AWS CloudFormation
- NDepend 入門 (NDepend 文件）
- .NET 的移植助理

其他資訊

下表提供舊版購物車應用程式的範例專案，以及現代化 ASP.NET Core 應用程式中的同等專案。

舊版解決方案：

Project name (專案名稱)	專案範本	目標架構

業務界面	類別程式庫	.NET Framework
BusinessLogic	類別程式庫	.NET Framework
WebApplication	ASP.NET Framework Web 應用程式	.NET Framework
UnitTests	NUnit 測試專案	.NET Framework
共用 -> 常見	類別程式庫	.NET Framework
共用 -> 架構	類別程式庫	.NET Framework

新解決方案：

Project name (專案名稱)	專案範本	目標架構
BusinessLogic	類別程式庫	.NET 5.0
<WebAPI>	ASP.NET Core Web API	.NET 5.0
<WebAPI>。UnitTests	NUnit 3 測試專案	.NET 5.0
共用 -> 常見	類別程式庫	.NET 5.0
共用 -> 架構	類別程式庫	.NET 5.0

使用 AWS Fargate 大規模執行事件驅動和排程工作負載

HARI OHM PRASATH RAJAGOPAL，Amazon Web Services

總結

此模式說明如何使用 AWS Fargate 在 Amazon Web Services (AWS) 雲端上大規模執行排程和事件驅動的工作負載。

在設定此模式的使用案例中，只要提交提取請求，就會掃描程式碼以取得 AWS 敏感資訊，例如 AWS 帳戶號碼和登入資料。提取請求會啟動 Lambda 函數。Lambda 函數會叫用負責程式碼掃描的 Fargate 任務。每當提出新的提取請求時，就會啟動 Lambda。如果掃描發現任何敏感資訊，Amazon Simple Notification Service (Amazon SNS) 會在電子郵件訊息中傳送掃描結果。

此模式在下列商業使用案例中很有用：

- 如果您的企業必須執行許多因執行時間 (15 分鐘限制） 或記憶體的限制而無法由 AWS Lambda 執行的排程和事件驅動工作負載
- 如果您希望 AWS 管理為這些工作負載佈建的執行個體

當您使用此模式時，您可以選擇建立新的虛擬私有雲端 (VPC)。此模式也會使用 AWS CodeCommit。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 用於託管程式碼庫和建立提取請求的 AWS CodeCommit
- 在 macOS、Linux 或 Windows 上安裝和設定 AWS Command Line Interface (AWS CLI) 1.7 版或更新版本
- 在容器中執行的工作負載
- 在 classpath 中設定 Apache Maven 可執行檔

Architecture

整體流程包含下列步驟。

1. 每當在 CodeCommit 中提交新的提取請求時，就會啟動 Lambda 函數。Lambda 函數會透過 Amazon EventBridge 接聽`CodeCommit Pull Request State Change`事件。
2. Lambda 函數會使用下列環境參數來提交新的 Fargate 任務，以檢查程式碼並進行掃描。

```
RUNNER # <<TaskARN>>
SNS_TOPIC # <<SNSTopicARN>>
SUBNET # <<Subnet in which Fargate task gets launched>>
```

   如果掃描在程式碼中找到敏感資訊，Fargate 會將新訊息推送到 Amazon SNS 主題。
3. SNS 訂閱者會從主題讀取訊息並傳送電子郵件訊息。

技術

- AWS CodeCommit
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon Elastic Container Service (Amazon ECS)
- Amazon EventBridge
- AWS Fargate
- AWS Lambda
- Amazon SNS
- Docker

工具

工具

- AWS CLI – AWS Command Line Interface (CLI) 是管理 AWS 服務的統一工具。
- AWS CodeCommit – AWS CodeCommit 是一種全受管的來源控制服務，可託管安全的 Git 型儲存庫。使用 CodeCommit，團隊可以在安全且可擴展的環境中協作處理程式碼。
- Amazon ECR – Amazon Elastic Container Registry (Amazon ECR) 是全受管登錄檔，開發人員可用來存放、管理和部署 Docker 容器映像。
- Amazon ECS – Amazon Elastic Container Service (Amazon ECS) 是一種高度可擴展的快速容器管理服務。您可以使用 Amazon ECS 來執行、停止和管理叢集上的容器。
- AWS Fargate – AWS Fargate 是一種技術，您可以與 Amazon ECS 搭配使用來執行容器，而無需管理 Amazon EC2 執行個體的伺服器或叢集。

- AWS Lambda – AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一種受管服務，可將訊息從發佈者交付給訂閱者（也稱為生產者和消費者）。發佈者透過製作並傳送訊息到主題 (其為邏輯存取點和通訊管道) 與訂閱者進行非同步的通訊。訂閱 SNS 主題的用戶端會使用支援的通訊協定接收已發佈的訊息，例如 Lambda、電子郵件、行動推播通知和行動文字訊息 (SMS)。
- Docker – Docker 可協助您在稱為容器的套件中建置、測試和交付應用程式。
- Git 用戶端 – 用來查看必要成品的命令列或桌面工具
- Maven – Apache Maven 是一種專案管理工具，可集中管理專案的建置、報告和文件。

史詩

設定本機儲存庫

任務	Description	所需的技能
下載程式碼。	在附件區段中，下載 .zip 檔案並解壓縮檔案。	開發人員、AWS 系統管理員
設定儲存庫。	在根資料夾`mvn clean install`上執行 。	開發人員、AWS 系統管理員

建立 Amazon ECR 映像並推送映像

任務	Description	所需的技能
建立 Amazon ECR 儲存庫並登入。	開啟 Amazon ECR 主控台。在導覽窗格中，選擇儲存庫，然後選擇建立儲存庫。如需此案例和其他案例的協助，請參閱相關資源一節。	開發人員、AWS 系統管理員
推送您的容器映像。	開啟儲存庫，選擇檢視推送命令，然後登入 Docker。登入後，在其他資訊區段中的推送容器映像下，使用必要的替	開發人員、AWS 系統管理員

任務	Description	所需的技能
	換執行命令。這會上傳用於執行程式碼掃描的 Docker 容器映像。上傳完成後，複製 Amazon ECR 儲存庫中最新建置的 URL。	

建立 CodeCommit 儲存庫

任務	Description	所需的技能
建立 CodeCommit 儲存庫。	若要建立新的 AWS CodeCommit 儲存庫，請在其他資訊區段中的建立 CodeCommit 儲存庫下執行 命令。	開發人員、AWS 系統管理員

建立 VPC （選用）

任務	Description	所需的技能
建立 VPC。	如果您想要使用新的 VPC 而非現有的 VPC，請在其他資訊區段中的建立 VPC 下執行命令。AWS Cloud Development Kit (AWS CDK) 指令碼將輸出已建立的 VPC 和子網路 IDs。	開發人員、AWS 系統管理員

建立 Amazon ECS 叢集和 Fargate 任務

任務	Description	所需的技能
建立叢集和任務。	若要建立 Amazon ECS 叢集和 Fargate 任務定義，請在其	開發人員、AWS 系統管理員

任務	Description	所需的技能
	他資訊區段中的建立叢集和任務下執行命令。執行 shell 指令碼時，請確定傳入正確的 VPC ID 和 Amazon ECR 儲存庫 URI 做為參數。指令碼會建立指向 Docker 映像的 Fargate 任務定義（負責掃描）。指令碼接著會建立任務和相關聯的執行角色。	
驗證 Amazon ECS 叢集。	開啟 Amazon ECS 主控台。在導覽窗格中，選擇叢集，然後選擇新建立的名為 Fargate-Job-Cluster 的 Amazon ECS 叢集。之後，在導覽窗格中選擇任務定義，並確認有新的任務定義，字首為 `awscdkfargateecsTaskDef` 。	開發人員、AWS 系統管理員

建立 SNS 主題和訂閱者

任務	Description	所需的技能
建立 SNS 主題。	若要建立 SNS 主題，請在其他資訊區段中的建立 SNS 主題下執行 命令。建立成功後，請注意在下一個步驟`SNS ARN`中使用的 。	開發人員、AWS 系統管理員
建立 SNS 訂閱者。	若要為 SNS 主題建立電子郵件訂閱者，請在其他資訊區段中的建立 SNS 訂閱者下執行 命令。請務必取代 `TopicARN`，並在 CLI 命令`Email address`中使用 。	開發人員、AWS 系統管理員

任務	Description	所需的技能
	若要接收電子郵件通知，請務必確認做為訂閱者使用的電子郵件地址。	

建立 Lambda 函數和 CodeCommit 觸發

任務	Description	所需的技能
建立 函數和 觸發。	若要使用 CodeCommit 觸發條件建立 Lambda 函數，請在其他資訊區段中的 Lambda 函數和 CodeCommit 觸發條件下執行 命令。執行 命令之前，請務必將參數取代為對應的值。指令碼會建立 Lambda 函數，並將其設定為在提出新的提取請求時叫用。	開發人員、AWS 系統管理員

測試應用程式。

任務	Description	所需的技能
測試應用程式。	如果您簽入 CodeCommit 儲存庫的任何 AWS 敏感資訊，應該啟動 Lambda 函數。Lambda 函數會啟動 Fargate 任務，這會掃描程式碼，並在電子郵件通知中傳送掃描結果。	開發人員、AWS 系統管理員

相關資源

- 建立 Amazon ECR 儲存庫

- 將 Docker 映像推送至 Amazon ECR

其他資訊

推送容器映像

```
> cd 1-ecr-image-push
> ./run.sh <<ecr-repository>>
```

建立 CodeCommit 儲存庫

```
aws codecommit create-repository --repository-name test-repo --repository-description
 "My Test repository"
```

建立 VPC

```
> cd 2-create-vpc
> ./run.sh
```

輸出

```
aws-batch-cdk-vpc-efs-launch-template.privatesubnet = subnet-<<id>>
aws-batch-cdk-vpc-efs-launch-template.publicsubnet = subnet-<<id>>
aws-batch-cdk-vpc-efs-launch-template.vpcid = vpc-<<id>>
```

建立叢集和任務

```
> export CDK_DEFAULT_ACCOUNT = <<aws_account_id>>
> export CDK_DEFAULT_REGION = <<aws_region>>
> cd 3-create-ecs-task
> ./run.sh <<vpc-id>> <<ecr-repo-uri>>
```

輸出

```
aws-cdk-fargate-ecs.CLUSTERNAME = Fargate-Job-Cluster
aws-cdk-fargate-ecs.ClusterARN = <<cluster_arn>>
aws-cdk-fargate-ecs.ContainerARN = Fargate-Container
aws-cdk-fargate-ecs.TaskARN = <<task_arn>>
```

```
aws-cdk-fargate-ecs.TaskExecutionRole = <<execution_role_arn>>
aws-cdk-fargate-ecs.TaskRole = <<task_role_arn>>
```

建立 SNS 主題

```
aws sns create-topic --name code-commit-topic
```

建立 SNS 訂閱者

```
aws sns subscribe \
    --topic-arn <<topic_arn>> \
    --protocol email \
    --notification-endpoint <<email_address>>
```

Lambda 函數和CodeCommit 觸發

```
> export CDK_DEFAULT_ACCOUNT = <<aws_account_id>>
> export CDK_DEFAULT_REGION = <<aws_region>>
> cd 5-Lambda-CodeCommit-Trigger
> ./run.sh <<taskarn>> <<snstopicarn>> subnet-<<id>> <<codecommitarn>>
```

輸出

```
aws-cdk-fargate-lambda-event.Cloudwatchrule = <<cloudwatchrule>>
aws-cdk-fargate-lambda-event.CodeCommitLambda = AWS-Code-Scanner-Function
aws-cdk-fargate-lambda-event.LambdaRole = <<lambdaiamrole>>
```

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 C# 和 AWS CDK 在孤立模型的 SaaS 架構中加入租用戶

Tabby Ward、Susmitha Reddy Gankidi 和 Vijai Anand Ramalingam，Amazon Web Services

總結

軟體即服務 (SaaS) 應用程式可以使用各種不同的架構模型建置。孤島模型是指提供租戶專用資源的架構。

SaaS 應用程式依賴無摩擦模型，將新租戶引入其環境。這通常需要協調多個元件，才能成功佈建和設定建立新租用戶所需的所有元素。在 SaaS 架構中，此程序稱為租戶加入。應針對每個 SaaS 環境使用基礎設施做為加入程序中的程式碼，以完全自動化加入。

此模式會引導您完成在 Amazon Web Services (AWS) 上建立租用戶和佈建租用戶基本基礎設施的範例。模式使用 C# 和 AWS 雲端開發套件 (AWS CDK)。

由於此模式會建立帳單警示，因此建議您在美國東部（維吉尼亞北部）或 us-east-1 AWS 區域部署堆疊。如需詳細資訊，請參閱 AWS 文件。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶。
- 具有足夠 IAM 存取權的 AWS Identity and Access Management (IAM) 主體，可為此模式建立 AWS 資源。如需詳細資訊，請參閱 IAM 角色。
- 安裝 Amazon Command Line Interface (AWS CLI) 並設定 AWS CLI 以執行 AWS CDK 部署。
- Visual Studio 2022 已下載並安裝，或 Visual Studio Code 已下載並安裝。
- AWS Toolkit for Visual Studio 設定。
- .NET Core 3.1 或更新版本 (C# AWS CDK 應用程式需要）
- 已安裝 Amazon.Lambda.Tools。

限制

- AWS CDK 使用 AWS CloudFormation，因此 AWS CDK 應用程式受限於 CloudFormation 服務配額。如需詳細資訊，請參閱 AWS CloudFormation 配額。
- 租用戶 CloudFormation 堆疊是使用 CloudFormation 服務角色建立的，`infra-cloudformation-role`其動作上有萬用字元 (`sns*` 和 `sqs*`)，但資源鎖定為`tenant-cluster`字首。對於生產使

用案例，請評估此設定，並僅提供對此服務角色的必要存取權。`InfrastructureProvision` Lambda 函數也會使用萬用字元 (`cloudformation*`) 來佈建 CloudFormation 堆疊，但資源會鎖定為`tenant-cluster`字首。

- 此範例程式碼的 docker 組建使用 `--platform=linux/amd64` 來強制以映像`linux/amd64`為基礎。這是為了確保最終影像成品適用於 Lambda，依預設會使用 x86-64 架構。如果您需要變更目標 Lambda 架構，請務必同時變更 Dockerfile 和 AWS CDK 代碼。如需詳細資訊，請參閱部落格文章將 AWS Lambda 函數遷移至 Arm 型 AWS Graviton2 處理器。
- 堆疊刪除程序不會清除堆疊產生的 CloudWatch Logs （日誌群組和日誌）。您必須透過 AWS 管理主控台 Amazon CloudWatch 主控台或透過 API 手動清除日誌。

此模式設定為範例。針對生產用途，請評估下列設定，並根據您的業務需求進行變更：

- 此範例中的 AWS Simple Storage Service (Amazon S3) 儲存貯體尚未啟用版本控制以簡化操作。視需要評估和更新設定。
- 此範例設定 Amazon API Gateway REST API 端點，無需身分驗證、授權或限流即可簡化。對於生產用途，我們建議您將系統與商業安全基礎設施整合。評估此設定並視需要新增必要的安全設定。
- 在此租戶基礎設施範例中，Amazon Simple Notification Service (Amazon SNS) 和 Amazon Simple Queue Service (Amazon SQS) 只有最低設定。每個租用戶的 AWS Key Management Service (AWS KMS) 會開啟帳戶中的 Amazon CloudWatch 和 Amazon SNS 服務，以根據 AWS KMS 金鑰政策使用。設定只是範例預留位置。根據您的業務使用案例，視需要調整設定。
- 整個設定包括但不限於 API 端點，以及使用 AWS CloudFormation 進行後端租用戶佈建和刪除，僅涵蓋基本的快樂路徑案例。根據您的業務需求，使用必要的重試邏輯、額外的錯誤處理邏輯和安全邏輯來評估和更新設定。
- 範例程式碼使用up-to-date cdk-nag 進行測試，以在撰寫本文時檢查政策。未來可能會強制執行新政策。這些新政策可能需要您根據建議手動修改堆疊，才能部署堆疊。檢閱現有的程式碼，以確保其符合您的業務需求。
- 此程式碼倚賴 AWS CDK 產生隨機尾碼，而不是倚賴大多數建立資源的靜態指派實體名稱。此設定旨在確保這些資源是唯一的，並且不會與其他堆疊衝突。如需詳細資訊，請參閱 AWS CDK 文件。根據您的業務需求進行調整。
- 此範例程式碼會將 .NET Lambda 成品封裝至以 Docker 為基礎的映像，並使用 Lambda 提供的容器映像執行時間執行。容器映像執行時間對於標準傳輸和儲存機制（容器登錄檔）和更準確的本機測試環境（透過容器映像）具有優勢。您可以切換專案以使用 Lambda 提供的 .NET 執行時間，以減少 Docker 映像的建置時間，但接著您需要設定傳輸和儲存機制，並確保本機設定符合 Lambda 設定。調整程式碼以符合使用者的業務需求。

產品版本

- AWS CDK 2.45.0 版或更新版本
- Visual Studio 2022

Architecture

技術堆疊

- Amazon API Gateway
- AWS CloudFormation
- Amazon CloudWatch
- Amazon DynamoDB
- AWS Identity and Access Management (IAM)
- AWS KMS
- AWS Lambda
- Amazon S3
- Amazon SNS
- Amazon SQS

架構

下圖顯示租戶堆疊建立流程。如需控制平面和租戶技術堆疊的詳細資訊，請參閱其他資訊一節。

租戶堆疊建立流程

1. 使用者將具有 JSON 中新租用戶承載（租用戶名稱、租用戶描述）的 POST API 請求傳送至 Amazon API Gateway 託管的 REST API。API Gateway 會處理請求，並將其轉送至後端 Lambda 租戶加入函數。在此範例中，沒有授權或身分驗證。在生產設定中，此 API 應與 SaaS 基礎設施安全系統整合。
2. 租戶加入函數會驗證請求。然後，它會嘗試將租用戶記錄存放在 Amazon DynamoDB 租用戶加入資料表中，其中包含租用戶名稱、產生的租用戶通用唯一識別符 (UUID) 和租用戶描述。
3. DynamoDB 儲存記錄後，DynamoDB 串流會啟動下游 Lambda 租用戶基礎設施函數。

4. 租戶基礎設施 Lambda 函數會根據收到的 DynamoDB 串流來運作。如果串流適用於 INSERT 事件，則函數會使用串流的 NewImage 區段（最新更新記錄，租戶名稱欄位）來叫用 CloudFormation，以使用存放在 S3 儲存貯體中的範本建立新的租戶基礎設施。CloudFormation 範本需要租用戶名稱參數。
5. AWS CloudFormation 會根據 CloudFormation 範本和輸入參數建立租戶基礎設施。
6. 每個租戶基礎設施設定都有 CloudWatch 警示、帳單警示和警示事件。
7. 警示事件會變成 SNS 主題的訊息，由租戶的 AWS KMS 金鑰加密。
8. SNS 主題會將收到的警示訊息轉送至 SQS 佇列，該佇列由租戶的 AWS KMS 加密以進行加密金鑰。

其他系統可與 Amazon SQS 整合，以根據佇列中的訊息執行動作。在此範例中，若要保持程式碼通用，傳入的訊息會保留在佇列中，且需要手動刪除。

租戶堆疊刪除流程

1. 使用者將具有 JSON 中新租用戶承載（租用戶名稱、租用戶描述）的 DELETE API 請求傳送至 Amazon API Gateway 託管的 REST API，這會處理請求並轉送至租用戶加入函數。在此範例中，沒有授權或身分驗證。在生產設定中，此 API 將與 SaaS 基礎設施安全系統整合。
2. 租戶加入函數將驗證請求，然後嘗試從租戶加入資料表中刪除租戶記錄（租戶名稱）。
3. DynamoDB 成功刪除記錄後（記錄存在於資料表中並刪除），DynamoDB 串流會啟動下游 Lambda 租用戶基礎設施函數。
4. 租戶基礎設施 Lambda 函數會根據收到的 DynamoDB 串流記錄運作。如果串流是用於 REMOVE 事件，則函數會使用記錄的 OldImage 區段（記錄資訊和租用戶名稱欄位，在最新變更之前為刪除），根據該記錄資訊啟動現有堆疊的刪除。
5. AWS CloudFormation 會根據輸入刪除目標租用戶堆疊。

工具

AWS 服務

- Amazon API Gateway 可協助您建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。
- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS Cloud 基礎設施。

- AWS CDK Toolkit 是命令列雲端開發套件，可協助您與 AWS Cloud Development Kit (AWS CDK) 應用程式互動。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- AWS CloudFormation 可協助您設定 AWS 資源、快速一致地佈建資源，以及在整個 AWS 帳戶和區域的生命週期進行管理。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而不需要佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- Amazon Simple Queue Service (Amazon SQS) 提供安全、耐用且可用的託管佇列，可協助您整合和分離分散式軟體系統和元件。
- AWS Toolkit for Visual Studio 是 Visual Studio 整合開發環境 (IDE) 的外掛程式。Toolkit for Visual Studio 支援開發、偵錯和部署使用 AWS 服務的 .NET 應用程式。

其他工具

- Visual Studio 是一種 IDE，其中包含編譯器、程式碼完成工具、圖形設計師和其他支援軟體開發的功能。

Code

此模式的程式碼位於 SaaS Architecture for Silo Model APG 範例儲存庫中的租戶加入中。

史詩

設定 AWS CDK

任務	Description	所需的技能
驗證 Node.js 安裝。	若要確認您的本機電腦上已安裝 Node.js，請執行下列命令。 `node --version`	AWS 管理員、AWS DevOps
安裝 AWS CDK Toolkit。	若要在本機電腦上安裝 AWS CDK Toolkit，請執行下列命令。 `npm install -g aws-cdk` 如果未安裝 npm，您可以從 Node.js 網站安裝。	AWS 管理員、AWS DevOps
驗證 AWS CDK Toolkit 版本。	若要確認您的電腦上已正確安裝 AWS CDK Toolkit 版本，請執行下列命令。 `cdk --version`	AWS 管理員、AWS DevOps

檢閱租戶加入控制平面的程式碼

任務	Description	所需的技能
複製儲存庫。	複製儲存庫，然後導覽至 `\tenant-onboarding-in-saas-architecture-for-silo-model-apg-example` 資料夾。	AWS 管理員、AWS DevOps

任務	Description	所需的技能
	在 Visual Studio 2022 中，開啟`\src\TenantOnboardingInfra.sln` 解決方案。開啟 `TenantOnboardingInfraStack.cs` 檔案並檢閱程式碼。 下列資源會建立為此堆疊的一部分： • DynamoDB 表 • S3 儲存貯體（將 CloudFormation 範本上傳至 S3 儲存貯體。） • Lambda 執行角色 • Lambda 函式 • API Gateway API • Lambda 函數的事件來源	
檢閱 CloudFormation 範本。	在 `\tenant-onboarding-in-saas-architecture-for-silo-model-apg-example\template` 資料夾中，開啟 `infra.yaml` 並檢閱 CloudFormation 範本。此範本將使用從租戶加入 DynamoDB 資料表擷取的租戶名稱進行補充。 範本會佈建租戶特定的基礎設施。在此範例中，它會佈建 AWS KMS 金鑰、Amazon SNS、Amazon SQS 和 CloudWatch 警示。	應用程式開發人員、AWS DevOps

任務	Description	所需的技能
檢閱租戶加入函數。	開啟 `Function.cs` ，並檢閱租戶加入函數的程式碼，該函數是使用 Visual Studio AWS Lambda 專案 (.NET Core- C#) 範本搭配 .NET 6 （容器映像）藍圖所建立。 開啟 `Dockerfile` ，並檢閱程式碼。`Dockerfile` 是一個文字檔案，其中包含建置 Lambda 容器映像的說明。 請注意，下列 NuGet 套件會新增為`TenantOnboardingFunction` 專案的相依性： • `Amazon.Lambda.APIGatewayEvents` • `AWSSDK.DynamoDBv2` • `Newtonsoft.Json`	應用程式開發人員、AWS DevOps

任務	Description	所需的技能
檢閱租戶 InfraProvisioning 函數。	導覽至 `\tenant-onboarding-in-saas-architecture-for-silo-model-apg-example\src\InfraProvisioningFunction` 。 開啟 `Function.cs` ，並檢閱租用戶基礎設施佈建函數的程式碼，該函數是使用 Visual Studio AWS Lambda 專案 (.NET Core- C#) 範本搭配 .NET 6 （容器映像） 藍圖所建立。 開啟 `Dockerfile` ，並檢閱程式碼。 請注意，下列 NuGet 套件會新增為`InfraProvisioningFunction` 專案的相依性： • `Amazon.Lambda.DynamoDBEvents` • `AWSSDK.DynamoDBv2` • `AWSSDK.Cloudformation`	應用程式開發人員、AWS DevOps

部署 AWS 資源

任務	Description	所需的技能
建置解決方案。	若要建置解決方案，請執行下列步驟：	應用程式開發人員

任務	Description	所需的技能
	1. 在 Visual Studio 2022 中，開啟\tenant-onboarding-in-saas-architecture-for-silo-model-apg-example\src\TenantOnboardingInfra.sln 解決方案。 2. 開啟解決方案的內容（按一下滑鼠右鍵）選單，然後選擇建置解決方案。 **Note** 在建置解決方案之前，請務必將Amazon.CDK.Lib NuGet 套件更新至\tenant-onboarding-in-saas-architecture-for-silo-model-apg-example\src\TenantOnboardingInfra 專案中的最新版本。	

任務	Description	所需的技能

引導 AWS CDK 環境。	開啟 Windows 命令提示字元，並導覽至可使用 cdk.json 檔案的 AWS CDK 應用程式根資料夾 (\tenant-onboarding-in-saas-architecture-for-silo-model-apg-example)。執行下列命令進行引導。
	`cdk bootstrap`
	如果您已為登入資料建立 AWS 設定檔，請使用 命令搭配您的設定檔。
	`cdk bootstrap --profile <profile name>`
		AWS 管理員、AWS DevOps

列出 AWS CDK 堆疊。	若要列出要建立做為此專案一部分的所有堆疊，請執行下列命令。
	`cdk ls`
	`cdk ls --profile <profile name>`
	如果您已為登入資料建立 AWS 設定檔，請使用 命令搭配您的設定檔。
	`cdk ls --profile <profile name>`
		AWS 管理員、AWS DevOps

任務	Description	所需的技能

檢閱要建立的 AWS 資源。	若要檢閱將建立為此專案一部分的所有 AWS 資源，請執行下列命令。
	`cdk diff`
	如果您已為登入資料建立 AWS 設定檔，請使用 命令搭配您的設定檔。
	`cdk diff --profile <profile name>`
		AWS 管理員、AWS DevOps

任務	Description	所需的技能

使用 AWS CDK 部署所有 AWS 資源。	若要部署所有 AWS 資源，請執行下列命令。
	`cdk deploy --all --require-approval never`
	如果您已為登入資料建立 AWS 設定檔，請使用 命令搭配您的設定檔。
	`cdk deploy --all --require-approval never --profile <profile name>`
	部署完成後，請從命令提示中的輸出區段複製 API URL，如下列範例所示。
	`Outputs: TenantOnboardingInfraStack.TenantOnboardingAPIEndpoint42E526D7 = https://j2qmp8ds21i1i.execute-api.us-west-2.amazonaws.com/prod/`
		AWS 管理員、AWS DevOps

驗證功能

任務	Description	所需的技能
建立新的租用戶。	若要建立新的租用戶，請傳送下列 curl 請求。	應用程式開發人員、AWS 管理員、AWS DevOps

任務	Description	所需的技能

	`curl -X POST <TenantOnboardingAPIEndpoint* from CDK Output>tenant -d '{"Name":"Tenant123", "Description":"Stack for Tenant123"}'`
	
	將預留位置變更為 AWS CDK `<TenantOnboardingAPIEndpoint* from CDK Output>`的實際值，如下列範例所示。
	
	`curl -X POST https://j2qmp8ds21i1i.execute-api.us-west-2.amazonaws.com/prod/tenant -d '{"Name": "Tenant123", "Description":"test12"}'`
	
	下列範例顯示輸出。
	
	`{"message": "A new tenant added - 5/4/2022 7:11:30 AM"}`	

任務	Description	所需的技能
驗證 DynamoDB 中新建立的租戶詳細資訊。	若要在 DynamoDB 中驗證新建立的租戶詳細資訊，請執行下列步驟。 1. 開啟 AWS 管理主控台，然後導覽至 Amazon DynamoDB 服務。 2. 在左側導覽中，選擇探索項目，然後選擇`TenantOnboarding` 資料表。 **Note** 租戶名稱會加上 `tenantcluster-` 。如需詳細資訊，請參閱其他資訊一節。 3. 確認已使用租戶詳細資訊建立新項目。	應用程式開發人員、AWS 管理員、AWS DevOps

任務	Description	所需的技能

驗證新租用戶的堆疊建立。	根據 CloudFormation 範本，確認新堆疊已成功使用新建立租用戶的基礎設施建立和佈建。
	1. 開啟 CloudFormation 主控台。
	2. 在左側導覽中，選擇 Stacks，並確認已成功建立具有租用戶名稱的堆疊。
	3. 選擇新建立的租戶堆疊，然後選擇資源索引標籤。請注意警示資源和 Amazon SQS 資源。
	4. 開啟已設定 AWS 登入資料的新終端機，並指向正確的區域。若要引發測試警示，請輸入下列程式碼，`<alarm resource name>`將 取代為步驟 3 中記下的警示資源名稱。
	`aws cloudwatch set-alarm-state --alarm-name <alarm resource name> --state-value ALARM --state-reason 'Test setup'`
	下列範例顯示具有警示資源名稱的程式碼。
	`aws cloudwatch set-alarm-state --alarm-name tenantcluster-tenant123-alarm --state-value ALARM --`
		應用程式開發人員、AWS 管理員、AWS DevOps

任務	Description	所需的技能
	`state-reason 'Test setup'` 5. 開啟主控台並導覽至 Amazon SQS 主控台。選擇步驟 3 中識別的 Amazon SQS 資源名稱。遵循 AWS 文件指示，從步驟 4 中引發的警示接收和刪除測試訊息。	

任務	Description	所需的技能

刪除租戶堆疊。	若要刪除租戶堆疊，請傳送下列 curl 請求。
	
	`curl -X DELETE <TenantOnboardingAPIEndpoint* from CDK Output>tenant/<Tenant Name from previous step>`
	
	將預留位置`<TenantOnboardingAPIEndpoint* from CDK Output>` 變更為 AWS CDK 的實際值，並將 `<Tenant Name from previous step>`變更為上一個租用戶建立步驟的實際值，如下列範例所示。
	
	`curl -X DELETE https://j2qmp8ds21i1i.execute-api.us-west-2.amazonaws.com/prod/tenant/Tenant123`
	
	下列範例顯示輸出。
	
	`{"message": "Tenant destroyed - 5/4/2022 7:14:48 AM"}`
		應用程式開發人員、AWS DevOps、AWS 管理員

任務	Description	所需的技能
驗證現有租用戶的堆疊刪除。	若要確認現有租用戶堆疊已刪除，請執行下列步驟： 1. 開啟主控台並導覽至 CloudFormation 主控台。 2. 在左側導覽中，確認具有租用戶名稱的現有堆疊已不在主控台中（如果 CloudFormation 主控台設定為僅顯示作用中堆疊），或正在刪除中。如果堆疊不再位於 CloudFormation 主控台中，請使用下拉式清單將主控台的設定從作用中變更為已刪除，以查看已刪除的堆疊，並確認堆疊已成功刪除。	應用程式開發人員、AWS 管理員、AWS DevOps

清除

任務	Description	所需的技能
銷毀環境。	在清除堆疊之前，請確定下列事項： • DynamoDB 中的所有記錄都會透過先前的租用戶刪除操作，或透過 DynamoDB 主控台或 API 移除。每個刪除租戶記錄都會啟動其 AWS CloudFormation 對等的清除。 • 所有以租用戶為基礎的 AWS CloudFormation 堆疊都會在 AWS CloudFormation 主控	AWS 管理員、AWS DevOps

任務	Description	所需的技能
	台上清除（如果 DynamoDB 觸發程序清除邏輯失敗）。 測試完成後，AWS CDK 可以透過執行下列命令來銷毀所有堆疊和相關資源。 `cdk destroy --all;` 如果您已為登入資料建立 AWS 設定檔，請使用該設定檔。 確認堆疊刪除提示以刪除堆疊。	
清除 Amazon CloudWatch Logs。	堆疊刪除程序不會清除堆疊產生的 CloudWatch Logs（日誌群組和日誌）。使用 CloudWatch 主控台或 API 手動清除 CloudWatch 資源。	應用程式開發人員、AWS DevOps、AWS 管理員

相關資源

- AWS CDK .NET 研討會
- 在 C# 中使用 AWS CDK
- CDK .NET 參考

其他資訊

控制平面技術堆疊

以 .NET 撰寫的 CDK 程式碼用於佈建控制平面基礎設施，其中包含下列資源：

1. API Gateway

做為控制平面堆疊的 REST API 進入點。

2. 租用戶加入 Lambda 函數

   此 Lambda 函數是由 API Gateway 使用 m 方法啟動。

   POST 方法 API 請求會導致 (`tenant name`、`tenant description`) 插入 DynamoDB `Tenant Onboarding`資料表。

   在此程式碼範例中，租用戶名稱也會用作租用戶堆疊名稱的一部分，以及該堆疊內資源的名稱。這是為了讓這些資源更容易識別。此租用戶名稱在整個設定中必須是唯一的，以避免衝突或錯誤。詳細的輸入驗證設定會在 IAM 角色文件和限制一節中說明。

   DynamoDB 資料表的持久性程序只有在未用於資料表中任何其他記錄時，才會成功。

   在這種情況下，租戶名稱是此表格的分割區索引鍵，因為只有分割區索引鍵可以用作`PutItem`條件表達式。

   如果先前從未記錄過租戶名稱，則記錄將成功儲存至資料表。

   不過，如果 資料表中的現有記錄已使用租用戶名稱，則操作將會失敗並啟動 DynamoDB `ConditionalCheckFailedException`例外狀況。例外狀況將用於傳回失敗訊息 (`HTTP BadRequest`)，指出租用戶名稱已存在。

   `DELETE` 方法 API 請求會從 `Tenant Onboarding` 資料表中移除特定租用戶名稱的記錄。

   即使記錄不存在，此範例中的 DynamoDB 記錄刪除仍會成功。

   如果目標記錄存在且已刪除，則會建立 DynamoDB 串流記錄。否則，將不會建立任何下游記錄。

3. 啟用 Amazon DynamoDB Streams 的租戶加入 DynamoDB

   這會記錄租戶中繼資料資訊，而任何記錄儲存或刪除都會將串流傳送至 `Tenant Infrastructure` Lambda 函數下游。

4. 租戶基礎設施 Lambda 函數

   此 Lambda 函數是由上一個步驟的 DynamoDB 串流記錄啟動。如果記錄是針對`INSERT`事件，它會叫用 AWS CloudFormation，以使用存放在 S3 儲存貯體中的 CloudFormation 範本建立新的租用戶基礎設施。如果記錄適用於 `REMOVE`，則會根據串流記錄的 `Tenant Name` 欄位啟動現有堆疊的刪除。

5. S3 bucket (S3 儲存貯體)

這是用於存放 CloudFormation 範本。

6. 每個 Lambda 函數的 IAM 角色和 CloudFormation 的服務角色

   每個 Lambda 函數都有其唯一的 IAM 角色，具有實現其任務的最低權限許可。例如，`Tenant On-boarding`Lambda 函數具有 DynamoDB 的讀取/寫入存取權，Lambda `Tenant Infrastructure` 函數只能讀取 DynamoDB 串流。

   為租戶堆疊佈建建立自訂 CloudFormation 服務角色。此服務角色包含 CloudFormation 堆疊佈建的其他許可（例如 AWS KMS 金鑰）。這會在 Lambda 和 CloudFormation 之間分割角色，以避免單一角色（基礎設施 Lambda 角色）的所有許可。

   允許強大動作（例如建立和刪除 CloudFormation 堆疊）的許可會遭到鎖定，且僅允許在以 開頭的資源上`tenantcluster-`進行。例外狀況是 AWS KMS，因為其資源命名慣例。從 API 擷取的租用戶名稱將與其他驗證檢查`tenantcluster-`一起加上（僅含破折號的英數字元，且限制為少於 30 個字元，以符合大多數 AWS 資源命名）。這可確保租戶名稱不會意外導致核心基礎設施堆疊或資源中斷。

租戶技術堆疊

CloudFormation 範本存放在 S3 儲存貯體中。範本會佈建租戶特定的 AWS KMS 金鑰、CloudWatch 警示、SNS 主題、SQS 佇列和 SQS 政策。

Amazon SNS 和 Amazon SQS 會將 AWS KMS 金鑰用於其訊息的資料加密。AwsSolutions-SNS2 和 AwsSolutions-SQS2 的安全實務建議您使用加密設定 Amazon SNS 和 Amazon SQS。不過，使用 AWS 受管金鑰時，CloudWatch 警示不適用於 Amazon SNS，因此在這種情況下，您必須使用客戶受管金鑰。如需詳細資訊，請參閱 AWS 知識中心。

SQS 政策用於 Amazon SQS 佇列，以允許建立的 SNS 主題將訊息傳遞至佇列。如果沒有 SQS 政策，存取將被拒絕。如需詳細資訊，請參閱 Amazon SNS 文件。

使用 CQRS 和事件來源將整體分解為微服務

Rodolfo Jr. Cerrada、Dmitry Gulin 和 Tabby Ward，Amazon Web Services

總結

此模式結合兩種模式，同時使用命令查詢責任分離 (CQRS) 模式和事件來源模式。CQRS 模式會區隔命令和查詢模型的責任。事件來源模式會利用非同步事件驅動的通訊來改善整體使用者體驗。

您可以使用 CQRS 和 Amazon Web Services (AWS) 服務獨立維護和擴展每個資料模型，同時將整體應用程式重構為微服務架構。然後，您可以使用事件來源模式，將資料從命令資料庫同步到查詢資料庫。

此模式使用包含解決方案 (*.sln) 檔案的範例程式碼，您可以使用最新版本的 Visual Studio 開啟該檔案。此範例包含獎勵 API 程式碼，示範 CQRS 和事件來源如何在 AWS 無伺服器和傳統或內部部署應用程式中運作。

若要進一步了解 CQRS 和事件來源，請參閱其他資訊一節。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Amazon CloudWatch
- Amazon DynamoDB 資料表
- Amazon DynamoDB Streams
- AWS Identity and Access Management (IAM) 存取金鑰和私密金鑰；如需詳細資訊，請參閱相關資源區段中的影片
- AWS Lambda
- 熟悉 Visual Studio
- 熟悉 AWS Toolkit for Visual Studio；如需詳細資訊，請參閱相關資源區段中的 AWS Toolkit for Visual Studio 示範影片

產品版本

- Visual Studio 2019 Community Edition。
- AWS Toolkit for Visual Studio 2019。

- .NET Core 3.1。此元件是 Visual Studio 安裝中的選項。若要在安裝期間包含 .NET Core，請選取 NET Core 跨平台開發。

限制

- 傳統內部部署應用程式 (ASP.NET Core Web API 和資料存取物件） 的範例程式碼不會隨附資料庫。不過，它隨附記憶體`CustomerData`內物件，可做為模擬資料庫。提供的程式碼足以讓您測試模式。

Architecture

來源技術堆疊

- ASP.NET Core Web API 專案
- IIS Web 伺服器
- 資料存取物件
- CRUD 模型

來源架構

在來源架構中，CRUD 模型在一個應用程式中包含命令和查詢介面。如需範例程式碼，請參閱 `CustomerDAO.cs`（已連接）。

目標技術堆疊

- Amazon DynamoDB
- Amazon DynamoDB Streams
- AWS Lambda
- （選用） Amazon API Gateway
- （選用） Amazon Simple Notification Service (Amazon SNS)

目標架構

在目標架構中，命令和查詢界面會分開。下圖中顯示的架構可以使用 API Gateway 和 Amazon SNS 擴充。如需詳細資訊，請參閱其他資訊一節。

1. 命令 Lambda 函數會在資料庫上執行寫入操作，例如建立、更新或刪除。
2. 查詢 Lambda 函數會在資料庫上執行讀取操作，例如取得或選取。
3. 此 Lambda 函數會處理來自 命令資料庫的 DynamoDB 串流，並更新查詢資料庫以進行變更。

工具

工具

- Amazon DynamoDB – Amazon DynamoDB 是全受管的 NoSQL 資料庫服務，可提供快速且可預測的效能和無縫的可擴展性。
- Amazon DynamoDB Streams – DynamoDB Streams 會擷取任何 DynamoDB 資料表中項目層級修改的時間順序序列。然後，它會將此資訊存放在日誌中長達 24 小時。靜態加密功能會加密 DynamoDB Streams 中的資料。
- AWS Lambda – AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。只需為使用的運算時間支付費用，一旦未執行程式碼，就會停止計費。
- AWS 管理主控台 – AWS 管理主控台是一種 Web 應用程式，包含用於管理 AWS 服務的廣泛服務主控台集合。
- Visual Studio 2019 Community Edition – Visual Studio 2019 是整合式開發環境 (IDE)。開放原始碼參與者可免費使用 Community Edition。在此模式中，您將使用 Visual Studio 2019 Community Edition 來開啟、編譯和執行範例程式碼。僅供檢視，您可以使用任何文字編輯器或 Visual Studio 程式碼。
- AWS Toolkit for Visual Studio – AWS Toolkit for Visual Studio 是 Visual Studio IDE 的外掛程式。AWS Toolkit for Visual Studio 可讓您更輕鬆地開發、偵錯和部署使用 AWS 服務的 .NET 應用程式。

Code

已連接範例程式碼。如需部署範例程式碼的指示，請參閱 Epics 一節。

史詩

開啟並建置解決方案

任務	Description	所需的技能
開啟解決方案。	1. 從附件區段下載範例原始程式碼 (CQRS-ES Code.zip)，然後解壓縮檔案。 2. 在 Visual Studio IDE 中，選擇檔案、開啟、專案解決方案，然後導覽至您解壓縮原始程式碼的資料夾。 3. 選擇 AWS.APG.CQRSES.sln，然後選擇開啟。整個解決方案會載入 Visual Studio。	應用程式開發人員
建置解決方案。	開啟解決方案的內容（按一下滑鼠右鍵）選單，然後選擇建置解決方案。這將建置和編譯解決方案中的所有專案。它應該可以成功編譯。 Visual Studio Solution Explorer 應會顯示目錄結構。 • `CQRS On-Premises Code Sample` 包含使用內部部署 CQRS 的範例。 • `CQRS AWS Serverless` 包含使用 AWS 無伺服器服務的所有 CQRS 和事件來源範例程式碼。	應用程式開發人員

建置 DynamoDB 資料表

任務	Description	所需的技能
提供登入資料。	如果您還沒有存取金鑰，請參閱相關資源一節中的影片。 1. 在 Solution Explorer 中，展開 CQRS AWS Serverless，然後展開建置解決方案資料夾。 2. 展開 AwS.APG.CQRSES.Build 專案並檢視 `Program.cs` 檔案。 3. 捲動至 頂端`Program.cs` 並尋找 `Program()` 。 4. 將 取代`YOUR ACCESS KEY`為您的帳戶存取金鑰，並將 取代`YOUR SECRET KEY`為您的帳戶私密金鑰。請注意，在生產環境中，您不會硬式編碼您的金鑰。反之，您可以使用 AWS Secrets Manager 來存放和擷取登入資料。	應用程式開發人員、資料工程師、DBA
建置專案。	若要建置專案，請開啟 AwS.APG.CQRSES.Build 專案的內容（按一下滑鼠右鍵）選單，然後選擇建置。	應用程式開發人員、資料工程師、DBA
建置和填入資料表。	若要建置資料表並填入種子資料，請開啟 AwS.APG.CQRSES.Build 專案的內容（按一下滑鼠右鍵）選單，然後選擇偵錯、啟動新執行個體。	應用程式開發人員、資料工程師、DBA

任務	Description	所需的技能
驗證資料表建構和資料。	若要驗證，請導覽至 AWS Explorer，然後展開 Amazon DynamoDB。它應該會顯示資料表。開啟每個資料表以顯示範例資料。	應用程式開發人員、資料工程師、DBA

執行本機測試

任務	Description	所需的技能
建置 CQRS 專案。	1. 開啟解決方案，然後導覽至 CQRS AWS Services/CQRS/Tests 解決方案資料夾。 2. 在 AWS.APG.CQRSES.CQRSLambda.Tests 專案中，開啟 BaseFunctionTest.cs，並將 AccessKey 和 SecretKey 取代為您建立的 IAM 金鑰。 3. 儲存變更。 4. 若要編譯和建置測試專案，請開啟專案的內容（按一下滑鼠右鍵）選單，然後選擇建置。	應用程式開發人員、測試工程師
建置事件來源專案。	1. 導覽至 CQRS AWS Services/Event Source/Tests 解決方案資料夾。 2. 在 AWS.APG.CQRSES.EventSourceLambda.Tests 專案中，開啟 BaseFunctionTest.cs，並將	應用程式開發人員、測試工程師

任務	Description	所需的技能
	AccessKey 和 SecretKey 取代為您建立的 IAM 金鑰。 3. 儲存變更。 4. 若要編譯和建置測試專案，請開啟專案的內容（按一下滑鼠右鍵）選單，然後選擇建置。	
執行測試。	若要執行所有測試，請選擇檢視、測試總管，然後選擇在檢視中執行所有測試。所有測試都應通過，以綠色核取記號圖示表示。	應用程式開發人員、測試工程師

將 CQRS Lambda 函數發佈至 AWS

任務	Description	所需的技能
發佈第一個 Lambda 函數。	1. 在 Solution Explorer 中，開啟 AWS.APG.CQRSES.CommandCreateLambda 專案的內容（按一下滑鼠右鍵）選單，然後選擇發佈至 AWS Lambda。 2. 選取您要使用的設定檔，以及您要部署 Lambda 函數的 AWS 區域，以及函數名稱。 3. 對於其餘欄位，保留預設值，然後選擇下一步。 4. 在角色名稱下拉式清單中，選取 AWSLambdaFullAccess。	應用程式開發人員、DevOps 工程師

任務	Description	所需的技能
	5. 若要提供您的帳戶金鑰，請選擇新增，然後輸入 `AcessKey` 作為變數，然後輸入 存取金鑰作為值。然後再次選擇新增，輸入 `SecretKey` 做為變數，輸入您的私密金鑰做為值。 6. 對於其餘欄位，保留預設值，然後選擇上傳。Lambda 測試函數上傳後，會自動出現在 Visual Studio 中。 7. 針對下列專案重複步驟 1-6： • AWS.APG.CQRSES.CommandDeleteLambda • AWS.APG.CQRSES.CommandUpdateLambda • AWS.APG.CQRSES.CommandAddRewardLambda • AWS.APG.CQRSES.CommandRedeemRewardLambda • AWS.APG.CQRSES.QueryCustomerListLambda • AWS.APG.CQRSES.QueryRewqardLambda	

任務	Description	所需的技能
驗證函數上傳。	（選用）您可以透過導覽至 AWS Explorer 並展開 AWS Lambda 來驗證函數是否已成功載入。若要開啟測試視窗，請選擇 Lambda 函數（按兩下）。	應用程式開發人員、DevOps 工程師
測試 Lambda 函數。	1. 在其他資訊區段中輸入請求資料，或從測試資料複製範例請求資料。請確定您為要測試的函數選取資料。 2. 若要執行測試，請選擇叫用。回應和任何錯誤會顯示在回應文字方塊中，而日誌會顯示在日誌文字方塊或 CloudWatch Logs 中。 3. 若要驗證資料，請在 AWS Explorer 中選擇 DynamoDB 資料表（按兩下）。 所有 CQRS Lambda 專案都位於 `CQRS AWS Serverless\CQRS\Command Microservice` 和 `CQRS AWS Serverless\CQRS\Command Microservice` 解決方案資料夾下。如需解決方案目錄和專案，請參閱其他資訊區段中的原始程式碼目錄。	應用程式開發人員、DevOps 工程師

任務	Description	所需的技能
發佈剩餘的 函數。	針對下列專案重複上述步驟： • AWS.APG.CQRSES.CommandDeleteLambda • AWS.APG.CQRSES.CommandUpdateLambda • AWS.APG.CQRSES.CommandAddRewardLambda • AWS.APG.CQRSES.CommandRedeemRewardLambda • AWS.APG.CQRSES.QueryCustomerListLambda • AWS.APG.CQRSES.QueryRewqardLambda	應用程式開發人員、DevOps 工程師

將 Lambda 函數設定為事件接聽程式

任務	Description	所需的技能
發佈 Customer and Reward Lambda 事件處理常式。	若要發佈每個事件處理常式，請遵循上述史詩中的步驟。 專案位於 `CQRS AWS Serverless\Event Source\Customer Event`和 `CQRS AWS Serverless\Event Source\Reward Event` 解決方案資料夾下。如需詳細資訊，請參閱其他資訊區段中的原始程式碼目錄。	應用程式開發人員

任務	Description	所需的技能
連接事件來源 Lambda 事件接聽程式。	1. 使用發佈 Lambda 專案時所使用的相同帳戶登入 AWS 管理主控台。 2. 針對 區域，選取美國東部 1 或您在上一個史詩中部署 Lambda 函數的區域。 3. 導覽至 Lambda 服務。 4. 選取 `EventSourceCustomer` Lambda 函數。 5. 選擇新增觸發。 6. 在觸發組態下拉式清單中，選取 DynamoDB。 7. 在 DynamoDB 資料表下拉式清單中，選取 cqrses-customer-cmd。 8. 在開始位置下拉式清單中，從 選取裁剪地平線。裁剪地平線表示 DynamoDB 觸發條件會在最後一個（未修剪）串流記錄開始讀取，這是碎片中最舊的記錄。 9. 選取啟用觸發核取方塊。 10對於其餘欄位，保留預設值，然後選擇新增。 接聽程式成功連接到 DynamoDB 資料表後，它將顯示在 Lambda 設計工具頁面上。	應用程式開發人員

任務	Description	所需的技能
發佈並連接 EventSourceReward Lambda 函數。	若要發佈並連接 `EventSourceReward` Lambda 函數，請重複前兩個故事中的步驟，從 DynamoDB 資料表下拉式清單中選取 cqrses-reward-cmd。	應用程式開發人員

測試和驗證 DynamoDB 串流和 Lambda 觸發

任務	Description	所需的技能
測試串流和 Lambda 觸發。	1. 在 Visual Studio 中，導覽至 AWS Explorer。 2. 展開 AWS Lambda，然後選擇 CommandRedeemRewardfunction（按兩下）。在開啟的函數視窗中，您可以測試函數。 3. 在請求文字方塊中，以 JavaScript 物件標記法 (JSON) 格式輸入請求資料。如需範例請求，請參閱其他資訊區段中的測試資料。 4. 選擇調用。	應用程式開發人員
驗證，使用 DynamodDB 獎勵查詢表。	1. 開啟 cqrses-reward-query 資料表。 2. 檢查兌換獎勵的客戶點數。兌換的點數應該從客戶的總彙總點數中減去。	應用程式開發人員
使用 CloudWatch Logs 驗證。	1. 導覽至 CloudWatch，然後選擇日誌群組。	應用程式開發人員

任務	Description	所需的技能
	2. /aws/lambda/EventSourceRewardlog 群組包含EventSourceReward 觸發程序的日誌。會記錄所有 Lambda 呼叫，包括您在 context.Logger.LogLine 和 Lambda 程式碼Console.Writeline 中放置的訊息。	
驗證 EventSourceCustomer 觸發程序。	若要驗證EventSourceCustomer 觸發條件，請使用EventSourceCustomer 觸發條件各自的客戶資料表和 CloudWatch 日誌，重複此史詩中的步驟。	應用程式開發人員

相關資源

參考

- Visual Studio 2019 Community Edition 下載
- AWS Toolkit for Visual Studio 下載
- AWS Toolkit for Visual Studio 使用者指南
- AWS 上的無伺服器
- DynamoDB 使用案例和設計模式
- Martin Fowler CQRS
- Martin Fowler 事件來源

影片

- AWS Toolkit for Visual Studio 示範
- 如何為新的 IAM 使用者建立存取金鑰 ID？

其他資訊

CQRS 和事件來源

CQRS

CQRS 模式會將單一概念操作模型，例如資料存取物件單一 CRUD （建立、讀取、更新、刪除） 模型，分成命令和查詢操作模型。命令模型是指變更狀態的任何操作，例如建立、更新或刪除。查詢模型是指傳回值的任何操作。

1. Customer CRUD 模型包含下列界面：
   - `Create Customer()`
   - `UpdateCustomer()`
   - `DeleteCustomer()`
   - `AddPoints()`
   - `RedeemPoints()`
   - `GetVIPCustomers()`
   - `GetCustomerList()`
   - `GetCustomerPoints()`

隨著您的需求變得更加複雜，您可以從此單一模型方法中移動。CQRS 使用命令模型和查詢模型來區隔寫入和讀取資料的責任。如此一來，資料就可以獨立維護和管理。透過明確的責任分離，每個模型的增強功能不會影響另一個模型。此區隔可改善維護和效能，並降低應用程式的複雜性。

1. Customer Command 模型中的界面：
   - `Create Customer()`
   - `UpdateCustomer()`
   - `DeleteCustomer()`
   - `AddPoints()`
   - `RedeemPoints()`
2. 客戶查詢模型中的界面：
   - `GetVIPCustomers()`

- `GetCustomerList()`
- `GetCustomerPoints()`
- `GetMonthlyStatement()`

如需範例程式碼，請參閱原始程式碼目錄。

然後，CQRS 模式會解耦資料庫。這種解耦會導致每個服務的整體獨立性，這是微服務架構的主要組成部分。

在 AWS 雲端中使用 CQRS，您可以進一步最佳化每個服務。例如，您可以設定不同的運算設定，或在無伺服器或容器型微服務之間進行選擇。您可以使用 Amazon ElastiCache 取代內部部署快取。如果您有內部部署發佈/訂閱訊息，您可以將其取代為 Amazon Simple Notification Service (Amazon SNS)。此外，您可以利用pay-as-you-go定價和各種 AWS 服務，這些服務僅針對您使用的項目付費。

CQRS 包含下列優點：

- 獨立擴展 – 每個模型都可以調整其擴展策略，以滿足服務的需求。與高效能應用程式類似，分開讀取和寫入可讓模型獨立擴展，以滿足每個需求。您也可以新增或減少運算資源，以滿足某個模型的可擴展性需求，而不會影響另一個模型。
- 獨立維護 – 查詢和命令模型的分離可改善模型的可維護性。您可以對一個模型進行程式碼變更和增強功能，而不會影響另一個模型。
- 安全 – 將許可和政策套用至個別模型以進行讀取和寫入更為容易。
- 最佳化讀取 – 您可以定義針對查詢最佳化的結構描述。例如，您可以為彙總資料定義結構描述，並為事實資料表定義單獨的結構描述。
- 整合 – CQRS 非常適合事件型程式設計模型。
- 受管複雜性 – 對查詢和命令模型的分離適用於複雜的網域。

使用 CQRS 時，請記住下列注意事項：

- CQRS 模式僅適用於應用程式的特定部分，而非整個應用程式。如果實作在不符合 模式的網域上，它可以降低生產力、增加風險並引入複雜性。
- 此模式最適合具有不平衡讀取和寫入操作的常用模型。
- 對於大量讀取的應用程式，例如需要時間處理的大型報告，CQRS 可讓您選擇正確的資料庫並建立結構描述來存放彙總資料。這可透過僅處理一次報告資料並將其傾印在彙總資料表中，來改善讀取和檢視報告的回應時間。

- 對於寫入密集型應用程式，您可以設定用於寫入操作的資料庫，並允許命令微服務在寫入需求增加時獨立擴展。如需範例，請參閱 AWS.APG.CQRSES.CommandRedeemRewardLambda和 AWS.APG.CQRSES.CommandAddRewardLambda 微服務。

事件來源

下一個步驟是使用事件來源，在執行命令時同步查詢資料庫。例如，請考慮下列事件：

- 新增的客戶獎勵點需要更新查詢資料庫中的客戶總獎勵點或彙總獎勵點。
- 命令資料庫中會更新客戶的姓氏，這需要更新查詢資料庫中的代理客戶資訊。

在傳統 CRUD 模型中，您可以鎖定資料直到完成交易，以確保資料的一致性。在事件來源中，資料會透過發佈一系列事件來同步，訂閱者將使用該事件來更新其個別資料。

事件來源模式可確保並記錄對資料採取的完整一系列動作，並透過一系列事件發佈。這些事件代表一組對資料所做的變更，該事件的訂閱者必須處理這些變更，才能讓記錄保持最新狀態。訂閱者會使用這些事件，同步訂閱者資料庫中的資料。在這種情況下，這是查詢資料庫。

下圖顯示與 AWS 上的 CQRS 搭配使用的事件來源。

1. 命令 Lambda 函數會在資料庫上執行寫入操作，例如建立、更新或刪除。
2. 查詢 Lambda 函數會在資料庫上執行讀取操作，例如取得或選取。
3. 此 Lambda 函數會從命令資料庫處理 DynamoDB 串流，並更新查詢資料庫以進行變更。您也可以使用此函數來發佈訊息至 Amazon SNS，以便其訂閱者可以處理資料。
4. （選用） Lambda 事件訂閱者會處理 Amazon SNS 發佈的訊息，並更新查詢資料庫。
5. （選用） Amazon SNS 會傳送寫入操作的電子郵件通知。

在 AWS 上，DynamoDB Streams 可以同步查詢資料庫。DynamoDB 會以近乎即時的方式擷取 DynamobDB 資料表中項目層級修改的時間順序，並在 24 小時內持久儲存資訊。

啟用 DynamoDB Streams 可讓資料庫發佈一系列事件，使事件來源模式成為可能。事件來源模式會新增事件訂閱者。事件訂閱者應用程式會使用事件，並根據訂閱者的責任來處理事件。在上圖中，事件訂閱者會將變更推送至 Query DynamoDB 資料庫，以保持資料同步。使用 Amazon SNS、訊息中介裝置和事件訂閱者應用程式會保持架構解耦。

事件來源包含下列優點：

- 交易資料的一致性
- 可靠的稽核線索和動作歷史記錄，可用於監控資料中採取的動作
- 允許微型服務等分散式應用程式跨環境同步其資料
- 每當狀態變更時，可靠發佈事件
- 重建或重播過去狀態
- 鬆散耦合的實體，交換事件以從單體應用程式遷移到微服務
- 減少並行更新所造成的衝突；事件來源可避免直接在資料存放區中更新物件的需求
- 解耦任務和事件的彈性和可擴展性
- 外部系統更新
- 在單一事件中管理多個任務

使用事件來源時，請記住下列注意事項：

- 由於來源訂閱者資料庫之間的資料更新有所延遲，復原變更的唯一方法是將補償事件新增至事件存放區。
- 實作事件來源具有自其程式設計風格不同的學習曲線。

測試資料

成功部署後，請使用下列測試資料來測試 Lambda 函數。

CommandCreate 客戶

```
{  "Id":1501,  "Firstname":"John",  "Lastname":"Done",  "CompanyName":"AnyCompany",
 "Address": "USA",  "VIP":true }
```

CommandUpdate 客戶

```
{  "Id":1501,  "Firstname":"John",  "Lastname":"Doe",  "CompanyName":"Example Corp.",
 "Address": "Seattle, USA",  "VIP":true }
```

CommandDelete 客戶

輸入客戶 ID 做為請求資料。例如，如果客戶 ID 為 151，請輸入 151 做為請求資料。

```
151
```

QueryCustomerList

這是空白的。調用時，它會傳回所有客戶。

CommandAddReward

這會新增 40 點給 ID 為 1 的客戶 (Richard)。

```
{
  "Id":10101,
  "CustomerId":1,
  "Points":40
}
```

CommandRedeemReward

這會扣除 ID 為 1 (Richard) 的客戶 15 點。

```
{
  "Id":10110,
  "CustomerId":1,
  "Points":15
}
```

QueryReward

輸入客戶的 ID。例如，輸入 1 表示 Richard，2 表示 Arnav，3 表示 Shirley。

```
2
```

原始程式碼目錄

使用下表做為 Visual Studio 解決方案目錄結構的指南。

CQRS 現場部署程式碼範例解決方案目錄

客戶 CRUD 模型

CQRS 現場部署程式碼範例\CRUD Model\AWS.APG.CQRSES.DAL 專案

Customer CRUD 模型的 CQRS 版本

- 客戶命令：CQRS On-Premises Code Sample\CQRS Model\Command Microservice \AWS.APG.CQRSES.Command專案
- 客戶查詢：CQRS On-Premises Code Sample\CQRS Model\Query Microservice \AWS.APG.CQRSES.Query專案

命令和查詢微服務

命令微服務位於解決方案資料夾 下CQRS On-Premises Code Sample\CQRS Model\Command Microservice：

- AWS.APG.CQRSES.CommandMicroservice ASP.NET Core API 專案可做為消費者與服務互動的進入點。
- AWS.APG.CQRSES.Command .NET Core 專案是託管命令相關物件和界面的物件。

查詢微服務位於解決方案資料夾 下CQRS On-Premises Code Sample\CQRS Model\Query Microservice：

- AWS.APG.CQRSES.QueryMicroservice ASP.NET Core API 專案做為消費者與服務互動的進入點。
- AWS.APG.CQRSES.Query .NET Core 專案是託管查詢相關物件和界面的物件。

CQRS AWS Serverless 程式碼解決方案目錄

此程式碼是使用 AWS 無伺服器服務的現場部署程式碼的 AWS 版本。

在 C# .NET Core 中，每個 Lambda 函數都由一個 .NET Core 專案表示。在此模式的範例程式碼中，命令和查詢模型中的每個界面都有單獨的專案。

使用 AWS 服務的 CQRS

您可以在 CQRS AWS Serverless\CQRS資料夾中找到使用 AWS 無伺服器服務的 CQRS 根解決方案目錄。此範例包含兩種模型：客戶和獎勵。

Customer and Reward 的命令 Lambda 函數位於 CQRS\Command Microservice\Customer和 CQRS\Command Microservice\Reward 資料夾下。它們包含下列 Lambda 專案：

- 客戶命令：CommandDeleteLambda、 CommandCreateLambda和 CommandUpdateLambda

- 獎勵命令： CommandAddRewardLambda 和 CommandRedeemRewardLambda

客戶和獎勵的查詢 Lambda 函數位於 CQRS\Query Microservice\Customer和 CQRS \QueryMicroservice\Reward資料夾下。它們包含 QueryCustomerListLambda和 QueryRewardLambda Lambda 專案。

CQRS 測試專案

測試專案位於 CQRS\Tests 資料夾下。此專案包含測試指令碼，可自動測試 CQRS Lambda 函數。

使用 AWS 服務的事件來源

客戶和獎勵 DynamoDB 串流會啟動下列 Lambda 事件處理常式，以處理和同步查詢資料表中的資料。

- EventSourceCustomer Lambda 函數會映射至客戶資料表 (cqrses-customer-cmd) DynamoDB 串流。
- EventSourceReward Lambda 函數會映射至獎勵資料表 (cqrses-reward-cmd) DynamoDB 串流。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

更多模式

- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon EKS 上私下存取容器應用程式
- 使用 AWS Systems Manager 自動化新增或更新 Windows 登錄項目
- 使用 DR Orchestrator Framework 自動化跨區域容錯移轉和容錯回復
- 使用 CI/CD 管道自動建置 Java 應用程式並將其部署到 Amazon EKS
- 使用 AWS CDK 自動為微服務建置 CI/CD 管道和 Amazon ECS 叢集
- 使用 BMC AMI Cloud Data 將大型主機資料備份並封存至 Amazon S3
- 使用 Amazon EC2 Auto Scaling 和 Systems Manager 建置 Micro Focus Enterprise Server PAC
- 使用 Amazon DataZone 建置企業資料網格 AWS CDK，以及 AWS CloudFormation
- 容器化已由 Blu Age 現代化的大型主機工作負載
- 使用 Python 將 EBCDIC 資料轉換為 AWS 上的 ASCII
- 使用 Micro Focus 轉換具有複雜記錄配置的大型主機資料檔案
- 使用、 AWS Amplify Angular 和 Module Federation 建立微型前端的入口網站
- 使用 Elastic Beanstalk 部署容器
- 使用 PostgreSQL 相容 Aurora 全域資料庫模擬 Oracle DR
- 在 Quick Sight 中使用 AWS Mainframe Modernization 和 Amazon Q 產生資料洞見
- 使用 Quick Sight 中的 AWS Mainframe Modernization 和 Amazon Q 產生 Db2 z/OS 資料洞見
- 遷移至 Amazon ECR 儲存庫時，自動識別重複的容器映像
- 使用 K8sGPT 和 Amazon Bedrock 整合實作採用 AI 技術的 Kubernetes 診斷和故障診斷
- 在 AWS Blu Age 現代化大型主機應用程式中實作 Microsoft Entra ID 型身分驗證
- 在 Amazon API Gateway 中使用自訂網域實作路徑型 API 版本控制
- 使用 Oracle SQL Developer 和 AWS SCT，逐步從 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL
- 將stonebranch 通用控制器與 AWS Mainframe Modernization 整合
- 在多個 AWS 帳戶和 AWS 區域中管理 AWS Service Catalog 產品
- 將 AWS 成員帳戶從 遷移 AWS Organizations 至 AWS Control Tower
- 使用來自 Precisely 的 Connect 將 VSAM 檔案遷移和複寫至 Amazon RDS 或 Amazon MSK
- 使用 AWS DMS 從 SAP ASE 遷移至 Amazon RDS for SQL Server
- 將 Oracle 外部資料表遷移至 Amazon Aurora PostgreSQL 相容
- 使用 現代化 CardDemo 大型主機應用程式 AWS Transform

- 使用 和 Terraform 現代化 AWS Transform 和部署大型主機應用程式
- 使用 Rocket Enterprise Server 和 LRS VPSX/MFI AWS 在 上現代化大型主機批次列印工作負載
- 使用 Micro Focus Enterprise Server 和 LRS VPSX/MFI 將 AWS 上的大型主機線上列印工作負載現代化
- AWS 使用 Rocket Enterprise Server 和 LRS PageCenterX 在 上現代化大型主機輸出管理
- 使用 Transfer 系列將大型主機檔案直接移至 Amazon S3
- 使用 AWS CDK 和 GitHub Actions 工作流程最佳化多帳戶無伺服器部署
- 最佳化 AWS Blu Age 現代化應用程式的效能
- 使用 IaC 原則自動化 Amazon Aurora 全域資料庫的藍/綠部署
- 使用 Precisely Connect 將大型主機資料庫複寫至 AWS
- 使用 Amazon ECS Anywhere 在 Amazon WorkSpaces 上執行 Amazon ECS 任務 Amazon ECS Anywhere
- 將遙測資料從 AWS Lambda 傳送至 OpenSearch，以進行即時分析和視覺化
- 在多區域、多帳戶組織中設定 CloudFormation 偏離偵測
- 使用 AWS Lambda 在六邊形架構中建構 Python 專案
- 使用 LocalStack 和 Terraform Tests 測試 AWS 基礎設施
- 從 ENSA1 升級 SAP Pacemaker 叢集到 ENSA2
- 使用 Amazon Q Developer 作為編碼助理，以提高您的生產力
- 在本機驗證帳戶工廠的 Terraform (AFT) 程式碼

大型主機

主題

- 安裝 以 AWS 服務 從 IBM z/OS 存取 AWS CLI
- 使用 BMC AMI Cloud Data 將大型主機資料備份並封存至 Amazon S3
- 使用 AWS Mainframe Modernization 和 建置 COBOL Db2 程式 AWS CodeBuild
- 使用 Amazon EC2 Auto Scaling 和 Systems Manager 建置 Micro Focus Enterprise Server PAC
- 在 AWS 雲端中建置進階大型主機檔案檢視器
- 容器化已由 Blu Age 現代化的大型主機工作負載
- 使用 Python 將 EBCDIC 資料轉換為 AWS 上的 ASCII
- 使用 Amazon S3 將大型主機檔案從 EBCDIC 格式轉換為字元分隔 ASCII 格式 AWS Lambda
- 使用 Micro Focus 轉換具有複雜記錄配置的大型主機資料檔案
- 使用 Terraform 部署容器化 Blu Age 應用程式的環境
- 使用 Quick Sight 中的 AWS Mainframe Modernization 和 Amazon Q 產生 Db2 z/OS 資料洞見
- 在 Quick Sight 中使用 AWS Mainframe Modernization 和 Amazon Q 產生資料洞見
- 在 AWS Blu Age 現代化大型主機應用程式中實作 Microsoft Entra ID 型身分驗證
- 將stonebranch 通用控制器與 AWS Mainframe Modernization 整合
- 使用來自 Precisely 的 Connect 將 VSAM 檔案遷移和複寫至 Amazon RDS 或 Amazon MSK
- 使用 現代化 CardDemo 大型主機應用程式 AWS Transform
- 使用 和 Terraform 現代化 AWS Transform 和部署大型主機應用程式
- AWS 使用 Rocket Enterprise Server 和 LRS PageCenterX 在 上現代化大型主機輸出管理
- 使用 Rocket Enterprise Server 和 LRS VPSX/MFI AWS 在 上現代化大型主機批次列印工作負載
- 大型主機現代化：DevOps on AWS with Rocket Software Enterprise Suite
- 使用 Micro Focus Enterprise Server 和 LRS VPSX/MFI 將 AWS 上的大型主機線上列印工作負載現代化
- 使用 Transfer 系列將大型主機檔案直接移至 Amazon S3
- 最佳化 AWS Blu Age 現代化應用程式的效能
- 使用信任的內容來保護和簡化 AWS 上 Db2 聯合資料庫中的使用者存取
- 以 CSV 檔案將大規模 Db2 z/OS 資料傳輸至 Amazon S3
- 更多模式

安裝 以 AWS 服務 從 IBM z/OS 存取 AWS CLI

Souma Ghosh、Paulo Vitor Pereira 和 Phil de Valence，Amazon Web Services

總結

AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可透過在命令列 Shell 中使用命令來管理多個 AWS 服務。透過最少的組態，您可以從命令列工作階段執行命令，例如命令提示字元、終端機和 bash shell，以實作相當於瀏覽器型提供的功能 AWS 管理主控台。

API 和 AWS 管理主控台 提供 中所有 AWS 基礎設施即服務 (IaaS) 管理、管理和存取函數 AWS AWS CLI。您可以在 IBM z/OS 大型主機 AWS CLI 上安裝 ，以 AWS 服務 從 z/OS 直接存取、管理和與 互動。 AWS CLI 可讓使用者和應用程式執行各種任務，例如：

- 在 z/OS 和 Amazon Simple Storage Service (Amazon S3) 物件儲存體之間傳輸檔案或資料集，並檢視儲存貯體的內容
- 啟動和停止不同的 AWS 資源；例如，在 AWS Mainframe Modernization 環境中啟動批次任務
- 呼叫 AWS Lambda 函數以實作常見的商業邏輯
- 與人工智慧和機器學習 (AI/ML) 和分析服務整合

此模式說明如何在 z/OS AWS CLI 上安裝、設定和使用 。您可以全域安裝，以便所有 z/OS 使用者或在使用者層級使用。模式也詳細說明如何在來自 z/OS Unix System Services (USS) 的互動式命令列工作階段 AWS CLI 中使用 ，或用作批次任務。

先決條件和限制

先決條件

- 從 z/OS 到 的網路通訊 AWS

  根據預設， AWS 服務 會在 TCP 連接埠 443 上使用 HTTPS 將請求 AWS CLI 傳送至 。若要 AWS CLI 成功使用 ，您必須能夠在 TCP 連接埠 443 上進行傳出連線。您可以使用下列任何 z/OS USS 命令 （其中一些可能未安裝在您的環境中） 來測試從 z/OS 到 的網路連線 AWS：

```
ping amazonaws.com
dig amazonaws.com
traceroute amazonaws.com
curl -k https://docs.aws.amazon.com/cli/v1/userguide/cli-chap-welcome.html
```

- AWS 登入資料

為了與來自 z/OS AWS 雲端 的服務通訊， AWS CLI 要求您設定一些具有存取目標權限的登入資料 AWS 帳戶。對於 的程式設計命令 AWS，您可以使用存取金鑰，其中包含存取金鑰 ID 和私密存取金鑰。如果您沒有存取金鑰，可以從 AWS 管理主控台建立。最佳實務是，除非 AWS 帳戶 需要根使用者，否則請勿將根使用者的存取金鑰用於任何任務。反之，請建立新的管理員 IAM 使用者，並準備最低權限的許可 ，以使用存取金鑰設定使用者。建立使用者之後，您可以為此使用者建立存取金鑰 ID 和私密存取金鑰。

> ⚠ Warning
>
> AWS Identity and Access Management (IAM) 使用者具有存在安全風險的長期登入資料。為了協助降低此風險，建議您只為這些使用者提供執行任務所需的許可，並在不再需要這些使用者時將其移除。

- 適用於 z/OS 的 IBM Python

  AWS CLI 需要 Python 3.8 或更新版本。IBM 已使用適用於 z/OS 的 IBM Open Enterprise Python 在 z/OS 上執行 Python。IBM Open Enterprise Python 可透過 Shopz SMP/E 免費取得，您也可以從 IBM 網站下載 PAX 檔案。如需說明，請參閱適用於 z/OS 的 IBM Open Enterprise Python 安裝和組態文件。

限制

- 此模式中提供的安裝指示AWS CLI 僅適用於版本 1。的最新版本 AWS CLI 是第 2 版。不過，此模式使用較舊的版本，因為第 2 版的安裝方法不同，而第 2 版可用的二進位可執行檔與 z/OS 系統不相容。

產品版本

- AWS CLI 第 1 版
- Python 3.8 或更高版本

Architecture

技術堆疊

- 執行 z/OS 的大型主機

- 大型主機 z/OS UNIX 系統服務 (USS)
- Mainframe Open MVS (OMVS) – z/OS UNIX shell 環境命令界面
- 大型主機磁碟，例如直接存取儲存裝置 (DASD)
- AWS CLI

目標架構

下圖顯示 IBM z/OS 上的 AWS CLI 部署。您可以從 AWS CLI 互動式使用者工作階段叫用 ，例如 SSH 和 telnet 工作階段。您也可以使用任務控制語言 (JCL) 或任何可呼叫 z/OS Unix shell 命令的程式，從批次任務叫用它。

會透過 TCP/IP 網路與 AWS 服務 端點 AWS CLI 通訊。這種網路連線可以透過網際網路，或透過從客戶資料中心到 AWS 雲端 資料中心的私有 AWS Direct Connect 連線進行。通訊會使用 AWS 登入資料進行身分驗證並加密。

自動化和擴展

您可以使用 探索 的功能， AWS 服務 AWS CLI 並開發 USS shell 指令碼以從 z/OS 管理您的 AWS 資源。您也可以從 z/OS 批次環境執行 AWS CLI 命令和 shell 指令碼，也可以透過與大型主機排程器整合來自動化批次任務，以在特定排程執行。 AWS CLI 命令或指令碼可以在參數 (PARMs) 和程序 (PROCs) 內編碼，並且可以透過使用不同參數從不同批次任務呼叫 PARM 或 PROC 的標準方法進行擴展。

工具

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。

最佳實務

- 基於安全考量，請將存取許可限制在存放 AWS 存取金鑰詳細資訊的 USS 目錄。僅允許存取使用 的使用者或程式 AWS CLI。
- 請勿將 AWS 帳戶 根使用者存取金鑰用於任何任務。反之，請為自己建立新的管理員 IAM 使用者，並使用存取金鑰進行設定。

> **Warning**
>
> IAM 使用者具有存在安全風險的長期登入資料。為了協助降低此風險，建議您只為這些使用者提供執行任務所需的許可，並在不再需要這些使用者時將其移除。

史詩

在 z/OS USS 上安裝 AWS CLI 版本 1

任務	Description	所需的技能
安裝 Python 3.8 或更新版本。	1. 使用下列其中一種方法登入 z/OS USS 命令提示字元界面： • 從互動式系統生產力設施 (ISPF) 面板使用時間共用選項 (TSO) OMVS 命令，或 https://www.ibm.com/docs/en/zos-basic-skills?topic=interfaces-what-is-ispf • 使用 SSH 或 telnet 連線到大型主機邏輯分割區 (LPAR) 的 IP。 此模式假設 `cliuser` 是用來登入 USS 環境的 userid，`/u/cliuser/` 是使用者的主目錄。您可以根據您的安裝需求，在 z/OS 環境中以不同的方式設定使用者主目錄。 2. 如果尚未安裝 Python 3.8 或更新版本，請遵循適用於 z/	大型主機 z/OS 管理員

任務	Description	所需的技能
	OS 的 IBM Open Enterprise Python 安裝指南。	

任務	Description	所需的技能

設定 USS 環境變數。	將環境變數新增至設定檔。您可以將這些項目新增至個別使用者 (cliuser) 的 /u/cliuser/.profile 檔案，或新增至所有使用者的 /etc/profile 檔案。
	
	Note
	此模式假設 Python 已安裝在 /u/awscli/python目錄中。如果您的安裝目錄不同，請相應地更新程式碼。
	
	`# Python configuration`
	`export BPXKAUTOC`
	`VT='ON'`
	`export CEERUNOPT`
	`S='FILETAG(AUTOCVT`
	`,AUTOTAG) POSIX(ON)'`
	`export TAGREDIR_ERR=txt`
	`export TAGREDIR_IN=txt`
	`export TAGREDIR_`
	`OUT=txt`
	
	`# AWS CLI configura`
	`tion`
	`export PATH=/u/c`
	`liuser/python/bin:`
	`$PATH`
	`export PYTHONPATH=/u/`
	`cliuser/python:$PYTHON`
	`PATH`
		大型主機 z/OS 管理員

任務	Description	所需的技能
測試 Python 安裝。	執行 python 命令： `python --version` 輸出應確認您已正確安裝 Python 3.8 或更新版本。	大型主機 z/OS 管理員

驗證或安裝 pip。	1. 當您從 IBM 網站安裝 Python 時，通常會自動安裝 pip 命令。若要驗證，請執行 命令：
	`pip —-version`
	如果已安裝 pip，此命令應會顯示已安裝的版本。
	2. 如果找不到 pip 命令，請執行下列命令來安裝 pip：
	`python -m ensurepip --upgrade`
	如需更多安裝選項，請參閱 pip 文件。
		大型主機 z/OS 管理員

任務	Description	所需的技能

安裝 AWS CLI 版本 1。	1. 若要安裝 AWS CLI，請執行命令：
	`python -m pip install awscli`
	輸出格式應類似以下內容。
	`Successfully installed PyYAML-6.0.1 awscli-1.32.23 botocore-1.34.23 colorama-0.4.4 docutils-0.16 jmespath-1.0.1 pyasn1-0.5.1 python-dateutil-2.8.2 rsa-4.7.2 s3transfer-0.10.0 urllib3-2.0.7`
	2. 執行下列命令來變更 aws 可執行檔的許可。請務必使用 `<python_installation_dir>` Python 安裝路徑更新預留位置目錄。
	`chmod 744 <python_installation_dir>/bin/aws`
	3. 執行下列命令來測試 AWS CLI 安裝：
	`aws --version`
		大型主機 z/OS 管理員

任務	Description	所需的技能
	輸出應會顯示 AWS CLI、Python 和 botocore 的版本，如下所示。 `aws-cli/1.32.3 Python/3.9.5 OS/390/27.00 botocore/1.34.3`	

從 z/OS 設定 AWS CLI 存取權

任務	Description	所需的技能

設定 AWS 存取金鑰、預設區域和輸出。	AWS CLI 文件說明設定 AWS 存取權的不同選項。您可以根據組織的標準來選擇組態。此範例使用短期憑證組態。
	
	1. 使用下列命令設定 AWS CLI：
	
	`aws configure`
	
	2. 出現提示時，提供下列項目的詳細資訊。存取金鑰 ID 和私密存取金鑰值來自您在先決條件步驟中設定 AWS 登入資料時取得的金鑰。
	
	`AWS Access Key ID [None]: ASIAIOSFO DNN7EXAMPLE AWS Secret Access Key [None]: wJalrXUtn`
		AWS 管理員、大型主機 z/OS 管理員、大型主機 z/OS 開發人員

任務	Description	所需的技能
	FEMI/K7MDENG/bPxRf iCYEXAMPLEKEY Default region name [None]: us-east-1 Default output format [None]: aws configure set aws_session_token IQoJb3JpZ2luX2IQoJ b3JpZ2luX2IQoJb3Jp Z2luX2IQoJb3JpZ2lu X2IQoJb3JpZVERYLON GSTRINGEXAMPLE 此組態，包括存取金鑰，會存放在 /u/cliuser/.aws 資料夾中。基於安全考量，請將此資料夾限制為僅允許使用 的使用者或程式存取 AWS CLI。	

任務	Description	所需的技能
測試 AWS CLI。	1. 在命令提示字元中執行下列命令，以 AWS CLI 使用簡單的命令測試 ： `aws s3 ls` 輸出應列出已設定 的所有 S3 儲存貯體， AWS 帳戶沒有任何錯誤。 2. 遵循接下來兩個 epics 中的指示，將資料從 USS 傳輸到 Amazon S3。您可以選擇以下兩個選項之一： • 選項 1 （下一個圖示）：以互動方式將 EBCDIC 逗號分隔值 (CSV) 檔案傳輸至 Amazon S3，並從 Amazon Athena 查詢檔案。 • 選項 2：將 EBCDIC 固定長度資料集作為批次任務傳輸至 Amazon S3。	大型主機 z/OS 管理員、大型主機 z/OS 開發人員

選項 1 – 從 USS 工作階段以互動方式將資料從 USS 傳輸到 Amazon S3

任務	Description	所需的技能
下載並傳輸範例 CSV 檔案。	1. `sales-records.csv` 從附件區段下載。此檔案提供銷售記錄的範例 CSV 檔案。 2. 將檔案傳輸至 z/OS USS。	應用程式開發人員、大型主機 z/OS 開發人員

任務	Description	所需的技能
	3. 使用您選擇的文字編輯器，確認`/u/cliuser/sales-records.csv` 檔案可在 USS 中以 EBCDIC 格式讀取。	

任務	Description	所需的技能

建立 S3 儲存貯體並上傳 CSV 檔案。	1. 建立 S3 儲存貯體以存放 CSV 檔案。
	`aws s3 mb s3://<s3_bucket_name>`
	其中 <s3_bucket_name> 是儲存貯體的唯一名稱；例如：
	`aws s3 mb s3://DOC-EXAMPLE-BUCKET1`
	2. 將 CSV 檔案從 z/OS USS 上傳至 S3 儲存貯體：
	`aws s3 cp <csv_file_path> s3://<s3_bucket_name>`
	例如：
	`aws s3 cp /u/cliuser/sales-records.csv s3://DOC-EXAMPLE-BUCKET1`
	3. 列出 S3 儲存貯體的內容，並確認其中包含上傳的檔案：
	`aws s3 ls s3://<s3_bucket_name>`
	例如：
		應用程式開發人員、大型主機 z/OS 開發人員

任務	Description	所需的技能
	`aws s3 ls s3://DOC-EXAMPLE-BUCKET1`	
檢視 S3 儲存貯體和上傳的檔案。	1. 登入 AWS 管理主控台 並開啟 Amazon S3 主控台。 2. 導覽以查看新的 S3 儲存貯體和上傳的物件。 如需上傳物件的詳細資訊，請參閱 Amazon S3 文件中的 Amazon S3 入門。	一般 AWS

任務	Description	所需的技能

在 Amazon Athena 資料表上執行 SQL 查詢。	1. 開啟 Amazon Athena 主控台。
	2. 使用 Amazon S3 的 CSV 資料建立新的資料表（例如 DOC-EXAMPLE-BUCKET）。如需詳細資訊，請參閱 Amazon S3 文件中的使用 Amazon Athena 查詢 Amazon S3 庫存。 Amazon S3
	3. 針對資料表執行SELECT查詢以檢視資料。
	`SELECT * FROM <table_name>;`
	例如：
	`SELECT * FROM DOC-EXAMPLE-BUCKET;`
	SQL 查詢的輸出會顯示 CSV 檔案的內容。
		General AWS，應用程式開發人員

選項 2 – 使用批次 JCL 將資料從 USS 傳輸到 Amazon S3

任務	Description	所需的技能
上傳範例檔案。	1. `sales-records-fixed.txt` 從附件區段下載。這是包含銷售記錄的範例檔案。將文字檔案重新命	大型主機 z/OS 開發人員

任務	Description	所需的技能
	名為 ，例如 USER.DATA .FIXED 。 2. 將檔案以固定封鎖 (FB)、256 記錄長度 (LRECL)、實體序列 (PS) 資料集的形式傳輸至 z/OS。 3. 在 ISPF 選項 3.4 下，使用資料集清單公用程式來驗證USER.DATA.FIXED 資料集是否可讀取為 EBCDIC 格式。如需輸出範例，請參閱其他資訊一節。	

任務	Description	所需的技能

建立批次 JCL。	將批次 JCL 編碼如下，以建立目的地 S3 儲存貯體、上傳資料集，並列出儲存貯體內容。請務必將目錄名稱、檔案名稱和儲存貯體名稱取代為您自己的值。
	
	`//AWSCLICP JOB`
	`ACTINFO1,'IBMUSER'`
	`,CLASS=A,MSGCLASS=`
	`H,MSGLEVEL=(1,1),`
	`// NOTIFY=&SYSUID,TIM`
	`E=1440`
	`//*--------------`
	`------------------`
	`------------------`
	`-------`
	`//* Sample job for AWS`
	`CLI`
	`//*--------------`
	`------------------`
	`------------------`
	`-------`
	`//USSCMD EXEC PGM=BPXBA`
	`TCH`
	`//STDERR DD SYSOUT=*`
	`//STDOUT DD SYSOUT=*`
	`//STDENV DD *`
	`export PATH=/u/c`
	`liuser/python/bin:`
	`$PATH`
	`//STDPARM DD *`
	`SH`
	`export _BPXK_AUT`
	`OCVT=ON;`
	`aws s3 mb s3://DOC-`
	`EXAMPLE-BUCKET2;`
	`cp "//'USER.DATA.FIXE`
	`D'" /tmp/tmpfile;`
		大型主機 z/OS 開發人員

任務	Description	所需的技能

	<pre>aws s3 cp /tmp/tmpfile s3://DOC-EXAMPLE-BUCKET2/USER.DATA.FIXED;
	rm /tmp/tmpfile;
	aws s3 ls s3://DOC-EXAMPLE-BUCKET2;
	/*</pre>	

提交批次 JCL 任務。	1. 提交您在上一個步驟中編碼的 JCL 任務。 2. 在系統顯示和搜尋設施 (SDSF) 中檢查任務的狀態。如果成功，任務應以傳回碼 0 結尾。 3. 任務日誌中的標準輸出 (STDOUT) 會顯示儲存貯體建立成功、資料集上傳和儲存貯體內容清單。如需範例畫面圖例，請參閱其他資訊一節。	大型主機 z/OS 開發人員
檢視上傳至 S3 儲存貯體的資料集。	1. 登入 AWS 管理主控台 並開啟 Amazon S3 主控台。 2. 導覽以查看測試儲存貯體中上傳的檔案。 3. 您可以使用 Amazon Redshift 等分析服務，進一步處理或分析`USER.DATA.FIXED` 檔案。	一般 AWS

相關資源

- AWS CLI 第 1 版文件
- AWS Mainframe Modernization CLI 命令參考

- AWS Mainframe Modernization

其他資訊

ISPF 選項 3.4 中的 USER.DATA.FIXED （資料集清單公用程式）

已提交批次任務的 SYSOUT

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 BMC AMI Cloud Data 將大型主機資料備份並封存至 Amazon S3

Santosh Kumar Singh、Gilberto Biondo 和 Maggie Li，Amazon Web Services

Mikhael Liberman，Model9 大型主機軟體

總結

此模式示範如何將大型主機資料直接備份和封存至 Amazon Simple Storage Service (Amazon S3)，然後使用 BMC AMI Cloud Data （先前稱為 Model9 Manager) 將資料回收並還原至大型主機。如果您正在尋找一種方法將備份和封存解決方案現代化為大型主機現代化專案的一部分，或滿足合規要求，此模式有助於實現這些目標。

一般而言，在大型主機上執行核心業務應用程式的組織會使用虛擬磁帶庫 (VTL) 來備份資料存放區，例如檔案和日誌。此方法可能很昂貴，因為它會耗用計費 MIPS，而且無法存取存放在大型主機外部磁帶上的資料。若要避免這些問題，您可以使用 BMC AMI Cloud Data，以快速且經濟實惠的方式將操作和歷史大型主機資料直接傳輸至 Amazon S3。您可以使用 BMC AMI Cloud Data 透過 TCP/IP 備份和封存資料至 ， AWS 同時利用 IBM z 整合式資訊處理器 (zIIP) 引擎來降低成本、平行處理和傳輸時間。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 具有有效授權金鑰的 BMC AMI 雲端資料
- 大型主機與 AWS 之間的 TCP/IP 連線
- 用於讀取/寫入存取 S3 儲存貯體的 AWS Identity and Access Management (IAM) 角色
- 具備大型主機安全產品 (RACF) 存取權以執行 BMC AMI Cloud 程序
- BMC AMI Cloud z/OS 代理程式 (Java 版本 8 64 位元 SR5 FP16 或更新版本），具有可用的網路連接埠、允許存取 S3 儲存貯體的防火牆規則，以及專用 z/FS 檔案系統
- BMC AMI 雲端管理伺服器滿足的需求

限制

- BMC AMI Cloud Data 將其操作資料存放在 PostgreSQL 資料庫中，該資料庫中以 Docker 容器的形式在與管理伺服器相同的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上執

行。Amazon Relational Database Service (Amazon RDS) 目前不支援做為 BMC AMI Cloud Data 的後端。如需最新產品更新的詳細資訊，請參閱《BMC 文件》中的最新消息？。

- 此模式只會備份和封存 z/OS 大型主機資料。BMC AMI Cloud Data 只會備份和封存大型主機檔案。
- 此模式不會將資料轉換為標準開放格式，例如 JSON 或 CSV。使用額外的轉換服務，例如 BMC AMI Cloud Analytics （先前稱為 Model9 Gravity)，將資料轉換為標準開放格式。雲端原生應用程式和資料分析工具可以在資料寫入雲端後存取資料。

產品版本

- BMC AMI Cloud Data 2.x 版

Architecture

來源技術堆疊

- 執行 z/OS 的大型主機
- 大型主機檔案，例如資料集和 z/OS UNIX System Services (USS) 檔案
- 大型主機磁碟，例如直接存取儲存裝置 (DASD)
- 大型主機磁帶 （虛擬或實體磁帶庫）

目標技術堆疊

- Amazon S3
- 虛擬私有雲端 (VPC) 中的 Amazon EC2 執行個體
- AWS Direct Connect
- Amazon Elastic File System (Amazon EFS)

目標架構

下圖顯示參考架構，其中大型主機上的 BMC AMI Cloud Data 軟體代理程式會驅動將資料存放在 Amazon S3 中的舊版資料備份和封存程序。

該圖顯示以下工作流程：

1. BMC AMI Cloud Data 軟體代理程式會在大型主機邏輯分割區 (LPARs) 上執行。軟體代理程式會透過 TCP/IP 從 DASD 或磁帶直接讀取和寫入大型主機資料至 Amazon S3。
2. AWS Direct Connect 設定內部部署網路與 之間的實體隔離連線 AWS。為了增強安全性，請在 上執行site-to-site Direct Connect 以加密傳輸中的資料。
3. S3 儲存貯體會將大型主機檔案儲存為物件儲存資料，BMC AMI Cloud Data 代理程式會直接與 S3 儲存貯體通訊。憑證用於代理程式和 Amazon S3 之間所有通訊的 HTTPS 加密。Amazon S3 資料加密用於加密和保護靜態資料。
4. BMC AMI 雲端資料管理伺服器會在 EC2 執行個體上以 Docker 容器的形式執行。執行個體會與在大型主機 LPARs代理程式通訊。 S3
5. Amazon EFS 同時掛載在主動和被動 EC2 執行個體上，以共用網路檔案系統 (NFS) 儲存。這是為了確保與管理伺服器上建立的政策相關的中繼資料在容錯移轉時不會遺失。如果作用中伺服器發生容錯移轉，則可以存取被動伺服器，而不會遺失任何資料。如果被動伺服器失敗，則可以存取作用中伺服器，而不會遺失任何資料。

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 中提供可擴展的運算容量 AWS 雲端。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon Elastic File System (Amazon EFS) 可協助您在 中建立和設定共用檔案系統 AWS 雲端。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取幾乎任何數量的資料。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。
- AWS Direct Connect會透過標準乙太網路光纖纜線將您的內部網路連結至某個 AWS Direct Connect 位置。透過此連線，您可以直接建立與公有 AWS 服務的虛擬介面，同時略過網路路徑中的網際網路服務供應商。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。

BMC 工具

- BMC AMI 雲端管理伺服器是一種 GUI 應用程式，在 Amazon EC2 的 Amazon Linux Amazon Machine Image (AMI) 上執行為 Docker 容器。管理伺服器提供管理 BMC AMI Cloud 活動的功能，例如報告、建立和管理政策、執行封存，以及執行備份、召回和還原。
- BMC AMI Cloud 代理程式會在內部部署大型主機 LPAR 上執行，該 LPAR 會使用 TCP/IP 直接讀取和寫入檔案至物件儲存體。已啟動的任務會在大型主機 LPAR 上執行，並負責讀取和寫入往返 Amazon S3 的備份和封存資料。
- BMC AMI Cloud Mainframe Command Line Interface (M9CLI) 為您提供一組命令，讓您直接從 TSO/E 或批次操作中執行 BMC AMI Cloud 動作，而不需要依賴管理伺服器。

史詩

建立 S3 儲存貯體和 IAM 政策

任務	Description	所需的技能
建立 S3 儲存貯體。	建立 S3 儲存貯體，以存放您要從大型主機環境備份和封存的檔案和磁碟區。	一般 AWS
建立 IAM 政策。	所有 BMC AMI Cloud 管理伺服器和代理程式都需要存取您在上一個步驟中建立的 S3 儲存貯體。 若要授予必要的存取權，請建立下列 IAM 政策： `{ "Version": "2012-10-17", "Statement": [ { "Sid": "Listfolder", "Action": [ "s3:ListBucket",`	一般 AWS

任務	Description	所需的技能

	<pre>"s3:GetBucketLocation",
	"s3:ListBucketVersions"
	],
	"Effect": "Allow",
	"Resource": [
	"arn:aws:s3:::<BucketName>"
	]
	},
	{
	"Sid": "Objectaccess",
	"Effect": "Allow",
	"Action": [
	"s3:PutObject",
	"s3:GetObjectAcl",
	"s3:GetObject",
	"s3:DeleteObjectVersion",
	"s3:DeleteObject",
	"s3:PutObjectAcl",
	"s3:GetObjectVersion"
	],
	"Resource": [</pre>	

任務	Description	所需的技能

	<pre>"arn:aws:s3:::<Bucket Name>/*"
	            ]
	        }
	    ]
	}</pre>	

取得 BMC AMI Cloud 軟體授權並下載軟體

任務	Description	所需的技能
取得 BMC AMI Cloud 軟體授權。	若要取得軟體授權金鑰，請聯絡 BMC AMI 雲端團隊。產生授權需要 z/OS D M=CPU命令的輸出。	建置潛在客戶
下載 BMC AMI Cloud 軟體和授權金鑰。	依照 BMC 文件的指示取得安裝檔案和授權金鑰。	大型主機基礎設施管理員

在大型主機上安裝 BMC AMI Cloud 軟體代理程式

任務	Description	所需的技能
安裝 BMC AMI Cloud 軟體代理程式。	1. 開始安裝程序之前，請確認已符合代理程式的最低軟體和硬體需求。 2. 若要安裝代理程式，請遵循 BMC 文件中的指示。 3. 代理程式開始在大型主機 LPAR 上執行後，請檢查多工緩衝處理中的ZM91000I MODEL9 BACKUP AGENT INITIALIZED 訊息。透	大型主機基礎設施管理員

任務	Description	所需的技能
	過尋找代理程式 STDOUT 中的訊息，確認代理程式與 S3 儲存貯體之間已成功建立連線。 `Object store connectivity has been established successfully`	

在 EC2 執行個體上設定 BMC AMI 雲端管理伺服器

任務	Description	所需的技能
建立 Amazon EC2 Linux 2 執行個體。	依照 Amazon EC2 文件中的步驟 1：啟動執行個體中的指示，在不同可用區域中啟動兩個 Amazon EC2 Linux 2 執行個體。 Amazon EC2 執行個體必須符合下列建議的硬體和軟體需求： • CPU – 最少 4 個核心 • RAM – 最低 8 GB • 磁碟機 – 40 GB • 建議的 EC2 執行個體 – C5.xlarge • 作業系統 – Linux • 軟體 – Docker、Unzip、vi/VIM • 網路頻寬 – 最低 1 GB 如需詳細資訊，請參閱 BMC 文件。	雲端架構師、雲端管理員

任務	Description	所需的技能
建立 Amazon EFS 檔案系統。	遵循 Amazon EFS 文件中步驟 1：建立 Amazon EFS 檔案系統的指示來建立 Amazon EFS 檔案系統。 EFS 建立檔案系統時，請執行下列動作： • 選擇標準儲存體方案。 • 選擇您用來啟動 EC2 執行個體的相同 VPC。	雲端管理員、雲端架構師

任務	Description	所需的技能

安裝 Docker 並設定管理伺服器。	連線至 EC2 執行個體：
	
	遵循 Amazon EC2 文件中來自連接至 Linux 執行個體的指示，連線至 EC2 執行個體。Amazon EC2
	
	設定 EC2 執行個體：
	
	對於每個 EC2 執行個體，請執行下列動作：
	
	1. 若要安裝 Docker，請執行命令：
	
	`sudo yum install docker`
	
	2. 若要啟動 Docker，請執行命令：
	
	`sudo service docker start`
	
	3. 若要驗證 Docker 的狀態，請執行 命令：
	
	`sudo service docker status`
	
	4. 在 `/etc/selinux` 資料夾中，將`config`檔案變更為 `SELINUX=permissive`。
	
	5. 將 `model9-v2.x.y_build_build-id-server.zip` 和
		雲端架構師、雲端管理員

任務	Description	所需的技能

	VerificationScripts.zip 檔案（您先前下載的）上傳到其中一個 EC2 執行個體中的暫存資料夾（例如，上傳到執行個體中的 /var/tmp 資料夾）。
	6. 若要前往 tmp 資料夾，請執行 命令：
	`cd/var/tmp`
	7. 若要解壓縮驗證指令碼，請執行 命令：
	`unzip VerificationScripts.zip`
	8. 若要變更目錄，請執行 命令：
	`cd /var/tmp/sysutils/PrereqsScripts`
	9. 若要執行驗證指令碼，請執行 命令：
	`./M9VerifyPrereqs.sh`
	10驗證指令碼提示輸入後，輸入 Amazon S3 URL 和連接埠號碼。然後，輸入 z/OS IP/DNS 和連接埠號碼。	

任務	Description	所需的技能
	Note 指令碼會執行檢查，以確認 EC2 執行個體可與在大型主機上執行的 S3 儲存貯體和代理程式連線。如果已建立連線，則會顯示成功訊息。	

任務	Description	所需的技能

安裝管理伺服器軟體。	1. 在您計劃建立作用中伺服器的 EC2 執行個體的根目錄（例如 /data/model9 ）中建立資料夾和子資料夾。
	2. 若要安裝套件amazon-efs-utils 和掛載先前建立的 Amazon EFS 檔案系統，請執行下列命令：
	`sudo yum install -y amazon-efs-utils`
	`sudo mount -t efs -o tls <File System ID>:/ /data/model9`
	3. 若要使用 Amazon EFS 檔案系統的項目更新 EC2 執行個體/etc/fstab 的檔案（以便在 Amazon EC2 重新啟動時自動重新掛載 Amazon EFS)，請執行 命令： Amazon EC2
	`<Amazon-EFS-file-system-id>:/ /data/model9 efs defaults,_netdev 0 0`
	4. 若要定義 BMC AMI Cloud 安裝檔案的路徑和目標安裝位置，請執行下列命令來匯出變數：
	`export MODEL9_HOME=/data/model9`
	`export M9INSTALL=/var/tmp`
		雲端架構師、雲端管理員

任務	Description	所需的技能

	Note
	建議您將這些 EXPORT 命令新增至.bashrc指令碼。
	5. 若要變更目錄，請執行 cd $MODEL9_HOME 命令，然後執行 mkdir diag命令來建立另一個子目錄。
	6. 若要解壓縮安裝檔案，請執行 命令：
	`unzip $M9INSTALL/model9-<v2.x.y>_build_<build-id>-server.zip`
	Note
	將 x.y（版本）和取代build-id為您的值。
	7. 若要部署應用程式，請執行下列命令：
	`docker load -i $MODEL9_HOME/model9-<v2.x.y>_build_<build-id>.docker`
	`docker load -i $MODEL9_HOME/postg`	

任務	Description	所需的技能

	`res-12.10-x86.docker.gz`
	
	Note
	將 v2.x.y（版本）和 取代build-id為您的值。
	
	8. 在 $MODEL9_HOME/conf資料夾中，更新model9-local.yml 檔案。
	
	Note
	有些參數具有預設值，其他參數則可視需要更新。如需詳細資訊，請參閱 model9-local.yml 檔案中的說明。
	
	9. 建立名為 的檔案$MODEL9_HOME/conf，然後將下列參數新增至檔案：
	
	`TZ=America/New_York`
	`EXTRA_JVM_ARGS=-Xmx2048m`
	
	10. 若要建立 Docker 網路橋接器，請執行 命令：	

任務	Description	所需的技能

	`docker network create -d bridge model9network`
	11.若要啟動適用於 BMC AMI Cloud 的 PostgreSQL 資料庫容器，請執行下列命令：
	`docker run -p 127.0.0.1:5432:5432 \ -v $MODEL9_HOME/db/data:/var/lib/postgresql/data:z \ --name model9db --restart unless-stopped \ --network model9network \ -e POSTGRES_PASSWORD=model9 -e POSTGRES_DB=model9 -d postgres:12.10`
	12.PostgreSQL 容器開始執行後，請執行下列命令來啟動應用程式伺服器：
	`docker run -d -p 0.0.0.0:443:443 -p 0.0.0.0:80:80 \ --sysctl net.ipv4.tcp_keepalive_time=600 \ --sysctl net.ipv4.tcp_keepalive_intvl=30 \`	

任務	Description	所需的技能

```
--sysctl net.ipv4.
tcp_keepalive_prob
es=10 \
-v $MODEL9_HOME:/mode
l9:z -h $(hostname)
 --restart unless-st
opped \
--env-file $MODEL9_H
OME/conf/model9.env
 \
--network model9net
work \
--name model9-v2.x.y
 model9:<v2.x.y>.<b
uild-id>
```

> **Note**
>
> 將 v2.x.y（版本）和 取代build-id為您的值。

13.若要檢查兩個容器的運作狀態，請執行 命令：

```
docker ps -a
```

14.若要在被動 EC2 執行個體上安裝管理伺服器，請重複步驟 1–4、7 和 10–13。

> **Note**
>
> 若要疑難排解問題，請前往存放在 /data/model9/logs/ 資料夾中的日誌。如需詳

任務	Description	所需的技能
	細資訊，請參閱 BMC 文件。	

新增代理程式，並在 BMC AMI 雲端管理伺服器上定義備份或封存政策

任務	Description	所需的技能
新增客服人員。	新增客服人員之前，請確認下列事項： • BMC AMI Cloud 代理程式正在大型主機 LPAR 上執行，並已完全初始化。透過在多工緩衝處理中尋找`ZM91000I MODEL9 BACKUP AGENT INITIALIZED` 初始化訊息來識別代理程式。 • 管理伺服器的 Docker 容器已完全初始化並執行。 您必須先在管理伺服器上建立代理程式，才能定義任何備份和封存政策。若要建立代理程式，請執行下列動作： 1. 使用 Web 瀏覽器存取部署在 Amazon EC2 機器上的管理伺服器，然後使用您的大型主機登入資料登入。 2. 選擇AGENTS 索引標籤，然後選擇新增代理程式。	大型主機儲存管理員或開發人員

任務	Description	所需的技能
	3. 在名稱中，輸入客服人員名稱。 4. 針對主機名稱/IP 地址，輸入大型主機的主機名稱或 IP 地址。 5. 在連接埠中，輸入您的連接埠號碼。 6. 選擇測試連線。如果連線成功建立，您可以看到成功訊息。 7. 選擇 CREATE。 建立代理程式後，您會在資料表中出現的新視窗中看到物件儲存體和大型主機代理程式的連線狀態。	
建立備份或封存政策。	1. 選擇政策。 2. ChooseCREATE 政策。 3. 在建立新政策頁面上，輸入您的政策規格。 Note 如需可用規格的詳細資訊，請參閱《BMC 文件》中的建立新政策。 4. 選擇完成。 5. 新政策現在會列為資料表。若要查看此表格，請選擇POLICIES 標籤。	大型主機儲存管理員或開發人員

從管理伺服器執行備份或封存政策

任務	Description	所需的技能
執行備份或封存政策。	手動或自動（根據排程）執行您先前從管理伺服器建立的資料備份或封存政策。若要手動執行政策： 1. 從導覽功能表中選擇 POLICIES 標籤。 2. 在您要執行之政策的資料表右側，選擇三點功能表。 3. 選擇立即執行。 4. 在彈出式確認視窗中，選擇是，立即執行政策。 5. 政策執行後，請在政策活動區段中驗證執行狀態。 6. 針對執行的政策，選擇三點功能表，然後選擇檢視執行日誌以查看日誌。 7. 若要確認已建立備份，請檢查 S3 儲存貯體。	大型主機儲存管理員或開發人員
還原備份或封存政策。	1. 在導覽功能表中，選擇POLICIES 標籤。 2. 選擇要執行還原程序的政策。這將列出過去針對該特定政策執行的所有備份或封存活動。 3. 若要選取您要還原的備份，請選擇日期時間欄。`file/Volume/Storage` 群組名稱會顯示政策的執行詳細資訊。	大型主機儲存管理員或開發人員

任務	Description	所需的技能
	4. 在資料表右側，選擇三點選單，然後選擇RESTORE。 5. 在快顯視窗中，輸入您的目標名稱、磁碟區和儲存群組，然後選擇 RESTORE。 6. 輸入您的大型主機登入資料，然後再次選擇 RESTORE。 7. 若要驗證還原是否成功，請檢查日誌或大型主機。	

從大型主機執行備份或封存政策

任務	Description	所需的技能
使用 M9CLI 執行備份或封存政策。	使用 M9CLI 從 TSO/E、REXX 或透過 JCLs 執行備份和還原程序，而無需在 BMC AMI 雲端管理伺服器上設定規則。 使用 TSO/E： 如果您使用 TSO/E，請確定 `M9CLI REXX` 與 串連`TSO`。若要透過 TSO/E 備份資料集，請使用 `TSO M9CLI BACKDSN <DSNAME>`命令。 **Note** 如需 M9CLI 命令的詳細資訊，請參閱《BMC 文件》中的 CLI 參考。	大型主機儲存管理員或開發人員

任務	Description	所需的技能

	使用 JCLs：
	
	若要使用 JCLs 執行備份和封存政策，請執行 M9CLI命令。
	
	使用批次操作：
	
	下列範例示範如何透過批次執行 M9CLI命令來封存資料集：
	
	`//JOBNAME JOB …`
	`//M9CLI EXEC PGM=IKJEF`
	`T01`
	`//STEPLIB DD DISP=SHR,`
	`DSN=<MODEL9 LOADLIB>`
	`//SYSEXEC DD DISP=SHR,`
	`DSN=<MODEL9 EXEC LIB>`
	`//SYSTSPRT DD SYSOUT=*`
	`//SYSPRINT DD SYSOUT=*`
	`//SYSTSIN DD TSO M9CLI`
	` ARCHIVE`
	` M9CLI ARCHIVE <DSNNAME`
	` OR DSN PATTERN>`
	`/`	

任務	Description	所需的技能
在 JCL 批次中執行備份或封存政策。	BMC AMI Cloud 提供名為 M9SAPIJ 的範例 JCL 常式。您可以自訂 M9SAPIJ，以使用 JCL 在管理伺服器上建立的特定政策。此任務也可以是批次排程器的一部分，用於自動執行備份和還原程序。 批次任務預期下列必要值： • 管理伺服器 IP 地址/主機名稱 • 連接埠號碼 • 政策 ID 或政策名稱（在管理伺服器上建立） **Note** 您也可以遵循範例任務的指示來變更其他值。	大型主機儲存管理員或開發人員

相關資源

- Mainframe Modernization with AWS (AWS 文件)
- 適用於大型主機的雲端備份如何使用 Model9 和 AWS 降低成本 (AWS 合作夥伴網路部落格)
- 如何使用 Model9 在 AWS 上啟用大型主機資料分析 (AWS 合作夥伴網路部落格)
- AWS Direct Connect 彈性建議 (AWS 文件)
- BMC AMI Cloud 文件 (BMC 網站)

使用 AWS Mainframe Modernization 和 建置 COBOL Db2 程式 AWS CodeBuild

Luis Gustavo Dantas 和 Eduardo Zimelewicz，Amazon Web Services

總結

> **Note**
>
> AWS Mainframe Modernization 服務（受管執行期環境體驗）不再開放給新客戶。對於類似 AWS Mainframe Modernization Service（受管執行期環境體驗）的功能，請探索 AWS Mainframe Modernization Service（自我管理體驗）。現有客戶可以繼續正常使用該服務。如需詳細資訊，請參閱AWS Mainframe Modernization 可用性變更。

此模式說明如何建立簡單的 AWS CodeBuild 專案，以使用 Replatform AWS Mainframe Modernization 工具預先編譯和繫結 COBOL Db2 程式。這可在 Replatform AWS Mainframe Modernization 執行時間環境中部署和執行這些程式。

COBOL 是一種業務導向的程式設計語言，由於其可靠性和可讀性，為許多關鍵應用程式提供支援。IBM Db2 是關聯式資料庫管理系統，可有效管理大量資料，並透過 SQL 與 COBOL 程式整合。COBOL 和 Db2 共同構成金融和政府等產業中關鍵任務操作的骨幹，儘管有較新的技術出現。

從大型主機環境遷移 COBOL 和 Db2 元件到其他平台會導致平台相容性、整合複雜性、資料遷移和效能最佳化等挑戰。移動這些關鍵元件需要仔細的規劃、技術專業知識和資源，以確保順利遷移，同時保持可靠性和功能。

AWS Mainframe Modernization 此服務提供工具和資源，以重建要在 AWS 基礎設施上執行的大型主機應用程式和資料庫，例如 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。這涉及在不重大程式碼變更的情況下將大型主機工作負載移至雲端。

Db2 預先編譯和繫結程序對於最佳化資料庫應用程式的效能和可靠性至關重要。預先編譯會將內嵌 SQL 陳述式轉換為可執行程式碼，從而降低執行時間負荷並提高效率。繫結程序會將預先編譯的程式碼與資料庫結構連結，促進存取路徑和查詢最佳化。此程序可確保資料完整性、改善應用程式回應能力，並防止安全漏洞。正確預先編譯和繫結的應用程式可將資源消耗降至最低、增強可擴展性，並降低 SQL Injection 攻擊的風險。

先決條件和限制

先決條件

- AWS 帳戶 和管理層級主控台存取。
- IBM Db2 資料庫系統，例如 z/OS 的 IBM Db2 或 Linux、Unix 和 Windows (LUW) 的 Db2。
- IBM Data Server Client 軟體，可從 IBM 網站下載。如需詳細資訊，請參閱 IBM Data Server 用戶端和資料伺服器驅動程式類型。
- 要編譯和繫結的 COBOL Db2 程式。或者，此模式提供您可以使用的基本範例程式。
- 上的虛擬私有雲端 (VPC) AWS 與私有網路。如需建立 VPC 的資訊，請參閱 Amazon Virtual Private Cloud (Amazon VPC) 文件。
- 來源控制儲存庫，例如 GitHub 或 GitLab。

限制

- 如需 AWS CodeBuild 配額，請參閱 的配額 AWS CodeBuild。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

Architecture

來源技術堆疊

來源堆疊包括：

- 使用 Db2 資料庫存放資料的 COBOL 程式
- z/OS 前置編譯器的 IBM COBOL 編譯器和 Db2
- 大型主機設定的其他部分，例如檔案系統、交易管理員和多工緩衝處理

目標技術堆疊

此模式的方法適用於兩個選項：將資料從 z/OS 的 Db2 移至 LUW 的 Db2，或保留在 z/OS 的 Db2 上。目標架構包括：

- 使用 Db2 資料庫存放資料的 COBOL 程式
- AWS Mainframe Modernization Replatform 編譯工具
- AWS CodeBuild 做為建置應用程式的基礎設施
- 其他 AWS 雲端 資源，例如 Amazon Linux

目標架構

此圖展示了以下要點：

1. 使用者將其程式碼上傳至來源控制儲存庫，例如 GitHub 或 GitLab。
2. AWS CodePipeline 會注意到變更，並從儲存庫取得程式碼。
3. CodePipeline 會啟動 AWS CodeBuild 並傳送程式碼。
4. CodeBuild 遵循`buildspec.yml`範本中的指示 (在其他資訊區段中提供），以：
   a. 從 Amazon Simple Storage Service (Amazon S3) 儲存貯體取得 IBM Data Server 用戶端。
   b. 安裝和設定 IBM Data Server 用戶端。
   c. 從 擷取 Db2 登入資料 AWS Secrets Manager。
   d. 連線至 Db2 伺服器。
   e. 預先編譯、編譯和繫結 COBOL 程式。
   f. 將成品儲存在 S3 儲存貯體中 AWS CodeDeploy 以供使用。
5. CodePipeline 啟動 CodeDeploy。
6. CodeDeploy 會協調其代理程式，這些代理程式已安裝在執行時間環境中。代理程式會從 Amazon S3 擷取應用程式，並根據 中的指示進行安裝`appspec.yml`。

為了保持建置簡單且專注於建置，此模式中的指示涵蓋步驟 1 到 4，但不包含 COBOL Db2 程式的部署。

自動化和擴展

為了簡化，此模式說明如何手動佈建資源。不過，有許多自動化選項可供使用 CloudFormation AWS Cloud Development Kit (AWS CDK)，例如， 和 HashiCorp Terraform，可自動化這些任務。如需詳細資訊，請參閱 CloudFormation和 AWS CDK 文件。

工具

AWS 服務

- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeDeploy 會自動部署到 Amazon EC2 或內部部署執行個體、 AWS Lambda 函數或 Amazon Elastic Container Service (Amazon ECS) 服務。

- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- AWS Mainframe Modernization 提供工具和資源，協助您規劃和實作從大型主機到 AWS 受管執行期環境的遷移和現代化。

其他工具

- Replatform 工具的 Amazon ECR AWS Mainframe Modernization 映像。若要編譯 COBOL 應用程式，您需要使用包含 Replatform AWS Mainframe Modernization 工具的 Amazon Elastic Container Registry (Amazon ECR) 映像來啟動 CodeBuild：

  ```
  673918848628.dkr.ecr.<your-region>.amazonaws.com/m2-enterprise-build-tools:9.0.7.R1
  ```

  如需可用 ECR 映像的詳細資訊，請參閱AWS Mainframe Modernization 《使用者指南》中的教學課程。
- IBM Data Server Client 軟體對於在 CodeBuild 中預先編譯和繫結 COBOL Db2 程式至關重要。它充當 COBOL 編譯器和 Db2 之間的橋樑。

最佳實務

- 並非每個 COBOL 程式都依賴 Db2 作為其資料持久性層。確保用於存取 Db2 的編譯指令僅適用於專門設計用於與 Db2 互動的 COBOL 程式。實作邏輯來區分 COBOL Db2 程式和不使用 Db2 的 COBOL 程式。
- 我們建議您避免編譯尚未修改的程式。實作程序來識別哪些程式需要編譯。

史詩

建立雲端基礎設施

任務	Description	所需的技能
建立 S3 儲存貯體以託管 IBM Data Server 用戶端和管道成品。	您需要設定 S3 儲存貯體以 (a) 上傳 IBM Data Server 用戶端、(b) 從儲存庫存放程式碼，	一般 AWS

任務	Description	所需的技能
	以及 (c) 存放建置程序的結果。 1. 登入 AWS 管理主控台，然後開啟 Amazon S3 主控台。 2. 選擇現有的 S3 儲存貯體或建立新的儲存貯體。請注意儲存貯體的 Amazon Resource Name (ARN) 以供日後使用。 如需建立 S3 儲存貯體的方法，請參閱 Amazon S3 文件。	

任務	Description	所需的技能
將 IBM Data Server 用戶端上傳至 S3 儲存貯體。	1. 在 Amazon S3 主控台上，選擇要開啟的儲存貯體。 2. 選擇建立資料夾，將其名稱指定為用戶端，然後選擇建立資料夾。 3. 開啟用戶端資料夾，選擇上傳、新增檔案。 4. 選擇您先前從 IBM 網站下載到本機檔案系統的 IBM Data Server Client 檔案。 https://www.ibm.com/docs/en/db2/11.5?topic=overviews-data-server-clients 檔案名稱應類似於 `v11.5.8_linuxx64_client.tar.gz` 或 `v11.5.9_linuxx64_client.tar.gz` 。 5. 選擇開啟、上傳並等待上傳完成。 6. 在檔案和資料夾索引標籤上，選擇資料伺服器用戶端，並記下其 S3 URI。	一般 AWS

任務	Description	所需的技能
為您的 Db2 登入資料建立 AWS Secrets Manager 秘密。	若要建立秘密以安全地存放您的 DB2 登入資料： 1. 在 Secrets Manager 主控台上，選擇儲存新的秘密。 2. 在選擇秘密類型窗格中，選擇另一種類型的秘密和純文字。 3. 在純文字方塊中，使用下列 JSON 結構輸入您的 Db2 憑證。 `{ "username": "<your-db2-user-name>", "password": "<your-db2-password>", "db2node": "db2dev", "db2host": "<your-db2-hostname-or-IP>", "db2port": <your-db2-port>, "db2name": "<your-db2-location>", "qualifier": "<your-db2-qualifier>" }` 4. 選擇下一步，並為秘密命名，例如 `dev-db2-cred` 。	一般 AWS

任務	Description	所需的技能
	5. 選擇下一步、下一步和存放區。 如需建立秘密的詳細資訊，請參閱 Secrets Manager 文件。	
確認 Db2 可從 VPC 子網路存取。	AWS CodeBuild 需要與 Db2 伺服器的連線，資料伺服器用戶端才能執行預先編譯和繫結操作。確定 CodeBuild 可以透過安全連線到達 Db2 伺服器。 1. 開啟 Amazon VPC 主控台。 2. 在導覽窗格中，選擇子網路，並記下 CodeBuild 將運作之私有子網路的 IDs 和 IPv4 CIDRs。 3. 透過引入傳入規則，更新 Db2 系統的目前網路存取控制設定。此規則應僅從與 CodeBuild 專案相關聯的子網路 CIDR 啟用對 Db2 連接埠的自訂 TCP 存取。CIDRs	網路管理員，一般 AWS

建立應用程式成品

任務	Description	所需的技能
建立 COBOL Db2 資產。	1. 如果您想要使用簡單的 COBOL Db2 範例，請將下列原始碼儲存為 `CDB2SMP.cbl` 。或者，您	應用程式開發人員

任務	Description	所需的技能

	可以將此範例取代為您已擁有的程式。
	
	``
	       IDENTIFICATION
	  DIVISION.
	       PROGRAM-ID.
	  CDB2SMP.
	       DATA
	 DIVISION.
	       WORKING-S
	TORAGE SECTION.
	       01 WS-NAME
	                 PIC
	 X(100).
	       PROCEDURE
	 DIVISION.
	           EXEC SQL
	               SELECT
	 NAME INTO :WS-NAME
	               FROM
	 SYSIBM.SYSTABLES
	           END-EXEC
	           GOBACK.
	``
	2. 遞交變更並將檔案推送到您的儲存庫。	

建立 buildspec.yml 檔案。	1. 根據其他資訊區段中提供的範例產生buildspec.yml 檔案。 2. 遞交變更並將檔案推送到您的儲存庫。	AWS DevOps

任務	Description	所需的技能
將您的儲存庫連線至 CodePipeline。	1. 開啟AWS 開發人員工具主控台。 2. 在導覽窗格中，選擇設定、連線。 3. 針對您選擇的來源提供者，請遵循開發人員工具主控台文件中的指示。 在後續步驟中為 CodePipeline 建立 (IAM) 政策時，您將需要連線的 Amazon Resource Name AWS Identity and Access Management (ARN)。	AWS DevOps

設定 許可

任務	Description	所需的技能
為 CodeBuild 建立 IAM 政策。	CodeBuild 專案需要存取一些資源，包括 Secrets Manager 和 Amazon S3。 設定必要的許可： 1. 開啟 IAM 主控台。 2. 在導覽窗格中，選擇政策、建立政策，然後選取 CodeBuild 服務。 3. 將格式化從視覺化切換到 JSON，並將額外資訊區段中提供的 CodeBuild 政策複製到政策編輯器欄位。	一般 AWS

任務	Description	所需的技能
	4. 在下一個步驟中命名並儲存此政策以供未來參考。 如需建立 IAM 政策的詳細資訊，請參閱 IAM 文件。	
為 CodeBuild 建立 IAM 角色。	若要讓 CodeBuild 提供安全政策，您需要設定 IAM 角色。 若要建立此角色： 1. 在 IAM 主控台的導覽窗格中，選擇角色、建立角色。 3. 對於信任的實體類型，請保留AWS 服務預設設定。 4. 針對使用案例，選取 CodeBuild 服務，然後選擇下一步。 4. 在可用的 IAM 政策清單中，找到您為 CodeBuild 建立的政策，然後選擇下一步將其連接至角色。 5. 指定角色的名稱，然後選擇建立角色以儲存該角色，以供未來在 CodeBuild 中參考。 如需為 建立 IAM 角色的詳細資訊 AWS 服務，請參閱 IAM 文件。	一般 AWS

任務	Description	所需的技能
建立 CodePipeline 的 IAM 政策。	AWS CodePipeline 管道需要存取一些資源，包括您的程式碼儲存庫和 Amazon S3。 重複先前為 CodeBuild 提供的步驟，以建立 CodePipeline 的 IAM 政策（在步驟 2 中，選擇 CodePipeline 而非 CodeBuild）。	AWS DevOps

任務	Description	所需的技能

為 CodePipeline 建立 IAM 角色。	若要讓 CodePipeline 可以使用安全政策，您需要設定 IAM 角色。
	
	若要建立此角色：
	
	1. 在 IAM 主控台上，選擇角色、建立角色。
	2. 對於 Trusted entity type (信任的實體類型)，選擇 Custom trust policy (自訂信任政策)。
	
	將會顯示具有空白`Principal`元素的政策。
	3. 在大括號之間的`Principal`行上，新增：
	
	`"Service": "codepipeline.amazonaws.com"`
	
	信任政策看起來會如下所示：
	
	`{ "Version": "2012-10-17", "Statement": [ { "Sid": "Statement1", "Effect": "Allow", "Principal": {`
		AWS DevOps

任務	Description	所需的技能
	<pre>"Service": "codepipeline.amazonaws.com" }, "Action": "sts:AssumeRole" }] }</pre> 4. 選擇下一步。 5. 在可用的 IAM 政策清單中，找到您為 CodePipeline 建立的政策，然後選擇下一步將其連接至角色。 6. 指定角色的名稱，然後選擇建立角色以儲存該角色，以供未來在 CodePipeline 中參考。	

編譯並繫結 COBOL Db2 程式

任務	Description	所需的技能
建立 CodePipeline 管道和 CodeBuild 專案。	若要建立 CodePipeline 管道和 CodeBuild 專案，以編譯和繫結 COBOL Db2 程式： 1. 開啟 CodePipeline 主控台，然後選擇建立管道、建置自訂管道。 2. 指定管道的名稱。 3. 針對服務角色，選擇現有服務角色，然後選擇為您為 CodePipeline 建立的 IAM 角色指定 ARN。	AWS DevOps

任務	Description	所需的技能
	4. 展開進階設定，選擇自訂位置，選擇您先前建立的 S3 儲存貯體，然後選擇下一步。 5. 針對來源提供者，選取您的第三方來源提供者，並提供提供者的相關資訊： a. 針對連線，選取為來源提供者建立的連線。 b. 針對儲存庫名稱，選取您的儲存庫。 c. 針對預設分支，選取存放 COBOL 程式的分支和 `buildspec.yml` 。 d. 選擇下一步。 6. 針對建置提供者，選擇其他建置提供者，AWS CodeBuild。 7. 針對專案名稱，選擇建立專案。 主控台會顯示 CodeBuild 視窗，您可以在其中建立建置專案。在此視窗中： a. 輸入專案的名稱。 b. 針對 Environment image (環境映像)，選擇 Custom image (自訂映像)。 c. 針對環境類型，選擇 Linux 容器。 d. 針對 ECR 帳戶，選擇其他 ECR 帳戶。	

任務	Description	所需的技能
	e. 針對 Amazon ECR 儲存庫 URI，輸入：673918848628.dkr.ecr.<your-region>.amazonaws.com/m2-enterprise-build-tools:8.0.9.R1 。 f. 針對服務角色，選擇現有的服務角色，然後選取您為 CodeBuild 建立的角色。 g. 展開其他組態區段，然後選擇此專案的 VPC、私有子網路和安全群組。 h. 在 Buildspec 區段中，選擇使用 buildspec 檔案。 i. 在視窗結尾處，選擇繼續至 CodePipeline。CodeBuild 視窗會關閉，讓您可以返回 CodePipeline 主控台。 8. 返回 CodePipeline 主控台，選擇下一步。 9. 在新增部署階段窗格中，選擇略過部署階段並確認。 10.檢閱管道參數，然後選擇建立管道。	
檢閱輸出。	透過檢閱 CodePipeline 建置日誌來驗證建置是否成功。	AWS DevOps

任務	Description	所需的技能

在 Db2 中檢查結果。	驗證 SYSPLAN 資料表上的套件版本。
	
	`select CAST(NAME AS VARCHAR(10)) as name, VALIDATE, LAST_BIND_TIME, LASTUSED, CAST(PKGVERSION AS VARCHAR(10)) as PKGVERSION from SYSIBM.SYSPLAN where NAME = 'CDB2SMP' order by LAST_BIND_TIME desc`
	
	版本必須符合 CodeBuild 組建 ID，這CDB2SMP在我們的範例中：
	
	`NAME VALIDATE LAST_BIND_TIME LASTUSED PKGVERSION ---------- -------- ---------- ---------- -------- ---------- CDB2SMP B 2024-05-18-11.53.11.503738 01/01/0001 19`	

疑難排解

問題	解決方案
當您在服務之間移動時， AWS 主控台偶爾會切換區域。	每當您切換服務 AWS 區域 時，請務必驗證選取的 。

問題	解決方案
	選擇 AWS 區域 器位於主控台視窗的右上角。
從 CodeBuild 識別 Db2 連線問題可能很困難。	若要疑難排解連線問題，請將下列 DB2 連線命令新增至 `buildspec.yml` 檔案。此新增可協助您偵錯和解決連線問題。 `db2 connect to $DB_NAME user $DB2USER using $DB2PASS`
有時，IAM 主控台中的角色窗格不會立即顯示您建立的 IAM 政策。	如果您遇到延遲，請重新整理畫面以顯示最新資訊。

相關資源

IBM 文件

- IBM Data Server 用戶端和驅動程式類型
- 下載 IBM Data Server 用戶端和驅動程式類型

AWS 文件

- Amazon S3 使用者指南
- AWS CodeBuild 使用者指南
- AWS Mainframe Modernization 使用者指南
- AWS Secrets Manager 使用者指南
- AWS CodePipeline 使用者指南
- AWS CodeDeploy 使用者指南

其他資訊

CodeBuild 政策

將預留位置 `<RegionID>`、`<AccountID>`、`<BucketARN>`、 `<SubnetARN>`和 取代`<DB2CredSecretARN>`為您的值。

```
{"Version": "2012-10-17",
    "Statement": [
        {"Action": "ecr:GetAuthorizationToken", "Effect": "Allow", "Resource": "*" },
        {"Action": ["ecr:GetDownloadUrlForLayer", "ecr:BatchGetImage",
                    "ecr:BatchCheckLayerAvailability"],
         "Effect": "Allow",
         "Resource": "arn:aws:ecr:*:673918848628:repository/m2-enterprise-build-
tools"},
        {"Action": "s3:PutObject", "Effect": "Allow", "Resource": "arn:aws:s3:::aws-m2-
repo-*/*"},
        {"Action": ["logs:PutLogEvents", "logs:CreateLogStream",
 "logs:CreateLogGroup"],
         "Effect": "Allow", "Resource": "arn:aws:logs:<RegionId>:<AccountId>:*"},
        {"Action": ["ec2:DescribeVpcs", "ec2:DescribeSubnets",
                    "ec2:DescribeSecurityGroups", "ec2:DescribeNetworkInterfaces",
                    "ec2:DescribeDhcpOptions", "ec2:DeleteNetworkInterface",
                    "ec2:CreateNetworkInterface"],
         "Effect": "Allow", "Resource": "*"},
        {"Action": "ec2:CreateNetworkInterfacePermission",
         "Effect": "Allow", "Resource": ["<SubnetARN>"]},
        {"Action": "s3:*", "Effect": "Allow", "Resource": ["<BucketARN>/
*","<BucketARN>"]},
        {"Action": "secretsmanager:GetSecretValue",
         "Effect": "Allow", "Resource": "<DB2CredSecretARN>"}
    ]
}
```

CodePipeline 政策

將預留位置 <BucketARN>和 取代<ConnectionARN>為您的值。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {"Action": ["s3:List*", "s3:GetObjectVersion", "s3:GetObject",
 "s3:GetBucketVersioning" ],
        "Effect": "Allow",
        "Resource": ["<BucketARN>/*", "<BucketARN>"]},
        {"Action": ["codebuild:StartBuild", "codebuild:BatchGetBuilds"],
         "Effect": "Allow", "Resource": "*"},
        {"Action": ["codestar-connections:UseConnection"],
         "Effect": "Allow", "Resource": "<ConnectionARN>"}
        ]
```

```
}
```

buildspec.yml

將<your-bucket-name>預留位置取代為您實際的 S3 儲存貯體名稱。

```
version: 0.2
phases:
  pre_build:
    commands:
      - /var/microfocuslicensing/bin/mfcesd -no > /var/microfocuslicensing/logs/
mfcesd_startup.log 2>&1 &
      - |
        mkdir $CODEBUILD_SRC_DIR/db2client
        aws s3 cp s3://<your-bucket-name>/v11.5.8_linuxx64_client.tar.gz
 $CODEBUILD_SRC_DIR/db2client/ >> /dev/null 2>&1
        tar -xf $CODEBUILD_SRC_DIR/db2client/v11.5.8_linuxx64_client.tar.gz -C
 $CODEBUILD_SRC_DIR/db2client/
        cd $CODEBUILD_SRC_DIR/db2client/
        ./client/db2_install -f sysreq -y -b /opt/ibm/db2/V11.5 >> /dev/null 2>&1

        useradd db2cli
        /opt/ibm/db2/V11.5/instance/db2icrt -s client -u db2cli db2cli
        DB2CRED=$(aws secretsmanager get-secret-value --secret-id dev-db2-cred | jq -r
 '.SecretString | fromjson')
        read -r DB2USER DB2PASS DB_NODE DB_HOST DB_PORT DB_NAME DB_QUAL <<<$(echo
 $DB2CRED | jq -r
 '.username, .password, .db2node, .db2host, .db2port, .db2name, .qualifier')
        . /home/db2cli/sqllib/db2profile
        db2 catalog tcpip node $DB_NODE remote $DB_HOST server $DB_PORT
        db2 catalog db $DB_NAME as $DB_NAME at node $DB_NODE authentication server
  build:
    commands:
      - |
        revision=$CODEBUILD_SRC_DIR/loadlib
        mkdir -p $revision; cd $revision
        . /opt/microfocus/EnterpriseDeveloper/bin/cobsetenv
        cob -zU $CODEBUILD_SRC_DIR/CDB2SMP.cbl -C "DB2(DB==${DB_NAME} PASS==${DB2USER}.
${DB2PASS} VERSION==${CODEBUILD_BUILD_NUMBER} COLLECTION==DB2AWSDB"
artifacts:
  files:
    - "**/*"
  base-directory: $revision
```

使用 Amazon EC2 Auto Scaling 和 Systems Manager 建置 Micro Focus Enterprise Server PAC

Kevin Yung 和 Krithika Palani Selvam，Amazon Web Services

Peter Woods，無

Abraham Rondon，微型焦點

總結

此模式為在橫向擴展效能和可用性叢集 (PAC) 中使用 Micro Focus Enterprise Server 的大型主機應用程式引入可擴展架構，以及在 Amazon Web Services () 上使用 Amazon Elastic Compute Cloud (Amazon EC2) Auto Scaling 群組AWS。解決方案使用 AWS Systems Manager 和 Amazon EC2 Auto Scaling 生命週期掛鉤進行全自動化。透過使用此模式，您可以設定大型主機線上和批次應用程式，根據您的容量需求自動擴展和擴展，以達到高彈性。

> **Note**
>
> 此模式已使用 Micro Focus Enterprise Server 6.0 版進行測試。對於版本 8，請參閱設定 Micro Focus 執行期 （ 在 Amazon EC2 上 ）。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Micro Focus Enterprise Server 軟體和授權。如需詳細資訊，請聯絡 Micro Focus 銷售。
- 了解重建和交付大型主機應用程式的概念，以便在 Micro Focus Enterprise Server 中執行。如需高階概觀，請參閱 Micro Focus Enterprise Server 資料表。
- 了解 Micro Focus Enterprise Server 向外擴展效能和可用性叢集中的概念。如需詳細資訊，請參閱 Micro Focus Enterprise Server 文件。
- 了解具有持續整合 (CI) 的大型主機應用程式 DevOps 的整體概念。如需 AWS和 Micro Focus 開發的 AWS 方案指引模式，請參閱大型主機現代化： AWS 使用 Micro Focus 在 上進行 DevOps。

> **Note**
>
> 此模式已使用 Micro Focus Enterprise Server 第 6 版進行測試。對於版本 8，請參閱設定 Micro Focus 執行期（在 Amazon EC2 上）。

限制

- 如需 Micro Focus Enterprise Server 支援的平台清單，請參閱 Micro Focus Enterprise Server 資料表。
- 此模式中使用的指令碼和測試是以 Amazon EC2 Windows Server 2019 為基礎；未針對此模式測試其他 Windows Server 版本和作業系統。
- 模式是以適用於 Windows 的 Micro Focus Enterprise Server 6.0 為基礎；此模式的開發中未測試較舊版本或較新版本。

產品版本

- Micro Focus Enterprise Server 6.0
- Windows Server 2019

Architecture

在傳統大型主機環境中，您必須佈建硬體來託管應用程式和公司資料。為了滿足季節性、每月、每季或甚至前所未見或非預期需求的高峰，大型主機使用者必須透過購買額外的儲存和運算容量來橫向擴展。增加儲存和運算容量資源的數量可改善整體效能，但擴展不是線性的。

當您使用 Amazon EC2 Auto Scaling 和 Micro Focus Enterprise Server 開始在 AWS 上採用隨需耗用模型時，並非如此。下列各節詳細說明如何使用 Micro Focus Enterprise Server 橫向擴展效能和可用性叢集 (PAC) 搭配 Amazon EC2 Auto Scaling 群組來建置全自動化、可擴展的大型主機應用程式架構。

Micro Focus Enterprise Server 自動擴展架構

首先，請務必了解 Micro Focus Enterprise Server 的基本概念。此環境為傳統上在 IBM 大型主機上執行的應用程式提供與大型主機相容的 x86 部署環境。它同時提供線上和批次執行，以及支援下列項目的交易環境：

- IBM COBOL

- IBM PL/I
- IBM JCL 批次任務
- IBM CICS 和 IMS TM 交易
- Web 服務
- 常見的批次公用程式，包括 SORT

Micro Focus Enterprise Server 可讓大型主機應用程式以最少的變更執行。現有的大型主機工作負載可以移至 x86 平台並進行現代化，以利用 AWS 雲端原生擴充功能快速擴展到新市場或地理。

AWS 規範指引模式 大型主機現代化：使用 Micro Focus 的 AWS 上的 DevOps 引進了 架構，以使用 Micro Focus Enterprise Developer 和 Enterprise Test Server 搭配 AWS CodePipeline 和 AWS CodeBuild 來加速開發和測試 AWS 上的大型主機應用程式。此模式著重於將大型主機應用程式部署到 AWS 生產環境，以實現高可用性和彈性。

在大型主機生產環境中，您可能已在大型主機中設定 IBM Parallel Sysplex，以實現高效能和高可用性。為了建立類似於 Sysplex 的向外擴展架構，Micro Focus 將效能和可用性叢集 (PAC) 引入企業伺服器。PACs支援在多個 Enterprise Server 區域部署大型主機應用程式，並以單一映像形式管理，並在 Amazon EC2 執行個體中向外擴展。PACs也支援可預測的應用程式效能和隨需系統輸送量。

在 PAC 中，多個 Enterprise Server 執行個體可做為單一邏輯實體一起運作。因此，一個 Enterprise Server 執行個體的故障不會中斷業務連續性，因為容量會與其他區域共用，而新的執行個體會使用產業標準功能自動啟動，例如 Amazon EC2 Auto Scaling 群組。這可移除單一故障點，改善硬體、網路和應用程式問題的彈性。向外擴展的 Enterprise Server 執行個體可以使用 Enterprise Server Common Web Administration (ESCWA) APIs 操作和管理，簡化 Enterprise Server 的操作維護和服務性。

> **Note**
>
> Micro Focus 建議效能和可用性叢集 (PAC) 應至少包含三個企業伺服器區域，以便在企業伺服器區域故障或需要維護時，可用性不會受到影響。

PAC 組態需要支援的關聯式資料庫管理服務 (RDBMS) 來管理區域資料庫、跨區域資料庫和選用的資料存放區資料庫。資料存放區資料庫應使用 Micro Focus 資料庫檔案處理常式支援來管理虛擬儲存存取方法 (VSAM) 檔案，以改善可用性和可擴展性。支援的 RDBMSs包括下列項目：

- Microsoft SQL Server 2009 R2 及更新版本
- PostgreSQL 10.x，包括 Amazon Aurora PostgreSQL 相容版本

- DB2 10.4 及更新版本

如需支援的 RDBMS 和 PAC 需求的詳細資訊，請參閱 Micro Focus Enterprise Server - 先決條件和 Micro Focus Enterprise Server - 建議的 PAC 組態。

下圖顯示 Micro Focus PAC 的一般 AWS 架構設定。

	元件	Description
1	Enterprise Server 執行個體自動擴展群組	設定在 PAC 中使用 Enterprise Server 執行個體部署的自動擴展群組。Amazon CloudWatch 警示可以使用 CloudWatch 指標向外或向內擴展執行個體數量。
2	Enterprise Server ESCWA 執行個體自動擴展群組	設定使用 Enterprise Server Common Web Administration (ESCWA) 部署的自動擴展群組。ESCWA 提供叢集管理 APIs。 ESCWA 伺服器做為控制平面，以在 Enterprise Server 執行個體自動擴展事件期間新增或移除 Enterprise Server，以及啟動或停止 PAC 中的 Enterprise Server 區域。由於 ESCWA 執行個體僅用於 PAC 管理，其流量模式是可預測的，而且其自動擴展所需的容量需求可以設定為 1。
3	多可用區設定中的 Amazon Aurora 執行個體	設定關聯式資料庫管理系統 (RDBMS) 來託管要跨企業伺服器執行個體共用的使用者和系統資料檔案。

4	Amazon ElastiCache (Redis OSS) 執行個體和複本	設定 ElastiCache (Redis OSS) 主要執行個體和至少一個複本來託管使用者資料，並做為 Enterprise Server 執行個體的橫向擴展儲存庫 (SOR)。您可以設定一或多個橫向擴展儲存庫來存放特定類型的使用者資料。 Enterprise Server 使用 Redis NoSQL 資料庫做為 SOR，這是維護 PAC 完整性的需求。
5	Network Load Balancer	設定負載平衡器，為應用程式提供主機名稱，以連線至 Enterprise Server 執行個體提供的服務（例如，透過 3270 模擬器存取應用程式）。

這些元件構成 Micro Focus Enterprise Server PAC 叢集的最低需求。下一節涵蓋叢集管理自動化。

使用 AWS Systems Manager 自動化進行擴展

在 AWS 上部署 PAC 叢集之後，會透過 Enterprise Server Common Web Administration (ESCWA) APIs 來管理 PAC。

若要在自動擴展事件期間自動化叢集管理任務，您可以使用 Systems Manager Automation Runbook 和 Amazon EC2 Auto Scaling 搭配 Amazon EventBridge。這些自動化的架構如下圖所示。

	元件	Description
1	自動擴展生命週期掛鉤	設定自動擴展生命週期關聯，並在啟動新執行個體並在自動擴展群組中終止現有執行個體時將通知傳送至 Amazon EventBridge。

2	Amazon EventBridge	設定 Amazon EventBridge 規則，將自動擴展事件路由至 Systems Manager Automation Runbook 目標。
3	自動化 Runbook	設定 Systems Manager Automation Runbook 以執行 Windows PowerShell 指令碼，並叫用 ESCWA APIs 來管理 PAC。如需範例，請參閱其他資訊一節。
4	自動擴展群組中的 Enterprise Server ESCWA 執行個體	在自動擴展群組中設定 Enterprise Server ESCWA 執行個體。ESCWA 執行個體提供 APIs 來管理 PAC。

工具

- Micro Focus Enterprise Server – Micro Focus Enterprise Server 為使用 Enterprise Developer 的任何整合開發環境 (IDE) 變體建立的應用程式提供執行環境。
- Amazon EC2 Auto Scaling – Amazon EC2 Auto Scaling 可協助您確保有正確數量的 Amazon EC2 執行個體可用於處理應用程式的負載。您可以建立稱為 Auto Scaling 群組的 EC2 執行個體集合，並指定執行個體的最小和最大數量。
- Amazon ElastiCache (Redis OSS) – Amazon ElastiCache 是一種 Web 服務，用於設定、管理和擴展雲端中的分散式記憶體內資料存放區或快取環境。它提供高效能、可擴展且符合成本效益的快取解決方案。
- Amazon RDS – Amazon Relational Database Service (Amazon RDS) 是一種 Web 服務，可讓您更輕鬆地在 AWS 雲端中設定、操作和擴展關聯式資料庫。它為關聯式資料庫提供經濟實惠、可擴展的容量，並管理常見的資料庫管理任務。
- AWS Systems Manager – AWS Systems Manager 是一種 AWS 服務，可用來檢視和控制 AWS 上的基礎設施。使用 Systems Manager 主控台，您可以檢視來自多個 AWS 服務的操作資料，並自動化 AWS 資源的操作任務。Systems Manager 透過掃描您的受管執行個體並報告 (或採取修正動作) 其偵測的任何政策違規，協助您保持安全與合規。

史詩

建立 Amazon Aurora 執行個體

任務	Description	所需的技能
為 Amazon Aurora 執行個體建立 AWS CloudFormation 範本。	使用 AWS 範例程式碼片段建立 CloudFormation 範本，以建立 Amazon Aurora PostgreSQL 相容版本執行個體。	雲端架構師
部署 CloudFormation 堆疊以建立 Amazon Aurora 執行個體。	使用 CloudFormation 範本建立 Aurora PostgreSQL 相容執行個體，該執行個體已針對生產工作負載啟用異地同步備份複寫。	雲端架構師
設定 Enterprise Server 的資料庫連線設定。	遵循 Micro Focus 文件中的指示，為 Micro Focus Enterprise Server 準備連線字串和資料庫組態。	資料工程師、DevOps 工程師

為 Redis 執行個體建立 Amazon ElastiCache 叢集

任務	Description	所需的技能
為 Redis 執行個體的 Amazon ElastiCache 叢集建立 CloudFormation 範本。	使用 AWS 範例程式碼片段建立 CloudFormation 範本，為 Redis 執行個體建立 Amazon ElastiCache 叢集。	雲端架構師
部署 CloudFormation 堆疊，為 Redis 執行個體建立 Amazon ElastiCache 叢集。	為已針對生產工作負載啟用異地同步備份複寫的 Redis 執行個體建立 Amazon ElastiCache 叢集。	雲端架構師
設定 Enterprise Server PSOR 連線設定。	遵循 Micro Focus 文件中的指示，為 Micro Focus Enterprise	DevOps 工程師

任務	Description	所需的技能
	Server PAC 準備 PAC 橫向擴展儲存庫 (PSOR) 連線組態。	

建立 Micro Focus Enterprise Server ESCWA 自動擴展群組

任務	Description	所需的技能
建立 Micro Focus Enterprise Server AMI。	建立 Amazon EC2 Windows Server 執行個體，並在 EC2 執行個體中安裝 Micro Focus Enterprise Server 二進位檔。建立 EC2 執行個體的 Amazon Machine Image (AMI)。如需詳細資訊，請參閱Enterprise Server 安裝文件。	雲端架構師
為 Enterprise Server ESCWA 建立 CloudFormation 範本。	使用 AWS 範例程式碼片段建立範本，以在自動擴展群組中建立 Enterprise Server ESCWA 的自訂堆疊。	雲端架構師
部署 CloudFormation 堆疊，為 Enterprise Server ESCWA 建立 Amazon EC2 擴展群組。	使用 CloudFormation 範本，使用上一個案例建立的 Micro Focus Enterprise Server ESCWA AMI 部署自動擴展群組。	雲端架構師

建立 AWS Systems Manager Automation Runbook

任務	Description	所需的技能
建立 Systems Manager Automation Runbook 的 CloudFormation 範本。	使用其他資訊區段中的範例程式碼片段來建立 CloudFormation 範本，該範本將建立	雲端架構師

任務	Description	所需的技能
	Systems Manager Automation Runbook 以自動化 PAC 建立、Enterprise Server 向內擴展和 Enterprise Server 向外擴展。	
部署包含 Systems Manager Automation Runbook 的 CloudFormation 堆疊。	使用 CloudFormation 範本部署堆疊，其中包含用於建立 PAC、企業伺服器向內擴展和企業伺服器向外擴展的 Automation Runbook。	雲端架構師

建立 Micro Focus Enterprise Server 的自動擴展群組

任務	Description	所需的技能
建立 CloudFormation 範本，以設定 Micro Focus Enterprise Server 的自動擴展群組。	使用 AWS 範例程式碼片段建立將建立自動擴展群組的 CloudFormation 範本。此範本將重複使用為 Micro Focus Enterprise Server ESCWA 執行個體建立的相同 AMI。 然後使用 AWS 範例程式碼片段來建立自動擴展生命週期事件，並設定 Amazon EventBridge 來篩選相同 CloudFormation 範本中的擴展和縮減事件。	雲端架構師
部署 Micro Focus Enterprise Servers 自動擴展群組的 CloudFormation 堆疊。	部署包含 Micro Focus Enterprise Servers 自動擴展群組的 CloudFormation 堆疊。	雲端架構師

相關資源

- Micro Focus Enterprise Server 效能和可用性叢集 (PAC)
- Amazon EC2 Auto Scaling lifecycle hooks
- 使用 EventBridge 搭配觸發執行自動化

其他資訊

下列案例必須自動化以擴展或擴展 PAC 叢集。

啟動或重新建立 PAC 的自動化

在 PAC 叢集開始時，Enterprise Server 需要 ESCWA 呼叫 APIs來建立 PAC 組態。這會啟動 Enterprise Server 區域並將其新增至 PAC。若要建立或重新建立 PAC，請使用下列步驟：

1. 使用指定名稱在 ESCWA 中設定 PAC 橫向擴展儲存庫 (PSOR)。

```
POST /server/v1/config/groups/sors
```

2. 建立具有指定名稱的 PAC，並將 PSOR 連接到該 PAC。

```
POST /server/v1/config/groups/pacs
```

3. 如果這是您第一次設定 PAC，請設定區域資料庫和跨區域資料庫。

> **Note**
>
> 此步驟使用 SQL 查詢和 Micro Focus Enterprise Suite 命令列 dbhfhadmin 工具來建立資料庫並匯入初始資料。

4. 將 PAC 定義安裝到企業伺服器區域。

```
POST /server/v1/config/mfds
POST /native/v1/config/groups/pacs/${pac_uid}/install
```

5. 在 PAC 中啟動企業伺服器區域。

```
POST /native/v1/regions/${host_ip}/${port}/${region_name}/start
```

您可以使用 Windows PowerShell 指令碼來實作先前的步驟。

下列步驟說明如何透過重複使用 Windows PowerShell 指令碼來建置建立 PAC 的自動化。

1. 建立 Amazon EC2 啟動範本，以下載或建立 WindowsPowerShell 指令碼做為引導程序的一部分。例如，您可以使用 EC2 使用者資料從 Amazon Simple Storage Service (Amazon S3) 儲存貯體下載指令碼。
2. 建立 AWS Systems Manager Automation Runbook 以叫用 Windows PowerShell 指令碼。
3. 使用執行個體標籤將 Runbook 與 ESCWA 執行個體建立關聯。
4. 使用啟動範本建立 ESCWA 自動擴展群組。

您可以使用下列範例 AWS CloudFormation 程式碼片段來建立 Automation Runbook。

用於建立 PAC 的 Systems Manager Automation Runbook 的範例 CloudFormation 程式碼片段

```
PACInitDocument:
   Type: AWS::SSM::Document
   Properties:
     DocumentType: Command
     Content:
       schemaVersion: '2.2'
       description: Operation Runbook to create Enterprise Server PAC
       mainSteps:
       - action: aws:runPowerShellScript
         name: CreatePAC
         inputs:
           onFailure: Abort
           timeoutSeconds: "1200"
           runCommand:
           - |
             C:\Scripts\PAC-Init.ps1
PacInitAutomation:
   Type: AWS::SSM::Document
   Properties:
     DocumentType: Automation
     Content:
       description: Prepare Micro Focus PAC Cluster via ESCWA Server
       schemaVersion: '0.3'
       assumeRole: !GetAtt SsmAssumeRole.Arn
       mainSteps:
         - name: RunPACInitDocument
```

```
            action: aws:runCommand
            timeoutSeconds: 300
            onFailure: Abort
            inputs:
              DocumentName: !Ref PACInitDocument
              Targets:
                - Key: tag:Enterprise Server - ESCWA
                  Values:
                    - "true"
  PacInitDocumentAssociation:
    Type: AWS::SSM::Association
    Properties:
      DocumentVersion: "$LATEST"
      Name: !Ref PACInitDocument
      Targets:
        - Key: tag:Enterprise Server - ESCWA
          Values:
            - "true"
```

如需詳細資訊，請參閱 Micro Focus Enterprise Server - 設定 PAC。

使用新的 Enterprise Server 執行個體進行擴展的自動化

向外擴展 Enterprise Server 執行個體時，必須將其 Enterprise Server 區域新增至 PAC。下列步驟說明如何叫用 ESCWA APIs，並將企業伺服器區域新增至 PAC。

1. 將 PAC 定義安裝到企業伺服器區域。

```
POST '/server/v1/config/mfds'
POST /native/v1/config/groups/pacs/${pac_uid}/install
```

2. 暖啟動 PAC 中的區域。

```
POST /native/v1/regions/${host_ip}/${port}/${region_name}/start
```

3. 透過將自動擴展群組與負載平衡器建立關聯，將 Enterprise Server 執行個體新增至負載平衡器。

您可以使用 Windows PowerShell 指令碼來實作先前的步驟。如需詳細資訊，請參閱 Micro Focus Enterprise Server - 設定 PAC。

下列步驟可用來建置事件驅動自動化，透過重複使用 Windows PowerShell 指令碼將新啟動的 Enterprise Server 執行個體新增至 PAC。

1. 為 Enterprise Server 執行個體建立 Amazon EC2 啟動範本，在 Enterprise Server 執行個體的引導期間佈建 Enterprise Server 區域。例如，您可以使用 Micro Focus Enterprise Server 命令 mfds 來匯入區域組態。如需此命令可用的更多詳細資訊和選項，請參閱企業伺服器參考。
2. 建立使用上一個步驟中建立的啟動範本的 Enterprise Server 自動擴展群組。
3. 建立 Systems Manager Automation Runbook 以叫用 Windows PowerShell 指令碼。
4. 使用執行個體標籤將 Runbook 與 ESCWA 執行個體建立關聯。
5. 建立 Amazon EventBridge 規則來篩選 Enterprise Server 自動擴展群組的 EC2 執行個體啟動成功事件，並建立目標以使用 Automation Runbook。

您可以使用下列範例 CloudFormation 程式碼片段來建立 Automation Runbook 和 EventBridge 規則。

用於擴展 Enterprise Server 執行個體之 Systems Manager 的範例 CloudFormation 程式碼片段

```
 ScaleOutDocument:
    Type: AWS::SSM::Document
    Properties:
      DocumentType: Command
      Content:
        schemaVersion: '2.2'
        description: Operation Runbook to Adding MFDS Server into an existing PAC
        parameters:
          MfdsPort:
            type: String
          InstanceIpAddress:
            type: String
            default: "Not-Available"
          InstanceId:
            type: String
            default: "Not-Available"
        mainSteps:
        - action: aws:runPowerShellScript
          name: Add_MFDS
          inputs:
            onFailure: Abort
            timeoutSeconds: "300"
            runCommand:
            - |
              $ip = "{{InstanceIpAddress}}"
              if ( ${ip} -eq "Not-Available" ) {
                $ip = aws ec2 describe-instances --instance-id {{InstanceId}} --output
text --query "Reservations[0].Instances[0].PrivateIpAddress"
```

```
              }
              C:\Scripts\Scale-Out.ps1 -host_ip ${ip} -port {{MfdsPort}}

  PacScaleOutAutomation:
    Type: AWS::SSM::Document
    Properties:
      DocumentType: Automation
      Content:
        parameters:
          MfdsPort:
            type: String
          InstanceIpAddress:
            type: String
            default: "Not-Available"
          InstanceId:
            type: String
            default: "Not-Available"
        description: Scale Out 1 New Server in Micro Focus PAC Cluster via ESCWA
Server
        schemaVersion: '0.3'
        assumeRole: !GetAtt SsmAssumeRole.Arn
        mainSteps:
          - name: RunScaleOutCommand
            action: aws:runCommand
            timeoutSeconds: 300
            onFailure: Abort
            inputs:
              DocumentName: !Ref ScaleOutDocument
              Parameters:
                InstanceIpAddress: "{{InstanceIpAddress}}"
                InstanceId: "{{InstanceId}}"
                MfdsPort: "{{MfdsPort}}"
              Targets:
                - Key: tag:Enterprise Server - ESCWA
                  Values:
                    - "true"
```

在 Enterprise Server 執行個體中擴展的自動化

與向外擴展類似，在向內擴展 Enterprise Server 執行個體時，會啟動事件 EC2 執行個體終止生命週期動作，並且需要下列程序和 API 呼叫，才能從 PAC 中移除 Micro Focus Enterprise Server 執行個體。

1. 在終止 Enterprise Server 執行個體中停止 區域。

```
POST "/native/v1/regions/${host_ip}/${port}/${region_name}/stop"
```

2. 從 PAC 移除企業伺服器執行個體。

```
DELETE "/server/v1/config/mfds/${uid}"
```

3. 傳送訊號以繼續終止 Enterprise Server 執行個體。

先前的步驟可以在 Windows PowerShell 指令碼中實作。如需此程序的其他詳細資訊，請參閱 Micro Focus Enterprise Server 文件 - 管理 PAC。

下列步驟說明如何建置事件驅動型自動化，透過重複使用 Windows PowerShell 指令碼從 PAC 終止 Enterprise Server 執行個體。

1. 建立 Systems Manager Automation Runbook 以叫用 Windows PowerShell 指令碼。
2. 使用執行個體標籤將 Runbook 與 ESCWA 執行個體建立關聯。
3. 建立 EC2 執行個體終止的自動擴展群組生命週期關聯。
4. 建立 Amazon EventBridge 規則來篩選 Enterprise Server 自動擴展群組的 EC2 執行個體終止生命週期動作事件，並建立目標以使用 Automation Runbook。

您可以使用下列範例 CloudFormation 範本來建立 Systems Manager Automation Runbook、 lifecycle hook 和 EventBridge 規則。

用於在 Enterprise Server 執行個體中擴展的 Systems Manager Automation Runbook 的範例 CloudFormation 程式碼片段

```
ScaleInDocument:
   Type: AWS::SSM::Document
   Properties:
     DocumentType: Command
     Content:
       schemaVersion: '2.2'
       description: Operation Runbook to Remove MFDS Server from PAC
       parameters:
         MfdsPort:
           type: String
         InstanceIpAddress:
           type: String
```

```
            default: "Not-Available"
          InstanceId:
            type: String
            default: "Not-Available"
        mainSteps:
        - action: aws:runPowerShellScript
          name: Remove_MFDS
          inputs:
            onFailure: Abort
            runCommand:
            - |
              $ip = "{{InstanceIpAddress}}"
              if ( ${ip} -eq "Not-Available" ) {
                $ip = aws ec2 describe-instances --instance-id {{InstanceId}} --output
text --query "Reservations[0].Instances[0].PrivateIpAddress"
              }
              C:\Scripts\Scale-In.ps1 -host_ip ${ip} -port {{MfdsPort}}

  PacScaleInAutomation:
    Type: AWS::SSM::Document
    Properties:
      DocumentType: Automation
      Content:
        parameters:
          MfdsPort:
            type: String
          InstanceIpAddress:
            type: String
            default: "Not-Available"
          InstanceId:
            type: String
            default: "Not-Available"
        description: Scale In 1 New Server in Micro Focus PAC Cluster via ESCWA Server
        schemaVersion: '0.3'
        assumeRole: !GetAtt SsmAssumeRole.Arn
        mainSteps:
          - name: RunScaleInCommand
            action: aws:runCommand
            timeoutSeconds: "600"
            onFailure: Abort
            inputs:
              DocumentName: !Ref ScaleInDocument
              Parameters:
                InstanceIpAddress: "{{InstanceIpAddress}}"
```

```
              MfdsPort: "{{MfdsPort}}"
              InstanceId: "{{InstanceId}}"
            Targets:
              - Key: tag:Enterprise Server - ESCWA
                Values:
                  - "true"
        - name: TerminateTheInstance
          action: aws:executeAwsApi
          inputs:
            Service: autoscaling
            Api: CompleteLifecycleAction
            AutoScalingGroupName: !Ref AutoScalingGroup
            InstanceId: "{{ InstanceId }}"
            LifecycleActionResult: CONTINUE
            LifecycleHookName: !Ref ScaleInLifeCycleHook
```

Amazon EC2 自動擴展觸發程序的自動化

為 Enterprise Server 執行個體設定擴展政策的程序需要了解應用程式行為。在大多數情況下，您可以設定目標追蹤擴展政策。例如，您可以使用平均 CPU 使用率做為 Amazon CloudWatch 指標，來設定自動擴展政策。如需詳細資訊，請參閱 Amazon EC2 Auto Scaling 的目標追蹤擴展政策。對於具有一般流量模式的應用程式，請考慮使用預測擴展政策。如需詳細資訊，請參閱 Amazon EC2 Auto Scaling 的預測擴展。

在 AWS 雲端中建置進階大型主機檔案檢視器

Bopath GOPALSAMY 和 Jeremiah O'Connor，Amazon Web Services

總結

此模式提供程式碼範例和步驟，協助您建置進階工具，以使用 AWS 無伺服器服務來瀏覽和檢閱大型主機固定格式檔案。模式提供範例，說明如何將大型主機輸入檔案轉換為 Amazon OpenSearch Service 文件以進行瀏覽和搜尋。檔案檢視器工具可協助您達成下列目標：

- 保留相同的大型主機檔案結構和配置，以在 AWS 目標遷移環境中保持一致性（例如，您可以在將檔案傳輸至外部各方的批次應用程式中維護相同的檔案配置）
- 在大型主機遷移期間加速開發和測試
- 支援遷移後的維護活動

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 虛擬私有雲端 (VPC)，具有傳統平台可存取的子網路
-

> **Note**
>
> 輸入檔案及其對應的常見業務導向語言 (COBOL) 複製手冊（：如需輸入檔案和 COBOL 複製手冊範例，請參閱 GitHub 儲存庫上的 gfs-mainframe-solutions。 如需 COBOL 複本手冊的詳細資訊，請參閱 IBM 網站上的適用於 z/OS 6.3 的企業 COBOL 程式設計指南。）

限制

- 複製手冊剖析限制為不超過兩個巢狀層級 (OCCURS)

Architecture

來源技術堆疊

- FB（固定封鎖）格式的輸入檔案

- COBOL 複製手冊配置

目標技術堆疊

- Amazon Athena
- Amazon OpenSearch Service
- Amazon Simple Storage Service (Amazon S3)
- AWS Lambda
- AWS Step Functions

目標架構

下圖顯示剖析大型主機輸入檔案並將其轉換為 OpenSearch Service 文件以供瀏覽和搜尋的程序。

該圖顯示以下工作流程：

1. 管理員使用者或應用程式會將輸入檔案推送至一個 S3 儲存貯體，並將 COBOL 複製手冊推送至另一個 S3 儲存貯體。
2.
> **Note**
>
> 具有輸入檔案的 S3 儲存貯體會叫用 Lambda 函數，以啟動無伺服器 Step Functions 工作流程。：使用此模式使用 S3 事件觸發和 Lambda 函數來驅動 Step Functions 工作流程是選用的。此模式中的 GitHub 程式碼範例不包含這些服務的使用，但您可以根據您的需求使用這些服務。

3. Step Functions 工作流程會協調來自下列 Lambda 函數的所有批次程序：
   - `s3copybookparser.py` 函數會剖析複製手冊配置，並擷取欄位屬性、資料類型和位移（輸入資料處理時需要）。
   - `s3toathena.py` 函數會建立 Athena 資料表配置。Athena 會剖析`s3toathena.py`函數處理的輸入資料，並將資料轉換為 CSV 檔案。
   - `s3toelasticsearch.py` 函數會從 S3 儲存貯體擷取結果檔案，並將檔案推送至 OpenSearch Service。
4. 使用者使用 OpenSearch Service 存取 OpenSearch Dashboards，以各種資料表和資料欄格式擷取資料，然後針對索引資料執行查詢。

工具

AWS 服務

- Amazon Athena 是一種互動式查詢服務，可協助您使用標準 SQL 直接在 Amazon Simple Storage Service (Amazon S3) 中分析資料。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而不需要佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。在此模式中，您可以使用 Lambda 實作核心邏輯，例如剖析檔案、轉換資料，以及將資料載入 OpenSearch Service 以進行互動式檔案存取。
- Amazon OpenSearch Service 是一項受管服務，可協助您在 AWS 雲端中部署、操作和擴展 OpenSearch Service 叢集。在此模式中，您可以使用 OpenSearch Service 為轉換的檔案編製索引，並為使用者提供互動式搜尋功能。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 Lambda 函數和其他 AWS 服務來建置業務關鍵應用程式。在此模式中，您可以使用 Step Functions 來協調 Lambda 函數。

其他工具

- GitHub 是一種程式碼託管服務，可提供協作工具和版本控制。
- Python 是一種高階程式設計語言。

Code

此模式的程式碼可在 GitHub gfs-mainframe-patterns 儲存庫中使用。

史詩

準備目標環境

任務	Description	所需的技能
建立 S3 儲存貯體。	建立 anS3 儲存貯體以存放複本、輸入檔案和輸出檔案。建議您的 S3 儲存貯體使用下列資料夾結構： • copybook/ • input/ • output/ • query/ • results/	一般 AWS
建立 s3copybookparser 函數。	1. 建立名為 的 Lambda 函數，並從 GitHub 儲存庫上傳原始程式碼 (s3copybookparser.py 和 copybook.py)。s3copybookparser 2. 將 IAM 政策S3ReadOnly 連接至 Lambda 函數。	一般 AWS
建立 s3toathena 函數。	1. 建立名為 的 Lambda 函數，s3toathena 並從 GitHub 儲存庫上傳原始程式碼 (s3toathena.py)。將 Lambda 逾時設定為 > 60 秒。 2. 若要提供必要資源的存取權，請將 IAM 政策和 AmazonAthenaFullAc	一般 AWS

任務	Description	所需的技能
	cess S3FullAccess 連接到 Lambda 函數。	
建立 s3toelasticsearch 函數。	1. **Important** 將 Python 相依性新增至 Lambda 環境。：若要使用s3toelasticsearch 函數，您必須新增 Python 相依性，因為 Lambda 函數使用 Python Elasticsearch 用戶端相依性 (Elasticsearch==7.9.0 和 requests_aws4auth)。 2. 建立名為 的 Lambda 函數，s3toelasticsearch 並從 GitHub 儲存庫上傳原始程式碼 (s3toelasticsearch.py)。 3. 將 Python 相依性匯入為 Lambda 層。 4. 將 IAM 政策和 S3ReadOnly AmazonOpenSearchServiceReadOnlyAccess 連接到 Lambda 函數。	一般 AWS

任務	Description	所需的技能
建立 OpenSearch Service 叢集。	建立叢集 1. 建立 OpenSearch Service 叢集。當您建立叢集時，請執行下列動作： • **Note** 為可用於登入 OpenSearch Dashboards 的叢集建立主要使用者和密碼。：如果您透過 Amazon Cognito 使用身分驗證，則不需要此步驟。 • 選擇精細存取控制。這可讓您在 OpenSearch Service 中控制對資料的存取。 2. 複製網域 URL，並將其做為環境變數「HOST」傳遞至 Lambda 函數 `s3toelasticsearch` 。 授予 IAM 角色的存取權 若要提供 Lambda 函數 IAM 角色 (`arn:aws:iam::**:role/service-role/s3toelasticsearch-role-**`) 的精細存取權，請執行下列動作：	一般 AWS

任務	Description	所需的技能
	1. 以主要使用者身分登入 OpenSearch Dashboards。 2. 選擇安全索引標籤，然後選擇角色、all_access、映射使用者、後端角色。 3. 新增 Lambda 函數 IAM 角色的 Amazon Resource Name (ARN)，然後選擇儲存。如需詳細資訊，請參閱 OpenSearch Service 文件中的將角色映射至使用者。	
建立用於協同運作的 Step Functions。	1. 使用標準流程建立 Step Functions 狀態機器。定義包含在 GitHub 儲存庫中。 2. 在 JSON 指令碼中，將 Lambda 函數的 ARNs 取代為環境中 Lambda 函數 ARNs。	一般 AWS

部署和執行

任務	Description	所需的技能
將輸入檔案和複製手冊上傳至 S3 儲存貯體。	從 GitHub 儲存庫範例資料夾下載範例檔案，並將檔案上傳至您先前建立的 S3 儲存貯體。 1. 將 `acctix.cpy` `Mockedcopy.cpy` 和 上傳至 `<S3_Bucket>/copybook` 資料夾。 2. 將 `Modedupdate.txt` 和 `acctindex.cpy` 範例輸	一般 AWS

任務	Description	所需的技能
	入檔案上傳至 <S3_Bucket>/input 資料夾。	

任務	Description	所需的技能

叫用 Step Functions。	1. 登入 AWS 管理主控台並開啟 Step Functions 主控台。
	2. 在導覽窗格中，選擇狀態機器。
	3. 選擇您的狀態機器，然後選擇開始執行。
	4. 在輸入方塊中，輸入下列複本/檔案路徑做為 S3 儲存貯體的 JSON 變數，然後選擇開始執行。
	
	`{ "s3_copybook_bucket_name": "<BUCKET NAME>", "s3_copybook_bucket_key": "<COPYBOOK PATH>", "s3_source_bucket_name": "<BUCKET NAME", "s3_source_bucket_key": "INPUT FILE PATH" }`
	
	例如：
	
	`{ "s3_copybook_bucket_name": "fileaidtest", "s3_copybook_bucket_key": "copybook/acctix.cpy", "s3_source_bucket_name": "fileaidtest",`
		一般 AWS

任務	Description	所需的技能

	<pre>"s3_source_bucket_key": "input/actindex"
	}</pre>	

驗證 Step Functions 中的工作流程執行。	在 Step Functions 主控台中，檢閱圖形檢查器中的工作流程執行。執行執行狀態會以顏色編碼來表示執行狀態。例如，藍色表示進行中，綠色表示成功，紅色表示失敗。您也可以檢閱執行事件歷史記錄區段中的資料表，以取得執行事件的詳細資訊。 如需圖形工作流程執行的範例，請參閱此模式額外資訊區段中的 Step Functions 圖形。	一般 AWS
驗證 Amazon CloudWatch 中的交付日誌。	1. 登入 AWS 管理主控台並開啟 CloudWatch 主控台。 2. 在導覽窗格中，展開日誌，然後選擇日誌群組。 3. 在搜尋方塊中，搜尋 `s3toelasticsearch` 函數的日誌群組。 如需成功交付日誌的範例，請參閱此模式額外資訊區段中的 CloudWatch 交付日誌。	一般 AWS

任務	Description	所需的技能
驗證 OpenSearch Dashboards 中的格式化檔案，並執行檔案操作。	1. 登入 AWS 管理主控台。在 Analytics (分析) 下，選擇 Amazon OpenSearch Service。 2. 在導覽窗格中，選擇網域。 3. 在搜尋方塊中，在 OpenSearch Dashboards 中輸入網域的 URL。 4. 選擇您的儀表板，然後以主要使用者身分登入。 5. 以資料表格式瀏覽索引資料。 6. 比較輸入檔案與 OpenSearch Dashboards 中的格式化輸出檔案（索引文件）。儀表板檢視會顯示您格式化檔案新增的資料欄標頭。確認來自未格式化輸入檔案的來源資料符合儀表板檢視中的目標資料。 7. 針對索引檔案執行搜尋（例如，使用欄位名稱、值或表達式）、篩選條件和 DQL（儀表板查詢語言）操作等動作。	一般 AWS

相關資源

參考

- COBOL 複製手冊範例 (IBM 文件)
- BMC Compuware File-AID (BMC 文件)

教學課程

- 教學課程：使用 Amazon S3 觸發程序來叫用 Lambda 函數 (AWS Lambda 文件）
- 如何使用 AWS Step Functions 和 AWS Lambda 建立無伺服器工作流程 (AWS 文件）
- 搭配 Amazon OpenSearch Service 使用 OpenSearch Dashboards (AWS 文件）

其他資訊

Step Functions 圖形

下列範例顯示 Step Functions 圖形。此圖表顯示此模式中所用 Lambda 函數的執行執行狀態。

CloudWatch 交付日誌

下列範例顯示執行成功交付日誌`s3toelasticsearch`。

2022-08-10T15：53：33.033 年 5 月 00 日	處理文件數量：100	
	2022-08-10T15：53：33.171 年 5 月 00 日	【INFO】 2022-08-10T20：53：33.171Z a1b2c3d4-5678-90ab-cdef-EXAMPLE11111POST https://search-essearch-3h4uqclifeqaj2vg4mphe7ffle.us-east-2.es.amazonaws.com:443/_bulk 【status：200 request：0.100s】
	2022-08-10T15：53：33.172 年 5 月 00 日	大量寫入成功：100 個文件

容器化已由 Blu Age 現代化的大型主機工作負載

Richard Milner-Watts，Amazon Web Services

總結

此模式提供範例容器環境，用於執行已使用 Blu Age 工具進行現代化的大型主機工作負載。Blu Age 會將舊版大型主機工作負載轉換為現代 Java 程式碼。此模式提供 Java 應用程式的包裝函式，因此您可以使用 Amazon Elastic Container Service (Amazon ECS) 或 Amazon Elastic Kubernetes Service (Amazon EKS) 等容器協同運作服務來執行它。

如需使用 Blu Age 和 AWS 服務來現代化工作負載的詳細資訊，請參閱下列 AWS 規範指引出版物：

- 在無伺服器 AWS 基礎設施上執行現代化 Blu Age 大型主機工作負載
- 使用 Terraform 部署容器化 Blu Age 應用程式的環境

如需使用 Blu Age 來現代化大型主機工作負載的協助，請在 Blu Age 網站上選擇聯絡我們的專家，以聯絡 Blu Age 團隊。如需協助將現代化工作負載遷移至 AWS、將它們與 AWS 服務整合，並將它們移至生產環境，請聯絡您的 AWS 客戶經理或填寫 AWS Professional Services 表單。

先決條件和限制

先決條件

- 由 Blu Age 建立的現代化 Java 應用程式。基於測試目的，此模式提供範例 Java 應用程式，可用來做為概念驗證。
- 您可以使用 Docker 環境來建置容器。

限制

根據您使用的容器協同運作平台，可供容器使用的資源（例如 CPU、RAM 和儲存）可能會受到限制。例如，如果您使用 Amazon ECS 搭配 AWS Fargate，請參閱 Amazon ECS 文件以了解限制和考量事項。

Architecture

來源技術堆疊

- 藍齡

- Java

目標技術堆疊

- Docker

目標架構

下圖顯示 Docker 容器內 Blu Age 應用程式的架構。

1. 容器的進入點是包裝函式指令碼。此 bash 指令碼負責準備 Blu Age 應用程式和處理輸出的執行期環境。
2. 容器內的環境變數用於設定包裝函式指令碼中的變數，例如 Amazon Simple Storage Service (Amazon S3) 儲存貯體名稱和資料庫登入資料。環境變數是由 AWS Secrets Manager 或 Parameter Store 提供，AWS Systems Manager 的功能。如果您使用 Amazon ECS 做為容器協同運作服務，您也可以硬式編碼 Amazon ECS 任務定義中的環境變數。
3. 在您執行 Blu Age 應用程式之前，包裝函式指令碼負責將任何輸入檔案從 S3 儲存貯體提取至容器。AWS Command Line Interface (AWS CLI) 安裝在容器內。這提供了一種機制，可透過閘道虛擬私有雲端 (VPC) 端點存取存放在 Amazon S3 中的物件。
4. Blu Age 應用程式的 Java Archive (JAR) 檔案可能需要與其他資料來源通訊，例如 Amazon Aurora。
5. 完成後，包裝函式指令碼會將產生的輸出檔案傳送到 S3 儲存貯體，以供進一步處理 （例如，Amazon CloudWatch 記錄服務）。如果您使用標準 CloudWatch 記錄的替代方案， 模式也支援將壓縮日誌檔案交付至 Amazon S3。

工具

AWS 服務

- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- Amazon Elastic Container Service (Amazon ECS) 是快速、可擴展的容器管理服務，可協助您執行、停止和管理叢集上的容器。

工具

- Docker 是一種軟體平台，用於建置、測試和部署應用程式。Docker 會將軟體封裝至稱為容器的標準化單位，其中包含軟體執行所需的一切，包括程式庫、系統工具、程式碼和執行時間。您可以使用 Docker 將應用程式部署並擴展到任何環境。
- Bash 是 GNU 作業系統的命令語言界面（殼層）。
- Java 是此模式中使用的程式設計語言和開發環境。
- Blu Age 是一種 AWS 大型主機現代化工具，可將舊版大型主機工作負載，包括應用程式碼、相依性和基礎設施，轉換為雲端的現代工作負載。

程式碼儲存庫

此模式的程式碼可在 GitHub Blu Age 範例容器儲存庫中找到。

最佳實務

- 使用環境變數將變數外部化，以變更應用程式的行為。這些變數可讓容器協同運作解決方案變更執行時間環境，而無需重建容器。此模式包含可用於 Blu Age 應用程式的環境變數範例。
- 執行 Blu Age 應用程式之前，請先驗證任何應用程式相依性。例如，確認資料庫可用且登入資料有效。在包裝函式指令碼中寫入測試，以驗證相依性，如果不符合，則提早失敗。
- 在包裝函式指令碼中使用詳細記錄。直接與執行中的容器互動可能具有挑戰性，具體取決於協同運作平台和任務需要多長時間。請務必將有用的輸出寫入 STDOUT，以協助診斷任何問題。例如，輸出可能會在您執行應用程式之前和之後包含應用程式工作目錄的內容。

史詩

取得 Blu Age 應用程式 JAR 檔案

任務	Description	所需的技能
選項 1 - 使用 Blu Age 來取得應用程式的 JAR 檔案。	此模式中的容器需要 Blu Age 應用程式。或者，您可以使用此模式隨附的範例 Java 應用程式進行原型。 與 Blu Age 團隊合作，為您的應用程式取得可製作到容器中的 JAR 檔案。如果 JAR 檔案	雲端架構師

任務	Description	所需的技能
	無法使用，請參閱下一個任務以改用範例應用程式。	
選項 2 - 建置或使用提供的範例應用程式 JAR 檔案。	此模式提供預先建置的範例 JAR 檔案。此檔案會在休眠 30 秒並結束STDOUT之前，將應用程式的環境變數輸出至 。 此檔案名為 `bluAgeSample.jar` ，位於 GitHub 儲存庫的 docker 資料夾中。 如果您想要更改程式碼並建置自己的 JAR 檔案版本，請使用位於 GitHub 儲存庫中 ./java_sample/src/sample_java_app.java 的原始程式碼。您可以使用位於 ./java_sample/build.sh 的建置指令碼來編譯 Java 來源並建置新的 JAR fie。	應用程式開發人員

建置 Blu Age 容器

任務	Description	所需的技能
複製 GitHub 儲存庫。	使用 命令複製範本程式碼儲存庫： `git clone https://github.com/aws-samples/aws-blu-age-sample-container`	AWS DevOps

任務	Description	所需的技能
使用 Docker 建置容器。	使用 Docker 建置容器，然後再將其推送至 Docker 登錄檔，例如 Amazon ECR： 1. 從您選擇的終端機，導覽至本機 GitHub 儲存庫中的 `docker` 資料夾。 2. 使用此命令來建置容器： `docker build -t <tag> .` 其中 `<tag>` 是您想要使用的容器名稱。	AWS DevOps
測試 Blu Age 容器。	（選用）如有必要，請使用 命令在本機測試容器： `docker run -it <tag> /bin/bash`	AWS DevOps

任務	Description	所需的技能
驗證至您的 Docker 儲存庫。	如果您打算使用 Amazon ECR，請遵循 Amazon ECR 文件中的指示來安裝和設定 AWS CLI，並將 Docker CLI 驗證為您的預設登錄檔。 我們建議您使用 get-login-password 命令進行身分驗證。 **Note** 如果您使用檢視推送命令按鈕，Amazon ECR 主控台會提供此命令的預先填入版本。如需詳細資訊，請參閱 Amazon ECR 文件。 `aws ecr get-login -password --region <region> | docker login --username AWS --password-stdin <account>.dkr.ecr.<region>.amazonaws.com` 如果您不打算使用 Amazon ECR，請遵循為容器登錄系統提供的指示。	AWS DevOps

任務	Description	所需的技能
建立容器儲存庫。	在 Amazon ECR 中建立儲存庫。如需說明，請參閱 模式使用 Terraform 部署容器化 Blu Age 應用程式的環境。 如果您使用的是另一個容器登錄系統，請遵循該系統提供的指示。	AWS DevOps

標記您的容器並將其推送至目標儲存庫。	如果您使用的是 Amazon ECR：
	
	1. 使用 Amazon ECR 登錄檔和儲存庫標記本機 Docker 映像，以便您可以將其推送到遠端儲存庫：
	
	`docker tag <tag>:latest <account>.dkr.ecr.<region>.amazonaws.com/<repository>:<versionNumber>`
	
	2. 將映像推送至遠端儲存庫：
	
	`docker push <account>.dkr.ecr.<region>.amazonaws.com/<repository>:<versionNumber>`
	
	如需詳細資訊，請參閱《Amazon ECR 使用者指南》中的推送 Docker 映像。
		AWS DevOps

相關資源

AWS 資源

- AWS Blu Age 範例容器儲存庫
- 在無伺服器 AWS 基礎設施上執行現代化 Blu Age 大型主機工作負載
- 使用 Terraform 部署容器化 Blu Age 應用程式的環境
- 搭配 AWS CLI 使用 Amazon ECR (Amazon ECR 使用者指南)
- 私有登錄檔身分驗證 (Amazon ECR 使用者指南)
- Amazon ECS 文件
- Amazon EKS 文件

其他資源

- Blu Age 網站
- Docker 網站

使用 Python 將 EBCDIC 資料轉換為 AWS 上的 ASCII

Luis Gustavo Dantas，Amazon Web Services

總結

由於大型主機通常會託管關鍵業務資料，因此將資料遷移至 Amazon Web Services (AWS) 雲端或其他美國資訊交換標準碼 (ASCII) 環境時，現代化資料是最重要的任務之一。在大型主機上，資料通常會以延伸二進位編碼的小數交換碼 (EBCDIC) 格式進行編碼。匯出資料庫、虛擬儲存存取方法 (VSAM) 或一般檔案通常會產生封裝的二進位 EBCDIC 檔案，這些檔案更複雜的遷移。最常用的資料庫遷移解決方案是變更資料擷取 (CDC)，在大多數情況下會自動轉換資料編碼。不過，CDC 機制可能無法用於這些資料庫、VSAM 或一般檔案。對於這些檔案，需要替代方法來現代化資料。

此模式說明如何透過將其轉換為 ASCII 格式來現代化 EBCDIC 資料。轉換後，您可以將資料載入分散式資料庫，或讓雲端中的應用程式直接處理資料。模式會使用mainframe-data-utilities GitHub 儲存庫中的轉換指令碼和範例檔案。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- EBCDIC 輸入檔案及其對應的常見商業導向語言 (COBOL) 複製本。範例 EBCDIC 檔案和 COBOL 複製手冊包含在mainframe-data-utilities GitHub 儲存庫中。如需 COBOL 複本手冊的詳細資訊，請參閱 IBM 網站上的適用於 z/OS 6.4 的企業 COBOL 程式設計指南。

限制

- 不支援 COBOL 程式中定義的檔案配置。它們必須單獨提供。

產品版本

- Python 3.8 版或更新版本

Architecture

來源技術堆疊

- 大型主機上的 EBCDIC 資料

- COBOL 複製手冊

目標技術堆疊

- 虛擬私有雲端 (VPC) 中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體
- Amazon Elastic Block Store (Amazon EBS)
- Python 及其所需的套件、JavaScript 物件標記法 (JSON)、sys 和日期時間
- ASCII 平面檔案已準備好供現代應用程式讀取或載入關聯式資料庫資料表

目標架構

架構圖顯示將 EBCDIC 檔案轉換為 EC2 執行個體上 ASCII 檔案的程序：

1. 使用 parse_copybook_to_json.py 指令碼，您可以將 COBOL 複製手冊轉換為 JSON 檔案。
2. 使用 JSON 檔案和 extract_ebcdic_to_ascii.py 指令碼，您可以將 EBCDIC 資料轉換為 ASCII 檔案。

自動化和擴展

在第一個手動檔案轉換所需的資源就緒之後，您就可以自動化檔案轉換。此模式不包含自動化的指示。有多種方法可自動化轉換。以下是一種可能方法的概觀：

1. 將 AWS Command Line Interface (AWS CLI) 和 Python 指令碼命令封裝為 shell 指令碼。
2. 建立 AWS Lambda 函數，以非同步方式將 shell 指令碼任務提交至 EC2 執行個體。如需詳細資訊，請參閱使用 AWS Lambda 排程 SSH 任務。
3. 建立 Amazon Simple Storage Service (Amazon S3) 觸發程序，在每次上傳舊版檔案時叫用 Lambda 函數。如需詳細資訊，請參閱使用 Amazon S3 觸發來叫用 Lambda 函數。

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，並快速向上或向下擴展。

- Amazon Elastic Block Store (Amazon EBS) 提供區塊層級儲存磁碟區，可與 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體搭配使用。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。

其他工具

- GitHub 是一種程式碼託管服務，可提供協作工具和版本控制。
- Python 是一種高階程式設計語言。

程式碼儲存庫

此模式的程式碼可在mainframe-data-utilities GitHub 儲存庫中使用。

史詩

準備 EC2 執行個體

任務	Description	所需的技能
啟動 EC2 執行個體。	EC2 執行個體必須具有傳出網際網路存取。這可讓執行個體存取 GitHub 上可用的 Python 原始程式碼。若要建立執行個體： 1. 在 https：//https://console.aws.amazon.com/ec2 開啟 Amazon EC2 主控台。 2. 啟動 EC2 Linux 執行個體。使用公有 IP 地址，並允許透過連接埠 22 進行傳入存取。確定執行個體的儲存體大小至少是 EBCDIC 資料檔	一般 AWS

任務	Description	所需的技能
	案大小的兩倍。如需說明，請參閱 Amazon EC2 文件。	

安裝 Git。	1. 使用安全 shell (SSH) 用戶端，連線到您剛啟動的 EC2 執行個體。如需詳細資訊，請參閱連線至 Linux 執行個體。
	2. 在 Amazon EC2 主控台中，執行下列命令。這會在 EC2 執行個體上安裝 Git。
	`sudo yum install git`
	3. 執行下列命令並確認 Git 已成功安裝。
	`git --version`
		一般 AWS、Linux

任務	Description	所需的技能

安裝 Python。	1. 在 Amazon EC2 主控台中，執行下列命令。這會在 EC2 執行個體上安裝 Python。
	`sudo yum install python3`
	2. 在 Amazon EC2 主控台中，執行下列命令。這會在 EC2 執行個體上安裝 Pip3。
	`sudo yum install python3-pip`
	3. 在 Amazon EC2 主控台中，執行下列命令。這會在 EC2 執行個體上安裝適用於 Python (Boto3) 的 AWS 開發套件。
	`sudo pip3 install boto3`
	4. 在 Amazon EC2 主控台中，執行下列命令，其中 <us-east-1> 是您 AWS 區域的程式碼。如需區域代碼的完整清單，請參閱 Amazon EC2 文件中的可用區域。
	`export AWS_DEFAULT_REGION=<us-east-1>`
		一般 AWS、Linux

任務	Description	所需的技能
複製 GitHub 儲存庫。	1. 在 Amazon EC2 主控台中，執行下列命令。這會從 GitHub mainframe-data-utilities儲存庫，並開啟預設複製位置 home 資料夾。 `git clone https://github.com/aws-samples/mainframe-data-utilities.git` 2. 在 home 資料夾中，確認`mainframe-data-utilities` 資料夾存在。	一般 AWS、GitHub

從 EBCDIC 資料建立 ASCII 檔案

任務	Description	所需的技能
將 COBOL 複製手冊剖析到 JSON 配置檔案中。	在 `mainframe-data-utilities` 資料夾內，執行 parse_copybook_to_json.py 指令碼。此自動化模組會從 COBOL 複製手冊讀取檔案配置，並建立 JSON 檔案。JSON 檔案包含從來源檔案解譯和擷取資料所需的資訊。這會從 COBOL 複製手冊建立 JSON 中繼資料。 下列命令會將 COBOL 複製手冊轉換為 JSON 檔案。 `python3 parse_copybook_to_json.py \`	一般 AWS、Linux

任務	Description	所需的技能
	-copybook LegacyRef erence/COBPACK2.cpy \ -output sample-data/ cobpack2-list.json \ -dict sample-data/ cobpack2-dict.json \ -ebcdic sample-data/ COBPACK.OUTFILE.txt \ -ascii sample-data/ COBPACK.ASCII.txt \ -print 10000 指令碼會列印收到的引數。 ---------- ---------- ---------- ---------- Copybook file..... .....| LegacyRef erence/COBPACK2.cpy Parsed copybook (JSON List).| sample-data/ cobpack2-list.json JSON Dict (document ation)...| sample-da ta/cobpack2-dict.json ASCII file..... .....| sample- data/COBPACK.ASCII.t xt EBCDIC file..... .....| sample- data/COBPACK.OUTFILE .txt Print each..... .....| 10000 ---------- ----------	

任務	Description	所需的技能
	``` ------------------ -------------- ``` 如需引數的詳細資訊，請參閱 GitHub 儲存庫中的 README 檔案。	

任務	Description	所需的技能
檢查 JSON 配置檔案。	1. 導覽至 parse_copybook_to_json.py 指令碼中定義的輸出路徑。 2. 檢查 sample-data/cobpack2-list.json 檔案的建立時間，確認您已選取適當的 JSON 配置檔案。 3. 檢查 JSON 檔案並確認內容類似於以下內容。 `"input": "extract-ebcdic-to-ascii/COBPACK.OUTFILE.txt", "output": "extract-ebcdic-to-ascii/COBPACK.ASCII.txt", "max": 0, "skip": 0, "print": 10000, "lrecl": 150, "rem-low-values": true, "separator": "|", "transf": [ { "type": "ch", "bytes": 19, "name": "OUTFILE-TEXT" }` JSON 配置檔案最重要的屬性為： • input – 包含要轉換之 EBCDIC 檔案的路徑	一般 AWS、JSON

任務	Description	所需的技能
	• `output` – 定義產生 ASCII 檔案的路徑 • `lrecl` – 指定邏輯記錄長度的大小，以位元組為單位 • `transf` – 以位元組為單位列出所有欄位及其大小 如需 JSON 配置檔案的詳細資訊，請參閱 GitHub 儲存庫中的README 檔案。	

任務	Description	所需的技能
建立 ASCII 檔案。	執行 extract_ebcdic_to_ascii.py 指令碼，此指令碼包含在複製的 GitHub 儲存庫中。此指令碼會讀取 EBCDIC 檔案，並寫入已轉換且可讀取的 ASCII 檔案。 `python3 extract_ebcdic_to_ascii.py -local-json sample-data/cobpack2-list.json` 當指令碼處理 EBCDIC 資料時，它會列印每批次 10，000 筆記錄的訊息。請參閱以下範例。 `---------- 2023-05-15 21:21:46.322253 | Local Json file | -local-json | sample-data/cobpack2-list.json 2023-05-15 21:21:47.034556 | Records processed	10000 2023-05-15 21:21:47.736434	Records processed	20000 2023-05-15 21:21:48.441696	Records processed	30000 2023-05-15 21:21:49.173781	Records processed	40000`	一般 AWS

任務	Description	所需的技能
	2023-05-15 21:21:49. 874779 | Records processed | 50000 2023-05-15 21:21:50. 705873 | Records processed | 60000 2023-05-15 21:21:51. 609335 | Records processed | 70000 2023-05-15 21:21:52. 292989 | Records processed | 80000 2023-05-15 21:21:52. 938366 | Records processed | 89280 2023-05-15 21:21:52. 938448 Seconds 6.616232 如需如何變更列印頻率的資訊，請參閱 GitHub 儲存庫中的 README 檔案。	

任務	Description	所需的技能

檢查 ASCII 檔案。	1. 檢查 extract-ebcdic-to-ascii/COBPACK.ASCII.txt 檔案的建立時間，以確認其最近已建立。
	2. 在 Amazon EC2 主控台中，輸入下列命令。這會開啟 ASCII 檔案的第一個記錄。
	`head sample-data/COBPACK.ASCII.txt -n 1| xxd`
	3. 檢查第一個記錄的內容。由於 EBCDIC 檔案通常是二進位檔案，因此沒有歸位和換行 (CRLF) 特殊字元。extract_ebcdic_to_ascii.py 指令碼會將管道字元新增為資料欄分隔符號，該分隔符號在指令碼參數中定義。
	
	如果您使用提供的範例 EBCDIC 檔案，以下是 ASCII 檔案中的第一個記錄。
	`00000000: 2d30 3030 3030 3030 3030 3130 3030 3030  -000000000100000`
	`00000010: 3030 307c 3030 3030 3030 3030 3031 3030  000|0000000100`
	`00000020: 3030 3030 3030 7c2d 3030 3030 3030 3030  000000|-0000000`
		一般 AWS、Linux

任務	Description	所需的技能
	00000030: 3031 3030 3030 3030 3030 7c30 7c30 7c31 0100000000| 0|0|1 00000040: 3030 3030 3030 3030 7c2d 3130 3030 3030 00000000| -100000 00000050: 3030 307c 3130 3030 3030 3030 307c 2d31 000|10000 0000|-1 00000060: 3030 3030 3030 3030 7c30 3030 3030 7c30 00000000| 00000|0 00000070: 3030 3030 7c31 3030 3030 3030 3030 7c2d 0000|1000 00000|- 00000080: 3130 3030 3030 3030 307c 3030 3030 3030 100000000| 000000 00000090: 3030 3030 3130 3030 3030 3030 307c 2d30 000010000 0000|-0 000000a0: 3030 3030 3030 3030 3031 3030 3030 3030 000000000 1000000 000000b0: 3030 7c41 7c41 7c0a 00|A|A|.	

任務	Description	所需的技能
評估 EBCDIC 檔案。	在 Amazon EC2 主控台中，輸入下列命令。這會開啟 EBCDIC 檔案的第一個記錄。 `head sample-data/COBPACK.OUTFILE.txt -c 150 | xxd` 如果您使用範例 EBCDIC 檔案，結果如下。 `00000000: 60f0 f0f0 f0f0 f0f0 f0f0 f1f0 f0f0 f0f0 `........ ....` `00000010: f0f0 f0f0 f0f0 f0f0 f0f0 f0f0 f1f0 f0f0 ....` `00000020: f0f0 f0f0 f0f0 f0f0 f0f0 f0f0 f0f0 f1f0 ....` `00000030: f0f0 f0f0 f0f0 d000 0000 0005 f5e1 00fa ....` `00000040: 0a1f 0000 0000 0005 f5e1 00ff ffff fffa ....` `00000050: 0a1f 0000 000f 0000 0c10 0000 000f 1000 ....` `00000060: 0000 0d00 0000 0000 1000 0000 0f00 0000 ....`	一般 AWS、Linux、EBCDIC

任務	Description	所需的技能
	00000070: 0000 1000 0000 0dc1 c100 0000 0000 0000 .... 00000080: 0000 0000 0000 0000 0000 0000 0000 0000 .... 00000090: 0000 0000 0000 ...... 若要評估來源和目標檔案之間的等效性，需要 EBCDIC 的完整知識。例如，範例 EBCDIC 檔案的第一個字元是連字號 (-)。在 EBCDIC 檔案的十六進位表示法中，此字元以 表示60，而在 ASCII 檔案的十六進位表示法中，此字元以 表示2D。如需 EBCDIC-to-ASCII 轉換表，請參閱 IBM 網站上的 EBCDIC 至 ASCII。	

相關資源

參考

- EBCDIC 字元集 (IBM 文件)
- EBCDIC 到 ASCII (IBM 文件)
- COBOL (IBM 文件)
- 基本 JCL 概念 (IBM 文件)
- 連線至 Linux 執行個體 (Amazon EC2 文件)

教學課程

- 使用 AWS Lambda 排程 SSH 任務 (AWS 部落格文章)
- 使用 Amazon S3 觸發來叫用 Lambda 函數 (AWS Lambda 文件)

使用 Amazon S3 將大型主機檔案從 EBCDIC 格式轉換為字元分隔 ASCII 格式 AWS Lambda

Luis Gustavo Dantas，Amazon Web Services

總結

此模式說明如何啟動 AWS Lambda 函數，自動將大型主機延伸二進位編碼十進位交換碼 (EBCDIC) 檔案轉換為字元分隔的美國資訊交換標準碼 (ASCII) 檔案。Lambda 函數會在 ASCII 檔案上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體後執行。在檔案轉換之後，您可以讀取以 x86 為基礎的工作負載上的 ASCII 檔案，或將檔案載入現代資料庫。

此模式中示範的檔案轉換方法可協助您克服在現代環境中使用 EBCDIC 檔案的挑戰。以 EBCDIC 編碼的檔案通常包含以二進位或封裝小數格式表示的資料，欄位為固定長度。這些特性會產生障礙，因為現代 x86 型工作負載或分散式環境通常使用 ASCII 編碼資料，且無法處理 EBCDIC 檔案。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- Amazon S3 儲存貯體
- 具有管理許可的 AWS Identity and Access Management (IAM) 使用者
- AWS CloudShell
- Python 3.8.0 或更新版本
- 以 EBCDIC 及其對應資料結構編碼的平面檔案，採用常見的商業導向語言 (COBOL) 複製本

> **Note**
>
> 此模式使用範例 EBCDIC 檔案 (CLIENT.EBCDIC.txt) 及其對應的 COBOL 複製手冊 (COBKS05.cpy)。這兩個檔案都可在 GitHub mainframe-data-utilities儲存庫中使用。

限制

- COBOL 複製手冊通常包含多個配置定義。mainframe-data-utilities 專案可以剖析這種類型的複製手冊，但無法推斷要在資料轉換時考慮哪種配置。這是因為複製手冊不會保留此邏輯（而是保留在 COBOL 程式中）。因此，在剖析複製手冊之後，您必須手動設定用於選取配置的規則。

- 此模式受限於 Lambda 配額。

Architecture

來源技術堆疊

- IBM z/OS、IBM i 和其他 EBCDIC 系統
- 具有以 EBCDIC 編碼資料的序列檔案 （例如 IBM Db2 卸載）
- COBOL 複製手冊

目標技術堆疊

- Amazon S3
- Amazon S3 事件通知
- IAM
- Lambda 函式
- Python 3.8 或更高版本
- 大型主機資料公用程式
- JSON 中繼資料
- 字元分隔 ASCII 檔案

目標架構

下圖顯示將大型主機 EBCDIC 檔案轉換為 ASCII 檔案的架構。

該圖顯示以下工作流程：

1. 使用者執行複製手冊剖析器指令碼，將 COBOL 複製手冊轉換為 JSON 檔案。
2. 使用者將 JSON 中繼資料上傳至 Amazon S3 儲存貯體。這可讓資料轉換 Lambda 函數讀取中繼資料。
3. 使用者或自動化程序會將 EBCDIC 檔案上傳至 Amazon S3 儲存貯體。
4. Amazon S3 通知事件會觸發資料轉換 Lambda 函數。
5. AWS 驗證 Lambda 函數的 Amazon S3 儲存貯體讀寫許可。
6. Lambda 會從 Amazon S3 儲存貯體讀取檔案，並在本機將檔案從 EBCDIC 轉換為 ASCII。

7. Lambda 會在 Amazon CloudWatch 中記錄程序狀態。
8. Lambda 會將 ASCII 檔案寫回 Amazon S3。

> **Note**
>
> 複製手冊剖析器指令碼會執行一次，以將中繼資料轉換為 JSON 格式，隨後會存放在 Amazon S3 儲存貯體中。在初始轉換之後，在 Amazon S3 儲存貯體中參考相同 JSON 檔案的所有後續 EBCDIC 檔案將使用現有的中繼資料組態。

工具

AWS 服務

- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您在其上執行的應用程式。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS CloudShell 是一種瀏覽器型 shell，您可以使用 AWS Command Line Interface (AWS CLI) 和一系列預先安裝的開發工具 AWS 服務 來管理。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。Lambda 只會在需要時執行程式碼並自動擴展，因此您只需支付您使用的運算時間。

其他工具

- GitHub 是一種程式碼託管服務，可提供協作工具和版本控制。
- Python 是一種高階程式設計語言。

Code

此模式的程式碼可在 GitHub mainframe-data-utilities儲存庫中使用。

最佳實務

請考慮下列最佳實務：

- 在 Amazon Resource Name (ARN) 層級設定必要的許可。
- 一律授予 IAM 政策的最低權限許可。如需詳細資訊，請參閱 IAM 文件中的 IAM 安全最佳實務。

史詩

建立環境變數和工作資料夾

任務	Description	所需的技能

建立環境變數。	將下列環境變數複製到文字編輯器，然後將下列範例中<placeholder> 的值取代為您的資源值：
	
	`bucket=<your_bucket_name>`
	`account=<your_account_number>`
	`region=<your_region_code>`
	
	Note
	您將建立 Amazon S3 儲存貯體的參考 AWS 帳戶，以及 AWS 區域更新版本。
	
	若要定義環境變數，請開啟 CloudShell 主控台，然後將更新後的環境變數複製並貼到命令列。
		一般 AWS

任務	Description	所需的技能
	Note 每次 CloudShell 工作階段重新啟動時，您必須重複此步驟。	
建立工作資料夾。	若要在稍後簡化資源清除程序，請執行下列命令，在 CloudShell 中建立工作資料夾： `mkdir workdir; cd workdir` **Note** 每次失去與 CloudShell 工作階段的連線時，您必須將目錄變更為工作目錄 (`workdir`)。	一般 AWS

定義 IAM 角色和政策

任務	Description	所需的技能
建立 Lambda 函數的信任政策。	EBCDIC 轉換器會在 Lambda 函數中執行。函數必須具有 IAM 角色。在建立 IAM 角色之前，您必須定義信任政策文件，讓資源能夠擔任該政策。 從 CloudShell 工作資料夾中，執行下列命令來建立政策文件：	一般 AWS

任務	Description	所需的技能

	`E2ATrustPol=$(cat <<EOF`
	`{`
	`  "Version": "2012-10-17",`
	`  "Statement": [`
	`  {`
	`  "Effect": "Allow",`
	`  "Principal": {`
	`  "Service": "lambda.amazonaws.com"`
	`  },`
	`  "Action": "sts:AssumeRole"`
	`  }`
	`  ]`
	`}`
	`EOF`
	`)`
	`printf "$E2ATrustPol" > E2ATrustPol.json`	

建立用於 Lambda 轉換的 IAM 角色。	若要建立 IAM 角色，請從 CloudShell 工作資料夾執行下列 AWS CLI 命令： `aws iam create-role --role-name E2AConvLambdaRole --assume-role-policy-document file://E2ATrustPol.json`	一般 AWS

任務	Description	所需的技能

建立 Lambda 函數的 IAM 政策文件。	Lambda 函數必須具有 Amazon S3 儲存貯體的讀寫存取權，以及 Amazon CloudWatch Logs 的寫入許可。
	
	若要建立 IAM 政策，請從 CloudShell 工作資料夾執行下列命令：
	
	`E2APolicy=$(cat <<EOF`
	`{`
	`  "Version": "2012-10-17",`
	`  "Statement": [`
	`  {`
	`  "Sid": "Logs",`
	`  "Effect": "Allow",`
	`  "Action": [`
	`  "logs:PutLogEvents",`
	`  "logs:CreateLogStream",`
	`  "logs:CreateLogGroup"`
	`  ],`
	`  "Resource": [`
	`  "arn:aws:logs:*:*:log-group:*",`
	`  "arn:aws:logs:*:*:log-group:*:log-stream:*"`
	`  ]`
	`  },`
		一般 AWS

任務	Description	所需的技能

	<pre>{
	"Sid": "S3",
	"Effect":
	"Allow",
	"Action": [
	
	"s3:GetObject",
	
	"s3:PutObject",
	
	"s3:GetObjectVersion"
	],
	"Resource":
	[
	
	"arn:aws:s3:::%s/*",
	
	"arn:aws:s3:::%s"
	]
	}
	]
	}
	EOF
	)
	printf "$E2APolicy"
	"$bucket" "$bucket"
	> E2AConvLambdaPolic
	y.json</pre>	

將 IAM 政策文件連接至 IAM 角色。	若要將 IAM 政策連接至 IAM 角色，請從 CloudShell 工作資料夾輸入下列命令：
	
	<pre>aws iam put-role-policy
	--role-name E2AConvLa
	mbdaRole --policy-name
	E2AConvLambdaPolic
	y --policy-document
	file://E2AConvLamb
	daPolicy.json</pre>
		一般 AWS

建立用於 EBCDIC 轉換的 Lambda 函數

任務	Description	所需的技能
下載 EBCDIC 轉換原始程式碼。	從 CloudShell 工作資料夾中，執行下列命令，從 GitHub 下載 mainframe-data-utilities 原始碼： `git clone https://github.com/aws-samples/mainframe-data-utilities.git mdu`	一般 AWS
建立 ZIP 套件。	從 CloudShell 工作資料夾中，輸入下列命令來建立 ZIP 套件，以建立用於 EBCDIC 轉換的 Lambda 函數： `cd mdu; zip ../mdu.zip *.py; cd ..`	一般 AWS

建立 Lambda 函數。	從 CloudShell 工作資料夾中，輸入下列命令來建立用於 EBCDIC 轉換的 Lambda 函數：
	`aws lambda create-function \`
	`--function-name E2A \`
	`--runtime python3.9 \`
	`--zip-file fileb://mdu.zip \`
	`--handler extract_ebcdic_to_ascii.lambda_handler \`
	`--role arn:aws:iam::$account:role/E2AConvLambdaRole \`
		一般 AWS

任務	Description	所需的技能

	`--timeout 10 \`
	`--environment "Variable s={layout=$bucket/ layout/}"`
	
	Note
	環境變數配置會告知 Lambda 函數 JSON 中繼資料所在的位置。	

建立 Lambda 函數的資源型政策。	從 CloudShell 工作資料夾中，輸入下列命令，以允許 Amazon S3 事件通知觸發 Lambda 函數以進行 EBCDIC 轉換：
	
	`aws lambda add-permi ssion \`
	`--function-name E2A \`
	`--action lambda:In vokeFunction \`
	`--principal s3.amazon aws.com \`
	`--source-arn arn:aws:s 3:::$bucket \`
	`--source-account $account \`
	`--statement-id 1`
		一般 AWS

建立 Amazon S3 事件通知

任務	Description	所需的技能
建立 Amazon S3 事件通知的組態文件。	當檔案放置在輸入資料夾中時，Amazon S3 事件通知會啟	一般 AWS

任務	Description	所需的技能

	動 EBCDIC 轉換 Lambda 函數。
	
	從 CloudShell 工作資料夾中，執行下列命令來建立 Amazon S3 事件通知的 JSON 文件：
	
	`S3E2AEvent=$(cat <<EOF`
	`{`
	`"LambdaFunctionConfigurations": [`
	`{`
	`"Id": "E2A",`
	`"LambdaFunctionArn": "arn:aws:lambda:%s:%s:function:E2A",`
	`"Events": [ "s3:ObjectCreated:Put" ],`
	`"Filter": {`
	`"Key": {`
	`"FilterRules": [`
	`{`
	`"Name": "prefix",`
	`"Value": "input/"`
	`}`
	`]`
	`}`
	`}`
	`}`
	`]`
	`}`
	`EOF`
	`)`	

任務	Description	所需的技能
	`printf "$S3E2AEvent" "$region" "$account" > S3E2AEvent.json`	
建立 Amazon S3 事件通知。	從 CloudShell 工作資料夾中，輸入下列命令來建立 Amazon S3 事件通知： `aws s3api put-bucket-notification-configuration --bucket $bucket --notification-configuration file://S3E2AEvent.json`	一般 AWS

建立和上傳 JSON 中繼資料

任務	Description	所需的技能

剖析 COBOL 複製手冊。	從 CloudShell 工作資料夾中，輸入下列命令，將範例 COBOL 複製手冊剖析為 JSON 檔案（定義如何正確讀取和分割資料檔案）：
	`python3 mdu/parse_copybook_to_json.py \`
	`-copybook mdu/LegacyReference/COBKS05.cpy \`
	`-output CLIENT.json \`
	`-output-s3key CLIENT.ASCII.txt \`
	`-output-s3bkt $bucket \`
		一般 AWS

任務	Description	所需的技能
	`-output-type s3 \` `-print 25`	

新增轉換規則。	範例資料檔案及其對應的 COBOL 複製手冊是多配置檔案。這表示轉換必須根據特定規則分割資料。在此情況下，每一列位置 3 和 4 上的位元組會定義配置。
	
	從 CloudShell 工作資料夾中，編輯 `CLIENT.json` 檔案並將內容從 `"transf-rule": []`，變更為下列項目：
	
	`"transf-rule": [`
	`{`
	`"offset": 4,`
	`"size": 2,`
	`"hex": "0002",`
	`"transf": "transf1"`
	`},`
	`{`
	`"offset": 4,`
	`"size": 2,`
	`"hex": "0000",`
	`"transf": "transf2"`
	`}`
	`],`
		General AWS、IBM Mainframe、Cobol

任務	Description	所需的技能
將 JSON 中繼資料上傳至 Amazon S3 儲存貯體。	從 CloudShell 工作資料夾中，輸入下列 AWS CLI 命令，將 JSON 中繼資料上傳至您的 Amazon S3 儲存貯體： `aws s3 cp CLIENT.json s3://$bucket/layout/CLIENT.json`	一般 AWS

轉換 EBCDIC 檔案

任務	Description	所需的技能
將 EBCDIC 檔案傳送至 Amazon S3 儲存貯體。	從 CloudShell 工作資料夾中，輸入下列命令將 EBCDIC 檔案傳送至 Amazon S3 儲存貯體： `aws s3 cp mdu/sample-data/CLIENT.EBCDIC.txt s3://$bucket/input/` **Note** 我們建議您為輸入 (EBCDIC) 和輸出 (ASCII) 檔案設定不同的資料夾，以避免在 ASCII 檔案上傳至 Amazon S3 儲存貯體時再次呼叫 Lambda 轉換函數。	一般 AWS

任務	Description	所需的技能

檢查輸出。	從 CloudShell 工作資料夾中，輸入下列命令來檢查 ASCII 檔案是否在您的 Amazon S3 儲存貯體中產生：
	`aws s3 ls s3://$bucket/`
	Note
	資料轉換可能需要幾秒鐘的時間才會發生。建議您檢查 ASCII 檔案幾次。
	在 ASCII 檔案可用後，輸入下列命令以檢視 Amazon S3 儲存貯體中轉換後檔案的內容。您可以視需要從 Amazon S3 儲存貯體下載或使用它：
	`aws s3 cp s3://$bucket/ CLIENT.ASCII.txt - | head`
	檢查 ASCII 檔案內容：
	`0|0|220|`
	`1|1|HERBERT MOHAMED|1`
	`958-08-31|BACHELOR|`
	`0010000.00|`
	`1|2|36|THE ROE AVENUE|`
	`2|1|JAYLEN GEORGE|19`
	`69-05-29|ELEMENTARY|`
	`0020000.00|`
		一般 AWS

任務	Description	所需的技能
	2|2|365|HEATHFIELD ESPLANADE| 3|1|MIKAEEL WEBER|198 2-02-17|MASTER|003 0000.00| 3|2|4555|MORRISON STRAND| 4|1|APRIL BARRERA|1 967-01-12|DOCTOR|0 030000.00| 4|2|1311|MARMION PARK| 5|1|ALEEZA PLANT|198 5-03-01|BACHELOR|0 008000.00|	

清除環境

任務	Description	所需的技能

（選用）準備變數和資料夾。	如果您失去與 CloudShell 的連線，請重新連線，然後輸入下列命令將目錄變更為工作資料夾：
	`cd workdir`
	確定已定義環境變數：
	`bucket=<your_bucket_name>`
	`account=<your_account_number>`
	`region=<your_region_code>`
		一般 AWS

任務	Description	所需的技能

移除儲存貯體的通知組態。	從 CloudShell 工作資料夾中，執行下列命令來移除 Amazon S3 事件通知組態：
	
	`aws s3api put-bucket-notification-configuration \`
	`--bucket=$bucket \`
	`--notification-configuration="{}"`
		一般 AWS

刪除 Lambda 函數。	從 CloudShell 工作資料夾中，輸入下列命令來刪除 EBCDIC 轉換器的 Lambda 函數：
	
	`aws lambda delete-function \`
	`--function-name E2A`
		一般 AWS

刪除 IAM 角色和政策。	從 CloudShell 工作資料夾中，輸入下列命令來移除 EBCDIC 轉換器角色和政策：
	
	`aws iam delete-role-policy \`
	`--role-name E2AConvLambdaRole \`
	`--policy-name E2AConvLambdaPolicy`
	
	`aws iam delete-role \`
	`--role-name E2AConvLambdaRole`
		一般 AWS

任務	Description	所需的技能

刪除 Amazon S3 儲存貯體中產生的檔案。	從 CloudShell 工作資料夾中，輸入下列命令來刪除 Amazon S3 儲存貯體中產生的檔案：
	`aws s3 rm s3://$bucket/layout --recursive`
	`aws s3 rm s3://$bucket/input --recursive`
	`aws s3 rm s3://$bucket/CLIENT.ASCII.txt`
		一般 AWS

刪除工作資料夾。	從 CloudShell 工作資料夾中，輸入下列命令來移除 `workdir`及其內容： `cd ..; rm -Rf workdir`	一般 AWS

相關資源

- 大型主機資料公用程式 README (GitHub)
- EBCDIC 字元集 (IBM 文件)
- EBCDIC 到 ASCII (IBM 文件)
- COBOL (IBM 文件)
- 使用 Amazon S3 觸發來叫用 Lambda 函數 (AWS Lambda 文件)

使用 Micro Focus 轉換具有複雜記錄配置的大型主機資料檔案

Peter West，Amazon Web Services

總結

注意： AWS Mainframe Modernization 服務（受管執行期環境體驗）不再開放給新客戶。對於類似 AWS Mainframe Modernization Service（受管執行期環境體驗）的功能，請探索 AWS Mainframe Modernization Service（自我管理體驗）。現有客戶可以繼續正常使用該服務。如需詳細資訊，請參閱AWS Mainframe Modernization 可用性變更。

此模式說明如何使用 Micro Focus 結構檔案，將具有非文字資料和複雜記錄配置的大型主機資料檔案，從 EBCDIC（延伸二進位編碼十進位交換碼）字元編碼轉換為 ASCII（美國資訊交換標準碼）字元編碼。若要完成檔案轉換，您必須執行下列動作：

1. 準備單一來源檔案，描述大型主機環境中所有資料項目和記錄配置。
2. 使用 Micro Focus 資料檔案編輯器作為 Micro Focus Classic Data File Tools 或 Data File Tools 的一部分，建立包含資料記錄配置的結構檔案。結構檔案會識別非文字資料，讓您可以正確地將大型主機檔案從 EBCDIC 轉換為 ASCII。
3. 使用傳統資料檔案工具或資料檔案工具測試結構檔案。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Micro Focus Enterprise Developer for Windows，可透過 AWS Mainframe Modernization 取得

產品版本

- Micro Focus Enterprise Server 7.0 及更新版本

工具

- Micro Focus Enterprise Developer 為使用 Enterprise Developer 的任何整合開發環境 (IDE) 變體建立的應用程式提供執行環境。
- Micro Focus Classic Data File Tools 可協助您轉換、導覽、編輯和建立資料檔案。傳統資料檔案工具包括資料檔案轉換器、記錄配置編輯器和資料檔案編輯器。

- Micro Focus 資料檔案工具可協助您建立、編輯和移動資料檔案。資料檔案工具包括資料檔案編輯器、檔案轉換公用程式和資料檔案結構命令列公用程式。

史詩

準備來源檔案

任務	Description	所需的技能
識別來源元件。	識別檔案的所有可能記錄配置，包括包含非文字資料的任何重新定義。 如果您有包含重新定義的配置，您必須將這些配置分解為唯一配置，描述資料結構的每個可能變動。一般而言，資料檔案的記錄配置可以由下列原型描述： • 僅包含文字資料的記錄配置 • 使用非文字資料記錄配置 • 使用非文字資料次級於 REDEFINES 子句的記錄配置 如需為包含複雜記錄配置的檔案建立平面化記錄配置的詳細資訊，請參閱在 ASCII 環境中託管 EBCDIC 應用程式以進行大型主機遷移。	應用程式開發人員
識別記錄配置條件。	對於具有多個記錄配置的檔案，或包含複雜配置且具有 REDEFINES 子句的檔案，請識別記錄中的資料和條件，供您用來定義轉換期間要使用的	應用程式開發人員

任務	Description	所需的技能

	配置。我們建議您與主題專家 (SME) 討論此任務，該專家了解處理這些檔案的程式。
	
	例如，檔案可能包含兩種包含非文字資料的記錄類型。您可以檢查來源，並可能找到類似下列的程式碼：
	
	`MOVE "M" TO PART-TYPE`
	`MOVE "MAIN ASSEMBLY" TO PART-NAME`
	`MOVE "S" TO PART-TYPE`
	`MOVE "SUB ASSEMBLY 1" TO PART-NAME`
	
	此程式碼可協助您識別下列項目：
	
	• 「PART-TYPE」欄位用於判斷記錄類型
	• 值 "M" 用於 "M-PART-RECORD"
	• 值 "S" 用於 "S-PART-RECORD"
	
	您可以記錄此欄位所使用的值，將記錄配置與檔案中正確的資料記錄建立關聯。	

任務	Description	所需的技能
建置來源檔案。	如果透過多個來源檔案描述檔案，或記錄配置包含屬於 REDEFINES 子句的非文字資料，則請建立新的來源檔案，其中包含記錄配置。新程式不需要使用 SELECT 和 FD 陳述式描述檔案。程式可以直接在 Working-Storage 中將記錄描述包含為 01 個層級。 **Note** 您可以為每個資料檔案建立來源檔案，或建立描述所有資料檔案的主來源檔案。	應用程式開發人員

任務	Description	所需的技能
編譯來源檔案。	編譯來源檔案以建置資料字典。建議您使用 EBCDIC 字元集來編譯來源檔案。如果使用 IBMCOMP 指令或 ODOSLIDE 指令，則您也必須在來源檔案中使用這些指令。 **Note** IBMCOMP 會影響 COMP 欄位的位元組儲存，ODOSLIDE 會影響 OCCURS VARYING 結構的填補。如果這些指令設定不正確，轉換工具將無法正確讀取資料記錄。這會導致轉換後檔案中的資料錯誤。	應用程式開發人員

(選項 A) 使用傳統資料檔案工具建立結構檔案

任務	Description	所需的技能
啟動工具並載入字典。	1. 選擇 Windows 開始功能表圖示，搜尋並選擇 Micro Focus Enterprise Developer，然後選擇傳統資料檔案工具。 2. 選擇檔案，然後選擇記錄配置。 3. 在從對話方塊中選取要建構配置的檔案中，針對檔案名	應用程式開發人員

任務	Description	所需的技能
	稱，選取您先前編譯來源檔案時建立的 IDY (`.idy`) 檔案。然後選擇 Open (開啟)。 4. 若要確認傳統資料檔案工具正在使用 EBCDIC，請在資料檔案工具對話方塊中選擇是，如果 IDY 檔案設定為 EBCDIC，且 Datatools 設定為 ANSI。	
建立預設記錄配置。	針對不符合任何條件配置的所有記錄，使用預設記錄配置。 1. 在配置視窗中，展開資料結構，然後找到用於預設配置的 01 層級。 2. 在 01 項目上按一下滑鼠右鍵，然後選擇新配置。 3. 在新記錄配置精靈對話方塊中，選擇預設配置，然後選擇下一步。 4. 選擇完成。 預設配置會出現在配置窗格中，可透過紅色資料夾圖示識別。	應用程式開發人員

任務	Description	所需的技能
建立條件式記錄配置。	當檔案中有多個記錄配置時，請使用條件式記錄配置。 1. 在配置窗格中，展開資料結構，然後找到用於條件式配置的 01 關卡。 2. 在 01 項目上按一下滑鼠右鍵，然後選擇新配置。 3. 在新增記錄配置精靈對話方塊中，選擇條件式配置，然後選擇下一步。 4. 選擇完成。條件式配置會出現在配置窗格中，可透過黃色資料夾圖示識別。 5. 展開條件式配置，在必須放置條件的欄位上按一下滑鼠右鍵，然後選擇屬性。 6. 在欄位屬性對話方塊中，輸入 條件。確認字元集已設定為 EBCDIC，然後選擇確定。具有條件集的欄位旁會出現核取記號。 7. 針對需要此配置條件的任何其他欄位，重複步驟 5-6。 8. 針對任何其他必須新增的條件式配置，重複步驟 1–6。 9. 選擇檔案，選擇另存新檔，然後將結構檔案儲存至磁碟。	應用程式開發人員

(選項 B) 使用資料檔案工具建立結構檔案

任務	Description	所需的技能
啟動工具並載入字典。	1. 選擇 Windows 開始功能表圖示，搜尋並選擇 Micro Focus Enterprise Developer，然後選擇資料檔案工具。 2. 選擇檔案、新增、結構檔案。 3. 在開啟對話方塊中，針對檔案名稱，選取您先前編譯來源檔案時建立的 IDY (`.idy`) 檔案。然後選擇 Open (開啟)。 4. 若要確認資料檔案工具正在使用 EBCDIC，請確認偵錯檔案區段中的下拉式功能表已設定為 EBCDIC。	應用程式開發人員
建立預設記錄配置。	針對不符合任何條件式配置的所有記錄，使用預設記錄配置。 1. 在左側窗格的可用配置區段中，展開資料結構，然後找到用於預設配置的 01 層級。 2. 在 01 項目上按一下滑鼠右鍵，然後選擇建立預設配置。 預設配置會出現在配置窗格中，並且可以透過藍色「D」圖示來識別。	應用程式開發人員

任務	Description	所需的技能
建立條件式記錄配置。	當檔案中有多個記錄配置時，請使用條件式記錄配置。 1. 在右窗格的所選配置區段中，展開資料結構，然後找到用於條件式配置的 01 關卡。 2. 在 01 項目上按一下滑鼠右鍵，然後選擇建立條件式配置。條件式配置會出現在右側的配置窗格中，並以綠色 "C" 圖示識別。 3. 展開條件式配置，在必須放置條件的欄位上按一下滑鼠右鍵，然後選擇屬性。 4. 在欄位屬性對話方塊中，輸入 條件。確認字元集已設定為 EBCDIC，然後選擇確定。具有條件集的欄位旁會出現紅色「IF」圖示。 5. 針對需要此配置條件的任何其他欄位，重複步驟 3–4。 6. 對必須新增的任何其他條件式配置重複步驟 1–4。 7. 選擇檔案，選擇另存新檔，然後將結構檔案儲存至磁碟。	應用程式開發人員

(選項 A) 使用傳統資料檔案工具測試結構檔案

任務	Description	所需的技能
測試 EBCDIC 資料檔案。	確認您可以使用結構檔案來正確檢視 EBCDIC 測試資料檔案。 1. 選擇 Windows 開始功能表圖示，尋找並選擇 Micro Focus Enterprise Developer，然後選擇傳統資料工具。 2. 選擇檔案，然後選擇開啟。 3. 在開啟對話方塊中，針對檔案名稱選取 EBCDIC 資料集，然後選擇開啟。 4. 選擇檔案、資料檔案編輯器、載入記錄配置。 5. 在開啟對話方塊中，針對檔案名稱，選取結構檔案，然後選擇開啟。 6. 若要確認字元集模式設定為 EBCDIC，請確認下拉式選單設定為 EBCDIC。您可以在左側窗格中查看原始記錄資料，並在右側窗格中查看格式化資料。 7. 選擇各種記錄，以確保以正確的配置呈現所有格式。	應用程式開發人員

(選項 B) 使用資料檔案工具測試結構檔案

任務	Description	所需的技能
測試 EBCDIC 資料檔案。	確認您可以使用結構檔案來正確檢視 EBCDIC 測試資料檔案。 1. 選擇 Windows 開始功能表圖示，尋找並選取 Micro Focus Enterprise Developer，然後選擇資料檔案工具。 2. 選擇檔案、開啟、資料檔案。 3. 在開啟資料檔案對話方塊的本機索引標籤中，針對檔案名稱選擇瀏覽以尋找 EBCDIC 測試檔案的位置。 4. 針對結構檔案 （選用），選擇瀏覽以尋找結構檔案的位置。 5. 在檔案詳細資訊區段中，輸入檔案的詳細資訊，並確認編碼設定為 EBCDIC。 6. 根據您的需求選擇開啟共用模式或開啟獨佔模式。 7. 確認工具列外觀區段中的下拉式功能表已設定為 EBCDIC。您會在左側窗格中看到原始記錄資料，並在右側窗格中看到格式化的資料。 8. 選擇各種記錄，以確保以正確的配置呈現所有格式。	應用程式開發人員

測試資料檔案轉換

任務	Description	所需的技能
測試 EBCDIC 檔案的轉換。	1. 選擇 Windows 開始功能表圖示，尋找並選取 Micro Focus Enterprise Developer，然後選擇傳統資料工具。 2. 選擇工具，然後選擇轉換。 3. 在資料檔案轉換對話方塊中，於輸入檔案區段中，針對檔案名稱選擇瀏覽以尋找並選取 EBCDIC 輸入檔案。確認字元集設定為 EBCDIC。 4. 在字元集轉換區段中，選取轉換字元集和記錄包含非文字資料項目核取方塊。選擇選取轉換配置，然後選擇瀏覽以尋找並選取結構檔案。 5. 在新增檔案區段中，針對檔案名稱，輸入您要建立之 ASCII 輸出檔案的路徑和檔案名稱。根據預設，轉換工具預設為與輸入檔案相同的格式。若要進行測試，請將選項設定為其預設值。 6. 選擇轉換。 7. 請遵循 （選項 A) 使用傳統資料檔案工具測試結構檔案或 （選項 B) 使用資料檔案工具測試結構檔案一節中的步驟，但載入 ASCII 輸出檔案，而非 EBCDIC 檔案。	應用程式開發人員

任務	Description	所需的技能
	8. 將 EBCDIC 和 ASCII 檔案載入資料檔案編輯器，然後並排比較檔案以檢查轉換的準確性。	

相關資源

- Micro Focus (Micro Focus 文件)
- 大型主機和舊版程式碼 (AWS 部落格文章)
- AWS 規範指引 (AWS 文件)
- AWS 文件 (AWS 文件)
- AWS 一般參考 (AWS 文件)
- AWS 詞彙表 (AWS 文件)

使用 Terraform 部署容器化 Blu Age 應用程式的環境

Richard Milner-Watts，Amazon Web Services

總結

將舊版大型主機工作負載遷移至現代雲端架構，可以免除維護大型主機的成本，而這些成本只會隨著環境的老化而增加。不過，從大型主機遷移任務可能會帶來獨特的挑戰。內部資源可能不熟悉任務邏輯，與商品化、一般化 CPUs 相比，在這些特殊任務中大型主機的高效能可能很難複寫。重寫這些任務可能是一項大型任務，需要大量精力。

Blu Age 會將舊版大型主機工作負載轉換為現代 Java 程式碼，然後您可以做為容器執行。

此模式提供範例無伺服器架構，用於執行已使用 Blu Age 工具進行現代化化的容器化應用程式。隨附的 HashiCorp Terraform 檔案將建置安全架構，以協調 Blu Age 容器，同時支援批次任務和即時服務。

如需使用 Blu Age 和 AWS 服務來現代化工作負載的詳細資訊，請參閱下列 AWS 規範指引出版物：

- 在 AWS 無伺服器基礎設施上使用 Blu Age 進行現代化的大型主機工作負載
- 容器化已由 Blu Age 現代化的大型主機工作負載

如需使用 Blu Age 來現代化大型主機工作負載的協助，請在 Blu Age 網站上選擇聯絡我們的專家，以聯絡 Blu Age 團隊。如需協助將現代化工作負載遷移至 AWS、將它們與 AWS 服務整合，並將它們移至生產環境，請聯絡您的 AWS 客戶經理或填寫 AWS Professional Services 表單。

先決條件和限制

先決條件

- Containerize 大型主機工作負載所提供的範例容器化 Blu Age 應用程式，已透過 Blu Age 模式進行現代化。範例應用程式提供邏輯來處理現代化應用程式的輸入和輸出處理，並可與此架構整合。
- 部署這些資源需要 Terraform。

限制

- Amazon Elastic Container Service (Amazon ECS) 會限制可供容器使用的任務資源。這些資源包括 CPU、RAM 和儲存。例如，使用 Amazon ECS 搭配 AWS Fargate 時，會套用任務資源限制。

產品版本

此解決方案已使用下列版本進行測試：

- Terraform 1.3.6
- Terraform AWS 提供者 4.46.0

Architecture

來源技術堆疊

- 藍齡
- Terraform

目標技術堆疊

- Amazon Aurora PostgreSQL-Compatible Edition
- AWS Backup
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon ECS
- AWS Identity and Access Management Service (IAM)
- AWS Key Management Server (AWS KMS)
- AWS Secrets Manager
- Amazon Simple Notification Service (Amazon SNS)
- Amazon Simple Storage Service (Amazon S3)
- AWS Step Functions
- AWS Systems Manager

目標架構

下圖顯示解決方案架構。

1. 解決方案會部署下列 IAM 角色：
    - 批次任務角色
    - 批次任務執行角色

- 服務任務角色
- 服務任務執行角色
- Step Functions 角色
- AWS Backup角色
- RDS 增強型監控角色。

角色符合最低權限的存取原則。

2. Amazon ECR 用於存放由此模式協調的容器映像。
3. AWS Systems Manager 參數存放區會在執行時間將每個環境的組態資料提供給 Amazon ECS 任務定義。
4. AWS Secrets Manager 會在執行時間將環境的敏感組態資料提供給 Amazon ECS 任務定義。資料已由 AWS KMS 加密。
5. Terraform 模組會為所有即時和批次任務建立 Amazon ECS 任務定義。
6. Amazon ECS 使用 AWS Fargate 作為運算引擎來執行批次任務。這是短期任務，由 AWS Step Functions 視需要啟動。
7. Amazon Aurora PostgreSQL 相容提供資料庫以支援現代化應用程式。這會取代大型主機資料庫，例如 IBM Db2 或 IBM IMS 資料庫。
8. Amazon ECS 會執行長期服務，以提供現代化即時工作負載。這些無狀態應用程式會隨著跨可用區域的容器永久執行。
9. Network Load Balancer 用於授予即時工作負載的存取權。Network Load Balancer 支援舊版通訊協定，例如 IBM CICS。或者，您可以將 Application Load Balancer 與 HTTP 型工作負載搭配使用。
10. Amazon S3 為任務輸入和輸出提供物件儲存。容器應處理 Amazon S3 的提取和推送操作，以準備 Blu Age 應用程式的工作目錄。
11. AWS Step Functions 服務用於協調執行 Amazon ECS 任務以處理批次工作負載。
12. 每個批次工作負載的 SNS 主題會用來整合現代化應用程式與其他系統，例如電子郵件，或啟動其他動作，例如將輸出物件從 Amazon S3 交付至 FTP。

> **Note**
>
> 根據預設，解決方案無法存取網際網路。此模式假設虛擬私有雲端 (VPC) 將使用 AWS Transit Gateway 等服務連接到其他網路。因此，部署了多個界面 VPC 端點，以授予對解決方案所用 AWS 服務的存取權。若要開啟直接網際網路存取，您可以使用 Terraform 模組中的切換，將 VPC 端點取代為網際網路閘道和相關聯的資源。

自動化和擴展

在整個模式中使用無伺服器資源有助於確保，透過向外擴展，此設計的規模幾乎沒有限制。這可減少雜訊的鄰里問題，例如在原始大型主機上可能遇到的運算資源競爭。批次任務可以排程為視需要同時執行。

個別容器受限於 Fargate 支援的大小上限。如需詳細資訊，請參閱 Amazon ECS 文件中的任務 CPU 和記憶體一節。

若要水平擴展即時工作負載，您可以新增容器。

工具

AWS 服務

- Amazon Aurora PostgreSQL 相容版本是完全受管的 ACID 相容關聯式資料庫引擎，可協助您設定、操作和擴展 PostgreSQL 部署。
- AWS Backup 是一項全受管服務，可協助您集中和自動化跨雲端和內部部署 AWS 服務的資料保護。
- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。
- Amazon Elastic Container Service (Amazon ECS) 是快速、可擴展的容器管理服務，可協助您執行、停止和管理叢集上的容器。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- AWS Secrets Manager 可協助您以 API 呼叫 Secrets Manager，以程式設計方式擷取秘密，取代程式碼中的硬式編碼登入資料，包括密碼。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務 來建置業務關鍵應用程式。
- AWS Systems Manager 參數存放區為組態資料管理和秘密管理提供安全的階層式儲存。

其他服務

- HashiCorp Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。此模式使用 Terraform 來建立範例架構。

程式碼儲存庫

此模式的原始程式碼可在 GitHub Blu Age Sample ECS Infrastructure (Terraform) 儲存庫中使用。

最佳實務

- 對於測試環境，請使用 等功能`forceDate`來設定現代化應用程式，藉由一律在已知期間內執行 來產生一致的測試結果。
- 個別調校每個任務，以取用最佳數量的資源。您可以使用 Amazon CloudWatch Container Insights 來取得潛在瓶頸的指引。

史詩

準備環境以進行部署

任務	Description	所需的技能
複製解決方案原始程式碼。	從 GitHub 專案複製解決方案程式碼。	DevOps 工程師
透過部署資源來存放 Terraform 狀態來引導環境。	1. 開啟終端機視窗，並確認已安裝 Terraform 且 AWS 憑證可用。 2. 導覽至 `bootstrap-terraform` 資料夾。 3. `main.tf` 如果您想要變更 S3 儲存貯體 (`<accountId>-terraform-backend`) 和 Amazon DynamoDB 資料表 () 的名稱，請編輯 檔案`terraform-lock` 。 4. 執行 `terraform apply`命令來部署資源。記下 S3 儲	DevOps 工程師

任務	Description	所需的技能
	存貯體和 DynamoDB 資料表名稱。	

部署解決方案基礎設施

任務	Description	所需的技能
檢閱並更新 Terraform 組態。	在根目錄中，開啟 檔案`main.tf`,檢閱內容，並考慮進行下列更新： 1. AWS 區域 透過搜尋並將字串取代`eu-west-1` 為您想要使用的所需區域來更新 。 2. 如果在上一個史詩中更改了預設值，請在 Terraform Backend 區塊中更新儲存貯體名稱。 3. 如果在上一個史詩中更改了預設值，請更新該`dynamodb_table` 值。 4. 將`stack_prefix` 變數的值更新為您想要的字串。此字串會放在此模式建立的所有資源名稱前面。 5. 更新 的值 `vpc_cidr` 這應該至少是一個`/24`地址範圍。 6. 檢閱 `Locals`區段。這用於定義將部署的 Blu Age 任務。解決方案會逐一查看清單物件 `bluage_batch_modules` ，為清	DevOps 工程師

任務	Description	所需的技能
	單的每個元素建立相關聯的資源 (Step Functions 狀態機器、任務定義和 SNS 主題）。在某些情況下，您可能想要調整不同環境的變數。例如，若要在測試環境中強制執行時間，您可以變更 `force_execution_time` 變數的值。 7. 若要開啟網際網路存取，請將 的值`direct_internet_access_required` 從 `false`變更為 `true`。這將部署網際網路閘道，以及開啟基礎設施公有網際網路存取的 NAT 閘道和路由表。根據預設，解決方案會將介面 VPC 端點部署到 VPC，而不需要直接存取網際網路。 8. 若要授予透過 Elastic Load Balancing 提供的任何用戶端伺服器工作負載的存取權，請使用應允許的 `additional_nlb_ingress_cidrs` CIDR 網路更新 的值。	

任務	Description	所需的技能
部署 Terraform 檔案。	從終端機執行 `terraform apply`命令來部署所有資源。檢閱 Terraform 產生的變更，然後輸入 yes 以啟動建置。 請注意，部署此基礎設施可能需要超過 15 分鐘的時間。	DevOps 工程師

（選用）部署有效的 Blu Age 容器化應用程式

任務	Description	所需的技能
將 Blu Age 容器映像推送至 Amazon ECR。	將容器推送到您在上一個 epic 中建立的 Amazon ECR 儲存庫。如需說明，請參閱 Amazon ECR 文件。 記下容器映像 URI。	DevOps 工程師
更新 Terraform 以參考 Blu Age 容器映像。	更新 檔案`main.tf` 以參考您上傳的容器映像。	DevOps 工程師
重新部署 Terraform 檔案。	從終端機執行 `terraform apply`以部署所有資源。從 Terraform 檢閱建議的更新，然後輸入 yes 以繼續部署。	DevOps 工程師

相關資源

- 藍齡
- 在 AWS 無伺服器基礎設施上使用 Blu Age 進行現代化的大型主機工作負載
- 容器化已由 Blu Age 現代化的大型主機工作負載

使用 Quick Sight 中的 AWS Mainframe Modernization 和 Amazon Q 產生 Db2 z/OS 資料洞見

Shubham Roy、Roshna Razack 和 Santosh Kumar Singh，Amazon Web Services

總結

注意： AWS Mainframe Modernization 服務（受管執行期環境體驗）不再開放給新客戶。對於類似 AWS Mainframe Modernization Service（受管執行期環境體驗）的功能，請探索 AWS Mainframe Modernization Service（自我管理體驗）。現有客戶可以繼續正常使用該服務。如需詳細資訊，請參閱AWS Mainframe Modernization 可用性變更。

如果您的組織在 IBM Db2 大型主機環境中託管業務關鍵資料，從該資料中獲得洞見對於推動成長和創新至關重要。透過解鎖大型主機資料，您可以建立更快、安全和可擴展的商業智慧，以加速 Amazon Web Services (AWS) 雲端中的資料驅動型決策、成長和創新。

此模式提供解決方案，用於產生商業洞見，並從適用於 z/OS 資料表的 IBM Db2 中大型主機資料建立可分割敘述。大型主機資料變更會使用AWS Mainframe Modernization 資料複寫搭配 Precisely 串流至 Amazon Managed Streaming for Apache Kafka (Amazon MSK) 主題。使用 Amazon Redshift 串流擷取，Amazon MSK 主題資料會存放在 Amazon Redshift Serverless 資料倉儲資料表中，以在 Amazon Quick Sight 中進行分析。

在 Quick Sight 中提供資料後，您可以使用自然語言提示搭配 Amazon Q in Quick Sight 來建立資料摘要、提出問題和產生資料案例。您不需要撰寫 SQL 查詢或學習商業智慧 (BI) 工具。

商業內容

此模式提供大型主機資料分析和資料洞察使用案例的解決方案。使用 模式，您可以為公司資料建立視覺化儀表板。為了示範解決方案，此模式使用提供醫療、牙科和視覺計劃給美國成員的醫療保健公司。在此範例中，成員人口統計特性和計劃資訊會存放在 z/OS 資料表的 IBM Db2 中。視覺化儀表板會顯示下列項目：

- 區域的成員分佈
- 依性別分配成員
- 按年齡劃分的成員分佈
- 依計劃類型分配成員
- 尚未完成預防性預防預防接種的成員

如需依區域和尚未完成預防性預防預防接種的成員分發的範例，請參閱其他資訊一節。

建立儀表板後，您會產生一個資料案例，說明先前分析的洞見。資料案例提供建議，以增加已完成預防性預防接種的成員人數。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。此解決方案是在 Amazon Elastic Compute Cloud (Amazon EC2) 上的 Amazon Linux 2 上建置和測試。
- 具有子網路的虛擬私有雲端 (VPC)，可由您的大型主機系統存取。
- 具有商業資料的大型主機資料庫。如需用於建置和測試此解決方案的範例資料，請參閱附件一節。
- 在 Db2 z/OS 資料表上啟用變更資料擷取 (CDC)。若要在 Db2 z/OS 上啟用 CDC，請參閱 IBM 文件。
- 在託管來源資料庫的 z/OS 系統上安裝的 z/OS 的精確連線 CDC。Precisely Connect CDC for z/OS 映像會以 zip 檔案的形式提供於 AWS Mainframe Modernization - Data Replication for IBM z/OS Amazon Machine Image (AMI)。若要在大型主機上安裝適用於 z/OS 的 Precisely Connect CDC，請參閱 Precisely 安裝文件。

限制

- 您的大型主機 Db2 資料應該位於 Precisely Connect CDC 支援的資料類型中。如需支援的資料類型清單，請參閱 Precisely Connect CDC 文件。
- 您在 Amazon MSK 的資料應位於 Amazon Redshift 支援的資料類型中。如需支援的資料類型清單，請參閱 Amazon Redshift 文件。
- Amazon Redshift 對不同的資料類型有不同的行為和大小限制。如需詳細資訊，請參閱 Amazon Redshift 文件。
- Quick Sight 中近乎即時的資料取決於為 Amazon Redshift 資料庫設定的重新整理間隔。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。Amazon Q in Quick Sight 目前不適用於支援 Quick Sight 的每個區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

產品版本

- AWS Mainframe Modernization 使用精確的 4.1.44 版進行資料複寫
- Python 3.6 版或更新版本
- Apache Kafka 3.5.1 版

Architecture

目標架構

下圖顯示使用AWS Mainframe Modernization 資料複寫搭配 Precisely 和 Amazon Q in Quick Sight 從大型主機資料產生商業洞見的架構。

該圖顯示以下工作流程：

1. Precisely Log Reader Agent 會從 Db2 日誌讀取資料，並將資料寫入大型主機上 OMVS 檔案系統的暫時性儲存體。
2. 發佈者代理程式會從暫時性儲存讀取原始 Db2 日誌。
3. 內部部署控制器協助程式會驗證、授權、監控和管理操作。
4. Apply Agent 是使用預先設定的 AMI 部署在 Amazon EC2 上。它會使用 TCP/IP 透過控制器常駐程式與發佈者代理程式連線。Apply Agent 使用多個工作者將資料推送到 Amazon MSK 以進行高輸送量。
5. 工作者會以 JSON 格式將資料寫入 Amazon MSK 主題。作為複寫訊息的中繼目標，Amazon MSK 提供高可用性和自動化容錯移轉功能。
6. Amazon Redshift 串流擷取提供從 Amazon MSK 到 Amazon Redshift Serverless 資料庫的低延遲、高速資料擷取。Amazon Redshift 中的預存程序會對 Amazon Redshift 資料表執行大型主機變更資料 (insert/update/deletes) 調校。這些 Amazon Redshift 資料表做為 Quick Sight 的資料分析來源。
7. 使用者存取 Quick Sight 中的資料，以取得分析和洞見。您可以使用 Amazon Q in Quick Sight，透過自然語言提示與資料互動。

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，並快速將其向外擴展或向內擴展。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- Amazon Managed Streaming for Apache Kafka (Amazon MSK) 是一種全受管服務，可協助您建置和執行使用 Apache Kafka 處理串流資料的應用程式。
- Amazon Quick Sight 是一種雲端規模的商業智慧 (BI) 服務，可協助您在單一儀表板中視覺化、分析和報告您的資料。此模式使用 Quick Sight 中 Amazon Q 的生成式 BI 功能。

- Amazon Redshift Serverless 是 Amazon Redshift 的無伺服器選項，可讓您在幾秒鐘內更有效率地執行和擴展分析，而無需設定和管理資料倉儲基礎設施。
- AWS Secrets Manager 可協助您將程式碼中的硬式編碼憑證 (包括密碼) 取代為 Secrets Manager 的 API 呼叫，以便透過程式設計方法來擷取機密。

其他工具

- Precisely Connect CDC 會收集舊版系統的資料，並將其整合到雲端和資料平台。

程式碼儲存庫

此模式的程式碼可在 GitHub Mainframe_DataInsights_change_data_reconciliation 儲存庫中使用。此程式碼是 Amazon Redshift 中的預存程序。此預存程序會將大型主機資料變更 （插入、更新和刪除）從 Amazon MSK 協調至 Amazon Redshift 資料表。這些 Amazon Redshift 資料表可做為 Quick Sight 的資料分析來源。

最佳實務

- 設定 Amazon MSK 叢集時，請遵循最佳實務。
- 遵循 Amazon Redshift 資料剖析最佳實務來改善效能。
- 當您為精確設定建立 AWS Identity and Access Management (IAM) 角色時，請遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。

史詩

在 Amazon EC2 上使用精確設定 AWS Mainframe Modernization 資料複寫

任務	Description	所需的技能
設定安全群組。	若要連線至控制器協助程式和 Amazon MSK 叢集，請為 EC2 執行個體建立安全群組。新增下列傳入和傳出規則： • 傳入規則 1：	DevOps 工程師，AWS DevOps

任務	Description	所需的技能
	• 針對類型，選擇自訂 TCP。 • 針對 Protocol (通訊協定)，選擇 TCP。 • 針對連接埠範圍，選擇 2626（精確控制器協助程式的預設連接埠）或在大型主機上執行之控制器協助程式的連接埠號碼。 • 針對來源，選擇 CIDR 區塊。 • 傳入規則 2： • 針對 Type (類型)，請選擇 Custom TCP (自訂 TCP)。 • 針對通訊協定，選擇 SSH。 • 針對連接埠範圍，選擇 22。 • 針對來源，選擇 IP 地址或字首清單。 • 傳入規則 3： • 針對 Type (類型)，請選擇 Custom TCP (自訂 TCP)。 • 針對 Protocol (通訊協定)，選擇 TCP。 • 針對連接埠範圍，選擇 9092-9098。 • 針對來源，選擇 CIDR 區塊。 • 傳出規則 1：	

任務	Description	所需的技能
	• 針對 Type (類型)，請選擇 Custom TCP (自訂 TCP)。 • 針對 Protocol (通訊協定)，選擇 TCP。 • 針對連接埠範圍，選擇 9092-9098。 • 針對來源，選擇 CIDR 區塊。 • 傳出規則 2： • 針對 Type (類型)，請選擇 Custom TCP (自訂 TCP)。 • 針對 Protocol (通訊協定)，選擇 TCP。 • 針對連接埠範圍，選擇 2626（精確控制器協助程式的預設連接埠）或在大型主機上執行之控制器協助程式的連接埠號碼。 • 針對來源，選擇 CIDR 區塊。 請記下安全群組的名稱。當您啟動 EC2 執行個體並設定 Amazon MSK 叢集時，將需要參考名稱。	

任務	Description	所需的技能
建立 IAM 政策和 IAM 角色。	1. 若要建立 IAM 政策和 IAM 角色，請遵循 AWS 文件中的指示。 IAM 政策授予在 Amazon MSK 叢集上建立主題，以及將資料傳送至這些主題的存取權。 2. 建立 IAM 角色之後，請將政策與其建立關聯。請注意 IAM 角色名稱。當您啟動 EC2 執行個體時，此角色將用作 IAM 執行個體描述檔。	DevOps 工程師、AWS 系統管理員

任務	Description	所需的技能
佈建 EC2 執行個體。	若要佈建 EC2 執行個體來執行精確 CDC 並連線至 Amazon MSK，請執行下列動作： 1. 登入 AWS Marketplace 並訂閱 AWS Mainframe Modernization – IBM z/OS 的資料複寫。 2. 從受管訂閱中選取 AMI，然後在啟動新執行個體時選擇 。 3. 提供其他組態詳細資訊，例如執行個體名稱、執行個體類型、金鑰對、VPC 和子網路。如需詳細資訊，請參閱 Amazon EC2 說明文件。 4. 在下拉式清單中，選擇您先前建立的安全群組。 5. 在進階詳細資訊的 IAM 執行個體描述檔下，您必須選取先前建立的角色。 6. 選擇啟動執行個體。	AWS 管理員、DevOps 工程師

設定 Amazon MSK

任務	Description	所需的技能
建立 Amazon MSK 叢集。	若要建立 Amazon MSK 叢集，請執行下列 ： 1. 登入 AWS 管理主控台，然後導覽至 Amazon MSK。	AWS DevOps，雲端管理員

任務	Description	所需的技能
	2. 選擇 Create Cluster (建立叢集)。 3. 針對叢集建立方法，選擇自訂建立，針對叢集類型，選擇佈建。 4. 提供叢集的名稱。 5. 視需要更新叢集設定，並保留其他設定的預設值。 6. 請注意 <Kafka 版本>。 您將需要在 Kafka 用戶端設定期間使用它。 7. 選擇下一步。 8. 選擇您用於 Precisely EC2 執行個體的相同 VPC 和子網路，然後選擇您先前建立的安全群組。 9. 在安全設定區段中，啟用 SASL/SCRAM 和 IAM 角色型身分驗證。Precisely Connect CDC 使用 SASL/SCRAM （簡易身分驗證和安全層/鹽分挑戰回應機制），且需要 IAM 才能連線至 Amazon Redshift。 10選擇下一步。 11若要檢閱，請選擇監控和中介裝置日誌交付方法。 12選擇下一步，然後選擇建立叢集。	

任務	Description	所需的技能
	建立典型的佈建叢集最多需要 15 分鐘。建立叢集之後，其狀態會從建立變更為作用中。	
設定 SASL/SCRAM 身分驗證。	若要設定 Amazon MSK 叢集的 SASL/SCRAM 身分驗證，請執行下列動作： 1. 若要在 Secrets Manager 中設定秘密，請遵循 AWS 文件中的指示。 2. 開啟 Amazon MSK 主控台，然後選取您先前建立的 Amazon MSK 叢集。 3. 選擇屬性索引標籤。 4. 選擇關聯秘密、選擇秘密、選取您建立的秘密金鑰，然後選擇關聯秘密。 您將看到類似以下的成功訊息： `Successfully associated 1 secret for cluster <chosen cluster name>` 5. 選擇叢集名稱。 6. 在叢集摘要中，選擇檢視用戶端資訊。 7. 請注意身分驗證類型 SASL/SCRAM 的私有端點連線字串。	雲端架構師

任務	Description	所需的技能

建立 Amazon MSK 主題。	若要建立 Amazon MSK 主題，請執行下列動作：
	1. 連線至您先前建立的 EC2 執行個體，並執行下列命令來安裝最新的更新：
	`sudo yum update -y`
	2. 執行下列命令來安裝 Java 和 Kafka 程式庫：
	`sudo yum install -y java-11 librdkafka librdkafka-devel`
	3. 若要在 kafka中建立名為的資料夾/home/ec2-user，請導覽至該資料夾，然後執行下列命令：
	`mkdir kafka;cd kafka`
	4. 將kafka用戶端程式庫下載到 kafka 資料夾，<YOUR MSK VERSION>將 取代為您在 Amazon MSK 叢集建立期間記下的 Kafka 版本：
	`wget https://archive.apache.org/dist/kafka//kafka_2.13-<YOUR MSK VERSION>.tgz`
	5. 若要擷取下載的檔案，請執行下列命令，取代 YOUR MSK VERSION>：
		雲端管理員

任務	Description	所需的技能

	`tar -xzf kafka_2.13-<YOUR MSK VERSION>.tgz`
	6. 若要導覽至 kafka libs目錄並下載 Java IAM 身分驗證 Java Archive (JAR) 檔案，請執行下列命令，取代 <YOUR MSK VERSION>：
	`cd kafka_2.13-<YOUR MSK VERSION>/libs wget https://github.com/aws/aws-msk-iam-auth/releases/download/v1.1.1/aws-msk-iam-auth-1.1.1-all.jarkafka`
	7. 若要導覽至 Kafka bin目錄並建立 client.properties 檔案，請執行下列命令：
	`cd /home/ec2-user/kafka/kafka_2.13-<YOUR MSK VERSION>/bin cat >client.properties`
	8. 使用下列內容更新 client.properties 檔案：
	`security.protocol=SASL_SSL sasl.mechanism=AWS_MSK_IAM`	

任務	Description	所需的技能
	sasl.jaas.config=software.amazon.msk.auth.iam.IAMLoginModule required; sasl.client.callback.handler.class=software.amazon.msk.auth.iam.IAMClientCallbackHandler 9. 若要建立 Kafka 主題，請導覽至 Kafka bin 並執行下列命令，<kafka broker>將取代為您在建立 Amazon MSK 叢集期間記下的 IAM 引導伺服器私有端點： ./kafka-topics.sh --bootstrap-server <kafka broker> --command-config client.properties --create --replication-factor 3 –partitions 6 --topic <topic name> 10.當訊息Created topic <topic name>出現時，請記下主題名稱。	

在 Amazon EC2 上設定精確套用引擎

任務	Description	所需的技能
設定精確指令碼以複寫資料變更。	若要設定 Precisely Connect CDC 指令碼，將已變更的資	應用程式開發人員、雲端架構師

任務	Description	所需的技能

	料從大型主機複寫至 Amazon MSK 主題，請執行下列動作：
	
	1. 若要精確建立資料夾名稱並變更為該資料夾，請執行下列命令：
	
	`mkdir /home/ec2-user/precisely;cd /home/ec2-user/precisely`
	
	2. 若要在 `scripts`和 內建立兩個資料夾`ddls`，然後變更為 `scripts` 資料夾，請執行下列命令：
	
	`mkdir scripts;mkdir ddls;cd scripts`
	
	3. 若要`sqdata_kafka_producer.conf` 在 `scripts` 資料夾中建立名為的檔案，請執行下列命令：
	
	`cat >sqdata_kafka_producer.conf`
	
	4. 使用下列內容更新`sqdata_kafka_producer.conf` 檔案：
	
	`builtin.features=SASL_SCRAM`
	`security.protocol=SASL_SSL`
	`sasl.mechanism=SCRAM-SHA-512`	

任務	Description	所需的技能

```
sasl.username=<User
 Name>
sasl.passwor
d=<Password>
metadata.broke
r.list=<SASL/SCRAM
 Bootstrap servers>
```

> **Note**
>
> <SASL/SCRAM Bootstrap servers> 使用您先前設定的 Amazon MSK SASL/SCRAM 代理程式清單進行更新。<User Name> <Password> 使用您先前在 Secrets Manager 中設定的使用者名稱和密碼來更新 和 。

5. 在 scripts 資料夾中建立 script.sqd 檔案。

```
cat >script.sqd
```

   Apply Engine 使用 script.sqd 處理來源資料，並將來源資料複寫到目標。如需套用引擎指令碼範例，請參閱其他資訊一節。

6. 若要變更為 ddls 資料夾並為每個 Db2 資料表建立 .ddl 檔案，請執行下列命令：

任務	Description	所需的技能
	``` cd /home/ec2-user/precisely/ddls cat >mem_details.ddl cat >mem_plans.ddl ``` 如需範例 .ddl 檔案，請參閱其他資訊一節。	

任務	Description	所需的技能

產生網路 ACL 金鑰。	若要產生網路存取控制清單（網路 ACL) 金鑰，請執行下列動作：
	
	1. 若要匯出sqdata安裝路徑，請執行下列命令：
	
	`export PATH=$PATH:/usr/sbin:/opt/precisely/di/sqdata/bin`
	
	2. 若要變更為 /home/ec2-user目錄並產生網路 ACL 金鑰，請執行下列命令：
	
	`cd /home/ec2-user`
	`sqdutil keygen --force`
	
	產生公有和私有金鑰後，會顯示下列訊息：
	
	`SQDUT04I Generating a private key in file /home/ec2-user/.nacl/id_nacl`
	`SQDC017I sqdutil(pid=27344) terminated successfully`
	
	3. 請注意存放在 .nacl 資料夾中產生的公有金鑰。
		雲端架構師，AWS DevOps

準備大型主機來源環境

任務	Description	所需的技能
在 ISPF 畫面中設定預設值。	若要在互動式系統生產力設施 (ISPF) 中設定預設設定，請遵循精確文件中的指示。	大型主機系統管理員
設定控制器協助程式。	若要設定控制器協助程式，請執行下列動作： 1. 在 SQData z/OS 主功能表畫面上，選擇選項 2。 2. 在新增協助程式至清單畫面的協助程式名稱欄位中輸入協助程式的名稱，然後按 Enter 鍵。	大型主機系統管理員
設定發布者。	若要設定發佈者，請執行下列動作： 1. 在 SQData z/OS 主功能表畫面上，選擇選項 3。這會帶您前往擷取/發佈摘要畫面。 2. 選擇新增 CAB 檔案的選項。這將帶您前往將 CAB 檔案新增至清單畫面。 3. 在名稱欄位中，輸入 CAB 檔案的名稱。對於 Db2，輸入類型為 D。 4. 按 Enter。這會帶您前往建立新的 Db2 擷取 CAB 檔案畫面。 5. 在 zFS Dir 欄位中，指定儲存體掛載點。	大型主機系統管理員

任務	Description	所需的技能
	6. 按 Enter 鍵儲存並繼續。	

更新協助程式組態檔案。	若要更新控制器常駐程式組態檔案中的發佈者詳細資訊，請執行下列動作：
	1. 在 SQData z/OS 主功能表畫面上，選擇選項 2。
	2. 在您建立的協助程式S附近輸入 以查看協助程式詳細資訊。
	3. 輸入 1，然後按 Enter 來編輯客服人員檔案。
	4. 新增您的 CAB 檔案詳細資訊。下列範例顯示名為 之 CAB 檔案的詳細資訊DB2ZTOMSK 。使用您的大型主機使用者 ID 而非 <userid>。
	`ÝDB2ZTOMSK¨ type=capture cab=/u/<userid>/sqdata/DB2ZTOMSK.cab`
	5. 按 F3。
	6. 輸入 2以編輯 ACL 檔案。將 userid新增至acl組態檔案，如下列範例所示：
	`Ýacls¨ prod=admin,<userid>`
	7. 按 F3 儲存並結束。
		大型主機系統管理員

任務	Description	所需的技能
建立任務以啟動控制器協助程式。	若要建立任務，請執行下列動作： 1. 在選項中，輸入 G。 2. 輸入 JOB 卡、任務和程序程式庫，以及Db2 load程式庫詳細資訊。 3. 輸入網路 ACL 檔案詳細資訊，然後輸入選項 2 以在指定的任務程式庫中產生任務控制語言 (JCL) 檔案。	大型主機系統管理員
產生擷取發佈者 JCL 檔案。	若要產生擷取發佈者 JCL 檔案，請執行下列動作： 1. 在 SQData z/OS 主功能表畫面上，選擇選項 3。這會帶您前往擷取/發佈摘要畫面。 2. 在 CAB 檔案S旁輸入 以選取它。這會帶您前往 Db2 Capture/Publisher 詳細資訊畫面。 3. 在選項G中，輸入選項以產生capture/publisher 任務。 4. 輸入 JOB 卡、任務和程序程式庫，以及 Db2 載入程式庫詳細資訊。 5. 若要建立任務，請選擇選項 4。任務是在任務程式庫中指定的任務程式庫中建立。	大型主機系統管理員

任務	Description	所需的技能

檢查並更新 CDC。	1. 執行下列查詢，將 <table name>變更為您的 Db2 資料表名稱，以檢查 Db2 資料表的DATACAPTURE 旗標：
	`SELECT DATACAPTU RE FROM SYSIBM.SY STABLES WHERE NAME='<table name>';`
	確認結果顯示DATACAPTU RE 為 Y。
	2. 如果 DATACAPTURE 不是 Y，請執行下列查詢以在 Db2 資料表上啟用 CDC，<table name>並變更為您的 Db2 資料表名稱：
	`ALTER TABLE <table name> DATA CAPTURE CHANGES;`
		大型主機系統管理員

提交 JCL 檔案。	提交您在先前步驟中設定的下列 JCL 檔案： • 啟動控制器協助程式的 JCL 檔案 • 開始擷取和發佈的 JCL 檔案 提交 JCL 檔案後，您可以在 EC2 執行個體上精確地啟動套用引擎。	大型主機系統管理員

執行並驗證 CDC

任務	Description	所需的技能

啟動套用引擎並驗證 CDC。	若要在 EC2 執行個體上啟動套用引擎並驗證 CDC，請執行下列動作：
	1. 若要連線至 EC2 執行個體，請遵循 AWS 文件中的指示。
	2. 變更為包含 `script.sqd` 檔案的目錄：
	`cd /home/ec2-user/precisely/scripts`
	3. 若要啟動套用引擎，請執行下列sqdeng啟動命令：
	`sqdeng -s script.sqd --identity=/home/ec2-user/.nacl/id_nacl`
	Apply Engine 會開始等待大型主機來源的更新。
	4. 若要測試 CDC，請在 Db2 資料表中進行一些記錄插入或更新。
	5. 驗證套用引擎日誌顯示擷取並寫入目標的記錄數目。
		雲端架構師、應用程式開發人員

驗證 Amazon MSK 主題上的記錄。	若要從 Kafka 主題讀取訊息，請執行下列動作： 1. 若要變更為 EC2 執行個體上 Kafka 用戶端安裝路	應用程式開發人員、雲端架構師

任務	Description	所需的技能

	徑的bin目錄，請執行下列命令，將 取代<Kafka version>為您的版本：
	
	`cd /home/ec2-user/kafka/kafka_2.13-<Kafka version>/bin`
	
	2. 若要驗證在 Kafka 主題中寫入為訊息的 Db2 CDC，請執行下列命令，<Topic Name>將<kafka broker>和 取代為您先前建立的主題：
	
	`./kafka-console-consumer.sh --bootstrap-server <kafka broker>:9098 --topic <Topic Name> --from-beginning --consumer.config client.properties`
	
	3. 驗證訊息是否符合 Db2 資料表中更新的記錄數目。	

將大型主機變更資料儲存在 Amazon Redshift Serverless 資料倉儲中

任務	Description	所需的技能
設定 Amazon Redshift Serverless。	若要建立 Amazon Redshift Serverless 資料倉儲，請遵循 AWS 文件中的指示。	資料工程師

任務	Description	所需的技能
	在 Amazon Redshift Serverless 儀表板上，驗證命名空間和工作群組已建立且可供使用。在此範例模式中，程序可能需要 2-5 分鐘。	

任務	Description	所需的技能
設定串流擷取所需的 IAM 角色和信任政策。	若要從 Amazon MSK 設定 Amazon Redshift Serverless 串流擷取，請執行下列動作： 1. 為 Amazon Redshift 建立存取 Amazon MSK 的 IAM 政策。 [region] 將 取代 AWS 區域 為適用於 Amazon MSK 的 、[account-id] 將 取代為您的 AWS 帳戶 ID，並將 [msk-cluster-name] 取代為 Amazon MSK 叢集名稱，請執行下列程式碼： {"Version": "2012-10-17", "Statement": [{"Sid": "MSKIAMpolicy","Effect": "Allow","Action": ["kafka-cluster:ReadData","kafka-cluster:DescribeTopic","kafka-cluster:Connect"],"Resource": ["arn:aws:kafka:[region]:[account-id]:cluster/[msk-cluster-name]/*","arn:aws:kafka:[region]:[account-id]:topic/[msk-	資料工程師

任務	Description	所需的技能
	`cluster-name]/*"]},{"Effect":"Allow","Action":["kafka-cluster:AlterGroup","kafka-cluster:DescribeGroup"],"Resource":["arn:aws:kafka:[region]:[account-id]:group/[msk-cluster-name]/*"]}]}` 您可以在 Amazon MSK 主控台上找到叢集名稱和 Amazon Resource Name (ARN)。在 主控台上，選擇叢集摘要，然後選擇 ARN。 2. 若要建立 IAM 角色並連接政策，請遵循AWS 文件中的指示。 3. 若要將 IAM 角色連接至 Amazon Redshift Serverless 命名空間，請執行下列動作： a. 登入 主控台，並在 https：//https://console.aws.amazon.com/redshiftv2/ 開啟 Amazon Redshift 主控台。 b. 選擇 Serverless 儀表板。 c. 選擇命名空間。 d. 選擇安全和加密索引標籤。	

任務	Description	所需的技能
	e. 選擇許可，然後連接您建立的 IAM 角色。 4. 在您的 Amazon Redshift Serverless 安全群組中，建立具有下列詳細資訊的傳入規則： • 針對 Type (類型)，請選擇 Custom TCP (自訂 TCP)。 • 針對 Protocol (通訊協定)，選擇 TCP。 • 針對連接埠範圍，選擇 9098、9198。 • 針對來源，選擇 Amazon MSK 安全群組。 5. 在您的 Amazon MSK 安全群組中，建立具有下列詳細資訊的傳入規則： • 針對 Type (類型)，請選擇 Custom TCP (自訂 TCP)。 • 針對 Protocol (通訊協定)，選擇 TCP。 • 針對連接埠範圍，選擇 9098、9198。 • 針對來源，選擇 Amazon Redshift 安全群組。 此模式使用連接埠進行 Amazon Redshift 和 Amazon MSK 組態的 IAM 身分驗證。如需詳細資訊，	

任務	Description	所需的技能
	請參閱 AWS 文件 （步驟 2)。 6. 開啟 Amazon Redshift Serverless 工作群組的增強型 VPC 路由。如需詳細資訊，請參閱 AWS 文件。	
將 Amazon Redshift Serverless 連線至 Amazon MSK。	若要連線至 Amazon MSK 主題，請在 Amazon Redshift Serverless 中建立外部結構描述。在 Amazon Redshift 查詢編輯器 v2 中，執行下列 SQL 命令，`'iam_role_arn'` 將取代為您先前建立的角色，並將 `'MSK_cluster_arn '`取代為您叢集的 ARN。 `CREATE EXTERNAL SCHEMA member_schema FROM MSK IAM_ROLE 'iam_role _arn' AUTHENTICATION iam URI 'MSK_cluster_arn';`	遷移工程師

任務	Description	所需的技能
建立具體化視觀表。	若要使用 Amazon Redshift Serverless 中 Amazon MSK 主題的資料，請建立具體化檢視。在 Amazon Redshift 查詢編輯器 v2 中，執行下列 SQL 命令，<MSK_Topic_name> 將 取代為 Amazon MSK 主題的名稱。 `CREATE MATERIALIZED VIEW member_view AUTO REFRESH YES AS SELECT kafka_partition, kafka_offset, refresh_time, json_parse(kafka_value) AS Data FROM member_schema.<MSK_Topic_name> WHERE CAN_JSON_PARSE(kafka_value);`	遷移工程師

任務	Description	所需的技能

在 Amazon Redshift 中建立目標資料表。	Amazon Redshift 資料表提供 Quick Sight 的輸入。此模式使用與大型主機上的來源 Db2 資料表`member_plans` 相符的資料表 `member_dtls` 和 。
	
	若要在 Amazon Redshift 中建立兩個資料表，請在 Amazon Redshift 查詢編輯器 v2 中執行下列 SQL 命令：
	
	`-- Table 1: members_dtls`
	`CREATE TABLE members_dtls (`
	`memberid INT ENCODE AZ64,`
	`member_name VARCHAR(100) ENCODE ZSTD,`
	`member_type VARCHAR(50) ENCODE ZSTD,`
	`age INT ENCODE AZ64,`
	`gender CHAR(1) ENCODE BYTEDICT,`
	`email VARCHAR(100) ENCODE ZSTD,`
	`region VARCHAR(50) ENCODE ZSTD`
	`) DISTSTYLE AUTO;`
	
	`-- Table 2: member_plans`
	`CREATE TABLE member_plans (`
	`memberid INT ENCODE AZ64,`
	`medical_plan CHAR(1) ENCODE BYTEDICT,`
	`dental_plan CHAR(1) ENCODE BYTEDICT,`
		遷移工程師

任務	Description	所需的技能

	<pre>vision_plan CHAR(1)
	ENCODE BYTEDICT,
	preventive_immuniz
	ation VARCHAR(50)
	ENCODE ZSTD
	) DISTSTYLE AUTO;</pre>	

在 Amazon Redshift 中建立預存程序。	此模式使用預存程序，將變更資料 (INSERT、UPDATE、DELETE) 從來源大型主機同步到目標 Amazon Redshift 資料倉儲資料表，以便在 Quick Sight 中進行分析。 若要在 Amazon Redshift 中建立預存程序，請使用查詢編輯器 v2 來執行 GitHub 儲存庫中的預存程序程式碼。	遷移工程師

任務	Description	所需的技能
從串流具體化檢視讀取並載入目標資料表。	預存程序會從串流具體化檢視讀取資料變更，並將資料變更載入目標資料表。若要執行預存程序，請使用下列命令： `call SP_Members_Load();` 您可以使用 Amazon EventBridge 來排程 Amazon Redshift 資料倉儲中的任務，以根據您的資料延遲需求呼叫此預存程序。EventBridge 會以固定間隔執行任務。若要監控先前對程序的呼叫是否已完成，您可能需要使用AWS Step Functions狀態機器等機制。如需詳細資訊，請參閱下列資源： • 建立排程執行的 Amazon EventBridge 規則 • 使用 AWS Step Functions 和 Amazon Redshift Data API 加速 ELT 程序的協調。 另一個選項是使用 Amazon Redshift 查詢編輯器 v2 來排程重新整理。如需詳細資訊，請參閱使用查詢編輯器 v2 排程查詢。	遷移工程師

將 Quick Sight 連線至 Amazon Redshift 中的資料

任務	Description	所需的技能
設定 Quick Sight。	若要設定 Quick Sight，請遵循 AWS 文件中的指示。	遷移工程師
設定 Quick Sight 和 Amazon Redshift 之間的安全連線。	若要設定 Quick Sight 與 Amazon Redshift 之間的連線，請執行下列動作 1. 若要授權從 Quick Sight 到 Amazon Redshift 的連線，請開啟 Amazon Redshift 主控台，然後在 Amazon Redshift 安全群組中新增傳入規則。此規則應允許流量從您設定 Quick Sight 的 CIDR 範圍傳入連接埠 5439（預設 Redshift 連接埠）。如需 AWS 區域 及其 IP 地址的清單，請參閱AWS 區域 支援 Quick Sight。 在 Amazon Redshift 主控台上，選擇工作群組、資料存取、網路和安全性，並啟用公開存取。	遷移工程師
建立 Quick Sight 的資料集。	若要從 Amazon Redshift 建立 Quick Sight 的資料集，請執行下列動作： 1. 在 Quick Sight 主控台的導覽窗格中，選擇資料集。 2. 在資料集頁面上，選擇新建資料集。 3. 選擇 Redshift 手動連線。	遷移工程師

任務	Description	所需的技能
	4. 在新的 Redshift 資料來源視窗中，輸入連線資訊： • 針對資料來源名稱，輸入 Amazon Redshift 資料來源的名稱。 • 針對資料庫伺服器，輸入 Amazon Redshift 叢集的端點。您可以從 Amazon Redshift Serverless 儀表板上叢集工作群組的一般資訊區段中的端點欄位取得端點值。伺服器地址是冒號之前端點的第一個部分，如下列範例所示： `mfdata-insights.NNNNNNNN.us-east-1.redshift-serverless.amazonaws.com:5439/dev` • 針對連接埠，輸入 5439(Amazon Redshift 的預設連接埠）。 • 輸入資料庫的名稱（端點中的斜線後面）。在此情況下，資料庫名稱為 `dev`。 • 針對使用者名稱和密碼，輸入 Amazon Redshift 資料庫的使用者名稱和密碼。 5. 選擇驗證連線。如果成功，您應該會看到綠色核取記	

任務	Description	所需的技能
	號，表示驗證。如果驗證失敗，請參閱故障診斷一節。 6. 選擇 Create data source (建立資料來源)。	
加入資料集。	若要在 Quick Sight 中建立分析，請依照AWS 文件中的指示加入兩個資料表。 在聯結組態窗格中，選擇左表示聯結類型。在聯結子句下，使用 `memberid from member_plans = memberid from members_details` 。	遷移工程師

使用 Quick Sight 中的 Amazon Q 從大型主機資料中取得商業洞見

任務	Description	所需的技能
在 Quick Sight 中設定 Amazon Q。	若要設定 Amazon Q in Quick Sight 生成式 BI 功能，請遵循AWS 文件中的指示。	遷移工程師
分析大型主機資料並建置視覺化儀表板。	若要在 Quick Sight 中分析和視覺化您的資料，請執行下列動作： 1. 若要建立大型主機資料分析，請遵循 AWS 文件中的指示。針對資料集，選擇您建立的資料集。 2. 在分析頁面上，選擇建置視覺效果。	遷移工程師

任務	Description	所需的技能
	3. 在建立分析主題視窗中，選擇更新現有主題。 4. 在選取主題下拉式清單中，選擇您先前建立的主題。 5. 選擇主題連結。 6. 連結主題後，請選擇建置視覺效果以開啟 Amazon Q 建置視覺效果視窗。 7. 在提示列中，撰寫您的分析問題。用於此模式的範例問題如下： • 依區域顯示成員分佈 • 依年齡顯示成員分佈 • 依性別顯示成員分佈 • 依計劃類型顯示成員分佈 • 顯示未完成預防性預防預防接種的成員 輸入問題後，請選擇建置。Amazon Q in Quick Sight 會建立視覺效果。 8. 若要將視覺效果新增至視覺效果儀表板，請選擇新增至分析。 完成後，您可以發佈儀表板，與組織中的其他人共用。如需範例，請參閱其他資訊區段中的大型主機視覺化儀表板。	

從大型主機資料使用 Quick Sight 中的 Amazon Q 建立資料案例

任務	Description	所需的技能
建立資料案例。	建立資料案例來解釋先前分析的洞見，並產生建議，以增加成員的預防性預防預防接種： 1. 若要建立資料案例，請遵循 AWS 文件中的指示。 2. 針對資料案例提示，請使用下列項目： `Build a data story about Region with most numbers of members. Also show the member distribution by medical plan, vision plan, dental plan. Recommend how to motivate members to complete immunizat ion. Include 4 points of supportin g data for this pattern.` 您也可以建立自己的提示，為其他業務洞察產生資料案例。 3. 選擇新增視覺效果，然後新增與資料案例相關的視覺效果。對於此模式，請使用您先前建立的視覺效果。 4. 選擇 Build (建置)。	遷移工程師

任務	Description	所需的技能
	5. 如需資料案例輸出的範例，請參閱其他資訊區段中的資料案例輸出。	
檢視產生的資料案例。	若要檢視產生的資料案例，請在資料案例頁面上選擇該案例。	遷移工程師
編輯產生的資料案例。	若要變更資料案例中的格式、配置或視覺效果，請遵循AWS 文件中的指示。	遷移工程師
分享資料案例。	若要分享資料案例，請遵循AWS 文件中的指示。	遷移工程師

疑難排解

問題	解決方案
對於 Quick Sight 到 Amazon Redshift 資料集建立，`Validate Connection` 已淡出。	1. 確認連接至 Amazon Redshift Serverless 執行個體的安全群組允許來自與您設定 Quick Sight 的區域相關聯 IP 地址範圍的傳入流量。 2. 確認已公開提供部署 Amazon Redshift Serverless 的 VPC。 3. 確認您使用的是 Amazon Redshift 的正確使用者名稱和密碼。您可以在 Amazon Redshift 主控台上重設使用者名稱和密碼。
嘗試在 EC2 執行個體上啟動套用引擎會傳回下列錯誤： `-bash: sqdeng: command not found`	執行下列命令以匯出`sqdata`安裝路徑： `export PATH=$PATH:/usr/sbin:/opt/precisely/di/sqdata/bin`

問題	解決方案
嘗試啟動 Apply Engine 會傳回下列其中一個連線錯誤： • `SQDD018E Cannot connect to transfer socket(rc==0x18468). Agent:<Agent Name > Socket:/u /./sqdata/.DB2ZTOMSK.cab.data` • `SQDUR06E Error opening url cdc:// <VPC end point name>:2626/ DB2ZTOMSK/DB2ZTOMSK : errno:112 8 (Unknown error 1128)`	檢查大型主機集區，確認控制器協助程式任務正在執行。

相關資源

- 使用 Quick Sight 中的 AWS Mainframe Modernization 和 Amazon Q 產生洞見（模式）
- 使用 Quick Sight 中的 AWS Mainframe Modernization 和 Amazon Q 產生資料洞見（示範）
- AWS Mainframe Modernization - IBM z/OS 的資料複寫
- Amazon Redshift 串流擷取到具體化視觀表

其他資訊

範例 .ddl 檔案

members_details.ddl

```
CREATE TABLE MEMBER_DTLS (
memberid INTEGER NOT NULL,
member_name VARCHAR(50),
member_type VARCHAR(20),
age INTEGER,
gender CHAR(1),
email VARCHAR(100),
region VARCHAR(20)
);
```

member_plans.ddl

```
CREATE TABLE MEMBER_PLANS (
memberid INTEGER NOT NULL,
medical_plan CHAR(1),
dental_plan CHAR(1),
vision_plan CHAR(1),
preventive_immunization VARCHAR(20)
);
```

範例 .sqd 檔案

將 取代 <kafka topic name>為您的 Amazon MSK 主題名稱。

script.sqd

```
-- Name: DB2ZTOMSK: DB2z To MSK JOBNAME DB2ZTOMSK;REPORT EVERY 1;OPTIONS
 CDCOP('I','U','D');-- Source Descriptions
JOBNAME DB2ZTOMSK;
REPORT EVERY 1;
OPTIONS CDCOP('I','U','D');

-- Source Descriptions
BEGIN GROUP DB2_SOURCE;
DESCRIPTION DB2SQL /var/precisely/di/sqdata/apply/DB2ZTOMSK/ddl/mem_details.ddl AS
 MEMBER_DTLS;
DESCRIPTION DB2SQL /var/precisely/di/sqdata/apply/DB2ZTOMSK/ddl/mem_plans.ddl AS
 MEMBER_PLANS;
END GROUP;
-- Source Datastore
DATASTORE cdc://<zos_host_name>/DB2ZTOMSK/DB2ZTOMSK
OF UTSCDC
AS CDCIN
DESCRIBED BY GROUP DB2_SOURCE ;
-- Target Datastore(s)
DATASTORE 'kafka:///<kafka topic name>/key'
OF JSON
AS TARGET
DESCRIBED BY GROUP DB2_SOURCE;
PROCESS INTO TARGET
SELECT
{
REPLICATE(TARGET)
```

```
}
FROM CDCIN;
```

大型主機視覺化儀表板

下列資料視覺效果是由 Amazon Q in Quick Sight 針對分析問題 所建立`show member distribution by region`。

以下資料視覺效果是由 Amazon Q in Quick Sight 為問題 建立`show member distribution by Region who have not completed preventive immunization, in pie chart`。

資料案例輸出

下列螢幕擷取畫面顯示 Amazon Q in Quick Sight 為提示 建立的資料案例區段`Build a data story about Region with most numbers of members. Also show the member distribution by age, member distribution by gender. Recommend how to motivate members to complete immunization. Include 4 points of supporting data for this pattern`。

在簡介中，資料案例建議選擇擁有最多成員的區域，以從防盜工作中獲得最大的影響。

資料案例提供四個區域的成員號碼分析。美國東北部、西南部和東南亞區域的成員最多。

資料案例會呈現各年齡的成員分析。

資料案例著重於中西部的預防工作。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

在 Quick Sight 中使用 AWS Mainframe Modernization 和 Amazon Q 產生資料洞見

Shubham Roy、Roshna Razack 和 Santosh Kumar Singh，Amazon Web Services

總結

注意： AWS Mainframe Modernization 服務（受管執行期環境體驗）不再開放給新客戶。對於類似 AWS Mainframe Modernization Service（受管執行期環境體驗）的功能，請探索 AWS Mainframe Modernization Service（自我管理體驗）。現有客戶可以繼續正常使用該服務。如需詳細資訊，請參閱AWS Mainframe Modernization 可用性變更。

如果您的組織在大型主機環境中託管業務關鍵資料，從該資料中獲得洞見對於推動成長和創新至關重要。透過解鎖大型主機資料，您可以建立更快、安全和可擴展的商業智慧，以加速 Amazon Web Services (AWS) 雲端中的資料驅動型決策、成長和創新。

此模式透過使用AWS Mainframe Modernization 檔案傳輸搭配 BMC 和 Quick Sight 中的 Amazon Q，提供產生商業洞見並從大型主機資料建立可共享敘述的解決方案。大型主機資料集會使用 AWS Mainframe Modernization 檔案傳輸搭配 BMC 傳輸至 Amazon Simple Storage Service (Amazon S3)。 AWS Lambda 函數會格式化和準備大型主機資料檔案，以載入 Quick Sight。

在 Quick Sight 中提供資料之後，您可以使用自然語言提示搭配 Amazon Q in Quick Sight 來建立資料摘要、提出問題和產生資料案例。您不需要撰寫 SQL 查詢或學習商業智慧 (BI) 工具。

商業內容

此模式提供大型主機資料分析和資料洞察使用案例的解決方案。使用 模式，您可以為公司資料建置視覺化儀表板。為了示範解決方案，此模式使用提供醫療、牙科和視覺計劃給美國成員的醫療保健公司。在此範例中，成員人口統計特性和計劃資訊會存放在大型主機資料集。視覺化儀表板會顯示下列項目：

- 區域的成員分佈
- 依性別分配成員
- 按年齡分配成員
- 依計劃類型分配成員
- 尚未完成預防性預防預防接種的成員

建立儀表板後，您會產生一個資料案例，說明先前分析的洞見。資料案例提供建議，以增加已完成預防性預防預防接種的成員人數。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 具有業務資料的大型主機資料集
- 在大型主機上安裝檔案傳輸代理程式的存取權

限制

- 您的大型主機資料檔案應該採用 Quick Sight 支援的其中一種檔案格式。如需支援的檔案格式清單，請參閱支援的資料來源。
- 此模式使用 Lambda 函數，將大型主機檔案轉換為 Quick Sight 支援的格式。

Architecture

下圖顯示透過使用 AWS Mainframe Modernization 檔案傳輸搭配 BMC 和 Amazon Q in Quick Sight 從大型主機資料產生商業洞見的架構。

該圖顯示以下工作流程：

1. 包含商業資料的大型主機資料集會使用 AWS Mainframe Modernization 檔案傳輸搭配 BMC 傳輸至 Amazon S3。
2. Lambda 函數會將檔案傳輸目的地 S3 儲存貯體中的檔案轉換為逗號分隔值 (CSV) 格式。
3. Lambda 函數會將轉換後的檔案傳送至來源資料集 S3 儲存貯體。
4. Quick Sight 會擷取 檔案中的資料。
5. 使用者存取 Quick Sight 中的資料。您可以使用 Amazon Q in Quick Sight，透過自然語言提示與資料互動。

工具

AWS 服務

- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。

- AWS Mainframe Modernization 使用 BMC 進行檔案傳輸會將大型主機資料集轉換和傳輸到 Amazon S3，以用於大型主機現代化、遷移和擴增使用案例。
- Amazon Quick Sight 是一種雲端規模的 BI 服務，可協助您在單一儀表板中視覺化、分析和報告您的資料。此模式使用 Quick Sight 中 Amazon Q 的生成式 BI 功能。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

最佳實務

- 當您使用 BMC 和 Lambda 函數建立 AWS Mainframe Modernization 檔案傳輸的 AWS Identity and Access Management (IAM) 角色時，請遵循最低權限原則。
- 確保您的來源資料集支援 Quick Sight 的資料類型。如果您的來源資料集包含不支援的資料類型，請將它們轉換為支援的資料類型。如需有關不支援的大型主機資料類型以及如何將其轉換為 Amazon Q in Quick Sight 支援的資料類型的資訊，請參閱相關資源一節。

史詩

使用 BMC 設定 AWS Mainframe Modernization 檔案傳輸

任務	Description	所需的技能
安裝 檔案傳輸代理程式。	若要安裝 AWS Mainframe Modernization 檔案傳輸代理程式，請遵循 AWS 文件中的指示。	大型主機系統管理員
建立用於大型主機檔案傳輸的 S3 儲存貯體。	建立 S3 儲存貯體以存放使用 BMC AWS Mainframe Modernization 進行檔案傳輸的輸出檔案。在架構圖中，這是檔案傳輸目的地儲存貯體。	遷移工程師
建立資料傳輸端點。	1. 建立 S3 儲存貯體來暫存輸入大型主機檔案，以便使用 BMC 進行 AWS Mainframe Modernization 檔案傳輸。	AWS Mainframe Modernization 專家

任務	Description	所需的技能
	2. 若要建立大型主機資料傳輸端點，請遵循 AWS 文件中的指示。	

轉換大型主機檔案名稱副檔名以進行 Quick Sight 整合

任務	Description	所需的技能
建立 S3 儲存貯體。	為 Lambda 函數建立 S3 儲存貯體，將轉換後的大型主機檔案從來源複製到最終目的地儲存貯體。	遷移工程師
建立 Lambda 函式。	若要建立 Lambda 函數來變更副檔名，並將大型主機檔案複製到目的地儲存貯體，請執行下列動作： 1. 登入 AWS 管理主控台，然後導覽至 AWS Lambda 主控台。 2. 選擇建立函數，然後選擇從頭開始撰寫。 3. 針對函數名稱，輸入函數的名稱。 4. 在執行時間下拉式清單中，選擇 Python.3.X。 5. 展開變更預設執行角色，然後選擇使用基本 Lambda 許可建立新角色。 6. 選擇 Create function (建立函數)。 7. 選擇程式碼索引標籤，然後貼上其他資訊區段中提	遷移工程師

任務	Description	所需的技能
	供的 `S3CopyLambda.py` Python 程式碼。Python 程式碼是使用 Microsoft Visual Studio 整合開發環境 (IDE) 中的 Amazon Q 產生的。 8. 編輯 `destination_bucket_name` 至您先前建立的 S3 儲存貯體名稱，以及`change destination_file_key` 大型主機檔案名稱。 9. 部署 Lambda 函數。	

任務	Description	所需的技能
建立 Amazon S3 觸發程序來叫用 Lambda 函數。	若要設定叫用 Lambda 函數的觸發，請執行下列動作： 1. 在 Lambda 主控台上，開啟函數頁面。 2. 選擇 Lambda 函數。 3. 在函數概觀中，選擇新增觸發。 4. 在觸發組態下拉式清單中，選擇 S3。 5. 在儲存貯體欄位中，輸入來源儲存貯體的名稱。 6. 在事件類型下拉式清單中，選擇所有物件建立事件。 7. 選取我確認不建議對輸入和輸出使用相同的 S3 儲存貯體核取方塊，然後選擇新增。 如需詳細資訊，請參閱教學課程：使用 Amazon S3 觸發條件叫用 Lambda 函數。	遷移潛在客戶

任務	Description	所需的技能
提供 Lambda 函數的 IAM 許可。	Lambda 函數需要 IAM 許可才能存取檔案傳輸目的地和來源資料集 S3 儲存貯體。透過允許檔案傳輸目的地 S3 儲存貯 體的 `s3:GetObject` 和 `s3:DeleteObject` 許可，以及來源資料集 S3 儲存貯體的 `s3:PutObject` 存取，更新與 Lambda 函數執行角色相關聯的政策。 如需詳細資訊，請參閱教學課程：使用 Amazon S3 觸發來叫用 Lambda 函數中的建立許可政策 一節。	遷移潛在客戶

定義大型主機資料傳輸任務

任務	Description	所需的技能
建立傳輸任務，將大型主機檔案複製到 S3 儲存貯體。	若要建立大型主機檔案傳輸任務，請遵循 AWS Mainframe Modernization 文件中的指示。 **Note** 將來源碼頁面編碼指定為 IBM1047，並將目標碼頁面編碼指定為 UTF-8。	遷移工程師
驗證傳輸任務。	若要驗證資料傳輸是否成功，請遵循 AWS Mainframe Modernization 文件中的指示。	遷移潛在客戶

任務	Description	所需的技能
	確認大型主機檔案位於檔案傳輸目的地 S3 儲存貯體中。	
驗證 Lambda 複製函數。	確認 Lambda 函數已啟動，且檔案已使用 .csv 副檔名複製到來源資料集 S3 儲存貯體。 Lambda 函數建立的 .csv 檔案是 Quick Sight 的輸入資料檔案。如需範例資料，請參閱附件區段中的 `Sample-data-member-healthcare-APG` 檔案。	遷移潛在客戶

將 Quick Sight 連線至大型主機資料

任務	Description	所需的技能
設定 Quick Sight。	若要設定 Quick Sight，請遵循 AWS 文件中的指示。	遷移潛在客戶
建立 Quick Sight 的資料集。	若要建立 Quick Sight 的資料集，請遵循 AWS 文件中的指示。輸入資料檔案是在您定義大型主機資料傳輸任務時建立的轉換大型主機檔案。	遷移潛在客戶

使用 Quick Sight 中的 Amazon Q 從大型主機資料中取得商業洞見

任務	Description	所需的技能
在 Quick Sight 中設定 Amazon Q。	此功能需要 Enterprise Edition。若要在 Quick Sight 中設定 Amazon Q，請執行下列動作：	遷移潛在客戶

任務	Description	所需的技能
	1. 若要取得 Amazon Q 附加元件，請遵循AWS 文件中的步驟 1：取得 Q 附加元件。 2. 若要在 Amazon Q 中使用生成式 BI 功能，請升級使用者的帳戶。遵循 AWS 文件中的指示。 3. 使用您先前建立的資料集來建立 Amazon Q 主題。遵循 AWS 文件中的指示。 4. 若要設定主題中繼資料，使其易於理解的自然語言，請遵循AWS 文件中的指示。	

任務	Description	所需的技能
分析大型主機資料並建置視覺化儀表板。	若要在 Quick Sight 中分析和視覺化您的資料，請執行下列動作： 1. 若要建立大型主機資料分析，請遵循 AWS 文件中的指示。針對資料集，選擇在上一個步驟中建立的資料集。 2. 在分析頁面上，選擇建置視覺效果。 3. 在建立分析主題視窗中，選擇更新現有主題。 4. 在選取主題下拉式清單中，選擇您先前建立的主題。 5. 選擇主題連結。 6. 連結主題後，請選擇建置視覺效果以開啟 Amazon Q 建置視覺效果視窗。 7. 在提示列中，撰寫您的分析問題。用於此模式的範例問題如下： • 依區域顯示成員分佈 • 依年齡顯示成員分佈 • 依性別顯示成員分佈 • 依計劃類型顯示成員分佈 • 顯示成員未完成預防預防預防預防 輸入問題後，請選擇建置。Amazon Q in Quick Sight 會建立視覺效果。	遷移工程師

任務	Description	所需的技能
	8. 若要將視覺效果新增至視覺效果儀表板，請選擇新增至分析。 完成後，您可以發佈儀表板，與組織中的其他人共用。如需範例，請參閱其他資訊區段中的大型主機視覺化儀表板。	

從大型主機資料使用 Quick Sight 中的 Amazon Q 建立資料案例

任務	Description	所需的技能
建立資料案例。	建立資料案例來解釋先前分析的洞見，並產生建議，以增加成員的預防性預防預防接種： 1. 若要建立資料案例，請遵循AWS 文件中的指示。 2. 針對資料案例提示，請使用下列項目： `Build a data story about Region with most numbers of members. Also show the member distribution by medical plan, vision plan, dental plan. Recommend how to motivate members to complete immunizat ion. Include 4`	遷移工程師

任務	Description	所需的技能
	points of supportin g data for this pattern. 您也可以建立自己的提示，為其他業務洞察產生資料案例。 3. 選擇新增視覺效果，然後新增與資料案例相關的視覺效果。對於此模式，請使用您先前建立的視覺效果。 4. 選擇 Build (建置)。 5. 如需資料案例輸出的範例，請參閱其他資訊區段中的資料案例輸出。	
檢視產生的資料案例。	若要檢視產生的資料案例，請遵循 AWS 文件中的指示。	遷移潛在客戶
編輯產生的資料案例。	若要變更資料案例中的格式、配置或視覺效果，請遵循AWS 文件中的指示。	遷移潛在客戶
分享資料案例。	若要分享資料案例，請遵循AWS 文件中的指示。	遷移工程師

疑難排解

問題	解決方案
無法探索在資料集搜尋條件中輸入的大型主機檔案或資料集，以使用 BMC 在 AWS Mainframe Modernization 檔案傳輸中建立傳輸任務。	1. 首先，選擇 Transfer with BMC 主控台上的資料傳輸端點來檢查連線。 AWS Mainframe Modernization 如果最後一個活動訊號時間超過兩分鐘，則尚未建立檔案傳輸的連線。如果在大型主機上執行的代理程式上次活動訊號

問題	解決方案
	時間少於 2 分鐘，則與代理程式的連線會成功。繼續進行步驟 2。 2. 檢查 AWS Secrets Manager 設定。秘密金鑰必須在 Secrets Manager 中設定，其金鑰為 `userId`（大寫 I），其值為大型主機的使用者 ID，而金鑰為 `password`，其值為大型主機密碼。`userId` 和 `password` 私密金鑰區分大小寫，必須以原狀輸入。

相關資源

若要將 PACKED-DECIMAL (COMP-3) 或 BINARY (COMP 或 COMP-4) 等大型主機資料類型轉換為 Quick Sight 支援的資料類型，請參閱下列模式：

- AWS 使用 Python 將 EBCDIC 資料轉換為 上的 ASCII
- 使用 在 Amazon S3 中將大型主機檔案從 EBCDIC 格式轉換為字元分隔 ASCII 格式 AWS Lambda

其他資訊

S3CopyLambda.py

下列 Python 程式碼是透過在 IDE 中使用 Amazon Q 的提示產生：

```
#Create a lambda function triggered by S3. display the S3 bucket name and key
import boto3
s3 = boto3.client('s3')
def lambda_handler(event, context):
print(event)
bucket = event['Records'][0]['s3']['bucket']['name']
key = event['Records'][0]['s3']['object']['key']
print(bucket, key)
#If key starts with object_created, skip copy, print "copy skipped". Return lambda with
 key value.
if key.startswith('object_created'):
print("copy skipped")
return {
'statusCode': 200,
```

```
'body': key
}
# Copy the file from the source bucket to the destination bucket.
 Destination_bucket_name = 'm2-filetransfer-final-opt-bkt'. Destination_file_key =
 'healthdata.csv'
copy_source = {'Bucket': bucket, 'Key': key}
s3.copy_object(Bucket='m2-filetransfer-final-opt-bkt', Key='healthdata.csv',
 CopySource=copy_source)
print("file copied")
#Delete the file from the source bucket.
s3.delete_object(Bucket=bucket, Key=key)
return {
'statusCode': 200,
'body': 'Copy Successful'
}
```

大型主機視覺化儀表板

下列資料視覺效果是由 Amazon Q in Quick Sight 為分析問題 所建立`show member distribution by region`。

以下資料視覺效果是由 Amazon Q in Quick Sight 為問題 建立`show member distribution by Region who have not completed preventive immunization, in pie chart`。

資料案例輸出

下列螢幕擷取畫面顯示 Amazon Q in Quick Sight 建立的資料案例區段提示 `Build a data story about Region with most numbers of members. Also show the member distribution by medical plan, vision plan, dental plan. Recommend how to motivate members to complete immunization. Include 4 points of supporting data.`

在簡介中，資料案例建議選擇最多成員的區域，以從防制措施中獲得最大的影響。

資料案例提供前三個區域的成員號碼分析，並將西南地區命名為專注於防制措施的領導區域。

> **Note**
>
> 西南部和東北部各有八個成員。不過，西南部有更多成員未完全進行預防，因此更有可能受益於提高預防接種率的計劃。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

在 AWS Blu Age 現代化大型主機應用程式中實作 Microsoft Entra ID 型身分驗證

Vishal Jaswani 和 Rimpy Tewani，Amazon Web Services

總結

> **Note**
>
> AWS Mainframe Modernization 服務（受管執行期環境體驗）不再開放給新客戶。對於類似 AWS Mainframe Modernization Service（受管執行期環境體驗）的功能，請探索 AWS Mainframe Modernization Service（自我管理體驗）。現有客戶可以繼續正常使用該服務。如需詳細資訊，請參閱AWS Mainframe Modernization 可用性變更。

使用重構模式進行現代化的大型主機應用程式，例如 AWS Mainframe Modernization Refactor with AWS Blu Age，需要謹慎地將身分驗證機制整合到新的應用程式架構。此整合通常會以後建化活動的形式處理。任務可能很複雜，通常涉及現有身分驗證系統的遷移或外部化，以符合現代安全標準和雲端原生實務。開發人員需要考慮如何在現代化應用程式執行期環境和程式庫的限制範圍內有效實作身分驗證。現代化之後， AWS 提供方法讓您更輕鬆地整合 AWS Blu Age 現代程式碼與身分和存取管理系統，例如 Amazon Cognito 和 Microsoft Entra ID （先前稱為 Azure AD)。

此模式說明如何在身分驗證提供者為 Microsoft Entra ID 時，在現代化應用程式中實作身分驗證機制，而不必花時間進行研究和試驗。模式提供：

- 來自 Microsoft Authentication Library (MSAL) 和其他 Microsoft Entra ID 文件且對身分驗證實作至關重要的欄位測試和相關角度程式庫。
- 使用 OAuth 2.0 在 AWS Blu Age Runtime 上啟用 Spring Security 所需的組態。
- 擷取已驗證使用者身分並將其傳遞至 AWS Blu Age Runtime 的程式庫。
- 我們建議實作的安全措施。
- 針對 Microsoft Entra ID 設定中常見問題的疑難排解秘訣。

> **Note**
>
> 此模式使用 AWS Blu Age OAuth 延伸程式庫，該程式庫作為其AWS 專業服務業務的一部分提供給客戶。此程式庫不屬於 AWS Blu Age 執行期。

先決條件和限制

先決條件

- 由 AWS Blu Age 大型主機現代化重構工具產生的現代化應用程式。此模式使用 CardDemo 做為範例開放原始碼大型主機應用程式。
- AWS Blu Age OAuth 延伸程式庫，由 AWS Blu Age 團隊在您與 AWS Professional Services 互動期間提供。
- 部署和測試現代化應用程式的作用中 AWS 帳戶 。
- 熟悉 AWS Blu Age 組態檔案和 Microsoft Entra ID 基本概念。

限制

- 此模式涵蓋 OAuth 2.0 身分驗證和基本字符型授權流程。進階授權案例和精細存取控制機制不在範圍內。
- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

此模式的開發方式是使用：

- AWS Blu Age Runtime 4.1.0 版 （此模式也適用於回溯相容的更新版本）
- MSAL 程式庫 3.0.23 版
- Java 開發套件 (JDK) 第 17 版
- 角度版本 16.1

Architecture

來源技術堆疊

在一般大型主機環境中，身分驗證是透過使用者設定檔實作。這些設定檔可識別系統的使用者、定義誰可以登入，並指定使用者可以在系統資源上執行哪些函數。使用者設定檔由安全主管或安全管理員管理。

目標技術堆疊

- Microsoft Entra ID
- 現代化 Java Spring 型後端
- AWS Blu Age 執行時間
- 使用 OAuth 2.0 的 Spring Security
- 角度單頁應用程式 (SPA)

目標架構

AWS Blu Age 執行時間預設支援以 OAuth 2.0 為基礎的身分驗證，因此模式會使用該標準來保護後端 APIs。

下圖說明程序流程。

> **Note**
>
> 圖表包含 Amazon Aurora 做為資料庫現代化的範例，雖然此模式的步驟中不包含 Aurora。

其中：

1. 使用者嘗試使用 Microsoft Entra ID 進行身分驗證。
2. Microsoft Entra ID 會傳回應用程式在後續呼叫中使用的重新整理、存取和 ID 字符。
3. MSAL 攔截器會在 HTTPS 請求的 `Authorization`標頭中包含存取字符，以呼叫 AWS Blu Age 執行期。
4. AWS Blu Age `extension-oauth`程式庫會使用 AWS Blu Age Runtime 組態檔案 (`application-main.yml`) 從 標頭擷取使用者資訊，並將此資訊放在 `SharedContext` 物件中，讓商業邏輯可以使用它。

   > **Note**
   >
   > `SharedContext` 是由 AWS Blu Age 提供的執行期元件，可管理現代化應用程式中的應用程式內容和狀態資訊。如需 AWS Blu Age Runtime 元件和更新的詳細資訊，請參閱 AWS Mainframe Modernization 文件中的 AWS Blu Age 版本備註。如需 `application-main.yml` 檔案的詳細資訊，請參閱 AWS Mainframe Modernization 文件中的設定 AWS Blu Age Runtime 組態。

5. AWS Blu Age Runtime 會檢查字符是否存在。
   a. 如果字符存在，它會透過與 Microsoft Entra ID 通訊來檢查字符的有效性。
   b. 如果字符不存在， AWS 則 Blu Age Runtime 會傳回 HTTP 狀態碼為 403 的錯誤。
6. 如果字符有效， AWS 則藍時間執行時間允許業務邏輯繼續。如果字符無效， AWS 則 Blu Age Runtime 會傳回 HTTP 狀態碼為 403 的錯誤。

OAuth 2.0 工作流程

如需 OAuth 2.0 工作流程的高階圖表，請參閱 Microsoft Entra 文件。

工具

AWS 服務

AWS Mainframe Modernization 提供工具和資源，協助您規劃和實作從大型主機到 AWS 受管執行期環境的遷移和現代化。您可以使用 Blu AWS Age 提供的此服務重構功能，來轉換和現代化舊版大型主機應用程式。

> **Note**
>
> AWS Mainframe Modernization 服務（受管執行期環境體驗）不再開放給新客戶。對於類似 AWS Mainframe Modernization Service（受管執行期環境體驗）的功能，請探索 AWS Mainframe Modernization Service（自我管理體驗）。現有客戶可以繼續正常使用該服務。如需詳細資訊，請參閱AWS Mainframe Modernization 可用性變更。

程式碼儲存庫

CardDemo 應用程式已更新，以示範與 Microsoft Entra ID 的整合。您可以從 GitHub 儲存庫存取此模式的程式碼。

後端組態

此模式需要變更`application-main.yml` 組態檔案，才能在後端應用程式上使用 OAuth 2.0 來啟用 Spring Security。 `.yml` 檔案如下所示：

```
gapwalk-application.security: enabled
gapwalk-application:
  security:
```

```
    identity: oauth
    issuerUri: ${issuerUrl}
    claim:
      claims:
        -
          claimName: upn
          claimMapValue: username
spring:
  autoconfigure:
    exclude:
     -
 org.springframework.boot.autoconfigure.security.oauth2.client.servlet.OAuth2ClientAutoConfigur
     -
 org.springframework.boot.autoconfigure.security.oauth2.resource.servlet.OAuth2ResourceServerAu
  security:
    oauth2:
      client:
        registration:
          azure:
            client-id: {clientId}
            client-secret: ${clientSecret}
            provider: azure
            authorization-grant-type: authorization_code
            redirect-uri: ${redirectUri}
            scope: openid

        provider:
          azure:
            authorization-uri: ${gapwalk-application.security.issuerUri}/oauth2/v2.0/
authorize
            token-uri:  ${gapwalk-application.security.issuerUri}/oauth2/v2.0/token
            jwk-set-uri: ${gapwalk-application.security.issuerUri}/discovery/v2.0/keys
      resourceserver:
        jwt:
          jwk-set-uri: ${gapwalk-application.security.issuerUri}/discovery/v2.0/keys
```

AWS Blu Age OAuth 延伸篩選條件程式庫

AWS Blu Age OAuth 延伸程式庫由 AWS Blu Age 團隊在您與 AWS Professional Services 互動期間提供。

此程式庫會讀取上一個程式碼區塊中顯示的 `application-main.yml` fie 中的`claim.claims`組態。此組態是清單。清單中的每個項目提供兩個值： `claimName`和 `claimMapValue`。

claimName代表前端傳送的 JSON Web Token (JWT) 中的金鑰名稱，而 claimMapValue是 中的金鑰名稱SharedContext。例如，如果您想要在後端擷取使用者 ID，請將 claimName設定為 JWT 中的金鑰名稱，該 JWT 會保留由 Microsoft Entra ID userId提供的 ，並將 claimMapValue設定為金鑰名稱，以在後端程式碼中擷取使用者 ID。

例如，如果您在 UserId中設定 claimMapValue，您可以使用下列程式碼來擷取使用者 ID：

```
SharedContext.get().getValue("userId", [UserId]);
```

最佳實務

在您實作此模式時，請考量下列重要安全考量。

> **Important**
>
> 此模式提供身分驗證整合的基礎。除了本節討論的安全措施之外，我們建議您在部署到生產環境之前，根據您的業務需求實作安全措施。

- AWS 組態安全性。 將敏感組態值從 application-main.yml 移至 AWS Secrets Manager。例如，使用 Secrets Manager 設定下列屬性：

  ```
  security:
      oauth2:
        client:
          registration:
            azure:
              client-id: {clientId}
              client-secret: ${clientSecret}
  ```

  如需如何使用 Secrets Manager 設定 AWS Blu Age 參數的詳細資訊，請參閱 AWS Mainframe Modernization 文件中的 AWS Blu Age Runtime 秘密。
- 執行期環境保護。使用適當的 AWS 安全控制設定現代化應用程式環境：

  ```
  server:
    tomcat:
      remoteip:
       protocol-header: X-Forwarded-Proto
       remote-ip-header: X-Forwarded-For
  ```

```
  forward-headers-strategy: NATIVE
```

- Amazon CloudWatch 記錄。考慮新增 檔案`logback-spring.xml to src/main/resources`：

```
<configuration>
 <appender name="CLOUDWATCH"
 class="com.amazonaws.services.logs.logback.CloudWatchAppender">
   <logGroup>/aws/bluage/application</logGroup>
   <logStream>${AWS_REGION}-${ENVIRONMENT}</logStream>
   <layout>
    <pattern>%d{yyyy-MM-dd HH:mm:ss} [%thread] %-5level %logger{36} - %msg%n</
pattern>
   </layout>
 </appender>

 <root level="INFO">
 <appender-ref ref="CLOUDWATCH"/>
 </root>
</configuration>
```

如需使用 CloudWatch 啟用追蹤的資訊，請參閱 CloudWatch 文件中的啟用追蹤以記錄相互關聯。

- 字符組態和處理。在 Microsoft Entra ID 中設定權杖生命週期，以符合您的安全需求。將存取權杖設定為在 1 小時內過期，並將重新整理權杖設定為在 24 小時內過期。在 AWS Blu Age Runtime 組態 (`application-main.yml`) 中，請確定 JWT 驗證已使用來自 Entra ID 應用程式註冊的確切發行者 URI 和對象值正確設定。

  當字符過期並重新整理時：

  1. 角形應用程式的錯誤攔截器會透過 MSAL 取得新權杖來處理 401 回應。
  2. 新的字符會與後續請求一起傳送。
  3. AWS Blu Age Runtime 的 OAuth 篩選條件會驗證新的權杖，並自動`SharedContext`更新目前的使用者資訊。這可確保商業邏輯繼續透過`SharedContext.get().getValue()`呼叫存取有效的使用者內容。

  如需 AWS Blu Age Runtime 元件及其更新的詳細資訊，請參閱 AWS Blu Age 版本備註。

- AWS Blu Age 執行期安全性。 AWS Blu Age 提供的`oauth2-ext`程式庫必須放置在具有適當檔案許可的正確共用目錄位置 (`{app-server-home}/shared/`)。透過檢查日誌中的`SharedContext`物件人口，確認程式庫已成功從 JWTs擷取使用者資訊。
- 特定宣告組態。在 中`application-main.yml`，明確定義您需要從 Microsoft Entra ID 取得的宣告。例如，若要擷取使用者的電子郵件和角色，請指定：

```
gapwalk-application:
  security:
    claim:
      claims:
        - claimName: upn
          claimMapValue: username
        - claimName: roles
          claimMapValue: userRoles
        - claimName: email
          claimMapValue: userEmail
```

- 錯誤處理。新增錯誤處理，以解決 Angular 應用程式中的身分驗證失敗；例如：

```
@Injectable()
export class AuthErrorInterceptor implements HttpInterceptor {
  intercept(request: HttpRequest<any>, next: HttpHandler): Observable<HttpEvent<any>>
 {
    return next.handle(request).pipe(
      catchError((error: HttpErrorResponse) => {
        if (error.status === 401) {
          // Handle token expiration
          this.authService.login();
        }
        if (error.status === 403) {
          // Handle unauthorized access
          this.router.navigate(['/unauthorized']);
        }
        return throwError(() => error);
      })
    );
  }
}
```

- 工作階段逾時組態。在 AWS Blu Age Runtime 和 Microsoft Entra ID 中設定工作階段逾時設定。例如，將下列程式碼新增至您的 `application-main.yml` 檔案：

```
server:
  servlet:
    session:
      timeout: 3600 # 1 hour in seconds
```

- MsalGuard。您必須為所有受保護的路由實作 MsalGuard 功能，以防止未經授權的存取。例如：

```
const routes: Routes = [
    { path: '', redirectTo: '/transaction-runner', pathMatch: 'full' },
    { path: 'transaction-runner', component: TransactionRunnerComponent,
 canActivate:guards },
    { path: 'user-info', component: UserInfoComponent, canActivate:guards },
    { path: 'term/:transid/:commarea', component: TermComponent,
 canActivate:guards },
 { path: 'code', component: TransactionRunnerComponent  }
];
```

沒有 MsalGuard 保護的路由無需身分驗證即可存取，可能公開敏感功能。確定所有需要身分驗證的路由在其組態中都包含防護。

史詩

設定 Microsoft Entra ID

任務	Description	所需的技能
設定 Microsoft Azure 帳戶以建立 Entra ID。	如需選項和說明，請參閱 Microsoft Azure 網站。	應用程式開發人員
在應用程式中設定 Microsoft Entra ID。	若要了解如何將 Microsoft Entra ID B2C (Azure AD B2C) 身分驗證新增至您的角度 SPA，請參閱 Microsoft 文件。具體而言： 1. 註冊您的應用程式並記錄識別符。 2. 透過新增和公開自訂範圍來公開 AWS Blu Age 交易端點。 3. 在您定義範圍和 API URL 之後，請在程式碼儲存庫中 `environment.ts` 檔案的	應用程式開發人員

任務	Description	所需的技能
	`apiUri`和 `apiScope` 屬性中設定這些值。	

複製儲存庫並部署您的 AWS Blu Age 程式碼

任務	Description	所需的技能
複製 GitHub 儲存庫以取得身分驗證所需的角度程式碼。	執行下列命令，將隨此模式提供的 GitHub 儲存庫複製到您的本機目前工作目錄： `git clone https://github.com/aws-samples/sample-microsoft-entra-id-based-auth-in-aws-bluage-modernized-mainframe-app.git`	應用程式開發人員
在 Tomcat 伺服器上部署 AWS Blu Age 現代化程式碼以實作身分驗證。	若要設定包含 Tomcat 和 Angular 開發伺服器的本機環境，請遵循 AWS Blu Age 團隊提供的安裝步驟，做為您與 AWS Professional Services 客戶互動的一部分。	應用程式開發人員

建置身分驗證解決方案

任務	Description	所需的技能
啟用 AWS Blu Age Runtime 安全性來保護 AWS Blu Age REST API 端點。	設定 AWS Blu Age Runtime 使用`application-main.yml` 的檔案，如下所示。如需此檔案的範例，請參閱此模式中稍早的程式碼儲存庫一節。	應用程式開發人員

任務	Description	所需的技能
	• `spring:security:oauth2:client:*` 啟用 Spring Security OAuth 2.0 用戶端支援。將設定取代為 Microsoft Entra ID 設定專屬的值。 • `spring:security:oauth2:resourceserver` 啟用 Spring Security OAuth 2.0 資源伺服器支援。 • `gapwalk-application.security.issuerUri` 指定用於擷取組態資訊的身分提供者 URL，例如授權端點、字符端點等。 • `gapwalk-application.security.identity` 應該設定為 oauth。 • `gapwalk-application.security.claim.claims` 指定擷取使用者 IDs 和使用者名稱所需的宣告清單。 如需範例，請參閱此模式稍早的程式碼儲存庫一節。 • `gapwalk-application.security` 應該設定為 enabled。 • `spring.autoconfigure.exclude` 應移除以啟用身分驗證。	

任務	Description	所需的技能
將來自您本機環境的範例程式碼併入您的 Blu Age 現代化角度程式碼基底。	如需有關如何將範例併入您的 AWS Blu Age 現代化角度程式碼庫的資訊，請參閱此模式稍早的程式碼儲存庫一節。	應用程式開發人員

將程式oauth2-ext 庫放在共用目錄中。	將程式oauth2-ext 庫放在應用程式伺服器的 共用目錄中，以便您的 AWS Blu Age 現代化應用程式可以使用它。 執行下列命令：
	`cd oauth2-ext/target`
	`cp extension-oauth-filter-<version>.jar /{app-server-home}/shared/`
		應用程式開發人員

部署身分驗證解決方案

任務	Description	所需的技能

部署前端應用程式。	執行下列命令，在本機啟動前端應用程式：
	`npm install`
	`ng serve --ssl`
	`npm start`
	Note
	將 `--ssl` 旗標新增至 `ng serve`命令可確保開發伺服器使用 HTTPS，這比其他通
		應用程式開發人員

任務	Description	所需的技能
	訊協定更安全，並提供更好的生產環境模擬。	
啟動後端應用程式。	在 Eclipse 中啟動 Tomcat 伺服器。	應用程式開發人員

測試應用程式。

任務	Description	所需的技能
測試登入功能。	在 存取本機部署的應用程式，`http://localhost:4200` 以確認要求使用者確認其身分。 **Note** HTTP 在此處用於示範目的。在生產或其他可公開存取的環境中，為了安全起見，您必須使用 HTTPS。即使是本機開發，我們建議您盡可能設定 HTTPS。 應該會出現 Microsoft 登入提示，而且應該允許在 Microsoft Entra ID 中設定的使用者存取應用程式。	應用程式開發人員

任務	Description	所需的技能
在請求中測試授權標頭。	**Note** 下列步驟使用 CardDemo 應用程式做為範例。其他現代應用程式的測試步驟會有所不同。 1. CC00 在 CardDemo 應用程式中啟動交易。 2. 使用使用者登入資料登入。例如，如果您使用的是 AWS Blu Age Level 3 自定進度研討會，則可以使用PASSWORD登入資料的名稱USER0001和密碼。 3. 在瀏覽器的相同索引標籤中開啟開發人員工具視窗。 4. 開啟網路索引標籤，並檢查傳送至後端 的請求`http://localhost:8080/gapwalk-application/transaction` 。 5. 檢查請求標頭，並確認您是否看到具有 值的授權標頭`Bearer <token>`，其中 `<token>` 是 Microsoft Entra ID 產生的值。	應用程式開發人員
測試登出功能。	選擇退出以登出，然後嘗試再次存取應用程式。它應該會顯示新的登入提示。	應用程式開發人員

疑難排解

問題	解決方案
Microsoft Entra ID 發行的權杖與 Spring Boot OAuth 2.0 安全性不相容。	如需問題的解決方法，請參閱 OAuth 部落格上的 Microsoft Entra ID OAuth 流程。 OAuth
一般字符相關問題。	若要解碼和檢視 JWT 權杖的內容，請使用 https://jwt.io/ 網站。

相關資源

- 如需有關使用 AWS Blu Age 重構應用程式的資訊，請參閱 AWS Mainframe Modernization 文件。
- 若要了解 OAuth 2.0 的運作方式，請參閱 OAuth 2.0 網站。
- 如需 Microsoft Authentication Library (MSAL) 的概觀，請參閱 Microsoft Entra 文件。
- 如需 AS/400 系統上使用者設定檔的資訊，請參閱 IBM i (AS400) 教學課程。
- 如需 Microsoft 身分平台中的 OAuth 2.0 和 OpenID Connect (OIDC) 身分驗證流程，請參閱 Microsoft Entra 文件。

將stonebranch 通用控制器與 AWS Mainframe Modernization 整合

Vaidy Sankaran 和 Pablo Alonso Prieto，Amazon Web Services

Robert Lemieux 和 Huseyin Gomleksizoglu，斯通堡

總結

注意： AWS Mainframe Modernization 服務（受管執行期環境體驗）不再開放給新客戶。對於類似 AWS Mainframe Modernization Service（受管執行期環境體驗）的功能，請探索 AWS Mainframe Modernization Service（自我管理體驗）。現有客戶可以繼續正常使用該服務。如需詳細資訊，請參閱AWS Mainframe Modernization 可用性變更。

此模式說明如何將stonebranch Universal Automation Center (UAC) 工作負載協調與 Amazon Web Services (AWS) Mainframe Modernization 服務整合。AWS Mainframe Modernization 服務會將大型主機應用程式遷移和現代化至 AWS 雲端。它提供兩種模式：AWS Mainframe Modernization Replatform with Micro Focus Enterprise technology 和 AWS Mainframe Modernization Automated Refactor with AWS Blu Age。

stonebranch UAC 是即時 IT 自動化和協同運作平台。UAC 旨在自動化和協調混合式 IT 系統的任務、活動和工作流程，從內部部署到 AWS。使用大型主機系統的企業用戶端正在轉換至以雲端為中心的現代化基礎設施和應用程式。stonebranch 的工具和專業服務有助於將現有的排程器和自動化功能遷移至 AWS 雲端。

當您使用 AWS Mainframe Modernization 服務將大型主機程式遷移或現代化至 AWS 雲端時，您可以使用此整合來自動化批次排程、提高靈活性、改善維護並降低成本。

此模式提供整合stonebranch 排程器與遷移至 AWS Mainframe Modernization 服務 Micro Focus Enterprise 執行期之大型主機應用程式的指示。此模式適用於解決方案架構師、開發人員、顧問、遷移專家，以及其他在遷移、現代化、操作或 DevOps 工作的人員。

目標成果

此模式著重於提供下列目標結果：

- 能夠排程、自動化和執行從stonebranch Universal Controller 在 AWS Mainframe Modernization 服務 (Microfocus 執行時間）中執行的大型主機批次任務。
- 從stonebranch 通用控制器監控應用程式的批次程序。
- Start/Restart/Rerun/Stop 批次處理會自動或從stonebranch 通用控制器手動處理。

- 擷取 AWS Mainframe Modernization 批次程序的結果。
- 在stonebranch Universal Controller 中擷取批次任務的 AWS CloudWatch 日誌。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 具有任務控制語言 (JCL) 檔案的 Micro Focus Bankdemo 應用程式，以及部署在 AWS Mainframe Modernization 服務 (Micro Focus 執行時間） 環境中的批次程序
- 如何建置和部署在 Micro Focus Enterprise Server 上執行之大型主機應用程式的基本知識
- 對stonebranch Universal Controller 的基本了解
- 斯通堡試用授權 （ 請聯絡斯通堡)
- Windows 或 Linux Amazon Elastic Compute Cloud (Amazon EC2) 執行個體 （ 例如 xlarge)，至少包含四個核心、8 GB 記憶體和 2 GB 磁碟空間
- Apache Tomcat 8.5.x 或 9.0.x 版
- Oracle Java 執行期環境 (JRE) 或 OpenJDK 版本 8 或 11
- Amazon Aurora MySQL 相容版本
- 匯出儲存庫的 Amazon Simple Storage Service (Amazon S3) 儲存貯體
- Amazon Elastic File System (Amaon EFS) 適用於代理程式stonebranch Universal Message Service (OMS) 連線，以實現高可用性 (HA)
- stonebranch 通用控制器 7.2 通用代理程式 7.2 安裝檔案
- AWS Mainframe Modernization 任務排程範本 （ 最新發行的 .zip 檔案版本 ）

限制

- 產品和解決方案已經過測試，且僅通過 OpenJDK 8 和 11 的相容性驗證。
- aws-mainframe-modernization-stonebranch-integration 任務排程範本僅適用於 AWS Mainframe Modernization 服務。
- 此任務排程範本僅適用於 Gem Branch 代理程式的 Unix、Linux 或 Windows 版本。
- 並非所有 AWS 區域都提供某些 AWS 服務。如需區域可用性，請參閱依區域的 AWS 服務。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

Architecture

目標狀態架構

下圖顯示此試驗所需的 AWS 環境範例。

1. stonebranch Universal Automation Center (UAC) 包含兩個主要元件：Universal Controller 和 Universal Agents。stonebranch OMS 用作控制器和個別代理程式之間的訊息匯流排。
2. 通用控制器使用stonebranch UAC 資料庫。資料庫可以是 MySQL、Microsoft SQL Server、Oracle 或 Aurora MySQL 相容。
3. AWS Mainframe Modernization 服務 – 已部署 BankDemo 應用程式的 Micro Focus 執行期環境。BankDemo 應用程式檔案會存放在 S3 儲存貯體中。此儲存貯體也包含大型主機 JCL 檔案。
4. stonebranch UAC 可以針對批次執行執行下列函數：
   a. 使用與 AWS 大型主機現代化服務連結的 S3 儲存貯體中存在的 JCL 檔案名稱啟動批次任務。
   b. 取得批次任務執行的狀態。
   c. 等待批次任務執行完成。
   d. 擷取批次任務執行的日誌。
   e. 重新執行失敗的批次任務。
   f. 在任務執行時取消批次任務。
5. stonebranch UAC 可以為應用程式執行下列函數：
   a. 啟動應用程式
   b. 取得應用程式的狀態
   c. 等待應用程式啟動或停止
   d. 停止應用程式
   e. 應用程式操作的擷取日誌

stonebranch 任務轉換

下圖代表斯通堡在現代化旅程期間的任務轉換程序。它說明如何將任務排程和任務定義轉換為可執行 AWS Mainframe Modernization 批次任務的相容格式。

1. 對於轉換程序，任務定義會從現有的大型主機系統匯出。

2. JCL 檔案可以上傳到 Mainframe Modernization 應用程式的 S3 儲存貯體，以便 AWS Mainframe Modernization 服務可以部署這些 JCL 檔案。
3. 轉換工具會將匯出的任務定義轉換為 UAC 任務。
4. 建立所有任務定義和任務排程後，這些物件將匯入通用控制器。轉換後的任務接著會在 AWS Mainframe Modernization 服務中執行程序，而不是在大型主機上執行。

stonebranch UAC 架構

下列架構圖代表高可用性 (HA) 通用控制器的active-active-passive模型。stonebranch UAC 部署在多個可用區域中，以提供高可用性並支援災難復原 (DR)。

通用控制器

兩個 Linux 伺服器佈建為通用控制器。兩者都連接到相同的資料庫端點。每個伺服器都有一個通用控制器應用程式和 OMS。最新版本的 Universal Controller 會在佈建時使用。

通用控制器部署在 Tomcat Webapp 中做為文件 ROOT，並在連接埠 80 上提供。此部署可簡化前端負載平衡器的組態。

HTTP over TLS 或 HTTPS 是使用stonebranch 萬用字元憑證啟用的 （ 例如，`https://customer.stonebranch.cloud`)。這可保護瀏覽器與應用程式之間的通訊。

OMS

通用代理程式和 OMS (Opswise Message Service) 位於每個通用控制器伺服器上。所有從客戶端部署的通用代理程式都會設定為連線至這兩個 OMS 服務。OMS 做為通用代理程式和通用控制器之間的常見傳訊服務。

Amazon EFS 會在每個伺服器上掛載多工緩衝處理目錄。OMS 使用此共用多工緩衝處理目錄來保留控制器和代理程式的連線和任務資訊。OMS 可在高可用性模式中運作。如果作用中 OMS 故障，被動 OMS 可以存取所有資料，並自動恢復作用中的操作。通用代理程式會偵測此變更，並自動連線至新的作用中 OMS。

資料庫

Amazon Relational Database Service (Amazon RDS) 存放 UAC 資料庫，Amazon Aurora MySQL 相容作為其引擎。Amazon RDS 有助於定期管理和提供排程備份。兩個通用控制器執行個體都連接到相同的資料庫端點。

負載平衡器

Application Load Balancer 會為每個執行個體設定。負載平衡器會在任何指定時間將流量導向作用中的控制器。您的執行個體網域名稱會指向個別的負載平衡器端點。

URLs

每個執行個體都有一個 URL，如下列範例所示。

Environment	執行個體
生產	`customer.stonebranch.cloud`
開發（非生產）	`customerdev.stonebranch.cloud`
測試（非生產）	`customertest.stonebranch.cloud`

> **Note**
>
> 您可以根據您的需求設定非生產執行個體名稱。

高可用性

高可用性 (HA) 是系統在指定期間內持續運作而不會失敗的能力。此類故障包括但不限於儲存體、CPU 或記憶體問題所造成的伺服器通訊回應延遲，以及網路連線能力。

若要符合 HA 要求：

- 所有 EC2 執行個體、資料庫和其他組態都會鏡像到相同 AWS 區域內的兩個不同可用區域。
- 控制器是透過 Amazon Machine Image (AMI) 在兩個可用區域中的兩個 Linux 伺服器上佈建。例如，如果您在歐洲 eu-west-1 區域中佈建 ，則您在可用區域 eu-west-1a 和可用區域 eu-west-1c 中有通用控制器。
- 不允許任何任務直接在應用程式伺服器上執行，也不允許將資料存放在這些伺服器上。
- Application Load Balancer 會在每個通用控制器上執行運作狀態檢查，以識別作用中的控制器，並將流量引導至該控制器。如果一個伺服器發生問題，負載平衡器會自動將被動通用控制器提升為作用中狀態。然後，負載平衡器會從運作狀態檢查中識別新的作用中 Universal Controller 執行個體，並開始引導流量。容錯移轉會在四分鐘內發生，不會遺失任務，且前端 URL 保持不變。
- Aurora MySQL 相容資料庫服務存放通用控制器資料。對於生產環境，資料庫叢集會在單一 AWS 區域內的兩個不同可用區域中建置兩個資料庫執行個體。兩個通用控制器都使用指向單一資料庫叢集端

點的 Java Database Connectivity (JDBC) 介面。如果一個資料庫執行個體發生問題，資料庫叢集端點會動態指向運作狀態良好的執行個體。不需要手動介入。

備份和清除

stonebranch Universal Controller 設定為遵循表格中顯示的排程來備份和清除舊資料。

Type	Schedule
活動	7 天
稽核	90 天
歷程記錄	60 天

早於顯示日期的備份資料會匯出為 .xml 格式，並存放在檔案系統中。備份程序完成後，較舊的資料會從資料庫清除，並存檔在 S3 儲存貯體中長達一年的生產執行個體。

您可以在通用控制器界面中調整此排程。不過，增加這些時間範圍可能會在維護期間造成更長的停機時間。

工具

AWS 服務

- AWS Mainframe Modernization 是一種 AWS 雲端原生平台，可協助您將大型主機應用程式現代化為 AWS 受管執行期環境。它提供工具和資源來協助您規劃和實作遷移和現代化。
- Amazon Elastic Block Store (Amazon EBS) 提供區塊層級儲存體磁碟區，可搭配使用 Amazon EC2 執行個體。
- Amazon Elastic File System (Amazon EFS) 可協助您在 AWS 雲端中建立和設定共用檔案系統。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。此模式使用 Amazon Aurora MySQL 相容版本。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Elastic Load Balancing (ELB) 會將傳入的應用程式或網路流量分散到多個目標。例如，您可以在一或多個可用區域中跨 Amazon EC2 執行個體、容器和 IP 地址分配流量。此模式使用 Application Load Balancer。

斯通布蘭奇

- Universal Automation Center (UAC) 是企業工作負載自動化產品的系統。此模式使用下列 UAC 元件：
  - Universal Controller 是在 Tomcat Web 容器中執行的 Java Web 應用程式，是 Universal Automation Center 的企業任務排程器和工作負載自動化代理程式解決方案。控制器提供用於建立、監控和設定控制器資訊的使用者介面；處理排程邏輯；處理所有往返 Universal Agents 的訊息；以及同步 Universal Automation Center 的大部分高可用性操作。
  - Universal Agent 是獨立於廠商的排程代理程式，可在所有主要運算平台上與現有的任務排程器協同合作，包括舊版和分散式。支援在 z/Series、i/Series、Unix、Linux 或 Windows 上執行的所有排程器。
- Universal Agent 是獨立於廠商的排程代理程式，可在所有主要運算平台上與現有的任務排程器協同合作，包括舊版和分散式。支援在 z/Series、i/Series、Unix、Linux 或 Windows 上執行的所有排程器。
- stonebranch aws-mainframe-modernization-stonebranch-integration AWS Mainframe Modernization Universal Extension 是在 AWS Mainframe Modernization 平台中執行、監控和重新執行批次任務的整合範本。

Code

此模式的程式碼可在 aws-mainframe-modernization-stonebranch-integration GitHub 儲存庫中使用。

史詩

在 Amazon EC2 上安裝通用控制器和通用代理程式

任務	Description	所需的技能
下載安裝檔案。	從stonebranch 伺服器下載 安裝。若要取得安裝檔案，請聯絡stonebranch。	雲端架構師
啟動 EC2 執行個體。	您將需要大約 3 GB 的額外空間來安裝通用控制器和通用代理程式。因此，請為執行個體提供至少 30 GB 的磁碟空間。	雲端架構師

任務	Description	所需的技能
	將連接埠 8080 新增至安全群組，以便存取。	

任務	Description	所需的技能

檢查先決條件。	在安裝之前，請執行下列動作：
	
	1. 如下載 Java 執行期環境所述安裝 Java。
	
	``
	$ sudo yum -y update
	$ sudo yum install
	 java-11-amazon-cor
	retto
	``
	
	請務必使用其中一個支援的 JAVA 版本。先前的命令應該安裝 java-11。檢查 Java 版本，並確認您正在使用版本 11，然後再繼續。
	2. 如安裝 Apache Tomcat 中所述，執行下列命令。
	
	``
	$ sudo yum install
	 tomcat tomcat-admin-
	webapps
	$ sudo systemctl
	 enable tomcat
	$ sudo systemctl start
	 tomcat
	``
	
	3. 如建立和連線至 Aurora MySQL 資料庫叢集中所述，建立 Amazon Aurora 資料庫。使用 Amazon Aurora MySQL 相容版本。
	
	選擇主要使用者名稱和主要密碼。保留其餘設定的預設值。
		雲端管理員、Linux 管理員

任務	Description	所需的技能

安裝通用控制器。	1. 將universal-controller-7.2.0.0.tar 安裝檔案上傳至 EC2 執行個體。
	2. 將安裝檔案取消封存至temp資料夾。
	`$ tar -xvf universal-controller-7.2.0.0.tar`
	3. 提供安裝指令碼執行許可。
	`$ chmod a+x install-controller.sh`
	4. 安裝控制器。此範例使用下列命令在 /usr/share/tomcat 下安裝 Universal Controller。使用您在先前步驟中建立的 Amazon Aurora 資料庫。
	`$ sudo ./install-controller.sh --tomcat-dir /usr/share/tomcat/ --controller-file universal-controller-7.2.0.0-build.145.war --dbuser admin --dbpass "****" --dbname uc --rdbms mysql --dburl jdbc:mysql://database-2-instance-1.cih63miincgy.us-east-1.rds.amazonaws.com:3306/`
		雲端架構師、Linux 管理員

任務	Description	所需的技能
	指令碼輸出的最後一行應該是「安裝完成」。 5. 導覽至 EC2 執行個體中的下列 URL。 `http://<public_ip>:8080/uc` 6. 在登入畫面上，在使用者名稱區段中輸入 ops.admin，並將密碼欄位保持空白。 7. 為ops.admin 使用者設定新密碼。	

任務	Description	所需的技能

安裝 Universal Agent。	1. 將sb-7.2.0.1-linux-3.10-x86_64.tar.Z 安裝檔案上傳至 EC2 執行個體。
	2. 登入 EC2 執行個體。
	3. 取消封存 Universal Agent 安裝套件。
	`$ zcat sb-7.2.0.1-linux-3.10-x86_64.tar.Z | tar xvf -`
	4. 執行下列命令。
	`$ sudo ./unvinst --oms_servers 7878@localhost --oms_autostart yes --python yes`
	5. 建立 PAM 檔案。
	`$ cp /etc/pam.d/login /etc/pam.d/ucmd`
	6. 啟用通用代理程式的自動啟動。
	`$ /sbin/restorecon -v /etc/rc.d/init.d/ubrokerd`
		雲端管理員、Linux 管理員

任務	Description	所需的技能
將 OMS 新增至通用控制器。	1. 使用 `ops.admin` 使用者登入 Universal Controller。 2. 選擇畫面左上角的服務選單，然後在系統中選擇 OMS 伺服器選單 3. 在 OMS 伺服器地址欄位中，輸入 `localhost` ，然後儲存。 4. 您將看到 OMS 伺服器的狀態為已連線，工作階段狀態為可操作。	通用控制器管理員

匯入 AWS Mainframe Modernization Universal Extension 並建立任務

任務	Description	所需的技能
匯入整合範本。	在此步驟中，您需要 AWS Mainframe Modernization Universal Extension。確保已下載最新版本的 .zip 檔案。 1. 使用 `ops.admin` 使用者登入通用控制器。 2. 導覽至服務、匯入整合範本。 3. 選取整合範本 .zip 檔案 (`aws_mainframe_modernization_stonebranch_extension.zip`)，然後選擇匯入。	通用控制器管理員

任務	Description	所需的技能
	匯入整合範本後，您會在可用服務下看到 AWS Mainframe Modernization Tasks。	

任務	Description	所需的技能
啟用可解析的登入資料。	1. 導覽至服務、AWS 大型主機現代化任務。 2. 在右側面板上，填寫必要欄位： • 名稱：新的大型主機現代化任務 • 代理程式：選取唯一的代理程式 (AGNT0001)。 在 AWS Mainframe Modernization 詳細資訊下： • 動作：列出環境 • AWS 登入資料：如果您已將 AWS Identity and Access Management (IAM) 角色新增至 EC2 執行個體，您可以將此欄位保留空白。如果您將使用 `AWSAccessKeyID` 和 `AWSSecretKey` ，請選擇 欄位旁的圖示 ()。 在開啟的登入資料詳細資訊視窗中，輸入下列資訊，然後儲存。 • 名稱：AWS Mainframe Modernization Credentials • 執行時間使用者：在此欄位中寫入 AWS 存取金鑰 ID。	通用控制器管理員

任務	Description	所需的技能
	• 執行期密碼：在此欄位中寫入 AWS 私密金鑰。 • 結束點： 確定端點具有正確的 AWS 區域。預設值為 `https://m2.us-east-1.amazonaws.com` 。 • 區域：輸入 AWS Mainframe Modernization 服務的 區域。預設值為 `us-east-1` 。 3. 將預設值保留在其餘欄位，並儲存任務。	

任務	Description	所需的技能
啟動任務。	1. 在右面板頂端，選擇啟動任務。 2. 在確認視窗中，選擇啟動。之後，通用控制器主控台會顯示類似下列訊息的訊息。 2022-08-24 上午 10：11：49 使用任務執行個體 sys_id 166129149363414631 3NC8E38DB8OZJY 成功啟動通用任務「新大型主機現代化任務」。 3. 導覽至執行個體 如果您沒有看到執行個體索引標籤，請選擇向右箭頭以向右捲動。 4. 在清單中開啟任務執行個體的內容（按一下滑鼠右鍵）選單，選擇擷取輸出，然後在擷取輸出中選擇提交 5. 在擷取輸出視窗中，您會在 STDOUT 中看到環境清單。	通用控制器管理員

測試開始批次任務

任務	Description	所需的技能
為批次任務建立任務。	1. 導覽至服務、AWS 大型主機現代化任務。 2. 在右側面板上，填寫必要欄位：	通用控制器管理員

任務	Description	所需的技能
	• 名稱：新的大型主機現代化任務 • 代理程式：選取唯一的代理程式 (AGNT0001)。 在 AWS Mainframe 現代化詳細資訊下： • 動作：啟動批次（或啟動批次並等待執行批次任務，並等待任務在 AWS 中完成） • AWS 登入資料：如果您已將 IAM 角色新增至 EC2 執行個體，您可以將此欄位保留空白。如果您要使用 `AWSAccessKeyID` 和 `AWSSecretKey` ，請選擇 欄位旁的圖示 ()。 • 結束點： 確定端點具有正確的 AWS 區域。預設值為 https://m2.us-east-1.amazonaws.com。 • 區域：輸入 AWS Mainframe Modernization 服務的 區域。預設值為 `us-east-1` 。 • 應用程式：選擇欄位 () 旁的圖示，然後在重新整理應用程式選擇中選擇提交。這將連接到 AWS Mainframe Modernization 服務，並傳回應用程式清單。現在，您可以從下拉	

任務	Description	所需的技能
	式清單中選取應用程式。選取您要執行批次任務的應用程式。 • JCL 檔案名稱：RUNHELLO.jcl • 等待成功或失敗： 如果選取此選項，任務將等待批次任務的狀態成功或失敗。 • 輪詢間隔：這是每次輪詢之間的時間量。 • 擷取執行日誌：如果選取，批次任務完成時會自動擷取日誌。 • 日誌格式：這是要列印的日誌格式。它可以是文字或 JSON 格式。 3. 將預設值保留在其餘欄位，並儲存任務。	

任務	Description	所需的技能
啟動任務。	1. 在右面板頂端，選擇啟動任務。 2. 在確認視窗中，選擇啟動。之後，通用控制器主控台會顯示類似下列訊息的訊息。 2022-08-24 上午 11：11：59 使用任務執行個體 sys_id <sys id> 成功啟動通用任務「主機現代化開始批次」。 3. 導覽至執行個體 如果您沒有看到執行個體索引標籤，請選擇向右箭頭以向右捲動。 4. 在清單中開啟任務執行個體的內容（按一下滑鼠右鍵）選單，選擇擷取輸出，然後在擷取輸出中選擇提交 5. 在擷取輸出視窗中，您會在 STDOUT 中看到環境清單。	通用控制器管理員

為多個任務建立工作流程

任務	Description	所需的技能
複製任務。	1. 開啟您要建立複本之任務的內容（按一下滑鼠右鍵）選單，然後選擇複製。 2. 在複製 AWS Mainframe Modernization Task 視窗中，輸入新任務的下列新名稱：Mainframe	通用控制器管理員

任務	Description	所需的技能
	Modernization Start Batch - RUNAWS2。 3. 使用以下名稱再次複製任務：Mainframe Modernization Start Batch - RUNAWS3。 4. 使用下列名稱再次使用任務複製 ：大型主機現代化開始批次 - RUNAWS4。 5. 使用以下名稱，最後一次複製任務：大型主機現代化開始批次 - FOOBAR。	

任務	Description	所需的技能
更新任務。	1. 開啟（按兩下）Mainframe Modernization Start Batch - RUNAWS2 任務，將 JCL 檔案名稱欄位變更為 `RUNAWS2.jcl` ，然後儲存。 2. 開啟（按兩下）Mainframe Modernization Start Batch - RUNAWS3 任務，將 JCL 檔案名稱欄位變更為 `RUNAWS3.jcl` ，然後儲存。 3. 開啟（按兩下）Mainframe Modernization Start Batch - RUNAWS4 任務，將 JCL 檔案名稱欄位變更為 `RUNAWS4.jcl` ，然後儲存。 4. 開啟（按兩下）Mainframe Modernization Start Batch - FOOBAR 任務，將 JCL 檔案名稱欄位變更為 `MISSING.jcl` ，然後儲存。此任務會失敗，因為 JCL 檔案名稱值不正確。	通用控制器管理員

任務	Description	所需的技能
建立工作流程。	1. 導覽至服務、工作流程。 2. 在右側面板上，在名稱欄位中輸入大型主機現代化工作流程，然後儲存。 3. 在右側面板中，選擇編輯工作流程。 4. 在工作流程編輯器索引標籤上，新增任務按鈕 (+)。 5. 在任務尋找視窗中，選擇搜尋以查看通用控制器中的所有任務。 6. 按一下大型主機現代化開始批次任務旁的圖示，然後將圖示拖曳到工作流程編輯器中的空白位置。 7. 針對其他大型主機現代化任務重複相同的動作，並放置它們，如其他資訊一節所示。 8. 選擇連線按鈕 ()，然後將任務連接在一起。若要將任務與另一個任務連線，請按一下任務中間的 ，並將其拖曳至目標任務。 9. 如其他資訊區段所示連接任務，並儲存工作流程。 10.在工作流程編輯器中的空白位置按一下滑鼠右鍵，選擇啟動工作流程，然後選擇確定。	通用控制器管理員

任務	Description	所需的技能
檢查工作流程的狀態。	1. 在左側選單中，選擇活動 2. 在視窗中間，選擇開始。 您會在清單中看到任務執行個體的清單。 3. 在清單中開啟（按兩下）大型主機現代化工作流程，或開啟內容（按一下滑鼠右鍵）選單，然後選擇工作流程任務命令、檢視工作流程。 您將看到任務，如其他資訊一節所示。第二個任務預期會失敗，因為您使用缺少的 JCL 檔案。	Univeral Controller 管理員

故障診斷失敗的批次任務並重新執行

任務	Description	所需的技能
修正失敗的任務並重新執行。	1. 開啟（按兩下）失敗的任務，以查看任務的錯誤。 2. 您有兩個選項可以修正失敗的任務。 • 修正 JCL 檔案名稱，並將其設定為 `FOOBAR.jcl` 。 • 將正確的 JCL 檔案名稱新增至 JCL 檔案名稱 (Temp)。此欄位會覆寫 JCL 檔案名稱欄位。	通用控制器管理員

任務	Description	所需的技能
	在此試行中，選擇第二個選項，然後儲存任務執行個體。 3. 在工作流程監視器中，開啟失敗任務的內容（按一下滑鼠右鍵）選單，然後選擇命令、重新執行。 4. 之後，所有任務都會成功完成。	

建立啟動應用程式和停止應用程式任務

任務	Description	所需的技能
建立啟動應用程式動作。	1. 導覽至服務、AWS 大型主機現代化任務。 2. 在右側面板上，填寫必要欄位。 • 名稱：大型主機現代化啟動應用程式 • 代理程式：選取唯一的代理程式 (AGNT0001) 在 AWS Mainframe 現代化詳細資訊下： • 動作： 啟動應用程式 • AWS 登入資料：如果您已將 IAM 角色新增至 EC2 執行個體，您可以將此欄位保留空白。如果您將使用 `AWSAccessKeyID` 和 `AWSSecret`	通用控制器管理員

任務	Description	所需的技能
	Key ，請選取您之前建立的登入資料。 • 結束點： 確定端點具有正確的區域。預設值為 https://m2.us-east-1.amazonaws.com 。 • 區域：輸入 AWS Mainframe Modernization 服務的 區域。預設值為 us-east-1 。 • 應用程式：選擇欄位 () 旁的圖示，然後在重新整理應用程式選擇中選擇提交。這將連接到 AWS Mainframe Modernization 服務，並傳回應用程式清單。現在，您可以從下拉式清單中選取應用程式。選取您要執行批次任務的應用程式。 • 等待成功或失敗： 如果選取此選項，任務將等待批次任務的狀態成功或失敗。 • 輪詢間隔：這是每次輪詢之間的時間量。 • 擷取執行日誌：如果選取，會在批次任務完成時自動擷取日誌。 • 日誌格式：這是要列印的日誌格式。它可以是文字或 JSON 格式。	

任務	Description	所需的技能
	3. 將預設值保留在其餘欄位，並儲存任務。 4. 現在複製此任務並建立停止應用程式的任務。將名稱變更為大型主機模式化停止應用程式，並將動作變更為停止應用程式。	

建立取消批次執行任務

任務	Description	所需的技能
建立取消批次動作。	1. 導覽至服務、AWS 大型主機現代化任務。 2. 在右側面板上，填寫必要欄位。 • 名稱：大型主機現代化取消批次執行 • 代理程式：選取唯一的代理程式 (AGNT0001) 在 AWS Mainframe Modernization 詳細資訊下： • 動作： 取消批次執行 • AWS 登入資料：如果您已將 IAM 角色新增至 EC2 執行個體，您可以將此欄位保留空白。如果您將使用 `AWSAccessKeyID` 和 `AWSSecretKey` ，請選取您之前建立的登入資料。	

任務	Description	所需的技能
	• 結束點： 確定端點具有正確的區域。預設值為 `https://m2.us-east-1.amazonaws.com` 。 • 區域：輸入 AWS Mainframe Modernization 服務的 區域。預設值為 `us-east-1` 。 • 應用程式：選擇欄位 () 旁的圖示，然後在重新整理應用程式選擇中選擇提交。這將連接到 AWS Mainframe Modernization 服務，並傳回應用程式清單。現在，您可以從下拉式清單中選取應用程式。選取您要執行批次任務的應用程式。 • 等待成功或失敗： 如果選取此選項，任務會等到批次任務的狀態成功或失敗。 • 輪詢間隔：這是每次輪詢之間的時間量。 • 擷取執行日誌：如果選取，批次任務完成時會自動擷取日誌。 • 日誌格式：這是要列印的日誌格式。它可以是文字或 JSON 格式。 3. 將預設值保留在其餘欄位，並儲存任務。	

相關資源

- 通用控制器
- 通用代理程式
- LDAP 設定
- SAML 單一登入
- Xpress 轉換工具

其他資訊

工作流程編輯器中的圖示

所有已連線的任務

工作流程狀態

使用來自 Precisely 的 Connect 將 VSAM 檔案遷移和複寫至 Amazon RDS 或 Amazon MSK

Prachi Khanna 和 Bopath GOPALSAMY，Amazon Web Services

總結

此模式說明如何使用 Connect https://www.precisely.com/product/precisely-connect/connect from Precisely，將虛擬儲存存取方法 (VSAM) 檔案從大型主機遷移並複寫至 AWS 雲端中的目標環境。此模式涵蓋的目標環境包括 Amazon Relational Database Service (Amazon RDS) 和 Amazon Managed Streaming for Apache Kafka (Amazon MSK)。Connect 使用變更資料擷取 (CDC) 持續監控來源 VSAM 檔案的更新，然後將這些更新傳輸到一或多個 AWS 目標環境。您可以使用此模式來滿足您的應用程式現代化或資料分析目標。例如，您可以使用 Connect 將 VSAM 應用程式檔案遷移至低延遲的 AWS 雲端，或將 VSAM 資料遷移至 AWS 資料倉儲或資料湖進行分析，以容忍高於應用程式現代化所需的同步延遲。

先決條件和限制

先決條件

- IBM z/OS V2R1 或更新版本
- 適用於 z/OS 的 CICS 交易伺服器 (CICS TS) V5.1 版或更新版本 (CICS/VSAM 資料擷取）
- IBM MQ 8.0 或更新版本
- 符合 z/OS 安全要求 （例如 SQData 載入程式庫的 APF 授權）
- VSAM 復原日誌已開啟
- （選用）CICS VSAM 復原版本 (CICS VR) 以自動擷取 CDC 日誌
- 作用中的 AWS 帳戶
- Amazon Virtual Private Cloud (VPC)，具有傳統平台可存取的子網路
- 來自 Precisely 的 VSAM Connect 授權

限制

- Connect 不支援根據來源 VSAM 結構描述或複製手冊自動建立目標資料表。您必須首次定義目標資料表結構。
- 對於非串流目標，例如 Amazon RDS，您必須在套用引擎組態指令碼中指定目標映射的轉換來源。
- 記錄、監控和提醒功能是透過 APIs且需要外部元件 （例如 Amazon CloudWatch) 才能完全運作。

產品版本

- z/OS 的 SQData 40134
- Amazon Elastic Compute Cloud (Amazon EC2) 上 Amazon Linux Amazon Machine Image (AMI) 的 SQData 4.0.43

Architecture

來源技術堆疊

- 任務控制語言 (JCL)
- z/OS Unix shell 和互動式系統生產力設施 (ISPF)
- VSAM 公用程式 (IDCAMS)

目標技術堆疊

- Amazon EC2
- Amazon MSK
- Amazon RDS
- Amazon VPC

目標架構

將 VSAM 檔案遷移至 Amazon RDS

下圖顯示如何在來源環境（內部部署大型主機）中使用 CDC 代理程式/發佈者，以及在目標環境 (AWS 雲端）中使用套用引擎，即時或近乎即時地將 VSAM 檔案遷移至關聯式資料庫，例如 Amazon RDS。

下圖顯示下列批次工作流程：

1. Connect 會比較備份檔案中的 VSAM 檔案以識別變更，然後將變更傳送至日誌串流，藉此擷取檔案的變更。
2. 發佈者會從系統日誌串流取用資料。
3. 發佈者會透過 TCP/IP 將擷取的資料變更傳達給目標引擎。Controller Daemon 會驗證來源和目標環境之間的通訊。

4. 目標環境中的套用引擎會從發佈者代理程式接收變更，並將其套用至關聯式或非關聯式資料庫。

圖表顯示下列線上工作流程：

1. Connect 會使用日誌複寫擷取線上檔案中的變更，然後將擷取的變更串流至日誌串流。
2. 發佈者會從系統日誌串流取用資料。
3. 發佈者會透過 TCP/IP 將擷取的資料變更傳達給目標引擎。控制器協助程式會驗證來源和目標環境之間的通訊。
4. 目標環境中的套用引擎會從發佈者代理程式接收變更，然後將其套用至關聯式或非關聯式資料庫。

將 VSAM 檔案遷移至 Amazon MSK

下圖顯示如何在高效能模式下將 VSAM 資料結構從大型主機串流至 Amazon MSK，並自動產生與 Amazon MSK 整合的 JSON 或 AVRO 結構描述轉換。

下圖顯示下列批次工作流程：

1. Connect 使用 CICS VR 或比較備份檔案的 VSAM 檔案來識別變更，以擷取檔案的變更。擷取的變更會傳送至日誌串流。
2. 發佈者會從系統日誌串流取用資料。
3. 發佈者會透過 TCP/IP 將擷取的資料變更傳達給目標引擎。控制器協助程式會驗證來源和目標環境之間的通訊。
4. 在平行處理模式下操作的複寫器引擎會將資料分割為工作快取單位。
5. 工作者執行緒會從快取擷取資料。
6. 資料會從工作者執行緒發佈至 Amazon MSK 主題。
7. 使用者使用連接器，將 Amazon MSK 的變更套用至 Amazon DynamoDB、Amazon Simple Storage Service (Amazon S3) 或 Amazon OpenSearch Service 等目標。

圖表顯示下列線上工作流程：

1. 使用日誌複寫擷取線上檔案中的變更。擷取的變更會串流至日誌串流。
2. 發佈者會從系統日誌串流取用資料。
3. 發佈者會透過 TCP/IP 將擷取的資料變更傳達給目標引擎。控制器協助程式會驗證來源和目標環境之間的通訊。

4. 在平行處理模式下操作的複寫器引擎會將資料分割為工作快取單位。
5. 工作者執行緒會從快取擷取資料。
6. 資料會從工作者執行緒發佈至 Amazon MSK 主題。
7. 使用者使用連接器將 Amazon MSK 的變更套用到 DynamoDB、Amazon S3 或 OpenSearch Service 等目標。

工具

- Amazon Managed Streaming for Apache Kafka (Amazon MSK) 是一項全受管服務，可協助您建置和執行使用 Apache Kafka 處理串流資料的應用程式。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。

史詩

準備來源環境 （大型主機）

任務	Description	所需的技能
安裝 Connect CDC 4.1。	1. 請聯絡 Precisely Support 團隊以取得授權和安裝套件。 2. 使用範例 JCLs Connect CDC 4.1。如需說明，請參閱 Precisely 文件中的使用 JCL 安裝 Connect CDC (SQData)。 3. 執行 `SETPROG APF`命令來授權 Connect 負載程式庫 SQDATA.V4nnn.LOADLIB。	IBM 大型主機開發人員/管理員
設定 zFS 目錄。	若要設定 zFS 目錄，請遵循精確文件中 zFS 變數目錄的指示。	IBM 大型主機開發人員/管理員

任務	Description	所需的技能
	Note 控制器協助程式和擷取/發佈代理程式組態存放在 z/OS UNIX Systems Services 檔案系統中（稱為 zFS）。Controller Daemon、Capture、Storage 和 Publisher 代理程式需要預先定義的 zFS 目錄結構，才能存放少量檔案。	
設定 TCP/IP 連接埠。	若要設定 TCP/IP 連接埠，請遵循精確文件中 TCP/IP 連接埠的指示。 **Note** Controller Daemon 需要來源系統的 TCP/IP 連接埠。引擎會在目標系統上參考連接埠（其中處理擷取的變更資料）。	IBM 大型主機開發人員/管理員

任務	Description	所需的技能
建立 z/OS 日誌串流。	若要建立 z/OS 日誌串流，請遵循精確文件中建立 z/OS 系統 logStreams 的指示。 **Note** Connect 使用 logstream 在遷移期間擷取和串流來源環境和目標環境之間的資料。 如需建立 z/OS LogStream 的範例 JCL，請參閱精確文件中的建立 z/OS 系統 logStreams。	IBM 大型主機開發人員
識別並授權 zFS 使用者和已啟動任務IDs。	使用 RACF 授予 OMVS zFS 檔案系統的存取權。如需範例 JCL，請參閱精確文件中的識別和授權 zFS 使用者和啟動的任務 IDs。	IBM 大型主機開發人員/管理員
產生 z/OS 公有/私有金鑰和授權金鑰檔案。	執行 JCL 以產生金鑰對。如需範例，請參閱此模式額外資訊區段中的金鑰對範例。 如需說明，請參閱精確文件中的產生 z/OS 公有和私有金鑰和授權金鑰檔案。	IBM 大型主機開發人員/管理員

任務	Description	所需的技能

啟用 CICS VSAM Log Replicate 並將其連接至日誌串流。	執行下列 JCL 指令碼：
	
	``
	 //STEP1 EXEC PGM=IDCAMS
	 //SYSPRINT DD SYSOUT=*
	 //SYSIN DD *
	   ALTER SQDATA.CICS.FILEA -
	   LOGSTREAMID(SQDATA.VSAMCDC.LOG1) -
	   LOGREPLICATE
	``
		IBM 大型主機開發人員/管理員

透過 FCT 啟用 VSAM 檔案復原日誌。	修改檔案控制表 (FCT)，以反映下列參數變更：
	
	``
	 Configure FCT Parms
	   CEDA ALT FILE(name) GROUP(groupname)
	   DSNAME(data set name)
	   RECOVERY(NONE|BACKOUTONLY|ALL)
	   FWDRECOVLOG(NO|1-99)
	   BACKUPTYPE(STATIC|DYNAMIC)
	   RECOVERY PARAMETERS
	   RECOVery : None | Backoutonly | All
	   Fwdrecovlog : No | 1-99
	   BAckuptype : Static | Dynamic
	``
		IBM 大型主機開發人員/管理員

任務	Description	所需的技能
設定發佈者代理程式的 CDCzLog。	1. 建立 CDCzLog Publisher CAB 檔案。 2. 加密已發佈的資料。 3. 準備 CDCzLog Publisher 執行期 JCL。	IBM 大型主機開發人員/管理員
啟用控制器協助程式。	1. 開啟 ISPF 面板並執行下列命令，以開啟精確選單： `EXEC 'SQDATA.V4nnnnn.ISPFLIB(SQDC$STA)' 'SQDATA.V4nnnnn'` 2. 若要設定控制器協助程式，請從功能表中選擇選項 2。	IBM 大型主機開發人員/管理員
啟用發佈者。	1. 開啟 ISPF 面板並執行下列命令，以開啟精確選單： `EXEC 'SQDATA.V4nnnnn.ISPFLIB(SQDC$STA)' 'SQDATA.V4nnnnn'` 2. 若要設定發佈者，請從選單中選擇選項 3，然後從 I 插入。	IBM 大型主機開發人員/管理員

任務	Description	所需的技能
啟用 logstream。	1. 開啟 ISPF 面板並執行下列命令，以開啟精確選單： `EXEC 'SQDATA.V4nnnnn.ISPFLIB(SQDC$STA)' 'SQDATA.V4nnnnn'` 2. 若要設定日誌串流，請從選單中選擇選項 4，然後選擇 I 進行插入。然後，輸入在上述步驟中建立的日誌串流名稱。	IBM 大型主機開發人員/管理員

準備目標環境 (AWS)

任務	Description	所需的技能
在 EC2 執行個體上精確安裝 。	若要在 Amazon EC2 的 Amazon Linux AMI 上從 Precisely 安裝 Connect，請遵循 Precisely 文件中 UNIX 上的 Install Connect CDC (SQData) 的指示。	一般 AWS
開啟 TCP/IP 連接埠。	若要修改安全群組以包含用於傳入和傳出存取的控制器協助程式連接埠，請遵循精確文件中 TCP/IP 的指示。	一般 AWS
建立檔案目錄。	若要建立檔案目錄，請遵循 Precisely 文件中準備目標套用環境的指示。	一般 AWS
建立 Apply Engine 組態檔案。	在 Apply Engine 的工作目錄中建立 Apply Engine 組態檔案。	一般 AWS

任務	Description	所需的技能

	下列範例組態檔案顯示 Apache Kafka 為目標：
	
	`builtin.features=SASL_SCRAM`
	`security.protocol=SASL_SSL`
	`sasl.mechanism=SCRAM-SHA-512`
	`sasl.username=`
	`sasl.password=`
	`metadata.broker.list=`
	
	Note
	如需詳細資訊，請參閱 Apache Kafka 文件中的安全性。	

建立套用引擎處理的指令碼。	建立套用引擎的指令碼，以處理來源資料並將來源資料複寫至目標。如需詳細資訊，請參閱精確文件中的建立套用引擎指令碼。	一般 AWS
執行指令碼。	使用 SQDPARSE和 SQDENG命令來執行指令碼。如需詳細資訊，請參閱精確說明文件中剖析 zOS 的指令碼。	一般 AWS

驗證環境

任務	Description	所需的技能
驗證用於 CDC 處理的 VSAM 檔案和目標資料表清單。	1. 驗證 VSAM 檔案，包括複寫日誌、復原日誌、FCT 參數和日誌串流。 2. 驗證目標資料庫資料表，包括資料表是否根據所需的結構描述定義、資料表存取和其他條件建立。	一般 AWS、大型主機
確認 Connect CDC SQData 產品已連結。	執行測試任務，並確認此任務的傳回碼為 0（成功）。 **Note** Connect CDC SQData Apply Engine 狀態訊息應會顯示作用中的連線訊息。	一般 AWS、大型主機

執行並驗證測試案例（批次）

任務	Description	所需的技能
在大型主機中執行批次任務。	使用修改過的 JCL 執行批次應用程式任務。在修改後的 JCL 中包含執行下列動作的步驟： 1. 備份資料檔案。 2. 比較備份檔案與修改後的資料檔案、產生差異檔案，然後記下訊息中的差異記錄計數。	一般 AWS、大型主機

任務	Description	所需的技能
	3. 將 delta 檔案推送至 z/OS 日誌串流。 4. 執行 JCL。如需範例 JCL，請參閱精確文件中的準備檔案比較擷取 JCL。	
檢查日誌串流。	檢查 logstream 以確認您可以查看已完成大型主機批次任務的變更資料。	一般 AWS、大型主機
驗證來源差異變更和目標資料表的計數。	若要確認記錄的高度，請執行下列動作： 1. 從批次 JCL 訊息中收集來源差異計數。 2. 監控 Apply Engine 是否有在 VSAM 檔案中插入、更新或刪除的記錄數量的記錄層級計數。 3. 查詢目標資料表中的記錄計數。 4. 比較並計算所有不同的記錄計數。	一般 AWS、大型主機

執行並驗證測試案例 （線上）

任務	Description	所需的技能
在 CICS 區域中執行線上交易。	1. 執行線上交易以驗證測試案例。 2. 驗證交易執行代碼 (RC=0 – 成功）。	IBM 大型主機開發人員

任務	Description	所需的技能
檢查日誌串流。	確認日誌串流已填入特定的記錄層級變更。	AWS Mainframe 開發人員
驗證目標資料庫中的計數。	監控 Apply Engine 的記錄層級計數。	精確，Linux
驗證目標資料庫中的記錄計數和資料記錄。	查詢目標資料庫以驗證記錄計數和資料記錄。	一般 AWS

相關資源

- VSAM z/OS （正確文件）
- 套用引擎 （準確文件）
- 複寫器引擎 （正確文件）
- 日誌串流 (IBM 文件）

其他資訊

組態檔案範例

這是日誌串流的範例組態檔案，其中來源環境是大型主機，而目標環境是 Amazon MSK：

```
 -- JOBNAME -- PASS THE SUBSCRIBER NAME
 -- REPORT  progress report will be produced after "n" (number) of Source records
processed.

 JOBNAME VSMTOKFK;
 --REPORT EVERY 100;
 -- Change Op has been ‘I’ for insert, ‘D’ for delete , and ‘R’ for Replace. For RDS
it is 'U' for update
 -- Character Encoding on z/OS is Code Page 1047, on Linux and UNIX it is Code Page
819 and on Windows, Code Page 1252
 OPTIONS
 CDCOP('I', 'U', 'D'),
 PSEUDO NULL = NO,
 USE AVRO COMPATIBLE NAMES,
```

```
  APPLICATION ENCODING SCHEME = 1208;

  --       SOURCE DESCRIPTIONS

  BEGIN GROUP VSAM_SRC;
  DESCRIPTION COBOL ../copybk/ACCOUNT AS account_file;
  END GROUP;

  --       TARGET DESCRIPTIONS

  BEGIN GROUP VSAM_TGT;
  DESCRIPTION COBOL ../copybk/ACCOUNT AS account_file;
  END GROUP;

  --       SOURCE DATASTORE (IP & Publisher name)

  DATASTORE cdc://10.81.148.4:2626/vsmcdct/VSMTOKFK
  OF VSAMCDC
  AS CDCIN
  DESCRIBED BY GROUP VSAM_SRC ACCEPT ALL;

  --       TARGET DATASTORE(s) - Kafka and topic name

  DATASTORE 'kafka:///MSKTutorialTopic/key'
  OF JSON
  AS CDCOUT
  DESCRIBED BY GROUP VSAM_TGT FOR INSERT;

  --       MAIN SECTION

  PROCESS INTO
  CDCOUT
  SELECT
  {
  SETURL(CDCOUT, 'kafka:///MSKTutorialTopic/key')
  REMAP(CDCIN, account_file, GET_RAW_RECORD(CDCIN, AFTER), GET_RAW_RECORD(CDCIN,
BEFORE))
  REPLICATE(CDCOUT, account_file)
  }
  FROM CDCIN;
```

金鑰對範例

此範例說明如何執行 JCL 來產生金鑰對：

```
//SQDUTIL EXEC PGM=SQDUTIL //SQDPUBL DD DSN=&USER..NACL.PUBLIC, //
 DCB=(RECFM=FB,LRECL=80,BLKSIZE=21200), // DISP=(,CATLG,DELETE),UNIT=SYSDA, //
 SPACE=(TRK,(1,1)) //SQDPKEY DD DSN=&USER..NACL.PRIVATE, //
 DCB=(RECFM=FB,LRECL=80,BLKSIZE=21200), // DISP=(,CATLG,DELETE),UNIT=SYSDA, //
 SPACE=(TRK,(1,1)) //SQDPARMS DD  keygen //SYSPRINT DD SYSOUT= //SYSOUT DD SYSOUT=* //
SQDLOG DD SYSOUT=* //*SQDLOG8 DD DUMMY
```

使用 現代化 CardDemo 大型主機應用程式 AWS Transform

Santosh Kumar Singh 和 Cheryl du Preez，Amazon Web Services

摘要

AWS Transform 旨在加速大型主機應用程式的現代化。它使用生成式 AI 來簡化大型主機現代化程序。它可自動化複雜的任務，例如：舊版程式碼分析、大型主機文件、商業規則擷取、單體應用程式分解為商業網域，以及程式碼重構。它透過自動化複雜的任務來加速現代化專案，例如應用程式分析和遷移序列規劃。分解單體應用程式時， AWS Transform 智慧排序大型主機應用程式轉換，這可協助您平行 轉換業務功能。 AWS Transform 可以加速決策並增強操作敏捷性和遷移效率。

此模式提供step-by-step指示，協助您 AWS Transform 使用 CardDemo 來測試 的大型主機現代化功能，這是範例開放原始碼大型主機應用程式。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- AWS IAM Identity Center，已啟用
- 允許管理員啟用 的許可 AWS Transform
- 允許管理員接受 AWS Transform Web 應用程式的 Amazon Simple Storage Service (Amazon S3) 連線請求的許可

限制

- AWS Transform 僅適用於部分 AWS 區域。如需支援區域的完整清單，請參閱支援的 區域 AWS Transform。
- AWS Transform 支援從常見商業導向語言 (COBOL) 到 Java 的程式碼分析、文件產生、業務規則擷取、分解和重構。如需詳細資訊，請參閱功能和主要功能，以及大型主機應用程式轉換支援的檔案類型。
- 中有大型主機轉換功能的服務配額 AWS Transform。如需詳細資訊，請參閱 的配額 AWS Transform。
- 若要在共用工作區上協作，所有使用者都必須是與 AWS Transform Web 應用程式執行個體 AWS IAM Identity Center 相關聯的相同 執行個體的註冊使用者。

- Amazon S3 儲存貯體和 AWS Transform 必須位於相同的 AWS 帳戶 和 區域。

Architecture

下圖顯示您在此模式中設定的架構。

該圖顯示以下工作流程：

1. AWS Transform 使用連接器來存取存放在 Amazon S3 儲存貯體中的 CardDemo 大型主機應用程式。
2. AWS Transform 使用 AWS IAM Identity Center 來管理使用者存取和身分驗證。系統會實作多層安全控制以進行身分驗證、授權、加密和存取管理，以協助在處理期間保護程式碼和成品。使用者透過聊天界面與 AWS Transform 客服人員互動。您可以為 AI 代理器提供英文特定任務的說明。如需詳細資訊，請參閱 AWS Transform 文件中的循環中的人工 (HITL)。
3. AI 代理程式會解譯使用者的指示、建立任務計畫、將任務分割為可執行任務，以及自動執行動作。使用者可以檢閱和核准轉換。轉換任務包括下列項目：
   - 程式碼分析 – AWS Transform 分析每個檔案中的程式碼，以取得檔案名稱、檔案類型、程式碼行及其路徑等詳細資訊。代理程式會分析原始程式碼、執行分類、建立相依性映射，以及識別任何遺失的成品。它也會識別重複的元件。
   - 文件產生 – AWS Transform 產生大型主機應用程式的文件。透過分析程式碼，它可以自動建立應用程式的詳細文件，包括存在於舊版系統中的商業邏輯、流程、整合和相依性的說明。
   - 業務邏輯擷取 - AWS Transform 分析 COBOL 程式以記錄其核心業務邏輯，協助您了解基本業務邏輯。
   - 程式碼分解 – 將程式碼 AWS Transform 分解為考量程式與元件之間相依性的網域。將相同網域中的相關檔案和程式分組可改善組織，並在將其分解為較小的元件時協助保留應用程式的邏輯結構。
   - 遷移波動規劃 – 根據您在分解階段建立的網域， 會以建議的現代化順序 AWS Transform 產生遷移波動計劃。
   - 程式碼重構 – 將所有或所選網域檔案中的程式碼 AWS Transform 重構為 Java 程式碼。此步驟的目標是保留應用程式的關鍵商業邏輯，同時將其重構為現代化、雲端最佳化的 Java 應用程式。
4. AWS Transform 會將重構程式碼、產生的文件、相關聯的成品和執行時間程式庫存放在 Amazon S3 儲存貯體中。您可以執行下列動作：
   - 存取 Amazon S3 儲存貯體中的執行時間資料夾。
   - 遵循 AWS Transform 文件中的建置和部署現代化應用程式，以建置和部署應用程式。

- 透過聊天界面，請求和下載範例 AWS CloudFormation AWS Cloud Development Kit (AWS CDK)，或 Hashicorp Terraform 範本。這些範本可協助您部署支援重構應用程式所需的 AWS 資源。
- 使用 Reforge 透過使用大型語言模型 (LLMs) 來改善重構程式碼的品質。重構引擎會保留 COBOL 的功能等效性，同時將其轉換為 Java 程式碼。Reforge 是轉換後可用的選用步驟。此步驟使用 LLMs 來重組程式碼，使其與原生 Java 非常相似，這可以改善可讀性和可維護性。Reforge 也新增了人類可讀的註解，以協助您了解程式碼，並實作現代編碼模式和最佳實務。

工具

AWS 服務

- AWS Transform 使用代理式 AI 來協助您加速傳統工作負載的現代化，例如 .NET、大型主機和 VMware 工作負載。
- AWS IAM Identity Center 可協助您集中管理 AWS 帳戶 和雲端應用程式的單一登入 (SSO) 存取。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

程式碼儲存庫

您可以使用開放原始碼 AWS CardDemo 大型主機應用程式做為範例應用程式，以開始使用大型主機現代化。

最佳實務

- 從小開始 – 從較不複雜的小程式碼 (15，000–20，000 行程式碼） 開始，以了解 如何 AWS Transform 分析和轉換大型主機應用程式。
- 結合人類專業知識 – AWS Transform 用作加速器，同時套用人類專業知識以獲得最佳結果。
- 徹底檢閱和測試 – 務必仔細檢閱轉換後的程式碼，並在轉換後執行全面的測試，以驗證功能等效性。
- 提供意見回饋 – 若要提供意見回饋和改進建議，請使用 中的傳送意見回饋按鈕， AWS 管理主控台 或使用 建立案例AWS 支援。如需詳細資訊，請參閱建立支援案例。您的輸入對於服務增強和未來的開發很有價值。

史詩

準備大型主機應用程式

任務	Description	所需的技能
建立儲存貯體。	在啟用 的相同 AWS 帳戶 和區域中建立 Amazon S3 AWS Transform 儲存貯體。您可以使用此儲存貯體來存放大型主機應用程式程式碼， AWS Transform 並使用此儲存貯體來存放產生的文件、重構程式碼，以及與轉換相關聯的其他檔案。如需說明，請參閱 Amazon S3 文件中的建立儲存貯體。	一般 AWS
準備範例大型主機應用程式。	1. 輸入下列命令，將 CardDemo 儲存庫複製到本機工作站： `git clone https://github.com/aws-samples/aws-mainframe-modernization-carddemo.git` 2. 建立名為 的新資料夾`carddemo`。 3. 將包含大型主機原始碼的`app`資料夾從複製的儲存庫複製到 `carddemo` 資料夾。 4. 將`cardemo`資料夾壓縮為 ZIP 檔案。 5. 將 ZIP 檔案上傳至您建立的 Amazon S3 儲存貯體。	應用程式開發人員、DevOps 工程師

任務	Description	所需的技能
	如需說明，請參閱 Amazon S3 文件中的上傳物件。Amazon S3	

設定 IAM Identity Center 和 AWS Transform

任務	Description	所需的技能
將使用者新增至 IAM Identity Center。	將您的潛在使用者新增至 IAM Identity Center。遵循 AWS Transform 文件中在 IAM Identity Center 中新增使用者的指示。	AWS 管理員
啟用 AWS Transform 並新增使用者。	1. 啟用 AWS Transform。請遵循啟用 AWS Transform中的指示。 2. 將使用者新增至 AWS Transform。遵循將使用者新增至 AWS Transform 中的指示。	AWS 管理員
設定使用者存取 AWS Transform Web 應用程式。	每個使用者都必須接受存取 AWS Transform Web 應用程式的邀請。遵循 AWS Transform 文件中接受邀請的指示。	應用程式開發人員、應用程式擁有者
登入 AWS Transform Web 應用程式。	遵循登入 AWS Transform中的指示。	應用程式開發人員、應用程式擁有者
設定工作區。	設定工作區，讓使用者可以在 AWS Transform Web 應用程式中協作。請遵循 AWS	AWS 管理員

任務	Description	所需的技能
	Transform 文件中啟動專案中的指示。	

轉換大型主機應用程式

任務	Description	所需的技能
建立轉換任務。	建立轉換任務以現代化 CardDemo 大型主機應用程式。如需說明，請參閱 AWS Transform 文件中的建立和啟動任務。當您被要求在 AWS Transform 聊天界面中設定目標時，請選擇執行大型主機現代化 (IBM z/OS 至 AWS)，然後選擇分析程式碼、產生技術文件、商業邏輯、分解程式碼、計劃遷移序列，以及將程式碼轉換為 Java。	應用程式開發人員、應用程式擁有者
設定連接器。	建立 Amazon S3 儲存貯體的連接器，其中包含 CardDemo 大型主機應用程式。此連接器允許 AWS Transform 存取儲存貯體中的資源，並執行連續的轉換函數。如需說明，請參閱 AWS Transform 文件中的設定連接器。	AWS 管理員
執行程式碼分析。	1. 在指定資產位置頁面上，輸入您上傳之 `carddemo` ZIP 檔案的 Amazon S3 儲存貯體路徑。	應用程式開發人員、應用程式擁有者

任務	Description	所需的技能
	2. 選擇核准並傳送至 AWS Transform 。它開始分析程式碼。 3. 在 Worklog 索引標籤上監控狀態。 4. 分析完成後，在左側導覽窗格中的分析程式碼下，選擇檢視程式碼分析結果。 5. （選用）選擇下載以下載完整的資產清單、缺少原始程式碼和相依性檔案。 如需詳細資訊，請參閱 AWS Transform 文件中的程式碼分析。	

任務	Description	所需的技能
產生技術文件。	1. 在左側導覽窗格的產生技術文件下，選擇選取檔案並設定設定。 2. 展開 COBOL 或 JCL，然後選取一或多個檔案。 3. 選擇文件詳細資訊層級： • 摘要 – 提供範圍內每個檔案的高階概觀。此外，提供每個檔案的單行摘要。 • 詳細功能規格 – 提供大型主機應用程式轉換範圍內每個檔案的完整詳細資訊。有些詳細資訊包括邏輯和流程、已識別的業務規則、資料流程、相依性、輸入和輸出處理，以及各種交易詳細資訊。 4. 選擇傳送到 AWS Transform。 5. 在 Worklog 索引標籤中監控進度。 **Note** 文件產生時間從分鐘到小時不等，取決於檔案數量和程式碼行數。 6. 完成後，請選擇檢閱文件結果。 7. 透過 Web 應用程式存取產生的文件或下載。產生的文	應用程式開發人員、應用程式擁有者

任務	Description	所需的技能
	件也會存放在 Amazon S3 儲存貯體中。 8. 在聊天界面中詢問特定問題，以探索產生的文件。例如，您可以說「告訴我 CBACT01C 的功能」。 如需詳細資訊，請參閱 文件中的產生技術 AWS Transform 文件。	

任務	Description	所需的技能
擷取商業邏輯。	1. 在左側導覽窗格中，展開擷取商業邏輯，然後選擇選取檔案以進行商業邏輯擷取。 2. 展開 COBOL 或 JCL，然後選取一或多個檔案。 3. 選擇傳送到 AWS Transform。 4. 在 Worklog 索引標籤中監控進度。 **Note** 商業邏輯擷取時間從分鐘到小時不等，取決於檔案數量和程式碼行數。 5. 擷取完成時，選擇檢閱業務邏輯擷取結果。 6. 選取 COBOL 或 JCL 檔案，然後選擇檢視。 7. 變更流程圖和功能規則中的範圍，以檢閱流程圖的不同層級，包括支援程式碼。 8. （選用）檢閱產生的商業邏輯，該邏輯會以 JSON 格式存放在 Amazon S3 儲存貯體中。 如需詳細資訊，請參閱 AWS Transform 文件中的擷取商業邏輯。	應用程式開發人員、應用程式擁有者

任務	Description	所需的技能
分解程式碼。	1. 在左側導覽窗格中，展開分解程式碼，然後選擇分解為網域。 2. 在動作清單中，選擇建立網域。 3. 輸入新網域的名稱，例如 `Bill payment`。 4. （選用）提供描述。 5. 在尋找檔案搜尋列中，搜尋 `CB00`，然後選取檔案。 6. 選擇標記為種子。 7. 確認種子旗標從否變更為是。 8. 選擇建立。 9. 選擇儲存。 10選擇分解。 11選擇網域名稱來檢閱分解輸出。 12.當分解完成時，選擇傳送至 AWS Transform。 如需分解和種子的詳細資訊，請參閱 AWS Transform 文件中的分解。	應用程式開發人員、應用程式擁有者
規劃遷移波紋。	規劃 CardDemo 應用程式的遷移波紋。遵循 AWS Transform 文件中的遷移波動規劃中的指示，以檢閱和編輯波動計畫。	應用程式開發人員、應用程式擁有者

任務	Description	所需的技能
重構程式碼。	1. 將 CardDemo 大型主機應用程式重構為所有或所選網域檔案中的 Java 程式碼。遵循 AWS Transform 文件中的重構程式碼中的指示。 2. 重構程序完成後，建置和部署現代化 Java 應用程式。遵循 AWS Transform 文件中的建置和部署現代化應用程式重構後的指示。	應用程式開發人員、應用程式擁有者
（選用）使用 Reforge 改善 Java 程式碼。	1. 建立新的任務，然後輸入 的目標`Reforge the code`。 2. 輸入已透過 進行現代化之專案的 AWS Transform。此專案應該在您的 Amazon S3 儲存貯體中。 3. 輸入 Java 類別清單，指定要複寫的服務類別。 4. 檢閱 Amazon S3 儲存貯體中產生的輸出。 如需詳細資訊，請參閱 AWS Transform 文件中的重新建構。	應用程式開發人員、應用程式擁有者

任務	Description	所需的技能
簡化部署。	AWS Transform 可以為 CloudFormation AWS CDK或 Terraform 提供基礎設施做為程式碼 (IaC) 範本。這些範本可協助您部署核心元件，包括運算、資料庫、儲存和安全資源。 1. 在 AWS Transform 聊天介面中，輸入 來請求範本`Share the mainframe refactor <service> templates` ，其中 `<service>` 是其中一個支援的 IaC 服務。 2. 使用您偏好的服務部署範本。例如，請參閱下列資源： • 從 CloudFormation 主控台建立堆疊 (CloudFormation 文件） • 部署 AWS CDK 應用程式 (AWS CDK 文件） • 建置基礎設施 (Terraform 文件） 如需詳細資訊，請參閱 AWS Transform 文件中的部署功能。	應用程式開發人員、應用程式擁有者

疑難排解

問題	解決方案
您無法在 AWS Transform Web 應用程式中檢視原始程式碼或產生的文件。	將政策新增至 Amazon S3 儲存貯體的 CORS 許可，以允許 AWS Transform 做為原始伺服器。如需詳細資訊，請參閱 AWS Transform 文件中的 S3 儲存貯體 CORS 許可。

相關資源

AWS 文件

- 大型主機應用程式的轉換 (AWS Transform 文件）

其他 AWS 資源

- 使用 AI 代理器搭配 加速您的大型主機現代化旅程 AWS Transform(AWS 部落格文章）
- AWS Transform FAQs
- AWS IAM Identity Center FAQs

影片和教學課程

- Amazon Q 開發人員簡介：轉換AWS （技能建置器）
- AWS re：Invent 2024 - 使用 Amazon Q Developer (YouTube) 更快速地現代化大型主機應用程式 YouTube
- AWS re：Invent 2024 - 自動化遷移和現代化以加速轉型 (YouTube)
- AWS re：Invent 2024 - Toyota 透過新一代 AI (YouTube) 推動創新並增強營運效率 YouTube

> **Note**
>
> AWS Transform 先前稱為大型主機的 Amazon Q Developer 轉換。

使用 和 Terraform 現代化 AWS Transform 和部署大型主機應用程式

Mason Cahill、Qandi Jhandi、Prachi Khanna、Sivasubramanian Ramani 和 Santosh Kumar Singh，Amazon Web Services

摘要

AWS Transform 可以加速 .NET、大型主機和 VMware 工作負載的大規模現代化。它部署專門的 AI 代理器，可自動化複雜的任務，例如評估、程式碼分析、重構、分解、相依性映射、驗證和轉型規劃。此模式示範如何使用 AWS Transform 來現代化大型主機應用程式，然後使用 Hashicorp Terraform 將其部署至 AWS 基礎設施。這些step-by-step說明可協助您將 CardDemo 轉換為從 COBOL 到現代 Java 應用程式的範例開放原始碼大型主機應用程式。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 建立 AWS 資源和部署應用程式的管理許可
- Terraform 1.5.7 版或更新版本，已設定
- AWS Terraform 提供者，已設定
- AWS IAM Identity Center，已啟用
- AWS Transform，已啟用
- 加入 AWS Transform 工作區的使用者，具有可執行轉換任務的貢獻者角色

限制

- AWS Transform 僅適用於部分 AWS 區域。如需支援區域的完整清單，請參閱 支援的 區域 AWS Transform。
- 中有大型主機轉換功能的服務配額 AWS Transform。如需詳細資訊，請參閱 的配額 AWS Transform。
- 若要在共用工作區上協作，所有使用者都必須是與 AWS Transform Web 應用程式執行個體 AWS IAM Identity Center 相關聯的相同 執行個體的註冊使用者。
- Amazon Simple Storage Service (Amazon S3) 儲存貯體和 AWS Transform 必須位於相同的 AWS 帳戶 和 區域。

Architecture

下圖顯示舊版應用程式和部署至 的end-to-end現代化 AWS 雲端。應用程式和資料庫登入資料存放在中 AWS Secrets Manager，Amazon CloudWatch 提供監控和記錄功能。

該圖顯示以下工作流程：

1. 透過 AWS IAM Identity Center，使用者會在 AWS Transform 中驗證和存取 AWS 帳戶。
2. 使用者將 COBOL 大型主機程式碼上傳至 Amazon S3 儲存貯體，並在其中啟動轉換 AWS Transform。
3. AWS Transform 將 COBOL 程式碼現代化為雲端原生 Java 程式碼，並將現代化程式碼存放在 Amazon S3 儲存貯體中。
4. Terraform 會建立 AWS 基礎設施來部署現代化應用程式，包括 Application Load Balancer、Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和 Amazon Relational Database Service (Amazon RDS) 資料庫。Terraform 會將現代化程式碼部署到 Amazon EC2 執行個體。
5. VSAM 檔案會上傳至 Amazon EC2，並從 Amazon EC2 遷移至 Amazon RDS 資料庫。

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。在此模式中，SQL Server 容錯移轉叢集執行個體會安裝在 Amazon EC2 執行個體上。
- AWS IAM Identity Center 可協助您集中管理 AWS 帳戶 和雲端應用程式的單一登入 (SSO) 存取。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 中設定、操作和擴展關聯式資料庫 AWS 雲端。
- AWS Secrets Manager 可協助您將程式碼中的硬式編碼憑證 (包括密碼) 取代為 Secrets Manager 的 API 呼叫，以便透過程式設計方法來擷取機密。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Transform 使用代理式 AI 來協助您加速傳統工作負載的現代化，例如 .NET、大型主機和 VMware 工作負載。

其他工具

- Apache Maven 是一種開放原始碼軟體專案管理，並為 Java 專案建置自動化工具。
- Apache Tomcat 是適用於 Java 程式碼的開放原始碼 Servlet 容器和 Web 伺服器。
- HashiCorp Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。
- Spring Boot 是一種開放原始碼架構，建置在 Java 中的 Spring Framework 之上。

程式碼儲存庫

此模式的程式碼可在 GitHub 大型主機轉換 E2E 儲存庫中使用。此模式使用開放原始碼 AWS CardDemo 大型主機應用程式做為範例應用程式。

最佳實務

- 指派遷移目標程式碼和資源的完整擁有權。
- 在擴展到完整遷移之前，開發並測試概念驗證。
- 保護所有利益相關者的承諾。
- 建立清晰的通訊管道。
- 定義並記錄可行產品 (MVP) 的最低需求。
- 設定明確的成功條件。

史詩

準備和上傳大型主機應用程式程式碼

任務	Description	所需的技能
建立儲存貯體。	在啟用 的相同 AWS 帳戶 和區域中建立 Amazon S3 AWS Transform 儲存貯體。您可以使用此儲存貯體來存放建置和執行應用程式所需的大型主機應用程式程式碼、資料和其他指令碼。 AWS Transform 使用此儲存貯體來存放重構程	一般 AWS、AWS 管理員

任務	Description	所需的技能
	式碼，以及與轉換相關聯的其他檔案。如需說明，請參閱 Amazon S3 文件中的 建立儲存貯 體。	
設定儲存貯體的 CORS 許可。	設定儲存貯體進行 AWS Transform 存取時，您需要設定儲存貯體的跨來源資源共用 (CORS)。如果未正確設定，您可能無法使用 的內嵌檢視或檔案比較功能 AWS Transform。如需如何為儲存貯體設定 CORS 的說明，請參閱在 Amazon S3 儲存貯體中使用跨來源資源共用。Amazon S3 如需政策，請參閱 AWS Transform 文件中的 S3 儲存貯體 CORS 許可。	一般 AWS、AWS 管理員

任務	Description	所需的技能
準備範例大型主機應用程式程式碼。	1. 輸入下列命令，將 CardDemo 儲存庫複製到您的本機工作站： `git clone https://github.com/aws-samples/aws-mainframe-modernization-carddemo.git` 2. 將`aws-mainframe-modernization-carddemo` 資料夾壓縮為 ZIP 檔案。 3. 將 ZIP 檔案上傳至您建立的 Amazon S3 儲存貯體。如需說明，請參閱 Amazon S3 文件中的上傳物件。Amazon S3	General AWS，應用程式開發人員

轉換大型主機應用程式

任務	Description	所需的技能
設定 AWS Transform 任務。	1. 使用您的登入資料登入，以存取 AWS Transform Web 應用程式。 2. 依照 AWS Transform 文件中設定工作區的指示建立新的工作區。 3. 在工作區登陸頁面上，選擇請求 AWS Transform 以建立任務。	應用程式開發人員、應用程式擁有者

任務	Description	所需的技能
	4. 接著，選擇 Mainframe Modernization 做為任務類型。 5. 在聊天視窗中，輸入轉換程式碼到 Java。 6. 檢閱建議的任務類型、名稱和目標。若要確認，請輸入是。 7. 選擇建立任務。	
設定連接器。	1. 使用您建立的 Amazon S3 儲存貯體設定連接器。如需說明，請參閱 AWS Transform 文件中的設定連接器。出現提示時，請在 Amazon S3 儲存貯體中輸入 `aws-mainframe-modernization-carddemo` zip 檔案的路徑。 2. 等候分析完成。	應用程式開發人員、應用程式擁有者
轉換程式碼。	1. 根據 AWS Transform 文件中分析程式碼中的指示，檢閱程式碼分析的結果。 2. 根據 AWS Transform 文件中重構程式碼中的指示重構大型主機程式碼。對於範例 CardDemo 大型主機應用程式，您可以接受預設設定。 3. 等待重構完成。 4. 選擇檢視結果，以查看 Amazon S3 儲存貯體中重構程式碼的路徑。請記下此檔案路徑。以供稍後使用。	應用程式開發人員、應用程式擁有者

透過 Terraform 部署基礎設施

任務	Description	所需的技能

更新 範本。	1. 輸入下列命令，將大型主機轉換 E2E 儲存庫複製到本機工作站：
	`git clone https://github.com/aws-samples/sample-mainframe-transformation-e2e.git`
	2. 輸入下列命令以擷取您目前的公有 IP 地址：
	`curl checkip.amazonaws.com`
	3. 輸入下列命令以導覽至基礎設施目錄：
	`cd mainframe-transformation-e2e/infra`
	4. 開啟 `variables.tf` 檔案。
	5. 將 取代`YOUR_IP_ADDRESS_HERE` 為您的 IP 地址。
	6. 如果您有公有託管區域，請執行下列動作：
	a. `hosted_zone_name` 將 取代為您的託管區域名稱。
		一般 AWS、AWS 管理員

任務	Description	所需的技能

	b. 將 `hosted_zone_enabled` 設定為 `true`。
	
	如果您沒有公有託管區域，請執行下列動作：
	a. 輸入下列命令來產生自我簽署憑證：
	
	`openssl genrsa 2048 > my-private-key.pem`
	`openssl req -new -x509 -nodes -sha256 -days 365 -key my-private-key.pem -outform PEM -out my-certificate.pem`
	
	b. 輸入下列命令將憑證匯入 AWS Certificate Manager (ACM)：
	
	`aws acm import-certificate \`
	`--certificate fileb://my-certificate.pem \`
	`--private-key fileb://my-private-key.pem`
	
	此命令的輸出包含匯入憑證的 Amazon Resource Name (ARN)。	

任務	Description	所需的技能
	c. `self_signed_cert_arn` 以憑證的 ARN 取代。 d. 將 `hosted_zone_enabled` 設定為 `false`。 7. `aws_region` 變更為目標區域。預設值為 `us-east-1`。 8. 儲存並關閉 `variables.tf` 檔案。 **Note** 對於生產環境或類似生產環境，請設定其他安全元件。例如，啟用 AWS WAF Application Load Balancer 的保護。	

任務	Description	所需的技能

部署基礎設施。	1. 輸入下列命令以初始化 Terraform：
	`terraform init`
	2. 輸入下列命令來產生執行計畫：
	`terraform plan`
	3. 檢閱計劃，並驗證要建立的資源和基礎設施元件。
	4. 輸入下列命令來部署基礎設施：
	`terraform apply`
	5. 出現提示時，請輸入 yes 以確認部署。
	6. 等待部署完成。
		Terraform

在 Amazon EC2 執行個體上安裝和設定 Apache Tomcat

任務	Description	所需的技能
安裝必要的軟體。	1. 使用 連線至 Amazon EC2 執行個體AWS Systems Manager Session Manager。 2. 輸入下列命令以切換到根使用者： `sudo su -`	應用程式開發人員、遷移工程師

任務	Description	所需的技能

	3. 輸入下列命令以瀏覽至 `scripts` 目錄：
	`cd /opt/scripts`
	4. 檢閱 install_software.sh 指令碼。此指令碼會安裝 Java 17、Apache Maven 和 Apache Tomcat 10.0.23。
	5. 視需要為您的使用案例更新指令碼。
	6. 輸入下列命令讓指令碼可執行：
	`chmod +x install_software.sh`
	7. 輸入下列命令來執行指令碼：
	`./install_software.sh`	

任務	Description	所需的技能

驗證軟體安裝。	1. 輸入下列命令以啟動 Tomcat 伺服器：
	`/opt/tomcat/apache-tomcat-10.0.23/bin/startup.sh`
	2. 輸入下列命令來驗證 Web 伺服器回應：
	`curl http://localhost:8080`
	輸出應確認 Tomcat 正在服務 HTML 頁面。
		應用程式開發人員、遷移工程師

編譯和封裝現代化應用程式程式碼

任務	Description	所需的技能
下載並擷取產生的程式碼。	1. 輸入下列命令，讓 download_and_extract.sh 指令碼可執行。此指令碼會下載存放在 Amazon S3 儲存貯體中的重構程式碼和 Gapwalk 執行期程式庫： `chmod +x /opt/scripts/download_and_extract.sh` 2. 輸入以下命令以執行 指令碼。`<file_path>` 將 取代為 Amazon S3 儲存貯體中 generated.zip 檔案的路徑：	應用程式開發人員、遷移工程師

任務	Description	所需的技能

	`./download_and_extract.sh <file_path>`
	
	Note
	檔案路徑通常為 `s3://<bucket-name>/transform_output/<aws_transform_job_id>/code transformation/generated.zip` 。
	
	3. 輸入下列命令以導覽至 shared 資料夾：
	
	`cd /opt/runtime/velocity/shared`
	
	4. 輸入下列命令以複製 deploy-velocity-runtime.sh 指令碼：
	
	`cp /opt/scripts/deploy-velocity-runtime.sh .`
	
	5. 輸入下列命令，讓複製的指令碼可執行：
	
	`chmod +x deploy-velocity-runtime.sh`	

任務	Description	所需的技能
	6. 輸入以下命令以執行 指令碼。此指令碼會將專案物件模型 (POM) 檔案中存在的所有必要 Web 應用程式封存 (WAR) 相依性複製到儲存庫資料夾中： `./deploy-velocity-runtime.sh` 7. 檢查本機 Maven 儲存庫中是否安裝了所需的 WAR 相依性，以確認成功執行。	

任務	Description	所需的技能

建置現代化應用程式。	1. 輸入下列命令以導覽至app-pom專案目錄：
	`cd /opt/codebase/app-pom/`
	2. 輸入下列命令來安裝 Maven：
	`mvn clean install`
	3. 等待安裝和建置完成。
	Note
	為 CardDemo 應用程式執行此命令時，您可能會遇到app-web專案的警告訊息。您可以安全地忽略這些警告。
	4. 成功完成建置後，確認 `app-service/target/app-service-1.0.0.war` 和 是否存在`app-web/target/app-web-1.0.0.war` 。
	Important
	請勿在此階段重新啟動 Tomcat 伺服器。由於缺少必要的資料庫，這將導致錯誤。您必須先設定資料
		應用程式開發人員、遷移工程師

任務	Description	所需的技能
	庫，才能重新啟動伺服器。	

遷移資料庫

任務	Description	所需的技能

建立資料庫和 JICS 結構描述。	1. 輸入下列命令，將csd commands資料夾重新命名為 csd_commands 。這會從資料夾名稱中移除空格：
	`mv /opt/codebase/extra/csd\ commands/ /opt/codebase/extra/csd_commands`
	2. 輸入下列命令以導覽至scripts目錄：
	`cd /opt/scripts`
	3. 輸入下列命令，讓資料庫遷移指令碼可執行：
	`chmod +x database_migration_setup.sh`
	4. 輸入下列命令，將下列變數設定為參數：
	`RDS_ENDPOINT=<database_endpoint> SECRET_NAME=<secret_name> JICS_SQL_SCRIPT_DIR=/opt/runtim`
		應用程式開發人員、遷移工程師

任務	Description	所需的技能
	e/velocity/jics/sql/jics.sql INIT_JICS_SQL_SCRIPT_DIR=/opt/codebase/extra/csd_commands/sql/aws-mainframe-modernization-carddemo-main/app/csd/initJics.sql 其中： • <database_endpoint> 是您透過 Terraform 部署之 Amazon RDS 資料庫的端點。 • <secret_name> 是您透過 Terraform 部署的 AWS Secrets Manager 秘密名稱。 5. 輸入下列命令來執行資料庫遷移指令碼： ./database_migration_setup.sh $RDS_ENDPOINT $SECRET_NAME $JICS_SQL_SCRIPT_DIR $INIT_JICS_SQL_SCRIPT_DIR 6. 輸入下列命令，從 Amazon EC2 執行個體連線至資料庫： psql -h <Your Amazon RDS Endpoint> -U foo -p 5432 postgres	

任務	Description	所需的技能
	7. 出現提示時，輸入您的資料庫登入資料。	

驗證資料庫建立。	1. 輸入下列命令以檢視所有資料庫：
	`\l`
	2. 輸入下列命令以切換到`jics`資料庫：
	`\c jics`
	3. 輸入下列命令來檢閱建立的資料表清單：
	`\dt`
		應用程式開發人員、遷移工程師

任務	Description	所需的技能

將資料遷移至 JICS 資料庫。	1. 輸入下列命令，讓 execute_listcat_sql.sh 指令碼可執行：
	`chmod +x execute_listcat_sql.sh`
	2. 輸入下列命令來設定 PATH_TO_LISTCAT_SQL_FILES 變數，這是包含 LISTCAT SQL 檔案的目錄：
	`PATH_TO_LISTCAT_SQL_FILES=/opt/codebase/extra/listcat/sql/cluster/aws-mainframe-modernization-carddemo-main/app/catlg`
	3. 請確定 RDS_ENDPOINT 、 SECRET_NAME 和 PATH_TO_LISTCAT_SQL_FILES 變數已根據先前的說明正確設定。
	4. 輸入下列命令來執行 execute_listcat_sql.sh 指令碼：
	`./execute_listcat_sql.sh $RDS_ENDPOINT $SECRET_NAME $PATH_TO_LISTCAT_SQL_FILES`
		應用程式開發人員、遷移工程師

任務	Description	所需的技能
	此指令碼會更新 JICS 資料庫中的 VSAM 檔案屬性，並執行必要的查詢來修改資料庫。	

安裝現代化應用程式

任務	Description	所需的技能

在 Amazon EC2 執行個體上安裝現代化應用程式。	1. 輸入下列命令讓 application_installer.sh 指令碼可執行：
	`chmod +x /opt/scripts/application_installer.sh`
	2. 輸入下列命令，將下列變數設定為參數：
	`RDS_ENDPOINT=<database_endpoint> SECRET_NAME=<secret_name> AIX_JSON_FILE_PATH=/opt/codebase/extra/csd_commands/json/aws-mainframe-modernization-carddemo-main/jicsFileAix.json LISTCAT_JSON_FILES_DIR=/opt/codebase/extra/listcat/json/cluster/default/aws-mainframe-`
		應用程式開發人員、雲端架構師

任務	Description	所需的技能
	modernization-card demo-main/app/catlg S3_PATH_FOR_EBCD IC_DATA_FILES=s3:/ /<bucket_name>/tra nsform-output/<job _id>/inputs/aws-ma inframe-modernizat ion-carddemo-main/ app/data/EBCDIC 其中： • <database_endpoint> 是您透過 Terraform 部署之 Amazon RDS 資料庫的端點。 • <secret_name> 是您透過 Terraform 部署的 AWS Secrets Manager 秘密名稱。 • <bucket_name> 是包含現代化應用程式的 Amazon S3 儲存貯體名稱。 • <job_id> 是 AWS Transform 任務的 ID。 3. 輸入下列命令以執行 application_installer.sh 指令碼： ./application_inst aller.sh $RDS_ENDP OINT $SECRET_NAME $AIX_JSON_FILE_PAT H $LISTCAT_JSON_FILE S_DIR $S3_PATH_	

任務	Description	所需的技能
	FOR_EBCDIC_DATA_FILES 4. 在 /opt/tomcat/apache-tomcat-10.0.23/workingdir/config 資料夾的 application-utility-pgm.yml 檔案中，將編碼參數變更為下列項目： encoding : CP1047 **Note** 當您使用 AWS Blu Age AWS Transform 自動重構中的應用程式時，您可以透過 YAML 檔案設定應用程式及其執行時間環境。例如，您可以在 application-main.yml 檔案中為應用程式設定記錄。如需可用屬性的詳細資訊，請參閱啟用 AWS Blu Age Runtime 的屬性。	

任務	Description	所需的技能

重新啟動 Tomcat 伺服器。	1. 輸入下列命令以導覽至工作目錄：
	`cd /opt/tomcat/apache-tomcat-10.0.23/workingdir`
	2. 輸入下列命令來停止和啟動 Tomcat 伺服器：
	`/opt/tomcat/apache-tomcat-10.0.23/bin/shutdown.sh`
	`/opt/tomcat/apache-tomcat-10.0.23/bin/startup.sh`
	3. 輸入下列命令來監控 Tomcat 服務啟動日誌：
	`tail -f /opt/tomcat/apache-tomcat-10.0.23/logs/catalina.out`
		應用程式開發人員、雲端架構師

任務	Description	所需的技能
遷移 VSAM 資料集。	1. 開啟 Amazon EC2 主控台。 2. 在導覽窗格上，選擇 Load balancers (負載平衡器)。 3. 選擇透過 Terraform 建立的負載平衡器。 4. 找到 Application Load Balancer 的網域名稱系統 (DNS) 名稱，例如 `application-load-balancer-<id>.<region>.elb.amazonaws.com` 。 5. 在您的瀏覽器中，導覽至 `http://<dns_name>/gapwalk-application/scripts/data-load` ，其中 `<dns_name>` 是 Application Load Balancer 的 DSN 名稱。這會啟動資料載入指令碼。 6. 等待指令碼完成。完成後，瀏覽器會顯示 DONE。 7. 在 Amazon EC2 執行個體上，開啟新的終端機。 8. 輸入下列命令以連線至 Amazon RDS 資料庫，將 取代`<database_endpoint>` 為您的值： `psql -h <database_endpoint> -U foo -p 5432 postgres`	應用程式開發人員、遷移工程師

任務	Description	所需的技能

	9. 出現提示時，輸入您的登入資料以連線至資料庫。
	10輸入下列命令以檢視所有資料庫：
	`\l`
	11輸入下列命令以切換到bluesam資料庫：
	`\c bluesam`
	12輸入下列命令來檢閱建立的資料表清單：
	`\dt`
	13輸入下列命令來驗證資料載入：
	`SELECT * FROM public.aws_m2_carddemo_usrsec_vsam_ksds;`
	輸出應會顯示傳回的 10 筆記錄。	

任務	Description	所需的技能
更新 Groovy 指令碼中的參數。	1. 輸入下列命令以導覽至指令碼目錄： `cd /opt/tomcat/apache-tomcat-10.0.23/webapps/workingdir/scripts` 2. 在包含一般檔案參考的所有 Groovy 檔案中，更新下列檔案組態： • 本機檔案路徑 – Amazon EC2 執行個體本機目錄中的平面檔案路徑 • 檔案系統類型 – 包含一般檔案的檔案系統 • 記錄大小 – 平面檔案的大小 如需參考，請參閱程式碼儲存庫中的範例 DUSRSECJ.jcl.groovy 指令碼。 3. 儲存並關閉檔案。	應用程式開發人員

測試應用程式。

任務	Description	所需的技能
測試現代化應用程式。	1. 透過 Application Load Balancer (`http://<your-load-balancer-dns>`) 或透過託管區域 () 存取線上應用程	應用程式開發人員、測試工程師

任務	Description	所需的技能
	式https://myhostedzone.dev/ 。 2. 針對交易 ID，輸入 CC00。 3. 針對使用者名稱，輸入 USER0001。 4. 針對密碼，輸入 PASSWORD。 成功登入後，會顯示主選單。	
驗證批次指令碼。	1. 透過 Application Load Balancer (http://<your-load-balancer-dns>/gapwalk-application/scripts) 或透過託管區域 () 存取指令碼界面https://myhostedzone.dev/gapwalk-application/scripts 。 2. 選擇要執行的指令碼，例如 DUSRSECJ.jcl.groovy 指令碼。 3. 確認指令碼已成功執行。以下是成功執行後的範例輸出。 `{ "exitCode": 0, "stepName": "STEP03", "program": "IDCAMS", "status": "Succeeded" }`	應用程式開發人員、測試工程師

清除

任務	Description	所需的技能

準備刪除基礎設施。	1. 輸入下列命令，從 Amazon RDS 執行個體移除刪除保護：
	`aws rds modify-db-instance \ --db-instance-identifier <your-db-instance-name> \ --no-deletion-protection \ --apply-immediately`
	2. 輸入下列命令以移除 Application Load Balancer 的刪除保護：
	`aws elbv2 modify-load-balancer-attributes \ --load-balancer-arn <your-load-balancer-arn> \ --attributes Key=deletion_protection.enabled,Value=false`
	3. 輸入下列命令來刪除 Amazon S3 儲存貯體的內容：
	`ACCOUNT_NUMBER=$(aws sts get-caller-identity --query`
		一般 AWS

任務	Description	所需的技能
	Account --output text) aws s3 rm s3://mf-c arddemo-$ACCOUNT_N UMBER --recursive aws s3 rm s3://mf-c arddemo-logs-$ACCO UNT_NUMBER --recursi ve	
刪除基礎設施。	**Important** 這些步驟將永久刪除您的 資源。在繼續之前，請確定您已備份任何重要資料。 1. 輸入下列命令以導覽至 infra 資料夾： cd /mainframe-transfo rmation-e2e/infra 2. 輸入下列命令來刪除基礎設施： terraform destroy -- auto-approve	一般 AWS

疑難排解

問題	解決方案
Terraform 身分驗證	1. 請確定已正確設定 AWS 登入資料。 2. 確認您已選取正確的 AWS 設定檔。

問題	解決方案
	3. 確認您擁有必要的許可。
Tomcat 相關錯誤	1. 檢查 中的 catalina.out `/opt/tomcat/apache-tomcat-10.0.23/logs` 是否有任何例外狀況。 2. 輸入下列命令，將 Tomcat 資料夾的擁有權變更為 Tomcat 使用者： `chown -R tomcat:tomcat /opt/tomcat/*`
URL 名稱未載入	請確定 Application Load Balancer 安全群組在傳入規則中具有您的 IP 地址做為來源。
Tomcat 日誌中的身分驗證問題	確認 中的資料庫秘密密碼 AWS Secrets Manager 和 server.xml 中的密碼相符。

相關資源

AWS 方案指引

- 使用 AWS Transform 現代化 CardDemo 大型主機應用程式

AWS 服務文件

- AWS Blu Age Blusam 管理主控台
- AWS Blu Age Runtime 的基礎設施設定需求 （ 非受管 ）
- 加入 AWS Blu Age 執行期
- 大型主機應用程式的現代化
- 設定 AWS Blu Age Runtime 的組態

AWS 部落格文章

- 使用 AI 代理器搭配 加速您的大型主機現代化旅程 AWS Transform

AWS 使用 Rocket Enterprise Server 和 LRS PageCenterX 在 上現代化大型主機輸出管理

Shubham Roy，Amazon Web Services

Abraham Rondon，微型焦點

Guy Tucker、Levi、Ray 和 Shoup Inc

總結

透過現代化大型主機輸出管理，您可以節省成本、減輕維護舊版系統的技術負擔，並透過 DevOps 和 Amazon Web Services (AWS) 雲端原生技術提高彈性和敏捷性。此模式說明如何在 AWS 雲端上現代化業務關鍵型大型主機輸出管理工作負載。此模式使用 Rocket Enterprise Server 做為現代化大型主機應用程式的執行時間，並搭配 Levi、Ray & Shoup， Inc.。 (LRS) VPSX/MFI (Micro Focus Interface) 作為列印伺服器，而 LRS PageCenterX 作為封存伺服器。LRS PageCenterX 提供輸出管理解決方案，用於檢視、編製索引、搜尋、封存和保護對業務輸出的存取。

模式是以轉換大型主機現代化方法為基礎。大型主機應用程式會由 Amazon Elastic Compute Cloud (Amazon EC2) 上的 AWS Mainframe Modernization 進行遷移。大型主機輸出管理工作負載會遷移至 Amazon EC2，而大型主機資料庫，例如 z/OS 的 IBM Db2，則會遷移至 Amazon Relational Database Service (Amazon RDS)。LRS Directory Integration Server (LRS/DIS) 可與 AWS Directory Service for Microsoft Active Directory 搭配使用，以進行輸出管理工作流程身分驗證和授權。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 大型主機輸出管理工作負載。
- 如何重建和交付在 Rocket Enterprise Server 上執行之大型主機應用程式的基本知識。如需詳細資訊，請參閱 Rocket 軟體文件中的 Rocket Enterprise Server 資料表。
- LRS 雲端列印解決方案和概念的基本知識。如需詳細資訊，請參閱 LRS 文件中的輸出現代化。
- Rocket Enterprise Server 軟體和授權。如需詳細資訊，請聯絡 Rocket Software。
- LRS VPSX/MFI、LRS PageCenterX、LRS/Queue 和 LRS/DIS 軟體和授權。如需詳細資訊，請聯絡 LRS。您必須提供將安裝 LRS 產品的 EC2 執行個體主機名稱。

注意：如需大型主機輸出管理工作負載組態考量的詳細資訊，請參閱此模式額外資訊區段中的考量事項。

產品版本

- Rocket Enterprise Server 10.0
- LRS VPSX/MFI
- LRS PageCenterX V1R3 或更新版本

Architecture

來源技術堆疊

- 作業系統 – IBM z/OS
- 程式設計語言 – 常見商業導向語言 (COBOL)、工作控制語言 (JCL) 和客戶資訊控制系統 (CICS)
- 資料庫 – z/OS 的 IBM Db2、IBM 資訊管理系統 (IMS) 資料庫和虛擬儲存存取方法 (VSAM)
- 安全性 – 資源存取控制設施 (RACF)、z/OS 的 CA 最高秘密，以及存取控制設施 2 (ACF2)
- 列印和封存解決方案 – IBM 大型主機 z/OS 輸出和列印產品 （適用於 z/OS、LRS 和 CA 交付的 IBM Infoprint 伺服器） 和封存解決方案 (CA 交付、ASG Mobius 或 CA 套件）

來源架構

下圖顯示大型主機輸出管理工作負載的典型目前狀態架構。

該圖顯示以下工作流程：

1. 使用者在以 COBOL 撰寫的 IBM CICS 應用程式上建置的參與系統 (SoE) 上執行商業交易。
2. SoE 會叫用大型主機服務，該服務會將商業交易資料記錄在system-of-records(SoR) 資料庫中，例如 z/OS 的 IBM Db2。
3. SoR 會保留來自 SoE 的業務資料。
4. 批次任務排程器會啟動批次任務以產生列印輸出。
5. 批次任務會從資料庫擷取資料。它會根據業務需求格式化資料，然後產生業務輸出，例如帳單、ID 卡或貸款陳述式。最後，批次任務會根據業務需求，將輸出路由至輸出管理的輸出格式、發佈和儲存。

6. 輸出管理會從批次任務接收輸出。輸出管理索引、排列輸出，並將輸出發佈至輸出管理系統中的指定目的地，例如 LRS PageCenterX 解決方案（如此模式所示）或 CA 檢視。
7. 使用者可以檢視、搜尋和擷取輸出。

目標技術堆疊

- 作業系統 – 在 Amazon EC2 上執行的 Windows Server
- 運算 – Amazon EC2
- 儲存 – Amazon Elastic Block Store (Amazon EBS) 和 Amazon FSx for Windows File Server
- 程式設計語言 – COBOL、JCL 和 CICS
- 資料庫 – Amazon RDS
- 安全性 – AWS Managed Microsoft AD
- 列印和封存 – AWS 上的 LRS 列印 (VPSX) 和封存 (PageCenterX) 解決方案
- 大型主機執行期環境 – Rocket Enterprise Server

目標架構

下圖顯示部署在 AWS 雲端中大型主機輸出管理工作負載的架構。

該圖顯示以下工作流程：

1. 批次任務排程器會啟動批次任務來建立輸出，例如帳單陳述式、ID 卡或貸款陳述式。
2. 大型主機批次工作 (已修改為 Amazon EC2) 使用 Rocket Enterprise Server 執行期從應用程式資料庫擷取資料、將商業邏輯套用至資料，以及格式化資料。然後，它會使用 Rocket Software 印表機結束模組 (OpenText Micro Focus 文件）將資料傳送至輸出目的地。
3. 應用程式資料庫（在 Amazon RDS 上執行的 SoR) 會保留列印輸出的資料。
4. LRS VPSX/MFI 列印解決方案部署在 Amazon EC2 上，其操作資料存放在 Amazon EBS 中。LRS VPSX/MFI 使用 TCP/IP 型 LRS/佇列傳輸代理程式，透過 Rocket Software JES Print Exit API 收集輸出資料。

   LRS VPSX/MFI 會進行資料預先處理，例如 EBCDIC 轉換為 ASCII。它也會執行更複雜的任務，包括將大型主機獨佔資料串流，例如 IBM Advanced Function Presentation (AFP) 和 Xerox Line Conditioned Data Stream (LCDS)，轉換為更常見的檢視和列印資料串流，例如印表機命令語言 (PCL) 和 PDF。

在 LRS PageCenterX 的維護時段期間，LRS VPSX/MFI 會保留輸出佇列，並做為輸出佇列的備份。LRS VPSX/MFI 會使用 LRS/佇列通訊協定來連接並傳送輸出至 LRS PageCenterX。LRS/Queue 會針對任務執行準備和完成的交換，以協助確保資料傳輸發生。

備註：

如需從 Rocket Software Print Exit 傳遞至 LRS/Queue 和 LRS VPSX/MFI 支援的大型主機批次機制之列印資料的詳細資訊，請參閱其他資訊一節中的列印資料擷取。

LRS VPSX/MFI 可以在印表機機群層級執行運作狀態檢查。如需詳細資訊，請參閱此模式額外資訊區段中的印表機機群運作狀態檢查。

5. LRS PageCenterX 輸出管理解決方案部署在 Amazon EC2 上，其操作資料存放在 Amazon FSx for Windows File Server 中。LRS PageCenterX 提供匯入 LRS PageCenterX 的所有檔案的中央報告管理系統，以及可存取檔案的所有使用者。使用者可以檢視特定檔案內容，或跨多個檔案執行搜尋，以取得相符條件。

   LRS/NetX 元件是多執行緒 Web 應用程式伺服器，可為 LRS PageCenterX 應用程式和其他 LRS 應用程式提供常見的執行時間環境。LRS/Web Connect 元件安裝在您的 Web 伺服器上，並提供從 Web 伺服器到 LRS/NetX Web 應用程式伺服器的連接器。
6. LRS PageCenterX 提供檔案系統物件的儲存體。LRS PageCenterX 的操作資料存放在 Amazon FSx for Windows File Server 中。
7. 輸出管理身分驗證和授權是由 AWS Managed Microsoft AD 搭配 LRS/DIS 執行。

> **Note**
>
> 目標解決方案通常不需要變更應用程式來適應大型主機格式語言，例如 IBM AFP 或 Xerox LCDS。

AWS 基礎設施架構

下圖顯示大型主機輸出管理工作負載的高可用性和安全 AWS 基礎設施架構。

該圖顯示以下工作流程：

1. 批次排程器會啟動批次程序，並跨多個可用區域部署在 Amazon EC2 上，以實現高可用性 (HA)。

> **Note**
>
> 此模式不包含批次排程器的實作。如需實作的詳細資訊，請參閱排程器的軟體廠商文件。

2. 大型主機批次工作（以程式設計語言撰寫，例如 JCL 或 COBOL) 使用核心商業邏輯來處理和產生列印輸出，例如帳單、ID 卡和貸款陳述式。批次任務會跨 HA 的兩個可用區域部署在 Amazon EC2 上。它使用 Rocket Software Print Exit API 將列印輸出路由到 LRS VPSX/MFI 以進行資料預先處理。
3. LRS VPSX/MFI 列印伺服器部署在 Amazon EC2 上，跨 HA 的兩個可用區域（主動待命備援對）。它使用 Amazon EBS 做為操作資料存放區。Network Load Balancer 會對 LRS VPSX/MFI EC2 執行個體執行運作狀態檢查。如果作用中執行個體處於運作狀態不良狀態，負載平衡器會將流量路由到其他可用區域中的熱待命執行個體。列印請求會保留在每個 EC2 執行個體的本機 LRS 任務佇列中。發生故障時，必須先重新啟動失敗的執行個體，LRS 服務才能繼續處理列印請求。

> **Note**
>
> LRS VPSX/MFI 也可以在印表機機群層級執行運作狀態檢查。如需詳細資訊，請參閱此模式額外資訊區段中的印表機機群運作狀態檢查。

4. LRS PageCenterX 輸出管理部署在 Amazon EC2 上，橫跨 HA 的兩個可用區域（主動待命備援對）。它使用 Amazon FSx for Windows File Server 作為操作資料存放區。如果作用中執行個體處於運作狀態不良狀態，負載平衡器會對 LRS PageCenterX EC2 執行個體執行運作狀態檢查，並將流量路由到其他可用區域中的待命執行個體。
5. Network Load Balancer 提供 DNS 名稱，以整合 LRS VPSX/MFI 伺服器與 LRS PageCenterX。

> **Note**
>
> LRS PageCenterX 支援第 4 層負載平衡器。

6. LRS PageCenterX 使用 Amazon FSx for Windows File Server 做為作業資料存放區，部署在 HA 的兩個可用區域。LRS PageCenterX 只了解檔案共享中的檔案，而不是外部資料庫中的檔案。
7. AWS Managed Microsoft AD 會與 LRS/DIS 搭配使用，以執行輸出管理工作流程身分驗證和授權。如需詳細資訊，請參閱其他資訊區段中的列印輸出身分驗證和授權。

工具

AWS 服務

- AWS Directory Service for Microsoft Active Directory 可讓您的目錄感知工作負載和 AWS 資源在 AWS 雲端中使用 Microsoft Active Directory。
- Amazon Elastic Block Store (Amazon EBS) 提供區塊層級儲存磁碟區，可與 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體搭配使用。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Elastic Load Balancing (ELB) 會將傳入的應用程式或網路流量分配到多個目標。例如，您可以在一或多個可用區域中跨 Amazon EC2 執行個體、容器和 IP 地址分配流量。此模式使用 Network Load Balancer。
- Amazon FSx 提供檔案系統，可支援業界標準的連線通訊協定，並跨 AWS 區域提供高可用性和複寫。此模式使用 Amazon FSx for Windows File Server。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。

其他工具

- LRS PageCenterX 軟體提供可擴展的文件和報告內容管理解決方案，協助使用者透過自動化索引、加密和進階搜尋功能，從資訊中取得最大值。
- LRS VPSX/MFI (Micro Focus Interface)，由 LRS 和 Rocket Software 編寫，可從 Rocket Software JES 多工緩衝區擷取輸出，並可靠地將其交付至指定的列印目的地。
- LRS/Queue 是以 TCP/IP 為基礎的傳輸代理程式。LRS VPSX/MFI 使用 LRS/佇列透過 Rocket Software JES Print Exit 程式設計界面收集或擷取列印資料。
- 在列印工作流程期間，LRS Directory Integration Server (LRS/DIS) 用於身分驗證和授權。
- Rocket Enterprise Server 是大型主機應用程式的應用程式部署環境。它為使用任何版本的 Rocket Enterprise Developer 遷移或建立的大型主機應用程式提供執行期環境。

史詩

設定 Rocket 執行時間並部署大型主機批次應用程式

任務	Description	所需的技能
設定執行時間並部署示範應用程式。	若要在 Amazon EC2 上設定 Rocket Enterprise Server 並部署 Rocket Software BankDemo 示範應用程式，請遵循 AWS Mainframe Modernization 使用者指南中的指示。 BankDemo 應用程式是一種大型主機批次應用程式，可建立並啟動列印輸出。	雲端架構師

在 Amazon EC2 上設定 LRS 列印伺服器

任務	Description	所需的技能
建立 Amazon EC2 Windows 執行個體。	若要啟動 Amazon EC2 Windows 執行個體，請遵循 Amazon EC2 文件中的啟動 Amazon EC2 執行個體中的指示。 Amazon EC2 使用與 LRS 產品授權相同的主機名稱。 您的執行個體必須符合 LRS VPSX/MFI 的下列硬體和軟體需求： • CPU – 雙核心 • RAM – 16 GB • 磁碟機 – 500 GB	雲端架構師

任務	Description	所需的技能
	• 最小 EC2 執行個體 – `m5.xlarge` • 作業系統 – Windows • 軟體 – 網路資訊服務 (IIS) 或 Apache **Note** 上述硬體和軟體需求適用於小型印表機機群（約 500-1000)。若要取得完整需求，請洽詢您的 LRS 和 AWS 聯絡人。 1. 當您建立 Windows 執行個體時，請確認 EC2 主機名稱與用於 LRS 產品授權的主機名稱相同。 2. 依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線至您的 EC2 執行個體。 Amazon EC2 3. 在 Windows 開始功能表上，尋找並開啟 Server Manager。 4. 在伺服器管理員中，選擇儀表板、Quick Start、新增角色和功能，然後選擇伺服器角色。	

任務	Description	所需的技能
	5. 在伺服器角色中，選擇 WebServer (IIS)，然後選擇應用程式開發。 6. 在應用程式開發中，選取 CGI 核取方塊。 7. 若要安裝 CGI，請遵循 Windows Server 管理員新增角色和功能精靈中的指示。 8. 在 EC2 執行個體的 Windows 防火牆中開啟連接埠 5500，以進行 LRS/佇列通訊。	

任務	Description	所需的技能
在 EC2 執行個體上安裝 LRS VPSX/MFI。	1. 連線至 EC2 執行個體。 2. 從您應該收到的 LRS 電子郵件訊息開啟產品下載頁面的連結。 **Note** LRS 產品透過電子檔案傳輸 (EFT) 進行分佈。 3. 下載 LRS VPSX/MFI，然後解壓縮 檔案 （預設資料夾：`c:\LRS`)。 4. 若要安裝 LRS VPSX/MFI，請從解壓縮資料夾啟動 LRS 產品安裝程式。 5. 在選取功能選單上，選取 VPSX® 伺服器，然後選擇下一步以開始安裝程序。安裝完成時，您會收到成功訊息。	雲端架構師
安裝 LRS/佇列。	1. 連線至 Rocket Enterprise Server EC2 執行個體。 2. 從您應該收到的 LRS 電子郵件訊息開啟 LRS 產品下載頁面的連結，下載 LRS/佇列，然後解壓縮檔案。 3. 導覽至您下載檔案的位置，然後啟動 LRS 產品安裝程式來安裝 LRS/佇列。 4. 請依照 LRS 產品安裝程式中的指示完成安裝程序。	雲端架構師

任務	Description	所需的技能
安裝 LRS/DIS。	LRS/DIS 產品通常包含在 LRS VPSX 安裝中。不過，如果 LRS/DIS 未與 LRS VPSX 一起安裝，請使用下列步驟進行安裝： 1. 連線至您的 LRS VPSX/MFI EC2 執行個體。 2. 從您應該收到的 LRS 電子郵件訊息開啟 LRS 產品下載頁面的連結，下載 LRS/DIS，然後解壓縮檔案。 3. 導覽至您下載檔案的位置，然後啟動 LRS 產品安裝程式。 4. 在 LRS 產品安裝程式中，展開 LRS 其他工具，選取 LRS DIS，然後選擇下一步。 5. 請遵循 LRS 產品安裝程式中的其餘說明來完成安裝程序。	雲端架構師

任務	Description	所需的技能
建立目標群組。	遵循為 Network Load Balancer 建立目標群組中的指示來建立目標群組。當您建立目標群組時，請將 LRS VPSX/MFI EC2 執行個體註冊為目標： 1. 在指定群組詳細資訊頁面上，針對選擇目標類型，選擇執行個體。 2. 針對通訊協定， 選擇 TCP。 3. 針對連接埠，選擇 5500。 4. 在註冊目標頁面的可用執行個體區段中，選取 LRS VPSX/MFI EC2 執行個體。	雲端架構師
建立 Network Load Balancer。	若要建立 Network Load Balancer，請遵循 Elastic Load Balancing 文件中的指示。Network Load Balancer 會將流量從 Rocket Enterprise Server 路由到 LRS VPSX/MFI EC2 執行個體。 當您建立 Network Load Balancer 時，請在接聽程式和路由頁面上選擇下列值： 1. 針對 Protocol (通訊協定)，選擇 TCP。 2. 針對連接埠，選擇 5500。 3. 針對預設動作，針對您先前建立的目標群組選擇轉送至。	雲端架構師

整合 Rocket Enterprise Server 與 LRS/Queue 和 LRS VPSX/MFI

任務	Description	所需的技能
設定 Rocket Enterprise Server for LRS/Queue 整合。	1. 遵循 Amazon EC2 文件中的指示，連線至 Rocket Enterprise Server EC2 執行個體。 Amazon EC2 2. 在 Windows 開始功能表上，開啟 Rocket Enterprise Server 管理 UI。 3. 在選單列中，選擇 NATIVE。 4. 在導覽窗格中，選擇目錄伺服器，然後為您的企業伺服器區域選擇 BANKDEMO。 5. 從左側導覽窗格中的一般，向下捲動至其他區段，以設定環境變數 (LRSQ_ADDRESS 、LRSQ_PORT 、LRSQ_COMMAND) 以指向 LRSQ。 • 針對 LRSQ_ADDRESS，輸入您先前建立之 Network Load Balancer 的 IP 地址或 DNS 名稱。 • 對於 LRSQ_PORT，輸入 VPSX LRSQ 接聽程式連接埠 (5500)。 • 對於 LRSQ_COMMAND，輸入 LRSQ 可執行檔的路徑位置。	雲端架構師

任務	Description	所需的技能
	Note LRS 目前支援 DNS 名稱的字元限制上限為 50。如果您的 DNS 名稱超過 50 個字元，您可以使用 Network Load Balancer 的 IP 地址做為替代方案。	

任務	Description	所需的技能
設定 Rocket Enterprise Server for LRS VPSX/MFI 整合。	1. 將`VPSX_MFI_R2` 資料夾從 LRS VPSX/MFI 安裝程式複製到位於 的 Rocket Enterprise Server 位置`C\BANKDEMO\print` 。 2. 遵循 Amazon EC2 文件中的指示，連線至 Rocket Enterprise Server EC2 執行個體。 Amazon EC2 3. 在 Windows 開始功能表上，開啟 Rocket Enterprise Server 管理 UI。 4. 在選單列上，選擇 NATIVE。 5. 在導覽窗格中，選擇目錄伺服器，然後選擇 BANKDEMO。 6. 在 BANKDEMO 下，選擇 JES。 7. 在 JES 程式路徑下，從 新增`DLL(VPSX_MFI_R2)` 路徑`C\BANKDEMO\print` 。	雲端架構師

設定列印佇列和列印使用者

任務	Description	所需的技能
將 Rocket Software Print Exit 模組與 Rocket Enterprise Server 批次印表機伺服器執行程序建立關聯。	1. 遵循 Amazon EC2 文件中的指示，連線至 Rocket Enterprise Server EC2 執行個體。 Amazon EC2	雲端架構師

任務	Description	所需的技能
	2. 在 Windows 開始功能表上，開啟 Rocket Focus Enterprise Server 管理 UI。 3. 在選單列上，選擇 NATIVE。 4. 在導覽窗格中，選擇目錄伺服器，然後選擇 BANKDEMO。 5. 在 BANKDEMO 下，選擇 JES，然後向下捲動至印表機。 6. 在印表機中，將 Rocket Software Print Exit 模組 (LRSPRTE6 for Batch) 與 Rocket Enterprise Server 批次印表機伺服器執行程序 (SEP) 建立關聯。這可讓列印輸出路由至 LRS VPSX/MFI。 如需組態的詳細資訊，請參閱 OpenText Micro Focus 文件中的使用結束。	

任務	Description	所需的技能
在 LRS VPSX/MFI 中建立列印輸出佇列，並將其與 LRS PageCenterX 整合。	1. 連線至您的 LRS VPSX/MFI EC2 執行個體。 2. 在 Windows 開始功能表上，開啟 VPSX Web 介面。 3. 在導覽窗格中，選擇印表機。 4. 選擇新增，然後選擇新增印表機。 5. 在印表機組態頁面上，針對印表機名稱，輸入 Local。 6. 針對 VPSX ID，輸入 VPS1。 7. 針對 CommType，選取 TCPIP/LRSQ。 8. 針對主機/IP 地址，輸入 LRS PageCenterX EC2 執行個體前方的 Network Load Balancer IP 地址。 9. 針對遠端連接埠，輸入 5800。 10針對遠端佇列，輸入將存放輸出的 LRS PageCenterX 文件資料夾名稱。 11選擇新增。	雲端架構師

任務	Description	所需的技能
在 LRS VPSX/MFI 中建立列印使用者。	1. 連線至您的 LRS VPSX/MFI EC2 執行個體。 2. 在 Windows 開始功能表上，開啟 VPSX Web 介面。 3. 在導覽窗格中，選擇安全性，然後選擇使用者。 4. 在使用者名稱欄中，選擇管理員，然後選擇複製。 5. 在使用者設定檔維護視窗中，針對使用者名稱輸入使用者名稱（例如 PrintUser）。 6. 針對描述，輸入簡短描述（例如測試列印的使用者)。 7. 選擇更新。這會建立列印使用者（例如 PrintUser)。 8. 在導覽窗格的使用者下， 選擇您建立的新使用者。 9. 在命令功能表中，選擇安全性。 10.在安全規則頁面上，選擇所有適用的印表機安全和任務安全選項，然後選擇儲存。 11.若要將您的新列印使用者新增至管理員群組，請在導覽窗格中選擇安全性，然後選擇設定。 12.在安全組態視窗中，將新的列印使用者新增至管理員欄。	雲端架構師

在 Amazon EC2 上設定 LRS PageCenterX 伺服器

任務	Description	所需的技能
建立 Amazon EC2 Windows 執行個體。	按照 Amazon EC2 文件中的步驟 1：啟動執行個體中的指示啟動 Amazon EC2 Windows 執行個體。 Amazon EC2 使用您用於 LRS 產品授權的相同主機名稱。 您的執行個體必須符合 LRS PageCenterX 的下列硬體和軟體需求： • CPU – 雙核心 • RAM – 16 GB • 磁碟機 – 500 GB • 最小 EC2 執行個體 – `m5.xlarge` • 作業系統 – Windows • 軟體 –IIS 或 Apache **Note** 上述硬體和軟體需求適用於小型印表機機群（約 500–1000）。若要取得完整需求，請洽詢您的 LRS 和 AWS 聯絡人。 1. 當您建立 Windows 執行個體時，請確認 EC2 主機名稱	雲端架構師

任務	Description	所需的技能
	與用於 LRS 產品授權的主機名稱相同。 2. 遵循 Amazon EC2 文件中的指示，連線至 EC2 執行個體。 Amazon EC2 3. 在 Windows 開始功能表上，尋找並開啟 Server Manager。 4. 在伺服器管理員中，選擇儀表板、Quick Start、新增角色和功能，然後選擇伺服器角色。 5. 在伺服器角色中，選擇 WebServer (IIS)，然後選擇應用程式開發。 6. 在應用程式開發中，選取 CGI 核取方塊。 7. 若要安裝 CGI，請遵循 Windows Server 管理員新增角色和功能精靈中的指示。 8. 為 EC2 執行個體的 Windows 防火牆中的傳入 TCP/IP 流量開啟連接埠 5800。LRS VPSX 在 5800 連接埠上使用 TCPIP/LRSQ 通訊協定來與 LRS PageCenterX 通訊。	

任務	Description	所需的技能
在 EC2 執行個體上安裝 LRS PageCenterX。	1. 連線至 EC2 執行個體。 2. 從您應該收到的 LRS 電子郵件訊息開啟產品下載頁面的連結。 **Note** LRS 產品透過電子檔案傳輸 (EFT) 進行分佈。 3. 下載 LRS PageCenterX，然後解壓縮 檔案 （預設資料夾：c:\LRS)。 4. 若要安裝 LRS PageCenterX，請從解壓縮的資料夾啟動 LRS 產品安裝程式。 5. 在選取功能功能表上，選取 PageCenterX，然後選擇下一步以開始安裝程序。安裝完成時，您會收到成功訊息。	雲端架構師

任務	Description	所需的技能
安裝 LRS/DIS。	LRS/DIS 產品通常包含在 LRS VPSX 安裝中。不過，如果 LRS/DIS 未與 LRS VPSX 一起安裝，請使用下列步驟進行安裝： 1. 連線至您的 LRS PageCenterX EC2 執行個體。 2. 從您應該收到的 LRS 電子郵件開啟 LRS 產品下載頁面的連結，下載 LRS/DIS，然後解壓縮檔案。 3. 導覽至您下載檔案的位置，然後啟動 LRS 產品安裝程式。 4. 在 LRS 產品安裝程式中，展開 LRS 其他工具，選取 LRS DIS，然後選擇下一步。 5. 請遵循 LRS 產品安裝程式中的其餘說明來完成安裝程序。	雲端架構師

任務	Description	所需的技能
建立目標群組。	遵循為 Network Load Balancer 建立目標群組中的指示來建立目標群組。當您建立目標群組時，請將 LRS PageCenterX EC2 執行個體註冊為目標： 1. 在指定群組詳細資訊頁面上，針對選擇目標類型，選擇執行個體。 2. 針對通訊協定， 選擇 TCP。 3. 針對連接埠，選擇 5800。 4. 在註冊目標頁面的可用執行個體區段中，選取 LRS PageCenterX EC2 執行個體。	雲端架構師

任務	Description	所需的技能
建立 Network Load Balancer。	若要建立 Network Load Balancer，請遵循 Elastic Load Balancing 文件中的指示。Network Load Balancer 會將流量從 LRS VPSX/MFI 路由到 LRS PageCenterX EC2 執行個體。 當您建立 Network Load Balancer 時，請在接聽程式和路由頁面上選擇下列值： 1. 針對 Protocol (通訊協定)，選擇 TCP。 2. 針對連接埠，選擇 5800。 3. 針對預設動作，針對您先前建立的目標群組選擇轉送至。	雲端架構師

在 LRS PageCenterX 中設定輸出管理功能

任務	Description	所需的技能
在 LRS PageCenterX 中啟用匯入函數。	您可以使用 LRS PageCenterX 匯入函數，根據任務名稱或表單 ID 等條件來辨識 LRS PageCenterX 上的輸出登陸。然後，您可以將輸出路由到 LRS PageCenterX 中的特定資料夾。 若要啟用匯入函數，請執行下列動作：	雲端架構師

任務	Description	所需的技能
	1. 遵循 Amazon EC2 文件中的指示，連線至 LRS PageCenterX EC2 執行個體。 Amazon EC2 2. 在 Windows 開始功能表上，開啟 PCX Web 介面。 3. 在資料夾總管中，選擇管理員。 4. 在組態頁面上，選擇進階、匯入參數。 5. 在匯入參數區段中，選取進階匯入核取方塊。 6. 若要遞交變更，請選擇更新。	

任務	Description	所需的技能
設定文件保留政策。	LRS PageCenterX 使用文件保留政策來決定在 LRS PageCenterX 中保留文件的時間長度。 若要設定文件保留政策，請執行下列動作： 1. 連線至您的 LRS PageCenterX EC2 執行個體。 2. 在 Windows 開始功能表上，開啟 PCX Web 介面。 3. 在資料夾總管中，選擇管理員。 4. 在管理員頁面上，選擇封存群組清單/一般管理員， 然後選擇保留政策。 5. 在保留政策區段中， 選擇新增以建立保留政策。 6. 在保留政策資訊頁面上，輸入保留政策名稱、描述和文件保留期間。 7. 若要儲存變更並建立政策，請選擇確定。	雲端架構師

任務	Description	所需的技能
建立規則，將輸出文件路由到 LRS PageCenterX 中的特定資料夾。	在 LRS PageCenterX 中，目的地會決定當報告定義調用此目的地時，輸出將傳送的資料夾路徑。在此範例中，根據報告定義中的表單 ID 資料夾建立資料夾，並將輸出儲存到該資料夾。 1. 連線至您的 LRS PageCenterX EC2 執行個體。 2. 在 Windows 開始 功能表上，開啟 PCX Web Interface。 3. 在資料夾總管中，選擇管理員、進階匯入、目的地。 4. 在目的地區段中，選擇新增以開啟目的地維護表單。 5. 在目的地維護 表單上，輸入下列值： • 目的地名稱 – 表單 • 描述 – 目的地的描述，例如表單型資料夾結構 • 目的地類型 – 資料夾 • 資料夾參數 – 匯入資料夾路徑 （文件送達時將在 PageCenterX 中建立的資料夾路徑；例如，路徑`/Test/&FORM/&IMPORTDATE/&IMPORTTIME` 會建立基本`Test`資料夾、以名為的 Form-Id 為基礎的子資料夾 STD、以匯入日期為	雲端架構師

任務	Description	所需的技能
	基礎的子資料夾，以及以匯入時間為基礎的子資料夾） • 文件名稱 – 當文件存放在資料夾中時，指派給文件的動態名稱。 6. 在下拉式清單中，選擇保留政策。例如，選擇 Year1 將文件保留 1 年。 7. 若要儲存變更，請選擇確定。	

任務	Description	所需的技能
建立報告定義。	1. 連線至您的 LRS PageCenterX EC2 執行個體。 2. 在 Windows 開始功能表上，開啟 PCX Web 介面。 3. 在資料夾總管中，選擇管理員、進階匯入、報告定義，然後選擇新增。 4. 在報告定義維護頁面上的一般索引標籤上，輸入報告定義名稱。 5. 在一般索引標籤的欄位下，您可以指定選擇條件，例如任務名稱、表單、類別和作者。例如，您可以輸入 MFIDEMO 的任務名稱。任務名稱值將是將產生列印輸出的批次任務的名稱。 6. 在目的地索引標籤的可用目的地下，選擇先前建立的目的地 (表單)。 7. 選擇新增，將表單目的地新增為已指派目的地。 **Note** 此範例包含報告定義，其中由 MFIDEMO 產生並路由至 LRS PageCenterX 的輸出會儲存在目的地定義中定義的資料夾結構中。	雲端架構師

設定輸出管理的身分驗證和授權

任務	Description	所需的技能
使用使用者和群組建立 AWS Managed Microsoft AD 網域。	1. 若要在 AWS Managed Microsoft AD 上建立目錄，請遵循建立 AWS Managed Microsoft AD 目錄中的指示。 2. 若要部署 EC2 執行個體 (Active Directory 管理員) 並安裝 Active Directory 工具來管理您的 AWS Managed Microsoft AD，請遵循步驟 3：部署 EC2 執行個體以管理您的 AWS Managed Microsoft AD 中的指示。 3. 若要連線至 EC2 執行個體，請遵循 Amazon EC2 文件中的指示。 **Note** 當您連線到 EC2 執行個體時，請在 Windows 安全視窗中，輸入您在步驟 1 中建立之目錄的管理員登入資料。 4. 在 Windows 開始功能表的 Windows 管理工具下，選擇 Active Directory 使用者和電腦。	雲端架構師

任務	Description	所需的技能
	5. 若要在 Active Directory 網域中建立列印使用者，請遵循建立使用者中的指示。	
將 EC2 執行個體加入 AWS Managed Microsoft AD 網域。	自動將 LRS VPSX/MFI 和 LRS PageCenterX EC2 執行個體加入您的 AWS Managed Microsoft AD 網域 (AWS 知識中心文件） 或手動 (AWS Directory Service 文件）。	雲端架構師
設定 LRS/DIS 並與 LRS PageCenterX EC2 執行個體的 AWS Managed Microsoft AD 整合。	1. 連線至您的 LRS PageCenterX EC2 執行個體。 2. 在 Windows 開始功能表上，開啟 PCX Web Interface。 3. 在資料夾總管中，選擇管理員。 4. 在組態頁面的安全參數區段中，針對安全類型選取 LRS/DIS。 5. 在安全參數區段中輸入其餘選項的偏好設定。 6. 在 Windows 開始功能表上，開啟 PageCenterX 資料夾，選擇伺服器開始，然後選擇伺服器停止。 7. 使用您的 Active Directory 使用者名稱和密碼登入 LRS PageCenterX。	雲端架構師

任務	Description	所需的技能
設定匯入群組，將輸出從 LRS VPSX 匯入 LRS PageCenterX。	1. 連線至您的 LRS PageCenterX EC2 執行個體。 2. 在 Windows 開始功能表上，開啟 PCX Web Interface。 3. 在資料夾總管中，選擇管理員、安全管理員、群組。 4. 在 群組區段中，選擇新增以開啟群組偏好設定表單。 5. 在群組偏好設定表單中，輸入群組名稱和描述的值。 6. 展開一般選項，然後選取匯入核取方塊。 7. 若要儲存變更，請選擇確定。	雲端架構師
將安全規則新增至匯入群組。	1. 開啟匯入群組的內容（按一下滑鼠右鍵）選單。 2. 選擇進階，然後選擇安全性。 3. 在安全性區段中，選擇匯入，然後選取子資料夾核取方塊。 4. 若要儲存變更，請選擇套用。	雲端架構師

任務	Description	所需的技能
在 LRS PageCenterX 中建立使用者，以從 LRS VPSX/MFI 執行輸出匯入。	當您在 LRS PageCenterX 中建立使用者以執行輸出匯入時，使用者名稱應與 LRS VPSX/MFI 中列印輸出佇列的 VPSX ID 相同。在此範例中，VPSX ID 為 VPS1。 1. 連線至您的 LRS PageCenterX EC2 執行個體。 2. 在 Windows 開始功能表上，開啟 PCX Web 介面。 3. 在 Folder Explorer 中，選擇 Admin、Security admin、User。 4. 選擇新增以開啟使用者設定檔維護表單。 5. 在使用者設定檔維護中，針對使用者名稱輸入 VPS1。	雲端架構師

任務	Description	所需的技能
將 LRS PageCenterX 匯入使用者新增至僅匯入群組。	若要提供從 LRS VPSX 匯入至 LRS PageCenterX 的必要許可，請執行下列動作： 1. 連線至您的 LRS PageCenterX EC2 執行個體。 2. 在 Windows 開始功能表上，開啟 PCX Web Interface。 3. 在資料夾總管中，選擇管理員、安全管理員、群組。 4. 在群組區段中，開啟僅限匯入群組的內容（按一下滑鼠右鍵）選單，然後選擇進階、安全性。 5. 在資料夾安全記錄 (ImportOnly) 頁面上，選擇使用者索引標籤。 6. 在使用者索引標籤的名稱下，從下拉式清單中選取使用者 VPS1，然後選擇套用。	雲端架構師

任務	Description	所需的技能
使用 LRS VPSX/MFI EC2 執行個體的 AWS Managed Microsoft AD 設定 LRS/DIS。	1. 連線至您的 LRS VPSX/MFI EC2 執行個體。 2. 在 Windows 開始功能表上，開啟 VPSX Web 介面。 3. 在導覽窗格中，選擇安全性，然後選擇設定。 4. 在安全組態頁面的安全參數區段中，針對安全類型，選取 LRS/DIS （外部）。 5. 在安全參數區段中輸入其餘選項的偏好設定。 6. 在 Windows 開始功能表上，開啟 LRS 輸出管理資料夾，選擇伺服器開始，然後選擇伺服器停止。 7. 使用您的 Active Directory 使用者名稱和密碼登入 LRS VPSX/MFI。	雲端架構師

將 Amazon FSx for Windows File Server 設定為 LRS PageCenterX 的操作資料存放區

任務	Description	所需的技能
建立 LRS PageCenterX 的檔案系統。	若要在多可用區環境中使用 Amazon FSx for Windows File Server 做為 LRS PageCenterX 的操作資料存放區，請遵循步驟 1：建立檔案系統中的指示。	雲端架構師
將檔案共享映射至 LRS PageCenterX EC2 執行個體。	若要將上一個步驟中建立的檔案共享映射到 LRS PageCente	雲端架構師

任務	Description	所需的技能
	rX EC2 執行個體，請遵循步驟 2：將您的檔案共享映射到執行 Windows Server 的 EC2 執行個體中的指示。	
將 LRS PageCenterX 控制目錄和主資料夾目錄映射至 Amazon FSx 網路共享磁碟機。	1. 遵循 Amazon EC2 文件中的指示，連線至 LRS PageCenterX EC2 執行個體。 Amazon EC2 2. 在 Windows 開始功能表上，開啟 PCX Web Interface。 3. 在資料夾總管中，選擇管理員、組態。 4. 在組態頁面上，選擇目錄，然後選擇控制目錄。 5. 在控制目錄中，輸入 `\\FSx file share DNS name \share\cntl` 。 6. 在主資料夾目錄中，輸入 `\\FSx file share DNS name\share\mstr` 。	雲端架構師

測試輸出管理工作流程

任務	Description	所需的技能
從 Rocket Software BankDemo 應用程式啟動批次列印請求。	1. 在 Rocket Enterprise Server EC2 執行個體中開啟 3270 終端模擬器。 2. 執行命令 以連線至 BankDemo 應用程	測試工程師

任務	Description	所需的技能
	式`connect 127.0.0.1:9278` 。 3. 在 BankDemo 命令列界面上，針對使用者 ID，輸入 B0001。針對密碼，輸入非空白金鑰。 4. 針對請求列印的陳述式 (s) 選項，在空白行中輸入 X。 5. 在依 傳送陳述式區段中，針對郵件輸入 Y，然後按 F10。	

任務	Description	所需的技能
檢查 LRS PageCenterX 中的列印輸出。	1. 遵循 Amazon EC2 文件中的指示，連線至您的 LRS PageCenterX EC2 執行個體。 Amazon EC2 2. 在 Windows 開始功能表上，開啟 PCX Web Interface。 3. 在導覽窗格中，開啟測試資料夾，開啟 STD 資料夾，然後開啟具有任務執行日期的資料夾，例如 `08-03-2023` (MM-DD-YYYY)。 **Note** 這與案例中定義的資料夾結構相同 建立規則，將輸出文件路由到 LRS PageCenterX 中的特定資料夾。 4. 開啟 `formtest-STD.txt` 檔案。 您現在可以查看帳戶陳述式的列印輸出，其中包含帳戶編號、描述、日期、金額和餘額的資料欄。如需範例，請參閱此模式的`batch_print_output` 附件。	測試工程師

相關資源

- LRS
- 進階函數簡報資料串流 (IBM 文件）
- Line Conditioned Data Stream (LCDS)（組件文件）
- 使用 Micro Focus 在 AWS 上增強企業大型主機工作負載（部落格文章）
- 在 AWS 上現代化大型主機線上列印工作負載 (AWS 規範指引）
- 在 AWS 上現代化大型主機批次列印工作負載 (AWS 規範指引）

其他資訊

考量

在現代化旅程中，您可能會考慮大型主機批次和線上程序的各種組態，以及它們產生的輸出。大型主機平台是由每個使用大型主機平台的客戶和廠商自訂，其具有直接影響列印的特定要求。例如，您目前的平台可能會將 IBM AFP 資料串流或 Xerox LCD 納入目前的工作流程。此外，大型主機運輸控制字元和頻道命令文字可能會影響列印頁面的外觀，而且可能需要特殊處理。作為現代化規劃程序的一部分，我們建議您評估並了解特定列印環境中的組態。

列印資料擷取

Rocket Software Print Exit 會傳遞 LRS VPSX/MFI 的必要資訊，以有效地處理多工緩衝處理檔案。資訊由相關控制區塊中傳遞的欄位組成，如下所示：

- JOBNAME
- OWNER (USERID)
- 目的地
- FORM
- FILENAME
- 寫入

LRS VPSX/MFI 支援下列大型主機批次機制，用於從 Rocket Enterprise Server 擷取資料：

- 使用標準 z/OS JCL SYSOUT DD/OUTPUT 陳述式進行批次 COBOL 列印/多工緩衝處理。
- 使用標準 z/OS JCL CA-SPOOL SUBSYS DD 陳述式進行批次 COBOL 列印/多工緩衝處理。

- IMS/COBOL 列印/多工緩衝處理使用 CBLTDLI 界面。如需支援的方法和程式設計範例的完整清單，請參閱產品授權隨附的 LRS 文件。

印表機機群運作狀態檢查

LRS VPSX/MFI (LRS LoadX) 可執行深入研究運作狀態檢查，包括裝置管理和操作最佳化。裝置管理可以偵測印表機裝置中的失敗，並將列印請求路由到運作狀態良好的印表機。如需印表機機群深入運作狀態檢查的詳細資訊，請參閱產品授權隨附的 LRS 文件。

列印身分驗證和授權

LRS/DIS 可讓 LRS 應用程式使用 Microsoft Active Directory 或輕量型目錄存取協定 (LDAP) 伺服器來驗證使用者 IDs 和密碼。除了基本列印授權之外，LRS/DIS 也可以在下列使用案例中套用精細層級的列印安全控制：

- 管理誰可以瀏覽印表機任務。
- 管理其他使用者任務的瀏覽層級。
- 管理操作任務 - 例如，命令層級安全性，例如保留或釋出、清除、修改、複製和重新路由。安全可由 user-ID 或 群組設定，類似於 Active Directory 安全群組或 LDAP 群組。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Rocket Enterprise Server 和 LRS VPSX/MFI AWS 在 上現代化大型主機批次列印工作負載

Shubham Roy 和 Kevin Yung，Amazon Web Services

Abraham Rondon，微型焦點

Guy Tucker、Levi、Ray 和 Shoup Inc

總結

此模式說明如何使用 Rocket Enterprise Server 做為現代化大型主機應用程式的執行期，以及使用 LRS VPSX/MFI (Micro Focus Interface) 做為列印伺服器，在 Amazon Web Services (AWS) 雲端上現代化業務關鍵型大型主機批次列印工作負載。模式是以轉換大型主機現代化方法為基礎。在此方法中，您將大型主機批次任務遷移至 Amazon Elastic Compute Cloud (Amazon EC2)，並將大型主機資料庫遷移至 Amazon Relational Database Service (Amazon RDS)，例如 z/OS 的 IBM DB2。現代化列印工作流程的身分驗證和授權是由 AWS Directory Service for Microsoft Active Directory 執行，也稱為 AWS Managed Microsoft AD。LRS Directory Information Server (LRS/DIS) 已與 AWS Managed Microsoft AD 整合。透過現代化批次列印工作負載，您可以降低 IT 基礎設施成本、減輕維護舊版系統的技術負債、移除資料孤島、使用 DevOps 模型提高敏捷性和效率，以及利用 AWS 雲端中的隨需資源和自動化。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 大型主機列印或輸出管理工作負載
- 如何重建和交付在 Rocket Enterprise Server 上執行之大型主機應用程式的基本知識 （ 如需詳細資訊，請參閱 Rocket 文件中的 Rocket Enterprise Server 資料表。）
- LRS 雲端列印解決方案和概念的基本知識
- Rocket Enterprise Server 軟體和授權 （ 如需詳細資訊，請聯絡 Rocket 銷售。）
- LRS VPSX/MFI、LRS/Queue 和 LRS/DIS 軟體和授權 （ 如需詳細資訊，請聯絡 LRS 銷售)。

> **Note**
>
> 如需大型主機批次列印工作負載組態考量的詳細資訊，請參閱此模式額外資訊區段中的考量事項。

產品版本

- Rocket Enterprise Server 6.0 （ 產品更新 7)
- LRS VPSX/MFI V1R3 或更高版本

Architecture

來源技術堆疊

- 作業系統 – IBM z/OS
- 程式設計語言 – 常見商業導向語言 (COBOL)、工作控制語言 (JCL) 和客戶資訊控制系統 (CICS)
- 資料庫 – z/OS 的 IBM DB2 和虛擬儲存存取方法 (VSAM)
- 安全性 – 資源存取控制設施 (RACF)、z/OS 的 CA 最高秘密，以及存取控制設施 2 (ACF2)
- 列印和輸出管理 – IBM 大型主機 z/OS 列印產品 （ 適用於 z/OS、LRS 和 CA 檢視的 IBM Tivoli 輸出管理員 ）

目標技術堆疊

- 作業系統 – 在Amazon EC2 上執行的 Microsoft Windows Server
- 運算 – Amazon EC2
- 程式設計語言 – COBOL、JCL 和 CICS
- 資料庫 – Amazon RDS
- 安全性 – AWS Managed Microsoft AD
- 列印和輸出管理 – AWS 上的 LRS 列印解決方案
- 大型主機執行期環境 – Rocket Enterprise Server

來源架構

下圖顯示大型主機批次列印工作負載的典型目前狀態架構：

該圖顯示以下工作流程：

1. 使用者在以 COBOL 撰寫的 IBM CICS 應用程式上建置的參與系統 (SoE) 上執行商業交易。
2. SoE 會叫用大型主機服務，將商業交易資料記錄在system-of-records(SoR) 資料庫中，例如 z/OS 的 IBM DB2。
3. SoR 會保留來自 SoE 的業務資料。
4. 批次任務排程器會啟動批次任務以產生列印輸出。
5. 批次任務會從資料庫擷取資料、根據業務需求格式化資料，然後產生業務輸出，例如帳單、ID 卡或貸款陳述式。最後，批次任務會根據業務需求，將輸出路由到列印輸出管理以進行處理和輸出交付。
6. 列印輸出管理會從批次任務接收列印輸出，然後將該輸出交付至指定的目的地，例如電子郵件、使用安全 FTP 的檔案共用、使用 LRS 列印解決方案的實體印表機（如此模式所示）或 IBM Tivoli。

目標架構

下圖顯示部署在 AWS 雲端中大型主機批次列印工作負載的架構：

該圖顯示以下工作流程：

1. 批次任務排程器會啟動批次任務來建立列印輸出，例如帳單陳述式、ID 卡或貸款陳述式。
2. 大型主機批次工作 (已修改為 Amazon EC2) 使用 Rocket Enterprise Server 執行期從應用程式資料庫擷取資料、將商業邏輯套用至資料、格式化資料，然後使用 Rocket Software Print Exit (Micro Focus 文件）將資料傳送至列印目的地。
3. 應用程式資料庫（在 Amazon RDS 上執行的 SoR) 會保留列印輸出的資料。
4. LRS VPSX/MFI 列印解決方案部署在 Amazon EC2 上，其操作資料存放在 Amazon Elastic Block Store (Amazon EBS) 中。LRS VPSX/MFI 使用 TCP/IP 型 LRS/Queue 傳輸代理程式，透過 Rocket Software JES Print Exit API 收集列印資料，並將資料交付至指定的印表機目的地。

> **Note**
>
> 目標解決方案通常不需要應用程式變更來適應大型主機格式語言，例如 IBM 進階函數呈現 (AFP) 或 Xerox Line Condition Data Stream (LCDS)。如需在 AWS 上使用 Rocket Software

進行大型主機應用程式遷移和現代化的詳細資訊，請參閱使用 Micro Focus 在 AWS 上增強企業大型主機工作負載部落格文章。

AWS 基礎設施架構

下圖顯示適用於大型主機批次列印工作負載的高可用性和安全 AWS 基礎設施架構：

該圖顯示以下工作流程：

1. 批次排程器會啟動批次程序，並跨多個可用區域部署在 Amazon EC2 上，以實現高可用性 (HA)。

> **Note**
>
> 此模式不包含批次排程器的實作。如需實作的詳細資訊，請參閱排程器的軟體廠商文件。

2. 大型主機批次工作（以 JCL 或 COBOL 等程式設計語言撰寫）使用核心商業邏輯來處理和產生列印輸出，例如帳單、ID 卡和貸款陳述式。任務會跨 HA 的兩個可用區域部署在 Amazon EC2 上，並使用 Rocket Software Print Exit 將列印輸出路由到 LRS VPSX/MFI 以進行最終使用者列印。
3. LRS VPSX/MFI 使用 TCP/IP 型 LRS/佇列傳輸代理程式，從 Rocket Software JES Print Exit 程式設計界面收集或擷取列印資料。列印結束會傳遞必要資訊，讓 LRS VPSX/MFI 有效地處理多工緩衝處理檔案，並動態建置 LRS/佇列命令。接著會使用 Rocket Software 的標準內建函數執行命令。

> **Note**
>
> 如需從 Rocket Software Print Exit 傳遞至 LRS/Queue 和 LRS VPSX/MFI 支援之大型主機批次機制的列印資料的詳細資訊，請參閱此模式額外資訊區段中的列印資料擷取。

4.

> **Note**
>
> Network Load Balancer 提供 DNS 名稱，以整合 Rocket Enterprise Server 與 LRS VPSX/MFI。：LRS VPSX/MFI 支援第 4 層負載平衡器。Network Load Balancer 也會對 LRS VPSX/MFI 執行基本運作狀態檢查，並將流量路由至運作狀態良好的已註冊目標。

5.

> **Note**
>
> LRS VPSX/MFI 列印伺服器會跨 HA 的兩個可用區域部署在 Amazon EC2 上，並使用 Amazon EBS 做為操作資料存放區。LRS VPSX/MFI 支援主動-主動和主動-被動服務模式。

此架構使用主動-被動對中的多個 AZs 作為主動和熱待命。Network Load Balancer 會對 LRS VPSX/MFI EC2 執行個體執行運作狀態檢查，並在作用中執行個體處於運作狀態不佳時，將流量路由到其他 AZ 中的熱待命執行個體。列印請求會保留在每個 EC2 執行個體的本機 LRS 任務佇列中。在復原的情況下，必須重新啟動失敗的執行個體，LRS 服務才能繼續處理列印請求。 ：LRS VPSX/MFI 也可以在印表機機群層級執行運作狀態檢查。如需詳細資訊，請參閱此模式額外資訊區段中的印表機機群運作狀態檢查。

6. AWS Managed Microsoft AD 與 LRS/DIS 整合，以執行列印工作流程身分驗證和授權。如需詳細資訊，請參閱此模式額外資訊區段中的列印身分驗證和授權。
7. LRS VPSX/MFI 使用 Amazon EBS 進行區塊儲存。您可以將 Amazon EBS 資料以point-in-time快照的形式從作用中的 EC2 執行個體備份到 Amazon S3，並將其還原至熱待命 EBS 磁碟區。若要自動建立、保留和刪除 Amazon EBS 磁碟區快照，您可以使用 Amazon Data Lifecycle Manager 來設定自動快照的頻率，並根據 RTO/RPO 需求還原快照。

工具

AWS 服務

- Amazon Elastic Block Store (Amazon EBS) 提供區塊層級儲存磁碟區，可與 EC2 執行個體搭配使用。EBS 磁碟區的行為與未格式化的原始區塊型儲存設備相似。您可以將這些磁碟區做為裝置，掛載在您的執行個體上。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以使用 Amazon EC2 視需要啟動任意數量或任意數量的虛擬伺服器，也可以向外擴展或向內擴展。
- Amazon Relational Database Service (Amazon RDS) 是一種 Web 服務，可讓您更輕鬆地在 AWS 雲端中設定、操作和擴展關聯式資料庫。它為關聯式資料庫提供經濟實惠、可擴展的容量，並管理常見的資料庫管理任務。
- AWS Directory Service for Microsoft Active Directory，也稱為 AWS Managed Microsoft AD，可讓您的目錄感知工作負載和 AWS 資源在 AWS 雲端中使用 Microsoft Active Directory。

其他工具

- LRS VPSX/MFI (Micro Focus Interface)，由 LRS 和 Rocket Software 共同開發，可從 Rocket Enterprise Server JES 多工緩衝區擷取輸出，並可靠地將其交付至指定的列印目的地。
- LRS Directory Information Server (LRS/DIS) 用於在列印工作流程期間進行身分驗證和授權。
- LRS VPSX/MFI 使用 TCP/IP 型 LRS/佇列傳輸代理程式，透過 Rocket Software JES Print Exit 程式設計界面收集或擷取列印資料。

- Rocket Enterprise Server 是大型主機應用程式的應用程式部署環境。它為使用任何版本的 Rocket Software Enterprise Developer 遷移或建立的大型主機應用程式提供執行環境。

史詩

在 Amazon EC2 上設定 Rocket Enterprise Server 並部署大型主機批次應用程式

任務	Description	所需的技能
設定 Rocket Enterprise Server 並部署示範應用程式。	在 Amazon EC2 上設定 Rocket Enterprise Server，然後在 Amazon EC2 上部署 Rocket Software BankDemo 示範應用程式。 BankDemo 應用程式是一種大型主機批次應用程式，可建立並啟動列印輸出。	雲端架構師

在 Amazon EC2 上設定 LRS 列印伺服器

任務	Description	所需的技能
取得用於列印的 LRS 產品授權。	若要取得 LRS VPSX/MFI、LRS/Queue 和 LRS/DIS 的 LRS 產品授權，請聯絡 LRS 輸出管理團隊。您必須提供將安裝 LRS 產品的 EC2 執行個體的主機名稱。	建置潛在客戶
建立 Amazon EC2 Windows 執行個體以安裝 LRS VPSX/MFI。	依照 Amazon EC2 文件中啟動 Amazon EC2 執行個體的指示啟動 Amazon EC2 Windows 執行個體。 Amazon EC2 您的執行個體必須符合 LRS VPSX/MFI 的下列硬體和軟體需求：	雲端架構師

任務	Description	所需的技能
	• CPU – 雙核心 • RAM – 16 GB • 磁碟機 – 500 GB • 最小 EC2 執行個體 – m5.xlarge • 作業系統 – Windows/Linux • 軟體 – 網路資訊服務 (IIS) 或 Apache **Note** 上述硬體和軟體需求適用於小型印表機機群（約 500–1000)。若要取得完整需求，請洽詢您的 LRS 和 AWS 聯絡人。 當您建立 Windows 執行個體時，請執行下列動作： 1. 確認 EC2 主機名稱與用於 LRS 產品授權的主機名稱相同。 2. 完成下列操作，在 Amazon EC2 中啟用 CGI： a. 依照 的指示連接到 EC2 執行個體	

任務	Description	所需的技能
在 EC2 執行個體上安裝 LRS VPSX/MFI。	1. 依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線至您的 EC2 執行個體。 Amazon EC2 2. **Note** 從您應該收到的 LRS 電子郵件開啟產品下載頁面的連結。：LRS 產品透過電子檔案傳輸 (EFT) 分發。 3. 下載 LRS VPSX/MFI 並解壓縮 檔案 （預設資料夾：c:\LRS)。 4. 從解壓縮資料夾啟動 LRS 產品安裝程式，以安裝 LRS VPSX/MFI。 5. 在選取功能選單中，選取 VPSX® 伺服器 (V1R3.022)，然後選擇下一步以開始安裝程序。安裝完成時，您會收到成功訊息。	雲端架構師

任務	Description	所需的技能
安裝 LRS/佇列。	1. 依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線至您的 Rocket Enterprise Server EC2 執行個體。 Amazon EC2 2. 從您應該收到的 LRS 電子郵件開啟 LRS 產品下載頁面的連結，下載 LRS/佇列，然後解壓縮檔案。 3. 前往您下載檔案的位置，然後啟動 LRS 產品安裝程式來安裝 LRS/佇列。	雲端架構師
安裝 LRS/DIS。	1. 依照 Amazon EC2 文件中的步驟 2：連線至執行個體中的指示，連線至您的 LRS VPSX/MFI EC2 執行個體。 Amazon EC2 2. 從您應該收到的 LRS 電子郵件開啟 LRS 產品下載頁面的連結，下載 LRS/DIS，然後解壓縮檔案。 3. 前往您下載檔案的位置，然後啟動 LRS 產品安裝程式。 4. 在 LRS 產品安裝程式中，展開 LRS 其他工具，選取 LRS DIS，然後選擇下一步。 5. 請遵循 LRS 產品安裝程式中的其餘說明來完成安裝程序。	雲端架構師

任務	Description	所需的技能
建立目標群組，並將 LRS VPSX/MFI EC2 註冊為目標。	遵循 Elastic Load Balancing 文件中為 Network Load Balancer 建立目標群組的指示來建立目標群組。 當您建立目標群組時，請執行下列動作： 1. 在指定群組詳細資訊頁面上，針對選擇目標類型，選擇執行個體。 2. 針對通訊協定，選擇 TCP。 3. 針對連接埠，選擇 5500。 4. 在註冊目標頁面的可用執行個體區段中，選取 LRS VPSX/MFI EC2 執行個體。	雲端架構師
建立 Network Load Balancer。	遵循 Elastic Load Balancing Load Balancing 文件中建立 Network Load Balancer 的指示。Network Load Balancer 會將流量從 Rocket Enterprise Server 路由到 LRS VPSX/MFI EC2。 當您建立 Network Load Balancer 時，請在接聽程式和路由頁面上執行下列動作： 1. 針對 Protocol (通訊協定)，選擇 TCP。 2. 針對連接埠，選擇 5500。 3. 針對預設動作，針對您先前建立的目標群組選擇轉送至。	雲端架構師

整合 Rocket Enterprise Server 與 LRS VPSX/MFI 和 LRS/Queue

任務	Description	所需的技能
設定 Rocket Enterprise Server for LRS/Queue 整合。	1. 依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線至您的 Rocket Enterprise Server EC2 執行個體。 Amazon EC2 2. 在 Windows 開始選單中，開啟 Rocket Enterprise Server 管理 UI。 3. 在選單列中，選擇 NATIVE。 4. 在導覽窗格中，選擇目錄伺服器，然後選擇 BANKDEMO。 5. 從左側導覽窗格中的一般，向下捲動至其他區段，以設定環境變數 (LRSQ_ADDRESS、LRSQ_PORT、LRSQ_COMMAND) 以指向 LRSQ。 6. 針對 LRSQ_ADDRESS，輸入您先前建立之 Network Load Balancer 的 IP 地址或 DNS 名稱。 7. 對於 LRSQ_PORT，輸入 VPSX LRSQ 接聽程式連接埠 (5500)。 8. 對於 LRSQ_COMMAND，輸入 LRSQ 可執行檔的路徑位置。	雲端架構師

任務	Description	所需的技能
	Note LRS 目前支援 DNS 名稱的字元限制上限為 50，但未來可能會有所變更。如果您的 DNS 名稱大於 50，則可以使用 Network Load Balancer 的 IP 地址做為替代方案。	

任務	Description	所需的技能
設定 Rocket Enterprise Server for LRS VPSX/MFI 整合。	1. 將`VPSX_MFI_R2` 資料夾從 LRS VPSX/MFI 安裝程式複製到位於 的 Rocket Enterprise Server 位置`C:\BANKDEMO\print` 。 2. 依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線至您的 Rocket Enterprise Server EC2 執行個體。 Amazon EC2 3. 在 Windows 開始選單中，開啟 Rocket Enterprise Server 管理 UI。 4. 在選單列中，選擇 NATIVE。 5. 在導覽窗格中，選擇目錄伺服器，然後選擇 BANKDEMO。 6. 在 BANKDEMO 下，選擇 JES。 7. 在 JES 程式路徑下，從`C:\BANKDEMO\print` 位置新增`DLL(VPSX_MFI_R2)` 路徑。	雲端架構師

在 Rocket Enterprise Server 和 LRS VPSX/MFI 中設定印表機和列印使用者

任務	Description	所需的技能
將 Rocket Software Print Exit 模組與 Rocket Enterprise	1. 依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線至您的	雲端架構師

任務	Description	所需的技能
Server 批次印表機伺服器執行程序建立關聯。	Rocket Enterprise Server EC2 執行個體。 Amazon EC2 2. 在 Windows 開始選單中，開啟 Rocket Enterprise Server 管理 UI。 3. 在選單列中，選擇 NATIVE。 4. 在導覽窗格中，選擇目錄伺服器，然後選擇 BANKDEMO。 5. 在 BANKDEMO 下，選擇 JES，然後向下捲動至印表機。 6. 在印表機中，將 Rocket Software Print Exit 模組 (LRSPRTE6 for Batch) 與 Rocket Enterprise Server 批次印表機伺服器執行程序 (SEP) 建立關聯。這允許列印輸出路由到 LRS VPSX/MFI。 7. 登入 Enterprise Server Administration UI。 如需組態的詳細資訊，請參閱 Micro Focus 文件中的使用結束。	

任務	Description	所需的技能
在 LRS VPSX/MFI 中新增印表機。	1. 依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線至您的 LRS VPSX/MFI EC2 執行個體。Amazon EC2 2. 從 Windows 開始功能表開啟 VPSX Web 介面。 3. 在導覽窗格中，選擇印表機。 4. 選擇新增，然後選擇新增印表機。 5. 在印表機組態頁面上，針對印表機名稱，輸入 Local。 6. 針對 VPSX ID，輸入 VPS1。 7. 針對 CommType，選取 TCPIP/LRSQ。 8. 針對主機/IP 地址，輸入您要新增之實體印表機的 IP 地址。 9. 在裝置中，輸入裝置的名稱。 10選擇 Windows 驅動程式或 Linux/Mac 驅動程式。 11選擇新增。	雲端架構師

任務	Description	所需的技能
在 LRS VPSX/MFI 中建立列印使用者。	1. 依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線至您的 LRS VPSX/MFI EC2 執行個體。Amazon EC2 2. 從 Windows 開始功能表開啟 VPSX Web 介面。 3. 在導覽窗格中，選擇安全性，然後選擇使用者。 4. 在使用者名稱欄中，選擇管理員，然後選擇複製。 5. 在使用者設定檔維護視窗中，針對使用者名稱輸入使用者名稱（例如 PrintUser）。 6. 針對描述，輸入簡短描述（例如測試列印的使用者)。 7. 選擇更新。這會建立列印使用者（例如 PrintUser)。 8. 在導覽窗格的使用者下，選擇您建立的新使用者。 9. 從命令功能表中，選擇安全性。 10. 在安全規則頁面上，選擇所有適用的印表機安全和任務安全選項，然後選擇儲存。 11. 若要將新的列印使用者新增至管理員群組，請前往導覽窗格，選擇安全性，然後選擇設定。	雲端架構師

任務	Description	所需的技能
	12.在安全組態視窗中，將新的列印使用者新增至管理員欄。	

設定列印身分驗證和授權

任務	Description	所需的技能
使用使用者和群組建立 AWS Managed Microsoft AD 網域。	1. 遵循 AWS Directory Service 文件中建立 AWS Managed Microsoft AD 目錄中的指示，在 AWS Managed Microsoft AD 上建立 AWS Directory Service。 2. 部署 EC2 執行個體 (Active Directory 管理員） 並安裝 Active Directory 工具來管理 AWS Managed Microsoft AD，方法是遵循 AWS Directory Service 文件中的步驟 3：部署 EC2 執行個體以管理 AWS Managed Microsoft AD。AWS Directory Service 3. **Note** 依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線至您的 EC2 執行個體。Amazon EC2 ：當您連線到 EC2 執行個體時，請在	雲端架構師

任務	Description	所需的技能
	Windows 安全視窗中輸入您的管理員登入資料（適用於您在步驟 1 中建立的目錄）。 4. 在 Windows 開始功能表的 Windows 管理工具下，選擇 Active Directory 使用者和電腦。 5. 遵循 AWS Directory 服務文件中建立使用者的步驟，在 Active Directory 網域中建立列印使用者。	
將 LRS VPSX/MFI EC2 加入 AWS Managed Microsoft AD 網域。	自動將 LRS VPSX/MFI EC2 加入您的 AWS Managed Microsoft AD 網域 (AWS 知識中心文件）或手動 (AWS Directory Service 文件）。	雲端架構師

任務	Description	所需的技能
設定 LRS/DIS 並與 AWS Managed Microsoft AD 整合。	1. 依照 Amazon EC2 文件中的步驟 2：連線至執行個體中的指示，連線至您的 LRS VPSX/MFI EC2 執行個體。Amazon EC2 2. 在 Windows 開始選單中，開啟 VPSX Web 介面。 3. 在導覽窗格中，選擇安全性，然後選擇設定。 4. 在安全組態頁面的安全參數區段中，針對安全類型，選取內部。 5. 在安全參數區段中輸入其餘選項的偏好設定。 6. 從 Microsoft Windows Start 功能表中開啟 LRS Output Management 資料夾，選擇伺服器啟動，然後選擇伺服器停止。 7. 使用您的 Active Directory 使用者名稱和密碼登入 LRS VPSX/MFI。	雲端架構師

測試列印工作流程

任務	Description	所需的技能
從 Rocket Software BankDemo 應用程式啟動批次列印請求。	1. 在 Rocket Enterprise Server EC2 執行個體中開啟 3270 終端機模擬器。 2. 執行下列命令來連線至 BankDemo 應用程式：	測試工程師

任務	Description	所需的技能
	`connect 127.0.0.1 :9278` 3. 在 BankDemo 命令列界面上，針對使用者 ID，輸入 B0001。針對密碼，輸入非空白金鑰。 4. 在請求列印的陳述式 (X) 選項中，輸入空白行上的 X。 5. 在依 傳送陳述式區段中，針對郵件輸入 Y，然後按 F10。	

任務	Description	所需的技能
檢查 LRS VPSX/MFI 中的列印輸出。	1. 依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線至您的 LRS VPSX/MFI EC2 執行個體。Amazon EC2 2. 在 Windows 開始選單中，開啟 VPSX Web 介面。 3. 在導覽窗格中，選擇印表機，然後選擇輸出佇列。 4. 在多工緩衝處理 ID 欄中，選擇印表機佇列中請求的多工緩衝處理 ID。 5. 在動作索引標籤的 COMMAND 欄中，選擇瀏覽。 您現在可以查看帳戶陳述式的列印輸出，其中包含帳戶編號、描述、日期、金額和餘額的資料欄。如需範例，請參閱此模式的 batch_print_output 附件。	測試工程師

相關資源

- LRS 輸出現代化 (LRS 文件)
- ANSI 和機器承載控制 (IBM 文件)
- 頻道命令文字 (IBM 文件)
- 使用 Micro Focus 在 AWS 上增強企業大型主機工作負載 (AWS 合作夥伴網路部落格)
- 使用 Amazon EC2 Auto Scaling 和 Systems Manager 建置 Micro Focus Enterprise Server PAC (AWS 規範指引文件)
- 進階函數呈現 (AFP) 資料串流 (IBM 文件)

- Line Conditioned Data Stream (LCDS) （組件文件）

其他資訊

考量

在現代化旅程中，您可以考慮大型主機批次程序及其產生的輸出的各種組態。大型主機平台是由每個使用大型主機平台的客戶和廠商自訂，其具有直接影響列印的特定要求。例如，您目前的平台可能會將 IBM Advanced Function Presentation (AFP) 或 Xerox Line Condition Data Stream (LCDS) 納入目前的工作流程。此外，大型主機運輸控制字元和頻道命令文字可能會影響列印頁面的外觀，而且可能需要特殊處理。作為現代化規劃程序的一部分，我們建議您評估並了解特定列印環境中的組態。

列印資料擷取

Rocket Software Print Exit 會傳遞必要資訊，讓 LRS VPSX/MFI 有效地處理多工緩衝處理檔案。資訊由相關控制區塊中傳遞的欄位組成，例如：

- JOBNAME
- OWNER (USERID)
- 目的地
- FORM
- FILENAME
- 寫入

LRS VPSX/MFI 支援下列大型主機批次機制，用於從 Rocket Enterprise Server 擷取資料。

- 使用標準 z/OS JCL SYSOUT DD/OUTPUT 陳述式進行批次 COBOL 列印/多工緩衝處理
- 使用標準 z/OS JCL CA-SPOOL SUBSYS DD 陳述式進行批次 COBOL 列印/多工緩衝處理
- IMS/COBOL 列印/多工緩衝處理使用 CBLTDLI 介面 （如需支援的方法和程式設計範例的完整清單，請參閱產品授權隨附的 LRS 文件。）

印表機機群運作狀態檢查

LRS VPSX/MFI (LRS LoadX) 可執行深入研究運作狀態檢查，包括裝置管理和操作最佳化。裝置管理可以偵測印表機裝置中的失敗，並將列印請求路由到運作狀態良好的印表機。如需印表機機群深入研究運作狀態檢查的詳細資訊，請參閱產品授權隨附的 LRS 文件。

列印身分驗證和授權

LRS/DIS 可讓 LRS 應用程式使用 Microsoft Active Directory 或 LDAP 伺服器來驗證使用者 IDs 和密碼。除了基本列印授權之外，LRS/DIS 也可以在下列使用案例中套用精細層級的列印安全控制：

- 管理誰可以瀏覽印表機任務。
- 管理其他使用者任務的瀏覽層級。
- 管理操作任務。例如，命令層級安全性，例如 hold/release、 purge、 modify、 copy 和 reroute。安全可由 User-ID 或 Group（類似 AD 群組或 LDAP 群組）設定。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

大型主機現代化：DevOps on AWS with Rocket Software Enterprise Suite

Kevin Yung，Amazon Web Services

總結

客戶挑戰

當硬體需要擴展以滿足數位創新需求時，在大型主機硬體上執行核心應用程式的組織通常會遇到一些挑戰。這些挑戰包括下列限制條件。

- 由於大型主機硬體元件的彈性不足，以及變更的高成本，大型主機開發和測試環境無法擴展。
- 大型主機開發正面臨技能不足，因為新開發人員不熟悉且對傳統大型主機開發工具不感興趣。容器、持續整合/持續交付 (CI/CD) 管道和現代測試架構等現代技術不適用於大型主機開發。

模式結果

為了解決這些挑戰，Amazon Web Services (AWS) 和 Rocket Software Micro Focus AWS Partner Network (APN) 合作夥伴合作建立此模式。解決方案旨在協助您達成下列結果。

- 改善開發人員生產力。開發人員可以在幾分鐘內獲得新的大型主機開發執行個體。
- 使用 AWS 雲端 建立具有幾乎無限容量的新大型主機測試環境。
- 快速佈建新的大型主機 CI/CD 基礎設施。您可以在一小時內使用 AWS CloudFormation 和 在 上 AWS 完成佈建 AWS Systems Manager。
- 原生使用 AWS DevOps 工具進行大型主機開發，包括 AWS CodeBuild、 AWS CodeCommit AWS CodePipeline AWS CodeDeploy、 和 Amazon Elastic Container Registry (Amazon ECR)。
- 將傳統瀑布開發轉換為大型主機專案中的敏捷開發。

技術摘要

在此模式中，目標堆疊包含下列元件。

邏輯元件	實作解決方案	Description
原始程式碼儲存庫	Rocket Software AccuRev Server、CodeCommit、Amazon ECR	原始程式碼管理 – 解決方案使用兩種類型的原始程式碼：

		• 大型主機原始程式碼，例如 COBOL 和 JCL。 • AWS 基礎設施範本和自動化指令碼 這兩種類型的原始程式碼都需要版本控制，但它們是在不同的 SCMs中進行管理。部署到大型主機或 Rocket Software Enterprise Servers 的原始程式碼在 Rocket Software Micro Focus AccuRev Server 中受管。 AWS 範本和自動化指令碼在 CodeCommit 中受管。Amazon ECR 用於 Docker 映像儲存庫。
企業開發人員執行個體	Amazon Elastic Compute Cloud (Amazon EC2)、Rocket Software Enterprise Developer for Eclipse	大型主機開發人員可以使用 Rocket Software Enterpris e Developer for Eclipse 在 Amazon EC2 中開發程式碼。這不需要依賴大型主機硬體來寫入和測試程式碼。
Rocket Software Enterprise Suite 授權管理	Rocket Software Enterprise Suite License Manager	對於集中式 Rocket Software Enterprise Suite 授權管理和管控，解決方案會使用 Rocket Software Enterprise Suite License Manager 來託管所需的授權。

CI/CD 管道	CodePipeline、CodeBuild、CodeDeploy、容器中的 Rocket Software Enterprise Developer、容器中的 Rocket Software Enterprise Test Server、Rocket Software Micro Focus Enterprise Server	大型主機開發團隊需要 CI/CD 管道來執行程式碼編譯、整合測試和迴歸測試。在 中 AWS，CodePipeline 和 CodeBuild 可以在原生容器中使用 Rocket Software Enterprise Developer 和 Enterprise Test Server。

先決條件和限制

先決條件

名稱	描述
py3270	py3270 是 x3270 的 Python 界面，這是 IBM 3270 終端機模擬器。它為 x3270 或 s3270 子程序提供 API。
x3270	x3270 是適用於 X Window System 和 Windows 的 IBM 3270 終端機模擬器。 這可供開發人員用於本機單元測試。
Robot-Framework-Mainframe-3270-Library	Mainframe3270 是以 py3270 專案為基礎的機器人架構程式庫。
Rocket 軟體 Verastream	Rocket Software Verastream 是一種整合平台，可讓您以測試行動應用程式、Web 應用程式和 SOA Web 服務的方式測試大型主機資產。
Rocket 軟體統一功能測試 (UFT) 安裝程式和授權	Rocket Software Unified Functional Testing 是一種軟體，可為軟體應用程式和環境提供功能和迴歸測試自動化。
Rocket Software Enterprise Server 安裝程式和授權	Enterprise Server 為大型主機應用程式提供執行期環境。

Rocket Software Enterprise Test Server 安裝程式和授權	Rocket Software Enterprise Test Server 是 IBM 大型主機應用程式測試環境。
適用於 Server 的 Rocket Software AccuRev 安裝程式和授權，以及適用於 Windows 和 Linux 作業系統的 Rocket Software Micro Focus AccuRev 安裝程式和授權	AccuRev 提供原始程式碼管理 (SCM)。AccuRev 系統旨在供開發一組檔案的人員團隊使用。
Rocket Software Enterprise Developer for Eclipse 安裝程式、修補程式和授權	企業開發人員為大型主機開發人員提供平台，以開發和維護核心大型主機線上和批次應用程式。

限制

- CodeBuild 不支援建置 Windows Docker 映像。此回報的問題需要 Windows Kernel/HCS 和 Docker 團隊支援。解決方法是使用 Systems Manager 建立 Docker 映像建置 Runbook。此模式使用 因應措施來建置適用於 Eclipse 的 Rocket Software Enterpise 開發人員和 Rocket Software Micro Focus Enterprise Test Server 容器映像。
- Windows 尚未支援 CodeBuild 的虛擬私有雲端 (VPC) 連線，因此模式不會使用 Rocket Software License Manager 在 OpenText Rocket Software Enterprise Developer 和 Rocket Software Enterprise Test Server 容器中管理授權。

產品版本

- Rocket Software Enterprise Developer 5.5 或更新版本
- Rocket Software Enterprise Test Server 5.5 或更新版本
- Rocket Software Enterprise Server 5.5 或更新版本
- Rocket Software AccuRev 7.x 或更新版本
- 適用於 Rocket Software Enterprise Developer 和 Enterprise Test Server 的 Windows Docker 基礎映像：microsoft/dotnet-framework-4.7.2-runtime
- AccuRev 用戶端的 Linux Docker 基礎映像：amazonlinux：2

Architecture

大型主機環境

在傳統大型主機開發中，開發人員需要使用大型主機硬體來開發和測試程式。他們面臨容量限制，例如限制開發/測試環境的每秒數百萬次指令 (MIPS)，而且必須依賴大型主機電腦上可用的工具。

在許多組織中，大型主機開發遵循瀑布開發方法，團隊依賴長週期來發佈變更。這些發行週期通常比數位產品開發更長。

下圖顯示共用大型主機硬體以進行開發的多個大型主機專案。在大型主機硬體中，擴展更多專案的開發和測試環境非常昂貴。

AWS 架構

此模式會將大型主機開發延伸至 AWS 雲端。首先，它會使用 AccuRev SCM 託管大型主機原始碼 AWS。然後，它讓 Enterprise Developer 和 Enterprise Test Server 可用於建置和測試大型主機程式碼 AWS。

下列各節說明模式的三個主要元件。

1。SCM

在 中 AWS， 模式使用 AccuRev 為大型主機原始程式碼建立一組 SCM 工作區和版本控制。其串流型架構可讓多個團隊進行平行大型主機開發。為了合併變更，AccuRev 使用提升概念。為了將該變更新增至其他工作區，AccuRev 會使用更新概念。

在專案層級，每個團隊都可以在 AccuRev 中建立一或多個串流，以追蹤專案層級的變更。這些稱為專案串流。這些專案串流繼承自相同的父串流。父串流用於合併來自不同專案串流的變更。

每個專案串流都可以將程式碼提升為 AccuRev，並設定提升後觸發以啟動 AWS CI/CD 管道。專案串流變更的成功建置可以提升為其父串流，以進行更多迴歸測試。

通常，父串流稱為系統整合串流。當從專案串流提升到系統整合串流時，提升後觸發會啟動另一個 CI/CD 管道來執行迴歸測試。

除了大型主機程式碼之外，此模式還包括 AWS CloudFormation 範本、Systems Manager Automation 文件和指令碼。遵循infrastructure-as-code最佳實務，它們在 CodeCommit 中受版本控制。

如果您需要將大型主機程式碼同步回大型主機環境以進行部署，Rocket Software 會提供 Enterprise Sync 解決方案，將程式碼從 AccuRev SCM 同步回大型主機 SCM。

2. 開發人員和測試環境

在大型組織中，擴展超過一百或甚至超過一千個大型主機開發人員具有挑戰性。為了解決此限制， 模式會使用 Amazon EC2 Windows 執行個體進行開發。在執行個體上，已安裝 Enterprise Developer for Eclipse 工具。開發人員可以在執行個體本機執行所有大型主機程式碼測試和偵錯。

AWS Systems Manager 狀態管理員和自動化文件用於自動化開發人員執行個體佈建。建立開發人員執行個體的平均時間為 15 分鐘內。已備妥下列軟體和組態：

- AccuRev Windows 用戶端，用於簽出並將原始程式碼遞交至 AccuRev
- Enterprise Developers for Eclipse 工具，用於在本機寫入、測試和偵錯大型主機程式碼
- 開放原始碼測試架構 Python 行為驅動開發 (BDD) 測試架構行為、py3270 和 x3270 模擬器，用於建立指令碼以測試應用程式
- 用於建置 Enterprise Test Server Docker 映像並在 Enterprise Test Server Docker 容器中測試應用程式的 Docker 開發人員工具

在開發週期中，開發人員會使用 EC2 執行個體在本機開發和測試大型主機程式碼。成功測試本機變更時，開發人員會將變更提升為 AccuRev 伺服器。

3. CI/CD 管道

在 模式中，CI/CD 管道用於整合測試和迴歸測試，然後再部署到生產環境。

如 SCM 一節所述，AccuRev 使用兩種類型的串流：專案串流和整合串流。每個串流都會與 CI/CD 管道連接。為了在 AccuRev 伺服器與 之間執行整合 AWS CodePipeline， 模式會使用 AccuRev 提升後指令碼來建立事件以啟動 CI/CD。

例如，當開發人員將變更提升為 AccuRev 中的專案串流時，會啟動提升後指令碼以在 AccuRev Server 中執行。然後，指令碼會將變更的中繼資料上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體，以建立 Amazon S3 事件。此事件將啟動 CodePipeline 設定的管道來執行。

相同的事件啟動機制用於整合串流及其相關聯的管道。

在 CI/CD 管道中，CodePipeline 使用 CodeBuild 搭配 AccuRev Linux 用戶端容器，從 AccuRev 串流查看最新的程式碼。然後，管道會啟動 CodeBuild，以使用企業開發人員 Windows 容器來編譯原始程式碼，並使用 CodeBuild 中的企業測試伺服器 Windows 容器來測試大型主機應用程式。

CI/CD 管道是使用 CloudFormation 範本建置，藍圖將用於新專案。透過使用 範本，專案在其中建立新 CI/CD 管道所需的時間不到一小時 AWS。

為了擴展您的大型主機測試功能 AWS， 模式建置了 Rocket Software DevOps 測試套件、Verastream 和 UFT 伺服器。透過使用現代 DevOps 工具，您可以 AWS 視需要在 上執行任意數量的測試。

下圖 AWS 顯示開啟 Rocket Software 的大型主機開發環境範例。

目標技術堆疊

本節詳細介紹 模式中每個元件的架構。

1。原始程式碼儲存庫 – AccuRev SCM

AccuRev SCM 設定為管理大型主機原始碼版本。為了實現高可用性，AccuRev 支援主要和複本模式。在主節點上執行維護時，運算子可能會容錯移轉至複本。

為了加速 CI/CD 管道的回應， 模式會使用 Amazon CloudWatch Events 來偵測原始程式碼變更，並啟動管道的啟動。

1. 管道設定為使用 Amazon S3 來源。
2. CloudWatch Events 規則設定為從來源 S3 儲存貯體擷取 S3 事件。
3. CloudWatch Events 規則會將目標設定為管道。
4. AccuRev SCM 設定為在提升完成後於本機執行提升後指令碼。
5. AccuRev SCM 會產生 XML 檔案，其中包含提升的中繼資料，而且指令碼會將 XML 檔案上傳至來源 S3 儲存貯體。
6. 上傳後，來源 S3 儲存貯體會傳送事件以符合 CloudWatch Events 規則，而 CloudWatch Events 規則會啟動管道以執行。

當管道執行時，它會啟動 CodeBuild 專案，以使用 AccuRev Linux 用戶端容器，從相關聯的 AccuRev 串流查看最新的大型主機程式碼。

下圖顯示 AccuRev Server 設定。

2. 企業開發人員範本

模式使用 Amazon EC2 範本來簡化開發人員執行個體的建立。透過使用 State Manager，它可以一致地將軟體和授權設定套用至 EC2 執行個體。

Amazon EC2 範本會在其 VPC 內容設定和預設執行個體設定中建置，並遵循企業標記要求。透過使用範本，團隊可以建立自己的新開發執行個體。

當開發人員執行個體啟動時，透過與標籤建立關聯，Systems Manager 會使用 State Manager 套用自動化。自動化包含下列一般步驟。

1. 安裝 Enterprise Developer 軟體並安裝修補程式。
2. 安裝適用於 Windows 的 AccuRev 用戶端。
3. 安裝預先設定的指令碼，讓開發人員加入 AccuRev 串流。初始化 Eclipse 工作區。
4. 安裝開發工具，包括 x3270、py3270 和 Docker。
5. 設定授權設定以指向 License Manager 負載平衡器。

下圖顯示 Amazon EC2 範本建立的企業開發人員執行個體，其軟體和組態由 State Manager 套用至執行個體。企業開發人員執行個體會連線至 AWS License Manager 以啟用其授權。

3. CI/CD 管道

如 AWS 架構一節所述，在 模式中，有專案層級的 CI/CD 管道和系統整合管道。每個大型主機專案團隊都會建立管道或多個 CI/CD 管道，以建置他們在專案中開發的程式。這些專案 CI/CD 管道會從相關聯的 AccuRev 串流簽出原始程式碼。

在專案團隊中，開發人員會在相關聯的 AccuRev 串流中提升程式碼。然後，提升會啟動專案管道來建置程式碼並執行整合測試。

每個專案 CI/CD 管道使用 CodeBuild 專案搭配 Enterprise Developer 工具 Amazon ECR 映像和 Enterprise Test Server 工具 Amazon ECR 映像。

CodePipeline 和 CodeBuild 用於建立 CI/CD 管道。由於 CodeBuild 和 CodePipeline 沒有預付費用或承諾，您只需為使用量付費。相較於大型主機硬體，此 AWS 解決方案可大幅縮短硬體佈建前置時間，並降低測試環境的成本。

在現代開發中，會使用多個測試方法。例如，測試驅動的開發 (TDD)、BDD 和機器人架構。透過此模式，開發人員可以使用這些現代工具進行大型主機測試。例如，透過使用 x3270、py3270 和 Behave python 測試工具，您可以定義線上應用程式的行為。您也可以在這些 CI/CD 管道中使用建置大型主機 3270 機器人架構。

下圖顯示團隊串流 CI/CD 管道。

下圖顯示 Mainframe3270 機器人架構中 CodePipeline 產生的專案 CI/CD 測試報告。

下圖顯示 CodePipeline 在 Py3270 和 Behave BDD 中產生的專案 CI/CD 測試報告。

成功傳遞專案層級測試後，測試的程式碼會在 AccuRev SCM 中手動提升為整合串流。您可以在團隊對其專案管道的測試涵蓋範圍有信心之後，自動執行此步驟。

提升程式碼時，系統整合 CI/CD 管道會檢查合併的程式碼並執行迴歸測試。合併的程式碼會從所有平行專案串流提升。

根據測試環境需要的精細程度，客戶可以在不同的環境中擁有更多系統整合 CI/CD 管道，例如 UAT、Pre-Production。

在 模式中，系統整合管道中使用的工具是 Enterprise Test Server、UTF Server 和 Verastream。所有這些工具都可以部署到 Docker 容器中，並與 CodeBuild 搭配使用。

成功測試大型主機程式後，成品會以版本控制存放在 S3 儲存貯體中。

下圖顯示系統整合 CI/CD 管道。

在系統整合 CI/CD 管道中成功測試成品之後，即可提升成品以進行生產部署。

如果您需要將原始碼部署回大型主機，Rocket Software 會提供 Enterprise Sync 解決方案，將原始碼從 AccuRev 同步回大型主機 Endeavour。

下圖顯示將成品部署到企業伺服器的生產 CI/CD 管道。在此範例中，CodeDeploy 會協調將測試過的大型主機成品部署至 Enterprise Server。

除了 CI/CD 管道的架構演練之外，請參閱 AWS DevOps 部落格文章AWS 使用 Micro Focus Enterprise Suite 在 上自動化數千個大型主機測試，以取得在 CodeBuild 和 CodePipeline 中測試大型主機應用程式的詳細資訊。(Micro Focus 現在是 Rocket Software。) 如需執行大型主機測試的最佳實務和詳細資訊，請參閱部落格文章 AWS。

工具

AWS 自動化工具

- CloudFormation
- Amazon CloudWatch Events
- AWS CodeBuild
- AWS CodeDeploy
- AWS CodePipeline
- Amazon ECR
- Amazon S3
- AWS Secrets Manager
- AWS Systems Manager

Rocket 軟體工具

- 適用於 Eclipse 的 Rocket Enterprise 開發人員
- Rocket Enterprise 測試伺服器
- Rocket Enterprise Server（生產部署）
- Rocket 軟體 AccuRev
- Rocket Software Enterprise Suite License Manager
- Rocket Software Verastream 主機整合器
- Rocket 軟體 UFT One

其他工具

- x3270
- py3270
- Robot-Framework-Mainframe-3270-Library

史詩

建立 AccuRev SCM 基礎設施

任務	Description	所需的技能
使用 CloudFormation 部署主要 AccuRev SCM 伺服器。		AWS CloudFormation
建立 AccuRev 管理員使用者。	登入 AccuRev SCM Server，然後執行 CLI 命令來建立管理員使用者。	AccuRev SCM 伺服器管理員
建立 AccuRev 串流。	依序建立繼承自上方串流的 AccuRev 串流：生產、系統整合、團隊串流。	AccuRev SCM 管理員
建立開發人員 AccuRev 登入帳戶。	使用 AccuRev SCM CLI 命令為大型主機開發人員建立 AccuRev 使用者登入帳戶。	AccuRev SCM 管理員

建立企業開發人員 Amazon EC2 啟動範本

任務	Description	所需的技能
使用 CloudFormation 部署 Amazon EC2 啟動範本。	使用 CloudFormation 部署企業開發人員執行個體的 Amazon EC2 啟動範本。範本包含 Rocket Enterprise Developer 執行個體的 Systems Manager 自動化文件。	AWS CloudFormation
從 Amazon EC2 範本建立企業開發人員執行個體。		AWS 主控台登入和大型主機開發人員技能

建立企業開發人員工具 Docker 映像

任務	Description	所需的技能
建立企業開發人員工具 Docker 映像。	使用 Docker 命令和企業開發人員工具 Dockerfile 來建立 Docker 映像。	Docker
在 Amazon ECR 中建立 Docker 儲存庫。	在 Amazon ECR 主控台上，為企業開發人員 Docker 映像建立儲存庫。	Amazon ECR
將企業開發人員工具 Docker 映像推送至 Amazon ECR。	執行 Docker 推送命令，以推送企業開發人員工具 Docker 映像檔，將其儲存在 Amazon ECR 的 Docker 儲存庫中。	Docker

建立 Enterprise Test Server Docker 映像

任務	Description	所需的技能
建立 Enterprise Test Server Docker 映像。	使用 Docker 命令和 Enterprise Test Server Dockerfile 來建立 Docker 映像。	Docker
在 Amazon ECR 中建立 Docker 儲存庫。	在 Amazon ECR 主控台上，為 Enterprise Test Server Docker 映像建立 Amazon ECR 儲存庫。	Amazon ECR
將 Enterprise Test Server Docker 映像推送至 Amazon ECR。	執行 Docker push 命令，將 Enterprise Test Server Docker 映像推送並儲存在 Amazon ECR 中。	Docker

建立團隊串流 CI/CD 管道

任務	Description	所需的技能
建立 CodeCommit 儲存庫。	在 CodeCommit 主控台上，為基礎設施和 CloudFormation 程式碼建立 Git 型儲存庫。	AWS CodeCommit
將 CloudFormation 範本和自動化程式碼上傳至 CodeCommit 儲存庫。	執行 Git push 命令，將 CloudFormation 範本和自動化程式碼上傳至儲存庫。	Git
使用 CloudFormation 部署團隊串流 CI/CD 管道。	使用準備好的 CloudFormation 範本來部署團隊串流 CI/CD 管道。	AWS CloudFormation

建立系統整合 CI/CD 管道

任務	Description	所需的技能
建立 UFT Docker 映像。	使用 Docker 命令和 UFT Dockerfile 來建立 Docker 映像。	Docker
在 UFT 映像的 Amazon ECR 中建立 Docker 儲存庫。	在 Amazon ECR 主控台上，為 UFT 映像建立 Docker 儲存庫。	Amazon ECR
將 UFT Docker 映像推送至 Amazon ECR。	執行 Docker push 命令，將 Enterprise Test Server Docker 映像推送並儲存在 Amazon ECR 中。	Docker
建立 Verastream Docker 映像。	使用 Docker 命令和 Verastream Dockerfile 來建立 Docker 映像。	Docker

任務	Description	所需的技能
在 Amazon ECR 中為 Verastream 映像建立 Docker 儲存庫。	在 Amazon ECR 主控台上，為 Verastream 映像建立 Docker 儲存庫。	Amazon ECR
使用 CloudFormation 部署系統整合 CI/CD 管道。	使用準備好的 CloudFormation 範本來部署系統整合 CI/CD 管道。	AWS CloudFormation

建立生產部署 CI/CD 管道

任務	Description	所需的技能
使用 AWS Quick Start 部署 Enterprise Server。	若要使用 CloudFormation 部署 Enterprise Server，請在 AWS Quick Start 上啟動 Enterprise Server。	AWS CloudFormation
部署生產部署 CI/CD 管道。	在 CloudFormation 主控台上，使用 CloudFormation 範本來部署生產部署 CI/CD 管道。	AWS CloudFormation

相關資源

參考

- AWS DevOps 部落格 - AWS 使用 Micro Focus Enterprise Suite 在 上自動化數千個大型主機測試 (Micro Focus 現在是 Rocket Software。)
- py3270/py3270 GitHub 儲存庫
- Altran-PT-GDC/Robot-Framework-Mainframe-3270-Library GitHub 儲存庫
- 歡迎行為！
- APN 合作夥伴部落格 - 標籤：Micro Focus (Micro Focus 現在是 Rocket 軟體。)
- 從啟動範本啟動執行個體

AWS Marketplace

- Rocket 軟體 UFT One

AWS Quick Start

- 上的 Rocket Enterprise Server AWS

使用 Micro Focus Enterprise Server 和 LRS VPSX/MFI 將 AWS 上的大型主機線上列印工作負載現代化

Shubham Roy 和 Kevin Yung，Amazon Web Services

Abraham Rondon，微型焦點

Guy Tucker、Levi、Ray 和 Shoup Inc

總結

此模式說明如何使用 Micro Focus Enterprise Server 做為現代化大型主機應用程式的執行期，以及使用 LRS VPSX/MFI (Micro Focus Interface) 做為列印伺服器，在 Amazon Web Services (AWS) 雲端上現代化業務關鍵型大型主機線上列印工作負載。模式是以轉換大型主機現代化方法為基礎。在此方法中，您將大型主機線上應用程式遷移至 Amazon Elastic Compute Cloud (Amazon EC2)，並將大型主機資料庫遷移至 Amazon Relational Database Service (Amazon RDS)，例如 z/OS 的 IBM DB2。現代化列印工作流程的身分驗證和授權是由 AWS Directory Service for Microsoft Active Directory 執行，也稱為 AWS Managed Microsoft AD。LRS Directory Information Server (LRS/DIS) 已與 AWS Managed Microsoft AD 整合，用於列印工作流程身分驗證和授權。透過現代化線上列印工作負載，您可以降低 IT 基礎設施成本、減輕維護舊版系統的技術責任、移除資料孤島、使用 DevOps 模型提高敏捷性和效率，以及利用 AWS 雲端中的隨需資源和自動化。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 大型主機線上列印或輸出管理工作負載
- 如何重建和交付在 Micro Focus Enterprise Server 上執行之大型主機應用程式的基本知識（如需詳細資訊，請參閱 Micro Focus 文件中的企業伺服器資料表。）
- LRS 雲端列印解決方案和概念的基本知識（如需詳細資訊，請參閱 LRS 文件中的輸出現代化。）
- Micro Focus Enterprise Server 軟體和授權（如需詳細資訊，請聯絡 Micro Focus 銷售。）
- LRS VPSX/MFI、LRS/Queue 和 LRS/DIS 軟體和授權（如需詳細資訊，請聯絡 LRS 銷售)。

> **Note**
>
> 如需大型主機線上列印工作負載組態考量的詳細資訊，請參閱此模式額外資訊區段中的考量事項。

產品版本

- Micro Focus Enterprise Server 8.0 或更新版本
- LRS VPSX/MFI V1R3 或更新版本

Architecture

來源技術堆疊

- 作業系統 – IBM z/OS
- 程式設計語言 – 常見商業導向語言 (COBOL) 和客戶資訊控制系統 (CICS)
- 資料庫 – IBM DB2 for z/OS IBM 資訊管理系統 (IMS) 和虛擬儲存存取方法 (VSAM)
- 安全性 – 資源存取控制設施 (RACF)、z/OS 的 CA 最高機密，以及存取控制設施 2 (ACF2)
- 列印和輸出管理 – IBM 大型主機 z/OS 列印產品 （適用於 z/OS、LRS 和 CA 檢視的 IBM Infoprint 伺服器 ）

目標技術堆疊

- 作業系統 – 在Amazon EC2 上執行的 Microsoft Windows Server
- 運算 – Amazon EC2
- 程式設計語言 – COBOL 和 CICS
- 資料庫 – Amazon RDS
- 安全性 – AWS Managed Microsoft AD
- 列印和輸出管理 – AWS 上的 LRS 列印解決方案
- 大型主機執行期環境 – Micro Focus Enterprise Server

來源架構

下圖顯示大型主機線上列印工作負載的典型目前狀態架構。

該圖顯示以下工作流程：

1. 使用者在以 COBOL 撰寫的 IBM CICS 應用程式上建置的參與系統 (SoE) 上執行商業交易。
2. SoE 會叫用大型主機服務，該服務會將商業交易資料記錄在system-of-records(SoR) 資料庫中，例如 z/OS 的 IBM DB2。
3. SoR 會保留來自 SoE 的業務資料。
4. 使用者啟動從 CICS SoE 產生列印輸出的請求，這會啟動列印交易應用程式來處理列印請求。
5. 列印交易應用程式（例如 CICS 和 COBOL 程式）會從資料庫擷取資料、根據業務需求格式化資料，以及產生業務輸出（列印資料），例如帳單、ID 卡或貸款陳述式。然後，應用程式會使用虛擬電信存取方法 (VTAM) 傳送列印請求。z/OS 列印伺服器（例如 IBM Infoprint Server) 使用 NetSpool 或類似的 VTAM 元件來攔截列印請求，然後使用 JES 輸出參數在 JES 多工緩衝系統上建立列印輸出資料集。JES 輸出參數會指定路由資訊，供列印伺服器用來將輸出傳輸至特定網路印表機。VTAM 一詞是指 z/OS Communications Server 和 z/OS 的系統網路架構 (SNA) 服務元素。
6. 列印輸出傳輸元件會將輸出列印資料集從 JES 多工緩衝區傳輸至遠端印表機或列印伺服器，例如 LRS（如此模式所示）、IBM Infoprint Server 或電子郵件目的地。

目標架構

下圖顯示部署在 AWS 雲端中大型主機線上列印工作負載的架構：

該圖顯示以下工作流程：

1. 使用者從線上 (CICS) 使用者介面啟動列印請求，以建立列印輸出，例如帳單、ID 卡或貸款陳述式。
2. 大型主機線上應用程式 (已修改為 Amazon EC2) 使用 Micro Focus Enterprise Server 執行期從應用程式資料庫擷取資料、將商業邏輯套用至資料、格式化資料，然後使用 Micro Focus CICS Print Exit (DFHUPRNT) 將資料傳送至列印目的地。
3. 應用程式資料庫（在 Amazon RDS 上執行的 SoR) 會保留列印輸出的資料。
4. LRS VPSX/MFI 列印解決方案部署在 Amazon EC2 上，其操作資料存放在 Amazon Elastic Block Store (Amazon EBS) 中。LRS VPSX/MFI 使用 TCP/IP 型 LRS/Queue 傳輸代理程式，透過 Micro Focus CICS Print Exit API (DFHUPRNT) 收集列印資料，並將資料交付至指定的印表機目的地。現代化 CICS 應用程式中使用的原始 TERMID (TERM) 會用作 VPSX/MFI 佇列名稱。

> **Note**
>
> 目標解決方案通常不需要應用程式變更來容納大型主機格式語言，例如 IBM Advanced Function Presentation (AFP) 或 Xerox Line Condition Data Stream (LCDS)。如需在 AWS 上使用 Micro Focus 進行大型主機應用程式遷移和現代化的詳細資訊，請參閱 AWS 文件中的使用 Micro Focus 在 AWS 上增強企業大型主機工作負載。

AWS 基礎設施架構

下圖顯示適用於大型主機線上列印工作負載的高可用性和安全 AWS 基礎設施架構：

該圖顯示以下工作流程：

1. 大型主機線上應用程式（以程式設計語言撰寫，例如 CICS 或 COBOL) 使用核心商業邏輯來處理和產生列印輸出，例如帳單、ID 卡和貸款陳述式。線上應用程式部署在跨兩個可用區域 (AZ) 的 Amazon EC2 上以獲得高可用性 (HA)，並使用 Micro Focus CICS 列印結束將列印輸出路由到 LRS VPSX/MFI 進行最終使用者列印。
2. LRS VPSX/MFI 使用 TCP/IP 型 LRS/佇列傳輸代理程式，從 Micro Focus 線上列印結束程式設計界面收集或擷取列印資料。線上列印結束會傳遞必要資訊，讓 LRS VPSX/MFI 有效地處理列印檔案，並動態建置 LRS/佇列命令。

   > **Note**
   >
   > 如需各種用於列印的 CICS 應用程式程式設計方法，以及如何在 Micro Focus Enterprise 伺服器和 LRS VPSX/MFI 中支援它們的詳細資訊，請參閱此模式的其他資訊區段中的列印資料擷取。

3. > **Note**
   >
   > Network Load Balancer 提供 DNS 名稱，以整合 Micro Focus Enterprise Server 與 LRS VPSX/MFI。：LRS VPSX/MFI 支援第 4 層負載平衡器。Network Load Balancer 也會對 LRS VPSX/MFI 執行基本運作狀態檢查，並將流量路由至運作狀態良好的已註冊目標。

4. LRS VPSX/MFI 列印伺服器會跨 HA 的兩個可用區域部署在 Amazon EC2 上，並使用 Amazon EBS 做為操作資料存放區。LRS VPSX/MFI 支援主動-主動和主動-被動服務模式。此架構使用主動-被動配對中的多個可用區域作為主動和熱待命。Network Load Balancer 會對 LRS VPSX/MFI EC2 執行

個體執行運作狀態檢查，並在作用中執行個體處於運作狀態不佳時，將流量路由到另一個可用區域中的熱待命執行個體。列印請求會保留在每個 EC2 執行個體的本機 LRS 任務佇列中。在復原的情況下，必須重新啟動失敗的執行個體，LRS 服務才能繼續處理列印請求。

> **Note**
>
> LRS VPSX/MFI 也可以在印表機機群層級執行運作狀態檢查。如需詳細資訊，請參閱此模式額外資訊區段中的印表機機群運作狀態檢查。

5. AWS Managed Microsoft AD 與 LRS/DIS 整合，以執行列印工作流程身分驗證和授權。如需詳細資訊，請參閱此模式額外資訊區段中的列印身分驗證和授權。
6. LRS VPSX/MFI 使用 Amazon EBS 進行區塊儲存。您可以將 Amazon EBS 資料以point-in-time快照的形式從作用中的 EC2 執行個體備份到 Amazon S3，並將其還原至熱待命 EBS 磁碟區。若要自動建立、保留和刪除 Amazon EBS 磁碟區快照，您可以使用 Amazon Data Lifecycle Manager 來設定自動快照的頻率，並根據 RTO/RPO 需求還原快照。

工具

AWS 服務

- Amazon Elastic Block Store (Amazon EBS) 提供區塊層級儲存體磁碟區，可搭配使用 Amazon EC2 執行個體。EBS 磁碟區的行為與未格式化的原始區塊型儲存設備相似。您可以將這些磁碟區做為裝置，掛載在您的執行個體上。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。
- AWS Directory Service for Microsoft Active Directory (AD)，也稱為 AWS Managed Microsoft Active Directory，可讓您的目錄感知工作負載和 AWS 資源在 AWS 中使用受管 Active Directory。

其他工具

- LRS VPSX/MFI (Micro Focus Interface)，由 LRS 和 Micro Focus 共同開發，可從 Micro Focus Enterprise Server JES 多工緩衝區擷取輸出，並可靠地將其交付至指定的列印目的地。
- 在列印工作流程期間，LRS Directory Information Server (LRS/DIS) 用於身分驗證和授權。

- LRS/Queue 是一種以 TCP/IP 為基礎的 LRS/Queue 傳輸代理程式，由 LRS VPSX/MFI 使用，透過 Micro Focus 線上列印結束程式設計界面收集或擷取列印資料。
- Micro Focus Enterprise Server 是大型主機應用程式的應用程式部署環境。它為使用任何版本的 Micro Focus Enterprise Developer 遷移或建立的大型主機應用程式提供執行環境。

史詩

在 Amazon EC2 上設定 Micro Focus Enterprise Server 並部署大型主機線上應用程式

任務	Description	所需的技能
設定 Micro Focus Enterprise Server 並部署示範線上應用程式。	在 Amazon EC2 上設定 Micro Focus Enterprise Server，然後遵循 Micro Focus 文件中的教學課程：CICS Support 指示，在 Amazon EC2 上部署 Micro Focus 帳戶示範應用程式 (ACCT 示範）。 ACCT 示範應用程式是大型主機線上 (CICS) 應用程式，可建立並啟動列印輸出。	雲端架構師

在 Amazon EC2 上設定 LRS 列印伺服器

任務	Description	所需的技能
取得用於列印的 LRS 產品授權。	若要取得 LRS VPSX/MFI、LRS/Queue 和 LRS/DIS 的 LRS 產品授權，請聯絡 LRS 輸出管理團隊。您必須提供將安裝 LRS 產品的 EC2 執行個體的主機名稱。	建置潛在客戶
建立 Amazon EC2 Windows 執行個體以安裝 LRS VPSX/MFI。	依照 Amazon EC2 文件中的步驟 1：啟動執行個體中的指示啟動 Amazon EC2 Windows	雲端架構師

任務	Description	所需的技能
	執行個體。 Amazon EC2 您的執行個體必須符合 LRS VPSX/MFI 的下列硬體和軟體需求： • CPU – 雙核心 • RAM – 16 GB • 磁碟機 – 500 GB • 最小 EC2 執行個體 – m5.xlarge • 作業系統 – Windows/Linux • 軟體 – 網路資訊服務 (IIS) 或 Apache **Note** 上述硬體和軟體需求適用於小型印表機機群（約 500–1000)。若要取得完整需求，請洽詢您的 LRS 和 AWS 聯絡人。 當您建立 Windows 執行個體時，請執行下列動作： 1. 確認 EC2 主機名稱與用於 LRS 產品授權的主機名稱相同。 2. 完成下列操作，在 Amazon EC2 中啟用 CGI： a. 請依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線	

任務	Description	所需的技能
	至您的 EC2 執行個體。Amazon EC2 b. 在 Windows 開始選單中，尋找並開啟 Server Manager。 c. 在伺服器管理員中，選擇儀表板、Quick Start、新增角色和功能。然後，選擇伺服器角色。 d. 在伺服器角色中，選擇 WebServer (IIS)，然後選擇應用程式開發。 e. 在應用程式開發中，選取 CGI 核取方塊。 f. 遵循 Windows Server Manager 新增角色和功能精靈上的指示來安裝 CGI。 g. 在 EC2 執行個體的 Windows 防火牆中開啟連接埠 5500，以進行 LRS/佇列通訊。	

任務	Description	所需的技能
在 EC2 執行個體上安裝 LRS VPSX/MFI。	1. 依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線至您的 EC2 執行個體。 Amazon EC2 2. **Note** 從您應該收到的 LRS 電子郵件開啟產品下載頁面的連結。：LRS 產品透過電子檔案傳輸 (EFT) 分佈。 3. 下載 LRS VPSX/MFI 並解壓縮 檔案 （預設資料夾：c:\LRS)。 4. 從解壓縮資料夾啟動 LRS 產品安裝程式，以安裝 LRS VPSX/MFI。 5. 在選取功能選單中，選取 VPSX® 伺服器 (V1R3.022)，然後選擇下一步以開始安裝程序。安裝完成時，您會收到成功訊息。	雲端架構師

任務	Description	所需的技能
安裝 LRS/佇列。	1. 依照 Amazon EC2 文件中的步驟 2：連線至執行個體中的指示，連線至 Micro Focus Enterprise Server EC2 執行個體。 Amazon EC2 2. 從您應該收到的 LRS 電子郵件開啟 LRS 產品下載頁面的連結，下載 LRS/佇列，然後解壓縮檔案。 3. 前往您下載檔案的位置，然後啟動 LRS 產品安裝程式來安裝 LRS/佇列。	雲端架構師
安裝 LRS/DIS。	1. 請依照 Amazon EC2 文件中步驟 2：連線至執行個體中的指示，連線至 LRS VPSX/MFI EC2 執行個體。 Amazon EC2 2. 從您應該收到的 LRS 電子郵件開啟 LRS 產品下載頁面的連結，下載 LRS/DIS，然後解壓縮檔案。 3. 前往您下載檔案的位置，然後啟動 LRS 產品安裝程式。 4. 在 LRS 產品安裝程式中，展開 LRS 其他工具，選取 LRS DIS，然後選擇下一步。 5. 請遵循 LRS 產品安裝程式中的其餘說明來完成安裝程序。	雲端架構師

任務	Description	所需的技能
建立目標群組，並將 LRS VPSX/MFI EC2 註冊為目標。	遵循 Elastic Load Balancing 文件中為 Network Load Balancer 建立目標群組的指示來建立目標群組。 當您建立目標群組時，請執行下列動作： 1. 在指定群組詳細資訊頁面上，針對選擇目標類型，選擇執行個體。 2. 針對通訊協定，選擇 TCP。 3. 針對連接埠，選擇 5500。 4. 在註冊目標頁面的可用執行個體區段中，選取 LRS VPSX/MFI EC2 執行個體。	雲端架構師
建立 Network Load Balancer。	遵循 Elastic Load Balancing Load Balancing 文件中建立 Network Load Balancer 的指示。Network Load Balancer 會將流量從 Micro Focus Enterprise Server 路由到 LRS VPSX/MFI EC2。 當您建立 Network Load Balancer 時，請在接聽程式和路由頁面上執行下列動作： 1. 針對 Protocol (通訊協定)，選擇 TCP。 2. 針對連接埠，選擇 5500。 3. 針對預設動作，針對您先前建立的目標群組選擇轉送至。	雲端架構師

將 Micro Focus Enterprise Server 與 LRS VPSX/MFI 和 LRS/Queue 整合

任務	Description	所需的技能
設定 Micro Focus Enterprise Server for LRS/Queue 整合。	1. 依照 Amazon EC2 文件中的步驟 2：連線至執行個體中的指示，連線至 Micro Focus Enterprise Server EC2 執行個體。 Amazon EC2 2. 在 Windows 開始功能表中，開啟 Micro Focus Enterprise Server 管理 UI。 3. 在選單列中，選擇 NATIVE。 4. 在導覽窗格中，選擇目錄伺服器，然後選擇 BANKDEMO 或您的企業伺服器區域。 5. 從左側導覽窗格中的一般，向下捲動至其他區段，以設定環境變數 (LRSQ_ADDRESS、LRSQ_PORT、LRSQ_COMMAND) 以指向 LRSQ。 6. 針對 LRSQ_ADDRESS，輸入您先前建立之 Network Load Balancer 的 IP 地址或 DNS 名稱。 7. 針對 LRSQ_PORT，輸入 VPSX LRSQ 接聽程式連接埠 (5500)。	雲端架構師

任務	Description	所需的技能
	8. 對於 LRSQ_COMMAND，輸入 LRSQ 可執行檔的路徑位置。 9. **Note** LRS 目前支援 DNS 名稱的字元限制上限為 50，但未來可能會有所變更。如果您的 DNS 名稱大於 50，則可以使用 Network Load Balancer 的 IP 地址做為替代方案。	

任務	Description	所需的技能
讓 Micro Focus Enterprise Server 初始化可使用 CICS 列印結束 (DFHUPRNT)。	1. 依照 Amazon EC2 文件中的步驟 2：連線至執行個體中的指示，連線至 Micro Focus Enterprise Server EC2 執行個體。 Amazon EC2 2. **Note** 將 CICS 列印結束 (DFHUPRNT) 從 LRS VPSX/MFI 可執行檔資料夾（名為 VPSX_MFI_R2 ）複製到 Micro Focus Enterprise Server EC2 執行個體位置。對於 32 位元系統，位置為 C:\Program Files (x86)\Micro Focus\Enterprise Server\bin 。對於 64 位元系統，位置為 C:\Program Files (x86)\Micro Focus\Enterprise Server\bin64 。：DFHUPRNT_64.dll 複製DFHUPRNT.	雲端架構師

任務	Description	所需的技能
	dll 時，檔案必須重新命名為 。 驗證 Micro Focus Enterprise Server 已偵測到 CICS 列印結束 (DFHUPRNT) 1. 停止和啟動 Micro Focus Enterprise Server。 2. 在 Micro Focus Enterprise Server 的管理面板中，開啟監視器、日誌、主控台日誌。 3. 檢查主控台日誌是否有下列訊息：「3270 印表機使用者成功退出 DFHUPRNT。」	

任務	Description	所需的技能
將 CICS 印表機的終端機 ID (TERMIDs) 定義為 Micro Focus Enterprise Server。	在 Micro Focus Enterprise Server 中啟用 3270 列印 1. 在 Micro Focus Enterprise Server 的管理面板中，開啟 CICS、資源、依群組。 2. 從左側導覽面板中，選擇 SIT （系統初始化資料表），然後選擇 BNKCICV。 3. 在一般區段中，向下捲動至 3270，然後選取 3270 列印核取方塊。 在 Micro Focus Enterprise Server 中定義 CICS 印表機的終端機 1. 在 Micro Focus Enterprise Server 的管理面板中，開啟 CICS、資源、依類型。 2. 從左側導覽面板中，選擇 Term，然後選擇新增。建立終端機資源表單隨即開啟。 3. 針對名稱，輸入 LRS 列印佇列的名稱。(注意：此模式使用「P275」作為 CICS 印表機的終端機 ID 和 LRS VPSX 列印佇列。) 4. 針對群組，輸入 BANKTERM。 5. 對於自動安裝 – 模型，輸入 NO。	雲端架構師

任務	Description	所需的技能
	6. 針對終端機識別符 - 終端機類型，輸入 DFHPRT32。 7. 針對 Net name，輸入 VTAMP275。 8. 針對終端機用量，選取服務中核取方塊。 9. 在頁面頂端捲動，然後選擇儲存。 10選擇 Install (安裝)。快顯訊息會顯示成功的安裝訊息。	

在 Micro Focus Enterprise Server 和 LRS VPSX/MFI 中設定印表機和列印使用者

任務	Description	所需的技能
在 LRS VPSX 中建立列印佇列。	1. 遵循 Amazon EC2 文件中步驟 2：連線至執行個體中的指示，連線至 LRS VPSX/MFI EC2 執行個體。Amazon EC2 2. 從 Windows 開始功能表開啟 VPSX Web 介面。 3. 在導覽窗格中，選擇印表機。 4. 選擇新增，然後選擇新增印表機。 5. 在印表機組態頁面上，針對印表機名稱輸入 P275。 6. 針對 VPSX ID，輸入 VPS1。 7. 針對 CommType，選取 TCPIP/LRSQ。	雲端架構師

任務	Description	所需的技能
	8. 針對主機/IP 地址，輸入您要新增之實體印表機的 IP 地址。 9. 在裝置中，輸入裝置的名稱。 10選擇 Windows 驅動程式或 Linux/Mac 驅動程式。 11選擇新增。 **Note** 列印佇列必須等同於 Micro Focus Enterprise Server 中建立的列印 TERMIDs。	

任務	Description	所需的技能
在 LRS VPSX/MFI 中建立列印使用者。	1. 依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線至您的 LRS VPSX/MFI EC2 執行個體。Amazon EC2 2. 從 Windows 開始功能表開啟 VPSX Web 介面。 3. 在導覽窗格中，選擇安全性，然後選擇使用者。 4. 在使用者名稱欄中，選擇管理員，然後選擇複製。 5. 在使用者設定檔維護視窗中，針對使用者名稱輸入使用者名稱（例如 PrintUser）。 6. 針對描述，輸入簡短描述（例如測試列印的使用者)。 7. 選擇更新。這會建立列印使用者（例如 PrintUser)。 8. 在導覽窗格的使用者下，選擇您建立的新使用者。 9. 從命令功能表中，選擇安全性。 10. 在安全規則頁面上，選擇所有適用的印表機安全和任務安全選項，然後選擇儲存。 11. 若要將新的列印使用者新增至管理員群組，請前往導覽窗格，選擇安全性，然後選擇設定。	雲端架構師

任務	Description	所需的技能
	12.在安全組態視窗中，將新的列印使用者新增至管理員欄。	

設定列印身分驗證和授權

任務	Description	所需的技能
使用使用者和群組建立 AWS Managed Microsoft AD 網域。	1. 遵循 AWS Directory Service 文件中建立 AWS Managed Microsoft AD 目錄的指示，在 AWS Managed Microsoft AD 上建立 AWS Directory Service。 2. 部署 EC2 執行個體 (Active Directory 管理員） 並安裝 Active Directory 工具來管理 AWS Managed Microsoft AD，方法是依照 AWS Directory Service 文件中的步驟 3：部署 EC2 執行個體以管理 AWS Managed Microsoft AD。AWS Directory Service 3. **Note** 依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線至您的 EC2 執行個體。Amazon EC2 ：當您連線到 EC2 執行個體時，請在	雲端架構師

任務	Description	所需的技能
	Windows 安全視窗中輸入您的管理員登入資料（適用於您在步驟一中建立的目錄）。 4. 在 Windows 開始選單的 Windows 管理工具下，選擇 Active Directory 使用者和電腦。 5. 遵循 AWS Directory 服務文件中建立使用者的步驟，在 Active Directory 網域中建立列印使用者。	
將 LRS VPSX/MFI EC2 加入 AWS Managed Microsoft AD 網域。	以自動方式 (AWS 知識中心文件）或手動方式 (AWS Directory Service 文件）將 LRS VPSX/MFI EC2 加入您的 AWS Managed Microsoft AD 網域。	雲端架構師

任務	Description	所需的技能
設定 LRS/DIS 並與 AWS Managed Microsoft AD 整合。	1. 依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線至您的 LRS VPSX/MFI EC2 執行個體。Amazon EC2 2. 在 Windows 開始選單中，開啟 VPSX Web 介面。 3. 在導覽窗格中，選擇安全性，然後選擇設定。 4. 在安全組態頁面的安全參數區段中，針對安全類型，選取內部。 5. 在安全參數區段中輸入其餘選項的偏好設定。 6. 從 Microsoft Windows Start 功能表中開啟 LRS Output Management 資料夾，選擇伺服器啟動，然後選擇伺服器停止。 7. 使用您的 Active Directory 使用者名稱和密碼登入 LRS VPSX/MFI。	雲端架構師

測試線上列印工作流程

任務	Description	所需的技能
從 Micro Focus ACCT 示範應用程式啟動線上列印請求。	1. **Note** 在 Micro Focus Enterprise Server EC2 執行個體中開啟 TN3270 終端機	雲端架構師

任務	Description	所需的技能
	模擬器。(：此模式使用 3270 終端機模擬器。)	
	2. 連接至 TN3270 終端機模擬器 (Rumba)。對於主機名稱地址，請使用 127.0.0.1。對於 Telnet 連接埠，請使用 9270。 3. 連線至 3270 畫面後，按 CTL+SHIFT+Z 清除畫面。 4. 若要啟動 ACCT 示範應用程式，請在清除畫面中輸入 ACCT。這會開啟 ACCT 線上示範 (CICS) 應用程式主畫面。注意：主畫面包含選單選項，例如帳戶檔案、依名稱搜尋、輸入、請求類型、帳戶和印表機。 5. 若要從 ACCT 線上示範 (CICS) 應用程式提交列印請求，請在請求類型欄位中輸入 P、在帳戶欄位中輸入 111111，然後在印表機欄位中輸入 P275。請務必將印表機欄位中的值設定為 CICS 印表機終端機 ID 的值。 6. 按 Enter。 「已排程列印請求」訊息會顯示在畫面底部。這確認線上列印請求是從 ACCT 示範應用程	

任務	Description	所需的技能
	式產生，並傳送至 LRS VPS/MFI 進行列印處理。	
檢查 LRS VPSX/MFI 中的列印輸出。	1. 依照 Amazon EC2 文件中的步驟 2：連接至執行個體中的指示，連線至您的 LRS VPSX/MFI EC2 執行個體。Amazon EC2 2. 在 Windows 開始選單中，開啟 VPSX Web 介面。 3. 在導覽窗格中，選擇印表機，然後選擇輸出佇列。尋找您先前為線上列印建立的 P275 列印佇列。 4. 對於列印佇列 (P275)，在多工緩衝處理 ID 欄中，選擇印表機佇列中請求的多工緩衝處理 ID。 5. 在動作索引標籤的 COMMAND 欄中，選擇瀏覽。 您現在可以查看 帳戶陳述式的列印輸出，其中包含帳戶編號、SURNAME、FIRST、ADDRESS、TELEPHONE、發行的卡片數量、發行日期、金額和餘額的資料欄。 如需範例，請參閱此模式的 online_print_output 附件。	測試工程師

相關資源

- LRS 輸出現代化 (LRS 文件）
- VTAM 聯網概念 (IBM 文件）
- 邏輯單位 (LU) 類型摘要 (IBM 文件）
- ANSI 和機器承載控制 (IBM 文件）
- 使用 Micro Focus 在 AWS 上增強企業大型主機工作負載 (AWS 合作夥伴網路部落格）
- 使用 Amazon EC2 Auto Scaling 和 Systems Manager 建置 Micro Focus Enterprise Server PAC (AWS 規範性指導文件）
- 進階函數呈現 (AFP) 資料串流 (IBM 文件）
- Line Conditioned Data Stream (LCDS) （組件文件）

其他資訊

考量

在現代化旅程中，您可以考慮大型主機線上程序及其產生的輸出的各種組態。大型主機平台是由每個使用大型主機平台的客戶和廠商自訂，其具有直接影響列印的特定要求。例如，您目前的平台可能會將 IBM 進階函數簡報 (AFP) 或 Xerox Line Condition Data Stream (LCDS) 納入目前的工作流程。此外，大型主機運輸控制字元和頻道命令文字可能會影響列印頁面的外觀，並且可能需要特殊處理。作為現代化規劃程序的一部分，我們建議您評估並了解特定列印環境中的組態。

列印資料擷取

本節摘要說明您可以在 IBM 大型主機環境中用於列印的 CICS 應用程式程式設計方法。LRS VPSX/MFI 元件提供技術，允許相同的應用程式以相同的方式建立資料。下表說明如何在 AWS 和 Micro Focus Enterprise Server 中搭配 LRS VPSX/MFI 列印伺服器執行的現代化 CICS 應用程式中支援每個應用程式程式設計方法。

Method	Description	在現代化環境中支援 方法
EXEC CICS SEND TEXT.. 或 EXEC CICS SEND MAP.。	這些 CICS 和 VTAM 方法負責建立 3270/SCS 列印資料串流並將其交付至 LUTYPE0, LUTYPE1和 LUTYPE3 列印裝置。	Micro Focus 線上列印結束 (DFHUPRNT) 應用程式介面 (API) 可讓列印資料在建立 3270/SCS 列印資料串流時，由 VPSX/MFI 處理。

EXEC CICS SEND TEXT.. 或 EXEC CICS SEND MAP..（使用第三方 IBM 大型主機軟體）	CICS 和 VTAM 方法負責建立 3270/SCS 列印資料串流並將其交付至 LUTYPE0, LUTYPE1 和 LUTYPE3 列印裝置。第三方軟體產品會攔截列印資料、使用 ASA/MCH 控制字元將資料轉換為標準列印格式資料，並將資料放置在 JES 多工緩衝系統上，以供使用 JES 的大型主機列印系統處理。	Micro Focus 線上列印結束 (DFHUPRNT) API 可讓使用其中一種方法建立 3270/SCS 列印資料串流時，VPSX/MFI 處理列印資料。
EXEC CICS SPOOLOPEN	CICS 應用程式使用此方法直接將資料寫入 JES 多工緩衝處理。然後，資料可供使用 JES 的大型主機型列印系統處理。	Micro Focus Enterprise Server 會將資料多工緩衝處理至 Enterprise Server 多工緩衝處理，而 VPSX/MFI 批次列印結束 (LRSPRTE6) 可將資料多工緩衝處理至 VPSX。
DRS/API	LRS 提供的程式設計界面用於將列印資料寫入 JES。	VPSX/MFI 提供替換界面，可將列印資料直接多工緩衝處理至 VPSX。

印表機機群運作狀態檢查

LRS VPSX/MFI (LRS LoadX) 可執行深入研究運作狀態檢查，包括裝置管理和操作最佳化。裝置管理可以偵測印表機裝置中的失敗，並將列印請求路由到運作狀態良好的印表機。如需印表機機群深入探索運作狀態檢查的詳細資訊，請參閱產品授權隨附的 LRS 文件。

列印身分驗證和授權

LRS/DIS 可讓 LRS 應用程式使用 Microsoft Active Directory 或 LDAP 伺服器來驗證使用者 IDs 和密碼。除了基本列印授權之外，LRS/DIS 也可以在下列使用案例中套用精細層級的列印安全控制：

- 管理誰可以瀏覽印表機任務。
- 管理其他使用者任務的瀏覽層級。
- 管理操作任務。例如，命令層級安全性，例如 hold/release、 purge、 modify、 copy 和 reroute。安全可由 User-ID 或 Group（類似 AD 群組或 LDAP 群組）設定。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Transfer 系列將大型主機檔案直接移至 Amazon S3

Luis Gustavo Dantas，Amazon Web Services

總結

在現代化旅程中，您可能會面臨在內部部署伺服器與 Amazon Web Services (AWS) 雲端之間傳輸檔案的挑戰。從大型主機傳輸資料可能是一項重大挑戰，因為大型主機通常無法存取現代資料存放區，例如 Amazon Simple Storage Service (Amazon S3)、Amazon Elastic Block Store (Amazon EBS) 或 Amazon Elastic File System (Amazon EFS)。

許多客戶使用內部部署 Linux、Unix 或 Windows 伺服器等中繼預備資源，將檔案傳輸至 AWS 雲端。您可以避免這種間接方法，方法是使用 AWS Transfer Family 搭配 Secure Shell (SSH) 檔案傳輸通訊協定 (SFTP)，將大型主機檔案直接上傳至 Amazon S3。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 虛擬私有雲端 (VPC)，具有傳統平台可存取的子網路
- VPC 的 Transfer Family 端點
- 大型主機虛擬儲存存取方法 (VSAM) 檔案轉換為循序、固定長度的檔案 (IBM 文件）

限制

- 根據預設，SFTP 會以二進位模式傳輸檔案，這表示檔案會上傳到 Amazon S3 並保留 EBCDIC 編碼。如果您的檔案不包含二進位或封裝資料，則可以使用 sftpascii 子命令 (IBM 文件） 在傳輸期間將檔案轉換為文字。
- 您必須解壓縮包含封裝和二進位內容的大型主機檔案 (AWS 規範指引），才能在目標環境中使用這些檔案。
- Amazon S3 物件的大小範圍從最小 0 個位元組到最大 5 TB。如需 Amazon S3 功能的詳細資訊，請參閱 Amazon S3 FAQs。

Architecture

來源技術堆疊

- 工作控制語言 (JCL)
- z/OS Unix shell 和 ISPF
- SFTP
- VSAM 和一般檔案

目標技術堆疊

- Transfer 系列
- Amazon S3
- Amazon Virtual Private Cloud (Amazon VPC)

目標架構

下圖顯示使用 Transfer Family 搭配 SFTP 將大型主機檔案直接上傳至 S3 儲存貯體的參考架構。

該圖顯示以下工作流程：

1. 您可以使用 JCL 任務，透過 Direct Connect 將大型主機檔案從舊版大型主機傳輸到 AWS 雲端。
2. Direct Connect 可讓您的網路流量保留在 AWS 全球網路上，並略過公有網際網路。Direct Connect 也會增強網路速度，從 50 Mbps 開始，擴展到 100 Gbps。
3. VPC 端點可在不使用公有網際網路的情況下，啟用 VPC 資源與支援的 服務之間的連線。存取 Transfer Family 和 Amazon S3 可透過位於兩個私有子網路和可用區域的彈性網路界面實現高可用性。
4. Transfer Family 會驗證使用者，並使用 SFTP 從舊版環境接收您的檔案，並將其移至 S3 儲存貯體。

自動化和擴展

使用 Transfer Family 服務後，您可以使用 JCL 任務做為 SFTP 用戶端，將無限數量的檔案從大型主機傳輸到 Amazon S3。您也可以使用大型主機批次任務排程器，在準備好傳輸大型主機檔案時執行 SFTP 任務，以自動化檔案傳輸。

工具

- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路類似於您在自己的資料中心內操作的傳統網路，具有使用可擴展的 AWS 基礎設施的優勢。
- AWS Transfer Family 可讓您使用 SFTP、FTPS 和 FTP 通訊協定，安全地將週期business-to-business檔案傳輸至 Amazon S3 和 Amazon EFS。 Amazon S3

史詩

建立 S3 儲存貯體和存取政策

任務	Description	所需的技能
建立 S3 儲存貯體。	建立 S3 儲存貯體以託管您從舊版環境傳輸的檔案。	一般 AWS
建立 IAM 角色和政策。	Transfer Family 使用您的 AWS Identity and Access Management (IAM) 角色，授予您先前建立的 S3 儲存貯體存取權。 建立包含下列 IAM 政策的 IAM 角色： https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_create.html `{ "Version": "2012-10-17", "Statement": [ { "Sid": "UserFolderListing", "Action": [ "s3:ListBucket",`	一般 AWS

任務	Description	所需的技能

	"s3:GetBucketLocation"
	],
	"Effect": "Allow",
	"Resource": [
	"arn:aws:s3:::<your-bucket-name>"
	]
	},
	{
	"Sid": "HomeDirObjectAccess",
	"Effect": "Allow",
	"Action": [
	"s3:PutObject",
	"s3:GetObjectAcl",
	"s3:GetObject",
	"s3:DeleteObjectVersion",
	"s3:DeleteObject",
	"s3:PutObjectAcl",
	"s3:GetObjectVersion"
	],
	"Resource": "arn:aws:s3:::<your-bucket-name>/*"
	}
	]	

任務	Description	所需的技能
	} **Note** 建立 IAM 角色時，您必須選擇轉移使用案例。	

定義傳輸服務

任務	Description	所需的技能
建立 SFTP 伺服器。	1. 登入 AWS 管理主控台，開啟 Transfer Family 主控台，然後選擇建立伺服器。 2. 僅選擇 SFTP (SSH 檔案傳輸通訊協定） - 透過 Secure Shell 通訊協定進行檔案傳輸，然後選擇下一步。 3. 針對身分提供者，選擇服務受管，然後選擇下一步。 4. 針對端點類型，選擇託管的 VPC。 5. 針對存取，選擇內部。 6. 在 VPC 中，選擇您的 VPC。 7. 在可用區域區段中，選擇您的可用區域和子網路。 8. 在安全群組區段中，選擇您的安全群組，然後選擇下一步。	一般 AWS

任務	Description	所需的技能
	9. 針對網域，選擇 Amazon S3，然後選擇下一步。 10. 在設定其他詳細資訊頁面上保留預設選項，然後選擇下一步。 11. 選擇 Create server (建立伺服器)。 **Note** 如需如何設定 SFTP 伺服器的詳細資訊，請參閱建立啟用 SFTP 的伺服器 (AWS Transfer Family 使用者指南)。	
取得伺服器地址。	1. 開啟 Transfer Family 主控台，然後在伺服器 ID 欄中選擇您的伺服器 ID。 2. 在端點詳細資訊區段中，針對端點類型選擇端點 ID。這會帶您前往 Amazon VPC 主控台。 3. 在 Amazon VPC 主控台的詳細資訊索引標籤上，尋找 DNS 名稱旁的 DNS 名稱。	一般 AWS
建立 SFTP 用戶端金鑰對。	為 Microsoft Windows 或 macOS/Linux/UNIX 建立 SSH 金鑰對。	一般 AWS、SSH

任務	Description	所需的技能
建立 SFTP 使用者。	1. 開啟 Transfer Family 主控台，從導覽窗格中選擇伺服器，然後選取您的伺服器。 2. 在伺服器 ID 欄中，選擇您伺服器的伺服器 ID，然後選擇新增使用者。 3. 針對使用者名稱，輸入符合您 SSH 金鑰對使用者名稱的使用者名稱。 4. 針對角色，選擇您先前建立的 IAM 角色。 5. 針對主目錄，選擇您先前建立的 S3 儲存貯體。 6. 針對 SSH 公有金鑰， 輸入您先前建立的金鑰對。 7. 選擇新增。	一般 AWS

傳輸大型主機檔案

任務	Description	所需的技能

將 SSH 私有金鑰傳送至大型主機。	使用 SFTP 或 SCP 將 SSH 私有金鑰傳送至舊版環境。
	
	SFTP 範例：
	
	`sftp [USERNAME@mainframeIP]`
	`[password]`
	`cd [/u/USERNAME]`
	`put [your-key-pair-file]`
	
	SCP 範例：
		Mainframe、z/OS Unix shell、FTP、SCP

任務	Description	所需的技能
	`scp [your-key-pair-file] [USERNAME@MainframeIP]:/[u/USERNAME]` 接著，將 SSH 金鑰存放在 z/OS Unix 檔案系統中，並使用稍後將執行檔案傳輸批次任務的使用者名稱（例如，`/u/CONTROLM`）。 **Note** 如需 z/OS Unix shell 的詳細資訊，請參閱 z/OS shell 簡介 (IBM 文件)。	

任務	Description	所需的技能

建立 JCL SFTP 用戶端。	由於大型主機沒有原生 SFTP 用戶端，您必須使用 BPXBATCH 公用程式從 z/OS Unix shell 執行 SFTP 用戶端。
	
	在 ISPF 編輯器中，建立 JCL SFTP 用戶端。例如：
	
	`//JOBNAM JOB ...`
	`//************`
	`******************`
	`******************`
	`******************`
	`****`
	`//SFTP EXEC PGM=BPXBA`
	`TCH,REGION=0M`
	`//STDPARM DD *`
	`SH cp "//'MAINF`
	`RAME.FILE.NAME'"`
	` filename.txt;`
	`echo 'put filename.txt'`
	` > uplcmd;`
	`sftp -b uplcmd -i`
	` ssh_private_key_fi`
	`le ssh_username@<tran`
	`sfer service ip or`
	` DNS>;`
	`//SYSPRINT DD SYSOUT=*`
	`//STDOUT DD SYSOUT=*`
	`//STDENV DD *`
	`//STDERR DD SYSOUT=*`
	
	Note
	如需如何在 z/OS Unix shell 中執行命令的詳細資訊，請參閱 BPXBATCH 公用程式
		JCL、大型主機、z/OS Unix shell

任務	Description	所需的技能
	(IBM 文件）。如需如何在 z/OS 中建立或編輯 JCL 任務的詳細資訊，請參閱什麼是 ISPF？和 ISPF 編輯器 (IBM 文件）。	
執行 JCL SFTP 用戶端。	1. 在 ISPF 編輯器中，輸入 SUB，然後在建立 JCL 任務後按 ENTER 鍵。 2. 在 SDSF 中監控大型主機的檔案傳輸批次任務活動。 **Note** 如需如何檢查批次任務活動的詳細資訊，請參閱 z/OS SDSF 使用者指南 (IBM 文件）。	Mainframe、JCL、ISPF
驗證檔案傳輸。	1. 登入 AWS 管理主控台，開啟 Amazon S3 主控台，然後從導覽窗格中選擇儲存貯體。 2. 選擇與 Transfer Family 相關聯的儲存貯體。 3. 在物件索引標籤的物件區段中，尋找您從大型主機傳輸的檔案。	一般 AWS

任務	Description	所需的技能
自動化 JCL SFTP 用戶端。	使用任務排程器自動觸發 JCL SFTP 用戶端。 **Note** 您可以使用大型主機任務排程器，例如 BMC Control-M 或 CA 工作負載自動化，根據時間和其他批次任務相依性來自動化檔案傳輸的批次任務。	任務排程器

相關資源

- AWS Transfer 系列的運作方式

最佳化 AWS Blu Age 現代化應用程式的效能

Vishal Jaswani、Manish Roy 和 Himanshu Sah，Amazon Web Services

總結

使用 AWS Blu Age 進行現代化的大型主機應用程式需要先進行功能和效能相等性測試，才能部署到生產環境。在效能測試中，現代化應用程式的效能可能比傳統系統慢，尤其是在複雜的批次任務中。這種差異是因為大型主機應用程式是單體的，而現代應用程式使用多層架構。此模式提供最佳化技術，以解決透過使用自動重構搭配 AWS Blu Age 進行現代化應用程式的效能差距。

模式使用 AWS Blu Age 現代化架構搭配原生 Java 和資料庫調校功能，來識別和解決效能瓶頸。模式說明如何使用分析和監控來識別 SQL 執行時間、記憶體使用率和 I/O 模式等指標的效能問題。接著說明如何套用目標最佳化，包括資料庫查詢重組、快取和商業邏輯精簡。

改善批次處理時間和系統資源使用率，可協助您符合現代化系統中的大型主機效能等級。此方法在轉換為現代雲端架構期間維持功能等效性。

若要使用此模式，請依照 Epics 區段中的指示設定您的系統並識別效能熱點，並套用架構區段中詳細說明的最佳化技術。

先決條件和限制

先決條件

- AWS Blu Age 現代化應用程式
- JProfiler 授權
- 安裝資料庫用戶端和分析工具的管理權限
- AWS Blu Age Level 3 認證
- 中階了解 AWS Blu Age 架構、產生的程式碼結構和 Java 程式設計

限制

下列最佳化功能在此模式範圍之外：

- 應用程式層之間的網路延遲最佳化
- 透過 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體類型和儲存最佳化進行基礎設施層級最佳化

- 並行使用者負載測試和壓力測試

產品版本

- JProfiler 13.0 版或更新版本 （ 建議最新版本 ）
- pgAdmin 8.14 版或更新版本

Architecture

此模式使用 JProfiler 和 pgAdmin 等工具，為 AWS Blu Age 應用程式設定分析環境。它支援透過 AWS Blu Age 提供的 DAOManager 和 SQLExecutionBuilder APIs進行最佳化。

本節的其餘部分提供詳細資訊和範例，用於識別現代化應用程式的效能熱點和最佳化策略。Epics 區段中的步驟會參考此資訊，以取得進一步指引。

識別現代化大型主機應用程式中的效能熱點

在現代化大型主機應用程式中，效能熱點是程式碼中會導致顯著變慢或效率不佳的特定區域。這些熱點通常是由大型主機和現代化應用程式之間的架構差異所造成。若要識別這些效能瓶頸並最佳化現代化應用程式的效能，您可以使用三種技術：SQL 記錄、查詢EXPLAIN計畫和 JProfiler 分析。

熱點識別技術：SQL 記錄

現代 Java 應用程式，包括使用 AWS Blu Age 進行現代化的應用程式，具有記錄 SQL 查詢的內建功能。您可以在 AWS Blu Age 專案中啟用特定記錄器，以追蹤和分析應用程式執行的 SQL 陳述式。此技術對於識別效率不佳的資料庫存取模式特別有用，例如過多的個別查詢或結構不良的資料庫呼叫，可透過批次處理或查詢精簡進行最佳化。

若要在 AWS Blu Age 現代化應用程式中實作 SQL 記錄，請將 `application.properties` 檔案中 DEBUG SQL 陳述式的日誌層級設定為 ，以擷取查詢執行詳細資訊：

```
level.org.springframework.beans.factory.support.DefaultListableBeanFactory : WARN
level.com.netfective.bluage.gapwalk.runtime.sort.internal: WARN
level.org.springframework.jdbc.core.StatementCreatorUtils: DEBUG
level.com.netfective.bluage.gapwalk.rt.blu4iv.dao: DEBUG
level.com.fiserv.signature: DEBUG
level.com.netfective.bluage.gapwalk.database.support.central: DEBUG
level.com.netfective.bluage.gapwalk.rt.db.configuration.DatabaseConfiguration: DEBUG
level.com.netfective.bluage.gapwalk.rt.db.DatabaseInteractionLoggerUtils: DEBUG
```

```
level.com.netfective.bluage.gapwalk.database.support.AbstractDatabaseSupport: DEBUG
level.com.netfective.bluage.gapwalk.rt: DEBUG
```

使用記錄的資料來識別最佳化目標，以監控高頻率和效能緩慢的查詢。專注於批次程序中的查詢，因為它們通常具有最高的效能影響。

熱點識別技術：查詢 EXPLAIN 計劃

此方法使用關聯式資料庫管理系統的查詢規劃功能。您可以使用 EXPLAIN PostgreSQL 或 MySQL 或 Oracle EXPLAIN PLAN中的 等命令，來檢查資料庫執行指定查詢的方式。這些命令的輸出可提供查詢執行策略的寶貴洞見，包括將使用索引還是執行完整資料表掃描。此資訊對於最佳化查詢效能至關重要，尤其是在適當的索引可以大幅縮短執行時間的情況下。

從應用程式日誌中擷取最常重複的 SQL 查詢，並使用資料庫特有的EXPLAIN命令來分析執行緩慢查詢的執行路徑。以下是 PostgreSQL 資料庫的範例。

查詢：

```
SELECT * FROM tenk1 WHERE unique1 < 100;
```

EXPLAIN 命令：

```
EXPLAIN SELECT * FROM tenk1 where unique1 < 100;
```

輸出：

```
Bitmap Heap Scan on tenk1 (cost=5.06..224.98 rows=100 width=244)
Recheck Cond: (unique1 < 100)
-> Bitmap Index Scan on tenk1_unique1 (cost=0.00..5.04 rows=100 width=0)
Index Cond: (unique1 < 100)
```

您可以解譯EXPLAIN輸出，如下所示：

- 從最內到最外（從下到上）操作讀取EXPLAIN計劃。
- 尋找關鍵術語。例如， Seq Scan 表示完整資料表掃描，並Index Scan顯示索引用量。
- 檢查成本值：第一個數字是啟動成本，第二個數字是總成本。
- 請參閱預估輸出資料列數量rows的值。

在此範例中，查詢引擎會使用索引掃描來尋找相符的資料列，然後只擷取這些資料列 (Bitmap Heap Scan)。這比掃描整個資料表更有效率，即使個別資料列存取的成本較高。

EXPLAIN 計劃輸出中的資料表掃描操作表示缺少索引。最佳化需要建立適當的索引。

熱點識別技術：JProfiler 分析

JProfiler 是全方位的 Java 分析工具，可透過識別緩慢的資料庫呼叫和 CPU 密集型呼叫，協助您解決效能瓶頸。此工具在識別慢速 SQL 查詢和低效率記憶體使用量方面特別有效。

查詢的範例分析：

```
select evt.
 com.netfective.bluage.gapwalk.rt.blu4iv.dao.Blu4ivTableManager.queryNonTrasactional
```

JProfiler 熱點檢視提供下列資訊：

- 時間欄
  - 顯示總執行持續時間 （ 例如 329 秒 ）
  - 顯示應用程式總時間的百分比 （ 例如 58.7%)
  - 協助識別最耗時的操作
- 平均時間欄
  - 顯示每個執行持續時間 （ 例如 2，692 微秒 ）
  - 指出個別操作效能
  - 協助找出慢速個別操作
- 事件欄
  - 顯示執行計數 （ 例如 122，387 次 ）
  - 指出操作頻率
  - 協助識別經常呼叫的方法

針對範例結果：

- 高頻率：122，387 個執行表示可能進行最佳化
- 效能問題：平均時間 2，692 微秒表示效率低下
- 重大影響：總時間的 58.7% 表示重大瓶頸

JProfiler 可以分析應用程式的執行時間行為，以透過靜態程式碼分析或 SQL 記錄顯示可能不明顯的熱點。這些指標可協助您識別需要最佳化的操作，並判斷最有效的最佳化策略。如需 JProfiler 功能的詳細資訊，請參閱 JProfiler 文件。

結合使用這三種技術 (SQL 記錄、查詢EXPLAIN計畫和 JProfiler) 時，您可以全面了解應用程式的效能特性。透過識別和解決最重要的效能熱點，您可以彌補原始大型主機應用程式與現代化雲端系統之間的效能差距。

識別應用程式的效能熱點後，您可以套用最佳化策略，下一節會說明這些策略。

大型主機現代化最佳化策略

本節概述最佳化從大型主機系統現代化應用程式的關鍵策略。它著重於三種策略：使用現有的 APIs、實作有效的快取，以及最佳化商業邏輯。

最佳化策略：使用現有的 APIs

AWS Blu Age 在 DAO 介面中提供數個強大的 APIs，您可以用來最佳化效能。兩個主要界面 — DAOManager 和 SQLExecutionBuilder — 提供增強應用程式效能的功能。

DAOManager

DAOManager 做為現代化應用程式中資料庫操作的主要界面。它提供多種方法來增強資料庫操作和改善應用程式效能，尤其是直接建立、讀取、更新和刪除 (CRUD) 操作和批次處理。

- 使用 SetMaxResults。在 DAOManager API 中，您可以使用 SetMaxResults 方法來指定單一資料庫操作中要擷取的記錄數目上限。根據預設，DAOManager 一次只會擷取 10 筆記錄，這可能會在處理大型資料集時導致多個資料庫呼叫。當您的應用程式需要處理大量記錄，且目前正在進行多個資料庫呼叫來擷取記錄時，請使用此最佳化。這在您透過大型資料集反覆運算的批次處理案例中特別有用。在下列範例中，左側的程式碼（最佳化之前）會使用 10 筆記錄的預設資料擷取值。右側的程式碼（最佳化後）會將 setMaxResults 設定為一次擷取 100，000 筆記錄。

> **Note**
>
> 請仔細選擇較大的批次大小，並檢查物件大小，因為此最佳化會增加記憶體使用量。

- 將 SetOnGreatorOrEqual 取代為 SetOnEqual。此最佳化涉及變更您用來設定擷取記錄條件的方法。SetOnGreatorOrEqual 方法會擷取大於或等於指定值的記錄，而 SetOnEqual 只會擷取完全符合指定值的記錄。

當您知道需要完全相符，且目前使用 SetOnEqual SetOnGreatorOrEqual，如下列程式碼範例所示。readNextEqual 此最佳化可減少不必要的資料擷取。

- 使用批次寫入和更新操作。您可以使用批次操作將多個寫入或更新操作分組為單一資料庫交易。這可減少資料庫呼叫的數量，並可大幅改善涉及多個記錄的操作效能。

  在下列範例中，左側的程式碼會在迴圈中執行寫入操作，這會降低應用程式的效能。您可以使用批次寫入操作來最佳化此程式碼：在WHILE迴圈的每個反覆運算期間，您可以將記錄新增至批次，直到批次大小達到預先決定的大小 100。然後，您可以在達到預定批次大小時排清批次，然後將任何剩餘的記錄排清至資料庫。這在您處理需要更新的大型資料集時特別有用。

- 新增索引。新增索引是一種資料庫層級最佳化，可大幅改善查詢效能。索引可讓資料庫快速找到具有特定資料欄值的資料列，而無需掃描整個資料表。在子WHERE句、JOIN條件或ORDER BY陳述式中常用的資料欄上使用索引。這對大型資料表或快速資料擷取至關重要時尤其重要。

SQLExecutionBuilder

SQLExecutionBuilder 是一種靈活的 API，可用來控制將執行的 SQL INSERT 查詢、僅使用 擷取特定資料欄SELECT，以及使用動態資料表名稱。在下列範例中，SQLExecutorBuilder 會使用您定義的自訂查詢。

在 DAOManager 和 SQLExecutionBuilder 之間進行選擇

這些 APIs之間的選擇取決於您的特定使用案例：

- 如果您希望 AWS Blu Age Runtime 產生 SQL 查詢，而不是自行撰寫查詢，請使用 DAOManager。
- 當您需要寫入 SQL 查詢以利用資料庫特定的功能或寫入最佳的 SQL 查詢時，請選擇 SQLExecutionBuilder。

最佳化策略：快取

在現代化應用程式中，實作有效的快取策略可以大幅減少資料庫呼叫並改善回應時間。這有助於彌補大型主機和雲端環境之間的效能差距。

在 AWS Blu Age 應用程式中，簡單的快取實作使用內部資料結構，例如雜湊圖或陣列清單，因此您不需要設定需要成本和程式碼重組的外部快取解決方案。此方法對於經常存取但不常變更的資料特別有

效。當您實作快取時，請考慮記憶體限制條件和更新模式，以確保快取的資料保持一致並提供實際的效能優勢。

成功快取的關鍵是識別要快取的正確資料。在下列範例中，左側的程式碼一律會從資料表讀取資料，而當本機雜湊映射沒有指定金鑰的值時，右側的程式碼會從資料表讀取資料。 `cacheMap` 是在程式內容中建立並在程式內容的清除方法中清除的雜湊映射物件。

使用 DAOManager 快取：

使用 SQLExecutionBuilder 進行快取：

最佳化策略：商業邏輯最佳化

商業邏輯最佳化著重於重組 Blu AWS Age 自動產生的程式碼，以更符合現代架構功能。當產生的程式碼維護與舊版大型主機程式碼相同的邏輯結構時，這會變得必要，這可能不適用於現代系統。目標是改善效能，同時維持與原始應用程式的功能等效性。

此最佳化方法不僅止於簡單的 API 調校和快取策略。它涉及應用程式處理資料和與資料庫互動方式的變更。常見的最佳化包括避免簡單更新不必要的讀取操作、移除備援資料庫呼叫，以及重組資料存取模式，以更符合現代應用程式架構。以下是幾個範例：

- 直接在資料庫中更新資料。 使用直接 SQL 更新而非具有迴圈的多個 DAOManager 操作來重組您的業務邏輯。例如，下列程式碼（左側）會進行多個資料庫呼叫，並使用過多的記憶體。具體而言，它會在迴圈中使用多個資料庫讀取和寫入操作、個別更新而非批次處理，以及在每次反覆運算時建立不必要的物件。

  下列最佳化程式碼（右側）使用單一 Direct SQL 更新操作。具體而言，它使用單一資料庫呼叫，而不是多個呼叫，而且不需要迴圈，因為所有更新都是在單一陳述式中處理。此最佳化可提供更好的效能和資源使用率，並降低複雜性。它可防止 SQL 注入，提供更好的查詢計劃快取，並有助於提高安全性。

> **Note**
>
> 一律使用參數化查詢來防止 SQL 插入，並確保適當的交易管理。

- 減少備援資料庫呼叫。備援資料庫呼叫可能會大幅影響應用程式效能，尤其是在迴圈中發生時。簡單但有效的最佳化技術是避免多次重複相同的資料庫查詢。下列程式碼比較示範將`retrieve()`資料庫呼叫移至迴圈之外如何防止重複執行相同的查詢，進而提高效率。

- 使用 SQL 子句減少資料庫呼叫`JOIN`。實作 SQLExecutionBuilder，將對資料庫的呼叫降至最低。SQLExecutionBuilder 提供更多對 SQL 產生的控制，對於 DAOManager 無法有效處理的複雜查詢特別有用。例如，下列程式碼使用多個 DAOManager 呼叫：

```
List<Employee> employees = daoManager.readAll();
for(Employee emp : employees) {
    Department dept = deptManager.readById(emp.getDeptId());  // Additional call for
 each employee
    Project proj = projManager.readById(emp.getProjId());     // Another call for
 each employee
    processEmployeeData(emp, dept, proj);
}
```

  最佳化程式碼在 SQLExecutionBuilder 中使用單一資料庫呼叫：

```
SQLExecutionBuilder builder = new SQLExecutionBuilder();
builder.append("SELECT e.*, d.name as dept_name, p.name as proj_name");
builder.append("FROM employee e");
builder.append("JOIN department d ON e.dept_id = d.id");
builder.append("JOIN project p ON e.proj_id = p.id");
builder.append("WHERE e.status = ?", "ACTIVE");

List<Map<String, Object>> results = builder.execute();  // Single database call
for(Map<String, Object> result : results) {
    processComplexData(result);
}
```

同時使用最佳化策略

這三種策略可協同運作：APIs提供有效率的資料存取工具、快取可減少重複資料擷取的需求，而商業邏輯最佳化可確保以最有效的方式使用這些 APIs。這些最佳化的定期監控和調整可確保持續提升效能，同時維持現代化應用程式的可靠性和功能。成功的關鍵在於了解根據您的應用程式特性和效能目標，何時以及如何套用每個策略。

工具

- JProfiler 是一種 Java 分析工具，專為開發人員和效能工程師而設計。它可分析 Java 應用程式，並協助識別效能瓶頸、記憶體流失和執行緒問題。JProfiler 提供 CPU、記憶體和執行緒分析，以及資料庫和 Java 虛擬機器 (JVM) 監控，以深入了解應用程式行為。

> **Note**
>
> 作為 JProfiler 的替代方案，您可以使用 Java VisualVM。這是適用於 Java 應用程式的免費開放原始碼效能分析和監控工具，可提供 CPU 用量、記憶體消耗、執行緒管理和垃圾收集統計資料的即時監控。由於 Java VisualVM 是內建的 JDK 工具，因此對於基本分析需求，比 JProfiler 更具成本效益。

- pgAdmin 是 PostgreSQL 的開放原始碼管理和開發工具。它提供圖形界面，可協助您建立、維護和使用資料庫物件。您可以使用 pgAdmin 來執行各種任務，從撰寫簡單的 SQL 查詢到開發複雜的資料庫。其功能包括強調 SQL 編輯器的語法、伺服器端程式碼編輯器、SQL、殼層和批次任務的排程代理程式，以及支援新手和經驗豐富的 PostgreSQL 使用者的所有 PostgreSQL 功能。

最佳實務

識別效能熱點：

- 開始最佳化之前，請記錄基準效能指標。
- 根據業務需求設定明確的效能改善目標。
- 進行基準測試時，請停用詳細記錄，因為它可能會影響效能。
- 設定效能測試套件並定期執行。
- 使用最新版本的 pgAdmin。（舊版不支援`EXPLAIN`查詢計畫。）
- 針對基準測試，請在最佳化完成後分離 JProfiler，因為它會增加延遲。
- 針對基準測試，請務必以啟動模式而非偵錯模式執行伺服器，因為偵錯模式會增加延遲。

最佳化策略：

- 在 `application.yaml` 檔案中設定 SetMaxResults 值，根據您的系統規格指定適當大小的批次。
- 根據資料磁碟區和記憶體限制來設定 SetMaxResults 值。
- 只有在後續呼叫為 時，才將 SetOnGreatorOrEqual 變更為 SetOnEqual`.readNextEqual()`。

- 在批次寫入或更新操作中，請分別處理最後一個批次，因為它可能小於設定的批次大小，且寫入或更新操作可能會遺漏。

快取：

- 在 中引入用於快取的欄位`processImpl`，其會隨著每次執行而變動，應一律在該 的內容中定義`processImpl`。這些欄位也應該使用 `doReset()`或 `cleanUp()`方法清除。
- 當您實作記憶體內快取時，請調整快取的大小。儲存在記憶體中的非常大型快取可能會佔用所有資源，這可能會影響應用程式的整體效能。

SQLExecutionBuilder：

- 對於您計劃在 SQLExecutionBuilder 中使用的查詢，請使用金鑰名稱，例如 `PROGRAMNAME_STATEMENTNUMBER`。
- 當您使用 SQLExecutionBuilder 時，請務必檢查 `Sqlcod` 欄位。此欄位包含一個值，指定查詢是否正確執行或遇到任何錯誤。
- 使用參數化查詢來防止 SQL 注入。

商業邏輯最佳化：

- 在重組程式碼時維持功能等效性，並針對相關程式子集執行迴歸測試和資料庫比較。
- 維護分析快照以進行比較。

史詩

安裝 JProfiler 和 pgAdmin

任務	Description	所需的技能
安裝和設定 JProfiler。	1. 在 JProfiler 下載頁面上，為您的作業系統 (Windows、macOS 或 Linux) 下載 JProfiler。 2. 遵循 JProfiler 文件中的指示完成安裝程序。	應用程式開發人員

任務	Description	所需的技能
	3. 執行初始組態： a. 啟動 JProfiler 應用程式。 b. 啟用授權金鑰。 c. 設定預設 JVM 設定。 4. 驗證安裝： a. 啟動 JProfiler。 b. 執行測試分析工作階段。如需說明，請參閱 JProfiler 文件的分析 JVM 區段中的連接至本機服務。	
安裝和設定 pgAdmin。	在此步驟中，您會安裝並設定資料庫用戶端來查詢資料庫。此模式使用 PostgreSQL 資料庫和 pgAdmin 做為資料庫用戶端。如果您使用的是另一個資料庫引擎，請遵循對應資料庫用戶端的文件。 1. 從 pgAdmin 下載頁面下載並安裝作業系統的 pgAdmin。在安裝期間設定主要密碼。 2. 連線至資料庫伺服器。如需說明，請參閱 pgAdmin 文件。 3. 使用 pgAdmin Query Tool 執行基本 SQL 查詢來驗證安裝。	應用程式開發人員

識別熱點

任務	Description	所需的技能
在 AWS Blu Age 應用程式中啟用 SQL 查詢記錄。	在 AWS Blu Age 應用程式的 `application.properties` 檔案中啟用 SQL 查詢記錄的記錄器，如架構一節所述。	應用程式開發人員
產生和分析查詢EXPLAIN計劃，以識別資料庫效能熱點。	如需詳細資訊，請參閱架構一節。	應用程式開發人員
建立 JProfiler 快照以分析效能緩慢的測試案例。	1. 啟動 JProfiler： a. 開啟 JProfiler 應用程式。 b. 在開始中心，選擇快速連接索引標籤。 c. 從清單中選擇 Tomcat 程序。 d. 選擇 開始使用。 2. 設定分析設定： a. 選擇建議的取樣選項。 b. 在組態畫面上保留預設選項，然後選擇確定。 c. 等待連線狀態顯示在狀態列中。 3. 記錄效能資料： a. 在工具列上，選擇開始錄製。 b. 執行您的測試案例並等待其完成。 c. 選擇停止錄製。 4. 儲存並檢視快照： a. 選擇儲存快照。	應用程式開發人員

任務	Description	所需的技能
	b. 選擇您要儲存快照的位置，然後選擇儲存。	

任務	Description	所需的技能
分析 JProfiler 快照以識別效能瓶頸。	請依照下列步驟分析 JProfiler 快照。 1. 開啟快照： a. 選擇檔案、開啟快照。 b. 導覽至儲存的快照位置。 c. 選擇開啟新視窗以開始分析。 2. 分析資料庫操作： a. 導覽至 JDBC/資料庫區段，然後選擇 JPA/Hibernate 標籤。 b. 依時間檢閱查詢（這是預設排序選項）。 c. 選擇最上方的查詢以進行最佳化。 3. 檢閱 CPU 用量： a. 在 CPU 區段中，選擇熱點檢視。 b. 識別時間百分比最高的程式碼以進行最佳化。 c. 請注意任何週期性模式。 4. 識別熱點後，請建立基準，然後實作適當的最佳化策略，如下列語彙所述。專注於這些最佳化： • 對於慢查詢，請使用 SQLExecutionBuilder 或 DAOManager 最佳化。 • 若要經常存取資料，請實作快取機制。	應用程式開發人員

任務	Description	所需的技能
	• 對於複雜的商業邏輯，簡化邏輯、避免進行太多資料庫呼叫，以及透過執行簡單的查詢來判斷程式碼是否可以簡化。例如，如果程式碼將資料表 A 中的資料插入資料表 B，請重寫此程式碼以執行類似如下的 SQL 命令： `INSERT INTO TABLE A SELECT * FROM TABLE B` 5. 繼續依影響順序分析和最佳化程式碼（從最高到最低），直到應用程式符合您的效能需求為止。 如需使用 JProfiler 的詳細資訊，請參閱架構一節和 JProfiler 文件。	

建立基準

任務	Description	所需的技能
實作最佳化之前，請先建立效能基準。	1. 做為初始效能測量，請執行您的測試案例，而不進行任何最佳化。建立目前狀態的 JProfiler 快照，並記錄下列指標： • 總執行時間 • 資料庫呼叫的數量	應用程式開發人員

任務	Description	所需的技能
	• CPU 使用率百分比 • 記憶體用量模式 2. 建立文件基準： a. 儲存具有時間戳記和測試條件的所有指標。 b. 請注意可能影響效能的任何外部因素。 c. 儲存 JProfiler 快照以在最佳化後進行比較。 3. 規劃下一史詩中描述的最佳化變更： a. 檢閱基準指標。 b. 識別您要最佳化的操作。 c. 建立目前程式碼的備份。 d. 記錄預期的改進。 4. 在實作之前測量每個變更： a. 使用基準條件執行測試案例。 b. 記錄執行時間。 c. 記錄目前的行為。	

套用最佳化策略

任務	Description	所需的技能
最佳化讀取呼叫。	使用 DAOManager SetMaxResults 方法最佳化資料擷取。如需此方法的詳細資訊，請參閱架構一節。	應用程式開發人員，DAOManager
重構商業邏輯，以避免對資料庫進行多次呼叫。	使用 SQL `JOIN`子句減少資料庫呼叫。如需詳細資訊和範	應用程式開發人員，SQLExecutionBuilder

任務	Description	所需的技能
	例，請參閱《架構》中的商業邏輯最佳化。	
重構程式碼以使用快取來降低讀取呼叫的延遲。	如需此技術的資訊，請參閱架構一節中的快取。	應用程式開發人員
重寫使用多個 DAOManager 操作進行簡單更新操作的無效程式碼。	如需直接在資料庫中更新資料的詳細資訊，請參閱架構一節中的商業邏輯最佳化。	應用程式開發人員

測試最佳化策略

任務	Description	所需的技能
反覆驗證每個最佳化變更，同時保持功能等效性。	1. 在您實作最佳化之後： a. 從您建立的基準執行相同的測試案例。 b. 記錄新的執行時間。 c. 確認結果符合基準輸出。 2. 如果最佳化導致改善，請記錄實際收益，然後繼續進行下一個最佳化。 如果您發現問題，請復原變更、檢閱您的方法，並嘗試替代解決方案。 3. 追蹤每個變更的進度： a. 比較新指標與基準指標。 b. 計算改善百分比。 c. 維持總增益。	應用程式開發人員

任務	Description	所需的技能
	Note 使用基準指標做為參考，可確保準確測量每個最佳化的影響，同時維護系統可靠性。	

疑難排解

問題	解決方案
當您執行現代應用程式時，您會看到錯誤為 的例外狀況`Query_ID not found`。	若要解決此問題： 1. 檢查應用程式伺服器工作目錄中的 `query.sql sql` 資料夾中是否存在名為 的檔案。 2. 確認`.sql`檔案名稱符合程式名稱。例如，對於名為 的程式`ABC`，檔案名稱應為 `ABC.sql`。
您已新增索引，但沒有看到任何效能改善。	請依照下列步驟來確保查詢引擎正在使用 索引： 1. 如果您使用的是 PostgreSQL，請執行命令 `EXPLAIN <sql query>`。 2. 檢查 `Seq Scan`或 命令的輸出`Index Scan`。 `Seq Scan`表示索引集不適用於 SQL 查詢 `where`子句正在使用的欄。 3. 在正確的資料欄集上建立索引，並嘗試形成使用所建立索引的查詢。
您遇到out-of-memory例外狀況。	驗證程式碼是否釋出資料結構所保留的記憶體。

問題	解決方案
批次寫入操作會導致資料表中的記錄遺失	檢閱程式碼，以確保在批次計數不為零時執行額外的寫入操作。
SQL 記錄不會出現在應用程式日誌中。	1. 驗證記錄組態檔案位置和語法。 2. 確定指定套件DEBUG的日誌層級設定為 。例如： `level.com.netfective.bluage .gapwalk.rt.blu4iv.dao: DEBUG level.org.springframework.jdbc .core: DEBUG` 3. 檢查應用程式伺服器日誌檔案許可。 4. 在組態變更後重新啟動應用程式伺服器。

相關資源

- 使用 AWS Blu Age 自動重構應用程式 (AWS Mainframe Modernization 使用者指南)
- pgAdmin 文件
- JProfiler 文件

使用信任的內容來保護和簡化 AWS 上 Db2 聯合資料庫中的使用者存取

Sai Parthasaradhi，Amazon Web Services

總結

許多公司正在將舊版大型主機工作負載遷移至 Amazon Web Services (AWS)。此遷移包括將 z/OS 資料庫的 IBM Db2 轉移至 Amazon Elastic Compute Cloud (Amazon EC2) 上的適用於 Linux、Unix 和 Windows (LUW) 的 Db2。Amazon EC2 從現場部署到 AWS 的分階段遷移期間，使用者可能需要存取 Amazon EC2 上的 IBM Db2 z/OS 和 Db2 LUW 中的資料，直到所有應用程式和資料庫完全遷移至 Db2 LUW。在這種遠端資料存取案例中，使用者身分驗證可能具有挑戰性，因為不同的平台使用不同的身分驗證機制。

此模式涵蓋如何在 Db2 for LUW 上設定聯合伺服器，並將 Db2 for z/OS 設定為遠端資料庫。模式使用信任的內容，將使用者的身分從 Db2 LUW 傳播到 Db2 z/OS，而無需在遠端資料庫上重新驗證。如需受信任內容的詳細資訊，請參閱其他資訊一節。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在 Amazon EC2 執行個體上執行的 Db2 執行個體 Amazon EC2
- 在內部部署執行之 z/OS 資料庫的遠端 Db2
- 透過 AWS Site-to-Site VPN 或 AWS Direct Connect 連線至 AWS 的內部部署網路

Architecture

目標架構

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- AWS Site-to-Site VPN 可協助您在 AWS 上啟動的執行個體與您自己的遠端網路之間傳遞流量。

其他工具

- db2cli 是 Db2 互動式命令列界面 (CLI) 命令。

史詩

在 AWS 上執行的 Db2 LUW 資料庫上啟用聯合

任務	Description	所需的技能
在 DB2 LUW 資料庫上啟用聯合。	若要在 DB2 LUW 上啟用聯合，請執行下列命令。 `update dbm cfg using federated YES`	DBA
重新啟動資料庫。	若要重新啟動資料庫，請執行下列命令。 `db2stop force; db2start;`	DBA

為遠端資料庫編製目錄

任務	Description	所需的技能
為遠端 Db2 z/OS 子系統編製目錄。	若要在 AWS 上執行的 Db2 LUW 上為遠端 Db2 z/OS 資料庫編製目錄，請使用下列範例命令。 Db2 `catalog TCPIP NODE tcpnode REMOTE mainframehost SERVER mainframeport`	DBA

任務	Description	所需的技能
為遠端資料庫編製目錄。	若要為遠端資料庫編製目錄，請使用下列範例命令。 `catalog db dbnam1 as ndbnam1 at node tcpnode`	DBA

建立遠端伺服器定義

任務	Description	所需的技能
收集遠端 Db2 z/OS 資料庫的使用者憑證。	在繼續後續步驟之前，請收集下列資訊： • Db2 z/OS 子系統名稱 – 上一個步驟中 LUW 上的目錄化 Db2 z/OS 名稱（例如，ndbnam1) • Db2 z/OS 版本 – Db2 z/OS 子系統版本（例如 12) • Db2 z/OS 使用者 ID – 具有 BIND 權限的使用者，只需要建立伺服器定義（例如，dbuser1) • Db2 z/OS 密碼 – 的密碼 dbuser1（例如 dbpasswd) • Db2 z/OS 代理使用者 – 代理使用者的 ID，將用於建立信任的連線（例如 zproxy) • Db2 z/OS 代理密碼 – zproxy使用者的密碼（例如 zproxy)	DBA

任務	Description	所需的技能
建立 DRDA 包裝函式。	若要建立 DRDA 包裝函式，請執行下列命令。 `CREATE WRAPPER DRDA;`	DBA
建立伺服器定義。	若要建立伺服器定義，請執行下列範例命令。 `CREATE SERVER ndbserver TYPE DB2/ZOS VERSION 12 WRAPPER DRDA AUTHORIZATION "dbuser1" PASSWORD "dbpasswd " OPTIONS ( DBNAME 'ndbnam1',FED_PROXY_USER 'ZPROXY' );` 在此定義中， FED_PROXY_USER 指定將用於建立 Db2 z/OS 資料庫信任連線的代理使用者。只有在 Db2 LUW 資料庫中建立遠端伺服器物件時，才需要授權使用者 ID 和密碼。它們稍後不會在執行時間使用。	DBA

建立使用者映射

任務	Description	所需的技能
為代理使用者建立使用者映射。	若要為代理使用者建立使用者映射，請執行下列命令。 `CREATE USER MAPPING FOR ZPROXY SERVER ndbserver OPTIONS (REMOTE_AUTHID`	DBA

任務	Description	所需的技能
	'ZPROXY', REMOTE_PA SSWORD 'zproxy');	
在 Db2 LUW 上為每個使用者建立使用者映射。	為 AWS 上需要透過代理使用者存取遠端資料的 Db2 LUW 資料庫上的所有使用者建立使用者映射。若要建立使用者映射，請執行下列命令。 CREATE USER MAPPING FOR PERSON1 SERVER ndbserver OPTIONS (REMOTE_AUTHID 'USERZID', USE_TRUST ED_CONTEXT 'Y'); 陳述式指定 Db2 LUW (PERSON1) 上的使用者可以建立與遠端 Db2 z/OS 資料庫 () 的信任連線USE_TRUSTED_CONTEXT 'Y' 。透過代理使用者建立連線後，使用者可以使用 Db2 z/OS 使用者 ID () 存取資料REMOTE_AUTHID 'USERZID' 。	DBA

建立信任的內容物件

任務	Description	所需的技能
建立信任的內容物件。	若要在遠端 Db2 z/OS 資料庫上建立信任的內容物件，請使用下列範例命令。 CREATE TRUSTED CONTEXT CTX_LUW_ZOS	DBA

任務	Description	所需的技能
	BASED UPON CONNECTIO N USING SYSTEM AUTHID ZPROXY ATTRIBUTES (ADDRESS '10.10.10.10') NO DEFAULT ROLE ENABLE WITH USE FOR PUBLIC WITHOUT AUTHENTIC ATION; 在此定義中，CTX_LUW_Z OS 是受信任內容物件的任意名稱。物件包含代理使用者 ID，以及信任連線必須源自的伺服器 IP 地址。在此範例中，AWS 上的 Db2 LUW 資料庫伺服器。您可以使用網域名稱，而不是 IP 地址。子句WITH USE FOR PUBLIC WITHOUT AUTHENTICATION 指出每個使用者 ID 允許在信任的連線上切換使用者 ID。不需要提供密碼。	

相關資源

- IBM 資源存取控制設施 (RACF)
- IBM Db2 LUW 聯合
- 信任的內容

其他資訊

Db2 信任的內容

受信任內容是 Db2 資料庫物件，可定義聯合伺服器與遠端資料庫伺服器之間的信任關係。若要定義信任關係，信任的內容會指定信任屬性。信任屬性有三種類型：

- 發出初始資料庫連線請求的系統授權 ID
- 進行連線的 IP 地址或網域名稱
- 資料庫伺服器與資料庫用戶端之間資料通訊的加密設定

當連線請求的所有屬性符合伺服器上定義之任何信任內容物件中指定的屬性時，就會建立信任的連線。信任的連線有兩種類型：隱含和明確。建立隱含信任的連線後，使用者會繼承在該信任的連線定義範圍外無法使用的角色。建立明確信任的連線後，使用者可以在相同實體連線上開啟，無論是否進行身分驗證。此外，可以授予 Db2 使用者角色，這些角色會指定只能在信任連線內使用的權限。此模式使用明確的信任連線。

此模式中的信任內容

模式完成後，Db2 LUW 上的 PERSON1 會使用聯合信任內容從 Db2 z/OS 存取遠端資料。如果連線源自於信任內容定義中指定的 IP 地址或網域名稱，則會透過代理使用者建立 PERSON1 的連線。建立連線後，會切換 PERSON1 的對應 Db2 z/OS 使用者 ID，無需重新驗證，而且使用者可以根據為該使用者設定的 Db2 權限來存取資料或物件。

聯合信任內容的優點

- 此方法透過消除使用一般使用者 ID 或應用程式 ID 來維護最低權限原則，這些 ID 需要所有使用者所需權限的超集。
- 對聯合資料庫和遠端資料庫執行交易的使用者的真實身分始終是已知的，並且可以進行稽核。
- 效能提升，因為實體連線會在使用者之間重複使用，而不需要重新驗證聯合伺服器。

以 CSV 檔案將大規模 Db2 z/OS 資料傳輸至 Amazon S3

Bruno Sahinoglu、Abhijit Kshirsagar 和 Ivan Schuster，Amazon Web Services

總結

大型主機仍然是許多企業的記錄系統，其中包含大量資料，包括具有目前記錄的主要資料實體，以及歷史商業交易。它通常是孤立的，且不易被同一企業內的分散式系統存取。隨著雲端技術和大數據民主化的出現，企業有興趣使用大型主機資料中隱藏的洞察來開發新的業務功能。

有了這個目標，企業希望將大型主機 Db2 資料開放至其 Amazon Web Services (AWS) 雲端環境。業務原因有幾個，轉移方法因案例而異。您可能偏好將應用程式直接連接到大型主機，或者您可能偏好近乎即時地複寫資料。如果使用案例是饋送資料倉儲或資料湖，則不再需要擁有up-to-date複本，而且此模式中描述的程序可能就足夠，尤其是如果您想要避免任何第三方產品授權成本。另一個使用案例可能是遷移專案的大型主機資料傳輸。在遷移案例中，需要資料才能執行功能相等性測試。本文章中所述的方法是一種經濟實惠的方式，可將 Db2 資料傳輸到 AWS 雲端環境。

由於 Amazon Simple Storage Service (Amazon S3) 是最整合的 AWS 服務之一，因此您可以從該處存取資料，並使用 Amazon Athena、AWS Lambda 函數或 Amazon QuickSight 等其他 AWS 服務直接收集洞見。您也可以使用 AWS Glue 或 AWS Database Migration Service (AWS DMS) 將資料載入 Amazon Aurora 或 Amazon DynamoDB。考慮到這一點，這說明如何在大型主機上以 ASCII 格式卸載 CSV 檔案中的 Db2 資料，並將檔案傳輸至 Amazon S3。

為此，已開發大型主機指令碼，以協助產生任務控制語言 (JCLs)，以視需要卸載和傳輸任意數量的 Db2 資料表。

先決條件和限制

先決條件

- IBM z/OS 作業系統使用者，有權執行 Restructured Extended Executor (REXX) 和 JCL 指令碼。
- 存取 z/OS Unix System Services (USS) 以產生 SSH （安全殼層） 私有和公有金鑰。
- 可寫入的 S3 儲存貯體。如需詳細資訊，請參閱 Amazon S3 文件中的建立您的第一個 S3 儲存貯體。 Amazon S3
- 啟用 AWS Transfer 系列 SSH 檔案傳輸通訊協定 (SFTP) 的伺服器，使用做為身分提供者管理的 Service 和做為 AWS 儲存服務的 Amazon S3。如需詳細資訊，請參閱 AWS Transfer Family 文件中的建立啟用 SFTP 的伺服器。

限制

- 此方法不適用於近乎即時或即時的資料同步。
- 資料只能從 Db2 z/OS 移至 Amazon S3，不能以其他方式移動。

Architecture

來源技術堆疊

- 在 z/OS 上執行 Db2 的大型主機

目標技術堆疊

- AWS Transfer 系列
- Amazon S3
- Amazon Athena
- Amazon QuickSight
- AWS Glue
- Amazon Relational Database Service (Amazon RDS)
- Amazon Aurora
- Amazon Redshift

來源和目標架構

下圖顯示以 ASCII CSV 格式產生、擷取和傳輸 Db2 z/OS 資料至 S3 儲存貯體的程序。

1. 系統會選取資料表清單，以便從 Db2 目錄進行資料遷移。
2. 此清單用於使用外部格式的數值和資料欄來推動卸載任務的產生。
3. 然後，資料會使用 AWS Transfer Family 傳輸到 Amazon S3。
4. AWS Glue 擷取、轉換和載入 (ETL) 任務可以轉換資料，並以指定的格式將其載入已處理的儲存貯體，或者 AWS Glue 可以直接將資料饋送至資料庫。
5. Amazon Athena 和 Amazon QuickSight 可用於查詢和轉譯資料以推動分析。

下圖顯示整個程序的邏輯流程。

1. 第一個稱為 TABNAME 的 JCL 將使用 Db2 公用程式 DSNTIAUL 來擷取和產生您計劃從 Db2 卸載的資料表清單。若要選擇資料表，您必須手動調整 SQL 輸入以選取和新增篩選條件，以包含一或多個 Db2 結構描述。
2. 第二個 JCL 稱為 REXXEXEC，將使用 JCL 骨架和 REXX 程式來處理 JCL TABNAME 建立的資料表清單，並為每個資料表名稱產生一個 JCL。每個 JCL 都會包含一個卸載資料表的步驟，以及另一個使用 SFTP 通訊協定將檔案傳送至 S3 儲存貯體的步驟。
3. 最後一個步驟包含執行 JCL 以卸載資料表並將檔案傳輸至 AWS。整個程序可以使用內部部署或 AWS 上的排程器來自動化。

工具

AWS 服務

- Amazon Athena 是一種互動式查詢服務，可協助您使用標準 SQL 直接在 Amazon Simple Storage Service (Amazon S3) 中分析資料。
- Amazon Aurora 是全受管關聯式資料庫引擎，專為雲端而建置，並與 MySQL 和 PostgreSQL 相容。
- AWS Glue 是全受管的擷取、轉換和載入 (ETL) 服務。它可協助您可靠地分類、清理、擴充和移動資料存放區和資料串流之間的資料。
- Amazon QuickSight 是一種雲端規模的商業智慧 (BI) 服務，可協助您在單一儀表板中視覺化、分析和報告您的資料。
- Amazon Redshift 是 AWS 雲端中的受管 PB 級資料倉儲服務。
- Amazon Relational Database Service (Amazon RDS) 可協助您在 AWS 雲端中設定、操作和擴展關聯式資料庫。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Transfer Family 是一種安全傳輸服務，可讓您將檔案傳入和傳出 AWS 儲存服務。

大型主機工具

- SSH 檔案傳輸通訊協定 (SFTP) 是一種安全的檔案傳輸通訊協定，允許遠端登入 並在伺服器之間傳輸檔案。SSH 透過加密所有流量來提供安全性。
- DSNTIAUL 是由 IBM 提供的範例程式，用於卸載資料。
- DSNUTILB 是由 IBM 提供的公用程式批次程式，用於從 DSNTIAUL 卸載具有不同選項的資料。

- z/OS OpenSSH 是在 Unix System Service 上執行的開放原始碼軟體 SSH 連接埠，位於 IBM 作業系統 z/OS 下。SSH 是在 TCP/IP 網路上執行的兩部電腦之間的安全加密連線程式。它提供多個公用程式，包括 ssh-keygen。
- REXX（重組延伸執行器）指令碼用於使用 Db2 卸載和 SFTP 步驟自動產生 JCL。

Code

此模式的程式碼可在 GitHub unloaddb2 儲存庫中使用。

最佳實務

對於第一次卸載，產生的 JCLs應該卸載整個資料表資料。

第一次完全卸載後，請執行增量卸載，以獲得更好的效能並節省成本。 在範本 JCL 面板中修補 SQL 查詢，以適應卸載程序的任何變更。

您可以手動轉換結構描述，或使用 Lambda 上的指令碼搭配 Db2 SYSPUNCH 做為輸入。對於工業程序，AWS Schema Conversion Tool (SCT) 是偏好的選項。

最後，使用大型主機型排程器或 AWS 上的排程器搭配大型主機上的代理程式，以協助管理和自動化整個程序。

史詩

設定 S3 儲存貯體

任務	Description	所需的技能
建立 S3 儲存貯體。	如需說明，請參閱建立您的第一個 S3 儲存貯體。	一般 AWS

設定 Transfer Family 伺服器

任務	Description	所需的技能
建立啟用 SFTP 的伺服器。	若要在 AWS Transfer Family 主控台上開啟和建立 SFTP 伺服器，請執行下列動作：	一般 AWS

任務	Description	所需的技能
	1. 在選擇通訊協定頁面上，選取 SFTP (SSH 檔案傳輸通訊協定） – 透過 Secure Shell 傳輸檔案核取方塊。 2. 針對身分提供者，選擇服務受管。 3. 針對端點，選擇可公開存取。 4. 針對網域，選擇 Amazon S3。 5. 在設定其他詳細資訊頁面上，保留預設設定。 6. 建立伺服器。	
建立 Transfer Family 的 IAM 角色。	若要為 Transfer Family 建立 AWS Identity and Access Management (IAM) 角色以存取 Amazon S3，請遵循建立 IAM 角色和政策中的指示。	AWS 管理員
新增 Amazon S3 服務受管使用者。	若要新增 Amazon S3 服務受管使用者，請遵循 AWS 文件中的指示，並使用大型主機使用者 ID。	一般 AWS

保護通訊協定

任務	Description	所需的技能
建立 SSH 金鑰。	在大型主機 USS 環境中，執行下列命令。 `ssh-keygen -t rsa`	大型主機開發人員

任務	Description	所需的技能
	Note 出現密碼短語提示時，請保留空白。	

將正確的授權層級提供給 SSH 資料夾和金鑰檔案。	根據預設，公有和私有金鑰會存放在使用者目錄 中`/u/home/username/.ssh` 。
	您必須將授權 644 提供給金鑰檔案，並將 700 提供給 資料夾。
	`chmod 644 .ssh/id_rsa`
	`chmod 700 .ssh`
		大型主機開發人員

將公有金鑰內容複製到您的 Amazon S3 服務受管使用者。	若要複製 USS 產生的公有金鑰內容，請開啟 AWS Transfer Family 主控台。 1. 在導覽窗格中，選擇 Servers (伺服器)。 2. 在伺服器 ID 欄中選擇識別符，以查看伺服器詳細資訊 3. 在使用者下，選擇使用者名稱以查看使用者詳細資訊 4. 在 SSH 公有金鑰下，選擇新增 SSH 公有金鑰，將公有金鑰新增至使用者。對於 SSH 公有金鑰，請輸入您的公有金鑰。您的金鑰會先經過 服務驗證，然後才能新增您的新使用者。 5. 選擇 Add key (新增金鑰)。	大型主機開發人員

產生 JCLs

任務	Description	所需的技能
產生範圍內的 Db2 資料表清單。	提供輸入 SQL 以建立資料遷移範圍的資料表清單。此步驟要求您使用 SQL where 子句指定選取條件，以佇列 Db2 目錄資料表 SYSIBM.SYSTABLES。您可以自訂篩選條件，以包含以特定字首開頭或根據增量卸載時間戳記的特定結構描述或資料表名稱。輸出是在大型主機上的實體序列 (PS) 資料集中擷取。此資料集將做為 JCL 產生下一階段的輸入。 在使用 JCL TABNAME 之前（您可以視需要重新命名它），請進行下列變更： 1. 以授權執行 Db2 公用程式的任務類別和使用者取代 <Jobcard>。 2. 取代 <HLQ1> 或自訂輸出資料集名稱，以符合您的站點標準。 3. 根據您的站點標準更新 PDSEs（延伸分割資料集）的 STEPLIB 堆疊。此模式中的範例使用 IBM 預設值。 4. 以您的安裝特定值取代 PLAN 名稱和 LIB。 5. 使用 Db2 目錄的選擇條件取代 <Schema> 和 <Prefix>。	大型主機開發人員

任務	Description	所需的技能

	6. 將產生的 JCL 儲存在 PDS （分割資料集）程式庫中。
	7. 提交 JCL。
	
	Db2 資料表清單擷取任務
	
	`<Jobcard>`
	`//*`
	`//* UNLOAD ALL THE TABLE NAMES FOR A PARTICULAR SCHEMA`
	`//*`
	`//STEP01 EXEC PGM=IEFBR14`
	`//*`
	`//DD1 DD DISP=(MOD,DELETE,DELETE),`
	`// UNIT=SYSDA,`
	`// SPACE=(1000,(1,1)),`
	`// DSN=<HLQ1>.DSN81210.TABLIST`
	`//*`
	`//DD2 DD DISP=(MOD,DELETE,DELETE),`
	`// UNIT=SYSDA,`
	`// SPACE=(1000,(1,1)),`
	`// DSN=<HLQ1>.DSN81210.SYSPUNCH`
	`//*`
	`//UNLOAD EXEC PGM=IKJEFT01,DYNAMNBR=20`
	`//SYSTSPRT DD SYSOUT=*`
	`//STEPLIB DD DISP=SHR,DSN=DSNC10.DBCG.SDSNEXIT`
	`// DD DISP=SHR,DSN=DSNC10.SDSNLOAD`	

任務	Description	所需的技能

	<pre>// DD DISP=SHR,
	DSN=CEE.SCEERUN
	// DD DISP=SHR,
	DSN=DSNC10.DBCG.RU
	NLIB.LOAD
	//SYSTSIN DD *
	  DSN SYSTEM(DBCG)
	  RUN PROGRAM(D
	SNTIAUL) PLAN(DSNT
	IB12) PARMS('SQL') -
	       LIB('DSNC
	10.DBCG.RUNLIB.LOAD')
	  END
	//SYSPRINT DD SYSOUT=*
	//*
	//SYSUDUMP DD SYSOUT=*
	//*
	//SYSREC00 DD DISP=(NEW
	,CATLG,DELETE),
	// UNIT=SYSD
	A,SPACE=(32760,(10
	00,500)),
	// DSN=<HLQ1
	>.DSN81210.TABLIST
	//*
	//SYSPUNCH DD DISP=(NEW
	,CATLG,DELETE),
	// UNIT=SYSD
	A,SPACE=(32760,(10
	00,500)),
	// VOL=SER=S
	CR03,RECFM=FB,LREC
	L=120,BLKSIZE=12
	// DSN=<HLQ1
	>.DSN81210.SYSPUNCH
	//*
	//SYSIN DD *
	   SELECT CHAR(CREA
	TOR), CHAR(NAME)
	     FROM SYSIBM.SY
	STABLES</pre>	

任務	Description	所需的技能

	<pre> WHERE OWNER =
	'<Schema>'
	     AND NAME LIKE
	'<Prefix>%'
	     AND TYPE = 'T';
	/*</pre>	

任務	Description	所需的技能
修改 JCL 範本。	此模式隨附的 JCL 範本包含一般任務卡和程式庫名稱。不過，大多數大型主機站點都有自己的資料集名稱、程式庫名稱和任務卡命名標準。例如，執行 Db2 任務可能需要特定任務類別。任務項目子系統實作 JES2 和 JES3 可以實施其他變更。標準負載程式庫的第一個限定詞可能與 不同SYS1，這是 IBM 預設。因此，在執行範本之前，請自訂範本以考量您的網站特定標準。 在骨架 JCL UNLDSKEL 中進行下列變更： 1. 使用有權執行 Db2 公用程式的任務類別和使用者修改任務卡。 2. 自訂輸出資料集名稱以符合您的站點標準。 3. 根據您的站點標準更新 PDSEs的 STEPLIB 堆疊。此模式中的範例使用 IBM 預設值。 4. <DSN> 以您的 Db2 子系統名稱和相互關聯 ID 取代 。 5. 將產生的 JCL 儲存在屬於 ISPSLIB 堆疊的 PDS 程式庫中，這是 ISPF 的標準骨架範本程式庫。 卸載和 SFTP JCL 骨架	大型主機開發人員

任務	Description	所需的技能

	`//&USRPFX.U JOB (DB2UNLOAD),'JOB', CLASS=A,MSGCLASS=A,`
	`//  TIME=1440 ,NOTIFY=&USRPFX`
	`//* DELETE DATASETS`
	`//STEP01  EXEC PGM=IEFBR14`
	`//DD01  DD DISP=(MOD ,DELETE,DELETE),`
	`//  UNIT=SYSD A,`
	`//  SPACE=(TR K,(1,1)),`
	`// DSN=&USRPFX..DB2.P UNCH.&JOBNAME`
	`//DD02  DD DISP=(MOD ,DELETE,DELETE),`
	`//  UNIT=SYSD A,`
	`//  SPACE=(TR K,(1,1)),`
	`// DSN=&USRPFX..DB2.U NLOAD.&JOBNAME`
	`//*`
	`//* RUNNING DB2 EXTRACTION BATCH JOB FOR AWS DEMO`
	`//*`
	`//UNLD01  EXEC PGM=DSNUTILB,REGIO N=0M,`
	`// PARM='<DSN>,UNLOAD'`
	`//STEPLIB  DD DISP=SHR,DSN=DSNC1 0.DBCG.SDSNEXIT`
	`//  DD  DISP=SHR, DSN=DSNC10.SDSNLOAD`
	`//SYSPRINT DD  SYSOUT=*`
	`//UTPRINT  DD  SYSOUT=*`
	`//SYSOUT  DD  SYSOUT=*`	

任務	Description	所需的技能
	`//SYSPUN01 DD DISP=(NEW,CATLG,DELETE), // SPACE=(CYL,(1,1),RLSE), // DSN=&USRPFX..DB2.PUNCH.&JOBNAME //SYSREC01 DD DISP=(NEW,CATLG,DELETE), // SPACE=(CYL,(10,50),RLSE), // DSN=&USRPFX..DB2.UNLOAD.&JOBNAME //SYSPRINT DD SYSOUT=* //SYSIN DD * UNLOAD DELIMITED COLDEL ',' FROM TABLE &TABNAME UNLDDN SYSREC01 PUNCHDDN SYSPUN01 SHRLEVEL CHANGE ISOLATION UR; /* //* //* FTP TO AMAZON S3 BACKED FTP SERVER IF UNLOAD WAS SUCCESSFUL //* //SFTP EXEC PGM=BPXBATCH,COND=(4,LE),REGION=0M //STDPARM DD * SH cp "//'&USRPFX..DB2.UNLOAD.&JOBNAME'" &TABNAME..csv; echo "ascii " >> uplcmd; echo "PUT &TABNAME..csv " >>>> uplcmd;`	

任務	Description	所需的技能

	sftp -b uplcmd -i .ssh/id_rsa &FTPUSER.@&FTPSITE;
	rm &TABNAME..csv;
	//SYSPRINT DD SYSOUT=*
	//STDOUT DD SYSOUT=*
	//STDENV DD *
	//STDERR DD SYSOUT=*	

任務	Description	所需的技能
產生大量卸載 JCL。	此步驟涉及使用 JCL 在 ISPF 環境下執行 REXX 指令碼。提供在第一個步驟上建立的範圍內資料表清單，做為針對TABLIST DD名稱產生大量 JCL 的輸入。JCL 會在針對名稱指定的使用者指定分割資料集中，為每個資料表ISPFILE DD名稱產生一個新的 JCL。事先配置此程式庫。每個新的 JCL 將有兩個步驟：一個步驟將 Db2 資料表卸載至檔案，一個步驟將檔案傳送至 S3 儲存貯體。 在 JCL REXXEXEC 中進行下列變更（您可以變更名稱）： 1. Job card user ID 使用在資料表上具有卸載授權的大型主機使用者 ID 來取代 。取代 SYSPROC、ISPPLIB、ISPMLI ISPSLIB和 ISPTLIB<HLQ1>值，或自訂 DSN 以符合您的站點標準。若要了解您的安裝特定值，請使用命令 TSO ISRDDN。 2. <MFUSER> 以使用者 ID 取代，該使用者 ID 在您的安裝中具有任務執行權限。 3. <FTPUSER> 以安裝中具有 USS 和 FTP 權限的使用者 ID 取代。假設此使用	大型主機開發人員

任務	Description	所需的技能

	者 ID 及其 SSH 安全金鑰位於大型主機上適當的 Unix Systems Services 目錄中。
	4. <AWS TransferFamily IP> 以 AWS Transfer 系列 IP 地址或網域名稱取代 。此地址將用於 SFTP 步驟。
	5. 套用網站標準調整並更新 REXX 計畫後，請提交 JCL，如下所述。
	
	大量 JCL 產生任務
	
	`//RUNREXX JOB (CREATEJCL),'RUNS ISPF TABLIST',`
	`CLASS=A,MSGCLASS=A,`
	`//         TIME=1440`
	`,NOTIFY=&SYSUID`
	`//* Most of the values`
	`  required can be updated`
	`  to your site specific`
	`//* values using the`
	`  command 'TSO ISRDDN' in`
	`  your ISPF session.`
	`//* Update all the lines`
	`  tagged with //update`
	`  marker to desired`
	`//* site specific`
	`  values.`
	`//ISPF EXEC PGM=IKJEFT01,REGION=2048K,DYNAMNBR=25`
	`//SYSPROC   DD`
	`  DISP=SHR,DSN=USER.Z23D.CLIST`	

任務	Description	所需的技能

	//SYSEXEC DD
	DISP=SHR,DSN=<HLQ1
	>.TEST.REXXLIB
	//ISPPLIB DD
	DISP=SHR,DSN=ISP.S
	ISPPENU
	//ISPSLIB DD
	DISP=SHR,DSN=ISP.S
	ISPSENU
	// DD
	DISP=SHR,DSN=<HLQ1
	>.TEST.ISPSLIB
	//ISPMLIB DD
	DSN=ISP.SISPMENU,D
	ISP=SHR
	//ISPTLIB DD
	DDNAME=ISPTABL
	// DD DSN=ISP.S
	ISPTENU,DISP=SHR
	//ISPTABL DD
	LIKE=ISP.SISPTENU,
	UNIT=VIO
	//ISPPROF DD
	LIKE=ISP.SISPTENU,
	UNIT=VIO
	//ISPLOG DD
	SYSOUT=*,RECFM=VA,
	LRECL=125
	//SYSPRINT DD SYSOUT=*
	//SYSTSPRT DD
	SYSOUT=*
	//SYSUDUMP DD
	SYSOUT=*
	//SYSDBOUT DD
	SYSOUT=*
	//SYSTSPRT DD
	SYSOUT=*
	//SYSUDUMP DD
	SYSOUT=*
	//SYSDBOUT DD
	SYSOUT=*	

任務	Description	所需的技能

	<pre>//SYSHELP DD
	 DSN=SYS1.HELP,DISP
	=SHR
	//SYSOUT DD SYSOUT=*
	//* Input list of
	 tablenames
	//TABLIST DD
	 DISP=SHR,DSN=<HLQ1
	>.DSN81210.TABLIST
	//* Output pds
	//ISPFILE DD
	 DISP=SHR,DSN=<HLQ1
	>.TEST.JOBGEN
	//SYSTSIN DD *
	ISPSTART CMD(ZSTEPS
	 <MFUSER> <FTPUSER> <AWS
	 TransferFamily IP>)
	/*</pre>
	使用 REXX 指令碼之前，請進行下列變更：
	1. 將 REXX 指令碼儲存在在 JCL REXXEXEC SYSEXEC堆疊下定義的 PDS 程式庫中，在上一個步驟中以 ZSTEPS 做為成員名稱進行編輯。如果您想要重新命名它，您應該更新 JCL 以滿足您的需求。
	2. 此指令碼使用追蹤選項，在發生錯誤時列印其他資訊。您可以改為在 EXECIO、ISPEXEC和 TSO陳述式後面新增錯誤處理程式碼，並移除追蹤列。
	3. 此指令碼使用LODnnnnn命名慣例產生成員名稱，最多	

任務	Description	所需的技能
	可支援 100，000 名成員。如果您有超過 100，000 個資料表，請使用較短的字首，並調整tempjob陳述式中的數字。 ZSTEPS REXX 指令碼 `/*REXX - - -

任務	Description	所需的技能

	<pre>ADDRESS ISPEXEC
	"FTOPEN "
	ADDRESS ISPEXEC
	"FTINCL UNLDSKEL"
	/* member will be
	saved in ISPDSN library
	allocated in JCL */
	ADDRESS ISPEXEC
	"FTCLOSE NAME("tem
	pjob")"
	END
	
	ADDRESS TSO "FREE
	F(TABLIST) "
	ADDRESS TSO "FREE
	F(ISPFILE) "
	
	exit 0</pre>	

執行 JCLs

任務	Description	所需的技能
執行 Db2 卸載步驟。	在產生 JCL 之後，您將擁有與需要卸載的資料表一樣多 JCLs。 此案例使用 JCL 產生的範例來解釋結構和最重要的步驟。 您不需要執行任何操作。以下資訊僅供參考。如果您打算提交您在上一個步驟中產生的 JCLs，請跳到提交 LODnnnnn JCLs任務。 使用 JCL 搭配 IBM 提供的 DSNUTILB Db2 公用程式卸	Mainframe 開發人員、系統工程師

任務	Description	所需的技能

	載 Db2 資料時，您必須確定卸載的資料不包含壓縮的數值資料。若要達成此目的，請使用 DSNUTILB `DELIMITED` 參數。
	
	`DELIMITED` 參數支援以 CSV 格式卸載資料，方法是新增字元做為文字欄位的分隔符號和雙引號，移除 VARCHAR 欄中的填補，並將所有數值欄位轉換為 EXTERNAL FORMAT，包括 DATE 欄位。
	
	下列範例顯示所產生 JCL 中的卸載步驟，使用逗號字元做為分隔符號。
	
	`UNLOAD`
	`DELIMITED COLDEL ','`
	`FROM TABLE SCHEMA_NA`
	`ME.TBNAME`
	`UNLDDN SYSREC01`
	`PUNCHDDN SYSPUN01`
	`SHRLEVEL CHANGE`
	`ISOLATION UR;`	

任務	Description	所需的技能
執行 SFTP 步驟。	若要從 JCL 使用 SFTP 通訊協定，請使用 BPXBATCH 公用程式。 SFTP 公用程式無法直接存取 MVS 資料集。您可以使用 copy 命令 (cp) 將序列檔案複製到 &USRPFX..DB2.UNLOAD.&JOBNAME USS 目錄，並在其中變成 &TABNAME..csv 。 使用私有金鑰 (id_rsa) 並使用 RACF 使用者 ID 做為使用者名稱來執行sftp命令，以連線至 AWS Transfer Family IP 地址。 `SH cp "//'&USRPFX..DB2.UNLOAD.&JOBNAME'" &TABNAME..csv; echo "ascii " >> uplcmd; echo "PUT &TABNAME..csv " >>>> uplcmd; sftp -b uplcmd -i .ssh/id_rsa &FTPUSER.@&FTP_TF_SITE; rm &TABNAME..csv;`	Mainframe 開發人員、系統工程師

任務	Description	所需的技能
提交 LODnnnnn JCLs。	先前的 JCL 已產生所有需要卸載、轉換為 CSV 並傳輸至 S3 儲存貯體的 LODnnnnn nnJCL 資料表。 在已產生的所有 JCLs上執行 `submit`命令。	Mainframe 開發人員、系統工程師

相關資源

如需本文件中使用的不同工具和解決方案的詳細資訊，請參閱下列各項：

- z/OS OpenSSH 使用者指南
- Db2 z/OS – UNLOAD 控制陳述式範例
- Db2 z/OS – 卸載分隔檔案
- Transfer 系列 – 建立啟用 SFTP 的伺服器
- Transfer Family – 使用服務受管使用者

其他資訊

在 Amazon S3 上取得 Db2 資料後，您有許多方法可以開發新的洞見。由於 Amazon S3 與 AWS 資料分析服務整合，因此您可以在分散式端自由使用或公開這些資料。例如，您可以執行下列動作：

- 在 Amazon S3 上建置資料湖，並使用query-in-place、分析和機器學習工具擷取寶貴的洞見，而無需移動資料。
- 透過設定與 AWS Transfer 系列整合的上傳後處理工作流程來啟動 Lambda 函數。
- 使用 AWS Glue 開發新的微服務來存取 Amazon S3 或全受管資料庫中的資料，AWS Glue 是一種無伺服器資料整合服務，可讓您輕鬆探索、準備和結合資料，以進行分析、機器學習和應用程式開發。AWS Glue

在遷移使用案例中，由於您可以將任何資料從大型主機傳輸到 S3，因此您可以執行下列動作：

- 淘汰實體基礎設施，並使用 Amazon S3 Glacier 和 S3 Glacier Deep Archive 建立符合成本效益的資料封存策略。

- 使用 Amazon S3 和其他 AWS 服務建置可擴展、耐用且安全的備份和還原解決方案，例如 S3 Glacier 和 Amazon Elastic File System (Amazon EFS)，以增強或取代現有的內部部署功能。

更多模式

- 使用 Terraform 部署 AWS WAF 解決方案的安全自動化
- 使用 Precisely Connect 將大型主機資料庫複寫至 AWS

管理

主題

- 成本管理
- 最終使用者運算
- 高效能運算
- 混合雲端
- 管理與治理
- 訊息與通訊
- 多帳戶策略

成本管理

主題

- 使用 AWS Cost Explorer 建立 AWS Glue 任務的詳細成本和用量報告 Cost Explorer
- 使用 AWS Cost Explorer 建立 Amazon EMR 叢集的詳細成本和用量報告
- 更多模式

使用 AWS Cost Explorer 建立 AWS Glue 任務的詳細成本和用量報告 Cost Explorer

Parijat Bhide 和 Aromal Raj Jayarajan，Amazon Web Services

總結

此模式說明如何透過設定使用者定義的成本分配標籤來追蹤 AWS Glue 資料整合任務的使用成本。https://docs.aws.amazon.com/awsaccountbilling/latest/aboutv2/custom-tags.html您可以使用這些標籤，在 AWS Cost Explorer 中為多個維度的任務建立詳細的成本和用量報告。例如，您可以在團隊、專案或成本中心層級追蹤用量成本。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 一或多個已啟用使用者定義標籤的 AWS Glue 任務

Architecture

目標技術堆疊

- AWS Glue
- AWS Cost Explorer

下圖顯示如何套用標籤來追蹤 AWS Glue 任務的使用成本。

該圖顯示以下工作流程：

1. 資料工程師或 AWS 管理員會為 AWS Glue 任務建立使用者定義的成本分配標籤。
2. AWS 管理員會啟用標籤。
3. 這些標籤會向 AWS Cost Explorer 報告中繼資料。

工具

- AWS Glue 是全受管的擷取、轉換和載入 (ETL) 服務。它可協助您可靠地分類、清理、擴充和移動資料存放區和資料串流之間的資料。
- AWS Cost Explorer 可協助您檢視和分析 AWS 成本和用量。

史詩

為您的 AWS Glue 任務建立和啟用標籤

任務	Description	所需的技能
為您的 AWS Glue 任務建立使用者定義的成本分配標籤。	將標籤新增至現有的 AWS Glue 任務 1. 登入 AWS 管理主控台，然後開啟 AWS Glue 主控台。 2. 在左側導覽窗格中的 ETL 下，選擇任務。 3. 在任務區段中，選擇您要標記的任務名稱。 4. 選擇 Job details (任務詳細資訊) 索引標籤。然後，展開進階屬性區段。 5. 針對標籤，選擇新增標籤。 6. 針對金鑰，輸入標籤的名稱。 7. （選用）針對值，輸入您要與金鑰相關聯的值。 8. （選用）針對您要為任務建立的每個標籤重複步驟 5-7。 9. 選擇儲存。	資料工程師

任務	Description	所需的技能
	將標籤新增至新的 AWS Glue 任務 1. 根據您的使用案例需求建立新的 AWS Glue 任務。如需說明，請參閱《AWS Glue 開發人員指南》中的在 AWS Glue 主控台上使用任務。 AWS Glue 2. 當您設定任務詳細資訊設定時，請遵循新增標籤至此任務現有 AWS Glue 任務區段的步驟 4-9。 **Note** 如需詳細資訊，請參閱《AWS Glue 開發人員指南》中的 AWS Glue 標籤。	
啟用使用者定義的成本分配標籤。	遵循 AWS Billing 使用者指南中啟用使用者定義的成本分配標籤中的指示。	AWS 管理員

為您的 AWS Glue 任務建立成本和用量報告

任務	Description	所需的技能
使用 AWS Cost Explorer 中的標籤篩選條件，為您的 AWS Glue 任務建立成本和用量報告。 Cost Explorer	1. 登入 AWS 管理主控台並開啟 AWS Cost Management 主控台。	一般 AWS、AWS 管理員

任務	Description	所需的技能
	2. 在左側導覽窗格中，請選擇報告。 3. 選擇建立新報告。 4. 針對選取報告類型，選擇成本和用量（建議）。然後，選擇建立報告。 5. 針對篩選條件，選擇服務。服務下拉式清單隨即出現。 6. 選取 Glue 旁的核取方塊。然後，選擇套用篩選條件。 7. 對於篩選條件，選擇標籤。標籤下拉式清單隨即出現。 8. 選擇團隊。然後，選取您已指派標籤之團隊旁的核取方塊。排除您尚未指派標籤的任何團隊。然後，選擇套用篩選條件。 9. 在圖表頂端，選擇標籤。然後，選擇您要為其建立報告的 AWS Glue 任務的標籤。 10.在圖表頂端，選擇過去 3 個月下拉式清單，然後選擇您要報告涵蓋的時間範圍。然後，選擇每月下拉式清單，並根據時間範圍選擇您希望報告中的明細項目彙總方式。 11選擇 Save as (另存為)。然後，輸入報告的標題。 12選擇儲存報告。	

任務	Description	所需的技能
	如需詳細資訊，請參閱《AWS Cost Management 使用者指南》中的使用 Cost Explorer 探索您的資料。	

使用 AWS Cost Explorer 建立 Amazon EMR 叢集的詳細成本和用量報告

Parijat Bhide 和 Aromal Raj Jayarajan，Amazon Web Services

總結

此模式說明如何透過設定使用者定義的成本分配標籤來追蹤 Amazon EMR 叢集的使用成本。您可以使用這些標籤，在 AWS Cost Explorer 中為多個維度的叢集建立詳細的成本和用量報告。例如，您可以在團隊、專案或成本中心層級追蹤用量成本。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 一或多個已啟用使用者定義標籤的 EMR 叢集

Architecture

目標技術堆疊

- Amazon EMR
- AWS Cost Explorer

目標架構

下圖顯示如何套用標籤來追蹤特定 Amazon EMR 叢集的使用成本。

該圖顯示以下工作流程：

1. 資料工程師或 AWS 管理員會為 Amazon EMR 叢集建立使用者定義的成本分配標籤。
2. AWS 管理員會啟用標籤。
3. 這些標籤會向 AWS Cost Explorer 報告中繼資料。

工具

工具

- Amazon EMR 是受管叢集平台，可簡化在 AWS 上執行大數據架構，以處理和分析大量資料。
- AWS Cost Explorer 可協助您檢視和分析 AWS 成本和用量。

史詩

為您的 Amazon EMR 叢集建立和啟用標籤

任務	Description	所需的技能
為您的 Amazon EMR 叢集建立使用者定義的成本分配標籤。	將標籤新增至現有的 Amazon EMR 叢集 請遵循 Amazon EMR 管理指南中將標籤新增至現有叢集中的指示。 將標籤新增至新的 Amazon EMR 叢集 請遵循《Amazon EMR 管理指南》中將標籤新增至新叢集中的指示。 如需如何設定 Amazon EMR 叢集的詳細資訊，請參閱《Amazon EMR 管理指南》中的規劃和設定叢集。	資料工程師
啟用使用者定義的成本分配標籤。	請遵循 AWS Billing 使用者指南中啟用使用者定義的成本分配標籤中的指示。	AWS 管理員

為您的 Amazon EMR 叢集建立成本和用量報告

任務	Description	所需的技能
使用 AWS Cost Explorer 中的標籤篩選條件，為您的 Amazon EMR 叢集建立成本和用量報告。	1. 登入 AWS 管理主控台並開啟 AWS Cost Management 主控台。 2. 在左側導覽窗格中，請選擇報告。 3. 選擇建立新報告。 4. 針對選取報告類型，選擇成本和用量 （建議）。然後，選擇建立報告。 5. 針對篩選條件，選擇服務。服務下拉式清單隨即出現。 6. 選取 EMR (Elastic MapReduce) 和 EC2-Instances (Elastic Compute Cloud – Compute) 旁的核取方塊。然後，選擇套用篩選條件。 7. 針對篩選條件，選擇標籤。標籤下拉式清單隨即出現。 8. 選擇團隊。然後，選取您指派標籤的團隊旁的核取方塊。排除您尚未指派標籤的任何團隊。然後，選擇套用篩選條件。 9. 在圖表頂端，選擇標籤。然後，選擇您要為其建立報告的 Amazon EMR 叢集的標籤。	一般 AWS、AWS 管理員

任務	Description	所需的技能
	10.在圖表頂端，選擇過去 3 個月下拉式清單，然後選擇您希望報告涵蓋的時間範圍。然後，選擇每月下拉式清單，並根據時間範圍選擇您希望報告中的明細項目彙總方式。 11選擇 Save as (另存為)。然後，輸入報告的標題。 12選擇儲存報告。 如需詳細資訊，請參閱《AWS Cost Management 使用者指南》中的使用 Cost Explorer 探索您的資料。	

更多模式

- 使用 Amazon Bedrock 自動化 AWS 基礎設施操作
- 跨多個帳戶和區域自動清查 AWS 資源
- 使用 自動建立 Amazon WorkSpaces 應用程式資源 AWS CloudFormation
- 使用 DynamoDB TTL 自動將項目封存至 Amazon S3
- 使用 AWS Systems Manager 維護 Windows 自動停止和啟動 Amazon RDS 資料庫執行個體
- 建立 Amazon RDS 和 Amazon Aurora 的詳細成本和用量報告
- Amazon DynamoDB 資料表的預估儲存成本
- 預估 DynamoDB 資料表的隨需容量成本
- 使用 Amazon EKS Pod Identity 和 KEDA 在 Amazon EKS 中設定事件驅動的自動擴展
- 使用 AWS Fargate WaitCondition 勾點建構來協調資源相依性和任務執行

最終使用者運算

主題

- 使用 Auth0 和 實作 Amazon WorkSpaces 的 SAML 2.Auth0驗證 AWS Managed Microsoft AD
- 更多模式

使用 Auth0 和 實作 Amazon WorkSpaces 的 SAML 2.Auth0驗證 AWS Managed Microsoft AD

Siva Vinnakota 和 Shantanu Padhye，Amazon Web Services

總結

此模式會探索如何將 Auth0 與 整合 AWS Directory Service for Microsoft Active Directory ，為您的 Amazon WorkSpaces 環境建立強大的 SAML 2.0 身分驗證解決方案。它說明如何在這些項目之間建立聯合 AWS 服務 ，以啟用多重要素驗證 (MFA) 和自訂登入流程等進階功能，同時保留無縫桌面存取 AWS Managed Microsoft AD。無論您是只管理少數使用者還是數千個使用者，此整合都有助於為您的組織提供靈活性和安全性。此模式提供設定程序的步驟，讓您可以在自己的環境中實作此解決方案。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- AWS Managed Microsoft AD
- Amazon WorkSpaces Personal 中與 相關聯的佈建桌面 AWS Managed Microsoft AD
- Amazon Elastic Compute Cloud (Amazon EC2) 執行個體
- Auth0 帳戶

限制

有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

Architecture

WorkSpaces 用戶端應用程式的 SAML 2.0 身分驗證程序包含五個步驟，如下圖所示。這些步驟代表用於登入的典型工作流程。您可以在遵循此模式中的指示後，使用此分散式方法進行身分驗證，以協助提供結構化且安全的使用者存取方法。

工作流程：

1. 註冊。使用者啟動 WorkSpaces 的用戶端應用程式，並為其已啟用 SAML 的 WorkSpaces 目錄輸入 WorkSpaces 註冊碼。WorkSpaces 會將 Auth0 身分提供者 (IdP) URL 傳回至用戶端應用程式。

2. 登入。 WorkSpaces 用戶端會使用 Auth0 URL 重新導向至使用者的 Web 瀏覽器。 使用者使用其使用者名稱和密碼進行身分驗證。Auth0 會將 SAML 聲明傳回至用戶端瀏覽器。SAML 聲明是可宣告使用者身分的加密字符。
3. 驗證。用戶端瀏覽器會將 SAML 聲明發佈至 AWS 登入 端點以進行驗證。 AWS 登入 允許發起人擔任 AWS Identity and Access Management (IAM) 角色。這會傳回包含 IAM 角色臨時登入資料的字符。
4. WorkSpaces 登入。WorkSpaces 用戶端會將權杖呈現給 WorkSpaces 服務端點。WorkSpaces 會交換工作階段字符，並使用登入 URL 將工作階段字符傳回給 WorkSpaces 用戶端。當 WorkSpaces 用戶端載入登入頁面時，使用者名稱值會填入在 SAML 回應中傳遞`NameId`的值。
5. 串流。使用者輸入其密碼並對 WorkSpaces 目錄進行身分驗證。身分驗證之後，WorkSpaces 會將權杖傳回給用戶端。用戶端會重新導向回 WorkSpaces 服務，並呈現權杖。這會代理 WorkSpaces 用戶端和 WorkSpace 之間的串流工作階段。

> **Note**
>
> 若要設定不需要密碼提示的無縫單一登入體驗，請參閱 WorkSpaces 文件中的憑證型身分驗證和 WorkSpaces Personal。 WorkSpaces

工具

AWS 服務

- Amazon WorkSpaces 是一項全受管的虛擬桌面基礎設施 (VDI) 服務，為使用者提供雲端型桌面，而無需購買和部署硬體或安裝複雜的軟體。
- AWS Directory Service for Microsoft Active Directory 可讓您的目錄感知工作負載 AWS 和資源在 中使用 Microsoft Active Directory AWS 雲端。

其他工具

- Auth0 是一種身分驗證和授權平台，可協助您管理對應用程式的存取。

史詩

在 Auth0 中設定 Active Directory LDAP 連接器

任務	Description	所需的技能
使用 在 Auth0 中安裝 Active Directory LDAP 連接器 AWS Managed Microsoft AD。	1. 登入 Auth0 儀表板，然後選擇身分驗證、企業、Active Directory/LDAP。選擇建立連線。 2. 提供 Active Directory 連線的名稱，然後選擇建立。 3. 在設定索引標籤上，下載作業系統的代理程式。 安裝完成時，您的預設瀏覽器會顯示票證 URL 的提示。 4. 輸入佈建票證 URL，這對於您的連線字串應該是唯一的，然後選擇繼續。 5. 在 AD LDAP 組態對話方塊中，針對使用者名稱和密碼，輸入您的管理員登入資料，然後選擇儲存。 建立與 Auth0 的連線後，組態日誌會顯示所有檢查的報告狀態為 OK。	雲端管理員、雲端架構師
在 Auth0 中建立應用程式，以產生 SAML 中繼資料清單檔案。	1. 登入 Auth0 儀表板，並遵循 Auth0 文件中的指示建立新的應用程式。 2. 在 Auth0 儀表板上，選擇應用程式名稱以存取其組態設定。在 AddOns 索引標籤	雲端管理員、雲端架構師

任務	Description	所需的技能
	上，選擇 SAML2 Web 應用程式。 3. 在附加元件的設定索引標籤上，針對應用程式回呼 URL，輸入 https://signin.aws.amazon.com/saml。這是 SAML 字符將傳送POST請求的位置。 4. 在設定索引標籤的設定方塊中，以 JSON 格式貼上下列 SAML 組態碼： `{ "audience": "https://signin.aws.amazon.com/saml", "mappings": { "email": "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/sAMAccountName", "name": "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name" }, "createUpnClaim": false, "passthroughClaimsWithNoMapping": false, "mapUnknownClaimsAsIs": false, "mapIdentities": false, "nameIdentifierFormat": "urn:oasis:names:tc:SAML:2.`	

任務	Description	所需的技能

	<pre>0:nameid-format:persistent",
	  "nameIdentifierProbes": [
	  "http://schemas.auth0.com/sAMAccountName"
	  ]
	}</pre>
	5. 儲存，然後選擇啟用。
	6. 選擇用量索引標籤，然後下載身分提供者的中繼資料資訊清單檔案。後續步驟需要此資訊。
	7. 關閉 SAML2 Web 應用程式視窗。
	8. 在應用程式畫面上，選擇連線。在企業下，選取正確的 Active Directory/LDAP 連接器並啟用它。	

在 IAM 中設定 SAML 2.0 的 IdP、角色和政策

任務	Description	所需的技能
在 IAM 中建立 SAML 2.0 IdP。	若要將 SAML 2.0 設定為 IdP，請遵循 IAM 文件中在 IAM 中建立 SAML 身分提供者中所述的步驟。	雲端管理員
建立 SAML 2.0 聯合的 IAM 角色和政策。	1. 為 SAML 2.0 聯合建立 IAM 角色。如需說明，請參閱 WorkSpaces 文件中設定 WorkSpaces Personal 專	雲端管理員

任務	Description	所需的技能
	用 SAML 2.0 的說明中的 WorkSpaces步驟 2。 2. 建立 IAM 政策，並將其與您在上一個步驟中建立的角色建立關聯。如需說明，請參閱 WorkSpaces 文件中設定 WorkSpaces Personal 專用 SAML 2.0 的說明中的 WorkSpaces步驟 3。	

在 Auth0 中設定宣告

任務	Description	所需的技能
設定 Auth0 和 SAML 聲明。	您可以使用 Auth0 動作在 SAML 2.0 回應中設定宣告。SAML 聲明是可宣告使用者身分的加密字符。 1. 登入 Auth0 儀表板。選擇動作、程式庫、建立動作、從頭開始建置。 2. 提供下列值，然後選擇建立。 名稱：指定動作的名稱 觸發：選擇登入/登入後 執行時間：選擇節點 18 3. 在下一個畫面上，輸入下列程式碼：	雲端管理員

任務	Description	所需的技能
	exports.onExecutePostLogin = async (event, api) => { if (event.client.name === "Workspace_Saml") { const awsRole = 'arn:aws:iam::030784294031:role/Workspace_Auth0,arn:aws:iam::030784294031:saml-provider/Auth0'; const awsRoleSession = event.user.sAMAccountName; const email = event.user.emails[0]; api.samlResponse.setDestination('https://signin.aws.amazon.com/saml'); api.samlResponse.setAttribute('https://aws.amazon.com/SAML/Attributes/Role', awsRole) api.samlResponse.setAttribute('https://aws.amazon.com/SAML/Attributes/RoleSessionName', awsRoleSession) api.samlResponse.setAttribute('https://aws.amazon.com/SAML/Attr	

任務	Description	所需的技能

	<pre>ibutes/PrincipalTa
	g:Email', email)
	} return;
	};</pre>
	4. 選擇部署。
	
	如此即完成 WorkSpaces Personal 桌面的 SAML 2.0 身分驗證設定。架構區段說明設定後的身分驗證程序。	

疑難排解

問題	解決方案
WorkSpaces 中的 SAML 2.0 身分驗證問題	如果您在實作 WorkSpaces Personal 的 SAML 2.0 身分驗證時遇到任何問題，請遵循 AWS re：Post 文章中關於對 SAML 2.0 身分驗證進行疑難排解的步驟和連結。 如需有關在存取 WorkSpaces 時調查 SAML 2.0 錯誤的其他資訊，請參閱： • 如何從瀏覽器為 AWS 支援 案例建立 HAR 檔案和主控台日誌？(AWS re：Post) • 在瀏覽器中檢視 SAML 回應 (IAM 文件)

相關資源

- 為 WorkSpaces Personal 設定 SAML 2.0 (WorkSpaces 文件)
- Auth0 文件

更多模式

- 使用 自動建立 Amazon WorkSpaces 應用程式資源 AWS CloudFormation
- 改善 Amazon Connect 聯絡中心中客服人員工作站的通話品質
- 從 AWS Step Functions 同步執行 AWS Systems Manager 自動化任務 AWS Step Functions

高效能運算

主題

- 使用 Terraform 和 DRA 部署 Lustre 檔案系統以進行高效能資料處理
- 設定 AWS ParallelCluster 的 Grafana 監控儀表板
- 更多模式

使用 Terraform 和 DRA 部署 Lustre 檔案系統以進行高效能資料處理

Arun Bagal 和 Ishwar Chauthaiwale，Amazon Web Services

總結

此模式會自動在 上部署 Lustre 檔案系統， AWS 並將其與 Amazon Elastic Compute Cloud (Amazon EC2) 和 Amazon Simple Storage Service (Amazon S3) 整合。

此解決方案可協助您快速設定具有整合式儲存、運算資源和 Amazon S3 資料存取的高效能運算 (HPC) 環境。它結合了 Lustre 的儲存功能與 Amazon EC2 提供的彈性運算選項，以及 Amazon S3 中可擴展的物件儲存，因此您可以在機器學習、HPC 和大數據分析中處理資料密集型工作負載。

模式使用 HashiCorp Terraform 模組和 Amazon FSx for Lustre 來簡化下列程序：

- 佈建 Lustre 檔案系統
- 在 FSx for Lustre 和 S3 儲存貯體之間建立資料儲存庫關聯 (DRA)，以將 Lustre 檔案系統與 Amazon S3 物件連結
- 建立 EC2 執行個體
- 在 EC2 執行個體上使用 Amazon S3 連結 DRA 掛載 Lustre 檔案系統

此解決方案的優點包括：

- 模組化設計。您可以輕鬆維護和更新此解決方案的個別元件。
- 延展性。 您可以跨 AWS 帳戶 或 區域快速部署一致的環境。
- 彈性。您可以自訂部署以符合您的特定需求。
- 最佳實務。此模式使用遵循 AWS 最佳實務的預先設定模組。

如需 Lustre 檔案系統的詳細資訊，請參閱 Lustre 網站。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 最低權限 AWS Identity and Access Management (IAM) 政策 （請參閱說明）

限制

FSx for Lustre 會將 Lustre 檔案系統限制在單一可用區域，如果您有高可用性需求，這可能會令人擔憂。如果包含檔案系統的可用區域失敗，則會失去對檔案系統的存取，直到復原為止。若要實現高可用性，您可以使用 DRA 將 Lustre 檔案系統與 Amazon S3 連結，並在可用區域之間傳輸資料。

產品版本

- Terraform 1.9.3 版或更新版本
- HashiCorp AWS 提供者 4.0.0 版或更新版本

Architecture

下圖顯示 FSx for Lustre 和 AWS 服務 中互補的架構 AWS 雲端。

架構包含下列項目：

- S3 儲存貯體可做為資料耐用、可擴展且符合成本效益的儲存位置。FSx for Lustre 和 Amazon S3 之間的整合提供與 Amazon S3 無縫連結的高效能檔案系統。
- FSx for Lustre 會執行和管理 Lustre 檔案系統。
- Amazon CloudWatch Logs 會從檔案系統收集和監控日誌資料。這些日誌可讓您深入了解 Lustre 檔案系統的效能、運作狀態和活動。
- Amazon EC2 用於使用開放原始碼 Lustre 用戶端存取 Lustre 檔案系統。EC2 執行個體可以從相同虛擬私有雲端 (VPC) 中的其他可用區域存取檔案系統。網路組態允許在 VPC 內的子網路之間存取 。在執行個體上掛載 Lustre 檔案系統之後，您可以使用其檔案和目錄，就像使用本機檔案系統一樣。
- AWS Key Management Service (AWS KMS) 透過提供靜態資料的加密，增強檔案系統的安全性。

自動化和擴展

Terraform 可讓您更輕鬆地跨多個環境部署、管理和擴展 Lustre 檔案系統。在 FSx for Lustre 中，單一檔案系統具有大小限制，因此您可能需要建立多個檔案系統來水平擴展。您可以使用 Terraform 根據您的工作負載需求佈建多個 Lustre 檔案系統。

工具

AWS 服務

- Amazon CloudWatch Logs 可協助您集中所有系統、應用程式和 的日誌， AWS 服務 以便您可以監控日誌並將其安全地存檔。
- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon FSx for Lustre 可讓您輕鬆且經濟實惠地啟動、執行和擴展高效能 Lustre 檔案系統。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

程式碼儲存庫

此模式的程式碼可在 GitHub Provision FSx for Lustre Filesystem 中使用 Terraform 儲存庫。

最佳實務

- 下列變數定義 Lustre 檔案系統。請務必根據您的環境正確設定這些項目，如 Epics 一節所述。
  - `storage_capacity` – Lustre 檔案系統的儲存容量，以 GiBs為單位。最小和預設設定為 1200 GiB。
  - `deployment_type` – Lustre 檔案系統的部署類型。如需兩個選項 `PERSISTENT_1`和 `PERSISTENT_2`（預設）的說明，請參閱 FSx for Lustre 文件。
  - `per_unit_storage_throughput` – 讀取和寫入輸送量，以每秒每 TiB MBs為單位。
  - `subnet_id` – 您要部署 FSx for Lustre 的私有子網路 ID。
  - `vpc_id` – 您想要在 AWS 其中部署 FSx for Lustre 的虛擬私有雲端 ID。
  - `data_repository_path` – 將連結至 Lustre 檔案系統的 S3 儲存貯體路徑。
  - `iam_instance_profile` – 用來啟動 EC2 執行個體的 IAM 執行個體描述檔。
  - `kms_key_id` – 將用於資料加密之 AWS KMS 金鑰的 Amazon Resource Name (ARN)。
- 使用 和 `vpc_id`變數，確保 VPC 內的適當網路存取`security_group`和放置。
- 如 Epics 章節所述執行 `terraform plan`命令，以在套用變更之前預覽和驗證變更。這有助於發現潛在問題，並確保您知道要部署的內容。
- 如 Epics 一節所述使用 `terraform validate`命令來檢查語法錯誤，並確認組態是否正確。

史詩

設定您的環境

任務	Description	所需的技能
安裝 Terraform。	若要在本機電腦上安裝 Terraform，請遵循 Terraform 文件中的指示。	AWS DevOps，DevOps 工程師
設定 AWS 登入資料。	若要設定帳戶的 AWS Command Line Interface (AWS CLI) 設定檔，請遵循 AWS 文件中的指示。	AWS DevOps，DevOps 工程師
複製 GitHub 儲存庫。	若要複製 GitHub 儲存庫，請執行 命令： `git clone https://github.com/aws-samples/provision-fsx-lustre-with-terraform.git`	AWS DevOps，DevOps 工程師

設定和部署 FSx for Lustre

任務	Description	所需的技能
更新部署組態。	1. 在本機電腦上複製的儲存庫中，導覽至 `fsx_deployment` 目錄： `cd fsx_deployment` 2. 開啟 `terraform.tfvars` 檔案，並更新下列變數的值： • `vpc_id`	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	• subnet_id • data_repository_path • iam_instance_profile • kms_key_id 如需這些變數的說明，請參閱最佳實務一節。 3. 在相同的目錄中，開啟 locals.tf 檔案並更新 fsx_inress 和 fsx_egress 安全群組變數的 CIDR 範圍。 4. 如有需要，請開啟 variables.tf 檔案並更新這些變數的預設值： • storage_capacity • deployment_type • per_unit_storage_throughput 如需這些變數的說明，請參閱最佳實務一節。	
初始化 Terraform 環境。	若要初始化您的環境以執行 Terraform fsx_deployment 模組，請執行： `terraform init`	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
驗證 Terraform 語法。	若要檢查語法錯誤並確認組態是否正確，請執行： `terraform validate`	AWS DevOps，DevOps 工程師
驗證 Terraform 組態。	若要建立 Terraform 執行計劃並預覽部署，請執行： `terraform plan -var-file terraform.tfvars`	AWS DevOps，DevOps 工程師
部署 Terraform 模組。	若要部署 FSx for Lustre 資源，請執行： `terraform apply -var-file terraform.tfvars`	AWS DevOps，DevOps 工程師

清除 AWS 資源

任務	Description	所需的技能
移除 AWS 資源。	完成使用 FSx for Lustre 環境後，您可以移除 Terraform 部署 AWS 的資源，以避免產生不必要的費用。程式碼儲存庫中提供的 Terraform 模組會自動執行此清除。 1. 在本機儲存庫中，導覽至 `fsx_deployment` 目錄： `cd fsx_deployment` 2. 執行 命令：	AWS DevOps，DevOps 工程師

任務	Description	所需的技能
	`terraform destroy -var-file terraform.tfvars`	

疑難排解

問題	解決方案
FSx for Lustre 傳回錯誤。	如需 FSx for Lustre 問題的協助，請參閱 FSx for Lustre 文件中的疑難排解 Amazon FSx for Lustre。

相關資源

- 使用 Terraform 建置 Amazon FSx for Lustre (Terraform 文件中的AWS 提供者參考）
- Amazon FSx for Lustre 入門 (FSx for Lustre 文件）
- AWS 有關 Amazon FSx for Lustre 的部落格文章

設定 AWS ParallelCluster 的 Grafana 監控儀表板

Dario La Porta 和 William Lu，Amazon Web Services

總結

AWS ParallelCluster 可協助您部署和管理高效能運算 (HPC) 叢集。它支援 AWS Batch 和 Slurm 開放原始碼任務排程器。雖然 AWS ParallelCluster 已與 Amazon CloudWatch 整合，用於記錄和指標，但不會為工作負載提供監控儀表板。

AWS ParallelCluster (GitHub) 的 Grafana 儀表板是 AWS ParallelCluster 的監控儀表板。 GitHub 它在作業系統 (OS) 層級提供任務排程器洞察和詳細監控指標。如需此解決方案中包含之儀表板的詳細資訊，請參閱 GitHub 儲存庫中的範例儀表板。這些指標可協助您進一步了解 HPC 工作負載及其效能。不過，最新版本的 AWS ParallelCluster 或解決方案中使用的開放原始碼套件不會更新儀表板程式碼。此模式可增強解決方案，以提供下列優點：

- 支援 AWS ParallelCluster v3
- 使用最新版本的開放原始碼套件，包括 Prometheus、Grafana、Prometheus Slurm Exporter 和 NVIDIA DCGM-Exporter
- 增加 Slurm 任務使用的 CPU 核心和 GPUs 數量
- 新增任務監控儀表板
- 為具有 4 或 8 個圖形處理單元 (GPUs) 的節點增強 GPU 節點監控儀表板

此增強型解決方案版本已在 AWS 客戶的 HPC 生產環境中實作和驗證。

先決條件和限制

先決條件

- AWS ParallelCluster CLI，已安裝並設定。
- AWS ParallelCluster 支援的網路組態。此模式使用 AWS ParallelCluster 使用兩個子網路組態，需要公有子網路、私有子網路、網際網路閘道和 NAT 閘道。
- 所有 AWS ParallelCluster 叢集節點都必須具有網際網路存取權。這是必要的，以便安裝指令碼可以下載開放原始碼軟體和 Docker 映像。
- Amazon Elastic Compute Cloud (Amazon EC2) 中的金鑰對。具有此金鑰對的資源具有前端節點的安全殼層 (SSH) 存取權。

限制

- 此模式旨在支援 Ubuntu 20.04 LTS。如果您使用的是不同版本的 Ubuntu，或是使用 Amazon Linux 或 CentOS，則需要修改此解決方案隨附的指令碼。這些修改不包含在此模式中。

產品版本

- Ubuntu 20.04 LTS
- ParallelCluster 3.X

帳單和成本考量

- 免費方案不會涵蓋在此模式中部署的解決方案。Amazon EC2、Amazon FSx for Lustre、Amazon VPC 中的 NAT 閘道和 Amazon Route 53 需支付費用。

Architecture

目標架構

下圖顯示使用者如何存取前端節點上 AWS ParallelCluster 的監控儀表板。前端節點執行 NICE DCV、Prometheus、Grafana、Prometheus Slurm Exporter、Prometheus Node Exporter 和 NGINX Open Source。運算節點會執行 Prometheus Node Exporter，如果節點包含 GPUs，也會執行 NVIDIA DCGM-Exporter。前端節點會從運算節點擷取資訊，並在 Grafana 儀表板中顯示該資料。

在大多數情況下，前端節點不會大量載入，因為任務排程器不需要大量的 CPU 或記憶體。使用者在連接埠 443 上使用 SSL 存取前端節點上的儀表板。

所有授權檢視者都可以匿名檢視監控儀表板。只有 Grafana 管理員可以修改儀表板。您可以在 `aws-parallelcluster-monitoring/docker-compose/docker-compose.head.yml` 檔案中設定 Grafana 管理員的密碼。

工具

AWS 服務

- NICE DCV 是一種高效能遠端顯示通訊協定，可協助您在不同的網路條件下，將遠端桌面和應用程式串流從任何雲端或資料中心交付到任何裝置。

- AWS ParallelCluster 可協助您部署和管理高效能運算 (HPC) 叢集。它支援 AWS Batch 和 Slurm 開放原始碼任務排程器。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。

其他工具

- Docker 是一組平台即服務 (PaaS) 產品，可在作業系統層級使用虛擬化在容器中交付軟體。
- Grafana 是一種開放原始碼軟體，可協助您查詢、視覺化、提醒和探索指標、日誌和追蹤。
- NGINX 開放原始碼是開放原始碼 Web 伺服器和反向代理。
- NVIDIA Data Center GPU Manager (DCGM) 是一組工具，可用於管理和監控叢集環境中的 NVIDIA 資料中心圖形處理單元 (GPUs)。在此模式中，您會使用 DCGM-Exporter，這可協助您從 Prometheus 匯出 GPU 指標。
- Prometheus 是一種開放原始碼系統監控工具組，可收集並儲存其指標，做為具有相關鍵值對的時間序列資料，稱為標籤。在此模式中，您也使用 Prometheus Slurm Exporter 收集和匯出指標，並使用 Prometheus Node Exporter 從運算節點匯出指標。
- Ubuntu 是一種開放原始碼的 Linux 作業系統，專為企業伺服器、桌面、雲端環境和 IoT 而設計。

程式碼儲存庫

此模式的程式碼可在 GitHub pcluster-monitoring-dashboard 儲存庫中使用。

史詩

建立必要的資源

任務	Description	所需的技能
建立 S3 儲存貯體。	建立 Amazon S3 儲存貯體。您可以使用此儲存貯體來存放組態指令碼。如需說明，請參閱 Amazon S3 文件中的建立儲存貯體。	一般 AWS

任務	Description	所需的技能
複製儲存庫。	執行下列命令，複製 GitHub pcluster-monitoring-dashboard 儲存庫。 `git clone https://github.com/aws-samples/parallelcluster-monitoring-dashboard.git`	DevOps 工程師
建立管理員密碼。	1. 選擇 `aws-parallelcluster-monitoring` 資料夾，選擇 `docker-compose` 資料夾，然後開啟 docker-compose.head.yml 檔案。 2. 在 `GF_SECURITY_ADMIN_PASSWORD` 變數中，`Grafana4PC!` 以您選擇的密碼取代 。這是您用來管理 Grafana 帳戶的管理密碼。 3. 儲存並關閉 docker-compose.head.yml 檔案。	Linux Shell 指令碼
將必要的檔案複製到 S3 儲存貯體。	將 post_install.sh 指令碼和 aws-parallelcluster-monitoring 資料夾複製到您建立的 S3 儲存貯體。如需說明，請參閱 Amazon S3 文件中的上傳物件。 Amazon S3	一般 AWS

任務	Description	所需的技能
為前端節點設定額外的安全群組。	1. 建立前端節點的安全群組。此安全群組將允許對前端節點上監控儀表板的傳入流量。如需說明，請參閱 Amazon VPC 文件中的建立安全群組。 2. 將傳入規則新增至安全群組。如需說明，請參閱 Amazon VPC 文件中的將規則新增至安全群組。規則請使用下列參數： • 類型 – HTTPS • 通訊協定 – TCP • 連接埠範圍 – 443 • 來源 – 輸入您的 IP 地址 • 描述 – 允許使用者存取監控儀表板	AWS 管理員
設定前端節點的 IAM 政策。	為前端節點建立身分型政策。此政策允許節點從 Amazon CloudWatch 擷取指標資料。GitHub 儲存庫包含範例政策。如需說明，請參閱 AWS Identity and Access Management (IAM) 文件中的建立 IAM 政策。	AWS 管理員

任務	Description	所需的技能
設定運算節點的 IAM 政策。	為運算節點建立身分型政策。此政策允許節點建立包含任務 ID 和任務擁有者的標籤。GitHub 儲存庫包含範例政策。如需說明，請參閱 IAM 文件中的建立 IAM 政策。 如果您使用提供的範例檔案，請取代下列值： • <REGION> – 託管叢集的 AWS 區域 • <ACCOUNT_ID> – AWS 帳戶 ID	AWS 管理員

建立叢集

任務	Description	所需的技能
修改提供的叢集範本檔案。	建立 AWS ParallelCluster 叢集。使用提供的 cluster.yaml AWS CloudFormation 範本檔案作為建立叢集的起點。在提供的範本中取代下列值： • <REGION> – 託管叢集的 AWS 區域。 • <HEADNODE_SUBNET> – VPC 的公有子網路。 • <ADDITIONAL_HEAD_NODE_SG> – 您為前端節點建立的安全群組名稱。 • <KEY_NAME> – 輸入現有 Amazon EC2 金鑰對的名	AWS 管理員

任務	Description	所需的技能
	稱。具有此金鑰對的資源具有前端節點的安全殼層 (SSH) 存取權。 • <ALLOWED_IPS> –輸入允許對前端節點進行 SSH 連線的 CIDR 格式 IP 地址範圍。 • <ADDITIONAL_HEAD_NODE_POLICY> – 輸入您為前端節點建立的 IAM 政策名稱。 • <BUCKET_NAME> – 輸入您建立的 S3 儲存貯體名稱。 • <COMPUTE_SUBNET> – 在 VPC 中輸入私有子網路的名稱。 • <ADDITIONAL_COMPUTE_NODE_POLICY> – 輸入您為運算節點建立的 IAM 政策名稱。	
建立 叢集	在 AWS ParallelCluster CLI 中，輸入下列命令。這會部署 CloudFormation 範本並建立叢集。如需此命令的詳細資訊，請參閱 AWS ParallelCluster 文件中的 pcluster create-cluster。 ParallelCluster `pcluster create-cluster -n <cluster_name> -c cluster.yaml`	AWS 管理員

任務	Description	所需的技能
監控叢集建立。	輸入下列命令來監控叢集建立。如需此命令的詳細資訊，請參閱 AWS ParallelCluster 文件中的 pcluster describe-cluster。 ParallelCluster `pcluster describe-cluster -n <cluster_name>`	AWS 管理員

使用 Grafana 儀表板

任務	Description	所需的技能
存取 Grafana 入口網站。	1. 輸入下列命令以擷取前端節點的公有 IP 地址。 `pcluster describe-cluster -n <cluster_name> --query headNode.publicIpAddress` 2. 在 Web 瀏覽器中，導覽至下列 URL 以存取 Grafana 儀表板。 https://<head_node_public_ip_address> 3. 在 Grafana 首頁上，選擇左側選單上的 4 平方儀表板圖示，然後選擇一般。這會顯示已設定的儀表板清單。Grafana 提供下列儀表板：	AWS 管理員

任務	Description	所需的技能
	• 叢集成本 – 包含叢集成本的相關資訊 • 叢集日誌 – 包含叢集日誌的相關資訊 • 運算節點詳細資訊 – 包含運算節點用量統計資料的相關資訊 • 運算節點清單 – 包含叢集的運算節點清單 • GPU 節點 – 包含 GPU 節點用量統計資料的相關資訊 • 任務詳細資訊 – 包含任務資源使用率的相關資訊 • 頭部節點詳細資訊 – 包含頭部節點用量統計資料的相關資訊 • ParallelCluster 摘要 – 包含叢集用量的相關資訊	

清除解決方案以停止產生相關聯的成本

任務	Description	所需的技能
刪除叢集。	輸入下列命令來刪除叢集。如需此命令的詳細資訊，請參閱 AWS ParallelCluster 文件中的 pcluster delete-cluster。ParallelCluster `pcluster delete-cluster -n <cluster_name>`	AWS 管理員

任務	Description	所需的技能
刪除 IAM 政策。	刪除您為前端節點和運算節點建立的政策。如需刪除政策的詳細資訊，請參閱 IAM 文件中的刪除 IAM 政策。	AWS 管理員
刪除安全群組和規則。	刪除您為前端節點建立的安全群組。如需詳細資訊，請參閱 Amazon VPC 文件中的刪除安全群組規則和刪除安全群組。	AWS 管理員
刪除 S3 儲存貯體。	刪除您建立來存放組態指令碼的 S3 儲存貯體。如需詳細資訊，請參閱 Amazon S3 文件中的刪除儲存貯體。	一般 AWS

疑難排解

問題	解決方案
無法在瀏覽器中存取前端節點。	檢查安全群組並確認傳入連接埠 443 已開啟。
Grafana 未開啟。	在前端節點上，檢查 的容器日誌`docker logs Grafana`。
有些指標沒有資料。	在前端節點上，檢查所有容器的容器日誌。

相關資源

AWS 文件

- 適用於 Amazon EC2 的 IAM 政策

其他 AWS 資源

- AWS ParallelCluster

- AWS ParallelCluster 的監控儀表板 (AWS 部落格文章)

其他資源

- Prometheus 監控系統
- 格拉法納

更多模式

- 使用 K8sGPT 和 Amazon Bedrock 整合實作採用 AI 技術的 Kubernetes 診斷和故障診斷

混合雲端

主題

- 使用 AWS CDK 和 GitLab 在 Amazon ECS Anywhere 上設定混合工作負載的 CI/CD 管道
- 更多模式

使用 AWS CDK 和 GitLab 在 Amazon ECS Anywhere 上設定混合工作負載的 CI/CD 管道

Rafael Ortiz，Amazon Web Services

摘要

Amazon ECS Anywhere 是 Amazon Elastic Container Service (Amazon ECS) 的延伸。它支援向 Amazon ECS 叢集註冊外部執行個體，例如現場部署伺服器或虛擬機器 (VM)。 功能有助於降低成本並減輕複雜的本機容器協同運作和操作。您可以使用 ECS Anywhere 在內部部署和雲端環境中部署和執行容器應用程式。它消除了您的團隊學習多個網域和技能集的需求，或自行管理複雜軟體的需求。

此模式描述使用 step-by-step方法。 Amazon ECS Anywhere 然後，您可以使用 AWS CodePipeline 來設定持續整合和持續部署 (CI/CD) 管道。然後，您將 GitLab 程式碼儲存庫複寫至 AWS CodeCommit，並在 Amazon ECS 叢集上部署容器化應用程式。

此模式旨在協助使用現場部署基礎設施來執行容器應用程式，並使用 GitLab 管理應用程式程式碼庫的人員。您可以使用 AWS 雲端服務來管理這些工作負載，而不會干擾現有的現場部署基礎設施。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在內部部署基礎設施上執行的容器應用程式。
- 您管理應用程式程式碼庫的 GitLab 儲存庫。如需詳細資訊，請參閱儲存庫 (GitLab)。
- 安裝並設定 AWS Command Line Interface (AWS CLI)。如需詳細資訊，請參閱安裝或更新最新版本的 AWS CLI (AWS CLI 文件）。
- AWS CDK Toolkit，全域安裝和設定。如需詳細資訊，請參閱安裝 AWS CDK (AWS CDK 文件）。
- npm，在 TypeScript 中為 AWS CDK 安裝和設定。如需詳細資訊，請參閱下載並安裝 Node.js 和 npm (npm 文件）。

限制

- 如需限制和考量，請參閱 Amazon ECS Anywhere)。

產品版本

- AWS CDK Toolkit 2.27.0 版或更新版本
- npm 7.20.3 版或更新版本
- Node.js 16.6.1 版或更新版本

Architecture

目標技術堆疊

- AWS CDK
- AWS CloudFormation
- AWS CodeBuild
- AWS CodeCommit
- AWS CodePipeline
- Amazon ECS Anywhere
- Amazon Elastic Container Registry (Amazon ECR)
- AWS Identity and Access Management (IAM)
- AWS System Manager
- GitLab 儲存庫

目標架構

此圖表代表此模式中描述的兩個主要工作流程，佈建 Amazon ECS 叢集並設定 CI/CD 管道，以設定和部署 CI/CD 管道，如下所示：

1. 佈建 Amazon ECS 叢集
   a. 當您部署第一個 AWS CDK 堆疊時，它會在 AWS 上建立 CloudFormation 堆疊。
   b. 此 CloudFormation 堆疊會佈建 Amazon ECS 叢集和相關的 AWS 資源。
   c. 若要向 Amazon ECS 叢集註冊外部執行個體，您必須在 VM 上安裝 AWS Systems Manager Agent (SSM Agent)，並將 VM 註冊為 AWS Systems Manager 受管執行個體。
   d. 您還必須在 VM 上安裝 Amazon ECS 容器代理程式和 Docker，以將其註冊為 Amazon ECS 叢集的外部執行個體。
   e. 使用 Amazon ECS 叢集註冊和設定外部執行個體時，它可以在您的 VM 上執行多個容器，其已註冊為外部執行個體。

f. Amazon ECS 叢集處於作用中狀態，可以透過容器執行應用程式工作負載。Amazon ECS Anywhere 容器執行個體會在內部部署環境中執行，但與雲端中的 Amazon ECS 叢集相關聯。

2. 設定和部署 CI/CD 管道

a. 當您部署第二個 AWS CDK 堆疊時，它會在 AWS 上建立另一個 CloudFormation 堆疊。

b. 此 CloudFormation 堆疊會在 CodePipeline 和相關的 AWS 資源中佈建管道。

c. 您可以將應用程式程式碼變更推送並合併到現場部署 GitLab 儲存庫。

d. GitLab 儲存庫會自動複寫至 CodeCommit 儲存庫。

e. CodeCommit 儲存庫的更新會自動啟動 CodePipeline。

f. CodePipeline 從 CodeCommit 複製程式碼，並在 CodeBuild 中建立可部署的應用程式建置。

g. CodePipeline 會建立 CodeBuild 組建環境的 Docker 映像，並將其推送至 Amazon ECR 儲存庫。

h. CodePipeline 會啟動 CodeDeploy 動作，從 Amazon ECR 儲存庫提取容器映像。

i. CodePipeline 在 Amazon ECS 叢集上部署容器映像。

自動化和擴展

此模式使用 AWS CDK 做為基礎設施做為程式碼 (IaC) 工具來設定和部署此架構。AWS CDK 可協助您協調 AWS 資源，並設定 Amazon ECS Anywhere 和 CI/CD 管道。

工具

AWS 服務

- AWS 雲端開發套件 (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端基礎設施。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- Amazon Elastic Container Registry (Amazon ECR) 是一種受管容器映像登錄服務，安全、可擴展且可靠。

- Amazon Elastic Container Service (Amazon ECS) 是快速、可擴展的容器管理服務，可協助您執行、停止和管理叢集上的容器。此模式也使用 Amazon ECS Anywhere，可提供將內部部署伺服器或 VM 註冊至 Amazon ECS 叢集的支援。

其他工具

- Node.js 是一種事件驅動的 JavaScript 執行期環境，旨在建置可擴展的網路應用程式。
- npm 是在 Node.js 環境中執行的軟體登錄檔，用於共用或借用套件和管理私有套件的部署。
- Vagrant 是一種開放原始碼公用程式，用於建置和維護可攜式虛擬軟體開發環境。基於示範目的，此模式使用 Vagrant 來建立內部部署 VM。

程式碼儲存庫

此模式的程式碼可在 Amazon ECS Anywhere 的 GitHub CI/CD 管道中使用 AWS CDK 儲存庫。Amazon ECS Anywhere

最佳實務

部署此模式時，請考慮下列最佳實務：

- 使用 AWS CDK 開發和部署雲端基礎設施的最佳實務
- 使用 AWS CDK 開發雲端應用程式的最佳實務 (AWS 部落格文章)

史詩

驗證 AWS CDK 組態

任務	Description	所需的技能
驗證 AWS CDK 版本。	輸入下列命令來驗證 AWS CDK Toolkit 的版本。 `cdk --version` 此模式需要 2.27.0 版或更新版本。如果您有較舊的版本，請	DevOps 工程師

任務	Description	所需的技能
	遵循 AWS CDK 文件中的指示進行更新。	
驗證 npm 版本。	輸入下列命令來驗證 npm 的版本。 `npm --version` 此模式需要 7.20.3 版或更新版本。如果您有較舊的版本，請遵循 npm 文件中的指示進行更新。	DevOps 工程師

設定 AWS 登入資料。	輸入 `aws configure` 命令並遵循提示來設定 AWS 登入資料。
	`$aws configure`
	`AWS Access Key ID [None]: <your-access-key-ID>`
	`AWS Secret Access Key [None]: <your-secret-access-key>`
	`Default region name [None]: <your-Region-name>`
	`Default output format [None]:`
		DevOps 工程師

引導 AWS CDK 環境

任務	Description	所需的技能
複製 AWS CDK 程式碼儲存庫。	1. 輸入下列命令，針對此模式使用 AWS CDK 儲存庫複	DevOps 工程師

任務	Description	所需的技能

	製 Amazon ECS Anywhere 的 CI/CD 管道。
	`git clone https://github.com/aws-samples/amazon-ecs-anywhere-cicd-pipeline-cdk-sample.git`
	2. 輸入下列命令，導覽至複製的目錄。
	`cd amazon-ecs-anywhere-cicd-pipeline-cdk-sample`	

引導環境。	輸入下列命令，將 CloudFormation 範本部署到您想要使用的帳戶和 AWS 區域。 `cdk bootstrap <account-number>/<Region>` 如需詳細資訊，請參閱 AWS CDK 文件中的引導。	DevOps 工程師

建置和部署 Amazon ECS Anywhere 的基礎設施

任務	Description	所需的技能

安裝套件相依性並編譯 TypeScript 檔案。	輸入下列命令來安裝套件相依性並編譯 TypeScript 檔案。
	`$cd EcsAnywhereCdk`
	`$npm install`
	`$npm fund`
		DevOps 工程師

任務	Description	所需的技能
	這些命令會從範例儲存庫安裝所有套件。如需詳細資訊，請參閱 npm 文件中的 npm ci 和 npm 安裝。如果您在輸入這些命令時收到有關遺失套件的任何錯誤，請參閱此模式的故障診斷一節。	
建置專案。	若要建置專案程式碼，請輸入下列命令。 `npm run build` 如需建置和部署專案的詳細資訊，請參閱 AWS CDK 文件中的您的第一個 AWS CDK 應用程式。	DevOps 工程師

部署 Amazon ECS Anywhere 基礎設施堆疊。	1. 輸入下列命令來列出堆疊。
	`$cdk list`
	2. 確認輸出傳回 `EcsAnywhereInfraStack` 和 `ECSAnywherePipelineStack` 堆疊。
	3. 輸入下列命令來部署 `EcsAnywhereInfraStack` 堆疊。
	`$cdk deploy EcsAnywhereInfraStack`
		DevOps 工程師

任務	Description	所需的技能
驗證堆疊建立和輸出。	1. 登入 AWS 管理主控台，然後開啟位於 https://console.aws.amazon.com/cloudformation/ 的 CloudFormation 主控台。 2. 在堆疊頁面上，選取`EcsAnywhereInfraStack` 堆疊。 3. 確認堆疊狀態為 `CREATE_IN_PROGRESS` 或 `CREATE_COMPLETE` 。 設定 Amazon ECS 叢集可能需要一些時間。在堆疊建立完成之前，請勿繼續。	DevOps 工程師

設定內部部署 VM

任務	Description	所需的技能
設定您的 VM。	從 Vagrantfile 所在的根目錄輸入 `vagrant up`命令，以建立 Vagrant VM。如需詳細資訊，請參閱 Vagrant 文件。	DevOps 工程師
將您的 VM 註冊為外部執行個體。	1. 使用 `vagrant ssh`命令登入 Vagrant VM。如需詳細資訊，請參閱 Vagrant 文件。 2. 遵循 AWS CLI 安裝說明，並輸入下列命令，在 VM 上安裝 AWS CLI。	DevOps 工程師

任務	Description	所需的技能

	``
	$ curl "https://
	awscli.amazonaws.c
	om/awscli-exe-linu
	x-x86_64.zip" \
	> -o "awscliv2.zip"
	$sudo apt install
	 unzip
	$unzip awscliv2.zip
	$sudo ./aws/install
	$aws configure
	AWS Access Key ID
	 [None]: <your-acc
	ess-key-ID>
	AWS Secret Access Key
	 [None]: <your-sec
	ret-access-key>
	Default region name
	 [None]: <your-Reg
	ion-name>
	Default output format
	 [None]:
	``
	
	1. 建立啟用代碼和 ID，可用來向 AWS Systems Manager 註冊 VM，以及啟用外部執行個體。此命令的輸出包含啟用 ID 和啟用代碼值。
	
	``
	aws ssm create-ac
	tivation \
	> --iam-role
	 EcsAnywhereInstanc
	eRole \
	> | tee ssm-activ
	ation.json
	``	

任務	Description	所需的技能

	如果您在執行此命令時收到錯誤，請參閱疑難排解一節。
	2. 匯出啟用 ID 和程式碼值。
	`export ACTIVATION_ID=<activation-ID>`
	`export ACTIVATION_CODE=<activation-code>`
	3. 將安裝指令碼下載到您的 VM。
	`curl --proto "https" -o "ecs-anywhere-install.sh" \`
	`> "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-install-latest.sh"`
	4. 在 VM 上執行安裝指令碼。
	`sudo bash ecs-anywhere-install.sh \`
	`--cluster EcsAnywhereCluster \`
	`--activation-id $ACTIVATION_ID \`
	`--activation-code $ACTIVATION_CODE \`
	`--region <region-name>`	

任務	Description	所需的技能
	這會設定您的 VM 是 Amazon ECS Anywhere 外部執行個體，並在 Amazon ECS 叢集中註冊執行個體。如需詳細資訊，請參閱 Amazon ECS 文件中的向叢集註冊外部執行個體。如果您遇到任何問題，請參閱故障診斷一節。	

驗證 Amazon ECS Anywhere 和外部 VM 的狀態。	若要驗證您的 VM 是否已連線至 Amazon ECS 控制平面並執行，請使用下列命令。
	`$aws ssm describe-instance-information`
	`$aws ecs list-container-instances --cluster $CLUSTER_NAME`
		DevOps 工程師

部署 CI/CD 管道

任務	Description	所需的技能

在 CodeCommit 儲存庫中建立分支。	透過建立儲存庫的第一個遞交，在 CodeCommit 儲存庫`main`中建立名為 的分支。您可以依照 AWS 文件在 CodeCommit 中建立遞交。下列是範例命令。
	`aws codecommit put-file \`
	`--repository-name EcsAnywhereRepo \`
	`--branch-name main \`
		DevOps 工程師

任務	Description	所需的技能
	`--file-path README.md \ --file-content "Test" \ --name "Dev Ops" \ --email "devops@example.com" \ --commit-message "Adding README."`	
設定儲存庫鏡像。	您可以在外部來源之間鏡像 GitLab 儲存庫。您可以選擇哪個儲存庫做為來源。分支、標籤和遞交會自動同步。在託管應用程式的 GitLab 儲存庫與 CodeCommit 儲存庫之間設定推送鏡。如需說明，請參閱設定從 GitLab 到 CodeCommit 的推播鏡 (GitLab 文件）。 **Note** 根據預設，鏡像會自動同步儲存庫。如果您想要手動更新儲存庫，請參閱更新鏡像 (GitLab 文件）。	DevOps 工程師
部署 CI/CD 管道堆疊。	輸入下列命令來部署 `EcsAnywherePipelineStack` 堆疊。 `$cdk deploy EcsAnywherePipelineStack`	DevOps 工程師

任務	Description	所需的技能
測試 CI/CD 管道。	1. 變更應用程式程式碼，並將其推送至來源內部部署 GitLab 儲存庫。例如，編輯 `../application/index.html` 檔案以更新應用程式版本值。 2. 當程式碼複寫到 CodeCommit 儲存庫時，這會啟動 CI/CD 管道。執行以下任意一項： • 如果您使用自動鏡射來同步 GitLab 儲存庫與 CodeCommit 儲存庫，請繼續下一個步驟。 • 如果您使用手動鏡像，請依照更新鏡像 (GitLab 文件) 中的指示，將應用程式程式碼變更推送至 CodeCommit 儲存庫。 3. 在本機電腦上，於 Web 瀏覽器中輸入 http://localhost:80。這會開啟 NGINX 網頁，因為連接埠 80 會轉送至 Vagrantfile 中的 localhost。確認您可以檢視更新的應用程式版本值。這會驗證管道和映像部署。 4. （選用）如果您想要在 AWS 管理主控台中驗證部署，請執行下列動作： a. 開啟 Amazon ECS 主控台，網址為 https://	DevOps 工程師

任務	Description	所需的技能
	console.aws.amazon.com/ecs/。 b. 從導覽列中選取要使用的「區域」。 c. 在導覽窗格中，選擇叢集。 d. 在叢集頁面上，選取 EcsAnywhereCluster 叢集。 e. 選擇任務定義。 f. 確認容器正在執行。	

清除

任務	Description	所需的技能

清除和刪除資源。	逐步解說此模式之後，您應該移除您建立的proof-of-concept資源。若要清除，請輸入下列命令。
	`$cdk destroy EcsAnywherePipelineStack`
	`$cdk destroy EcsAnywhereInfraStack`
		DevOps 工程師

疑難排解

問題	解決方案
安裝套件相依性時遺失套件的錯誤。	輸入下列其中一個命令來解決遺失的套件。

問題	解決方案

	`$npm ci`
	
	或
	
	`$npm install -g @aws-cdk/<package_name>`

當您在 VM 上執行 `aws ssm create-activation` 命令時，您會收到下列錯誤。 `An error occurred (ValidationException) when calling the CreateActivation operation: Nonexistent role or missing ssm service principal in trust policy: arn:aws:iam::000000000000:role/EcsAnywhereInstanceRole`	`EcsAnywhereInfraStack` 堆疊尚未完全部署，且執行此命令所需的 IAM 角色尚未建立。在 CloudFormation 主控台中檢查堆疊狀態。在狀態變更為 後重試命令`CREATE_COMPLETE` 。

Amazon ECS 運作狀態檢查會傳回 `UNHEALTHY` ，而您會在 Amazon ECS 主控台中叢集的服務區段中看到下列錯誤。
 	
`service EcsAnywhereService was unable to place a task because no container instance met all of its requirements. Reason: No Container Instances were found in your cluster.`
	輸入下列命令，在您的 Vagrant VM 上重新啟動 Amazon ECS 代理程式。
	
	`$vagrant ssh`
	`$sudo systemctl restart ecs`
	`$sudo systemctl status ecs`

相關資源

- Amazon ECS Anywhere 行銷頁面
- Amazon ECS Anywhere 文件
- Amazon ECS Anywhere 示範 （影片）

- Amazon ECS Anywhere 研討會範例 (GitHub)
- 儲存庫鏡像 (GitLab 文件)

更多模式

- 使用 AWS Transit Gateway 自動化區域間對等互連的設定
- 在做為 Docker 容器 AWS IoT Greengrass V2 執行時部署容器化應用程式
- 使用 AWS CDK 設定 Amazon ECS Anywhere 來管理內部部署容器應用程式
- 當您從 F5 遷移到 AWS 上的 Application Load Balancer 時修改 HTTP 標頭
- 在 AWS 雲端中重新託管內部部署工作負載：遷移檢查清單
- 使用 BMC Discovery 查詢來擷取遷移資料以進行遷移規劃

管理與治理

主題

- 當 Amazon Data Firehose 資源未使用 AWS KMS 金鑰加密時，識別和提醒
- AWS 帳戶 使用 AFT 自動化新 的 Amazon VPC IPAM IPv4 CIDR 配置
- 使用 AWS Systems Manager 自動化新增或更新 Windows 登錄項目
- 使用 Python 在 AMS 中自動建立 RFC
- 使用 AWS Systems Manager 維護 Windows 自動停止和啟動 Amazon RDS 資料庫執行個體
- 使用 Terraform 集中 AWS Organizations 中的軟體套件分佈
- 使用 NLog 在 Amazon CloudWatch Logs 中設定 .NET 應用程式的記錄
- 將 AWS Service Catalog 產品複製到不同的 AWS 帳戶和 AWS 區域
- 為雲端操作模型建立 RACI 或 RASCI 矩陣
- 使用 Amazon CloudWatch 異常偵測為自訂指標建立警示
- 建立使用具有預設加密之 Amazon EBS 磁碟區的 AWS Cloud9 IDE
- 自動建立標籤型 Amazon CloudWatch 儀表板
- 記錄您的 AWS 登陸區域設計
- 使用 AWS CDK 跨多個 AWS 區域、帳戶和 OUs 啟用 Amazon DevOps Guru，以改善營運效能
- 使用 Account Factory for Terraform 管理多個帳戶的許可集
- 使用引導管道實作 Account Factory for Terraform (AFT)
- 在多個 AWS 帳戶和 AWS 區域中管理 AWS Service Catalog 產品
- 使用 AWS 服務監控 SAP RHEL Pacemaker 叢集
- 使用 CloudWatch Logs Insights 監控應用程式活動
- 跨多個 監控共用 Amazon Machine Image 的使用 AWS 帳戶
- 檢視 AWS 帳戶或組織的 EBS 快照詳細資訊
- 更多模式

當 Amazon Data Firehose 資源未使用 AWS KMS 金鑰加密時，識別和提醒

Ram Kandaswamy，Amazon Web Services

總結

為了合規，某些組織必須在 Amazon Data Firehose 等資料交付資源上啟用加密。此模式顯示一種方法來監控、偵測和通知資源不合規的情況。

為了維持加密需求，此模式可用於 ， AWS 以自動監控和偵測未以 AWS Key Management Service (AWS KMS) 金鑰加密的 Amazon Data Firehose 交付資源。解決方案會傳送提醒通知，並且可以延伸以執行自動修復。此解決方案可套用至個別帳戶或多帳戶環境，例如使用 AWS 登陸區域 或 的環境 AWS Control Tower。

先決條件和限制

先決條件

- Amazon Data Firehose 交付串流
- 有足夠的許可和熟悉度 CloudFormation，用於此基礎設施自動化

限制

- 解決方案不是即時的，因為它使用 AWS CloudTrail 事件進行偵測，而且在建立未加密資源和傳送通知之間存在延遲。

Architecture

目標技術堆疊

解決方案使用無伺服器技術和下列服務：

- AWS CloudTrail
- Amazon CloudWatch
- AWS Command Line Interface (AWS CLI)
- AWS Identity and Access Management (IAM)
- Amazon Data Firehose
- AWS Lambda

- Amazon Simple Notification Service (Amazon SNS)

目標架構

圖表說明這些步驟：

1. 使用者建立或修改 Amazon Data Firehose。
2. 偵測到並比對 CloudTrail 事件。
3. 叫用 Lambda。
4. 識別不合規的資源。
5. 系統會傳送電子郵件通知。

自動化和擴展

您可以使用 CloudFormation StackSets，透過單一命令將此解決方案套用至多個 AWS 區域 或 帳戶。

工具

- AWS CloudTrail 是 AWS 服務 ，可協助您啟用 的控管、合規，以及操作和風險稽核 AWS 帳戶。使用者、角色或 採取的動作 AWS 服務 會在 CloudTrail 中記錄為事件。事件包括在 AWS 管理主控台、 AWS CLI、 AWS SDKs和 API 操作中採取的動作。
- Amazon CloudWatch Events 提供近乎即時的系統事件串流，描述 AWS 資源的變更。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可讓您在命令列 Shell 中使用命令 AWS 服務 與 互動。
- AWS Identity and Access Management (IAM) 是一種 Web 服務，可協助您安全地控制對 AWS 資源的存取。您可以使用 IAM 來控制能通過身分驗證 (登入) 和授權使用資源的 (具有許可) 的人員。
- Amazon Data Firehose 是一種全受管服務，可提供即時串流資料。使用 Firehose，您不需要撰寫應用程式或管理資源。將您的資料產生來源設定為把資料傳送至 Firehose，它就會將資料自動交付至您指定的目的地。
- AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。您只需為使用的運算時間付費，程式碼未執行時無需付費。
- Amazon Simple Notification Service (Amazon SNS) 是一種受管服務，可將訊息從發佈者交付給訂閱者 （也稱為生產者和消費者）。

史詩

強制加密以符合規範

任務	Description	所需的技能
Deploy CloudFormation StackSets。	在 中 AWS CLI，執行下列命令，使用 `firehose-encryption-checker.yaml` 範本（已連接）來建立堆疊集。 為 參數提供有效的 Amazon SNS 主題 Amazon Resource Name (ARN)。部署應該成功建立 CloudWatch Events 規則、Lambda 函數，以及具有必要許可的 IAM 角色，如範本中所述。 `aws cloudformation create-stack-set --stack-set-name my-stack-set --template-body file://firehose-encryption-checker.yaml`	雲端架構師、系統管理員
建立堆疊執行個體。	您可以在 AWS 區域 您選擇的 以及一或多個帳戶中建立堆疊。 若要建立堆疊執行個體，請執行下列命令。將堆疊名稱、帳戶號碼和區域取代為您自己的 。 `aws cloudformation create-stack-instances --stack-set-name my-stack-set --accounts 123456789012 223456789012 --`	雲端架構師、系統管理員

任務	Description	所需的技能
	`regions us-east-1 us-east-2 us-west-1 us-west-2 --operation-preferences FailureToleranceCount=1`	

相關資源

- 使用 CloudFormation StackSets
- 什麼是 Amazon CloudWatch Events？

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

AWS 帳戶 使用 AFT 自動化新 的 Amazon VPC IPAM IPv4 CIDR 配置

Kien Pham 和 Alex Pazik，Amazon Web Services

總結

此模式說明如何 AWS 帳戶 使用 AWS Control Tower Account Factory for Terraform (AFT) 自動化新的 Amazon VPC IP Address Manager (IPAM) IPv4 CIDR 配置。這是使用帳戶層級自訂來完成的，該自訂會使用 `aft-account-customizations`模組，將 IPv4 CIDR 區塊從 IPAM 配置到新的虛擬私有雲端 (VPC)。

使用 IPAM，您可以大規模組織、指派、監控和稽核 IP 地址，讓您輕鬆地規劃、追蹤和監控 AWS 工作負載的 IP 地址。您可以建立 IPAM 和 IPAM 集區，以在帳戶販賣程序期間用來將 IPv4 CIDR 區塊配置給新的 VPC。

先決條件和限制

先決條件

- 在支援的 AWS 區域和部署的 AFT 中 AWS Control Tower 啟用 AWS 帳戶 的作用中
- 支援的版本控制系統 (VCS) 供應商，例如 BitBucket、GitHub 和 GitHub Enterprise
- 已安裝 Terraform 命令列界面 (CLI)
- 執行期環境，您可以在其中執行安裝 AFT 的 Terraform 模組
- AWS Command Line Interface (AWS CLI) 已安裝並設定

限制

- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

產品版本

- AWS Control Tower 登陸區域 3.0 版或更新版本，早於 4.0 版
- AFT 1.13.0 版或更新版本，早於 2.0.0 版
- Terraform OSS 1.2.0 版或更新版本，早於 2.0.0 版
- Terraform AWS Provider (`terraform-provider-aws`) 5.11.0 版或更新版本，早於 6.0.0 版
- IPAM () 2.1.0 版或更新版本的 Terraform 模組 `aws-ia/ipam/aws`

Architecture

下圖顯示此模式的工作流程和元件。

工作流程包含下列主要任務：

1. 觸發器變更 – Terraform 和 IPAM 自訂的變更會遞交至 GitHub 儲存庫並推送。此任務會自動觸發 AWS CodeBuild 管道。
2. 自動化建置 – 在 CodeBuild 中，會觸發多個建置專案 AWS Step Functions。
3. 套用自訂 – Step Functions 與 CodeBuild 協調，以規劃和套用 Terraform 變更。此任務使用 AFT Terraform 模組來協調 AWS 佈建帳戶的 IPAM 集區 IP 指派。

工具

AWS 服務

- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- AWS Control Tower 協調其他數個 的功能 AWS Organizations，AWS 服務包括 AWS Service Catalog和 AWS IAM Identity Center。它可協助您設定和管理 AWS 多帳戶環境，並遵循規範性最佳實務。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- 適用於 Python (Boto3) 的 AWS SDK 是一種軟體開發套件，可協助您整合 Python 應用程式、程式庫或指令碼 AWS 服務。
- AWS Service Catalog 可協助您集中管理已核准的 IT 服務目錄 AWS。最終使用者可在機構所設的限制範圍內，迅速地只部署自己需要且經核准的 IT 服務。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務 來建置業務關鍵型應用程式。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優

勢。Amazon VPC IP Address Manager (IPAM) 是一種 VPC 功能，可讓您更輕鬆地規劃、追蹤和監控 AWS 工作負載的 IP 地址。

其他工具

- GitHub 是一種開發人員平台，開發人員可用來建立、存放、管理和共用程式碼。
- HashiCorp Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。這包括低階元件，例如運算執行個體、儲存和聯網，以及高階元件，例如 DNS 項目和軟體服務 (SaaS) 功能。
- Python 是一種一般用途的電腦程式設計語言。您可以使用它在 上建置應用程式、自動化任務和開發服務AWS 雲端。

程式碼儲存庫

- 此模式的程式碼可在 GitHub AWS Control Tower Account Factory for Terraform 儲存庫中取得。

最佳實務

部署 AFT 時，建議您遵循最佳實務，以協助確保安全、有效率且成功的實作。實作和操作 AFT 的重要準則和建議包括下列項目：

- 徹底檢閱輸入 – 仔細檢閱並了解每個輸入。正確的輸入組態對於 AFT 的設定和運作至關重要。
- 定期範本更新 – 使用最新 AWS 功能和 Terraform 版本來更新範本。定期更新可協助您利用新功能並維護安全性。
- 版本控制 – 固定您的 AFT 模組版本，並盡可能使用個別的 AFT 部署進行測試。
- 範圍 – 僅使用 AFT 來部署基礎設施護欄和自訂。請勿使用它來部署您的應用程式。
- 內嵌和驗證 – AFT 管道需要內嵌和驗證的 Terraform 組態。在將組態推送至 AFT 儲存庫之前，執行 lint、驗證和測試。
- Terraform 模組 – 建置可重複使用的 Terraform 程式碼做為模組，並一律指定 Terraform 和 AWS 提供者版本以符合組織的需求。

史詩

設定您的 AWS 環境

任務	Description	所需的技能
部署 AWS Control Tower。	AWS Control Tower 在您的 AWS 環境中設定和配置，以確保集中管理和控管您的 AWS 帳戶。如需詳細資訊，請參閱 AWS Control Tower 文件中的入門 AWS Control Tower。	雲端管理員
部署 AWS Control Tower Account Factory for Terraform (AFT)。	在新的專用 AFT 管理帳戶中設定 AFT。如需詳細資訊，請參閱 AWS Control Tower 文件中的 Configure 和啟動 AWS Control Tower Account Factory for Terraform。	雲端管理員
完成 AFT 部署後。	AFT 基礎設施部署完成後，請完成 AWS Control Tower 文件中的部署後步驟。	雲端管理員

建立 IPAM

任務	Description	所需的技能
委派 IPAM 管理員。	若要委派 AWS 組織中的 IPAM 管理員帳戶，請使用下列步驟： 1. 使用 AWS Organizations 管理帳戶，在 https：//https://console.aws.amazon.com/ipam/ 開啟 IPAM 主控台。	AWS 管理員

任務	Description	所需的技能
	2. 在 中 AWS 管理主控台，選擇您要在 AWS 區域 其中使用 IPAM 的 。 3. 在導覽窗格中，選擇 Organization settings (組織設定)。 4. 選擇委派。委派選項只有在您以 AWS Organizations 管理帳戶身分登入 主控台時才能使用。 5. 輸入 IPAM 帳戶的 AWS 帳戶 ID。IPAM 管理員必須位於 AWS Organizations 成員帳戶中。 6. 選擇儲存變更。 或者，您可以使用 AWS CLI 並執行下列命令： `aws ec2 enable-ipam-organization-admin-account \ --delegated-admin-account-id 012345678901` 如需詳細資訊，請參閱《Amazon VPC 文件》中的將 IPAM 與 AWS 組織中的帳戶整合，以及《 AWS CLI 命令參考》中的 enable-ipam-organization-admin-account。	

任務	Description	所需的技能
	Important 若要繼續使用 IPAM，您必須登入委派管理員帳戶。下一個步驟中指定的 SSO 設定檔或 AWS 環境變數必須允許您登入該帳戶，並授予建立 IPAM 頂層和區域集區的許可。	

任務	Description	所需的技能

建立 IPAM 頂層和區域集區。	此模式的 GitHub 儲存庫包含 Terraform 範本，可用來建立 IPAM 頂層集區和區域集區。然後，您可以使用 AWS Resource Access Manager () 與組織、組織單位 (OU) AWS 帳戶或其他資源共用集區AWS RAM。
	
	使用下列步驟：
	
	1. 執行下列命令：
	
	`# Navigate to the IPAM module`
	`cd ipam/terraform`
	
	`# Initiate the IPAM module`
	`terraform init`
	
	2. 開啟 `main.tf` 檔案，依照下列註解所述修改程式碼，然後儲存檔案。
	
	`terraform { required_providers { aws = { source = "hashicorp/aws" version = ">= 5.11.0, < 6.0.0" } } }`
	
	`provider "aws" {`
		AWS 管理員

任務	Description	所需的技能

	<pre> region = "us-east-1" # Replace with the region to create your top-level pool
	}
	
	module "ipam" {
	  source = "aws-ia/ipam/aws"
	  version = ">= 2.1.0"
	
	  top_name = "my-top-level-pool" # Replace with your desired top-level pool name
	  top_description = "top-level pool" # Replace with your desired top-level level pool description
	  top_cidr = ["10.0.0.0/16"] # Replace with your desired top-level pool CIDR
	
	  pool_configurations = {
	    my-regional-pool = { # (Optional) Replace with a different resource name
	      name = "my-regional-pool" # Replace with your desired pool name
	      description = "regional pool" # Replace with your desired pool description</pre>	

任務	Description	所需的技能

	<pre>cidr
	= ["10.0.0.0/23"]
	# Replace with your
	desired pool CIDR
	locale
	= "us-east-1"
	# Replace with your
	desired pool locale
	ram_share
	_principals =
	["arn:aws:organiza
	tions::01234567890
	1:ou/ou-ab1c2de345
	/ou-ab1c2de345"]
	# Replace with your
	desired principal
	ARN to share with
	via Resource Access
	Manager (RAM)
	}
	}
	}
	
	output "my_regio
	nal_pool_id" {
	description = "The
	ID of the regional
	pool"
	value =
	module.ipam.pools_
	level_1["my-region
	al-pool"].id #
	Replace with your
	desired resource name
	if changed above
	}
	
	# Create the IPAM
	top-level and
	resource pool
	terraform apply</pre>	

任務	Description	所需的技能
	記下建立後輸出的資源集區 ID。提交帳戶請求時，您將需要 ID。如果您忘記資源集區 ID，可以稍後從 取得 AWS 管理主控台。 ⚠ Important 請確定建立的集區 CIDRs 不會與您工作區域中的任何其他集區重疊。您可以在沒有 CIDR 的情況下建立集區，但是在為其佈建 CIDR 之前，您將無法使用集區進行配置。您可以隨時編輯集區，將 CIDR 新增至集區。	

整合 IPAM 與 AFT

任務	Description	所需的技能

開始建立帳戶自訂。	若要開始新的帳戶自訂，請從終端機執行下列命令：
	
	`# Default name for customization repo`
	`cd aft-account-customizations # Replace with your actual repo name if different than the default`
	`mkdir -p APG-AFT-IPAM/terraform # Replace APG-AFT-IPAM with your`
		DevOps 工程師

任務	Description	所需的技能
	desired customization name cd APG-AFT-IPAM/terraform	

建立 aft-providers.jinja 檔案。	將動態程式碼新增至 檔案aft-providers.jinja ，指定要使用的 Terraform 後端和提供者。
	
	使用下列步驟：
	
	1. 在 terraform 目錄中建立新的aft-providers.jinja 檔案。
	2. 開啟 檔案，新增下列程式碼，然後儲存檔案。
	
	provider "aws" {
	region = "{{ provider_region }}"
	assume_role {
	role_arn = "{{ target_admin_role_arn }}"
	}
	default_tags {
	tags = {
	managed_by = "AFT"
	}
	}
	}
		DevOps 工程師

任務	Description	所需的技能

建立 backend.jinja 檔案。	將動態程式碼新增至 檔案backend.jinja ，指定要使用的 Terraform 後端和提供者。
	
	使用下列步驟：
	
	1. 在 terraform 目錄中建立新的backend.jinja 檔案。
	2. 開啟 檔案，新增下列程式碼，然後儲存檔案。
	
	``
	## Auto generated
	 backend.tf ##
	## Updated on:
	 {{ timestamp }} ##
	
	{% if tf_distri
	bution_type == "oss"
	 -%}
	terraform {
	  required_version
	 = ">= {{ tf_versio
	n }}"
	  backend "s3" {
	    region =
	 "{{ region }}"
	    bucket =
	 "{{ bucket }}"
	    key =
	 "{{ key }}"
	    dynamodb_table
	 = "{{ dynamodb_
	table }}"
	    encrypt =
	 "true"
	    kms_key_id =
	 "{{ kms_key_id }}"
	``
		DevOps 工程師

任務	Description	所需的技能

	role_arn
	= "{{ aft_admin
	_role_arn }}"
	}
	}
	{% else -%}
	terraform {
	backend "remote" {
	organizat
	ion = "{{ terraform
	_org_name }}"
	workspaces {
	name =
	"{{ terraform
	_workspace_name }}"
	}
	}
	}
	{% endif %}	

任務	Description	所需的技能

建立 main.tf 檔案。	建立新的main.tf檔案並新增程式碼，以定義從 AWS Systems Manager (aws_ssm) 擷取兩個值並建立 VPC 的兩個資料來源。
	
	使用下列步驟：
	
	1. 在 terraform 目錄中建立新的main.tf檔案。
	2. 開啟 檔案，新增下列程式碼，然後儲存檔案。
	
	``
	# Define data sources
	data "aws_ssm_
	parameter" "vpc_ipam
	_id" {
	  name = "/aft/acc
	ount-request/custo
	m-fields/vpc-
	ipam-id" # Value
	 is defined in the
	 account-request.tf
	 file
	}
	
	data "aws_ssm_
	parameter" "vpc_netm
	ask" {
	  name = "/aft/acc
	ount-request/custo
	m-fields/vpc-ipam-
	netmask" # Value
	 is defined in the
	 account-request.tf
	 file
	}
	
	# Create new VPC
	``
		DevOps 工程師

任務	Description	所需的技能

	resource "aws_vpc" "vpc1" {
	
	ipv4_ipam_pool_id = data.aws_ssm_parameter.vpc_ipam_id.value # Retrieved from SSM - this is how we integrate with IPAM
	ipv4_netmask_length = data.aws_ssm_parameter.vpc_netmask.value # Retrieved from SSM
	
	assign_generated_ipv6_cidr_block = var.enable_ipv6 ? true : null
	ipv6_cidr_block = var.ipv6_cidr
	ipv6_ipam_pool_id = var.ipv6_ipam_pool_id
	ipv6_netmask_length = var.ipv6_netmask_length
	ipv6_cidr_block_network_border_group = var.ipv6_cidr_block_network_border_group
	
	instance_tenancy = var.instance_tenancy
	enable_dns_hostnames =	

任務	Description	所需的技能

	<pre>var.enable_dns_hostnames
	enable_dns_support = var.enable_dns_support
	enable_network_address_usage_metrics = var.enable_network_address_usage_metrics
	
	tags = var.tags
	
	lifecycle {
	ignore_changes = [
	tags, # Any changes made to VPC tags after creation will not be overwritten - remove to revert these changes during future 'terraform apply' operations
	]
	}
	}</pre>	

任務	Description	所需的技能

建立 variables.tf 檔案。	建立宣告 Terraform 模組所用變數variables.tf 的檔案。
	
	使用下列步驟：
	
	1. 在 terraform 目錄中建立新的variables.tf 檔案。
	2. 開啟 檔案，新增下列程式碼，然後儲存檔案。
	
	`# Copied from AWS VPC module`
	`# https://github.com/terraform-aws-modules/terraform-aws-vpc/blob/master/variables.tf`
	
	`variable "name" {`
	`  description = "Name to be used on all the resources as identifier"`
	`  type        = string`
	`  default     = ""`
	`}`
	
	`variable "enable_ipv6" {`
	`  description = "Requests an Amazon-provided IPv6 CIDR block with a /56 prefix length for the VPC. You cannot specify the range of IP addresses, or`
		DevOps 工程師

任務	Description	所需的技能

	the size of the CIDR block"
	type = bool
	default = false
	}
	
	variable "ipv6_cidr" {
	description = "(Optional) IPv6 CIDR block to request from an IPAM Pool. Can be set explicitly or derived from IPAM using `ipv6_netmask_length`"
	type = string
	default = null
	}
	
	variable "ipv6_ipam_pool_id" {
	description = "(Optional) IPAM Pool ID for a IPv6 pool. Conflicts with `assign_generated_ipv6_cidr_block`"
	type = string
	default = null
	}
	
	variable "ipv6_netmask_length" {
	description = "(Optional) Netmask length to request from IPAM Pool. Conflicts with `ipv6_cidr_block`. This can be omitted if IPAM pool as a	

任務	Description	所需的技能

	`allocation_default_netmask_length` set. Valid values: `56`"
	type = number
	default = null
	}
	
	variable "ipv6_cidr_block_network_border_group" {
	description = "By default when an IPv6 CIDR is assigned to a VPC a default ipv6_cidr_block_network_border_group will be set to the region of the VPC. This can be changed to restrict advertisement of public addresses to specific Network Border Groups such as LocalZones"
	type = string
	default = null
	}
	
	variable "instance_tenancy" {
	description = "A tenancy option for instances launched into the VPC"
	type = string
	default = "default"
	}
	
	variable "enable_dns_hostnames" {	

任務	Description	所需的技能

	<pre> description =
	 "Should be true to
	 enable DNS hostnames
	 in the VPC"
	  type = bool
	  default = true
	}
	
	variable "enable_d
	ns_support" {
	  description =
	 "Should be true to
	 enable DNS support in
	 the VPC"
	  type = bool
	  default = true
	}
	
	variable "enable_n
	etwork_address_usa
	ge_metrics" {
	  description =
	 "Determines whether
	 network address usage
	 metrics are enabled
	 for the VPC"
	  type = bool
	  default = null
	}
	
	variable "tags" {
	  description = "A
	 map of tags to add to
	 all resources"
	  type =
	 map(string)
	  default = {}
	}</pre>	

任務	Description	所需的技能

建立 terraform.tfvars 檔案。	建立 terraform.tfvars 檔案，定義傳遞至 main.tf 檔案的變數值。
	
	使用下列步驟：
	
	1. 在 terraform 目錄中建立新的terraform .tfvars 檔案。
	2. 開啟 檔案，新增下列程式碼，然後儲存檔案。
	
	`name = "demo-ipam"`
	`enable_ipv6 = false`
	`enable_dns_hostnames = true`
	`enable_dns_support = true`
		DevOps 工程師

任務	Description	所需的技能

建立 outputs.tf 檔案。	建立新的outputs.tf 檔案，公開 CodeBuild 中的一些值。
	
	使用下列步驟：
	
	1. 在 terraform 目錄中建立新的outputs.tf 檔案。
	2. 開啟 檔案，新增下列程式碼，然後儲存檔案。
	
	``
	# Output VPC ID and
	 ARN in CodeBuild
	output "vpc1" {
	  description = "VPC
	 1 information"
	  value = {
	  id = try(aws_v
	pc.vpc1.id, null)
	  arn = try(aws_v
	pc.vpc1.arn, null)
	  }
	}
	``
		DevOps 工程師

遞交自訂項目。	若要將新的自訂遞交至帳戶自訂儲存庫，請執行下列命令： ``` # Assumes you are still in the /terraform directory cd .. # Skip if you are in the account customization root directory (APG-AFT- IPAM) git add . git commit -m "APG customization" git push origin ```	DevOps 工程師

任務	Description	所需的技能

套用自訂。	使用新建立的帳戶自訂，將程式碼新增至請求新帳戶的 `account-requests.tf` 檔案。自訂欄位會在使用正確 IPAM 配置的 IPv4 CIDR 建立 VPC 所需的付費帳戶中建立 Systems Manager 參數。
	
	使用下列步驟：
	
	1. 從終端機導覽至 `aft-account-request/terraform` 目錄。
	2. 開啟 `account-requests.tf` 檔案，使用值新增和修改下列程式碼，然後儲存檔案。
	
	`module "sandbox_account_01" {`
	`  source = "./modules/aft-account-request"`
	
	`  control_tower_parameters = {`
	`  AccountEmail = "john.doe@example.com"`
	`  AccountName  = "sandbox-account-01"`
	`  # Syntax for top-level OU`
	`  ManagedOrganizationalUnit = "Sandbox"`
		AWS DevOps

任務	Description	所需的技能

	# Syntax for nested OU
	# ManagedOrganizationalUnit = "Sandbox (ou-xfe5-a8hb8ml8)"
	SSOUserEmail = "john.doe@example.com"
	SSOUserFirstName = "John"
	SSOUserLastName = "Doe"
	}
	
	account_tags = {
	"ABC:Owner" = "john.doe@example.com"
	"ABC:Division" = "ENT"
	"ABC:Environment" = "Dev"
	"ABC:CostCenter" = "123456"
	"ABC:Vended" = "true"
	"ABC:DivCode" = "102"
	"ABC:BUCode" = "ABC003"
	"ABC:Project" = "123456"
	}
	
	change_management_parameters = {
	change_requested_by = "John Doe"
	change_reason = "testing	

任務	Description	所需的技能

```
 the account vending
 process"
  }

  custom_fields = {
    vpc-ipam-id =
 "ipam-pool-123456a
b789041cd2"
    vpc-ipam-netmask
 = "24"

  }

  account_customizat
ions_name = "APG-AFT-
IPAM" # Do not change
 this value
}
```

> **Important**
>
> 請確定該vpc-ipam-netmask 值小於或等於 IPAM 集區配置 netmask，且 IPAM 集區中有足夠的 IPs 可配置給 VPC。

3. 若要將新帳戶遞交至帳戶請求儲存庫，請使用下列命令：

```
git add .
git commit -m
 'requested APG
 account'
git push origin
```

任務	Description	所需的技能
驗證自訂。	登入新付費帳戶並確認已成功套用自訂。 使用下列步驟： 1. 開啟 AWS IAM Identity Center 您的登入頁面，並以您在 `account-requests.tf` 檔案中指定的使用者身分進行驗證。 2. 從 IAM Identity Center 帳戶清單中，選擇您先前提供的帳戶，並使用 AWS `AdministratorAccess` 角色登入。 3. 導覽至 VPC 主控台，然後選取您的 VPCs。 4. 選取您剛建立之 VPC 的 VPC ID。 5. 選擇 CIDRs 索引標籤。您應該會看到 IPAM 配置的 IPv4 CIDR。 6. （選用）登入您的 IPAM 委派管理員帳戶，並確認配置顯示在 IPAM 主控台中。	DevOps 工程師

疑難排解

問題	解決方案
由於許可不足，您在資源建立或管理時遇到失敗。	檢閱連接到 Step Functions、CodeBuild 和部署所涉及其他服務的 AWS Identity and Access Management (IAM) 角色和政策。確認他們具有

問題	解決方案
	必要的許可。如果有許可問題，請調整 IAM 政策以授予必要的存取權。
您在部署期間達到 AWS 服務 配額。	部署管道之前，請檢查 資源的配額，例如 Amazon Simple Storage Service AWS 服務 (Amazon S3) 儲存貯體、IAM 角色和 AWS Lambda 函數。如有必要，請求增加配額。如需詳細資訊，請參閱《AWS 一般參考》中的AWS 服務 配額。

相關資源

AWS 服務 文件

- AWS Control Tower 使用者指南
- IPAM 的運作方式
- IAM 中的安全最佳實務
- AWS 服務 配額

其他資源

- Terraform AWS 提供者文件

使用 AWS Systems Manager 自動化新增或更新 Windows 登錄項目

Appasaheb Bagali，Amazon Web Services

總結

AWS Systems Manager 是 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的遠端管理工具。Systems Manager 提供對 Amazon Web Services 上基礎設施的可見性和控制。此多用途工具可用來修正安全性漏洞掃描報告識別為漏洞的 Windows 登錄檔變更。

此模式涵蓋透過自動化基於環境安全而建議的登錄檔變更，來保護執行 Windows 作業系統之 EC2 執行個體安全的步驟。模式使用 Run 命令來執行 Command 文件。程式碼已連接，一部分包含在程式碼區段中。

先決條件和限制

- 作用中的 AWS 帳戶
- 存取 EC2 執行個體和 Systems Manager 的許可

Architecture

目標技術堆疊

- 具有兩個子網路和網路位址轉譯 (NAT) 閘道的虛擬私有雲端 (VPC)
- 用於新增或更新登錄檔名稱和值的 Systems Manager 命令文件
- Systems Manager Run Command 在指定的 EC2 執行個體上執行命令文件

目標架構

工具

工具

- IAM 政策和角色 – AWS Identity and Access Management (IAM) 是一種 Web 服務，可協助您安全地控制對 AWS 資源的存取。您可以使用 IAM 來控制能通過身分驗證 (登入) 和授權使用資源的 (具有許可) 的人員。

- Amazon Simple Storage Service – Amazon Simple Storage Service (Amazon S3) 是網際網路的儲存體。此服務旨在降低開發人員進行網路規模運算的難度。在此模式中，會使用 S3 儲存貯體來存放 Systems Manager 日誌。
- AWS Systems Manager – AWS Systems Manager 是一種 AWS 服務，可用來檢視和控制 AWS 上的基礎設施。Systems Manager 透過掃描受管執行個體並報告 （或對其採取修正動作） 偵測到的任何政策違規，協助您維護安全性和合規性。
- AWS Systems Manager Command 文件 – Run Command 會使用 AWS Systems Manager Command 文件。Systems Manager 支援的所有 Linux 和 Windows Server 作業系統都支援大多數命令文件。
- AWS Systems Manager Run Command – AWS Systems Manager Run Command 可讓您從遠端安全地管理受管執行個體的組態。您可以使用 Run Command 自動化常見的管理任務，並大規模執行一次性組態變更。

Code

您可以使用下列範例程式碼，將 Microsoft Windows 登錄檔名稱新增至 或更新，將`Version`登錄檔路徑新增至 `HKCU:\Software\ScriptingGuys\Scripts`，並將值新增至 2。

```
#Windows registry path which needs to add/update
$registryPath ='HKCU:\\Software\\ScriptingGuys\\Scripts'
#Windows registry Name  which needs to add/update
$Name = 'Version'
#Windows registry value  which needs to add/update
$value = 2
# Test-Path cmdlet to see if the registry key exists.
IF(!(Test-Path $registryPath))
        {
           New-Item -Path $registryPath -Force | Out-Null
           New-ItemProperty -Path $registryPath -Name $name -Value     $value ` -
PropertyType DWORD -                 Force | Out-        Null
        } ELSE {
                      New-ItemProperty -Path $registryPath -Name $name -Value $value `
 -PropertyType            DWORD        -Force | Out-Null
            }
echo 'Registry Path:'$registryPath
 echo 'Registry Name:'$registryPath
 echo 'Registry Value:'(Get-ItemProperty -Path $registryPath -Name $Name).version
```

連接完整的 Systems Manager 命令文件 JavaScript 物件標記法 (JSON) 程式碼範例。

史詩

設定 VPC

任務	Description	所需的技能
建立 VPC。	在 AWS 管理主控台上，建立具有公有和私有子網路和 NAT 閘道的 VPC。如需詳細資訊，請參閱 AWS 文件。	雲端管理員
建立安全群組。	確保每個安全群組允許從來源 IP 地址存取遠端桌面通訊協定 (RDP)。	雲端管理員

建立 IAM 政策和 IAM 角色

任務	Description	所需的技能
建立 IAM 政策。	建立可讓您存取 Amazon S3、Amazon EC2 和 Systems Manager 的 IAM 政策。	雲端管理員
建立 IAM 角色。	建立 IAM 角色，並連接提供 Amazon S3、Amazon EC2 和 Systems Manager 存取權的 IAM 政策。	雲端管理員

執行自動化

任務	Description	所需的技能
建立 Systems Manager 命令文件。	建立 Systems Manager Command 文件，以部署要新增或更新的 Microsoft Windows 登錄檔變更。	雲端管理員

任務	Description	所需的技能
執行 Systems Manager Run Command。	執行 Systems Manager Run Command，選取 Command 文件和 Systems Manager 目標執行個體。這會將所選命令文件中的 Microsoft Windows 登錄檔變更推送到目標執行個體。	雲端管理員

相關資源

- AWS Systems Manager
- AWS Systems Manager 文件
- AWS Systems Manager 執行命令

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Python 在 AMS 中自動建立 RFC

Gnanasekaran Kailasam，Amazon Web Services

摘要

AWS Managed Services (AMS) 透過持續管理 Amazon Web Services (AWS) 基礎設施，協助您更有效率且安全地操作雲端型基礎設施。若要變更受管環境，您需要建立並提交新的變更請求 (RFC)，其中包含特定操作或動作的變更類型 (CT) ID。

不過，手動建立 RFC 可能需要大約五分鐘的時間，而組織中的團隊可能需要每天提交多個 RFCs。此模式可協助您自動化 RFC 建立程序、縮短每個 RFC 的建立時間，並消除手動錯誤。

此模式說明如何使用 Python 程式碼自動建立 `Stop EC2 instance` RFC，以停止 AMS 帳戶中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。然後，您可以將此模式的方法和 Python 自動化套用至其他 RFC 類型。

先決條件和限制

先決條件

- AMS 進階帳戶。如需詳細資訊，請參閱 AWS Managed Services 文件中的 AMS 操作計劃。 AWS Managed Services
- 您的 AMS 帳戶中至少有一個現有的 EC2 執行個體。
- 了解如何在 AMS 中建立和提交 RFCs。
- 熟悉 Python。

限制

- 您只能將 RFCs用於 AMS 帳戶中的變更。您的 AWS 帳戶會針對類似的變更使用不同的程序。

Architecture

技術堆疊

- AMS
- AWS 命令列界面 (AWS CLI)

- 適用於 Python 的 AWS SDK (Boto3)
- Python 及其所需的套件 (JSON 和 Boto3)

自動化和擴展

此模式提供範例程式碼來自動化 `Stop EC2 instance` RFC，但您可以將此模式的範例程式碼和方法用於其他 RFCs。

工具

- AWS Managed Services – AMS 可協助您更有效率且安全地操作 AWS 基礎設施。
- AWS CLI – AWS Command Line Interface (AWS CLI) 是管理 AWS 服務的統一工具。在 AMS 中，變更管理 API 提供建立和管理 RFCs的操作。
- 適用於 Python 的 AWS 開發套件 (Boto3) – 適用於 Python 的開發套件可讓您輕鬆地將 Python 應用程式、程式庫或指令碼與 AWS 服務整合。

Code

`AMS Stop EC2 Instance.zip` 檔案（已連接）包含用於建立 `Stop EC2 instance` RFC 的 Python 程式碼。您也可以將此程式碼設定為為多個 EC2 執行個體提交單一 RFC。

史詩

選項 1 – 設定 macOS 或 Linux 的環境

任務	Description	所需的技能
安裝並驗證 Python。	1. 開啟終端機視窗並執行 `brew install python3`命令。 2. 執行 `python --version` 命令來驗證 Python 是否正確安裝。 3. 執行 `pip --version` 命令來驗證 `pip` 是否正確安裝。	AWS 系統管理員

任務	Description	所需的技能
安裝 AWS CLI。	執行 `pip install awscli --upgrade –user`命令來安裝 AWS CLI。	AWS 系統管理員
安裝 Boto3。	執行 `pip install boto3`命令來安裝 Boto3。	AWS 系統管理員
安裝 JSON。	執行 `pip install json`命令來安裝 JSON。	AWS 系統管理員
設定 AMS CLI。	登入 AWS 管理主控台，開啟 AMS 主控台，然後選擇 文件。下載包含 AMS CLI 的 .zip 檔案，將其解壓縮，然後將其安裝在本機電腦上。 安裝 AMS CLI 之後，請執行 `aws amscm help`命令。輸出提供 AMS 變更管理程序的相關資訊。	AWS 系統管理員

選項 2 – 設定 Windows 的環境

任務	Description	所需的技能
安裝並驗證 Python。	1. 開啟適用於 Windows 的 Python 版本頁面，下載最新版本，然後安裝 Python。 2. 執行 `python --version` 命令來驗證 Python 是否正確安裝。 3. 執行 `pip --version` 命令來驗證 `pip` 是否正確安裝。	AWS 系統管理員

任務	Description	所需的技能
安裝 AWS CLI。	執行 `pip install awscli --upgrade –user`命令來安裝 AWS CLI。	AWS 系統管理員
安裝 Boto3。	執行 `pip install boto3`命令來安裝 Boto3。	AWS 系統管理員
安裝 JSON。	執行 `pip install json`命令來安裝 JSON。	AWS 系統管理員
設定 AMS CLI。	登入 AWS 管理主控台，開啟 AMS 主控台，然後選擇 文件。下載包含 AMS CLI 的 .zip 檔案，將其解壓縮，然後將其安裝在本機電腦上。 安裝 AMS CLI 之後，請執行 `aws amscm help`命令。輸出提供有關 AMS 變更管理程序的資訊	AWS 系統管理員

擷取 RFC 的 CT ID 和執行參數

任務	Description	所需的技能
擷取 RFC 的 CT ID、版本和執行參數。	每個 RFC 都有不同的 CT ID、版本和執行參數。您可以使用下列其中一個選項來擷取此資訊： 1. 請遵循在 RFC 中使用 CLI 區段尋找變更請求 (RFC) 中的指示，使用 AWS Managed Services 文件的	AWS 系統管理員

任務	Description	所需的技能
	範例。 AWS Managed Services 2. 開啟類似類型的現有 RFC，或透過 AMS 主控台建立新的 RFC 做為測試。使用 RFC 的 CT ID 和執行參數。如需詳細資訊，請參閱 AWS Managed Services 文件中的使用主控台尋找 RFC。 **Note** 若要針對其他 RFCs 調整此模式的 Python 自動化，請將 `ams_stop_ec2_instance` Python 程式碼檔案中的 CT 類型和參數值取代為您解壓縮AMS Stop EC2 Instance.zip 的檔案（已連接）。	

執行 Python 自動化

任務	Description	所需的技能
執行 Python 自動化。	1. 下載 AMS Stop EC2 Instance.zip 檔案（已連接）到您的本機電腦並解壓縮 檔案。	AWS 系統管理員

任務	Description	所需的技能
	2. `input_instances` 使用 EC2 執行個體資訊進行更新。 3. 開啟終端機並導覽至解壓縮程式碼的路徑 4. 執行 `pythonams_stop_ec2_instance.py` 命令。	

相關資源

- 什麼是變更類型？
- CLI 教學課程：高可用性雙層堆疊 (Linux/RHEL)

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS Systems Manager 維護 Windows 自動停止和啟動 Amazon RDS 資料庫執行個體

Ashita Dsilva，Amazon Web Services

總結

此模式示範如何使用 AWS Systems Manager 維護時段，依特定排程自動停止和啟動 Amazon Relational Database Service (Amazon RDS) 資料庫執行個體 （例如，在營業時間外關閉資料庫執行個體以降低成本）。為此，Systems Manager 對於典型的使用案例具有成本效益。

AWS Systems Manager 自動化提供 `AWS-StopRdsInstance`和 `AWS-StartRdsInstance` Runbook 來停止和啟動 Amazon RDS 資料庫執行個體。這表示您不需要使用 AWS Lambda 函數撰寫自訂邏輯或建立 Amazon CloudWatch Events 規則。

Systems Manager 提供兩種排程任務的功能：State Manager 和維護 Windows。State Manager 會為您的 Amazon Web Services (AWS) 帳戶中的資源設定和維護所需的狀態組態一次，或依特定排程進行。維護 Windows 會在特定時段內對您帳戶中的資源執行任務。雖然您可以在狀態管理員或維護時段使用此模式的方法，但我們建議您使用維護時段，因為它可以根據指派的優先順序執行一或多個任務，也可以執行 AWS Lambda 函數和 AWS Step Functions 任務。如需 State Manager 和維護 Windows 的詳細資訊，請參閱 Systems Manager 文件中的在 State Manager 和維護 Windows 之間進行選擇。

此模式提供詳細步驟來設定兩個單獨的維護時段，這些時段使用 cron 表達式來停止，然後啟動 Amazon RDS 資料庫執行個體。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 您要在特定排程中停止和啟動的現有 Amazon RDS 資料庫執行個體。
- 所需排程的 Cron 表達式。例如，表達式會在每個星期一、星期二、星期三、星期四和星期五的 09：00 `cron(0 9 ? * MON-FRI *)`執行任務。如需詳細資訊，請參閱 Systems Manager 文件中的維護時段的 Cron 和 Rate 表達式。
- 熟悉 Systems Manager。
- 啟動和停止 RDS 執行個體的許可。如需詳細資訊，請參閱 Epics 區段。

限制

- Amazon RDS 資料庫執行個體一次最多可停止七天。七天後，資料庫執行個體會自動重新啟動，以確保收到任何必要的維護更新。
- 您無法停止僅供讀取複本或具有僅供讀取複本的資料庫執行個體。
- 您無法在多可用區組態中停止 Amazon RDS for SQL Server 資料庫執行個體。
- 服務配額適用於維護 Windows 和 Systems Manager 自動化。如需服務配額的詳細資訊，請參閱 AWS 一般參考 文件中的AWS Systems Manager 端點和配額。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

Architecture

下圖顯示自動停止和啟動 Amazon RDS 資料庫執行個體的工作流程。

工作流程有下列步驟：

1. 建立維護時段並使用 cron 表達式來定義 Amazon RDS 資料庫執行個體的停止和啟動排程。

2. 使用 `AWS-StopRdsInstance`或 `AWS-StartRdsInstance` Runbook 將 Systems Manager Automation 任務註冊到維護時段。

3. 為您的 Amazon RDS 資料庫執行個體使用標籤型資源群組，向維護時段註冊目標。

技術堆疊

- AWS CloudFormation
- AWS Identity and Access Management (IAM)
- Amazon RDS
- Systems Manager

自動化和擴展

您可以同時停止和啟動多個 Amazon RDS 資料庫執行個體，方法是標記所需的 Amazon RDS 資料庫執行個體、建立包含所有已標記資料庫執行個體的資源群組，以及將此資源群組註冊為維護時段的目標。

工具

- AWS CloudFormation 是一項服務，可協助您建立和設定 AWS 資源的模型。
- AWS Identity and Access Management (IAM) 是一種 Web 服務，可協助您安全地控制對 AWS 資源的存取。
- Amazon Relational Database Service (Amazon RDS) 是一種 Web 服務，可讓您更輕鬆地在 中設定、操作和擴展關聯式資料庫 AWS 雲端。
- AWS Resource Groups 可協助您將 AWS 資源組織成群組、標記資源，以及管理、監控和自動化分組資源上的任務。
- AWS Systems Manager 是 AWS 服務 ，可用來檢視和控制您的基礎設施 AWS。此模式使用 Systems Manager 的下列功能：
  - AWS Systems Manager 自動化可簡化 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和其他 AWS 資源的常見維護和部署任務。
  - AWS Systems Manager 維護 Windows 可協助您定義何時對執行個體執行潛在破壞性動作的排程。

史詩

建立和設定 Systems Manager Automation 的 IAM 服務角色

任務	Description	所需的技能
設定 Systems Manager Automation 的 IAM 服務角色。	登入 AWS 管理主控台 並建立 Systems Manager Automation 的服務角色。您可以使用下列兩種方法之一來建立此服務角色： • 使用 AWS CloudFormation 設定 Systems Manager Automation 的服務角色 • 使用 IAM 設定 Systems Manager 自動化的角色	AWS 管理員

任務	Description	所需的技能
	Systems Manager Automation 工作流程會使用服務角色在 Amazon RDS 資料庫執行個體上執行啟動和停止動作，以叫用 Amazon RDS。 必須使用下列內嵌政策來設定服務角色，該政策具有啟動和停止 Amazon RDS 資料庫執行個體的許可： `{ "Version": "2012-10-17", "Statement": [ { "Sid": "RdsStartStop", "Effect": "Allow", "Action": [ "rds:StopDBInstance", "rds:StartDBInstance" ], "Resource": "<RDS_Instance_ARN>" }, { "Sid": "RdsDescribe", "Effect": "Allow", "Action": "rds:DescribeDBInstances", "Resource": "*"`	

任務	Description	所需的技能
	}] } 請務必將 <RDS_Instance_ARN> 取代為 Amazon RDS 資料庫執行個體的 Amazon Resource Name (ARN)。 如果您不熟悉使用 IAM 政策和角色，請遵循排程 Amazon RDS 停止和開始使用 AWS Systems Manager部落格文章的解決方案概觀一節中的指示。 **Important** 請務必記錄服務角色的 ARN。	

建立資源群組

任務	Description	所需的技能
標記 Amazon RDS 資料庫執行個體。	開啟 Amazon RDS 主控台，並標記您要新增至資源群組的 Amazon RDS 資料庫執行個體。標籤是指派給 AWS 資源的中繼資料，由索引鍵/值對組成。我們建議您使用動作做為標籤索引鍵，並使用 StartStop 做為值。	AWS 管理員

任務	Description	所需的技能
	如需詳細資訊，請參閱 Amazon RDS 文件中的新增、列出和移除標籤。	
為您的標記 Amazon RDS 資料庫執行個體建立資源群組。	開啟 AWS Resource Groups 主控台，並根據您為 Amazon RDS 資料庫執行個體建立的標籤建立資源群組。 在分組條件下，請確定您為資源類型選擇 AWS::RDS::DBInstance，然後提供標籤的鍵值對（例如，「Action-StartStop」）。這可確保服務只會檢查 Amazon RDS 資料庫執行個體，而不是具有此標籤的其他資源。請確定您記錄資源群組的名稱。 如需詳細資訊和詳細步驟，請參閱 AWS Resource Groups 文件中的建置標籤型查詢和建立群組。	AWS 管理員

設定維護時段以停止 Amazon RDS 資料庫執行個體

任務	Description	所需的技能
建立維護時段。	1. 開啟 Systems Manager 主控台，選擇維護時段，然後選擇建立維護時段。提供維護時段的名稱（例如，「StopRdsInstance」），輸入描述，然後取消核取允許未註冊的目標。	AWS 管理員

任務	Description	所需的技能
	2. 選擇 CRON/Rate 表達式，並提供排程表達式來定義何時應停止 Amazon RDS 資料庫執行個體。針對持續時間和 0forStop 啟動任務輸入 1。根據預設，時區會顯示 UTC。您可以根據 cron 表達式中定義的時間戳記，變更時區以啟動維護時段。 3. 選擇建立維護時段。系統會將您返回維護時段頁面，且維護時段的狀態為已啟用。 ⚠ Important 停止資料庫執行個體的任務會在啟動時幾乎立即執行，而且不會跨越維護時段的整個持續時間。此模式提供持續時間和停止啟動任務的最小值，因為它們是維護時段的必要參數。 如需詳細資訊和詳細步驟，請參閱 Systems Manager 文件中的建立維護時段（主控台）。	

任務	Description	所需的技能
將目標指派給維護時段。	1. 在 Systems Manager 主控台上，選擇維護時段，選擇動作，然後選擇註冊目標。 2. 在目標區域中，指定選擇資源群組，然後選擇您帳戶中現有資源群組的名稱。 3. 針對資源類型，選擇 AWS::RDS::DBInstance，然後選擇註冊目標。 如需詳細資訊和詳細步驟，請參閱 Systems Manager 文件中的將目標指派給維護時段（主控台）。	AWS 管理員

任務	Description	所需的技能
將任務指派給維護時段。	1. 在 Systems Manager 主控台上，選擇維護時段，然後選擇您的維護時段。選擇動作，然後選擇註冊自動化任務。 2. 針對文件，選擇 AWS-StopRdsInstance。 3. 在目標區段中，選擇選取已註冊的目標群組，然後選擇您使用目前維護時段註冊的維護時段目標。 4. ForRate 控制，請針對並行和錯誤閾值指定 100%。您可以根據任務並行和錯誤閾值的需求來變更速率控制值。如需詳細資訊，請參閱 Systems Manager 文件中的關於並行和錯誤閾值。 5. 在 IAM 服務角色區段中，針對服務角色，將此方塊保留空白或建立您自己的自訂角色。如果您將方塊保留空白，Systems Manager 會自動建立服務連結角色 AWSServiceRoleForAmazonSSM，然後將角色與任務建立關聯。若要建立您自己的自訂角色，請參閱建立維護時段的自訂服務角色（主控台），然後將該自訂角色與任務建立關聯。	AWS 管理員

任務	Description	所需的技能
	6. 在輸入參數區段中，指定 Runbook 的下列參數： • InstanceId： `{{RESOURCE_ID}}` **Note** 對於 InstanceId，虛擬參數用於從 ARN 擷取 Amazon RDS 資料庫資源 ID。若要進一步了解虛擬參數，請參閱 Systems Manager 文件中的關於虛擬參數。 • AutomationAssumeRole：提供您為 Systems Manager Automation 建立之服務角色的 ARN。 7. 選擇註冊自動化任務。 **Important** 服務角色選項定義維護時段執行任務所需的服務角色。不過，此角色與您先前為 Systems Manager Automation 建立的服務角色不同。	

任務	Description	所需的技能
	如需詳細資訊和詳細步驟，請參閱 Systems Manager 文件中的將任務指派給維護時段（主控台）。	

設定維護時段以啟動 Amazon RDS 資料庫執行個體

任務	Description	所需的技能
設定維護時段以啟動 Amazon RDS 資料庫執行個體。	重複設定維護時段中的步驟，停止 Amazon RDS 資料庫執行個體 epic，以設定另一個維護時段，在排程時間啟動 Amazon RDS 資料庫執行個體。 **Important** 當您設定維護時段以啟動資料庫執行個體時，必須進行下列變更： • 使用維護時段的新名稱（例如，「StartRdsInstance」）。 • 將 cron 表達式取代為您要用來啟動資料庫執行個體的 cron 表達式。 • 在`AWS-StartRdsInstance` 任務中將 `AWS-StopRdsInstance` Runbook 取代為 。	AWS 管理員

相關資源

- 使用 Systems Manager Automation 文件來管理執行個體並降低非上班時間的成本 (AWS 部落格文章)

使用 Terraform 集中 AWS Organizations 中的軟體套件分佈

Pradip kumar Pandey、Chintamani Aphale、T.V.R.L.Phani Kumar Dadi、Pratap Kumar Nanda、Aarti Rajput 和 Mayuri Shinde、Amazon Web Services

總結

企業通常會維護分散在多個 的多個 AWS 帳戶 ， AWS 區域 以便在工作負載之間建立強大的隔離障礙。為了保持安全與合規，其管理團隊會安裝代理程式型工具，例如用於安全性掃描的 CrowdStrike、SentinelOne 或 TrendMicro 工具，以及用於監控的 Amazon CloudWatch 代理程式、Datadog Agent 或 AppDynamics 代理程式。這些團隊通常會在想要集中自動化軟體套件管理和分佈到這個大型環境中時面臨挑戰。

Distributor 是 的一項功能AWS Systems Manager，可透過單一簡化界面，將軟體封裝和發佈至雲端和內部部署伺服器中受管 Microsoft Windows 和 Linux 執行個體的程序自動化。此模式示範如何使用 Terraform 來進一步簡化管理軟體安裝的程序，並在 內的大量執行個體和成員帳戶上執行指令碼 AWS Organizations ，而只需最少的努力。

此解決方案適用於由 Systems Manager 管理的 Amazon、Linux 和 Windows 執行個體。

先決條件和限制

- 要安裝軟體的 Distributor 套件
- Terraform 0.15.0 版或更新版本
- 由 Systems Manager 管理且具有存取目標帳戶中 Amazon Simple Storage Service (Amazon S3Amazon EC2) 執行個體
- 您組織使用 設定的登陸區域 AWS Control Tower
- （選用） 適用於 Terraform (AFT) 的帳戶工廠

Architecture

資源詳細資訊

此模式使用 Account Factory for Terraform (AFT) 建立所有必要 AWS 的資源和程式碼管道，以在部署帳戶中部署資源。程式碼管道在兩個儲存庫中執行：

- 全域自訂包含將跨向 AFT 註冊的所有帳戶執行的 Terraform 程式碼。
- 帳戶自訂包含將在部署帳戶中執行的 Terraform 程式碼。

您也可以在帳戶自訂資料夾中執行 Terraform 命令，在不使用 AFT 的情況下部署此解決方案。

Terraform 程式碼會部署下列資源：

- AWS Identity and Access Management (IAM) 角色和政策
  - SystemsManager-AutomationExecutionRole 授予使用者在目標帳戶中執行自動化的許可。
  - SystemsManager-AutomationAdministrationRole 授予使用者在多個帳戶和組織單位 (OUs許可。
- 套件的壓縮檔案和 manifest.json
  - 在 Systems Manager 中，套件包含至少一個軟體或可安裝資產的 .zip 檔案。
  - JSON 資訊清單包含套件程式碼檔案的指標。
- S3 bucket (S3 儲存貯體)
  - 跨組織共用的分散式套件會安全地存放在 Amazon S3 儲存貯體中。
- AWS Systems Manager 文件 (SSM 文件)
  - `DistributeSoftwarePackage` 包含將軟體套件分發到成員帳戶中每個目標執行個體的邏輯。
  - `AddSoftwarePackageToDistributor` 包含可封裝可安裝軟體資產並將其新增至 Automation 的邏輯。 AWS Systems Manager
- Systems Manager 關聯
  - Systems Manager 關聯用於部署解決方案。

架構和工作流程

此圖說明了下列步驟：

1. 若要從集中式帳戶執行解決方案，您可以將套件或軟體以及部署步驟上傳至 S3 儲存貯體。
2. 您的自訂套件可在 Systems Manager 主控台文件區段的「由我擁有」索引標籤中使用。
3. State Manager 是 Systems Manager 的功能，可在整個組織中建立、排程和執行套件的關聯。關聯指定必須先在受管節點上安裝和執行軟體套件，才能安裝在目標節點上。
4. 關聯會指示 Systems Manager 在目標節點上安裝套件。
5. 對於任何後續安裝或變更，使用者可以定期或從單一位置手動執行相同的關聯，以跨帳戶執行部署。
6. 在成員帳戶中，自動化會將部署命令傳送至 Distributor。
7. Distributor 會將軟體套件分散到各個執行個體。

此解決方案使用 內的管理帳戶 AWS Organizations，但您也可以指定 帳戶 （委派管理員） 來代表組織管理此帳戶。

工具

AWS 服務

- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。此模式使用 Amazon S3 來集中和安全地存放分散式套件。
- AWS Systems Manager 可協助您管理在 中執行的應用程式和基礎設施 AWS 雲端。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。此模式使用以下 Systems Manager 功能：
  - Distributor 可協助您封裝軟體並將其發佈至 Systems Manager 受管執行個體。
  - 自動化可簡化許多 AWS 服務的常見維護、部署和修復任務。
  - 文件會在您的組織和帳戶中對 Systems Manager 受管執行個體執行動作。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 AWS 帳戶合併到您建立和集中管理的組織。

其他工具

- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。

程式碼儲存庫

此模式的說明和程式碼可在 GitHub 集中式套件分發儲存庫中取得。

最佳實務

- 若要將標籤指派給關聯，請使用 AWS Command Line Interface (AWS CLI) 或 AWS Tools for PowerShell。不支援使用 Systems Manager 主控台將標籤新增至關聯。如需詳細資訊，請參閱 Systems Manager 文件中的標記 Systems Manager 資源。
- 若要使用從另一個帳戶共用的文件新版本來執行關聯，請將文件版本設定為 `default`。
- 若要僅標記目標節點，請使用一個標籤索引鍵。如果您想要使用多個標籤索引鍵將節點設為目標，請使用資源群組選項。

史詩

設定來源檔案和帳戶

任務	Description	所需的技能

複製儲存庫。	1. 複製 GitHub 集中式套件分發儲存庫：
	`git clone https://github.com/aws-samples/aws-organization-centralised-package-distribution`
	2. Terraform 程式碼儲存庫需要兩個由 AFT 管理的自訂資料夾。確認儲存庫的本機副本包含下列資料夾：
	`$ cd centralised-package-distribution`
	`$ ls`
	`global-customization account-customization`
		DevOps 工程師

更新全域變數。	更新 `global-customization/variables.tf` 檔案中的下列輸入參數。這些變數適用於由 AFT 建立和管理的所有帳戶。 • `account_id` ：將部署 Distributor 解決方案的帳戶 ID。 • `aws_region` ：將部署關聯的 AWS 區域 。	DevOps 工程師

任務	Description	所需的技能
更新帳戶變數。	在 `account-customization/variables.tf` 檔案中更新下列輸入參數。這些變數僅適用於由 AFT 建立和管理的特定帳戶。 • `package_bucket_name` ：包含套件分佈檔案的 S3 儲存貯體名稱。 • `package_name` ：套件分佈檔案的名稱。 • `package_version` ：安裝程式的套件版本。	DevOps 工程師

自訂參數和部署檔案

任務	Description	所需的技能
更新狀態管理員關聯的輸入參數。	更新 `account-customization/association.tf` 檔案中的下列輸入參數，以定義您想要在執行個體上維護的狀態。如果預設參數值支援您的使用案例，您可以使用這些參數值。 • `targetAccounts` ：AWS Organizations 中的組織單位 (OU) IDs，代表具有要分佈之目標執行個體的帳戶。OU IDs以 "ou" 開頭。 • `targetRegions` ：目標執行個體正在執行的 AWS 區	DevOps 工程師

任務	Description	所需的技能

	域（例如，「us-east-1」或「ap-southeast-2」）。
	• `action`：指定是否要安裝或解除安裝套件。
	• `installationType`：下列其中一個安裝類型：
	  • `uninstall`：套件已解除安裝。
	  • `reinstall`：應用程式會離線，直到重新安裝程序完成為止。
	  • `In-place update`：當新的或更新的檔案新增至安裝時，應用程式即可使用。
	• `name`：要安裝或解除安裝的套件名稱。
	• `version`：要安裝或解除安裝的套件版本。如果未安裝套件版本，系統會傳回錯誤。
	• `bucketName`：已部署套件的 S3 儲存貯體名稱。此儲存貯體應僅包含套件和資訊清單檔案。
	• `bucketPrefix`：儲存套件資產的 S3 字首。
	• `AutomationAssumeRole`：的 Amazon Resource Name (ARN)`SystemsManager-AutomationAdministrationRole`。	

任務	Description	所需的技能
準備壓縮檔案和套件 `manifest.json` 的檔案。	此模式提供範例 PowerShell 可安裝檔案 (Windows 為 .msi，Linux 為 .rpm)，並在 `account-customization/package` 資料夾中安裝和解除安裝指令碼。 1. 將 PowerShell 可安裝檔案取代為您自己的檔案，或提供可安裝檔案、安裝和解除安裝指令碼，以及資訊清單檔案，以在帳戶中的 `account-customization` 資料夾中建立套件。 2. 根據您的需求自訂 Terraform 在`account-customization` 資料夾中產生的預設`manifest.json` 檔案。	DevOps 工程師

執行 Terraform 命令來佈建資源

任務	Description	所需的技能

初始化 Terraform 組態。	若要使用 AFT 自動部署解決方案，請將程式碼推送至 AWS CodeCommit：
	
	`$ git add *`
	`$ git commit -m "message"`
	`$ git push`
		DevOps 工程師

任務	Description	所需的技能
	您也可以從 `account-customization` 資料夾執行 Terraform 命令，在不使用 AFT 的情況下部署此解決方案。若要初始化包含 Terraform 檔案的工作目錄，請執行： `$ terraform init`	
預覽變更。	若要預覽 Terraform 對基礎設施所做的變更，請執行 命令： `$ terraform plan` 此命令會評估 Terraform 組態，以判斷已宣告之資源的所需狀態。它也會比較所需狀態與要在工作區內佈建的實際基礎設施。	DevOps 工程師
套用變更。	執行下列命令來實作您對`variables.tf` 檔案所做的變更： `$ terraform apply`	DevOps 工程師

驗證資源

任務	Description	所需的技能
驗證 SSM 文件的建立。	1. 在 Systems Manager 主控台的左側導覽窗格中，選擇文件。	DevOps 工程師

任務	Description	所需的技能
	2. 選擇 Owned by me (我所擁有) 索引標籤。 您應該會看到 `DistributeSoftwarePackage` 和 `AddSoftwarePackageToDistributor` 套件。	
驗證自動化的成功部署。	1. 在 Systems Manager 主控台的左側導覽窗格中，選擇自動化。 2. 在自動化執行清單中，您應該會看到最新的 `DistributeSoftwarePackage` 和 `AddSoftwarePackageToDistributor` 部署。 3. 選擇執行 ID 以驗證它們是否成功完成。	DevOps 工程師
驗證部署到目標成員帳戶執行個體的套件。	1. 在 Systems Manager 主控台的導覽窗格中，選擇執行命令。 2. 在命令歷史記錄中，您會看到每個調用及其狀態。 3. 選擇任何命令 ID 以查看每個目標執行個體的部署歷史記錄。 4. 選擇執行個體 ID 並檢查分佈的輸出區段。	DevOps 工程師

疑難排解

問題	解決方案
狀態管理員關聯失敗或停滯在待定狀態。	請參閱 AWS 知識中心的疑難排解資訊。
排程的關聯無法執行。	您的排程規格可能無效。State Manager 目前不支援在關聯 cron 表達式中指定月份。使用 Cron 或 Rate 運算式來確認排程。

相關資源

- 集中式套件分佈 (GitHub 儲存庫)
- Terraform 帳戶工廠 (AFT)
- 使用案例和最佳實務 (AWS Systems Manager 文件)

使用 NLog 在 Amazon CloudWatch Logs 中設定 .NET 應用程式的記錄

Amazon Web Services 的聖胡島和羅布希爾 (AWS)

總結

此模式說明如何使用 NLog 開放原始碼記錄架構，在 Amazon CloudWatch Logs 中記錄 .NET 應用程式用量和事件。在 CloudWatch 主控台中，您可以近乎即時地檢視應用程式的日誌訊息。您也可以設定指標和設定警示，以便在超過指標閾值時通知您。使用 CloudWatch Application Insights，您可以檢視自動或自訂儀表板，以顯示受監控應用程式的潛在問題。CloudWatch Application Insights 旨在協助您快速隔離應用程式和基礎設施的持續問題。

若要將日誌訊息寫入 CloudWatch Logs，請將 `AWS.Logger.NLog` NuGet 套件新增至 .NET 專案。然後，您更新 `NLog.config` 檔案以使用 CloudWatch Logs 做為目標。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 符合下列條件的 .NET Web 或主控台應用程式：
  - 使用支援的 .NET Framework 或 .NET Core 版本。如需詳細資訊，請參閱產品版本。
  - 使用 NLog 將日誌資料傳送至 Application Insights。
- 為 AWS 服務建立 IAM 角色的許可。如需詳細資訊，請參閱服務角色許可。
- 將角色傳遞至 AWS 服務的許可。如需詳細資訊，請參閱授予使用者將角色傳遞至 AWS 服務。

產品版本

- .NET Framework 3.5 版或更新版本
- .NET Core 1.0.1、2.0.0 或更新版本

Architecture

目標技術堆疊

- NLog
- Amazon CloudWatch Logs

目標架構

1. .NET 應用程式會將日誌資料寫入 NLog 記錄架構。
2. NLog 會將日誌資料寫入 CloudWatch Logs。
3. 您可以使用 CloudWatch 警示和自訂儀表板來監控 .NET 應用程式。

工具

AWS 服務

- Amazon CloudWatch Application Insights 可協助您觀察應用程式和基礎 AWS 資源的運作狀態。
- Amazon CloudWatch Logs 可協助您集中所有系統、應用程式和 AWS 服務的日誌，以便您可以監控日誌並將其安全地存檔。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Tools for PowerShell 是一組 PowerShell 模組，可協助您從 PowerShell 命令列對 AWS 資源執行指令碼操作。

其他工具

- Logger.NLog 是 NLog 目標，可將日誌資料記錄到 CloudWatch Logs。
- NLog 是 .NET 平台的開放原始碼記錄架構，可協助您將日誌資料寫入目標，例如資料庫、日誌檔案或主控台。
- PowerShell 是在 Windows、Linux 和 macOS 上執行的 Microsoft 自動化和組態管理程式。
- Visual Studio 是一種整合的開發環境 (IDE)，其中包含編譯器、程式碼完成工具、圖形設計師和其他支援軟體開發的功能。

最佳實務

- 設定目標日誌群組的保留政策。這必須在 NLog 組態之外完成。在預設情況下，日誌資料會無限期存放於 CloudWatch Logs。
- 遵守管理 AWS 存取金鑰的最佳實務。

史詩

設定存取和工具

任務	Description	所需的技能
建立 IAM 政策。	請遵循 IAM 文件中的使用 JSON 編輯器建立政策中的指示。輸入下列 JSON 政策，該政策具有允許 CloudWatch Logs 讀取和寫入日誌所需的最低權限許可。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "logs:CreateLogGroup", "logs:CreateLogStream", "logs:GetLogEvents", "logs:PutLogEvents", "logs:DescribeLogGroups", "logs:DescribeLogStreams", "logs:PutRetentionPolicy" ],`	AWS 管理員、AWS DevOps

任務	Description	所需的技能
	"Resource": ["*"] }] }	
建立 IAM 角色。	遵循 IAM 文件中建立角色以將許可委派給 AWS 服務中的指示。選取您先前建立的政策。這是 CloudWatch Logs 執行記錄動作所擔任的角色。	AWS 管理員、AWS DevOps
設定適用於 PowerShell 的 AWS 工具。	1. 請遵循安裝適用於 PowerShell 的 AWS 工具中的作業系統說明。 2. 使用適用於 PowerShell cmdlet 的 AWS 工具，將您的存取金鑰和私密金鑰存放在設定檔中。如需說明，請參閱 AWS Tools for PowerShell 文件中的管理設定檔。	一般 AWS

設定 NLog

任務	Description	所需的技能
安裝 NuGet 套件。	1. 在 Visual Studio 中，選擇檔案，然後選擇開啟專案或解決方案。 2. 選擇您要安裝 NLog 的專案。	應用程式開發人員

任務	Description	所需的技能
	3. 在 Visual Studio 中，選擇工具、NuGet 套件管理員、套件管理員主控台。 4. 輸入下列命令來安裝 `AWS.Logger.NLog` NuGet 套件。 `Install-Package AWS.Logger.NLog -Version 3.1.0`	

任務	Description	所需的技能
設定記錄目標。	1. 開啟 `NLog.config` 檔案。 2. 針對目標 `type`，輸入 `AWSTarget` 。 3. 針對目標 `logGroup`，輸入您要使用的日誌群組名稱。如果日誌群組尚不存在，會自動建立具有所提供名稱的新日誌群組。 4. 針對目標 `region`，輸入設定 CloudWatch Logs 的 AWS 區域。 5. 針對目標 `profile`，輸入您先前建立用來存放存取金鑰和私密金鑰的設定檔名稱。 6. 儲存並關閉 `NLog.config` 檔案。 如需範例組態檔案，請參閱此模式的其他資訊一節。當您執行應用程式時，NLog 會撰寫日誌訊息，並將其傳送至 CloudWatch Logs。	應用程式開發人員

驗證和監控日誌

任務	Description	所需的技能
驗證記錄。	遵循 CloudWatch Logs 文件中檢視傳送至 CloudWatch Logs 的日誌資料中的指示。CloudWatch 驗證日誌事件是否正在為 .NET 應用程式記	一般 AWS

任務	Description	所需的技能
	錄。如果未記錄日誌事件，請參閱此模式中的故障診斷一節。	
監控 .NET 應用程式堆疊。	根據您的使用案例，視需要在 CloudWatch 中設定監控。您可以使用 CloudWatch Logs Insights、CloudWatch Metrics Insights 和 CloudWatch Application Insights 來監控 .NET 工作負載。您也可以設定警示，以便接收警示，也可以建立自訂儀表板，從單一檢視監控工作負載。	一般 AWS

疑難排解

問題	解決方案
日誌資料不會顯示在 CloudWatch Logs 中。	請確定 IAM 政策已連接至 CloudWatch Logs 擔任的 IAM 角色。如需說明，請參閱《Epics》中的設定存取和工具一節。

相關資源

- 使用日誌群組和日誌串流 (CloudWatch Logs 文件)
- Amazon CloudWatch Logs 和 .NET Logging Framework (AWS 部落格文章)

其他資訊

以下是範例NLog.config檔案。

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
```

```
  <configSections>
    <section name="nlog" type="NLog.Config.ConfigSectionHandler, NLog" />
  </configSections>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
  </startup>
  <nlog>
    <extensions>
      <add assembly="NLog.AWS.Logger" />
    </extensions>
    <targets>
      <target name="aws" type="AWSTarget" logGroup="NLog.TestGroup" region="us-east-1"
 profile="demo"/>
    </targets>
    <rules>
      <logger name="*" minlevel="Info" writeTo="aws" />
    </rules>
  </nlog>
</configuration>
```

將 AWS Service Catalog 產品複製到不同的 AWS 帳戶和 AWS 區域

Sachin Vighe 和 Santosh Kale，Amazon Web Services

總結

AWS Service Catalog 是一項區域服務，這表示 AWS Service Catalog 產品組合和產品只能在建立它們的 AWS 區域中顯示。如果您在新區域中設定 AWS Service Catalog 中樞，則必須重新建立現有的產品，這可能會是耗時的程序。

此模式的方法透過描述如何將來源 AWS 帳戶或區域中的 AWS Service Catalog 中樞中的產品複製到目的地帳戶或區域中的新中樞，協助簡化此程序。如需 AWS Service Catalog 中樞和語音模型的詳細資訊，請參閱 AWS Service Catalog 中樞和語音模型：如何將 AWS Service Catalog 的部署和管理自動化至 AWS 管理和控管部落格上的許多帳戶。

模式也提供跨帳戶或其他區域複製 AWS Service Catalog 產品所需的個別程式碼套件。透過使用此模式，您的組織可以節省時間、在新的 AWS Service Catalog 中樞中提供現有和先前的產品版本、將手動錯誤的風險降至最低，以及將方法擴展到多個帳戶或區域。

> **Note**
>
> 此模式的 Epics 區段提供兩種複製 產品的選項。您可以使用選項 1 跨帳戶複製產品，或選擇選項 2 跨區域複製產品。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 來源帳戶或區域中現有的 AWS Service Catalog 產品。
- 目的地帳戶或區域中現有的 AWS Service Catalog 中樞。
- 如果您想要跨帳戶複製產品，您必須共用，然後將包含產品的 AWS Service Catalog 產品組合匯入目的地帳戶。如需詳細資訊，請參閱 AWS Service Catalog 文件中的共用和匯入產品組合。

限制

- 您要跨區域或帳戶複製的 AWS Service Catalog 產品不能屬於多個產品組合。

Architecture

下圖顯示將 AWS Service Catalog 產品從來源帳戶複製到目的地帳戶。

下圖顯示將 AWS Service Catalog 產品從來源區域複製到目的地區域。

技術堆疊

- Amazon CloudWatch
- AWS Identity and Access Management (IAM)
- AWS Lambda
- AWS Service Catalog

自動化和擴展

您可以使用 Lambda 函數來擴展此模式的方法，該函數可根據收到的請求數量或您需要複製的 AWS Service Catalog 產品數量進行擴展。如需詳細資訊，請參閱 AWS Lambda Lambda 文件中的 Lambda 函數擴展。

工具

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而不需要佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Service Catalog 可協助您集中管理針對 AWS 核准的 IT 服務目錄。最終使用者可在機構所設的限制範圍內，迅速地只部署自己需要且經核准的 IT 服務。

Code

您可以使用 `cross-account-copy`套件（已連接）跨帳戶複製 AWS Service Catalog 產品，或使用 `cross-region-copy`套件（已連接）跨區域複製產品。

`cross-account-copy` 套件包含下列檔案：

- copyconf.properties – 包含區域和 AWS 帳戶 ID 參數的組態檔案，用於跨帳戶複製產品。
- scProductCopyLambda.py – 用於跨帳戶複製產品的 Python 函數。
- createDestAccountRole.sh – 在目的地帳戶中建立 IAM 角色的指令碼。
- createSrcAccountRole.sh – 在來源帳戶中建立 IAM 角色的指令碼。
- copyProduct.sh – 用來建立和叫用 Lambda 函數以跨帳戶複製產品的指令碼。

cross-region-copy 套件包含下列檔案：

- copyconf.properties – 包含區域和 AWS 帳戶 ID 參數的組態檔案，用於跨區域複製產品。
- scProductCopyLambda.py – 跨區域複製產品的 Python 函數。
- copyProduct.sh – 用來建立 IAM 角色，以及建立和叫用 Lambda 函數以跨區域複製產品的指令碼。

史詩

選項 1 – 跨帳戶複製 AWS Service Catalog 產品

任務	Description	所需的技能
更新組態檔案。	1. 將 cross-account-copy 套件（已連接）下載到您的本機電腦。 2. 使用下列值更新copyconf.properties 組態檔案： • srcRegion – 提供包含產品的來源區域。 • destRegion – 提供產品的目的地區域。 • sourceAccountId – 提供來源帳戶的 AWS 帳戶 ID。 • destAccountId – 提供目的地帳戶的 AWS 帳戶 ID。	AWS 管理員、AWS 系統管理員、雲端管理員

任務	Description	所需的技能

在目的地帳戶中設定 AWS CLI 的登入資料。	執行 `aws configure` 命令並提供下列值，以設定您的登入資料來存取目的地帳戶中的 AWS CLI：
	
	`$aws configure`
	`AWS Access Key ID [None]: <your_access_key_id>`
	`AWS Secret Access Key [None]: <your_secret_access_key>`
	`Default region name [None]: Region`
	`Default output format [None]:`
	
	如需詳細資訊，請參閱 AWS Command Line Interface 文件中的組態基本概念。
		AWS 管理員、AWS 系統管理員、雲端管理員

任務	Description	所需的技能

在來源帳戶中設定 AWS CLI 的登入資料。	執行 `aws configure` 命令並提供下列值，以設定您的登入資料來存取來源帳戶中的 AWS CLI：
	
	`$aws configure`
	`AWS Access Key ID`
	`[None]: <your_access_key_id>`
	`AWS Secret Access Key`
	`[None]: <your_secret_access_key>`
	`Default region name`
	`[None]: Region`
	`Default output format`
	`[None]:`
	
	如需詳細資訊，請參閱 AWS 命令列界面文件中的組態基本概念。
		AWS 管理員、AWS 系統管理員、雲端管理員

在目的地帳戶中建立 Lambda 執行角色。	在您的目的地帳戶中執行`createDestAccountRole.sh` 指令碼。指令碼會實作下列動作： • 在目的地帳戶中建立 Lambda 執行角色 • 建立並連接 Lambda 執行角色的 IAM 政策	AWS 管理員、AWS 系統管理員、雲端管理員

任務	Description	所需的技能
在來源帳戶中建立跨帳戶 IAM 角色。	在您的來源帳戶中執行`createSrcAccountRole.sh` 指令碼。指令碼會實作下列動作： • 在來源帳戶中建立跨帳戶 IAM 角色，由目的地帳戶中的 Lambda 執行角色擔任以複製產品 • 建立並連接來源帳戶中跨帳戶角色的 IAM 政策	AWS 管理員、AWS 系統管理員、雲端管理員
在目的地帳戶中執行 copyProduct 指令碼。	在目的地帳戶中執行`copyProduct.sh` 指令碼。指令碼會實作下列動作： • 建立並叫用 Lambda 函數，將產品從來源帳戶複製到目的地帳戶	AWS 管理員、AWS 系統管理員、雲端管理員

選項 2 – 將 AWS Service Catalog 產品從來源區域複製到目的地區域

任務	Description	所需的技能
更新組態檔案。	1. 將`cross-region-copy`套件（已連接）下載到您的本機電腦。 2. 使用下列值更新`copyconf.properties` 組態檔案： • `srcRegion` – 提供包含產品的來源區域。 • `destRegion` – 提供產品的目的地區域。	AWS 系統管理員、雲端管理員、AWS 管理員

任務	Description	所需的技能
	• `accountId` – 提供您的 AWS 帳戶 ID。	

設定 AWS CLI 的登入資料。	執行 `aws configure` 命令並提供下列值，以設定您的登入資料來存取您環境中的 AWS CLI：
	`$aws configure`
	`AWS Access Key ID [None]: <your_access_key_id>`
	`AWS Secret Access Key [None]: <your_secret_access_key>`
	`Default region name [None]: Region`
	`Default output format [None]:`
	如需詳細資訊，請參閱 AWS 命令列界面文件中的組態基本概念。
		AWS 管理員、AWS 系統管理員、雲端管理員

執行 copyProduct 指令碼。	在目的地區域中執行 `copyProduct.sh` 指令碼。指令碼會實作下列動作： • 建立 Lambda 執行角色 • 建立並連接 Lambda 執行角色的 IAM 政策 • 建立並叫用 Lambda 函數，將產品從來源區域複製到目的地區域	AWS 管理員、AWS 系統管理員、雲端管理員

相關資源

- 建立 Lambda 執行角色 (AWS Lambda 文件）
- 建立 Lambda 函數 (AWS Lambda 文件）
- AWS Service Catalog API 參考
- AWS Service Catalog 文件

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

為雲端操作模型建立 RACI 或 RASCI 矩陣

Teddy Germade、Jerome Descreux、Florian Leroux 和 Josselin LE MINEUR，Amazon Web Services

總結

Cloud Center of Excellence (CCoE) 或 CEE (Cloud Enablement Engine) 是一個強大且負責任的團隊，專注於為雲端做好營運準備。其主要重點是將資訊 IT 組織從內部部署操作模型轉換為雲端操作模型。CCoE 應該是一個跨職能團隊，其中包含來自基礎設施、應用程式、操作和安全性的表示。

雲端操作模型的其中一個關鍵元件是 RACI 矩陣或 RASCI 矩陣。這用於定義涉及遷移活動和雲端操作的所有各方的角色和責任。矩陣名稱衍生自矩陣中定義的責任類型：負責人 (R)、責任 (A)、支援 (S)、諮詢 (C) 和知情 (I)。支援類型為選用。如果您包含它，則稱為 RASCI 矩陣，如果您排除它，則稱為 RACI 矩陣。

從連接的範本開始，您的 CCoE 團隊可以為您的組織建立 RACI 或 RASCI 矩陣。範本包含雲端操作模型中常見的團隊、角色和任務。此矩陣的基礎是與操作整合和 CCoE 功能相關的任務。不過，您可以自訂此範本，以滿足組織結構和使用案例的需求。

RACI 矩陣的實作沒有限制。此方法適用於大型組織、新創公司及其間的所有項目。對於小型組織，相同的資源可以填滿多個角色。

史詩

建立矩陣

任務	Description	所需的技能
識別關鍵利益相關者。	識別與您雲端營運模型策略目標連結的關鍵服務和團隊經理。	專案經理
自訂矩陣範本。	在附件區段中下載範本，然後更新 RACI 或 RASCI 矩陣，如下所示： • 在雲端團隊工作表上，視需要為您的組織更新 CCoE 串	專案經理

任務	Description	所需的技能

	流名稱、團隊名稱和團隊描述。
	• 在雲端角色工作表上，視需要為您的組織更新角色、團隊名稱和角色描述。
	• 在 RASCI 工作表上，視需要為您的組織更新下列項目：
	  • 在資料列 1 和資料欄 A 中，更新 CCoE 串流。
	  • 在第 2 列中，更新團隊名稱。
	  • 在第 3 列中，更新角色名稱。
	  • 在資料欄 D 和 E 中，更新您要包含在 RASCI 圖表中的一般欄位和活動。	

規劃會議。	1. 向所有利益相關者傳達 RASCI 目標。 2. 規劃一或多個會議，讓每個團隊的獲授權代表可以參加。	專案經理

任務	Description	所需的技能
完成矩陣。	在與所有利益相關者的會議中，執行下列動作： 1. 確認每個團隊的代表都存在。團隊參與是強制性的，因此您可以準確地為每個任務指派責任類型。 2. 與參與者一起檢閱什麼是 RASCI 矩陣和目標。 3. 與參與者一起檢閱共同的責任模型，讓他們了解其組織在雲端中安全的責任範圍。 4. 在 RASCI 工作表上，針對每個任務或活動，完成 F 欄到 AN 欄，以指派下列責任類型： • 負責任 (R) – 此角色負責執行工作以完成任務。 • 問責 (A) – 此角色負責確保任務已完成。此角色也負責確保符合先決條件，並將任務委派給負責的人員。 • 支援 (S) – 此角色可協助負責完成任務的人員。此責任類型是選用的，您可以選擇將其排除，以建立更傳統的 RACI 矩陣。 • 已諮詢 (C) – 應諮詢此角色以取得有關任務的意見或專業知識。視任務而定，可能不需要此責任類型。	專案經理

任務	Description	所需的技能
	• 通知 (I) – 此角色應保持在任務進度的最新狀態，並在任務完成時收到通知。 • 空白 – 此角色不參與活動或任務。	
共用 RASCI 矩陣。	當 RACI 或 RASCI 矩陣完成時，讓領導層核准。將它儲存在共用儲存庫或所有利益相關者都可以存取的中央位置。我們建議您使用標準文件控制程序來記錄和核准矩陣的修訂。	專案經理

相關資源

- AWS 共同責任模型

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Amazon CloudWatch 異常偵測為自訂指標建立警示

Ram Kandaswamy 和 Raheem Jiwani，Amazon Web Services

總結

在 Amazon Web Services (AWS) 雲端上，您可以使用 Amazon CloudWatch 建立警示，以監控指標並傳送通知，或在超過閾值時自動進行變更。

若要避免受到靜態閾值的限制，您可以根據過去的模式建立警示，並在特定指標超出正常操作時段時通知您。例如，您可以從 Amazon API Gateway 監控 API 的回應時間，並接收有關無法滿足服務層級協議 (SLA) 的異常通知。

此模式說明如何針對自訂指標使用 CloudWatch 異常偵測。模式說明如何在 Amazon CloudWatch Logs Insights 中建立自訂指標，或使用 AWS Lambda 函數發佈自訂指標，然後使用 Amazon Simple Notification Service (Amazon SNS) 設定異常偵測和建立通知。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 現有的 SNS 主題，設定為傳送電子郵件通知。如需詳細資訊，請參閱《Amazon SNS 文件》中的 Amazon SNS 入門。
- 使用 CloudWatch Logs 設定的現有應用程式。

限制

- CloudWatch 指標不支援毫秒時間間隔。如需一般和自訂指標精細程度的詳細資訊，請參閱 Amazon CloudWatch FAQs。

Architecture

該圖顯示以下工作流程：

1. 使用 CloudWatch Logs 建立和更新的指標的日誌會串流至 CloudWatch。
2. 警示會根據閾值啟動，並將警示傳送至 SNS 主題。
3. Amazon SNS 會傳送電子郵件通知給您。

技術堆疊

- CloudWatch
- AWS Lambda
- Amazon SNS

工具

- Amazon CloudWatch 提供可靠、可擴展且靈活的監控解決方案。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而無需佈建或管理伺服器。
- Amazon Simple Notification Service (Amazon SNS) 是一種受管服務，可將訊息從發佈者交付給訂閱者。

史詩

設定自訂指標的異常偵測

任務	Description	所需的技能
選項 1 - 使用 Lambda 函數建立自訂指標。	下載 `lambda_function.py` 檔案（已連接），然後取代 AWS 文件 GitHub 上 aws-lambda-developer-guide 儲存庫中的範例`lambda_function.py` 檔案。這為您提供了將自訂指標傳送至 CloudWatch Logs 的範例 Lambda 函數。Lambda 函數使用 Boto3 API 與 CloudWatch 整合。 執行 Lambda 函數之後，您可以登入 AWS 管理主控台、開啟 CloudWatch 主控台，而且已發佈的指標可在已發佈的命名空間下使用。	DevOps 工程師，AWS DevOps

任務	Description	所需的技能
選項 2 – 從 CloudWatch 日誌群組建立自訂指標。	登入 AWS 管理主控台，開啟 CloudWatch 主控台，然後選擇日誌群組。選擇您要為其建立指標的日誌群組。 選擇動作，然後選擇建立指標篩選條件。針對篩選條件模式，輸入您要使用的篩選條件模式。如需詳細資訊，請參閱 CloudWatch 文件中的篩選和模式語法。 若要測試篩選條件模式，請在測試模式下輸入一或多個日誌事件。每個日誌事件都必須在一行內，因為 Log event messages (日誌事件訊息) 方塊中使用換行來分隔日誌事件。測試模式之後，您可以在指標詳細資訊下輸入指標的名稱和值。 如需建立自訂指標的詳細資訊和步驟，請參閱 CloudWatch 文件中的為日誌群組建立指標篩選條件。	DevOps 工程師，AWS DevOps

任務	Description	所需的技能
為您的自訂指標建立警示。	在 CloudWatch 主控台上，選擇警示，然後選擇建立警示。選擇選取指標，然後在搜尋方塊中輸入您先前建立的指標名稱。選擇圖形化指標索引標籤，並根據您的需求設定選項。 在條件下，選擇異常偵測，而非靜態閾值。這會顯示以兩個標準預設偏差為基礎的頻帶。您可以設定閾值，並根據需求調整閾值。 選擇下一步。 **Note** 頻帶是動態的，取決於資料點的品質。當您開始彙總更多資料時，頻帶和閾值會自動更新。	DevOps 工程師，AWS DevOps

任務	Description	所需的技能
設定 SNS 通知。	在通知下，選擇要在警示處於ALARM狀態、OK狀態或INSUFFICIENT_DATA 狀態時通知的 SNS 主題。 若要讓警示針對相同的警示狀態或不同警示狀態傳送多個通知，請選擇 Add notification (新增通知)。選擇下一步。輸入警示的名稱與說明。名稱只能包含 ASCII 字元。然後選擇下一步。 在預覽和建立下，確認資訊和條件正確無誤，然後選擇建立警示。	DevOps 工程師，AWS DevOps

相關資源

- 將自訂指標發佈至 CloudWatch
- 使用 CloudWatch 異常偵測
- 警示事件和 Amazon EventBridge
- 將自訂指標推送至 Cloud Watch 時，應遵循哪些最佳實務？（影片）
- CloudWatch Application Insights 簡介 （影片）
- 使用 CloudWatch（影片）偵測異常

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

建立使用具有預設加密之 Amazon EBS 磁碟區的 AWS Cloud9 IDE

Janardhan Malyala 和 Dhrubajyoti Mukherjee，Amazon Web Services

總結

注意： AWS Cloud9 不再提供給新客戶。的現有客戶 AWS Cloud9 可以繼續正常使用服務。進一步了解

您可以在 Amazon Web Services (AWS) 雲端上使用預設加密，強制加密 Amazon Elastic Block Store (Amazon EBS) 磁碟區和快照副本。

您可以建立使用預設加密之 EBS 磁碟區的 AWS Cloud9 整合開發環境 (IDE)。不過，AWS Cloud9 的 AWS Identity and Access Management (IAM) 服務連結角色需要存取這些 EBS 磁碟區的 AWS Key Management Service (AWS KMS) 金鑰。 AWS Cloud9 如果未提供存取權，AWS Cloud9 IDE 可能無法啟動，且偵錯可能很困難。

此模式提供將 AWS Cloud9 的服務連結角色新增至 EBS 磁碟區所使用的 AWS KMS 金鑰的步驟。此模式描述的設定可協助您成功建立和啟動 IDE，該 IDE 預設使用具有加密功能的 EBS 磁碟區。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- EBS 磁碟區的預設加密已開啟。如需預設加密的詳細資訊，請參閱《Amazon Elastic Compute Cloud (Amazon EC2) 文件》中的 Amazon EBS 加密。Amazon EC2
- 用於加密 EBS 磁碟區的現有客戶受管 KMS 金鑰。

> **Note**
>
> 您不需要為 AWS Cloud9 建立服務連結角色。當您建立 AWS Cloud9 開發環境時，AWS Cloud9 會為您建立服務連結角色。

Architecture

技術堆疊

- AWS Cloud9
- IAM
- AWS KMS

工具

- AWS Cloud9 是整合式開發環境 (IDE)，可協助您編寫、建置、執行、測試和偵錯軟體。它還可協助您將軟體發佈至 AWS 雲端。
- Amazon Elastic Block Store (Amazon EBS) 提供區塊層級儲存磁碟區，可與 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體搭配使用。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。

史詩

尋找預設加密金鑰值

任務	Description	所需的技能
記錄 EBS 磁碟區的預設加密金鑰值。	登入 AWS 管理主控台並開啟 Amazon EC2 主控台。選擇 EC2 儀表板，然後在帳戶屬性中選擇資料保護和安全性。在 EBS 加密區段中，複製並記錄預設加密金鑰中的值。	雲端架構師、DevOps 工程師

提供 AWS KMS 金鑰的存取權

任務	Description	所需的技能
提供 AWS Cloud9 存取 EBS 磁碟區的 KMS 金鑰。	1. 開啟 AWS KMS 主控台，然後選擇客戶受管金鑰。選取用於 Amazon EBS 加密	雲端架構師、DevOps 工程師

任務	Description	所需的技能
	的 AWS KMS 金鑰，然後選擇檢視金鑰。 2. 在金鑰政策索引標籤上，確認您可以看到金鑰政策的文字形式。如果您看不到文字表單，請選擇切換到政策檢視。 3. 選擇編輯。將其他資訊區段中的程式碼新增至政策，然後選擇儲存變更。政策變更允許 AWS Cloud9 的服務連結角色 `AWSServiceRoleForAWSCloud9` 存取金鑰。 如需更新金鑰政策的詳細資訊，請參閱如何變更金鑰政策 (AWS KMS 文件）。 ⚠ Important 當您啟動第一個 IDE 時，會自動建立 AWS Cloud9 的服務連結角色。如需詳細資訊，請參閱 AWS Cloud9 文件中的建立服務連結角色。	

建立和啟動 IDE

任務	Description	所需的技能
建立並啟動 AWS Cloud9 IDE。	開啟 AWS Cloud9 主控台，並依照 AWS Cloud9 文件中建立 EC2 環境的步驟，根據您的需求選擇建立 AWS Cloud9Configure IDE。	雲端架構師、DevOps 工程師

相關資源

- 加密 AWS Cloud9 使用的 EBS 磁碟區
- 為 AWS Cloud9 建立服務連結角色
- 在 AWS Cloud9 中建立 EC2 環境

其他資訊

AWS KMS 金鑰政策更新

使用您的 AWS 帳戶 ID 取代 <aws_accountid>。

```
{
            "Sid": "Allow use of the key",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::<aws_accountid>:role/aws-service-role/
cloud9.amazonaws.com/AWSServiceRoleForAWSCloud9"
            },
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:DescribeKey"
            ],
            "Resource": "*"
        },
        {
```

```
            "Sid": "Allow attachment of persistent resources",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::<aws_accountid>:role/aws-service-role/
cloud9.amazonaws.com/AWSServiceRoleForAWSCloud9"
            },
            "Action": [
                "kms:CreateGrant",
                "kms:ListGrants",
                "kms:RevokeGrant"
            ],
            "Resource": "*",
            "Condition": {
                "Bool": {
                    "kms:GrantIsForAWSResource": "true"
                }
            }
        }
```

使用跨帳戶金鑰

如果您想要使用跨帳戶 KMS 金鑰，您必須使用授權搭配 KMS 金鑰政策。這可讓跨帳戶存取 金鑰。在您用來建立 Cloud9 環境的相同帳戶中，在終端機中執行下列命令。

```
aws kms create-grant \
 --region <Region where Cloud9 environment is created> \
 --key-id <The cross-account KMS key ARN> \
 --grantee-principal arn:aws:iam::<The account where Cloud9 environment is
 created>:role/aws-service-role/cloud9.amazonaws.com/AWSServiceRoleForAWSCloud9 \
 --operations "Encrypt" "Decrypt" "ReEncryptFrom" "ReEncryptTo" "GenerateDataKey"
 "GenerateDataKeyWithoutPlaintext" "DescribeKey" "CreateGrant"
```

執行此命令後，您可以使用 EBS 加密搭配不同帳戶中的金鑰來建立 Cloud9 環境。

自動建立標籤型 Amazon CloudWatch 儀表板

Janak Vadaria、Vinodkumar Mandalapu 和 RAJNEESH TYAGI，Amazon Web Services

總結

手動建立不同的 Amazon CloudWatch 儀表板可能很耗時，尤其是當您必須建立和更新多個資源以自動擴展環境時。自動建立和更新 CloudWatch 儀表板的解決方案可以節省您的時間。此模式可協助您部署全自動化 AWS Cloud Development Kit (AWS CDK) 管道，以根據標籤變更事件建立和更新 AWS 資源的 CloudWatch 儀表板，以顯示 Golden Signals 指標。

在網站可靠性工程 (SRE) 中，黃金訊號是指一組全面的指標，可從使用者或消費者的角度提供服務的廣泛檢視。這些指標包含延遲、流量、錯誤和飽和。如需詳細資訊，請參閱 AWS 網站上的什麼是網站可靠性工程 (SRE)？。

此模式提供的解決方案是事件驅動的。部署後，它會持續監控標籤變更事件，並自動更新 CloudWatch 儀表板和警示。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- AWS Command Line Interface (AWS CLI)，已安裝並設定
- AWS CDK v2 的先決條件
- 上的引導環境 AWS
- Python 第 3 版
- AWS 適用於 Python 的 SDK (Boto3)，已安裝
- Node.js 第 18 版或更新版本
- 節點套件管理員 (npm)，已安裝並針對 設定 AWS CDK
- 中等 （ 層級 200) 熟悉 AWS CDK 和 AWS CodePipeline

限制

此解決方案目前僅針對下列 AWS 服務建立自動化儀表板：

- Amazon Relational Database Service (Amazon RDS)

- AWS Auto Scaling
- Amazon Simple Notification Service (Amazon SNS)
- Amazon DynamoDB
- AWS Lambda

Architecture

目標技術堆疊

- CloudWatch 儀表板
- CloudWatch 警示

目標架構

1. 已設定應用程式標籤或程式碼變更的 AWS 標籤變更事件會在 中啟動管道， AWS CodePipeline 以建置和部署更新的 CloudWatch 儀表板。
2. AWS CodeBuild 執行 Python 指令碼來尋找已設定標籤的資源，並將資源 IDs存放在 CodeBuild 環境中的本機檔案中。
3. CodeBuild 會執行 cdk 合成來產生部署 CloudWatch 儀表板和警示的 CloudFormation 範本。
4. CodePipeline 會將 CloudFormation 範本部署到指定的 AWS 帳戶 和 區域。
5. 堆疊成功 CloudFormation 部署後，您可以檢視 CloudWatch 儀表板和警示。

自動化和擴展

此解決方案已使用 自動化 AWS CDK。您可以在 Amazon CloudWatch 儲存庫上的 GitHub Golden Signals Dashboards 中找到程式碼。 Amazon CloudWatch 對於其他擴展和建立自訂儀表板，您可以設定多個標籤索引鍵和值。

工具

Amazon 服務

- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與各種來源的即時資料連線，包括 AWS Lambda 函數、使用 API 目的地的 HTTP 呼叫端點，或其他事件匯流排 AWS 帳戶。

- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

最佳實務

作為安全最佳實務，您可以對連接到管道的來源儲存庫使用加密和身分驗證。如需其他最佳實務，請參閱 CodePipeline 文件中的 CodePipeline 最佳實務和使用案例。 CodePipeline

史詩

設定和部署範例應用程式

任務	Description	所需的技能
設定和部署範例應用程式。	1. 使用 命令複製 GitHub 範例程式碼儲存庫： `git clone https://github.com/aws-samples/golden-signals-dashboards-sample-app` 2. 導覽至電腦上複製的儲存庫，然後使用您選擇的編輯器開啟 `src/project-settings.ts` 檔案。 3. 根據您的 AWS 資源標籤和應用程式映射變	AWS DevOps

任務	Description	所需的技能
	更projectSettings 常數值。 4. 設定 AWS_ACCOUNT 、 AWS_REGION 和 GS_DASHBOARD_INSTANCE 環境變數： • 將 AWS_ACCOUNT 設定為您帳戶的帳戶 ID AWS 。 • 將 AWS_REGION 設定為您要部署範例應用程式的區域。 • 根據您的開發環境prod，將 GS_DASHBOARD_INSTANCE 設定為devtest、 或 。（我們建議test使用此模式所述的測試程序。） 5. AWS CLI 使用您的 AWS 登入資料設定 。如需詳細資訊，請參閱 AWS CLI 文件中的使用命令設定和檢視組態設定。 6. 執行下列命令來部署 Golden Signals 儀表板範例應用程式： `sh deploy.sh`	

任務	Description	所需的技能
自動建立儀表板和警示。	部署範例應用程式後，您可以使用預期的標籤值來建立此解決方案支援的任何資源，這會自動建立指定的儀表板和警示。 若要測試此解決方案，請建立 AWS Lambda 函數： 1. 在您 AWS 區域 部署範例應用程式的 AWS 管理主控台中登入 。 2. 開啟位於 https：//https://console.aws.amazon.com/lambda/ 的 Lambda 主控台。 3. 選擇建立函數，然後輸入函數名稱。 4. 在進階設定窗格中，選取啟用標籤，然後選擇新增標籤。輸入下列索引鍵和值： • 索引鍵：`AutoDashboard` • 值：`True` 5. 選擇 Create function (建立函數)。 Lambda 函數會立即啟動程式碼管道，自動為該特定 Lambda 函數建立儀表板和警示。 6. 若要檢視自動化儀表板和警示，請開啟位於 https：//https://console.aw	AWS DevOps

任務	Description	所需的技能
	s.amazon.com/cloudwatch/ 的 CloudWatch 主控台。您可以檢視您在projectSettings 常數中指定的函數的自訂儀表板和警示（預設為 APP1-lambda)。 7. 選取 Lambda 函數的儀表板，以檢視在此解決方案中建立的其他自動化儀表板。 8. 針對其他 服務重複這些步驟，例如 Amazon RDS AWS Auto Scaling、Amazon SNS 和 DynamoDB，以產生相關聯的儀表板。如需 Amazon RDS 的範例，請參閱其他資訊一節。	

移除範例應用程式

任務	Description	所需的技能
移除 golden-signals-dashboard 建構。	1. 若要移除範例應用程式建立的所有 CloudFormation 堆疊，您必須重新設定 AWS_ACCOUNT 、 AWS_REGION 和 GS_DASHBOARD_INSTANCE 環境變數。destroy.sh 命令需要這些組態。 • AWS_ACCOUNT 是您帳戶的帳戶 ID AWS 。	AWS DevOps

任務	Description	所需的技能
	• AWS_REGION 是您部署範例應用程式的 區域。 • GS_DASHBOARD_INSTANCE 根據先前的設定prod，為 devtest、或 。 2. AWS CLI 使用您的 AWS 登入資料設定 。 3. 執行下列命令來移除範例應用程式和所有相關聯的 CloudFormation 堆疊： `sh destroy.sh`	

疑難排解

問題	解決方案
找不到 Python 命令 （請參閱第 8 findresources.sh 行）。	檢查 Python 安裝的版本。如果您已安裝 Python 第 3 版，請將 resources.sh 檔案第 8 行python3的 python取代為 ，然後再次執行 sh deploy.sh 命令以部署解決方案。

相關資源

- 引導 (AWS CDK 文件）
- 使用具名設定檔 (AWS CLI 文件）
- AWS CDK 研討會

其他資訊

下圖顯示在此解決方案中建立的 Amazon RDS 範例儀表板。

記錄您的 AWS 登陸區域設計

Michael Daehnert、Florian Langer 和 Michael Lodemann，Amazon Web Services

總結

登陸區域是架構良好的多帳戶環境，以安全和合規最佳實務為基礎。這是整個企業的容器，可存放所有組織單位 (OUs)、 AWS 帳戶使用者和其他資源。登陸區域可以擴展以符合任何大小企業的需求。AWS 有兩種建立登陸區域的選項：使用 的服務型登陸區域AWS Control Tower或您建置的自訂登陸區域。每個選項都需要不同層級 AWS 的知識。

AWS 建立 AWS Control Tower 以協助您透過自動設定登陸區域來節省時間。 AWS Control Tower 由管理 AWS ，並使用最佳實務和指導方針來協助您建立基礎環境。 AWS Control Tower 使用整合的服務，例如 AWS Service Catalog和 AWS Organizations，在您的登陸區域中佈建帳戶，並管理這些帳戶的存取權。

AWS 登陸區域專案的需求、實作詳細資訊和操作動作項目會有所不同。每個登陸區域實作都需要處理自訂層面。這包括（但不限於）如何處理存取管理、使用哪個技術堆疊，以及監控對卓越營運有何要求。此模式提供的範本可協助您記錄登陸區域專案。透過使用 範本，您可以更快速地記錄專案，並協助您的開發和營運團隊了解您的登陸區域。

先決條件和限制

限制

此模式不會描述什麼是登陸區域或如何實作登陸區域。如需這些主題的詳細資訊，請參閱相關資源一節。

史詩

建立設計文件

任務	Description	所需的技能
識別關鍵利益相關者。	識別與您的登陸區域連結的關鍵服務和團隊經理。	專案經理
自訂 範本。	在附件區段中下載範本，然後更新範本，如下所示：	專案經理

任務	Description	所需的技能
	1. 移除任何不適用於組織登陸區域或程序的區段。 2. 為您的組織新增任何唯一的區段。	
完成範本。	在與利益相關者的會議或使用write-and-review程序時，完成範本，如下所示： 1. 使用藍色方塊中的指引和資訊來完成每個區段。 2. 使用組織的自訂值取代或移除任何黃色欄位。 3. 使用自訂架構或流程圖取代或移除任何映像欄位。 4. 完成範本的修訂歷史記錄和貢獻者區段。	專案經理
共用設計文件。	當您的登陸區域設計文件完成後，請將其儲存在共用儲存庫或所有利益相關者都可以存取的中央位置。我們建議您使用標準文件控制程序來記錄和核准設計文件的修訂。	專案經理

相關資源

- AWS Control Tower 文件
  - 規劃您的 AWS Control Tower 登陸區域
  - AWSAWS Control Tower 登陸區域的多帳戶策略
  - 登陸區域設定的管理秘訣
  - 對登陸區域組態的期望
- 的自訂 AWS Control Tower(AWS 解決方案程式庫）

- 設定安全且可擴展的多帳戶 AWS 環境 (AWS 方案指引)

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 AWS CDK 跨多個 AWS 區域、帳戶和 OUs 啟用 Amazon DevOps Guru，以改善營運效能

Rahul Sharad Gaikwad 醫生，Amazon Web Services

摘要

此模式示範使用 TypeScript 中的 AWS 雲端開發套件 (AWS CDK)，跨多個 Amazon Web Services (AWS) 區域、帳戶和組織單位 (OUs) 啟用 Amazon DevOps Guru 服務的步驟。您可以使用 AWS CDK 堆疊從管理員（主要）AWS 帳戶部署 AWS CloudFormation StackSets，以跨多個帳戶啟用 Amazon DevOps Guru，而不是登入每個帳戶，並為每個帳戶個別啟用 DevOps Guru。

Amazon DevOps Guru 提供人工智慧操作 (AIOps) 功能，可協助您改善應用程式的可用性，並更快速地解決操作問題。DevOps Guru 透過套用機器學習 (ML) 支援的建議來減少手動工作量，而不需要任何 ML 專業知識。DevOps Guru 會分析您的資源和操作資料。如果偵測到任何異常，它會提供指標、事件和建議，以協助您解決問題。

此模式說明啟用 Amazon DevOps Guru 的三個部署選項：

- 對於跨多個帳戶和區域的所有堆疊資源
- 對於跨 OUs 的所有堆疊資源
- 對於跨多個帳戶和區域的特定堆疊資源

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 安裝並設定 AWS Command Line Interface (AWS CLI)。（請參閱 AWS CLI 文件中的安裝、更新和解除安裝 AWS CLI。）
- 安裝並設定 AWS CDK Toolkit。（請參閱 AWS CDK 文件中的 AWS CDK Toolkit。）
- Node Package Manager (npm)，已安裝並設定用於 TypeScript 中的 AWS CDK。（請參閱 npm 文件中的下載和安裝 Node.js 和 npm。）
- 已安裝並設定 Python3，用於執行 Python 指令碼將流量注入範例無伺服器應用程式。（請參閱 Python 文件中的 Python 設定和用量。）
- Pip，已安裝並設定為安裝 Python 請求程式庫。（請參閱 PyPI 網站上的 pip 安裝說明。）

產品版本

- AWS CDK Toolkit 1.107.0 版或更新版本
- npm 7.9.0 版或更新版本
- Node.js 15.3.0 版或更新版本

Architecture

技術

此模式的架構包含下列服務：

- Amazon DevOps Guru
- AWS CloudFormation
- Amazon API Gateway
- AWS Lambda
- Amazon DynamoDB
- Amazon CloudWatch
- AWS CloudTrail

AWS CDK 堆疊

模式使用下列 AWS CDK 堆疊：

- `CdkStackSetAdminRole` – 建立 AWS Identity and Access Management (IAM) 管理員角色，以在管理員和目標帳戶之間建立信任關係。
- `CdkStackSetExecRole` – 建立 IAM 角色以信任管理員帳戶。
- `CdkDevopsGuruStackMultiAccReg` – 跨多個 AWS 區域和帳戶為所有堆疊啟用 DevOps Guru，並設定 Amazon Simple Notification Service (Amazon SNS) 通知。
- `CdkDevopsGuruStackMultiAccRegSpecStacks` – 讓 DevOps Guru 跨多個 AWS 區域和帳戶進行特定堆疊，並設定 Amazon SNS 通知。
- `CdkDevopsguruStackOrgUnit` – 跨 OUs 啟用 DevOps Guru，並設定 Amazon SNS 通知。
- `CdkInfrastructureStack` – 在管理員帳戶中部署範例無伺服器應用程式元件，例如 API Gateway、Lambda 和 DynamoDB，以示範錯誤注入和洞見產生。

範例應用程式架構

下圖說明跨多個帳戶和區域部署的範例無伺服器應用程式的架構。模式會使用管理員帳戶來部署所有 AWS CDK 堆疊。它也會使用管理員帳戶做為設定 DevOps Guru 的目標帳戶之一。

1. 啟用 DevOps Guru 時，它會先將每個資源的行為建立基準，然後從 CloudWatch 提供的指標擷取操作資料。
2. 如果偵測到異常，它會將其與 CloudTrail 中的事件建立關聯，並產生洞見。
3. 洞見提供相關的事件序列以及規定的建議，讓運算子能夠識別犯罪資源。
4. Amazon SNS 會將通知訊息傳送至運算子。

自動化和擴展

此模式隨附的 GitHub 儲存庫使用 AWS CDK 做為基礎設施做為程式碼 (IaC) 工具，來建立此架構的組態。AWS CDK 可協助您協調資源，並跨多個 AWS 帳戶、區域和 OUs 啟用 DevOps Guru。

工具

AWS 服務

- AWS CDK – AWS Cloud Development Kit (AWS CDK) 可協助您以五種支援的程式設計語言之一將雲端基礎設施定義為程式碼：TypeScript、JavaScript、Python、Java 和 C#。
- AWS CLI – AWS 命令列界面 (AWS CLI) 是一種統一的工具，可提供一致的命令列界面，以便與 AWS 服務和資源互動。

Code

此模式的原始碼可在 Amazon DevOps Guru CDK 範例儲存庫的 GitHub 上取得。AWS CDK 程式碼是以 TypeScript 撰寫。若要複製和使用儲存庫，請遵循下一節中的指示。

> **Important**
>
> 此模式中的一些案例包括針對 Unix、Linux 和 macOS 格式化的 AWS CDK 和 AWS CLI 命令範例。對於 Windows，將每一行結尾的反斜線 (\) 接續字元替換為插入符號 (^)。

史詩

準備 AWS 資源以進行部署

任務	Description	所需的技能

設定名為 的 AWS 設定檔。	如下所示設定您的 AWS 命名設定檔，以在多帳戶環境中部署堆疊。
	
	對於管理員帳戶：
	
	`$aws configure --profile administrator`
	`AWS Access Key ID [****]: <your-administrator-access-key-ID>`
	`AWS Secret Access Key [****]: <your-administrator-secret-access-key>`
	`Default region name [None]: <your-administrator-region>`
	`Default output format [None]: json`
	
	對於目標帳戶：
	
	`$aws configure --profile target`
	`AWS Access Key ID [****: <your-target-access-key-ID>`
	`AWS Secret Access Key [****]: <your-target-secret-access-key>`
	`Default region name [None]: <your-target-region>`
		DevOps 工程師

任務	Description	所需的技能
	`Default output format [None]: json` 如需詳細資訊，請參閱 AWS CLI 文件中的使用具名設定檔。	
驗證 AWS 設定檔組態。	（選用）您可以遵循 AWS CLI 文件中的設定和檢視組態設定中的指示，在 `credentials` 和 `config` 檔案中驗證您的 AWS 設定檔組態。 https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-files.html#cli-configure-files-methods	DevOps 工程師
驗證 AWS CDK 版本。	執行下列命令來驗證 AWS CDK Toolkit 的版本： `$cdk --version` 此模式需要 1.107.0 版或更新版本。如果您有舊版的 AWS CDK，請遵循 AWS CDK 文件中的指示進行更新。	DevOps 工程師
複製專案程式碼。	使用 命令複製此模式的 GitHub 儲存庫： `$git clone https://github.com/aws-samples/amazon-devopsguru-cdk-samples.git`	DevOps 工程師

任務	Description	所需的技能

安裝套件相依性並編譯 TypeScript 檔案。	安裝套件相依性，並執行下列命令編譯 TypeScript 檔案：
	`$cd amazon-devopsguru-cdk-samples`
	`$npm install`
	`$npm fund`
	這些命令會從範例儲存庫安裝所有套件。
	Important
	如果您收到有關遺失套件的任何錯誤，請使用下列其中一個命令：
	`$npm ci`
	—或—
	`$npm install -g @aws-cdk/<package-name>`
	您可以在 `/amazon-devopsguru-cdk-samples/package.json` 檔案的 `Dependencies` 區段中找到套件名稱和版本的清單。如需詳細資訊，請參閱 npm 文件中的 npm ci 和 npm 安裝。
		DevOps 工程師

建置（合成）AWS CDK 堆疊

任務	Description	所需的技能
設定 Amazon SNS 通知的電子郵件地址。	請依照下列步驟提供 Amazon SNS 通知的電子郵件地址： 1. 編輯檔案 `/amazon-devopsguru-cdk-samples/lib/cdk-devopsguru-multi-acc-reg-stack.ts` 和 `/amazon-devopsguru-cdk-samples/lib/cdk-devopsguru-org-uni-stack.ts` 。 2. 在 `DevOpsGuruTopic Subscription` 區段中，使用您的電子郵件地址更新 `Endpoint` 參數。 3. 儲存並關閉檔案。	DevOps 工程師

建置專案程式碼。	執行 命令來建置專案程式碼並合成堆疊：
	`npm run build && cdk synth`
	您應該會看到類似下列的輸出：
	`$npm run build && cdk synth`
	`> cdk-devopsguru@0.1.0 build`
	`> tsc`
		DevOps 工程師

任務	Description	所需的技能
	`Successfully synthesized to ~/amazon-devopsguru-cdk-samples/cdk.out Supply a stack id (CdkDevopsGuruStackMultiAccReg,CdkDevopsGuruStackMultiAccRegSpecStacks, CdkDevopsguruStackOrgUnit, CdkInfrastructureStack, CdkStackSetAdminRole, CdkStackSetExecRole) to display its template.` 如需詳細資訊和步驟，請參閱 AWS CDK 文件中的您的第一個 AWS CDK 應用程式。	

列出 AWS CDK 堆疊。	執行下列命令以列出所有 AWS CDK 堆疊：
	
	`$cdk list`
	
	命令會顯示下列清單：
	
	`CdkDevopsGuruStackMultiAccReg CdkDevopsGuruStackMultiAccRegSpecStacks CdkDevopsguruStackOrgUnit CdkInfrastructureStack CdkStackSetAdminRole CdkStackSetExecRole`
		DevOps 工程師

選項 1 - 為跨多個帳戶的所有堆疊資源啟用 DevOps Guru

任務	Description	所需的技能
部署 AWS CDK 堆疊以建立 IAM 角色。	此模式使用 AWS CloudFormation StackSets 跨多個帳戶執行堆疊操作。如果您要建立第一個堆疊集，則必須建立下列 IAM 角色，才能在 AWS 帳戶中設定必要的許可： • `AWSCloudFormationStackSetAdministrationRole` • `AWSCloudFormationStackSetExecutionRole` **Note** 角色必須具有這些確切名稱。 1. 執行下列 CLI 命令，在管理員（主要）帳戶中建立 IAM `AWSCloudFormationStackSetAdministrationRole` 角色： `$cdk deploy CdkStackSetAdminRole --profile administrator` 2. 在您要執行堆疊執行個體的所有目標帳戶中建立 IAM	DevOps 工程師

任務	Description	所需的技能

	AWSCloudFormationStackSetExecutionRole 角色。若要建立此角色，請執行下列 CLI 命令：
	
	`$cdk deploy CdkStackSetExecRole \`
	`  --parameters AdministratorAccountId=<administrator-account-ID> \`
	`  --profile administrator`
	
	`$cdk deploy CdkStackSetExecRole \`
	`  --parameters AdministratorAccountId=<administrator-account-ID> \`
	`  --profile target`
	
	如需詳細資訊，請參閱 AWS CloudFormation 文件中的授予自我管理許可。	

任務	Description	所需的技能
部署 AWS CDK 堆疊以跨多個帳戶啟用 DevOps Guru。	AWS CDK CdkDevopsGuruStackMultiAccReg 堆疊會建立堆疊集，以跨多個帳戶和區域部署堆疊執行個體。若要部署堆疊，請使用指定的參數執行下列 CLI 命令： `$cdk deploy CdkDevopsGuruStackMultiAccReg \ --profile administrator \ --parameters AdministratorAccountId=<administrator-account-ID> \ --parameters TargetAccountId=<target-account-ID> \ --parameters RegionIds="<region-1>,<region-2>"` Amazon DevOps Guru 目前可在 DevOps Guru 常見問答集中列出的 AWS 區域中使用。	DevOps 工程師

選項 2 - 為跨 OUs 的所有堆疊資源啟用 DevOps Guru

任務	Description	所需的技能
擷取 OU IDs。	在 AWS Organizations 主控台上，識別您要啟用 DevOps Guru 之組織單位IDs。	DevOps 工程師

任務	Description	所需的技能
啟用 OUs 的服務受管許可。	如果您使用 AWS Organizations 進行帳戶管理，則必須授予服務受管許可以啟用 DevOps Guru。使用組織型受信任存取和服務連結角色 (SLRs)，而不是手動建立 IAM 角色。	DevOps 工程師
部署 AWS CDK 堆疊以跨 OUs 啟用 DevOps Guru。	AWS CDK `CdkDevopsguruStackOrgUnit` 堆疊可跨 OUs 啟用 DevOps Guru 服務。若要部署堆疊，請使用指定的參數執行下列命令： `$cdk deploy CdkDevopsguruStackOrgUnit \ --profile administrator \ --parameters RegionIds="<region-1>,<region-2>" \ --parameters OrganizationalUnitIds="<OU-1>,<OU-2>"`	DevOps 工程師

選項 3 - 為多個帳戶的特定堆疊資源啟用 DevOps Guru

任務	Description	所需的技能
部署 AWS CDK 堆疊以建立 IAM 角色。	如果您尚未建立第一個選項中顯示的必要 IAM 角色，請先執行此操作： 1. 執行下列 CLI 命令，在管理員（主要）帳戶中建立 IAM `AWSCloudF`	DevOps 工程師

任務	Description	所需的技能

	ormationStackSetAdministrationRole 角色：
	
	``
	$cdk deploy CdkStackSetAdminRole --profile administrator
	``
	
	2. 在您要執行堆疊執行個體的所有目標帳戶中建立 IAM AWSCloudFormationStackSetExecutionRole 角色。若要建立此角色，請執行 CLI 命令：
	
	``
	$cdk deploy CdkStackSetExecRole \
	  --parameters AdministratorAccountId=<administrator-account-ID> \
	  --profile administrator
	
	$cdk deploy CdkStackSetExecRole \
	  --parameters AdministratorAccountId=<administrator-account-ID> \
	  --profile target
	``
	
	如需詳細資訊，請參閱 AWS CloudFormation 文件中的授予自我管理許可。	

任務	Description	所需的技能
刪除現有的堆疊。	如果您已使用第一個選項為所有堆疊資源啟用 DevOps Guru，您可以使用下列命令刪除舊堆疊： `$cdk destroy CdkDevopsGuruStackMultiAccReg --profile administrator` 或者，您可以在重新部署堆疊時變更 `RegionIds` 參數，以避免堆疊已存在錯誤。	DevOps 工程師
使用堆疊清單更新 AWS CDK 堆疊。	1. 編輯 `/amazon-devopsguru-cdk-samples/lib/cdk-devopsguru-multi-acc-reg-spec-stack.ts` 檔案。 2. 在 `Resources`、`CloudFormation`、下`StackNames`，列出您要啟用 DevOps Guru 的堆疊。基於示範目的，參數會指定`CdkInfrastructureStack` 堆疊，但您可以根據您的需求編輯此項目。 3. 儲存並關閉檔案。 4. 若要合成和更新堆疊範本，請執行： `$cdk synth`	資料工程師

任務	Description	所需的技能
部署 AWS CDK 堆疊，以便在多個帳戶中為特定堆疊資源啟用 DevOps Guru。	AWS CDK `CdkDevopsGuruStackMultiAccRegSpecStacks` 堆疊可讓 DevOps Guru 跨多個帳戶進行特定堆疊資源。若要部署堆疊，請執行下列命令： `$cdk deploy CdkDevopsGuruStackMultiAccRegSpecStacks \ --profile administrator \ --parameters AdministratorAccountId=<administrator-account-ID> \ --parameters TargetAccountId=<target-account-ID> \ --parameters RegionIds="<region-1>,<region-2>"` **Note** 如果您先前已針對選項 1 部署此堆疊，請變更 `RegionIds` 參數（請務必從可用區域中選擇），以避免堆疊已存在錯誤。	DevOps 工程師

部署 AWS CDK 基礎設施堆疊

任務	Description	所需的技能
部署範例無伺服器基礎設施堆疊。	AWS CDK `CdkInfrastructureStack` 堆疊會部署無伺服器元件，例如 API Gateway、Lambda 和 DynamoDB 資料表，以示範 DevOps Guru 洞見。若要部署堆疊，請執行下列命令： `$cdk deploy CdkInfrastructureStack --profile administrator`	DevOps 工程師

在 DynamoDB 中插入範例記錄。	執行下列命令，將範例記錄填入 DynamoDB 資料表。提供`populate-shops-dynamodb-table.json` 指令碼的正確路徑。
	`$aws dynamodb batch-write-item \ --request-items file://scripts/populate-shops-dynamodb-table.json \ --profile administrator`
	該命令會顯示下列輸出：
	`{ "UnprocessedItems": {} }`
		DevOps 工程師

任務	Description	所需的技能

驗證在 DynamoDB 中插入的記錄。	若要驗證 DynamoDB 資料表是否包含 `populate-shops-dynamodb-table.json` 檔案的範例記錄，請存取 `ListRestApiEndpointMonitorOperator` API 的 URL，這會發佈為 AWS CDK 堆疊的輸出。您也可以在`CdkInfrastructureStack` 堆疊的 AWS CloudFormation 主控台的輸出索引標籤中找到此 URL。AWS CDK 輸出看起來會類似以下內容：
	
	`CdkInfrastructureStack.CreateRestApiMonitorOperatorEndpointD1D00045 = https://oure17c5vob.execute-api.<your-region>.amazonaws.com/prod/`
	
	`CdkInfrastructureStack.ListRestApiMonitorOperatorEndpointABBDB8D8 = https://cdff8icfrn4.execute-api.<your-region>.amazonaws.com/prod/`
		DevOps 工程師

任務	Description	所需的技能
等待資源完成底線。	此無伺服器堆疊有幾個資源。建議您等待 2 小時，然後再執行後續步驟。如果您在生產環境中部署此堆疊，最多可能需要 24 小時才能完成基準化，具體取決於您選取要在 DevOps Guru 中監控的資源數量。	DevOps 工程師

產生 DevOps Guru 洞見

任務	Description	所需的技能

更新 AWS CDK 基礎設施堆疊。	若要試用 DevOps Guru 洞見，您可以進行一些組態變更，以重現典型的操作問題。
	1. 編輯 `/amazon-devopsguru-cdk-samples/lib/infrastructure-stack.ts` 檔案。
	2. 在 `DDB Table`區段中，將 DynamoDB 資料表的讀取容量從 5 變更為 1。
	3. 儲存並關閉檔案。
	4. 執行下列命令來合成和部署更新的 AWS CDK 基礎設施堆疊：
	`$cdk synth`
	`$cdk deploy CdkInfrastructureStack --profile administrator`
		DevOps 工程師

任務	Description	所需的技能
在 API 上注入 HTTP 請求。	在 `ListRestApiMonitorOperatorEndpointxxxx` API 上以 HTTP 請求的形式注入輸入流量： 1. 編輯 Python 指令碼 `/amazon-devopsguru-cdk-samples/scripts/sendAPIRequest.py` 。 2. 使用 的 API 連結更新`url`變數`ListRestApiMonitorOperatorEndpointxxxx` 。您可以在 AWS CDK 部署命令的輸出或 AWS Cloudformation 主控台的堆疊的輸出索引標籤中找到此 URL。 3. 儲存並關閉檔案。 4. 使用 命令執行 Python 指令碼： `$python sendAPIRequest.py` 5. 請確定您取得 200 狀態碼。 6. 您可能需要透過多個 （最好是四個） 終端機執行指令碼，才能以高速率注入流量。 7. 指令碼在迴圈中執行約 10 分鐘後，您可以看到 DevOps Guru 主控台的操作洞見。	DevOps 工程師

任務	Description	所需的技能
檢閱 DevOps Guru 洞察。	在標準條件下，DevOps Guru 儀表板會在持續洞察計數器中顯示零。如果偵測到異常，它會以洞見的形式發出提醒。在導覽窗格中，選擇 Insights 以查看異常的詳細資訊，包括概觀、彙總指標、相關事件和建議。如需檢閱洞見的詳細資訊，請參閱使用 Amazon DevOps Guru 透過 AIOps 取得營運洞見部落格文章。	DevOps 工程師

清除

任務	Description	所需的技能

清除和刪除資源。	完成此模式之後，您應該移除您建立的資源，以避免產生任何進一步的費用。執行這些命令：
	
	`$cdk destroy CdkDevops GuruStackMultiAccR eg --profile administr ator`
	`$cdk destroy CdkDevops guruStackOrgUnit -- profile administrator`
	`$cdk destroy CdkDevops GuruStackMultiAccR egSpecStacks --profile administrator`
	`$cdk destroy CdkInfras tructureStack -- profile administrator`
		DevOps 工程師

任務	Description	所需的技能

	`$cdk destroy CdkStackSetAdminRole --profile administrator`
	`$cdk destroy CdkStackSetExecRole --profile administrator`
	`$cdk destroy CdkStackSetExecRole --profile target`	

相關資源

- 使用 Amazon DevOps Guru 獲得 AIOps 的運作洞見
- 使用 AWS CloudFormation StackSets 輕鬆跨多個帳戶和區域設定 Amazon DevOps Guru
- DevOps Guru 研討會

使用 Account Factory for Terraform 管理多個帳戶的許可集

Anand Krishna Varanasi 和 Siamak Heshmati，Amazon Web Services

總結

此模式可協助您將 AWS Control Tower Account Factory Terraform (AFT) 與 AWS IAM Identity Center 整合，以 AWS 帳戶 大規模設定多個 的許可。此方法使用自訂 AWS Lambda 函數來自動化以組織身分管理 AWS 帳戶 的 許可集指派。這可簡化程序，因為它不需要您平台工程團隊的手動介入。此解決方案可以增強營運效率、安全性和一致性。它提升了 的安全和標準化加入程序 AWS Control Tower，讓企業無法取代其雲端基礎設施的靈活性和可靠性。

先決條件和限制

先決條件

- AWS 帳戶，透過 管理 AWS Control Tower。如需詳細資訊，請參閱 入門 AWS Control Tower。
- 適用於 Terraform 的 Account Factory，部署在您環境中的專用帳戶中。如需詳細資訊，請參閱部署適用於 Terraform AWS Control Tower 的帳戶工廠。
- IAM Identity Center 執行個體，在您的環境中設定。如需詳細資訊，請參閱 IAM Identity Center 入門。
- 已設定的作用中 IAM Identity Center 群組。 如需詳細資訊，請參閱將群組新增至您的 IAM Identity Center 目錄 。
- Python 3.9 版或更新版本，已安裝

限制

- 此解決方案只能與透過 管理的帳戶搭配使用 AWS Control Tower。此解決方案是透過使用 Account Factory for Terraform 進行部署。
- 此模式不包含使用身分來源設定聯合身分的指示。如需如何完成此設定的詳細資訊，請參閱 IAM Identity Center 文件中的 IAM Identity Center 身分來源教學課程。

Architecture

AFT 概觀

AFT 會設定 Terraform 管道，協助您在其中佈建和自訂帳戶 AWS Control Tower。AFT 遵循可自動化帳戶佈建程序的 GitOps 模型 AWS Control Tower。您可以建立帳戶請求 Terraform 檔案並將其遞交至

儲存庫。這會啟動帳戶佈建的 AFT 工作流程。帳戶佈建完成後，AFT 可以自動執行其他自訂步驟。如需詳細資訊，請參閱 AWS Control Tower 文件中的 AFT 架構。

AFT 提供下列主要儲存庫：

- `aft-account-request` – 此儲存庫包含要建立或更新的 Terraform 程式碼 AWS 帳戶。
- `aft-account-customizations` – 此儲存庫包含 Terraform 程式碼，以根據每個帳戶建立或自訂資源。
- `aft-global-customizations` – 此儲存庫包含 Terraform 程式碼，可大規模為所有帳戶建立或自訂資源。
- `aft-account-provisioning-customizations` – 此儲存庫會管理僅套用至由 AFT 建立和使用 AFT 管理之特定帳戶的自訂。例如，您可以使用此儲存庫在 IAM Identity Center 中自訂使用者或群組指派，或自動關閉帳戶。

解決方案概觀

此自訂解決方案包含 AWS Step Functions 狀態機器和 AWS Lambda 函數，可將許可集指派給多個帳戶的使用者和群組。透過此模式部署的狀態機器會搭配預先存在的 AFT `aft_account_provisioning_customizations` 狀態機器運作。建立新的 時或帳戶建立後，使用者會提交更新 IAM Identity Center AWS 帳戶 使用者和群組指派的請求。他們會將變更推送至`aft-account-request`儲存庫來執行此操作。建立或更新帳戶的請求會在 Amazon DynamoDB Streams 中啟動串流。這會啟動 Lambda 函數，以更新目標的 IAM Identity Center 使用者和群組 AWS 帳戶。

以下是您可以在 Lambda 函數中提供的參數範例，用於將許可集指派至目標使用者和群組：

```
custom_fields = {
    "InstanceArn"         = "<Organization ID>",
    "PermissionSetArn"    = "<Permission set ARN>",
    "PrincipalId"         = "<Principal ID>",
  }
```

以下是此陳述式中的參數：

- `InstanceArn` – 組織的 Amazon Resource Name (ARN)
- `PermissionSetArn` – 許可集的 ARN
- `PrincipalId` – IAM Identity Center 中要套用許可集之使用者或群組的識別符

> **Note**
>
> 您必須先建立目標許可集、使用者和群組，才能執行此解決方案。

雖然InstanceArn值必須保持一致，但您可以修改 Lambda 函數，將多個許可集指派給多個目標身分。許可集的參數必須以 結尾PermissionSetArn，而使用者和群組的參數必須以 結尾PrincipalId。您必須定義兩個屬性。以下是如何定義多個許可集和目標使用者和群組的範例：

```
custom_fields = {
    "InstanceArn"                    = "<Organization ID>",
    "AdminAccessPermissionSetArn"    = "<Admin privileges permission set ARN>",
    "AdminAccessPrincipalId"         = "<Admin principal ID>",
    "ReadOnlyAccessPermissionSetArn" = "<Read-only privileges permission set ARN>",
    "ReadOnlyAccessPrincipalId"      = "<Read-only principal ID>",
  }
```

下圖顯示解決方案如何 AWS 帳戶 大規模更新目標中使用者和群組許可集step-by-step工作流程。當使用者啟動帳戶建立請求時，AFT 會啟動 aft-account-provisioning-framework Step Functions 狀態機器。此狀態機器會啟動 extract-alternate-sso Lambda 函數。Lambda 函數會將許可集指派給目標中的使用者和群組 AWS 帳戶。這些使用者或群組可以來自 IAM Identity Center 中的任何已設定身分來源。身分來源的範例包括 Okta、Active Directory 或 Ping Identity。

圖表顯示建立新帳戶時的下列工作流程：

1. 使用者將custom_fields變更推送至aft-account-request儲存庫。
2. AWS CodePipeline 會啟動將使用者定義中繼資料記錄到 Amazon DynamoDB aft-request-audit 資料表 AWS CodeBuild 的任務。此資料表具有屬性來記錄使用者定義的中繼資料。ddb_event_name 屬性會定義 AFT 操作的類型：
   - 如果值為 INSERT，則解決方案會在建立新的 時，將許可集指派給目標身分 AWS 帳戶 。
   - 如果值為 UPDATE，則解決方案會在 AWS 帳戶 建立 之後，將 許可集指派給目標身分。
3. Amazon DynamoDB Streams 會啟動 aft_alternate_sso_extract Lambda 函數。
4. aft_alternate_sso_extract Lambda 函數在 AWS Control Tower 管理帳戶中擔任 AWS Identity and Access Management (IAM) 角色。

5. Lambda 函數透過對 IAM Identity Center 進行 an 適用於 Python (Boto3) 的 AWS SDK create_account_assignment API 呼叫，將許可集指派給目標使用者和群組。它會從 Amazon DynamoDB `aft-request-audit` 資料表擷取許可集和身分指派。
6. 當 Step Functions 工作流程完成時，會將許可集指派給目標身分。

自動化和擴展

AFT 使用 CodePipeline AWS CodeBuild、DynamoDB 和高度可擴展的 Lambda AWS 服務 等大規模運作。如需其他自動化，您可以將此解決方案與票證或問題管理系統整合，例如 Jira。如需詳細資訊，請參閱此模式的其他資訊一節。

工具

AWS 服務

- Account Factory for Terraform (AFT) 是此解決方案中的主要工具。`aft-account-provisioning-customizations` 儲存庫包含用於建立自訂的 Terraform 程式碼 AWS 帳戶，例如自訂 IAM Identity Center 使用者或群組指派。
- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務 來建置業務關鍵型應用程式。

其他工具

- Python 是一種一般用途的電腦程式設計語言。
- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。

程式碼儲存庫

AFT 的程式碼儲存庫可在 GitHub AWS Control Tower Account Factory for Terraform 儲存庫中使用。此模式的程式碼可在 AWS 帳戶 Govern SSO 指派中使用 Account Factory for Terraform (AFT) 儲存庫。

最佳實務

- 了解AWS 共同的責任模型。
- 遵循 的安全建議 AWS Control Tower。如需詳細資訊，請參閱 中的安全性 AWS Control Tower。
- 遵循最低權限原則。如需詳細資訊，請參閱套用最低權限許可。
- 為群組和業務單位建置特定且著重的許可集和 IAM 角色。

史詩

部署解決方案

任務	Description	所需的技能

建立 IAM 角色。	在 AWS Control Tower 管理帳戶中，使用 Terraform 建立 IAM 角色。此角色具有跨帳戶存取和信任政策，允許來自身分提供者的聯合存取。它還具有透過 授予其他帳戶存取權的許可 AWS Control Tower。Lambda 函數將擔任此角色。請執行下列操作：
	1. 從 GitHub 程式碼儲存庫下載 `AFTCrossAccountRole.tf` 檔案。
	2. 視需要為您的 AWS 環境修改 `AFTCrossAccountRole.tf` 檔案。
	3. 在 Terraform 中，輸入下列命令來建立此 IAM 角色：
	`terraform init`
	`terraform plan`
	`terraform apply`
		AWS DevOps，雲端架構師

任務	Description	所需的技能
	4. 驗證角色已成功部署，並具有預期的跨帳戶存取。	
為您的環境自訂解決方案。	1. 輸入下列命令，將AWS 帳戶 適用於使用 Account Factory for Terraform (AFT) 儲存庫的治理 SSO 指派複製到本機工作站。 `git clone https://github.com/aws-samples/aft-custom-sso-assignment.git` 2. 在 `aft-account-provisioning-customizations/terraform` 資料夾中，開啟 `variables.tf` 檔案。 3. 視需要為您的環境修改變數。 4. 儲存並關閉 `variables.tf` 檔案。 5. 在儲存`aft-account-request` 庫中開啟 `account-request.tf` 檔案。 6. 修改`custom_fields` 參數以定義許可集和目標使用者和群組。如需詳細資訊，請參閱此模式的架構區段。 7. 儲存並關閉 `account-request.tf` 檔案。	AWS DevOps，雲端架構師

任務	Description	所需的技能
部署解決方案。	1. 在複製的儲存庫中，複製`terraform` 資料夾的內容，然後將其貼入`aft-account-provisioning-customizations` 儲存庫中的`terraform` 資料夾。 2. 在 AFT 管理帳戶中，啟動`ct-aft-account-provisioning-customizations` 管道。這會部署自訂解決方案。如需說明，請參閱在 CodePipeline 中啟動管道。 3. 驗證資源是否已成功部署在 AFT 管理帳戶中。	AWS DevOps，雲端架構師
設定程式碼儲存庫連線。	在您要存放組態檔案的程式碼儲存庫與 之間設定連線 AWS 帳戶。如需說明，請參閱 AWS CodePipeline 文件中的使用 CodeConnections 將第三方來源提供者新增至管道。	AWS DevOps，雲端架構師

使用 解決方案

任務	Description	所需的技能
啟動 AFT 管道以部署新帳戶。	請遵循使用 AFT 佈建新帳戶中的指示，以啟動 AWS 帳戶 在您的 AWS Control Tower 環境中建立新 的管道。等待帳戶建立程序完成。	AWS DevOps，雲端架構師

任務	Description	所需的技能
驗證變更。	1. 開啟 AWS IAM Identity Center 主控台。 2. 在帳戶清單中，選擇新建立的帳戶。 3. 驗證是否已指派許可集，以將存取權授予目標使用者和群組。	AWS DevOps，雲端架構師

疑難排解

問題	解決方案
許可集指派無法運作。	確定群組 ARN、組織 ID 和 Lambda 參數正確。如需範例，請參閱此模式的解決方案概觀一節。
更新儲存庫中的程式碼不會啟動管道。	此問題與 AWS 帳戶 和 儲存庫之間的連線有關。在 中 AWS 管理主控台，驗證連線是否作用中。如需詳細資訊，請參閱 AWS CodePipeline 文件中的 GitHub 連線。

其他資訊

與票證管理工具整合

您可以選擇將此解決方案與票證或問題管理工具整合，例如 Jira 或 ServiceNow。下圖顯示此選項的範例工作流程。您可以使用工具的連接器，將票證管理工具與 AFT 解決方案儲存庫整合。如需 Jira 連接器，請參閱將 Jira 與 GitHub 整合。如需 ServiceNow 連接器，請參閱與 GitHub 整合。您甚至可以建置自訂解決方案，要求使用者在提取請求核准過程中提供票證 ID。如果 AWS 帳戶 使用 AFT 建立新的請求獲得核准，該事件可能會啟動工作流程，將自訂欄位新增至 `aft-account-request` GitHub 儲存庫。您可以設計任何符合您使用案例需求的自訂工作流程。

該圖顯示以下工作流程：

1. 使用者在票證管理工具中請求自訂許可集指派，例如 Jira。

2. 核准案例後，工作流程會開始更新許可集指派。（選用）您可以使用外掛程式來自訂此步驟的自動化。
3. 運算子會將 Terraform 程式碼與更新的許可集參數傳送至開發或功能分支中的`aft-account-request`儲存庫。
4. GitHub 動作 AWS CodeBuild 會使用 OpenID Connect (OIDC) 呼叫啟動。CodeBuild 使用 tfsec 和 checkov 等工具，以程式碼 (IaC) 安全性掃描的形式執行基礎設施。它會警告運算子任何安全違規。
5. 如果找不到違規，GitHub Actions 會建立自動提取請求，並將程式碼檢閱指派給程式碼擁有者。它也會為提取請求建立標籤。
6. 如果程式碼擁有者核准程式碼檢閱，則會啟動另一個 GitHub 動作工作流程。它會檢查提取請求標準，包括：
   - 如果提取請求標題符合要求。
   - 如果提取請求內文包含核准的案例編號。
   - 如果提取請求已正確標記。
7. 如果提取請求符合標準，GitHub 動作會啟動 AFT 產品工作流程。它使用 啟動`ct-aft-account-request`管道 AWS CodePipeline。此管道會在 Step Functions 中啟動`aft-account-provisioning-framework`自訂狀態機器。此狀態機器的運作方式如先前此模式的解決方案概觀一節所述。

使用引導管道實作 Account Factory for Terraform (AFT)

Vinicius Elias 和 Edgar Costa Filho，Amazon Web Services

總結

此模式提供簡單且安全的方法來從 管理帳戶部署 AWS Control Tower Account Factory for Terraform (AFT) AWS Organizations。解決方案的核心是 CloudFormation 範本，透過建立 Terraform 管道來自動化 AFT 組態，其結構可輕鬆適應初始部署或後續更新。

安全性和資料完整性是 的首要任務 AWS，因此 Terraform 狀態檔案是追蹤受管基礎設施和組態狀態的關鍵元件，可安全地存放在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中。此儲存貯體設定了數種安全措施，包括伺服器端加密和封鎖公開存取的政策，以協助確保您的 Terraform 狀態免於未經授權的存取和資料外洩。

管理帳戶會協調和監督整個環境，因此它是其中的關鍵資源 AWS Control Tower。此模式遵循 AWS 最佳實務，並確保部署程序不僅有效率，也符合安全與控管標準，提供全方位、安全且有效率的方式來在您的 AWS 環境中部署 AFT。

如需 AFT 的詳細資訊，請參閱 AWS Control Tower 文件。

先決條件和限制

先決條件

- 至少具有下列帳戶的基本 AWS 多帳戶環境：管理帳戶、日誌封存帳戶、稽核帳戶，以及一個用於 AFT 管理的額外帳戶。
- 已建立 AWS Control Tower 的環境。管理帳戶應該正確設定，因為 CloudFormation 範本會部署在其中。
- AWS 管理帳戶中的必要許可。您需要足夠的許可來建立和管理資源，例如 S3 儲存貯體、 AWS Lambda 函數、 AWS Identity and Access Management (IAM) 角色和 AWS CodePipeline 專案。
- 熟悉 Terraform。了解 Terraform 的核心概念和工作流程很重要，因為部署涉及產生和管理 Terraform 組態。

限制

- 請注意您帳戶中AWS 的資源配額。部署可能會建立多個資源，而遇到服務配額可能會阻礙部署程序。

- 範本專為特定版本的 Terraform 和 而設計 AWS 服務。升級或變更版本可能需要修改範本。
- 範本不支援自我管理版本控制系統 (VCS) 服務，例如 GitHub Enterprise。

產品版本

- Terraform 1.6.6 版或更新版本
- AFT 1.11 版或更新版本

Architecture

目標技術堆疊

- CloudFormation
- AWS CodeBuild
- AWS CodeCommit
- AWS CodePipeline
- Amazon EventBridge
- IAM
- AWS Lambda
- Amazon S3

目標架構

下圖說明此模式中討論的實作。

工作流程包含三個主要任務：建立資源、產生內容和執行管道。

建立 資源

隨此模式提供的 CloudFormation 範本會建立和設定所有必要的資源，取決於您在部署範本時選取的參數。範本至少會建立下列資源：

- 實作 AFT 的 CodePipeline 管道
- 存放與 AFT 實作相關聯之 Terraform 狀態檔案的 S3 儲存貯體

- 兩個 CodeBuild 專案實作 Terraform 計劃，並在管道的不同階段套用命令
- CodeBuild 和 CodePipeline 服務的 IAM 角色
- 儲存管道執行期成品的第二個 S3 儲存貯體

範本會根據您選取的 VCS 提供者 (CodeCommit 或外部 VCS) 建立下列資源。

- 對於 CodeCommit：
  - 用來存放 AFT Terraform 引導程式碼的 CodeCommit 儲存庫
  - 擷取`main`分支上 CodeCommit 儲存庫變更的 EventBridge 規則
  - EventBridge 規則的另一個 IAM 角色
- 對於任何其他外部 VCS 提供者，例如 GitHub：
  - AWS CodeConnections 連線

此外，當您選取 CodeCommit 做為 VCS 供應商時，如果您將 `Generate AFT Files` 參數設定為 `true`，範本會建立這些額外的資源來產生內容：

- 儲存產生內容並用作 CodeCommit 儲存庫來源的 S3 儲存貯體
- 處理指定參數並產生適當內容的 Lambda 函數
- 執行 Lambda 函數的 IAM 函數
- 部署範本時執行 Lambda 函數的 CloudFormation 自訂資源

產生內容

若要產生 AFT 引導檔案及其內容，解決方案會使用 Lambda 函數和 S3 儲存貯體。函數會在 儲存貯體中建立資料夾，然後在 資料夾內建立兩個檔案： `main.tf`和 `backend.tf`。函數也會處理提供的 CloudFormation 參數，並以預先定義的程式碼填入這些檔案，取代個別的參數值。

若要檢視用來產生檔案的範本程式碼，請參閱解決方案的 GitHub 儲存庫。基本上，檔案的產生方式如下。

main.tf

```
module "aft" {
  source = "github.com/aws-ia/terraform-aws-control_tower_account_factory?
ref=<aft_version>"
```

```
  # Required variables
  ct_management_account_id  = "<ct_management_account_id>"
  log_archive_account_id    = "<log_archive_account_id>"
  audit_account_id          = "<audit_account_id>"
  aft_management_account_id = "<aft_management_account_id>"
  ct_home_region            = "<ct_home_region>"

  # Optional variables
  tf_backend_secondary_region = "<tf_backend_secondary_region>"
  aft_metrics_reporting       = "<false|true>"

  # AFT Feature flags
  aft_feature_cloudtrail_data_events      = "<false|true>"
  aft_feature_enterprise_support          = "<false|true>"
  aft_feature_delete_default_vpcs_enabled = "<false|true>"

  # Terraform variables
  terraform_version      = "<terraform_version>"
  terraform_distribution = "<terraform_distribution>"

  # VCS variables (if you have chosen an external VCS)
  vcs_provider                                = "<github|githubenterprise|gitlab|
gitlabselfmanaged|bitbucket>"
  account_request_repo_name                   = "<org-name>/aft-account-request"
  account_customizations_repo_name            = "<org-name>/aft-account-
customizations"
  account_provisioning_customizations_repo_name = "<org-name>/aft-account-provisioning-
customizations"
  global_customizations_repo_name             = "<org-name>/aft-global-
customizations"

}
```

backend.tf

```
terraform {
  backend "s3" {
    region = "<aft-main-region>"
    bucket = "<s3-bucket-name>"
    key    = "aft-setup.tfstate"
  }
}
```

在建立 CodeCommit 儲存庫期間，如果您將 `Generate AFT Files` 參數設定為 `true`，範本會使用 S3 儲存貯體搭配產生的內容做為`main`分支的來源，以自動填入儲存庫。

執行管道

建立資源並設定引導檔案之後，管道就會執行。第一階段 (來源) 從儲存庫的主分支擷取原始程式碼，第二階段 (建置) 會執行 Terraform 計劃命令，並產生要檢閱的結果。在第三個階段 (核准) 中，管道會等待手動動作核准或拒絕最後一個階段 (部署)。在最後一個階段，管道會使用先前 Terraform `apply`命令的結果做為輸入，來執行 Terraform `plan`命令。最後，管理帳戶中的跨帳戶角色和許可將用於在 AFT 管理帳戶中建立 AFT 資源。

> **Note**
>
> 如果您選擇外部 VCS 提供者，則需要授權與您的 VCS 提供者憑證的連線。若要完成設定，請遵循 AWS 開發人員工具主控台文件中更新待定連線的步驟。

工具

AWS 服務

- CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和 區域的生命週期中管理這些資源。
- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- AWS CodeConnections 可讓 CodePipeline 等 AWS 資源和服務連線至外部程式碼儲存庫，例如 GitHub。
- AWS Lambda 是一種運算服務，可執行程式碼以回應事件並自動管理運算資源，提供快速的方式來建立現代化、無伺服器的應用程式以供生產。
- 適用於 Python (Boto3) 的 AWS SDK 是一種軟體開發套件，可協助您整合 Python 應用程式、程式庫或指令碼 AWS 服務。

其他工具

- Terraform 是一種基礎設施即程式碼 (IaC) 工具，可讓您安全且有效率地建置、變更和版本基礎設施。這包括低階元件，例如運算執行個體、儲存和聯網；以及高階元件，例如 DNS 項目和 SaaS 功能。
- Python 是一種易於學習、功能強大的程式設計語言。它具有高效的高階資料結構，並提供簡單但有效的物件導向程式設計方法。

程式碼儲存庫

此模式的程式碼可在 GitHub AFT 引導管道儲存庫中使用。

如需官方 AFT 儲存庫，請參閱 GitHub 中的 AWS Control Tower Account Factory for Terraform。

最佳實務

當您使用提供的 CloudFormation 範本部署 AFT 時，建議您遵循最佳實務，以協助確保安全、有效率且成功的實作。實作和操作 AFT 的重要準則和建議包括下列項目。

- 徹底檢閱參數：仔細檢閱並了解 CloudFormation 範本中的每個參數。準確的參數組態對於 AFT 的正確設定和運作至關重要。
- 定期範本更新：使用最新的 AWS 功能和 Terraform 版本來更新範本。定期更新可協助您利用新功能並維護安全性。
- 版本控制：固定您的 AFT 模組版本，並盡可能使用個別的 AFT 部署進行測試。
- 範圍：僅使用 AFT 來部署基礎設施護欄和自訂。請勿使用它來部署您的應用程式。
- 內嵌和驗證：AFT 管道需要內嵌和驗證的 Terraform 組態。在將組態推送至 AFT 儲存庫之前，執行 lint、驗證和測試。
- Terraform 模組：建置可重複使用的 Terraform 程式碼做為模組，並一律指定 Terraform 和 AWS 提供者版本以符合組織的需求。

史詩

設定環境 AWS

任務	Description	所需的技能
準備 AWS Control Tower 環境。	AWS Control Tower 在您的 AWS 環境中設定和配置，以確	雲端管理員

任務	Description	所需的技能
	保集中管理和控管您的 AWS 帳戶。如需詳細資訊，請參閱 AWS Control Tower 文件中的入門 AWS Control Tower。	
啟動 AFT 管理帳戶。	使用 AWS Control Tower Account Factory 啟動新的 AWS 帳戶 做為您的 AFT 管理帳戶。如需詳細資訊，請參閱 AWS Control Tower 文件中的使用 Account Factory 佈建 AWS Service Catalog 帳戶。	雲端管理員

在管理帳戶中部署 CloudFormation 範本

任務	Description	所需的技能
啟動 CloudFormation 範本。	在此史詩中，您將部署此解決方案隨附的 CloudFormation 範本，以在 AWS 管理帳戶中設定 AFT 引導管道。管道會將 AFT 解決方案部署到您在上一個史詩中所設定的 AFT 管理帳戶中。 步驟 1：開啟 CloudFormation 主控台 • 登入 AWS 管理主控台 並開啟 CloudFormation 主控台。請確定您在正確的 AWS Control Tower 主要區域內操作。 步驟 2：建立新的堆疊	雲端管理員

任務	Description	所需的技能
	1. 選擇 以建立新的堆疊。 2. 選取 選項以上傳範本檔案，並上傳此模式隨附的 CloudFormation 範本。 步驟 3：設定堆疊參數 • `VCS Provider`：選取要使用的版本控制系統 (VCS) 提供者。您可以選取 GitHub 等外部 VCS，或在您的帳戶獲允許使用服務時使用 CodeCommit。 • `Repository Name` ：指定儲存庫名稱以存放 AFT 引導模組。對於外部 VCS 提供者，請使用完整路徑，包括組織名稱（例如 `my-github-org/my-repo`)。 • `Branch Name`：指定來源儲存庫分支。 • `CodeBuild Docker Image`：選擇要用作 CodeBuild Docker 基礎映像的檔案。 • 如果您將 VCS 提供者設定為 CodeCommit 以外的選項，請前往步驟 8。 步驟 4：決定檔案產生 • 如果您選取 CodeCommit 做為 VCS 提供者，您可以使用	

任務	Description	所需的技能
	Generate AFT Files 參數來控制預設 AFT 部署檔案的產生。將此參數設定為： • true 在指定的儲存庫中自動建立和存放 AFT 部署檔案。 • false 如果您想要手動處理檔案建立或已備妥檔案。 • 如果您選取 false，請前往步驟 8；否則，請先遵循步驟 5-7。 步驟 5：填寫 AWS Control Tower 和 AFT 帳戶詳細資訊 • 如果您將 Generate AFT Files 參數設定為 true，請提供下列 AWS Control Tower 和 AFT 帳戶特定資訊。 • Log Archive Account ID： 中日誌封存帳戶 ID 的 ID AWS Control Tower。 • Audit Account ID：中稽核帳戶的 ID AWS Control Tower。 • AFT Management Account ID：您在第一個史詩中建立的 AFT 管理帳戶的 ID。	

任務	Description	所需的技能
	• `AFT Main Region` 和 `AFT Secondary Region`：AWS 區域 AFT 部署的主要和次要。 步驟 6：設定 AFT 選項 • 設定指標報告： • `AFT Enable Metrics Reporting` ：啟用或停用 AFT 指標報告。如需詳細資訊，請參閱 AWS Control Tower 文件中的操作指標。 • 設定 AFT 功能選項： • `Enable AFT CloudTrail Data Events`：在所有 AFT 受管帳戶中啟用 CloudTrail 資料事件。如需詳細資訊，請參閱 AWS Control Tower 文件中的AWS CloudTrail 資料事件。 • `Enable AFT Enterprise Support`：在所有 AFT 受管帳戶中啟用企業支援。如需詳細資訊，請參閱 AWS Control Tower 文件中的AWS 企業支援計劃。 • `Enable AFT Delete Default VPC`：僅刪除	

任務	Description	所需的技能
	AFT 管理帳戶中的所有 VPCs。如需詳細資訊，請參閱 AWS Control Tower 文件中的刪除 AWS 預設 VPC。 步驟 7：指定版本 • AFT Terraform Version：選擇要在 AFT 管道中使用的 Terraform 版本。 • AFT Version：定義要部署的 AFT 版本。保留預設設定 (latest) 以使用最新的 AFT 版本。 步驟 8：檢閱並建立堆疊 • 檢閱所有參數和設定。如果一切正常，請繼續建立堆疊。 步驟 9：監控堆疊建立 • CloudFormation 會佈建和設定您定義的資源。在 CloudFormation 主控台上監控堆疊建立程序。此程序可能需要幾分鐘的時間。 步驟 10：驗證部署 • 當堆疊狀態顯示 CREATE_COMPLETE 時，	

任務	Description	所需的技能
	請確認所有資源都已正確建立。 • 在輸出區段中，記下 `TerraformBackendBucketName` 值。	

填入並驗證 AFT 引導儲存庫和管道

任務	Description	所需的技能
選項 1：填入外部 VCS 的 AFT 引導儲存庫。	如果您將 VCS 提供者設定為外部 VCS（而不是 CodeCommit），請遵循下列步驟。 （選用）部署 CloudFormation 範本之後，您可以在新建立的 AFT 引導儲存庫中填入或驗證內容，並測試管道是否已成功執行。 步驟 1：更新連線 1. 在 CodePipeline 主控台的導覽窗格中，選擇設定、連線。 2. 選取 `aft-vcs-connection` 連線。它應該處於 `Pending` 狀態。 3. 選擇更新待定連線，並遵循開發人員工具主控台文件中更新待定連線的指示。 4. 當連線處於 `Available` 狀態時，請移至下一個步驟。	雲端管理員

任務	Description	所需的技能
	步驟 2：填入儲存庫 1. 使用您的外部 VCS 登入資料，將您在範本中指定的儲存庫複製到本機電腦。如果您保留預設名稱，儲存庫稱為 aft-setup 。 2. 在儲存庫中，建立名為 terraform 的資料夾，其中包含兩個空白檔案： backend.tf 和 main.tf。 3. 開啟 backend.tf 檔案並新增此程式碼片段： `terraform { backend "s3" { region = "<aft-main-region>" bucket = "<s3-bucket-name>" key = "aft-setup" } }` 在 檔案中： • <aft-main-region> 將 取代為主要 AFT 區域。這應與 AWS Control Tower 主要區域相符。 • <s3-bucket-name> 將 取代為 Terraform 後端儲存貯體的名稱。您可以在先前部署的 CloudFormation 範本	

任務	Description	所需的技能
	所產生的`Terraform BackendBucketName` 輸出中找到此項目。 4. 開啟 `main.tf` 檔案，並使用 AFT 儲存庫中可用的其中一個範例來部署 AFT。例如，您可以使用您偏好的 VCS 提供者 (CodeCommit、GitHub 或 Bitbucket) 或自訂 AFT VPC。如需更多 AFT 輸入選項，請參閱 AFT 儲存庫中的 README 檔案。 步驟 2：遞交並推送變更 • 在您建立並填入資料夾和檔案之後，請確認您的變更，然後將程式碼上傳至儲存庫。管道會自動啟動，透過來源和建置階段執行，然後在部署階段之前等待核准動作。	

任務	Description	所需的技能

選項 2：填入 CodeCommit 的 AFT 引導儲存庫。	如果您將 VCS 提供者設定為 CodeCommit，請遵循下列步驟。
	
	（選用）部署 CloudFormation 範本之後，您可以在新建立的 AFT 引導儲存庫中填入或驗證內容，並測試管道是否已成功執行。
	
	如果您將 `Generate AFT Files` 參數設定為 `true`，請跳到下一個案例（驗證管道）。
	
	步驟 1：填入儲存庫
	
	1. 開啟 AWS CodeCommit 主控台，然後選取新建立的儲存庫。如果您保留預設名稱，儲存庫名稱應為 `aft-setup` 。
	2. 使用 SSH、HTTPS 或 HTTPS (GRC) 將儲存庫複製到本機機器，然後在編輯器中開啟儲存庫。
	3. 建立名為 的資料夾`terraform` 和其中兩個空白檔案： `backend.tf` 和 `main.tf`。
	4. 開啟 `backend.tf` 檔案並新增此程式碼片段：
	
	`terraform {`
	`  backend "s3" {`
		雲端管理員

任務	Description	所需的技能

	<pre>region = "<aft-main-region>"
	bucket = "<s3-bucket-name>"
	key = "aft-setup"
	}
	}</pre>
	在 檔案中：
	• <aft-main-region> 將 取代為主要 AFT 區域。這應與 AWS Control Tower 主要區域相符。
	• <s3-bucket-name> 將 取代為 Terraform 後端儲存貯體的名稱。您可以在先前部署的 CloudFormation 範本產生的Terraform BackendBucketName 輸出中找到此項目。
	5. 開啟 main.tf 檔案，並使用 AFT 儲存庫中可用的其中一個範例來部署 AFT。例如，您可以使用您偏好的版本控制系統 (VCS) 供應商 (CodeCommit、GitHub 或 Bitbucket) 或自訂 AFT VPC。如需更多 AFT 輸入選項，請參閱 AFT 儲存庫中的 README 檔案。
	
	步驟 2：遞交並推送您的變更	

任務	Description	所需的技能
	• 在您建立並填入資料夾和檔案之後，請確認您的變更，然後將程式碼上傳至儲存庫。管道會自動啟動，透過來源和建置階段執行，然後在部署階段之前等待核准動作。	

任務	Description	所需的技能
驗證 AFT 引導管道。	步驟 1：檢視管道 • 開啟 CodePipeline 主控台，並檢查`aft-bootstrap-pipeline` 管道是否已成功啟動。它應該執行 Terraform 計劃或等待手動核准動作。 步驟 2：核准 Terraform 計劃結果 • 您可以查看建置階段的執行日誌來檢閱 Terraform 計畫的結果，然後在核准階段核准或拒絕執行。如果您核准，管道會開始在提供的 AFT 管理帳戶中部署 AFT 資源。 步驟 3：等待部署 • 等待管道成功執行。這大約需要 30 分鐘。您可能遇到的任何失敗通常是由 API 配額造成。在這些情況下，您可以重新執行管道以繼續部署。 步驟 4：檢查已建立的資源 • 存取 AFT 管理帳戶並確認資源已建立。	雲端管理員

疑難排解

問題	解決方案
CloudFormation 範本中包含的自訂 Lambda 函數在部署期間失敗。	檢查 Lambda 函數的 Amazon CloudWatch logs 以識別錯誤。日誌提供詳細資訊，可協助找出特定問題。確認 Lambda 函數具有必要的許可，並且已正確設定環境變數。
由於許可不足，您在資源建立或管理時遇到失敗。	檢閱連接到 Lambda 函數、CodeBuild 和涉及部署之其他服務的 IAM 角色和政策。確認他們具有必要的許可。如果有許可問題，請調整 IAM 政策以授予必要的存取權。
您正在搭配較新 AWS 服務 或 Terraform 版本使用過時的 CloudFormation 範本版本。	定期更新 CloudFormation 範本，使其與最新 AWS 和 Terraform 版本相容。檢查版本備註或文件是否有任何版本特定的變更或要求。
您在部署期間達到 AWS 服務 配額。	部署管道之前，請檢查 S3 AWS 服務 儲存貯體、IAM 角色和 Lambda 函數等資源的配額。如有必要，請求會增加。如需詳細資訊，請參閱 AWS 網站上的AWS 服務 配額。
您因為 CloudFormation 範本中的輸入參數不正確而發生錯誤。	仔細檢查所有輸入參數是否有錯字或不正確的值。確認資源識別符，例如帳戶 IDs和區域名稱是準確的。

相關資源

若要成功實作此模式，請檢閱下列資源。這些資源提供其他資訊和指引，在設定和管理 AFT 時，這些資訊和指引非常寶貴 CloudFormation。

AWS 文件：

- AWS Control Tower 使用者指南提供有關設定和管理的詳細資訊 AWS Control Tower。
- CloudFormation 文件提供 CloudFormation 範本、堆疊和資源管理的洞見。

IAM 政策和最佳實務：

- IAM 中的安全最佳實務說明如何使用 IAM 角色和政策來協助保護 AWS 資源。

上的 Terraform AWS：

- Terraform AWS 提供者文件提供有關搭配 Terraform 使用的完整資訊 AWS。

AWS 服務 配額：

- AWS 服務 配額提供如何檢視 AWS 服務 配額以及如何請求增加的資訊。

在多個 AWS 帳戶和 AWS 區域中管理 AWS Service Catalog 產品

Ram Kandaswamy，Amazon Web Services

總結

Amazon Web Services (AWS) Service Catalog 簡化並加速企業基礎設施即程式碼 (IaC) 範本的控管和分發。您可以使用 AWS CloudFormation 範本來定義產品所需的 AWS 資源集合（堆疊)。AWS CloudFormation StackSets 可讓您透過單一操作，跨多個帳戶和 AWS 區域建立、更新或刪除堆疊，藉此擴充此功能。

AWS Service Catalog 管理員使用開發人員撰寫的 CloudFormation 範本來建立產品，並發佈它們。這些產品接著會與產品組合建立關聯，並套用限制來控管。若要將您的產品提供給其他 AWS 帳戶或組織單位 (OUs) 中的使用者，您通常會與他們共用您的產品組合。此模式說明管理以 AWS CloudFormation StackSets 為基礎的 AWS Service Catalog 產品方案的替代方法。 AWS CloudFormation StackSets 您可以使用堆疊集限制來設定可部署和使用產品的 AWS 區域和帳戶，而不是共用產品組合。透過使用此方法，您可以在多個帳戶、OUs 和 AWS 區域中佈建 AWS Service Catalog 產品，並從中央位置管理它們，同時滿足您的控管需求。

此方法的優點：

- 產品是從主要帳戶佈建和管理，不會與其他帳戶共用。
- 此方法提供以特定產品為基礎的所有佈建產品（堆疊）的合併檢視。
- 使用 AWS Service Management Connector 的組態比較容易，因為它只以一個帳戶為目標。
- 查詢和使用來自 AWS Service Catalog 的產品更容易。

先決條件和限制

先決條件

- IaC 和版本控制的 AWS CloudFormation 範本
- 用於佈建和管理 AWS 資源的多帳戶設定和 AWS Service Catalog

限制

- 此方法使用 AWS CloudFormation StackSets，且適用 StackSets 的限制：
  - StackSets 不支援透過巨集部署 CloudFormation 範本。如果您使用巨集來預先處理範本，您將無法使用 StackSets 型部署。

  - StackSets 可讓您取消堆疊與堆疊集的關聯，因此您可以鎖定特定堆疊來修正問題。不過，取消關聯的堆疊無法與堆疊集重新建立關聯。
- AWS Service Catalog 會自動產生 StackSet 名稱。目前不支援自訂。

Architecture

目標架構

1. 使用者會建立 AWS CloudFormation 範本，以 JSON 或 YAML 格式佈建 AWS 資源。
2. CloudFormation 範本會在 AWS Service Catalog 中建立產品，並將其新增至產品組合。
3. 使用者會建立佈建產品，以在目標帳戶中建立 CloudFormation 堆疊。
4. 每個堆疊會佈建 CloudFormation 範本中指定的資源。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶和區域的生命週期中管理這些資源。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Service Catalog 可協助您集中管理針對 AWS 核准的 IT 服務目錄。最終使用者可在機構所設的限制範圍內，迅速地只部署自己需要且經核准的 IT 服務。

史詩

跨帳戶佈建產品

任務	Description	所需的技能
建立組合。	產品組合是一種容器，其中包含一或多個根據特定條件分組	AWS Service Catalog、IAM

任務	Description	所需的技能
	在一起的產品。將產品組合用於您的產品，可協助您在產品集中套用常見的限制條件。 若要建立產品組合，請遵循 AWS Service Catalog 文件中的指示。如果您使用的是 AWS CLI，以下是範例命令： `aws servicecatalog create-portfolio --provider-name my-provider --display-name my-portfolio` 如需詳細資訊，請參閱 AWS CLI 文件。	
建立 CloudFormation 範本。	建立描述 資源的 CloudFormation 範本。適用時，應參數化資源屬性值。	AWS CloudFormation、JSON/YAML

任務	Description	所需的技能
建立具有版本資訊的產品。	當您在 AWS Service Catalog 中發佈 CloudFormation 範本時，範本會變成產品。提供選用版本詳細資訊參數的值，例如版本標題和描述；這有助於稍後查詢產品。 若要建立產品，請遵循 AWS Service Catalog 文件中的指示。如果您使用的是 AWS CLI，範例命令為： `aws servicecatalog create-product --cli-input-json file://create-product-input.json` 其中 `create-product-input.json` 是傳遞產品參數的檔案。如需此檔案的範例，請參閱其他資訊一節。如需詳細資訊，請參閱 AWS CLI 文件。	AWS Service Catalog
套用限制條件。	將堆疊集限制條件套用至產品組合，以設定產品部署選項，例如多個 AWS 帳戶、區域和許可。如需說明，請參閱 AWS Service Catalog 文件。	AWS Service Catalog

任務	Description	所需的技能
新增 許可。	提供許可給使用者，讓他們可以啟動產品組合中的產品。如需主控台說明，請參閱 AWS Service Catalog 文件。如果您使用的是 AWS CLI，以下是範例命令： `aws servicecatalog associate-principal-with-portfolio \ --portfolio-id port-2s6abcdefwdh4 \ --principal-arn arn:aws:iam::444455556666:role/Admin \ --principal-type IAM` 如需詳細資訊，請參閱 AWS CLI 文件。	AWS Service Catalog、IAM

任務	Description	所需的技能
佈建 產品。	佈建的產品是產品的資源執行個體。根據 CloudFormation 範本佈建產品會啟動 CloudFormation 堆疊及其基礎資源。 根據堆疊集限制，以適用的 AWS 區域和帳戶為目標來佈建產品。在 AWS CLI 中，以下是範例命令： `aws servicecatalog provision-product \ --product-id prod-abcdfz3syn2rg \ --provisioning-artifact-id pa-abc347pcsccfm \ --provisioned-product-name "mytestppname3"` 如需詳細資訊，請參閱 AWS CLI 文件。	AWS Service Catalog

相關資源

參考

- AWS Service Catalog 概觀
- 使用 AWS CloudFormation StackSets

教學課程和影片

- AWS re：Invent 2019：自動化一切：選項和最佳實務（影片）

其他資訊

當您使用 `create-product`命令時，`cli-input-json` 參數會指向指定資訊的檔案，例如產品擁有者、支援電子郵件和 CloudFormation 範本詳細資訊。以下是此類檔案的範例：

```
{
   "Owner": "Test admin",
      "SupportDescription": "Testing",
         "Name": "SNS",
            "SupportEmail": "example@example.com",
            "ProductType": "CLOUD_FORMATION_TEMPLATE",
               "AcceptLanguage": "en",
                  "ProvisioningArtifactParameters": {
                     "Description": "SNS product",
                        "DisableTemplateValidation": true,
                           "Info": {
                              "LoadTemplateFromURL": "<url>"
                     },
                           "Name": "version 1"
}
```

使用 AWS 服務監控 SAP RHEL Pacemaker 叢集

Harsh Thoria、Randy Germann 和 RAVEENDRA Voore，Amazon Web Services

總結

此模式概述使用 Amazon CloudWatch 和 Amazon Simple Notification Service (Amazon SNS) 監控和設定適用於 SAP 應用程式和 SAP HANA 資料庫服務的 Red Hat Enterprise Linux (RHEL) Pacemaker 叢集警示的步驟。

組態可讓您在 CloudWatch 日誌串流、指標篩選條件和警示的協助下，監控 SAP SCS 或 ASCS、Enqueue Replication Server (ERS) 和 SAP HANA 叢集資源處於「停止」狀態。Amazon SNS 會傳送電子郵件給基礎設施或 SAP Basis 團隊，告知已停止的叢集狀態。

您可以使用 AWS CloudFormation 指令碼 AWS 或服務主控台來建立此模式 AWS 的資源。此模式假設您使用主控台；它不提供 CloudFormation 指令碼或涵蓋 CloudWatch 和 Amazon SNS 的基礎設施部署。Pacemaker 命令用於設定叢集警示組態。

先決條件和限制

先決條件

- 作用中 AWS 的帳戶。
- Amazon SNS 會設定 來傳送電子郵件或行動通知。
- 適用於 ABAP 的 SAP ASCS/ERS 或適用於 Java 的 SCS/ERS，以及 SAP HANA 資料庫 RHEL Pacemaker 叢集。如需詳細說明，請參閱下列主題：
  - SAP HANA 叢集設定
  - SAP Netweaver ABAP/Java 叢集設定

限制

- 此解決方案目前適用於 RHEL 7.3 版和更新版本的 Pacemaker 型叢集。它尚未在 SUSE 作業系統上進行測試。

產品版本

- RHEL 7.3 及更新版本

Architecture

目標技術堆疊

- RHEL Pacemaker 警示事件驅動型代理程式
- Amazon Elastic Compute Cloud (Amazon EC2)
- CloudWatch 警示
- CloudWatch 日誌群組和指標篩選條件
- Amazon SNS

目標架構

下圖說明此解決方案的元件和工作流程。

自動化和擴展

- 您可以使用 CloudFormation 指令碼自動建立 AWS 資源。您也可以使用其他指標篩選條件來擴展和涵蓋多個叢集。

工具

AWS 服務

- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您執行的應用程式。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。

工具

- CloudWatch 代理程式（統一）是一種工具，可從 EC2 執行個體收集系統層級指標、日誌和追蹤，並從應用程式擷取自訂指標。
- Pacemaker 警示代理程式（適用於 RHEL 7.3 和更新版本）是一種工具，可在發生變更時啟動動作，例如當資源在 Pacemaker 叢集中停止或重新啟動時。

最佳實務

- 如需在 上使用 SAP 工作負載的最佳實務 AWS，請參閱 AWS Well-Architected Framework 的 SAP Lens。
- 考慮為 SAP HANA 叢集設定 CloudWatch 監控所涉及的成本。如需詳細資訊，請參閱 CloudWatch 文件。
- 請考慮使用分頁器或票證機制處理 Amazon SNS 提醒。
- 一律檢查 RPM 套件的 RHEL 高可用性 (HA) 版本是否有 pc、Pacemaker AWS 和 fencing 代理程式。

史詩

設定 Amazon SNS

任務	Description	所需的技能
建立 SNS 主題。	1. 登入 AWS 管理主控台，並在 https://console.aws.amazon.com/sns/v3/home：// 開啟 Amazon SNS 主控台。 2. 在 Amazon SNS 儀表板上，在 Common actions (常見的動作) 下，選擇 Create Topic (建立主題)。 3. 在建立新主題對話方塊中，針對類型，選擇標準。 4. 針對主題名稱，輸入主題的名稱（例如，`my-topic`)。 5. 請選擇建立主題。 這會使用可讓您發佈通知的資源政策來建立 SNS 主題。	AWS 管理員

任務	Description	所需的技能
	6. 複製主題 ARN（例如，`arn:aws:sns:us-east-1:111122223333:my-topic`）。您將在後續步驟中使用此 ARN。	

任務	Description	所需的技能
修改 SNS 主題的存取政策。	1. 在 Amazon SNS 主控台的導覽窗格中，選擇主題，然後選擇您建立的主題。 2. 選擇編輯並前往存取政策區段。 3. 請確定存取政策包含 CloudWatch 做為允許發佈至此主題的其中一個服務主體。例如： `{ "Sid": "Allow AWS CloudWatch to Publish to this SNS topic", "Effect": "Allow", "Principal": { "Service": [ "cloudwatch.amazonaws.com" ] }, "Action": "SNS:Publish", "Resource": "arn:aws:sns:us-east-1:111122223333:my-topic" }` 4. 選擇儲存變更。	AWS 系統管理員

任務	Description	所需的技能
訂閱 SNS 主題。	1. 在 Amazon SNS 主控台的導覽窗格中，選擇訂閱、建立訂閱。 2. 針對主題 ARN，貼上您在第一個任務中建立的 ARN。 3. 對於通訊協定，選擇電子郵件。 4. 針對端點，輸入負責 SAP Pacemaker 叢集且應接收通知的人員或團隊的電子郵件地址。例如，這可以是 SAP Basis 或基礎設施團隊分發清單的電子郵件地址。 5. 選擇建立訂閱。 6. 從您的電子郵件應用程式開啟來自 AWS 通知的訊息，並確認訂閱。 您的 Web 瀏覽器顯示自 Amazon SNS 的確認回覆。	AWS 系統管理員

確認叢集的設定

任務	Description	所需的技能
檢查叢集狀態。	使用 pcs 狀態命令來確認資源已上線。	SAP Basis 管理員

設定 Pacemaker 警示

任務	Description	所需的技能

在主要叢集執行個體上設定 Pacemaker 警示代理程式。	登入 pimary 叢集中的 EC2 執行個體，並執行下列命令：
	`install --mode=0755 /usr/share/pacemaker/alerts/alert_file.sh.sample`
	`touch /var/lib/pacemaker/alert_file.sh`
	`touch /var/log/pcmk_alert_file.log`
	`chown hacluster:haclient /var/log/pcmk_alert_file.log`
	`chmod 600 /var/log/pcmk_alert_file.log`
	`pcs alert create id=alert_file description="Log events to a file." path=/var/lib/pacemaker/alert_file.sh`
	`pcs alert recipient add alert_file id=my-alert_logfile value=/var/log/pcmk_alert_file.log`
		SAP Basis 管理員

在次要叢集執行個體上設定 Pacemaker 警示代理程式。	登入次要叢集中的次要叢集 EC2 執行個體，並執行下列命令： `install --mode=0755 /usr/share/pacemaker/alerts/alert_file.sh.sample`	SAP Basis 管理員

任務	Description	所需的技能

	`touch /var/lib/pacemaker/alert_file.sh`
	`touch /var/log/pcmk_alert_file.log`
	`chown hacluster:haclient /var/log/pcmk_alert_file.log`
	`chmod 600 /var/log/pcmk_alert_file.log`	

確認已建立 RHEL 提醒資源。	使用下列命令來確認警示資源已建立：
	`pcs alert`
	命令的輸出如下所示：
	`[root@xxxxxxx ~]# pcs alert`
	`Alerts:`
	` Alert: alert_file (path=/var/lib/pacemaker/alert_file.sh)`
	`  Description: Log events to a file.`
	`  Recipients:`
	`   Recipient: my-alert_logfile (value=/var/log/pcmk_alert_file.log)`
		SAP Basis 管理員

設定 CloudWatch 代理程式

任務	Description	所需的技能

安裝 CloudWatch 代理程式。	有數種方法可在 EC2 執行個體上安裝 CloudWatch 代理程式。若要使用命令列：
	1. 下載 CloudWatch 代理程式套件：
	`wget https://s3.<region>.amazonaws.com/amazoncloudwatch-agent-region/redhat/amd64/latest/amazon-cloudwatch-agent.rpm`
	其中 `<region>` 是 EC2 AWS 區域 執行個體所在的（例如 `us-west-2` ）。
	2. 選用）驗證套件簽章。如需說明，請參閱 CloudWatch 文件中的驗證 CloudWatch 代理程式套件的簽章。CloudWatch
	3. 在第一個執行個體上安裝 套件：
	`sudo rpm -U ./amazon-cloudwatch-agent.rpm`
	4. 對次要執行個體重複此步驟。
		AWS 系統管理員

任務	Description	所需的技能
	如需詳細資訊，請參閱 CloudWatch 文件。	
將 IAM 角色連接至 EC2 執行個體。	若要讓 CloudWatch 代理程式從執行個體傳送資料，您必須將 IAM CloudWatchAgentServerRole 角色連接至每個執行個體。或者，您可以將 CloudWatch 代理程式的政策新增至現有的 IAM 角色。如需詳細資訊，請參閱 CloudWatch 文件。	AWS 管理員

任務	Description	所需的技能
設定 CloudWatch 代理程式以監控主要叢集執行個體上的 Pacemaker 警示代理程式日誌檔案。	1. 執行 命令來設定主要叢集執行個體： `sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-config-wizard` 2. 針對 Linux 選擇 1，然後選取監控策略的選項。 3. 對於「是否要監控任何日誌檔案」問題，請選擇是，然後從 pcs 提醒命令提供 Pacemaker 日誌檔案的路徑。在我們的案例中，它是 `var/log/pcmk_alert_file.log` 。 4. 提供日誌群組和日誌串流的名稱。如果您未指定日誌串流，則會使用 AWS 執行個體 ID 做為預設值。 5. 針對次要叢集執行個體重複步驟 1-4。	AWS 管理員
在主要和次要叢集執行個體上啟動 CloudWatch 代理程式。	若要啟動代理程式，請在主要和次要叢集的 EC2 執行個體上執行下列命令： `sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetch-config -m ec2 -s -c file:/opt/aws/amazon-cloudwatch-agent/bin/config.json`	AWS 管理員

設定 CloudWatch 資源

任務	Description	所需的技能
設定 CloudWatch 日誌群組。	1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。 2. 在導覽窗格中，選擇日誌群組、建立日誌群組。 3. 輸入日誌群組的名稱，然後選擇建立日誌群組。 CloudWatch 代理程式會將 Pacemaker 警示檔案以日誌串流的形式傳輸至 CloudWatch 日誌群組。	AWS 管理員
設定 CloudWatch 指標篩選條件。	指標篩選條件可協助您搜尋模式，例如 CloudWatch 日誌串流`stop <cluster-resource-name>` 中的 。識別此模式時，指標篩選條件會更新自訂指標。 1. 在 CloudWatch 主控台的導覽窗格中，選擇日誌群組。 2. 選擇您在上一個任務中建立的日誌群組名稱。 3. 選擇 Actions (動作) > Create metric filter (建立指標篩選條件)。 4. 針對篩選條件模式，輸入要使用的篩選條件模式，例如`stop ABC_scs`，以符合名	AWS 管理員、SAP Basis 管理員

任務	Description	所需的技能
	為 之 SAP SCS 叢集資源的停止事件ABC_scs。 如需詳細資訊，請參閱 CloudWatch 文件中的篩選模式語法。 5. (選用) 若要測試篩選條件模式，請在 Test Pattern (測試模式) 下方，輸入一個或多個日誌事件，用以測試模式。每個日誌事件都必須在個別的行上指定，因為明細符號用於在日誌事件訊息方塊中分隔日誌事件。 6. 選擇 Next (下一步)，然後輸入篩選條件的名稱。 7. 在指標詳細資訊下，針對指標命名空間，輸入要發佈指標的 CloudWatch 命名空間名稱 （例如 sapcluster_monitoring ）。如果此命名空間尚不存在，請選取建立新。 8. 針對指標名稱，輸入新指標的名稱 （例如 sapcluster_<sid> ，其中 <sid>是 SAP 系統識別名稱）。 9. 針對指標值，輸入 1。 或者，您可以輸入字符，例如 $size。如此會針對包含 size 欄位的每個日誌事件，以 size 欄位中的數值遞增指標。	

任務	Description	所需的技能
	10針對預設值，輸入 0。 11選擇 Create metric filter (建立指標篩選條件)。 當指標篩選條件識別步驟 4 中的模式時，它會將 CloudWatch 自訂指標的值更新`sapcluster_abc` 為 1。 CloudWatch 警示會`SAP-Cluster-QA1-ABC` 監控指標，`sapcluster_abc` 並在指標值變更為 1 時傳送 SNS 通知。這表示叢集資源已停止，且需要採取動作。	

任務	Description	所需的技能
設定 SAP ASCS/SCS 和 ERS 指標的 CloudWatch 指標警示。	若要根據單一指標建立警示： 1. 在 CloudWatch 主控台的導覽窗格中，選擇警示、所有警示。 2. 選擇 Create alarm (建立警示)。 3. 選擇 Select Metric (選取指標)。 4. 搜尋在上一個任務中建立`sapcluster_monitoring` 的自訂指標。 5. 選擇也在上一個任務中建立的 SAP SCS 指標名稱 （例如 `sapcluster_<abc>`)。 6. 在圖形化指標索引標籤上，設定下列項目： • 對於 Statistic (統計數字)，選擇 Maximum (最大值)。 • 對於期間，選擇 1 分鐘。 • 針對閾值類型，選擇靜態，並將 的閾值設定為大於或等於 1 `sapcluster_<sid>` 的值。 7. 選擇下一步。 8. 針對通知，選取您在第一個史詩中建立的 SNS 主題。 9. 針對名稱和描述，提供警示名稱和簡短描述，然後選擇下一步。	AWS 管理員

任務	Description	所需的技能
	10選擇建立警示 。	
設定 SAP HANA 指標的 CloudWatch 指標警示。	針對下列變更，重複設定上一個任務的 CloudWatch 指標警示的步驟： • 針對步驟 5，選擇 SAP HANA 的指標名稱（例如，`sapcluster_db_<abc>` ）。 • 對於步驟 6，將 的閾值`sapcluster_<sid>` 設定為大於 0 的值。	AWS 管理員

相關資源

- 觸發叢集事件的指令碼 (RHEL 文件）
- 使用精靈建立 CloudWatch 代理程式組態檔案 (CloudWatch 文件）
- 在伺服器上安裝和執行 CloudWatch 代理程式 (CloudWatch 文件）
- 根據靜態閾值建立 CloudWatch 警示 (CloudWatch 文件）
- 使用高可用性叢集在 AWS 上手動部署 SAP HANA (AWS 網站上的 SAP 文件）
- SAP NetWeaver 指南 (AWS 網站上的 SAP 文件）

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 CloudWatch Logs Insights 監控應用程式活動

Ram Kandaswamy，Amazon Web Services

總結

此模式提供使用 Amazon CloudWatch Logs Insights 自動偵測和提醒應用程式例外狀況的解決方案。透過實作自動化日誌分析和提醒，您可以快速識別和回應生產環境中的應用程式問題。

日誌在監控系統行為、識別問題並確保最佳效能方面扮演重要角色。在遷移程序期間，日誌檔案對於驗證系統在新環境中的運作、偵測相容性問題，以及識別任何非預期行為來說非常寶貴。問題可能與操作或安全性有關。對於與安全相關的問題，儘早偵測未經授權的存取嘗試或可疑活動對於維護安全和法規合規至關重要。處理敏感資料或關鍵系統時，此功能特別重要。

對於需要執行下列動作的團隊來說，此模式特別重要：

- 維持高可用性的應用程式。
- 快速回應生產問題。
- 分析 AWS 服務 日誌未擷取的應用程式特定錯誤。
- 在沒有預先建置的基礎設施的情況下執行隨需日誌分析。

CloudWatch Logs Insights 最適合用於分析應用程式產生的日誌，其中錯誤內容僅存在於您的應用程式程式碼中。CloudWatch Logs Insights 擅長下列任務：

- 查詢非結構化或半結構化日誌資料。
- 在事件回應期間執行隨需分析。
- 關聯多個日誌群組的事件。
- 在沒有外部工具的情況下建立快速視覺化。

先決條件和限制

先決條件

- 部署在作用中的生產應用程式 AWS 帳戶
- 基本了解生產應用程式的記錄格式和例外狀況模式
- 設定為串流至 Amazon CloudWatch Logs 的應用程式日誌

限制

- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 依區域的服務。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。

Architecture

下圖顯示 CloudWatch Logs Insights 如何評估資源日誌，並將相關資料視覺化傳送至 CloudWatch 儀表板。

該圖顯示以下工作流程：

1. 資源會將日誌發佈至 CloudWatch Logs。資源可以包含 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體或 Amazon Simple Storage Service (Amazon S3) 儲存貯體等 AWS 資源。另一個範例包括已安裝 CloudWatch Agent 的內部部署系統，可將日誌發佈至 CloudWatch。
2. 相關模式字串的 CloudWatch Logs Insights 篩選條件。搜尋模式字串的範例包括「錯誤」、「例外」或特定的規則表達式。
3. 一般而言，生產支援團隊或開發人員會將模式視覺化新增至 CloudWatch 儀表板。

自動化和擴展

開發人員可以使用 AWS Cloud Development Kit (AWS CDK)、 或 AWS SDKs 來處理多個字串模式 AWS CloudFormation，來自動化此模式的解決方案。團隊可以將此自動化納入其持續整合和部署 (CI/CD) DevOps 程序。

工具

AWS 服務

- Amazon CloudWatch Logs 可協助您集中所有系統、應用程式的日誌， AWS 服務 以便您可以監控日誌並將其安全地存檔。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。

最佳實務

查詢效率

- 定義和設定日誌群組以分析相關日誌資料。
- 使用欄位總管來了解日誌資料中可用的結構和欄位。
- 使用 CloudWatch Logs Insights 查詢語法撰寫有效率的查詢。
- 根據您的特定需求調整範例查詢，以加快分析速度。
- 限制查詢時間範圍，以減少掃描的資料並改善效能。
- 儲存查詢以供日後使用，以節省時間並確保分析一致。

安全性

- 將適當的 IAM 政策套用至 CloudWatch Logs Insights 和日誌群組。遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。
- 針對敏感日誌資料使用 啟用日誌資料加密 AWS KMS。

成本最佳化

- 每個查詢掃描的資料每 GB CloudWatch Logs Insights 費用。縮小時間範圍並鎖定特定日誌群組以降低成本。
- 設定適當的日誌保留政策來管理儲存成本。
- 若要經常分析大型歷史資料集，請考慮將日誌匯出至 Amazon S3 並使用 Amazon Athena。
- 檢閱 CloudWatch 定價以了解使用案例的成本影響。

史詩

建立日誌群組並設定日誌以在儀表板中檢視。

任務	Description	所需的技能
設定 IAM 許可。	若要設定 IAM 許可，請執行下列動作： 1. 識別應用程式用於撰寫日誌的 IAM 角色，以及將建立儀	AWS 管理員、AWS DevOps、AWS 系統管理員、雲端管理員、雲端架構師、DevOps 工程師

任務	Description	所需的技能

	表板、查詢和警示的使用者或服務。
	2. 針對應用程式角色，將下列 API 動作和資源新增至政策：
	`{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "logs:CreateLogGroup", "logs:CreateLogStream", "logs:PutLogEvents" ], "Resource": [ "arn:aws:logs:*:*:*" ] } ] }`
	3. 若要管理 CloudWatch 資源，請將下列項目新增至政策：
	`{`	

任務	Description	所需的技能

	<pre>{
	"Version": "2012-10-17",
	"Statement": [
	{
	"Effect": "Allow",
	"Action": [
	"cloudwatch:PutDashboard",
	"cloudwatch:GetDashboard",
	"logs:StartQuery",
	"logs:StopQuery",
	"logs:GetQueryResults",
	"cloudwatch:PutMetricAlarm"
	],
	"Resource": "*"
	}
	]
	}</pre>	

	如需如何建立 IAM 政策或將許可新增至現有政策的詳細資訊，請參閱《IAM 使用者指南》中的使用客戶受管政策定義自訂 IAM 許可和編輯 IAM 政策。 如需詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的 Amazon	

任務	Description	所需的技能
	CloudWatch Logs 的 Identity and Access Management 和 CloudWatch Logs 許可參考。Amazon CloudWatch	

任務	Description	所需的技能

建立 日誌群組	若要建立日誌群組，請使用下列任一選項：
	• 建立或開啟具有 `.yaml`或 `.json`副檔名的 CloudFormation 範本檔案 (YAML 或 JSON 格式）。（下列程式碼使用 YAML 格式。）將下列資源定義新增至範本的資源區段：
	`MyLogGroup:`
	`  Type: AWS::Logs::LogGroup`
	`  Properties:`
	`  LogGroupName: my-log-group`
	如需詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的快速入門：使用 來 AWS CloudFormation 開始使用 CloudWatch Logs。Amazon CloudWatch
	• 使用 CloudWatch 主控台，如 Amazon CloudWatch Logs 使用者指南中的在 CloudWatch Logs 中建立日誌群組中所述。 Amazon CloudWatch
	• 使用 CloudWatch Logs API 中的 CreateLogGroup 操作。
		AWS 管理員、AWS DevOps、AWS 系統管理員、雲端管理員、雲端架構師、DevOps 工程師

任務	Description	所需的技能
產生 CloudWatch Logs Insights 查詢。	若要建立和儲存 CloudWatch Logs Insights 查詢，請執行下列動作： 1. 開啟 CloudWatch 主控台。 2. 導覽至 Logs Insights。 3. 使用下列其中一種方法來建立查詢： • 從查詢區段的範例查詢開始。 • 撰寫自訂查詢。 • 修改下列範例查詢： `fields @timestamp, @message | filter @message like /(?i)exception/|stats count() by bin(30s)` 此查詢會檢查日誌檔案、擷取時間戳記和訊息內容、篩選單字「例外」（不區分大小寫），並以 30 秒的間隔計數出現次數。	AWS 管理員、AWS DevOps、AWS 系統管理員、雲端管理員、雲端架構師、DevOps 工程師

任務	Description	所需的技能
在 CloudWatch 儀表板中建立視覺化。	若要使用 CloudWatch 儀表板建立視覺化效果，請執行下列動作： 1. 開啟 CloudWatch 主控台，並建立 CloudWatch 儀表板（如果您還沒有儀表板）或開啟現有的儀表板。 2. 新增您的例外狀況監控視覺化效果。根據您的業務需求，提供長條圖、折線圖和圓餅圖。如需詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的在 CloudWatch 儀表板上使用小工具。 Amazon CloudWatch 對於即時資料視覺化，您可以自訂小工具以符合您的需求。 如需儀表板選項和功能的詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的使用 Amazon CloudWatch 儀表板和使用儀表板變數建立靈活的 CloudWatch CloudWatch 儀表板。 Amazon CloudWatch	AWS 管理員、AWS DevOps、AWS 系統管理員、雲端管理員、雲端架構師、DevOps 工程師

疑難排解

問題	解決方案
找不到查詢結果或查詢似乎已中斷	從從範例查詢修改的工作查詢開始。對部分查詢（例如篩選條件或欄位）執行小型增量變更，並利用 CloudWatch Logs 查詢產生器功能。
日誌群組未建立日誌串流	在 IAM 政策中，請確定 CreateLogStream 和 CreateLogGroup 操作的資源具有萬用字元(*)值。如果沒有此萬用字元許可，`create` 操作將無法成功。
查詢逾時或效能緩慢	減少時間範圍、鎖定特定日誌群組，或簡化查詢。複雜的規則表達式 (`regex`) 模式和大型時間範圍會增加查詢時間。
沒有傳回有效時間範圍的資料	驗證日誌群組選擇，並檢查日誌是否正在擷取（檢閱日誌串流），並確認篩選條件模式與您的日誌格式相符。

相關資源

- 使用 CloudWatch Logs Insights 分析日誌資料
- Amazon CloudWatch 常見問答集
- 使用儀表板變數建立靈活的 CloudWatch 儀表板
- Logs Insights QL 入門：查詢教學課程
- 使用自然語言來產生和更新 CloudWatch Logs Insights 查詢
- 搭配 AWS SDK 或 CLI 使用 PutDashboard
- 使用日誌群組和日誌串流

跨多個 監控共用 Amazon Machine Image 的使用 AWS 帳戶

Naveen Suthar 和 Sandeep Gawande，Amazon Web Services

總結

Amazon Machine Image (AMIs) 用於在您的 Amazon Web Services () 環境中建立 Amazon Elastic Compute Cloud (Amazon EC2 AWS) 執行個體。您可以在單獨的集中AMIs，在此模式中稱為建立者帳戶。然後，您可以在相同 AWS 帳戶 中的多個 之間共用 AMI AWS 區域，在此模式中稱為取用者帳戶。從單一帳戶管理 AMIs 可提供可擴展性並簡化控管。在消費者帳戶中，您可以參考 Amazon EC2 Auto Scaling 啟動範本和 Amazon Elastic Kubernetes Service (Amazon EKS) 節點群組中的共用 AMI。

當共用 AMI 已棄用、取消註冊或未共用時， AWS 服務 取用者帳戶中的 AMI 無法使用此 AMI 啟動新的執行個體。相同執行個體的任何自動擴展事件或重新啟動都會失敗。這可能會導致生產環境中的問題，例如應用程式停機時間或效能降低。當 AMI 共用和用量事件在多個 中發生時 AWS 帳戶，可能很難監控此活動。

此模式可協助您監控相同區域中各帳戶的共用 AMI 用量和狀態。它使用無伺服器 AWS 服務，例如 Amazon EventBridge、Amazon DynamoDB AWS Lambda和 Amazon Simple Email Service (Amazon SES)。您可以使用 HashiCorp Terraform 將基礎設施佈建為程式碼 (IaC)。此解決方案會在消費者帳戶中的服務參考已取消註冊或未共用的 AMI 時提供提醒。

先決條件和限制

先決條件

- 兩個或多個作用中 AWS 帳戶：一個建立者帳戶和一個或多個取用者帳戶
- 從建立者帳戶與消費者帳戶共用的一或多個 AMIs
- Terraform CLI，已安裝 (Terraform 文件）
- Terraform AWS Provider，已設定 (Terraform 文件）
- （選用，但建議）已設定的 Terraform 後端 (Terraform 文件）
- Git，已安裝

限制

- 此模式會使用帳戶 ID 來監控已與特定帳戶共用的 AMIs。此模式不會監控已使用組織 ID 與組織共用的 AMIs。

- AMIs只能與相同 內的帳戶共用 AWS 區域。此模式會監控單一目標區域內AMIs。若要監控多個區域中 AMIs 的使用，您可以在每個區域中部署此解決方案。
- 此模式不會監控部署此解決方案之前共用的任何 AMIs。如果您想要監控先前共用AMIs，您可以取消共用 AMI，然後與消費者帳戶重新共用。

產品版本

- Terraform 1.2.0 版或更新版本
- Terraform AWS Provider 4.20 版或更新版本

Architecture

目標技術堆疊

下列資源會透過 Terraform 佈建為 IaC：

- Amazon DynamoDB 資料表
- Amazon EventBridge 規則
- AWS Identity and Access Management (IAM) 角色
- AWS Lambda 函數
- Amazon SES

目標架構

該圖顯示以下工作流程：

1. 建立者帳戶中的 AMI 會與相同 中的取用者帳戶共用 AWS 區域。
2. 共用 AMI 時，建立者帳戶中的 EventBridge 規則會擷取`ModifyImageAttribute`事件，並在建立者帳戶中啟動 Lambda 函數。
3. Lambda 函數會將與 AMI 相關的資料儲存在建立者帳戶中的 DynamoDB 資料表中。
4. 當取用 AWS 服務 者帳戶中的 使用共用 AMI 來啟動 Amazon EC2 執行個體，或當共用 AMI 與啟動範本相關聯時，取用者帳戶中的 EventBridge 規則會擷取共用 AMI 的使用。
5. EventBridge 規則會在取用者帳戶中啟動 Lambda 函數。Lambda 函數會執行下列動作：
   a. Lambda 函數會更新消費者帳戶中 DynamoDB 資料表中的 AMI 相關資料。

   b. Lambda 函數會擔任建立者帳戶中的 IAM 角色，並更新建立者帳戶中的 Lambda 資料表。在 `Mapping` 資料表中，它會建立將執行個體 ID 或啟動範本 ID 映射至其個別 AMI ID 的項目。

6. 在建立者帳戶中集中管理的 AMI 已棄用、取消註冊或未共用。
7. 建立者帳戶中的 EventBridge 規則會使用 `remove`動作擷取 `ModifyImageAttribute`或 `DeregisterImage`事件，並啟動 Lambda 函數。
8. Lambda 函數會檢查 DynamoDB 資料表，以判斷 AMI 是否在任何取用者帳戶中使用。如果`Mapping`資料表中沒有與 AMI 相關聯的執行個體 IDs 或啟動範本 IDs，表示程序已完成。
9. 如果任何執行個體 IDs或啟動範本 IDs 與 `Mapping` 資料表中的 AMI 相關聯，則 Lambda 函數會使用 Amazon SES 將電子郵件通知傳送給設定的訂閱者。

工具

AWS 服務

- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如， AWS Lambda 函數、使用 API 目的地的 HTTP 呼叫端點，或其他事件匯流排 AWS 帳戶。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Email Service (Amazon SES) 可協助您使用自己的電子郵件地址和網域來傳送和接收電子郵件。

其他工具

- HashiCorp Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。
- Python 是一種一般用途的電腦程式設計語言。

程式碼儲存庫

此模式的程式碼可在 GitHub cross-account-ami-monitoring-terraform-samples 儲存庫中使用。

最佳實務

- 遵循使用 AWS Lambda 函數的最佳實務。
- 遵循建置 AMIs最佳實務。
- 建立 IAM 角色時，請遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務。
- 設定 AWS Lambda 函數的監控和提醒。如需詳細資訊，請參閱監控和疑難排解 Lambda 函數。

史詩

自訂 Terraform 組態檔案

任務	Description	所需的技能
建立 AWS CLI 具名設定檔。	針對建立者帳戶和每個取用者帳戶，建立名為 的 AWS Command Line Interface (AWS CLI) 設定檔。如需說明，請參閱 AWS 《入門資源中心》中的設定 AWS CLI 。	DevOps 工程師
複製儲存庫。	輸入以下命令。這會使用 SSH 從 GitHub 複製cross-account-ami-monitoring-terraform-samples 儲存庫。 `git clone git@github.com:aws-samples/cross-account-ami-monitoring-terraform-samples.git`	DevOps 工程師
更新 provider.tf 檔案。	1. 輸入下列命令以導覽至複製儲存庫中的 `terraform` 資料夾。	DevOps 工程師

任務	Description	所需的技能
	`cd cross-account-ami-monitoring/terraform` 2. 開啟 `provider.tf` 檔案。 3. 更新建立者帳戶和消費者帳戶的 Terraform AWS 提供者組態，如下所示： • 針對 `alias`，輸入提供者組態的名稱。 • 針對 `region`，輸入 AWS 區域 您要部署此解決方案的目標。 • 針對 `profile`，輸入用於存取帳戶的 AWS CLI 具名設定檔。 4. 如果您要設定多個消費者帳戶，請為每個額外的消費者帳戶建立設定檔。 5. 儲存並關閉 `provider.tf` 檔案。 如需設定提供者的詳細資訊，請參閱 Terraform 文件中的多個提供者組態。	

任務	Description	所需的技能
更新 terraform.tfvars 檔案。	1. 開啟 `terraform.tfvars` 檔案。 2. 在 `account_email_mapping` 參數中，設定建立者帳戶和消費者帳戶的提醒，如下所示： • 針對 `account`，輸入帳戶 ID。 • 針對 `email`，輸入您要傳送提醒的電子郵件地址。每個帳戶只能輸入一個電子郵件地址。 3. 如果您要設定多個消費者帳戶，請為每個額外的消費者帳戶輸入帳戶和電子郵件地址。 4. 儲存並關閉 `terraform.tfvars` 檔案。	DevOps 工程師

任務	Description	所需的技能
更新 main.tf 檔案。	只有在您將此解決方案部署到多個消費者帳戶時，才完成這些步驟。如果您只將此解決方案部署到一個取用者帳戶，則不需要修改此檔案。 1. 開啟 `main.tf` 檔案。 2. 對於每個額外的消費者帳戶，請根據範本中的模組建立新的`consumer_account_A` 模組。對於每個取用者帳戶，對於 `provider`，值應與您在 `provider.tf` 檔案中輸入的別名相符。 3. 儲存並關閉 `main.tf` 檔案。	DevOps 工程師

使用 Terraform 部署解決方案

任務	Description	所需的技能
部署解決方案。	在 Terraform CLI 中，輸入下列命令來部署建立者和取用者帳戶中 AWS 的資源： 1. 輸入下列命令來初始化 Terraform。 `terraform init` 2. 輸入下列命令來驗證 Terraform 組態。	DevOps 工程師

任務	Description	所需的技能

	`terraform validate`
	3. 輸入下列命令來建立 Terraform 執行計畫。
	`terraform plan`
	4. 檢閱 Terraform 計劃中的組態變更，並確認您想要實作這些變更。
	5. 輸入下列命令來部署資源。
	`terraform apply`	

驗證電子郵件地址身分。	當您部署 Terraform 計劃時，Terraform 會為 Amazon SES 中的每個消費者帳戶建立電子郵件地址身分。您必須先驗證電子郵件地址，才能將通知傳送到該電子郵件地址。如需說明，請參閱 Amazon SES 文件中的驗證電子郵件地址身分。 Amazon SES	一般 AWS

驗證資源部署

任務	Description	所需的技能
驗證建立者帳戶中的部署。	1. 登入建立者帳戶。 2. 在導覽列中，確認 正在檢視目標 AWS 區域。如果您位於不同的區域，請選擇目前顯示區域的名稱，然後選擇目標區域。	DevOps 工程師

任務	Description	所需的技能
	3. 開啟 DynamoDB 主控台。 4. 在導覽窗格中，選擇 Tables (資料表)。 5. 在資料表清單中，驗證AmiShare資料表是否存在。 6. 開啟 Lambda 主控台。 7. 在導覽視窗中，選擇函數。 8. 在函數清單中，驗證ami-share 函數是否存在。 9. 開啟 IAM 主控台。 10.在導覽窗格中，選擇 Roles (角色)。 11.在角色清單中，驗證 external-ddb-role 角色是否存在。 12.開啟 EventBridge 主控台。 13.在導覽窗格中，選擇規則。 14.在規則清單中，驗證 modify_image_attribute_event 規則是否存在。 15.開啟 Amazon SES 主控台。 16.在導覽窗格中，選擇已驗證的身分。 17.在身分清單中，驗證每個消費者帳戶的電子郵件地址身分是否已註冊和驗證。	

任務	Description	所需的技能
驗證消費者帳戶中的部署。	1. 登入消費者帳戶。 2. 在導覽列中，確認 正在檢視目標 AWS 區域。如果您位於不同的區域，請選擇目前顯示區域的名稱，然後選擇目標區域。 3. 開啟 DynamoDB 主控台。 4. 在導覽窗格中，選擇 Tables (資料表)。 5. 在資料表清單中，驗證`Mapping`資料表是否存在。 6. 開啟 Lambda 主控台。 7. 在導覽視窗中，選擇函數。 8. 在函數清單中，驗證 `ami-usage-function` 和 `ami-deregister-function` 函數是否存在。 9. 開啟 EventBridge 主控台。 10. 在導覽窗格中，選擇規則。 11. 在規則清單中，驗證 `ami_usage_events` 和 `ami_deregister_events` 規則是否存在。	DevOps 工程師

驗證監控

任務	Description	所需的技能
在建立者帳戶中建立 AMI。	1. 在建立者帳戶中，建立私有 AMI。如需說明，請參閱從	DevOps 工程師

任務	Description	所需的技能
	Amazon EC2 執行個體建立 AMI。 2. 與其中一個消費者帳戶共用新的 AMI。如需說明，請參閱與特定 共用 AMI AWS 帳戶。	
使用消費者帳戶中的 AMI。	在消費者帳戶中，使用共用 AMI 來建立 Amazon EC2 執行個體或啟動範本。如需說明，請參閱如何從自訂 AMI (re：Post 知識中心） 啟動 Amazon EC2 執行個體，或為 Auto Scaling 群組建立啟動範本 (Amazon EC2 Auto Scaling 文件）。AWS	DevOps 工程師
驗證監控和提醒。	1. 登入建立者帳戶。 2. 開啟 Amazon EC2 主控台。 3. 在導覽窗格中，選擇 AMIs (AMI)。 4. 在清單中選取 AMI，然後選擇動作、編輯 AMI 許可。 5. 在共用帳戶區段中，選取取用者帳戶，然後選擇移除已選取。 6. 選擇儲存變更。 7. 驗證您為消費者帳戶定義的目標電子郵件地址，是否收到 AMI 已取消共用的通知。	DevOps 工程師

（選用）停止監控共用 AMIs

任務	Description	所需的技能
刪除資源。	1. 輸入下列命令以移除此模式部署的資源，並停止監控共用 AMIs。 `terraform destroy` 2. 輸入 以確認`destroy`命令`yes`。	DevOps 工程師

疑難排解

問題	解決方案
我未收到電子郵件提醒。	未傳送 Amazon SES 電子郵件的原因可能有多種。請檢查以下內容： 1. 在 Epics 區段中，使用驗證資源部署 epic 來確認基礎設施已完全正確佈建 AWS 帳戶。 2. 驗證 Amazon CloudWatch Logs 中的 Lambda 函數事件。如需說明，請參閱 Lambda 文件中的使用 CloudWatch 主控台。確認沒有許可問題，例如在任何以身分為基礎或以資源為基礎的政策中明確拒絕。如需詳細資訊，請參閱 IAM 文件中的政策評估邏輯。 3. 在 Amazon SES 中，驗證電子郵件地址身分的狀態已驗證。如需詳細資訊，請參閱驗證電子郵件地址身分。

相關資源

AWS 文件

- 使用 Python 建置 Lambda 函數 (Lambda 文件)
- 建立 AMI (Amazon EC2 文件)
- 與特定 (Amazon EC2 文件) 共用 AMI AWS 帳戶 Amazon EC2
- 取消註冊您的 AMI (Amazon EC2 文件)

Terraform 文件

- 安裝 Terraform
- Terraform 後端組態
- Terraform AWS 提供者
- Terraform 二進位下載

檢視 AWS 帳戶或組織的 EBS 快照詳細資訊

Arun Chandapillai 和 Parag Nagwekar，Amazon Web Services

總結

此模式說明如何自動產生 Amazon Web Services (AWS) 帳戶或 AWS Organizations 中的組織單位 (OU) 中所有 Amazon Elastic Block Store (Amazon EBS) 快照的隨需報告。

Amazon EBS 是一種easy-to-use、可擴展、高效能的區塊儲存服務，專為 Amazon Elastic Compute Cloud (Amazon EC2) 而設計。EBS 磁碟區提供耐用且持久的儲存體，您可以連接到 EC2 執行個體。您可以使用 EBS 磁碟區做為資料的主要儲存體，並透過建立快照來取得 EBS 磁碟區的point-in-time備份。您可以使用 AWS 管理主控台或 AWS 命令列界面 (AWS CLI) 來檢視特定 EBS 快照的詳細資訊。此模式提供以程式設計方式擷取您 AWS 帳戶或 OU 中所有 EBS 快照的相關資訊。

您可以使用此模式提供的指令碼來產生逗號分隔值 (CSV) 檔案，其中包含每個快照的下列相關資訊：帳戶 ID、快照 ID、磁碟區 ID 和大小、拍攝快照的日期、執行個體 ID 和描述。如果您的 EBS 快照已標記，報告也會包含擁有者和團隊屬性。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 安裝https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html#getting-started-install-instructions並設定 AWS CLI 第 2 版
- 具有適當許可的 AWS Identity and Access Management (IAM) 角色（如果您打算從 AWS Organizations 執行指令碼，則為特定帳戶或 OU 中所有帳戶的存取許可）

Architecture

下圖顯示指令碼工作流程，該工作流程產生 EBS 快照的隨需報告，這些快照分散在 OU 中的多個 AWS 帳戶。

工具

AWS 服務

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- Amazon Elastic Block Store (Amazon EBS) 提供區塊層級儲存體磁碟區，可與 EC2 執行個體搭配使用。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 AWS 帳戶合併到您建立並集中管理的組織。

Code

此模式中使用的範例應用程式的程式碼可在 aws-ebs-snapshots-awsorganizations 儲存庫的 GitHub 上取得。請依照下一節中的指示使用範例檔案。

史詩

下載指令碼

任務	Description	所需的技能
下載 Python 指令碼。	從 GitHub 儲存庫下載 GetSnapshotDetailsAllAccountsOU.py	一般 AWS

取得 AWS 帳戶的 EBS 快照詳細資訊

任務	Description	所需的技能
執行 Python 指令碼。	執行 命令： `python3 getsnapshotinfo.py --file <output-file>.csv --region <region-name>` 其中 `<output-file>` 是指您想要放置的 EBS 快照相	一般 AWS

任務	Description	所需的技能
	關資訊的 CSV 輸出檔案，而 <region-name> 是存放快照的 AWS 區域。例如： `python3 getsnapshotinfo.py --file snapshots.csv --region us-east-1`	

取得組織的 EBS 快照詳細資訊

任務	Description	所需的技能

執行 Python 指令碼。	執行 命令：
	`python3 getsnapshotinfo.py --file <output-file>.csv --role <IAM-role> --region <region-name>`
	其中 <output-file> 是指您想要放置的 EBS 快照相關資訊的 CSV 輸出檔案、<IAM-role> 提供存取 AWS Organizations 許可的角色，<region-name> 以及存放快照的 AWS 區域。例如：
	`python3 getsnapshotinfo.py --file snapshots.csv --role <IAM role> --region us-west-2`
		一般 AWS

相關資源

- Amazon EBS 文件
- Amazon EBS 動作
- Amazon EBS API 參考
- 改善 Amazon EBS 效能
- Amazon EBS 資源
- EBS 快照定價

其他資訊

EBS 快照類型

Amazon EBS 根據擁有權和存取權提供三種類型的快照：

- 由您擁有 – 預設情況下，只有您可以從您擁有的快照建立磁碟區。
- 公有快照 – 您可以與所有其他 AWS 帳戶公開共用快照。若要建立公有快照，您可以修改快照的許可，以與您指定的 AWS 帳戶共用快照。然後，您將授權的使用者可以透過建立自己的 EBS 磁碟區來使用您共用的快照，而您的原始快照仍然不受影響。您也可以將未加密的快照公開提供給所有 AWS 使用者。不過，基於安全考量，您無法公開您的加密快照。由於可能公開個人和敏感資料，公開快照會造成重大的安全風險。我們強烈建議不要與所有 AWS 帳戶共用您的 EBS 快照。如需共用快照的詳細資訊，請參閱 AWS 文件。
- 私有快照 – 您可以與您指定的個別 AWS 帳戶私下共用快照。若要私下與特定 AWS 帳戶共用快照，請遵循 AWS 文件中的指示，並為許可設定選擇私有。您授權的使用者可使用您共用的快照，以建立他們自己的 EBS 磁碟區，同時原始快照仍然不受影響。

概觀和程序

下表提供 EBS 快照詳細資訊的連結，包括如何透過尋找和刪除未使用的快照來降低 EBS 磁碟區成本，以及封存很少存取且不需要頻繁或快速擷取的快照。

如需 的相關資訊	請參閱
快照、其功能和限制	建立 Amazon EBS 快照
如何建立快照	主控台：建立快照

	AWS CLI：create-snapshot 命令 例如： `aws ec2 create-snapshot --volume-id vol-1234567890abcdef0 --description " volume snapshot"`
刪除快照（一般資訊）	刪除 Amazon EBS 快照
如何刪除快照	主控台：刪除快照 AWS CLI：Delete-snapshot 命令 例如： `aws ec2 delete-snapshot --snapshot-id snap-1234567890abcdef0`
封存快照（一般資訊）	封存 Amazon EBS 快照 Amazon EBS 快照封存（部落格文章）
如何封存快照	主控台：封存快照 AWS CLI： modify-snapshot-tier 命令
如何擷取封存的快照	主控台：還原封存的快照 AWS CLI： restore-snapshot-tier 命令
快照定價	Amazon EBS 定價

常見問答集

最短封存期間為何？

最短封存期間為 90 天。

還原封存快照需要多長時間？

將封存的快照從封存層還原到標準層最多可能需要 72 小時，取決於快照的大小。

封存的快照是否完整快照？

封存的快照一律是完整快照。

使用者可以封存哪些快照？

您只能封存您在帳戶中擁有的快照。

您可以封存已註冊 Amazon Machine Image (AMI) 的根裝置磁碟區的快照嗎？

否，您無法封存已註冊 AMI 根裝置磁碟區的快照。

共用快照的安全考量是什麼？

當您共用快照時，您會讓其他人存取快照上的所有資料。僅與您信任資料的人員共用快照。

如何與其他 AWS 區域共用快照？

快照受限於其建立的區域。若要與另一個區域共用快照，請將該快照複製到該區域，然後共用。

您可以共用已加密的快照嗎？

您無法共用使用預設 AWS 受管金鑰加密的快照。您只能共用使用客戶受管金鑰加密的快照。當您共用加密快照時，也必須共用用來加密快照的客戶受管金鑰。

未加密的快照呢？

您可以公開共用未加密的快照。

更多模式

- 在 上使用登陸區域加速器自動建立帳戶 AWS
- 使用 Amazon Bedrock 自動化 AWS 基礎設施操作
- 自動化 AWS 資源評估
- 跨多個帳戶和區域自動清查 AWS 資源
- 使用 AWS CDK 自動化 AWS Service Catalog 產品組合和產品部署
- 使用 AWS Service Catalog 和 自動化動態管道管理，以在 Gitflow 環境中部署 Hotfix 解決方案 AWS CodePipeline
- 使用 Terraform 在 Amazon Managed Grafana 上自動化 Amazon MWAA 自訂指標的擷取和視覺化
- 使用 Cloud Custodian 和 AWS CDK 將 Systems Manager 的 AWS 受管政策自動連接至 EC2 執行個體設定檔
- 自動加密現有和新的 Amazon EBS 磁碟區
- 建置包含 MongoDB Atlas 的 AWS 登陸區域
- 使用 Amazon CloudWatch Observability Access Manager 集中監控
- 在啟動時檢查 EC2 執行個體是否有強制性標籤
- 在狀態檔案遺失後，安全地清除 AWS Account Factory for Terraform (AFT) 資源
- 使用 Amazon EFS 在 EC2 執行個體上建立 Amazon ECS 任務定義並掛載檔案系統
- 使用 AWS CloudFormation Guard 政策建立 AWS Config 自訂規則
- 使用 AWS CDK 層面和逃生艙自訂預設角色名稱
- 使用 和 AWS CDK CloudFormation 部署和管理 AWS Control Tower 控制項
- 使用 Terraform 部署和管理 AWS Control Tower 控制項
- 使用 AWS CodePipeline、AWS CodeCommit 和 AWS CodeBuild 在多個 AWS 區域中部署程式碼
- 在做為 Docker 容器 AWS IoT Greengrass V2 執行時部署容器化應用程式
- 使用 AWS CloudFormation 範本有條件地啟用 Amazon GuardDuty
- 在 IBM Db2 資料庫中啟用直接封存至 Amazon S3 的 DB2 日誌 Amazon S3 Db2
- 使用 PowerShell 匯出 AWS IAM Identity Center 身分及其指派的報告
- 使用 Troposphere 產生包含 AWS Config 受管規則的 AWS CloudFormation 範本
- 讓 SageMaker 筆記本執行個體暫時存取另一個 AWS 帳戶中的 CodeCommit 儲存庫
- 將stonebranch 通用控制器與 AWS Mainframe Modernization 整合
- 使用 Step Functions 和 Lambda 代理函數跨 AWS 帳戶啟動 CodeBuild 專案

- 使用 Terraform 動態管理 AWS 許可集
- 使用 appcmd.exe 將 IIS 託管應用程式遷移至 Amazon EC2
- 使用 ACM 將 Windows SSL 憑證遷移至 Application Load Balancer
- 監控 IAM 根使用者活動
- AWS 使用 Terraform 在 上建立階層式多區域 IPAM 架構
- 使用 AWS CDK 和 GitHub Actions 工作流程最佳化多帳戶無伺服器部署
- 在非工作負載子網路的多帳戶 VPC 設計中保留可路由 IP 空間
- 透過部署角色販賣機解決方案來佈建最低權限的 IAM 角色
- 使用 Amazon SES AWS 帳戶 向單一電子郵件地址註冊多個
- 使用 AWS Lambda 自動化 AWS 帳戶 AWS Managed Microsoft AD 從 移除 Amazon EC2 項目
- 使用 AWS Lambda 自動化 AWS 帳戶 從 移除相同 中的 Amazon EC2 AWS Managed Microsoft AD 項目
- 使用 AWS Fargate 大規模執行事件驅動和排程工作負載
- 使用 Amazon Macie 保護 CloudWatch Logs 中的敏感資料
- 使用內部部署 SMTP 伺服器和 Database Mail 傳送 Amazon RDS for SQL Server 資料庫執行個體的通知
- 設定 AWS ParallelCluster 的 Grafana 監控儀表板
- 使用 Terraform 設定企業規模的集中式記錄
- 在 AWS 上設定 SAP on IBM Db2 的災難復原
- 使用 Amazon Bedrock 代理程式和 簡化 Amazon EC2 合規管理 AWS Config
- 使用 AWS Organizations 自動標記 Transit Gateway 連接
- 使用 BMC Discovery 查詢來擷取遷移資料以進行遷移規劃
- 使用 驗證 PCI DSS 4.0 的操作最佳實務 AWS Config
- 使用 Splunk 檢視 AWS Network Firewall 日誌和指標
- 使用 Amazon Quick Sight 視覺化所有 AWS 帳戶的 IAM 登入資料報告

訊息與通訊

主題

- 在 Amazon MQ 中自動化 RabbitMQ 組態 Amazon MQ
- 改善 Amazon Connect 聯絡中心中客服人員工作站的通話品質
- 更多模式

在 Amazon MQ 中自動化 RabbitMQ 組態 Amazon MQ

Yogesh Bhatia 和 Afroz Khan，Amazon Web Services

總結

Amazon MQ 是一種受管訊息代理程式服務，可與許多熱門訊息代理程式相容。搭配使用 Amazon MQ 與 RabbitMQ 提供強大的 RabbitMQ 叢集，在 Amazon Web Services (AWS) 雲端中受管，具有多個代理程式和組態選項。Amazon MQ 提供高可用性、安全和可擴展的基礎設施，並且可以輕鬆處理每秒大量訊息。多個應用程式可以使用具有不同虛擬主機、佇列和交換的基礎設施。不過，管理這些組態選項或手動建立基礎設施可能需要時間和精力。此模式描述透過單一檔案，在單一步驟中管理 RabbitMQ 組態的方法。您可以在任何持續整合 (CI) 工具中嵌入此模式提供的程式碼，例如 Jenkins 或 Bamboo。

您可以使用此模式來設定任何 RabbitMQ 叢集。只需要連線至叢集。雖然管理 RabbitMQ 組態有許多其他方式，但此解決方案會在一個步驟中建立整個應用程式組態，因此您可以輕鬆管理佇列和其他詳細資訊。

先決條件和限制

先決條件

- 安裝並設定 AWS Command Line Interface (AWS CLI) 以指向您的 AWS 帳戶（如需說明，請參閱 AWS CLI 文件）
- 已安裝 Ansible，因此您可以執行手冊來建立組態
- 已安裝 randommqadmin（如需說明，請參閱 RabbitMQ 文件）
- Amazon MQ 中的 RabbitMQ 叢集，使用運作狀態良好的 Amazon CloudWatch 指標建立

其他要求

- 請務必分別建立虛擬主機和使用者的組態，而不是做為 JSON 的一部分。
- 請確定組態 JSON 是 儲存庫的一部分，且受版本控制。
- 偵錯工具 CLI 的版本必須與 RabbitMQ 伺服器版本相同，因此最佳選項是從 RabbitMQ 主控台下載 CLI。
- 作為管道的一部分，請確保在每次執行之前驗證 JSON 語法。

產品版本

- AWS CLI 2.0 版
- Ansible 2.9.13 版
- visitorsmqadmin 3.9.13 版（必須與 RabbitMQ 伺服器版本相同）

Architecture

來源技術堆疊

- 在現有內部部署虛擬機器 (VM) 或 Kubernetes 叢集（內部部署或雲端）上執行的 RabbitMQ 叢集

目標技術堆疊

- Amazon MQ for RabbitMQ 上的自動化 RabbitMQ 組態

目標架構

設定 RabbitMQ 的方法有很多種。此模式使用匯入組態功能，其中單一 JSON 檔案包含所有組態。此檔案會套用所有設定，並可由 Bitbucket 或 Git 等版本控制系統進行管理。此模式使用 Ansible 透過附屬管理 CLI 實作組態。

工具

工具

- caughtmqadmin 是 RabbitMQ HTTP 型 API 的命令列工具。它用於管理和監控 RabbitMQ 節點和叢集。
- Ansible 是用於自動化應用程式和 IT 基礎設施的開放原始碼工具。
- AWS CLI 可讓您使用命令列 shell 中的命令與 AWS 服務互動。

AWS 服務

- Amazon MQ 是一種受管訊息中介裝置服務，可讓您輕鬆地在雲端中設定和操作訊息中介裝置。
- AWS CloudFormation 可協助您設定 AWS 基礎設施，並使用基礎設施做為程式碼來加速雲端佈建。

Code

此模式中使用的 JSON 組態檔案和範例 Ansible 手冊提供於附件中。

史詩

建立您的 AWS 基礎設施

任務	Description	所需的技能
在 AWS 上建立 RabbitMQ 叢集。	如果您還沒有 RabbitMQ 叢集，您可以使用 AWS CloudFormation 在 AWS 上建立堆疊。或者，您可以使用 Ansible 中的 Cloudformation 模組來建立堆疊。使用後者方法，您可以針對這兩個任務使用 Ansible： 來建立 RabbitMQ 基礎設施和管理組態。	AWS CloudFormation、Ansible

建立 Amazon MQ for RabbitMQ 組態

任務	Description	所需的技能
建立屬性檔案。	在附件中下載 JSON 組態檔案 (`rabbitmqconfig.json`)，或從 RabbitMQ 主控台匯出。 修改它以設定佇列、交換和繫結。此組態檔案示範下列項目： - 建立兩個佇列： `sample-queue1` 和 `sample-queue2` - 建立兩個交換： `sample-exchange1` 和 `sample-exchange2` - 實作佇列和交換之間的繫結	JSON

任務	Description	所需的技能
	這些組態會視需要在 root (/) 虛擬主機下執行。	
擷取 Amazon MQ for RabbitMQ 基礎設施的詳細資訊。	擷取 AWS 上 RabbitMQ 基礎設施的下列詳細資訊： • 代理程式名稱 • RabbitMQ 主機 • RabbitMQ 使用者名稱（在叢集建立期間建立的管理員使用者） • RabbitMQ 密碼 您可以使用 AWS 管理主控台或 AWS CLI 來擷取此資訊。這些詳細資訊可讓 Ansible 手冊連線至您的 AWS 帳戶，並使用 RabbitMQ 叢集來執行命令。 ⚠ Important 執行 Ansible 手冊的電腦必須能夠存取您的 AWS 帳戶，而且必須已設定 AWS CLI，如先決條件一節中所述。	AWS CLI、Amazon MQ

任務	Description	所需的技能

建立 hosts_var 檔案。	為 Ansible 建立 `hosts_var` 檔案，並確定檔案中已定義所有變數。考慮使用 Ansible Vault 來存放密碼。您可以設定 `hosts_var` 檔案，如下所示（以您的資訊取代星號）：
	`RABBITMQ_HOST: "**********.mq.us-east-2.amazonaws.com"`
	`RABBITMQ_VHOST: "/"`
	`RABBITMQ_USERNAME: "admin"`
	`RABBITMQ_PASSWORD: "*******"`
		Ansible

建立 Ansible 手冊。	如需範例手冊，請參閱附件`ansible-rabbit-config.yaml` 中的 。下載並儲存此檔案。Ansible 手冊會匯入和管理應用程式所需的所有 RabbitMQ 組態，例如佇列、交換和繫結。 遵循 Ansible 手冊的最佳實務，例如保護密碼。使用 Ansible Vault 進行密碼加密，並從加密的檔案擷取 RabbitMQ 密碼。	Ansible

部署組態

任務	Description	所需的技能
執行 手冊。	執行您在上一個史詩中所建立的 Ansible 手冊。 `ansible-playbook ansible-rabbit-config.yaml` 您可以在 RabbitMQ 主控台上驗證新組態。	RabbitMQ、Amazon MQ、Ansible

相關資源

- 從 RabbitMQ 遷移至 Amazon MQ (AWS 部落格文章)
- Management Command Line Tool (RabbitMQ 文件)
- 建立或刪除 AWS CloudFormation 堆疊 (Ansible 文件)
- 將訊息驅動的應用程式遷移至 Amazon MQ for RabbitMQ (AWS 部落格文章)

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

改善 Amazon Connect 聯絡中心中客服人員工作站的通話品質

Ernest Ozdoba，Amazon Web Services

摘要

通話品質問題是聯絡中心疑難排解最困難的問題之一。若要避免語音品質問題和複雜的疑難排解程序，您必須最佳化客服人員的工作環境和工作站設定。此模式說明 Amazon Connect 聯絡中心中客服人員工作站的語音品質最佳化技術。它在以下領域提供建議：

- 工作環境調整。客服人員的周圍環境不會影響語音透過網路傳輸的方式，但確實會影響通話品質。
- 客服人員工作站設定。聯絡中心工作站的硬體和網路組態會對通話品質產生重大影響。
- 瀏覽器設定。客服人員使用 Web 瀏覽器存取 Amazon Connect 聯絡控制面板 (CCP) 網站並與客戶通訊，因此瀏覽器設定可能會影響通話品質。

下列元件也可能影響通話品質，但它們不在工作站範圍內，且不在此模式中涵蓋：

- 透過 AWS Direct Connect、完整通道 VPN 或分割通道 VPN 流向 Amazon Web Services (AWS) 雲端的流量
- 在公司辦公室內外工作時的網路條件
- 公有交換電話網路 (PSTN) 連線
- 客戶的裝置和電話電信業者
- 虛擬桌面基礎設施 (VDI) 設定

如需有關這些區域的詳細資訊，請參閱 Amazon Connect 文件中的常見聯絡控制面板 (CCP) 問題和使用端點測試公用程式。

先決條件和限制

先決條件

- 耳機和工作站必須符合 Amazon Connect 管理員指南中指定的要求。

限制

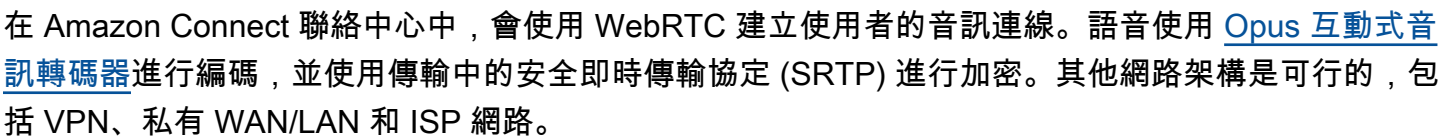
- 此模式中的最佳化技術適用於軟電話語音品質。當您在桌面電話模式下設定 Amazon Connect CCP 時，它們不適用。不過，如果您的軟電話設定沒有為通話提供可接受的語音品質，您可以使用桌面電話模式。

產品版本

- 如需支援的瀏覽器和版本，請參閱 Amazon Connect 管理員指南。

Architecture

此模式與架構無關，因為它以客服人員工作站設定為目標。如下圖所示，從客服人員到客戶的語音路徑會受到客服人員的耳機、瀏覽器、作業系統、工作站硬體和網路的影響。



在 Amazon Connect 聯絡中心中，會使用 WebRTC 建立使用者的音訊連線。語音使用 Opus 互動式音訊轉碼器進行編碼，並使用傳輸中的安全即時傳輸協定 (SRTP) 進行加密。其他網路架構是可行的，包括 VPN、私有 WAN/LAN 和 ISP 網路。

工具

- Amazon Connect Endpoint Test Utility – 此公用程式會檢查網路連線和瀏覽器設定。
- WebRTC 設定的瀏覽器組態編輯器：
  - 對於 Firefox：關於：config
  - 對於 Chrome：chrome：//flags
- CCP 日誌剖析器 – 此工具可協助您分析 CCP 日誌以進行故障診斷。

史詩

調整工作環境

任務	Description	所需的技能
減少背景雜訊。	避免嘈雜的環境。如果無法做到這一點，請使用這些隔音秘訣來最佳化環境：	客服人員、管理員

任務	Description	所需的技能
	• 使用遮光板、地毯和軟家具等消音表面來吸收噪音。 • 在桌面之間放置障礙物來封鎖雜訊。 • 考慮主動降噪 (ANC) 解決方案，例如白色降噪產生器，以協助集中注意力並確保隱私，或使用降噪耳機。 • 防止您的 呼叫出現回應。大型空白空間可能會產生回音效果或放大雜訊。覆蓋可以退信聲音的表面有助於減少回音。	

最佳化客服人員工作站設定

任務	Description	所需的技能
選擇正確的耳機。	• 如果環境有雜訊，請選擇立體聲耳機。將聲音導向雙耳有助於客服人員更好地聚焦和聆聽客戶，並降低客服人員提高聲音的可能性，從而降低整體噪音。 • 避免使用響亮的喇叭或內建的電腦音訊。為了獲得最佳品質，請使用專用於聯絡中心的有線耳機。無線耳機很方便，但由於無線電干擾和轉碼，它們可能是其他音訊延遲和音訊品質降低的來源。	客服人員、管理員

任務	Description	所需的技能
如預期使用耳機。	• 如果可用，請啟用耳機的主動降噪 和語音增強功能。尋找 ANC 或 ANR 等設定。如需啟用這些設定的指示，請參閱耳機的使用手冊。 • 調整您的麥克風，讓您可以直接對它說話。麥克風的最佳位置就在下巴下方。正確放置可以在聲音層級產生 10 分貝 (dB) 的差異。大多數耳機可讓您旋轉或彎曲麥克風臂 （突起），因此請務必在說話時將其保持在正確的位置。 • 有些耳機配備多個麥克風和語音波束形成等進階功能，有助於擷取語音，而無須繁重。為了確保您按照製造商的預期使用主麥克風，請參閱裝置的 使用者手冊。	客服人員
檢查工作站資源。	請確定客服人員的電腦具有效能。如果他們使用耗用資源的第三方應用程式，其電腦可能不符合執行 CCP 的最低硬體需求。如果客服人員遇到通話品質問題，請確定他們有足夠的處理能力 (CPU)、磁碟空間、網路頻寬和記憶體可供 CCP 使用。客服人員應關閉任何不必要的應用程式和索引標籤，以改善 CCP 效能和通話品質。	管理員

任務	Description	所需的技能
設定作業系統的聲音設定。	麥克風關卡和提升的預設設定通常可以正常運作。如果您發現外撥語音是安靜的，或麥克風的接聽過多，可能有助於調整這些設定。您可以在電腦的系統音效組態中找到麥克風設定 (音效、MacOS 上的輸入、Windows 中的麥克風屬性)。您可以透過系統工具或第三方應用程式存取可能影響語音品質的進階設定。以下是您可以檢查的一些設定： • 取樣率 – 此值決定每秒探測聲音的次數。預設設定通常是 44 或 48 kilohertz (kHz)。Amazon Connect 的最佳值為 48 kHz。您可以使用瀏覽器設定來覆寫預設值。如需詳細資訊，請參閱《Amazon Connect 管理員指南》中的疑難排解一節。 • 增益 – 此值決定麥克風放大聲音的程度。如果您提高增益，您的麥克風可能會產生更多背景雜訊。 • 位元深度 – 此數位解析度值會描述要辨識多少等級的聲音振幅。位元深度越高，語音聲音就越平滑。不過，許多傳統電話網路都使用脈衝碼調變 (PCM) 標準，僅支援 8 位元解析度。	客服人員、管理員

任務	Description	所需的技能
	• 開啟閾值 – 這是麥克風拾取的最小聲音振幅。 如果您遇到語音品質問題，請嘗試將這些值還原為其預設設定，然後再進一步調查。 如需這些和其他可調整設定的詳細資訊，請參閱裝置手冊。	

任務	Description	所需的技能
使用有線網路。	一般而言，有線乙太網路的延遲較低，因此更容易提供語音資料傳輸所需的一致傳輸品質。 我們建議每次呼叫至少 100 KB 頻寬。 • 如果客服人員在家中工作，我們建議透過無線連線進行有線連線。聽到客戶的聲音應該不會超過 150 毫秒。您可以從 Amazon Connect Endpoint Test Utility 存取 Amazon Connect 的延遲測試。不過，此公用程式會測量從瀏覽器到 Amazon Connect 區域的延遲，而不是客戶。150 毫秒的單向延遲建議可防止客服人員和客戶彼此交談。該值會從端對端測量，每個元素都會新增延遲，包括 Amazon Connect 區域和客戶之間的呼叫部分。 • 如果客服人員從辦公室工作，只要參數在建議範圍內，且即時傳輸協定 (RTP) 流量為優先，就可以接受公司 Wi-Fi。	網路管理員、客服人員

任務	Description	所需的技能
更新硬體驅動程式。	當您使用具有自己的韌體的 USB 或其他類型的耳機時，我們建議您使用最新版本來保持其最新狀態。使用輔助連接埠的簡單耳機使用電腦的內建音訊裝置，因此請確定作業系統硬體驅動程式是最新的。在極少數情況下，音訊驅動程式更新可能會導致音訊問題，您可能需要將其轉返。如需變更韌體和驅動程式版本的詳細資訊，請參閱裝置手冊。	管理員
避免 USB 集線器和硬體鎖。	當您連接耳機時，請避免使用其他裝置，例如硬體鎖、連接埠類型轉換器、集線器和延長纜線。 這些裝置可能會影響通話品質。請改為將裝置直接連接到電腦的連接埠。	客服人員

任務	Description	所需的技能
檢查 CCP 日誌。	CCP Log Parser 提供簡單的方式來檢查應用程式日誌。 1. 通話後下載 CCP 日誌。 2. 開啟 CCP 日誌剖析器。 3. 拖放日誌檔案以上傳日誌進行分析。 4. 分析日誌後，預設會選取快照和日誌索引標籤。選擇指標旁的指標索引標籤，以檢查洞見。 5. 在 WebRTC 指標 - audio_input 區段中，檢查下列項目： • 音訊層級圖表，查看收到的音訊層級是否高於 0。這表示已從來電者接收音訊。 • 任何遺失封包的封包圖表。如果此圖表顯示大幅增加，請聯絡您的 IT 支援團隊。 6. 在 WebRTC 指標 - audio_output 區段中，檢查下列項目： • 音訊層級圖表，確認音訊已從您的裝置送出。 • 封包圖表。如果您看到封包遺失峰值，請向 IT 支援團隊報告。 • 抖動緩衝區和 RTT 圖形。超過 300 的往返時間 (RTT) 值會影響通話體	客服人員 （進階技能）

任務	Description	所需的技能
	驗。向您的 IT 支援團隊報告這些問題。	

最佳化瀏覽器設定

任務	Description	所需的技能
還原預設 WebRTC 設定。	必須啟用 WebRTC，才能使用 CCP 進行軟性通話。我們建議您保留 WebRTC 相關功能的預設設定。 • 在 Chrome 中，您可以透過導覽至 URL chrome：//flags 來設定旗標。在搜尋方塊中輸入 WebRTC，以尋找可能干擾 CCP 的設定，並將這些設定設為預設。 • 在 Firefox 中，在地址列中輸入 about：config，然後在組態頁面上的搜尋方塊中輸入 WebRTC。非預設設定會以粗體文字顯示，並可變更為預設值。	管理員
故障診斷時停用瀏覽器擴充功能。	有些瀏覽器擴充功能可能會影響通話品質，甚至防止通話正常連線。在瀏覽器中使用 incognito 視窗或私有模式，並停用所有擴充功能。如果這樣可以解決問題，請檢閱您的瀏覽器延伸模組並尋找可疑的附加元件，或個別停用。	客服人員、管理員

任務	Description	所需的技能
檢查瀏覽器取樣率。	確認您的麥克風輸入已設定為最佳 48 kHz 取樣率。如需說明，請參閱 Amazon Connect 管理員指南。	客服人員、管理員

相關資源

如果您已遵循此模式中的步驟，但仍遇到通話品質的問題，請參閱下列資源以取得疑難排解秘訣。

- 檢閱常見的聯絡控制面板 (CCP) 問題。
- 檢查與 Endpoint Test Utility 的連線。
- 針對任何其他問題，請遵循疑難排解指南 。

如果您的疑難排解和調整無法解決呼叫品質問題，根本原因可能是工作站外部。如需進一步疑難排解，請聯絡您的 IT 支援團隊。

更多模式

- 使用 CQRS 和事件來源將整體分解為微服務
- 在聊天應用程式自訂動作和 中使用 Amazon Q Developer 部署 ChatOps 解決方案來管理 SAST 掃描結果 CloudFormation
- 將 Amazon API Gateway 與 Amazon SQS 整合，以處理非同步 REST APIs
- 使用自動化工作流程簡化 Amazon Lex 機器人開發和部署

多帳戶策略

主題

- 將 AWS 成員帳戶從 遷移 AWS Organizations 至 AWS Control Tower
- 在 AWS Organizations 中設定程式設計帳戶關閉提醒
- 更多模式

將 AWS 成員帳戶從 遷移 AWS Organizations 至 AWS Control Tower

Rodolfo Jr. Cerrada，Amazon Web Services

總結

此模式說明如何 AWS 帳戶 從 遷移 AWS Organizations，其中 是受管理帳戶管理的成員帳戶 AWS Control Tower。透過在 中註冊帳戶 AWS Control Tower，您可以利用預防性和偵測性控制和功能來簡化您的帳戶控管。如果您的 AWS Organizations 管理帳戶遭到入侵，而且您想要將成員帳戶移至由 管理的新組織，您可能也會想要遷移您的成員帳戶 AWS Control TowerAWS Control Tower。

AWS Control Tower 提供架構，可結合和整合其他多種功能 AWS Organizations， AWS 服務包括並確保跨多帳戶環境的一致性合規和管理。透過 AWS Control Tower，您可以遵循一組延伸 功能的規定規則和定義 AWS Organizations。例如，您可以使用控制項來確保安全日誌和必要的跨帳戶存取許可已建立，而不會變更。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- AWS Control Tower 在目標組織中設定 AWS Organizations （如需說明，請參閱 AWS Control Tower 文件中的設定)
- 的管理員登入資料 AWS Control Tower (AWSControlTowerAdmins 群組的成員）
- 來源的管理員登入資料 AWS 帳戶

限制

- 中的來源管理帳戶 AWS Organizations 必須與 中的目標管理帳戶不同 AWS Control Tower。

產品版本

- AWS Control Tower 2.3 版 (2020 年 2 月）或更新版本 （請參閱版本備註)

Architecture

下圖說明遷移程序和參考架構。此模式會將 AWS 帳戶 從來源組織遷移到由 管理的目標組織 AWS Control Tower。

註冊程序包含下列步驟：

1. 目標組織會傳送邀請給帳戶加入組織。
2. 帳戶接受邀請，並成為目標組織的成員。
3. 帳戶已註冊 AWS Control Tower 並移至已註冊的組織單位 (OU)。（我們建議您檢查 AWS Control Tower 儀表板以確認註冊。）此時，在註冊的 OU 中啟用的所有控制項都會生效。

工具

AWS 服務

- AWS Organizations 是一種帳戶管理服務，可讓您將多個 合併 AWS 帳戶 到您建立並集中管理的單一實體 (組織)。
- AWS Control Tower 整合了其他服務的功能 AWS Organizations，包括 AWS IAM Identity Center和 AWS Service Catalog，以協助您在 中的所有組織和帳戶中大規模強制執行和管理安全、操作和合規的控管規則 AWS 雲端。

史詩

邀請帳戶加入新組織 AWS Control Tower

任務	Description	所需的技能
登入 AWS Control Tower。	以管理員身分登入 AWS Control Tower 主控台。 目前，無法直接將 AWS 帳戶 從來源組織移動到由 管理的 OU 中的組織 AWS Control Tower。不過，當您將其註冊到已受 管理的 OU AWS 帳戶 時，您可以將 AWS Control Tower 管控擴展到現有的 AWS Control Tower。這就是為什	AWS Control Tower 管理員

任務	Description	所需的技能
	麼您必須在此步驟登入 AWS Control Tower 。	
邀請成員帳戶。	1. 登入 AWS Organizations 主控台，然後導覽至 AWS 帳戶頁面。 2. 在新增 AWS 帳戶頁面上，選擇邀請現有的 AWS 帳戶。 3. 完成帳戶資訊，包括 12 位數的帳號（不含破折號）以及選用的描述和標籤，然後選擇傳送邀請。 **Important** 確認帳戶轉移不會影響任何應用程式或網路連線。 此動作會傳送邀請電子郵件，其中包含成員帳戶的連結。當帳戶管理員遵循連結並接受邀請時，成員帳戶會出現在 AWS 帳戶頁面中。如需詳細資訊，請參閱 AWS Organizations 文件中的管理帳戶邀請。	AWS Control Tower 管理員

任務	Description	所需的技能
測試應用程式和連線。	當成員帳戶已註冊到新組織時，它會出現在根目錄中的 OU 中。它也會出現在AWS Control Tower 主控台中，標記為未註冊帳戶，因為它尚未註冊到 AWS Control Tower 已註冊的 OU。 請確認下列內容： • 檢查 AWS Control Tower 儀表板以查看是否有任何護欄違規。 • 檢查網路連線 (VPN 或 AWS Direct Connect)，確保它不受傳輸影響。 • （應用程式擁有者）測試與此帳戶相關聯的應用程式，以確認它們如預期般執行，而且相依性不受帳戶轉移影響。	AWS Control Tower 管理員、成員帳戶管理員、應用程式擁有者

準備帳戶以進行註冊

任務	Description	所需的技能
檢閱控制項並修正任何違規。	檢閱目標 OU 中定義的控制項，特別是預防性控制項，並修正任何違規。 設定 AWS Control Tower 登陸區域時，預設會啟用一些強制性的預防性控制。這些無法停用。您必須先檢閱這些強制性控制項並修正成員帳戶（手動	AWS Control Tower 管理員、成員帳戶管理員

任務	Description	所需的技能
	或使用指令碼），才能註冊帳戶。 **Note** 預防性控制可保持 AWS Control Tower 已註冊帳戶的合規性，並防止違反政策。任何違反預防性控制可能會影響註冊。成功註冊後，偵測到的控制違規會顯示在 AWS Control Tower 儀表板中，如果偵測到的話。它們不會影響註冊程序。如需詳細資訊，請參閱 AWS Control Tower 文件中的關於控制項。	
修正控制違規後檢查連線問題。	在某些情況下，您可能需要關閉特定連接埠或停用服務，以修正控制違規。在您註冊帳戶之前，請確定已修復使用這些連接埠和服務的應用程式。	應用程式擁有者

將帳戶註冊至 AWS Control Tower

任務	Description	所需的技能
登入 AWS Control Tower。	登入 AWS Control Tower 主控台。使用具有管理許可的登入憑證 AWS Control Tower。請勿使用根使用者（管理帳	AWS Control Tower 管理員

任務	Description	所需的技能
	戶）登入資料來註冊 AWS Organizations 帳戶。這會顯示錯誤訊息。	
註冊 帳戶。	1. 在 中的帳戶工廠頁面中 AWS Control Tower，選擇註冊帳戶。 2. 填寫詳細資訊，包括與您要註冊的帳戶相關聯的電子郵件地址、將出現在其中的顯示名稱 AWS Control Tower、IAM Identity Center 電子郵件地址、帳戶擁有者的名字和姓氏，以及您要註冊帳戶的 OU。IAM Identity Center 電子郵件地址是您偏好的使用者電子郵件地址。您可以使用與帳戶電子郵件相同的電子郵件地址。 3. 選擇 Enroll acount (註冊帳戶)。 如需詳細資訊，請參閱 AWS Control Tower 文件中的關於註冊現有帳戶。	AWS Control Tower 管理員

註冊後驗證帳戶

任務	Description	所需的技能
驗證帳戶。	從 AWS Control Tower中選擇帳戶。您剛註冊的帳戶具	AWS Control Tower 管理員、成員帳戶管理員

任務	Description	所需的技能
	有初始註冊狀態。註冊完成時，其狀態會變更為已註冊。	
檢查控制違規。	OU 中定義的控制項會自動套用至已註冊的成員帳戶。監控 AWS Control Tower 儀表板是否有違規，並相應地修正。如需詳細資訊，請參閱 AWS Control Tower 文件中的關於控制項。	AWS Control Tower 管理員、成員帳戶管理員

疑難排解

問題	解決方案
您會收到錯誤訊息：發生未知錯誤。請稍後再試，或聯絡 AWS Support。	當您在 中使用根使用者登入資料（管理帳戶）AWS Control Tower 註冊新帳戶時，會發生此錯誤。 AWS Service Catalog 無法將帳戶工廠產品組合或產品對應至根使用者，這會導致錯誤訊息。若要修復此錯誤，請使用非根、完整存取的使用者（管理員）登入資料來註冊新帳戶。如需如何將管理存取權指派給管理使用者的詳細資訊，請參閱 IAM Identity Center 文件中的入門。
AWS Control Tower 活動頁面會顯示取得災難性偏離動作。	此動作會反映服務的偏離檢查，並不表示 AWS Control Tower 設定的任何問題。無需採取任何動作。

相關資源

文件

- 術語和概念 (AWS Organizations 文件）
- 什麼是 AWS Control Tower？(AWS Control Tower 文件）

- 從組織移除成員帳戶 (AWS Organizations 文件）
- 設定 (AWS Control Tower 文件）

教學課程和影片

- AWS Control Tower 研討會 （自行安排進度的研討會）
- 什麼是 AWS Control Tower？（影片）
- 在 中佈建使用者 AWS Control Tower（影片）

在 AWS Organizations 中設定程式設計帳戶關閉提醒

Richard Milner-Watts、Debojit Bhadra 和 Manav Yadav，Amazon Web Services

總結

適用於 AWS Organizations 的 CloseAccount API 可讓您以程式設計方式關閉組織內的成員帳戶，而不必使用根登入資料登入該帳戶。RemoveAccountFromOrganization API 會從 AWS Organizations 中的組織提取帳戶，使其成為獨立帳戶。

這些 APIs 可能會增加可以關閉或移除 AWS 帳戶的運算子數量。有權透過 AWS Organizations 管理帳戶中的 AWS Identity and Access Management (IAM) 存取組織的所有使用者都可以呼叫這些 APIs，因此存取權不限於具有任何相關聯多重要素驗證 (MFA) 裝置的帳戶根電子郵件擁有者。 AWS Organizations

此模式會在呼叫 `CloseAccount`和 `RemoveAccountFromOrganization` APIs時實作提醒，讓您可以監控這些活動。對於提醒，它使用 Amazon Simple Notification Service (Amazon SNS) 主題。您也可以透過 Webhook 設定 Slack 通知。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- AWS Organizations 中的組織
- 在組織的根目錄下存取組織管理帳戶，以建立所需的資源

限制

- 如 AWS Organizations API 參考中所述，`CloseAccount`API 僅允許 10% 的作用中成員帳戶在 30 天的滾動期間內關閉。
- 當 AWS 帳戶關閉時，其狀態會變更為 SUSPENDED。在此狀態轉換後 90 天內，AWS Support 可以重新開啟帳戶。90 天後，帳戶會永久刪除。
- 有權存取 AWS Organizations 管理帳戶和 APIs，也可能具有停用這些提醒的許可。如果主要考量是惡意行為，而不是意外刪除，請考慮使用 IAM 許可界限保護此模式建立的資源。
- `CloseAccount` 和 的 API 呼叫`RemoveAccountFromOrganization`會在美國東部 （維吉尼亞北部） 區域 (`us-east-1`) 處理。因此，您必須部署此解決方案 `us-east-1` ，才能觀察事件。

Architecture

目標技術堆疊

- AWS Organizations
- AWS CloudTrail
- Amazon EventBridge
- AWS Lambda
- Amazon SNS

目標架構

下圖顯示此模式的解決方案架構。

1. AWS Organizations 會處理 `CloseAccount`或 `RemoveAccountFromOrganization`請求。
2. Amazon EventBridge 已與 AWS CloudTrail 整合，可將這些事件交付至預設事件匯流排。
3. 自訂 Amazon EventBridge 規則符合 AWS Organizations 請求並呼叫 AWS Lambda 函數。
4. Lambda 函數會傳送訊息至 SNS 主題，使用者可以訂閱該主題以進行電子郵件提醒或進一步處理。
5. 如果啟用 Slack 通知，Lambda 函數會傳送訊息至 Slack Webhook。

工具

AWS 服務

- AWS CloudFormation 透過將基礎設施視為程式碼，提供建立相關 AWS 和第三方資源集合模型、快速一致地佈建資源，以及在整個生命週期中管理資源的方法。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可用來將應用程式與來自各種來源的資料連線。EventBridge 會收到事件、環境變更的指標，並套用規則將事件路由至目標。規則會根據事件的結構、稱為事件模式或排程，將事件比對至目標。
- AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只會在需要時執行程式碼，並自動擴展，從每天幾個請求擴展到每秒數千個請求。您只需為使用的運算時間支付費用。程式碼未執行時無須付費。

- AWS Organizations 可協助您集中管理和控管您的環境。使用 AWS Organizations，您可以透過程式設計方式建立新的 AWS 帳戶並配置資源、分組帳戶來組織工作流程、將政策套用到帳戶或群組以進行控管，以及為所有帳戶使用單一付款方式來簡化計費。
- AWS CloudTrail 會監控和記錄 AWS 基礎設施的帳戶活動，並讓您控制儲存、分析和修復動作。
- Amazon Simple Notification Service (Amazon SNS) 是一種全受管傳訊服務application-to-application(A2A) 和application-to-person(A2P) 通訊。

其他工具

- AWS Lambda Powertools for Python 程式庫是一組公用程式，可為 Lambda 函數提供追蹤、記錄、指標和事件處理功能。

Code

此模式的程式碼位於 GitHub AWS Account Closer Notifier 儲存庫中。

解決方案包含部署此模式架構的 CloudFormation 範本。它使用適用於 Python 的 AWS Lambda Powertools 程式庫來提供記錄和追蹤。

史詩

部署 架構

任務	Description	所需的技能
啟動解決方案堆疊的 CloudFormation 範本。	此模式的 CloudFormation 範本位於 GitHub 儲存庫的主分支中。它會部署 IAM 角色、EventBridge 規則、Lambda 函數和 SNS 主題。 若要啟動範本： 1. 複製 GitHub 儲存庫以取得解決方案程式碼的副本。	AWS 管理員

任務	Description	所需的技能
	2. 開啟 AWS Organizations 管理帳戶的 AWS 管理主控台。 3. 選擇美國東部（維吉尼亞北部）區域 (us-east-1)，然後開啟 CloudFormation 主控台。 4. 使用 account-closure-notifier.yml 範本並指定下列值來建立堆疊： • 堆疊名稱：aws-account-closure-notifier-stack • ResourcePrefix 參數： aws-account-closure-notifier • SlackNotification 參數：如果需要 Slack 通知，請將此設定變更為 true。 • SlackWebhookEndpoint 參數：如果需要 Slack 通知，請指定 Webhook URL。 如需啟動 CloudFormation 堆疊的詳細資訊，請參閱 AWS 文件。	

任務	Description	所需的技能
確認解決方案已成功啟動。	1. 等待 CloudFormation 堆疊達到 CREATE_COMPLETE 狀態。 2. 在 中開啟 EventBridge 主控台us-east-1 。 3. 確認已建立名稱為 的新規則aws-account-closure-notifier-event-rule 。	AWS 管理員
訂閱 SNS 主題。	（選用）如果您想要訂閱 SNS 主題： 1. 在 中開啟 Amazon SNS 主控台us-east-1 ，並尋找名為 的主題aws-account-closure-notifier-sns-topic 。 2. 選擇主題名稱，然後選擇建立訂閱。 3. 對於通訊協定，選擇電子郵件。 4. 對於端點，指定應該接收通知的電子郵件地址，然後選擇建立訂閱。 5. 檢查您的電子郵件收件匣是否有來自 AWS Notifications 的訊息。使用此電子郵件中的連結來確認訂閱。 如需設定 SNS 通知的詳細資訊，請參閱 Amazon SNS 文件。	AWS 管理員

驗證解決方案

任務	Description	所需的技能
將測試事件傳送至預設事件匯流排。	GitHub 儲存庫提供範例事件，您可以傳送到 EventBridge 預設事件匯流排進行測試。EventBridge 規則也會對使用自訂事件來源 的事件做出反應`account.closure.notifier` 。 **Note** 您無法使用 CloudTrail 事件來源傳送此事件，因為無法以 AWS 服務傳送事件。 若要傳送測試事件： 1. 在 中開啟 EventBridge 主控台`us-east-1` 。 2. 在導覽窗格的匯流排下，選擇事件匯流排，然後選取預設事件匯流排。 3. 選擇傳送事件。 4. 針對事件來源，輸入 `account.closure.notifier` 。 5. 針對詳細資訊類型，請輸入 `AWS API Call via CloudTrail` 。 6. 如需事件詳細資訊，`tests/dummy-`	AWS 管理員

任務	Description	所需的技能
	`event.json` 請從 GitHub 儲存庫將 的內容複製並貼到文字方塊中。 7. 選擇傳送以啟動通知工作流程。	
驗證是否已收到電子郵件通知。	檢查訂閱 SNS 主題的信箱以取得通知。您應該會收到一封電子郵件，其中包含已關閉帳戶的詳細資訊，以及執行 API 呼叫的委託人。	AWS 管理員
確認已收到 Slack 通知。	（選用）如果您在部署 CloudFormation 範本時為 `SlackWebhookEndpoint` 參數指定 Webhook URL，請檢查映射至 Webhook 的 Slack 頻道。它應該會顯示一則訊息，其中包含已關閉帳戶的詳細資訊，以及執行 API 呼叫的委託人。	AWS 管理員

相關資源

- CloseAccount 動作 (AWS Organizations API 參考）
- RemoveAccountFromOrganization 動作 (AWS Organizations API 參考）
- 適用於 Python 的 AWS Lambda Powertools

更多模式

- 在 上使用登陸區域加速器自動建立帳戶 AWS
- 自動化刪除 AWS CloudFormation 堆疊和相關聯的資源
- 使用 AWS Service Catalog 和 自動化動態管道管理，以在 Gitflow 環境中部署 Hotfix 解決方案 AWS CodePipeline
- 使用 Amazon DataZone 建置企業資料網格 AWS CDK，以及 AWS CloudFormation
- 使用 Amazon CloudWatch Observability Access Manager 集中監控
- 使用 Account Factory for Terraform 管理多個帳戶的許可集
- 為多帳戶 DevOps 環境實作 Gitflow 分支策略
- 為多帳戶 DevOps 環境實作 GitHub 流程分支策略
- 實作多帳戶 DevOps 環境的主體分支策略
- 使用 Terraform 動態管理 AWS 許可集
- AWS 使用 Terraform 在 上建立階層式多區域 IPAM 架構
- 在多區域、多帳戶組織中設定 CloudFormation 偏離偵測
- 在多帳戶 AWS 環境中設定混合網路的 DNS 解析

安全性、身分與合規

主題

- 自動化事件回應和鑑識
- 自動稽核允許從公有 IP 地址存取 AWS 的安全群組
- 自動修復未加密的 Amazon RDS 資料庫執行個體和叢集
- 使用 CodePipeline、IAM Access Analyzer 和 AWS CloudFormation 巨集自動驗證和部署 IAM 政策和角色
- AWS Security Hub CSPM 與 Jira 軟體雙向整合
- 使用 EC2 Image Builder 和 Terraform 建置強化容器映像的管道
- 使用 Terraform 在 AWS Organizations 中集中管理 IAM 存取金鑰
- 檢查 Amazon CloudFront 分佈是否有存取記錄、HTTPS 和 TLS 版本
- 為企業應用程式選擇 Amazon Cognito 身分驗證流程
- 使用 AWS CloudFormation Guard 政策建立 AWS Config 自訂規則
- 從多個 建立 Prowler 安全性問題清單的合併報告 AWS 帳戶
- 使用 和 AWS CDK CloudFormation 部署和管理 AWS Control Tower 控制項
- 使用 Terraform 部署和管理 AWS Control Tower 控制項
- 使用 Terraform 部署 AWS WAF 解決方案的安全自動化
- 部署可同時偵測多個程式碼交付項目中安全問題的管道
- 使用 部署公有子網路的偵測屬性型存取控制 AWS Config
- 部署公有子網路的預防性屬性型存取控制
- 偵測具有即將到期 CA 憑證的 Amazon RDS 和 Aurora 資料庫執行個體
- 使用 Step Functions 透過 IAM Access Analyzer 動態產生 IAM 政策
- 使用 AWS CloudFormation 範本有條件地啟用 Amazon GuardDuty
- 在 Amazon RDS for SQL Server 中啟用透明資料加密
- 監控和修復 AWS KMS 金鑰的排程刪除
- 使用 Security Hub CSPM 識別 中的公 AWS Organizations 有 Amazon S3 儲存貯體
- 在 Microsoft Sentinel 中擷取和分析 AWS 安全日誌
- 使用 AWS CodePipeline 和 Amazon Bedrock 以程式碼形式管理 AWS Organizations 政策
- 使用 以程式碼形式管理 AWS IAM Identity Center 許可集 AWS CodePipeline

- 使用 AWS Secrets Manager 管理登入資料
- 監控 Amazon ElastiCache 叢集的靜態加密
- 監控 IAM 根使用者活動
- 建立 IAM 使用者時傳送通知
- 使用服務控制政策，防止帳戶層級的網際網路存取
- 使用 PowerShell 匯出 AWS IAM Identity Center 身分及其指派的報告
- 使用 根據 IP 地址或地理位置限制存取 AWS WAF
- 使用 git-secrets 掃描 Git 儲存庫是否有敏感資訊和安全問題
- 使用 Transfer Family、Amazon Cognito 和 GuardDuty 保護檔案傳輸
- 使用 Amazon Macie 保護 CloudWatch Logs 中的敏感資料
- 從 AWS Network Firewall 傳送提醒到 Slack 頻道
- 將自訂屬性傳送至 Amazon Cognito，並將其插入字符
- 使用 AWS Private CA 和 AWS RAM 簡化私有憑證管理
- 使用 Amazon Bedrock 代理程式和 簡化 Amazon EC2 合規管理 AWS Config
- 使用 PowerShell 從 AWS IAM Identity Center 更新 AWS CLI 憑證
- 使用 Network Firewall 從傳出流量的伺服器名稱指示擷取 DNS 網域名稱
- 使用 Terraform 自動為組織啟用 Amazon GuardDuty
- 使用 驗證 PCI DSS 4.0 的操作最佳實務 AWS Config
- 更多模式

自動化事件回應和鑑識

Lucas Kauffman 和 Tomek Jakubowski，Amazon Web Services

總結

此模式會部署一組使用 AWS Lambda 函數提供下列項目的程序：

- 以最低知識啟動事件-回應程序的方法
- 符合AWS 安全事件回應指南的自動化、可重複程序
- 分隔帳戶以操作自動化步驟、存放成品和建立鑑識環境

自動化事件回應和鑑識架構遵循由下列階段組成的標準數位鑑識程序：

1. 遏制
2. 擷取
3. 檢查
4. 分析

您可以對靜態資料（例如，取得的記憶體或磁碟映像）和即時但分離系統上的動態資料執行調查。

如需詳細資訊，請參閱其他資訊一節。

先決條件和限制

先決條件

- 兩個 AWS 帳戶：
  - 安全帳戶，可以是現有帳戶，但最好是新的
  - 鑑識帳戶，最好是新的
- AWS Organizations 設定
- 在 Organizations 成員帳戶中：
  - Amazon Elastic Compute Cloud (Amazon EC2) 角色必須具有 Amazon Simple Storage Service (Amazon S3) 的 Get and List 存取權，且可供 存取 AWS Systems Manager。建議使用 `AmazonSSMManagedInstanceCore` AWS 受管角色。請注意，啟動事件回應時，此角色會自動

連接到 Amazon EC2 執行個體。回應完成後， AWS Identity and Access Management (IAM) 會移除執行個體的所有權利。

  - AWS 成員帳戶中和事件回應和分析 VPCs中的虛擬私有雲端 (VPC) 端點。這些端點包括：S3 Gateway、EC2 Messages、SSM 和 SSM Messages。
- AWS Command Line Interface (AWS CLI) 安裝在 Amazon EC2 執行個體上。如果尚未 AWS CLI 安裝 Amazon EC2 執行個體，則需要網際網路存取，磁碟快照和記憶體擷取才能運作。在這種情況下，指令碼會連線到網際網路以下載 AWS CLI 安裝檔案，並將它們安裝在執行個體上。

限制

- 此架構不打算產生可視為電子證據的成品，在法院中是可允許的。
- 目前，此模式僅支援在 x86 架構上執行的 Linux 型執行個體。

Architecture

目標架構

除了成員帳戶之外，目標環境還包含兩個主要帳戶：安全帳戶和鑑識帳戶。使用兩個帳戶的原因如下：

- 將它們與任何其他客戶帳戶分隔，以便在鑑識分析失敗時減少爆量半徑
- 協助確保隔離和保護所分析成品的完整性
- 將調查保密
- 為了避免威脅行為者可能已 AWS 帳戶 透過達到服務配額立即使用您遭到入侵的所有資源，並防止您執行個體化 Amazon EC2 執行個體來執行調查的情況。

此外，擁有單獨的安全和鑑識帳戶可以建立單獨的角色，即取得證據的回應者和分析證據的調查者。每個角色都可以存取其個別帳戶。

下圖僅顯示帳戶之間的互動。每個帳戶的詳細資訊會顯示在後續圖表中，並連接完整的圖表。

下圖顯示成員帳戶。

1. 事件會傳送至 Slack Amazon Simple Notification Service (Amazon SNS) 主題。

下圖顯示 安全帳戶。

2. 安全帳戶中的 Amazon SNS 主題會啟動鑑識事件。

下圖顯示鑑識帳戶。

安全帳戶是為記憶體和磁碟映像擷取建立兩個主要 AWS Step Functions 工作流程的地方。工作流程執行後，他們會存取在事件中涉及 Amazon EC2 執行個體的成員帳戶，並啟動一組 Lambda 函數來收集記憶體傾印或磁碟傾印。然後，這些成品會存放在鑑識帳戶中。

鑑識帳戶會將 Step Functions 工作流程收集的成品保留在分析成品 Amazon S3 儲存貯體中。鑑識帳戶也會有建置鑑識執行個體 Amazon Machine Image (AMI) 的 Amazon EC2 Image Builder 管道。目前，映像是以 SANS SIFT 工作站為基礎。

建置程序使用維護 VPC，可連線至網際網路。該映像稍後可用於分割 Amazon EC2 執行個體，以分析分析分析 VPC 中收集的成品。

分析 VPC 沒有網際網路連線。根據預設， 模式會建立三個私有分析子網路。您最多可以建立 200 個子網路，這是 VPC 中子網路數量的配額，但 VPC 端點需要為 新增這些子網路， AWS Systems Manager Session Manager 才能自動執行其中的命令。

從最佳實務的角度來看，我們建議您使用 AWS CloudTrail 和 AWS Config 來執行下列動作：

- 追蹤在您的鑑識帳戶中所做的變更
- 監控存放和分析之成品的存取和完整性

工作流程

下圖顯示工作流程的關鍵步驟，其中包含執行個體遭到入侵時的處理程序和決策樹，直到分析和包含為止。

1. `SecurityIncidentStatus`標籤是否已使用值 Analyze 設定？ 如果是，請執行下列動作：
   a. 連接 AWS Systems Manager 和 Amazon S3 的正確 IAM 設定檔。
   b. 將 Amazon SNS 訊息傳送至 Slack 中的 Amazon SNS 佇列。
   c. 將 Amazon SNS 訊息傳送至 `SecurityIncident`佇列。
   d. 叫用記憶體和磁碟擷取狀態機器。
2. 是否已取得記憶體和磁碟？ 如果否，則表示發生錯誤。
3. 使用 標籤`Contain`標記 Amazon EC2 執行個體。

4. 連接 IAM 角色和安全群組，以完全隔離執行個體。

自動化和擴展

此模式的目的是提供可擴展的解決方案，以對單一 AWS Organizations 組織中的多個帳戶執行事件回應和鑑識。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和區域的生命週期中管理資源。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可透過命令列 shell 中的命令 AWS 服務 與 互動。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以保護資料。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Security Hub CSPM 提供 中安全狀態的完整檢視 AWS。它還可協助您根據安全產業標準和最佳實務來檢查 AWS 環境。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務 來建置業務關鍵型應用程式。
- AWS Systems Manager 可協助您管理在 中執行的應用程式和基礎設施 AWS 雲端。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。

Code

如需程式碼和特定實作和用量指引，請參閱 GitHub 自動化事件回應和鑑識架構儲存庫。

史詩

部署 CloudFormation 範本

任務	Description	所需的技能
部署 CloudFormation 範本。	CloudFormation 範本會以指令碼名稱的第一個單字標示 1 到 7，指出範本需要部署在哪個帳戶。請注意，啟動 CloudFormation 範本的順序很重要。 • `1-forensic-AnalysisVPCnS3Buckets.yaml` ：部署在鑑識帳戶中。它會建立 Amazon S3 儲存貯體和分析 VPC，並啟用 CloudTrail。 • `2-forensic-MaintenanceVPCnEC2ImageBuilderPipeline.yaml` ：根據 SANS SIFT 部署維護 VPC 和映像建置器管道。 • `3-security_IR-Disk_Mem_automation.yaml` ：部署安全帳戶中啟用磁碟和記憶體擷取的函數。 • `4-security_LiME_Volatility_Factory.yaml` ：啟動建置函數，以根據指定的 AMI IDs 開始建立記憶體模組。請注意，AMI IDs各不相同 AWS 區域。每當您需要新的記憶體模組時，您可以使用新的 AMI IDs 重新執行此指令碼。請	AWS 管理員

任務	Description	所需的技能
	考慮將此項目與您的黃金映像 AMI 建置器管道整合（如果在您的環境中使用）。 • `5-member-IR-automation.yaml` ：建立成員事件-回應自動化函數，以啟動事件-回應程序。它允許跨帳戶共用 Amazon Elastic Block Store (Amazon EBS) 磁碟區、在事件-回應程序期間自動發佈至 Slack 頻道、啟動鑑識程序，以及在程序完成後隔離執行個體。 • `6-forensic-artifact-s3-policies.yaml` ：部署所有指令碼之後，此指令碼會修正所有跨帳戶互動所需的許可。 • `7-security-IR-vpc.yaml` ：設定用於事件回應磁碟區處理的 VPC。 若要啟動特定 Amazon EC2 執行個體的事件回應架構，請使用 金鑰`SecurityIncidentStatus` 和 值 建立標籤`Analyze`。這將啟動成員 Lambda 函數，自動啟動隔離和記憶體，以及磁碟擷取。	

任務	Description	所需的技能
操作架構。	Lambda 函數也會使用 在資產結束時 （或失敗時） 重新標記資產`Contain`。這會啟動 遏制，這會完全隔離沒有 INBOUND/OUTBOUND 安全群組的執行個體，以及不允許所有存取的 IAM 角色。 請遵循 GitHub 儲存庫中的步驟。	AWS 管理員

部署自訂 Security Hub CSPM 動作

任務	Description	所需的技能
使用 CloudFormation 範本部署自訂 Security Hub CSPM 動作。	若要建立自訂動作，以便您可以使用 Security Hub CSPM 的下拉式清單，請部署 `Modules/SecurityHub Custom Actions/SecurityHubCustomActions.yaml` CloudFormation 範本。然後修改`IRAutomation` 每個成員帳戶中的角色，以允許執行動作以擔任`IRAutomation` 角色的 Lambda 函數。如需更多資訊，請參閱 GitHub 儲存庫。	AWS 管理員

相關資源

- AWS 安全事件回應指南

其他資訊

透過使用此環境，安全營運中心 (SOC) 團隊可以透過下列方式改善其安全事件回應程序：

- 能夠在隔離的環境中執行鑑識，以避免意外危及生產資源
- 採用標準化、可重複的自動化程序來執行遏制和分析。
- 讓任何帳戶擁有者或管理員能夠啟動事件-回應程序，並盡可能了解如何使用標籤
- 擁有標準化、乾淨的環境，可執行事件分析和鑑識，而不會產生較大環境的噪音
- 能夠平行建立多個分析環境
- 將 SOC 資源專注於事件回應，而不是雲端鑑識環境的維護和文件
- 從手動程序轉向自動化程序，以達到可擴展性
- 使用 CloudFormation 範本實現一致性並避免可重複的任務

此外，您可以避免使用持久性基礎設施，並在需要時支付資源費用。

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

自動稽核允許從公有 IP 地址存取 AWS 的安全群組

Eugene Shifer 和 Stephen DiCato，Amazon Web Services

總結

作為安全最佳實務，將 AWS 資源的暴露降至最低至絕對必要的資源至關重要。例如，滿足一般公有需求的 Web 伺服器需要允許從網際網路傳入存取，但對其他工作負載的存取應僅限於特定網路，以減少不必要的暴露。Amazon Virtual Private Cloud (Amazon VPC) 中的安全群組是一項有效的控制，可協助您限制資源存取。不過，評估安全群組可能是一項繁瑣的任務，尤其是在多帳戶架構中。 AWS Config 規則和AWS Security Hub CSPM 控制項可協助您識別允許從公有網際網路 (0.0.0.0/0) 存取特定網路通訊協定的安全群組，例如 Secure Shell (SSH)、HTTP、HTTPS 和 Windows 遠端桌面通訊協定 (RDP)。不過，如果服務在非標準連接埠上執行，或存取僅限於特定公有 IP 地址，則這些規則和控制項不適用。例如，當 Web 服務與 TCP 連接埠 8443 而非標準 TCP 連接埠 443 相關聯時，可能會發生這種情況。當開發人員能夠從其家用網路存取伺服器時，也可能發生這種情況，例如用於測試目的。

若要解決此問題，您可以使用此模式中提供的基礎設施做為程式碼 (IaC) 解決方案，來識別允許從任何非私有 (RFC 1918 不合規） IP 地址存取您 AWS 帳戶 或 AWS 組織中任何工作負載的安全群組。AWS CloudFormation 範本會佈建自訂 AWS Config 規則、 AWS Lambda函數和必要的許可。您可以將它部署為單一帳戶中的堆疊，或透過 管理，將其部署為整個組織的堆疊集 AWS Organizations。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 使用 GitHub 的經驗
- 如果您要部署到單一 AWS 帳戶：
  - 建立 CloudFormation 堆疊的許可
  - 在目標帳戶中AWS Config 設定
  - （選用） 目標帳戶中設定的 Security Hub CSPM
- 如果您要部署到 AWS 組織：
  - 建立 CloudFormation 堆疊集的許可
  - 使用 AWS Organizations 整合設定 Security Hub CSPM
  - 在您部署此解決方案的帳戶中AWS Config 設定

- 將 指定 AWS 帳戶 為 AWS Config 和 Security Hub CSPM 的委派管理員

限制

- 如果您要部署到未啟用 Security Hub CSPM 的個別帳戶，您可以使用 AWS Config 來評估問題清單。
- 如果您要部署到沒有 AWS Config 和 Security Hub CSPM 委派管理員的組織，您必須登入個別成員帳戶才能檢視問題清單。
- 如果您使用 AWS Control Tower 來管理組織中的帳戶，請使用 Customizations for(CfCT) 在此模式中部署 IaC。 AWS Control Tower CfCT 使用 CloudFormation AWS Control Tower 主控台會從護欄建立組態偏離，並要求您重新註冊組織單位 (OUs) 或受管帳戶。
- 有些 AWS 服務 不適用於所有 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額頁面，然後選擇服務的連結。

Architecture

部署至個別 AWS 帳戶

下列架構圖顯示資源在單一 AWS 中的部署 AWS 帳戶。您可以透過 CloudFormation 主控台直接使用 CloudFormation 範本來佈建資源。如果 Security Hub CSPM 已啟用，您可以在 AWS Config 或 Security Hub CSPM 中檢視結果。如果 Security Hub CSPM 未啟用，您只能在 中檢視結果 AWS Config。

該圖顯示以下工作流程：

1. 您可以建立 CloudFormation 堆疊。這會部署 Lambda 函數和 AWS Config 規則。規則和函數都會設定在 AWS Config 和 日誌中發佈資源評估所需的 AWS Identity and Access Management (IAM) 許可。
2. AWS Config 規則會以偵測評估模式運作，並每 24 小時叫用 Lambda 函數。
3. Lambda 函數會評估安全群組並傳送更新至 AWS Config。
4. Security Hub CSPM 會收到所有 AWS Config 調查結果。
5. 視您在帳戶中設定的服務而定 AWS Config，您可以在 Security Hub CSPM 或 中檢視問題清單。

部署至 AWS 組織

下圖顯示透過 AWS Organizations 和 管理的多個帳戶之間的模式部署 AWS Control Tower。您可以透過 CfCT 部署 CloudFormation 範本。評估結果集中在委派管理員帳戶中的 Security Hub CSPM 中。圖表的 AWS CodePipeline 工作流程區段顯示 CfCT 部署期間發生的背景步驟。

該圖顯示以下工作流程：

1. 在管理帳戶中，您將 IaC 範本的壓縮 (ZIP) 檔案上傳至由 CfCT 部署的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。
2. CfCT 管道會解壓縮 檔案、執行 cfn-nag (GitHub) 檢查，並將範本部署為 CloudFormation 堆疊集。
3. 根據您在 CfCT 資訊清單檔案中指定的組態，CloudFormation StackSets 會將堆疊部署到個別帳戶或指定的 OUs。這會在目標帳戶中部署 Lambda 函數和 AWS Config 規則。規則和函數都會設定在 AWS Config 和 日誌中發佈資源評估所需的 IAM 許可。
4. AWS Config 規則會以偵測評估模式運作，並每 24 小時叫用 Lambda 函數。
5. Lambda 函數會評估安全群組並傳送更新至 AWS Config。
6. AWS Config 會將所有調查結果轉送至 Security Hub CSPM。
7. Security Hub CSPM 調查結果會在委派的管理員帳戶中彙總。
8. 您可以在委派管理員帳戶中檢視 Security Hub CSPM 中的彙總調查結果。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。
- AWS Config 提供 中資源的詳細檢視 AWS 帳戶 及其設定方式。它可協助您識別資源彼此之間的關係，以及其組態隨著時間的變化。An AWS Config rule 會定義您的理想資源組態設定，並可 AWS Config 評估您的 AWS 資源是否符合規則中的條件。
- AWS Control Tower 可協助您設定和管理 AWS 多帳戶環境，並遵循規範最佳實務。自訂 AWS Control Tower (CfCT) 可協助您自訂 AWS Control Tower 登陸區域，並保持符合 AWS 最佳實務。此解決方案的自訂是透過 CloudFormation 範本 AWS Organizations 和服務控制政策 (SCPs實作。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。

- AWS Security Hub CSPM 提供 中安全狀態的完整檢視 AWS。它還可協助您根據安全產業標準和最佳實務來檢查 AWS 環境。

其他工具

- Python 是一種一般用途的電腦程式設計語言。

程式碼儲存庫

此模式的程式碼可在 GitHub Detect 易受攻擊的安全群組儲存庫中使用。

最佳實務

我們建議您遵守下列資源中的最佳實務：

- 使用 的組織單位最佳實務 AWS Organizations (AWS 雲端操作和遷移部落格)
- 在 AWS Control Tower 上使用 建立初始基礎的指導 AWS (AWS 解決方案程式庫)
- 建立和修改 AWS Control Tower 資源的指引 (AWS Control Tower 文件)
- CfCT 部署考量 (AWS Control Tower 文件)
- 套用最低權限許可 (IAM 文件)

史詩

檢閱 CloudFormation 範本

任務	Description	所需的技能
決定您的部署策略。	檢閱解決方案和程式碼，以判斷您 AWS 環境的部署策略。判斷您要部署到單一帳戶或 AWS 組織。	應用程式擁有者，一般 AWS
複製儲存庫。	輸入下列命令以複製偵測易受攻擊的安全群組儲存庫： `git clone https://github.com/aws-samp`	應用程式開發人員、應用程式擁有者

任務	Description	所需的技能
	`les/detect-public-security-groups.git`	
驗證 Python 版本。	1. 導覽至複製儲存庫中的最上層目錄： `cd detect-public-security-groups` 2. 開啟 Security-Group-Public-Assessment.yaml。 3. 在 `SgPublicAccessCheckLambdaFunction` 資源中，確認 Python 版本與您的目標相容 AWS 區域。根據預設，此函數會使用 Python 3.12。如需詳細資訊，請參閱 AWS Lambda 新增對 Python 3.12 的支援。如有必要，請更新 Python 版本。 4. 儲存並關閉 Security-Group-Public-Assessment.yaml。	AWS 管理員、應用程式開發人員

部署 CloudFormation 範本

任務	Description	所需的技能
部署 CloudFormation 範本。	將 CloudFormation 範本部署到您的 AWS 環境中。執行以下任意一項：	應用程式開發人員、AWS 管理員、一般 AWS

任務	Description	所需的技能
	• 如果您要部署到單一 AWS 帳戶，請遵循建立堆疊中的指示。 • 如果您要部署到非 管理的組織 AWS Control Tower，請遵循建立堆疊集中的指示。 • 如果您要部署到由 管理的組織 AWS Control Tower，請參閱建置您自己的自訂中的說明。	
驗證部署。	在 CloudFormation 主控台中，確認堆疊或堆疊集已成功部署。	AWS 管理員、應用程式擁有者

檢閱問題清單

任務	Description	所需的技能
檢視 AWS Config 規則調查結果。	在 Security Hub CSPM 中，執行下列動作以檢視個別問題清單： 1. 開啟 Security Hub CSPM 主控台。 2. 在導覽窗格中，選擇調查結果。 3. 在新增篩選條件方塊中，新增下列篩選條件： • 合規狀態為 FAILED • 標題為 SgPublicAccessCheck 4. 選擇套用。	AWS 管理員、AWS 系統管理員、雲端管理員

任務	Description	所需的技能
	在 Security Hub CSPM 中，執行下列動作以檢視依 分組的問題清單總數 AWS 帳戶： 1. 開啟 Security Hub CSPM 主控台。 2. 在導覽窗格中，選擇 Insights。 3. 選擇 Create insight (建立洞見)。 4. 選取洞見的分組屬性： a. 選擇搜尋方塊以顯示篩選條件選項。 b. 選擇 Group by (分組依據)。 c. 選取 AwsAccountId。 d. 選擇套用。 5. 在新增篩選條件方塊中，新增下列篩選條件： • 標題為 `SgPublicAccessCheck` • 合規狀態為 `FAILED` 6. 選擇 Create insight (建立洞見)。 7. 輸入 Insight 名稱，然後選擇建立洞見。 在 中 AWS Config，若要檢視問題清單，請遵循 AWS Config 文件中檢視合規資訊和評估結果中的指示。	

疑難排解

問題	解決方案
CloudFormation 堆疊集建立或刪除失敗。	部署 AWS Control Tower 時，它會強制執行必要的護欄，並取得 AWS Config 對彙總器和規則的控制。這包括防止透過 CloudFormation 進行任何直接變更。若要正確部署或移除此 CloudFormation 範本，包括所有相關資源，您必須使用 CfCT。
CfCT 無法刪除 CloudFormation 範本。	如果 CloudFormation 範本即使在資訊清單檔案中進行必要變更並移除範本檔案後仍存在，請確認資訊清單檔案包含 `enable_stack_set_deletion` 參數，且值設定為 `false`。如需詳細資訊，請參閱 CfCT 文件中的刪除堆疊集。

相關資源

- AWS Config 自訂規則 (AWS Config 文件)

自動修復未加密的 Amazon RDS 資料庫執行個體和叢集

Ajay Rawat 和 Josh Joy，Amazon Web Services

總結

此模式說明如何使用、 AWS Systems Manager 執行手冊和 () 金鑰 AWS Config，在 Amazon Web Services (AWS) 上自動修復未加密的 Amazon Relational Database Service AWS Key Management Service (Amazon RDS AWS KMS) 資料庫執行個體和叢集。

加密的 RDS 資料庫執行個體透過保護您的資料免於未經授權的基礎儲存體存取，提供額外的資料保護層。您可以使用 Amazon RDS 加密來提高部署在 中應用程式的資料保護 AWS 雲端，並滿足靜態加密的合規要求。您可以在建立 RDS 資料庫執行個體時啟用加密，但無法在建立之後啟用加密。不過，您可以透過建立資料庫執行個體的快照，然後建立該快照的加密複本，將加密新增至未加密的 RDS 資料庫執行個體。然後，您可以從加密快照還原資料庫執行個體，以取得原始資料庫執行個體的加密副本。

此模式使用 AWS Config 規則 來評估 RDS 資料庫執行個體和叢集。它使用 AWS Systems Manager Runbook 來套用修復，此 Runbook 定義要在不合規 Amazon RDS 資源上執行的動作，以及加密資料庫快照的 AWS KMS 金鑰。然後，它會強制執行服務控制政策 (SCPs)，以防止在未加密的情況下建立新的資料庫執行個體和叢集。

此模式的程式碼在 GitHub 中提供。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 此模式從 GitHub 原始程式碼儲存庫下載到您電腦的檔案
- 未加密的 RDS 資料庫執行個體或叢集
- 用於加密 RDS 資料庫執行個體和叢集的現有 AWS KMS 金鑰
- 更新 KMS 金鑰資源政策的存取權
- AWS Config 在您的 中啟用 AWS 帳戶 （ 請參閱 AWS 文件中的 入門 AWS Config)

限制

- 您只能在建立 RDS 資料庫執行個體時啟用加密，而不是在建立執行個體之後啟用加密。

- 未加密資料庫執行個體不可以有加密僅供讀取複本，加密資料庫執行個體也不可以有未加密僅供讀取複本。
- 您無法將未加密的備份或快照還原至已加密的資料庫執行個體。
- 大多數資料庫執行個體類別可以使用 Amazon RDS 加密。如需例外狀況清單，請參閱 Amazon RDS 文件中的加密 Amazon RDS 資源。
- 若要將加密快照從一個複製到 AWS 區域 另一個快照，您必須在目的地中指定 KMS 金鑰 AWS 區域。這是因為 KMS 金鑰專屬 AWS 區域 於它們建立所在的 。
- 在整個複製過程中來源快照仍會保持加密狀態。Amazon RDS 使用信封加密來保護複製程序期間的資料。如需詳細資訊，請參閱 AWS KMS 文件中的信封加密。
- 您無法取消加密加密的資料庫執行個體。不過，您可以從加密的資料庫執行個體匯出資料，並將資料匯入未加密的資料庫執行個體。
- 只有當您確定不再需要使用 KMS 金鑰時，才應該刪除該金鑰。如果您不確定，請考慮停用 KMS 金鑰，而不是將其刪除。如果您稍後需要再次使用已停用的 KMS 金鑰，但無法復原已刪除的 KMS 金鑰。
- 如果您不選擇保留自動備份，則會刪除與資料庫執行個體位於相同 AWS 區域 中的自動備份。刪除資料庫執行個體後，便無法復原自動備份內容。
- 您的自動備份會保留在您刪除資料庫執行個體時所設定的保留期間。無論您是否選擇建立最終資料庫快照，都會依照此一設定保留期間。
- 如果啟用自動修復，此解決方案會加密具有相同 KMS 金鑰的所有資料庫。

Architecture

下圖說明 CloudFormation 實作的架構。請注意，您也可以使用 實作此模式 AWS Cloud Development Kit (AWS CDK)。

工具

工具

- CloudFormation 可協助您自動設定 AWS 資源。它可讓您使用範本檔案來建立和刪除資源集合，做為單一單位（堆疊）。
- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可讓您在程式碼中定義雲端基礎設施，並使用熟悉的程式設計語言進行佈建。

AWS 服務和功能

- AWS Config 會追蹤 AWS 資源的組態及其與其他 資源的關係。它也可以評估這些 AWS 資源的合規性。此服務使用可設定為根據所需組態評估 AWS 資源的規則。您可以針對常見的合規案例使用一組 AWS Config 受管規則，也可以針對自訂案例建立自己的規則。當發現 AWS 資源不合規時，您可以透過 AWS Systems Manager Runbook 指定修補動作，並選擇性地透過 Amazon Simple Notification Service (Amazon SNS) 主題傳送提醒。換句話說，您可以將修復動作與 建立關聯， AWS Config 規則 並選擇自動執行這些動作來解決不合規的資源，而無需手動介入。如果資源在自動修復後仍然不合規，您可以將規則設定為再次嘗試自動修復。
- Amazon Relational Database Service (Amazon RDS) 可讓您更輕鬆地在雲端中設定、操作和擴展關聯式資料庫。Amazon RDS 的基本建置區塊是 資料庫執行個體，這是 中的隔離資料庫環境 AWS 雲端。Amazon RDS 提供一系列執行個體類型，這些執行個體類型已針對不同的關聯式資料庫使用案例進行最佳化。執行個體類型包含 CPU、記憶體、儲存和聯網容量的各種組合，可讓您靈活地為資料庫選擇適當的資源組合。每個執行個體類型都包含數個執行個體大小，可讓您根據目標工作負載的需求擴展資料庫。
- AWS Key Management Service (AWS KMS) 是一種受管服務，可讓您輕鬆地建立和控制 AWS KMS keys，以加密您的資料。KMS 金鑰是根金鑰的邏輯表示法。KMS 金鑰包含金鑰 ID、建立日期、說明和金鑰狀態等中繼資料。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- 服務控制政策 SCPs) 可讓您集中控制組織中所有帳戶的可用許可上限。SCPs可協助您確保帳戶符合組織的存取控制準則。SCP 不會影響管理帳戶中的使用者或角色。它們只會影響組織中的成員帳戶。在沒有將政策對帳戶的影響進行徹底測試之前，我們強烈建議您不要將 SCP 連接到組織的根帳戶。反之，請建立組織單位 (OU)，您可以將帳戶一次移至一個，或至少以小數字移動，以確保您不會不小心將使用者鎖定在金鑰服務之外。

Code

此模式的原始程式碼和範本可在 GitHub 儲存庫中使用。模式提供兩種實作選項：您可以部署 CloudFormation 範本來建立修補角色，以加密 RDS 資料庫執行個體和叢集，或使用 AWS CDK。儲存庫具有這兩個選項的個別資料夾。

Epicssection 提供部署 CloudFormation 範本的step-by-step說明。如果您想要使用 AWS CDK，請遵循 GitHub 儲存庫中 `README.md` 檔案的指示。

最佳實務

- 啟用靜態和傳輸中的資料加密。
- 在所有 帳戶和 AWS Config 中啟用 AWS 區域。
- 記錄所有資源類型的組態變更。
- 定期輪替您的 IAM 登入資料。
- 利用標記 AWS Config，讓 更容易管理、搜尋和篩選資源。

史詩

建立 IAM 修復角色和 Systems Manager Runbook

任務	Description	所需的技能
下載 CloudFormation 範本。	從 GitHub 儲存庫下載 `unencrypted-to-encrypted-rds.template.json` 檔案。	DevOps 工程師
建立 CloudFormation 堆疊。	1. 登入 AWS 管理主控台 並開啟 CloudFormation 主控台。 2. 啟動 `unencrypted-to-encrypted-rds.template.json` 範本以建立新的堆疊。 如需部署範本的詳細資訊，請參閱 CloudFormation 文件。	DevOps 工程師
檢閱 CloudFormation 參數和值。	1. 根據您的環境需求檢閱堆疊詳細資訊和更新值。 2. 選擇建立堆疊以部署範本。	DevOps 工程師
檢閱資源。	堆疊建立後，其狀態會變更為 CREATE_COMPLETE。	DevOps 工程師

任務	Description	所需的技能
	在 CloudFormation 主控台中檢閱建立的資源 (IAM 角色、Systems Manager Runbook)。	

更新 AWS KMS 金鑰政策

任務	Description	所需的技能
更新您的 KMS 金鑰政策。	1. 請確定金鑰別名`alias/RDSEncryptionAtRestKMSAlias` 存在。 2. 金鑰政策陳述式應包含 IAM 修補角色。（檢查您在上一個史詩中部署的 CloudFormation 範本所建立的資源。） 3. 在下列金鑰政策中，更新粗體部分，以符合您的帳戶和已建立的 IAM 角色。 `{ "Sid": "Allow access through RDS for all principals in the account that are authorized to use RDS", "Effect": "Allow", "Principal": { "AWS": "arn:aws:iam:: <your-AWS-account-ID>":role/<your-IAM-remediation-role>" }, "Action": [`	DevOps 工程師

任務	Description	所需的技能
	"kms:Encrypt", "kms:Decrypt", "kms:ReEncrypt*", "kms:GenerateDataKey*", "kms:CreateGrant", "kms:ListGrants", "kms:DescribeKey"], "Resource": "*", "Condition": { "StringEquals": { "kms:ViaService": "rds.us-east-1.amazonaws.com", "kms:CallerAccount": "<your-AWS-account-ID>" } } }	

尋找並修復不合規的資源

任務	Description	所需的技能
檢視不合規的資源。	1. 若要檢視不合規資源的清單，請開啟 AWS Config 主控台。 2. 在導覽窗格中，選擇規則，然後選擇`rds-storage-encrypted` 規則。	DevOps 工程師

任務	Description	所需的技能
	AWS Config 主控台中列出的不合規資源將是執行個體，而不是叢集。修復自動化會加密執行個體和叢集，並建立新加密的執行個體或新建立的叢集。不過，請勿同時修復屬於相同叢集的多個執行個體。 在您修復任何 RDS 資料庫執行個體或磁碟區之前，請確定未使用 RDS 資料庫執行個體。確認在建立快照時沒有發生寫入操作，以確保快照包含原始資料。考慮強制執行維護時段，在此期間將執行修復。	
修復不合規的資源。	1. 當您準備好且維護時段生效時，請選擇要修復的資源，然後選擇修復。 動作狀態欄現在應會顯示已排入佇列的動作執行。 2. 在 Systems Manager 中檢視修復的進度和狀態。開啟 Systems Manager 主控台。在導覽窗格中，選擇自動化，然後選取對應自動化的執行 ID 以檢視更多詳細資訊。	DevOps 工程師

任務	Description	所需的技能
確認 RDS 資料庫執行個體可用。	自動化完成後，新加密的 RDS 資料庫執行個體將變為可用。加密的 RDS 資料庫執行個體會有字首，`encrypted` 後面接著原始名稱。例如，如果未加密的 RDS 資料庫執行個體名稱為 `database-1` ，則新加密的 RDS 資料庫執行個體將為 `encrypted-database-1` 。	DevOps 工程師
終止未加密的執行個體。	修復完成且新加密的資源已經過驗證後，您就可以終止未加密的執行個體。請務必先確認新加密的資源符合未加密的資源，再終止任何資源。	DevOps 工程師

強制執行 SCPs

任務	Description	所需的技能
強制執行 SCPs。	強制執行 SCPs，以防止未來在沒有加密的情況下建立資料庫執行個體和叢集。為此，請使用 GitHub 儲存庫中提供`rds_encrypted.json` 的檔案，並遵循AWS 文件中的指示。	安全工程師

相關資源

參考

- 設定 AWS Config

- AWS Config 自訂規則
- AWS KMS 概念
- AWS Systems Manager 文件
- 服務控制政策

工具

- CloudFormation
- AWS Cloud Development Kit (AWS CDK)

指南和模式

- 在 AWS CloudTrail 中使用自訂修補規則自動重新啟用 AWS Config

其他資訊

如何 AWS Config 運作？

當您使用 時 AWS Config，它會先探索 帳戶中存在的支援 AWS 資源，並為每個資源產生組態項目。AWS Config 也會在資源組態變更時產生組態項目，並從您啟動組態記錄器時開始維護資源組態項目的歷史記錄。根據預設， 會為 中每個支援的資源 AWS Config 建立組態項目 AWS 區域。如果您不想為所有支援的資源 AWS Config 建立組態項目，您可以指定要追蹤的資源類型。

AWS Config 與 AWS Config 規則 有何關聯 AWS Security Hub CSPM？

AWS Security Hub CSPM 是一種安全與合規服務，可提供安全與合規狀態管理即服務。它使用 AWS Config 和 AWS Config 規則 作為其主要機制來評估 AWS 資源的組態。 AWS Config 規則 也可用於直接評估資源組態。其他 AWS 服務，例如 AWS Control Tower 和 AWS Firewall Manager也會使用 AWS Config 規則。

使用 CodePipeline、IAM Access Analyzer 和 AWS CloudFormation 巨集自動驗證和部署 IAM 政策和角色

Helton Ribeiro 和 Guilherme Simoes，Amazon Web Services

摘要

此模式說明步驟並提供程式碼來建立部署管道，讓您的開發團隊在您的 Amazon Web Services (AWS) 帳戶中建立 AWS Identity and Access Management (IAM) 政策和角色。此方法可協助您的組織降低營運團隊的開銷，並加速部署程序。它還可協助您的開發人員建立與您現有控管和安全控制相容的 IAM 角色和政策。

此模式的方法使用 AWS Identity and Access Management Access Analyzer 來驗證您要連接到 IAM 角色的 IAM 政策，並使用 AWS CloudFormation 部署 IAM 角色。不過，您的開發團隊不會直接編輯 AWS CloudFormation 範本檔案，而是建立 JSON 格式的 IAM 政策和角色。AWS CloudFormation 巨集會在開始部署之前，將這些 JSON 格式的政策檔案轉換為 AWS CloudFormation IAM 資源類型。

部署管道 (`RolesPipeline`) 具有來源、驗證和部署階段。在來源階段，您的開發團隊會將包含 IAM 角色和政策定義的 JSON 檔案推送至 AWS CodeCommit 儲存庫。AWS CodeBuild 接著會執行指令碼來驗證這些檔案，並將其複製到 Amazon Simple Storage Service (Amazon S3) 儲存貯體。由於您的開發團隊無法直接存取存放在個別 S3 儲存貯體中的 AWS CloudFormation 範本檔案，因此他們必須遵循 JSON 檔案建立和驗證程序。

最後，在部署階段，AWS CodeDeploy 會使用 AWS CloudFormation 堆疊來更新或刪除帳戶中的 IAM 政策和角色。

> **Important**
>
> 此模式的工作流程是一種概念驗證 (POC)，我們建議您只在測試環境中使用它。如果您想要在生產環境中使用此模式的方法，請參閱 IAM 文件中的 IAM 安全最佳實務，並對 IAM 角色和 AWS 服務進行必要的變更。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶

- RolesPipeline 管道的新或現有 S3 儲存貯體。請確定您使用的存取憑證具有將物件上傳至此儲存貯體的許可。
- 安裝並設定 AWS Command Line Interface (AWS CLI)。如需詳細資訊，請參閱 AWS CLI 文件中的安裝、更新和解除安裝 AWS CLI。
- 安裝並設定 AWS Serverless Application Model (AWS SAM) CLI。如需詳細資訊，請參閱 AWS SAM 文件中的安裝 AWS SAM CLI。
- Python 3，安裝在本機電腦上。如需詳細資訊，請參閱 Python 文件。
- 安裝和設定的 Git 用戶端。
- 複製到本機電腦的 GitHub IAM roles pipeline儲存庫。
- 現有的 JSON 格式 IAM 政策和角色。如需詳細資訊，請參閱 Github IAM roles pipeline儲存庫中的 ReadMe 檔案。
- 您的開發人員團隊不得具有編輯此解決方案 AWS CodePipeline、CodeBuild 和 CodeDeploy 資源的許可。

限制

- 此模式的工作流程是一種概念驗證 (POC)，我們建議您只在測試環境中使用它。如果您想要在生產環境中使用此模式的方法，請參閱 IAM 文件中的 IAM 安全最佳實務，並對 IAM 角色和 AWS 服務進行必要的變更。

Architecture

下圖說明如何使用 CodePipeline、IAM Access Analyzer 和 AWS CloudFormation 巨集，自動驗證 IAM 角色和政策並將其部署到帳戶。

該圖顯示以下工作流程：

1. 開發人員會撰寫 JSON 檔案，其中包含 IAM 政策和角色的定義。開發人員將程式碼推送到 CodeCommit 儲存庫，然後 CodePipeline 啟動RolesPipeline管道。
2. CodeBuild 使用 IAM Access Analyzer 驗證 JSON 檔案。如果有任何安全或錯誤相關的問題清單，部署程序會停止。
3. 如果沒有安全或錯誤相關的調查結果，則 JSON 檔案會傳送至 RolesBucket S3 儲存貯體。
4. 實作為 AWS Lambda 函數的 AWS CloudFormation 巨集會從儲存RolesBucket貯體讀取 JSON 檔案，並將其轉換為 AWS CloudFormation IAM 資源類型。

5. 預先定義的 AWS CloudFormation 堆疊會安裝、更新或刪除帳戶中的 IAM 政策和角色。

自動化和擴展

自動部署此模式的 AWS CloudFormation 範本會在 GitHub IAM 角色管道儲存庫中提供。

工具

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- IAM Access Analyzer 可協助您識別組織和帳戶中與外部實體共用的資源，例如 S3 儲存貯體或 IAM 角色。這可協助您識別對資源和資料的意外存取。
- AWS Serverless Application Model (AWS SAM) 是一種開放原始碼架構，可協助您在 AWS 雲端中建置無伺服器應用程式。

Code

此模式的原始程式碼和範本可在 GitHub IAM 角色管道儲存庫中使用。

史詩

複製儲存庫

任務	Description	所需的技能
複製範例儲存庫。	將 GitHub IAM 角色管道儲存庫複製到本機電腦。	應用程式開發人員，一般 AWS

部署 RolesPipeline 管道

任務	Description	所需的技能
部署管道。	1. 導覽至包含複製儲存庫的目錄。	應用程式開發人員，一般 AWS

任務	Description	所需的技能
	2. **Important** 執行 `make deploy bucket=<bucket_name>` 命令。：您必須<bucket_name> 將 取代為現有 S3 儲存貯體的儲存貯體名稱。 3. 執行 `aws codepipeline get-pipeline –name RolesPipeline` 命令來檢查您的部署是否成功。	
複製管道的儲存庫。	1. `RolesPipeline` AWS CloudFormation 堆疊會建立 `roles-pipeline-repo` CodeCommit 儲存庫。 2. 登入 AWS 管理主控台，開啟 AWS CodeCommit 主控台，然後複製 CodeCommit 儲存庫的 URL 以將其複製到本機電腦。如需詳細資訊，請參閱 AWS CodeCommit 文件中的連線至 AWS CodeCommit 儲存庫。 AWS CodeCommit	應用程式開發人員，一般 AWS

測試 RolesPipeline 管道

任務	Description	所需的技能
使用有效的 IAM 政策和角色測試 RolesPipeline 管道。	1. 為您的 IAM 政策和角色建立 JSON 檔案。您可以從 GitHub `IAM roles pipeline`儲存庫使用 `role-example` 目錄中的範例。 2. **Important** 使用所需的組態定義 IAM 政策和角色。：請確定您遵循 GitHub `IAM roles pipeline`儲存庫`ReadMe`檔案中描述的格式。 3. 將修改推送到 `roles-pipeline-repo` CodeCommit 儲存庫。 4. 驗證`RolesPipeline` 管道的實作。 5. 確定 IAM 政策和角色已正確部署在帳戶中。 6. 驗證是否有與 IAM 政策或角色相關聯的許可界限。如需詳細資訊，請參閱 IAM 文件中的 IAM 實體的許可界限。	應用程式開發人員，一般 AWS
使用無效的 IAM 政策和角色測試 RolesPipeline 管道。	1. 修改 `roles-pipeline-repo` CodeCommit 儲存庫，並包含無效的 IAM 角色或政策。例如，您可以使用	應用程式開發人員，一般 AWS

任務	Description	所需的技能
	不存在的動作或無效的 IAM 政策版本。 2. 驗證管道實作。如果 IAM Access Analyzer 在驗證階段偵測到無效的 IAM 政策或角色，則會停止管道。	

清除您的資源

任務	Description	所需的技能
準備清除。	清空 S3 儲存貯體，然後執行 `destroy`命令。	應用程式開發人員，一般 AWS
刪除 RolesStack 堆疊。	1. `RolesPipeline` 管道會建立部署 IAM 政策和角色的 `RolesStack` AWS CloudFormation 堆疊。您必須先刪除此堆疊，才能刪除`RolesPipeline` 管道。 2. 登入 AWS 管理主控台，開啟 AWS CloudFormation 主控台，然後選擇`RolesStack` 堆疊，然後選擇刪除。	應用程式開發人員，一般 AWS
刪除 RolesPipeline 堆疊。	若要刪除 `RolesPipeline` AWS CloudFormation 堆疊，請遵循 Github `IAM roles pipeline`儲存庫中 ReadMe 檔案的指示。	應用程式開發人員，一般 AWS

相關資源

- IAM Access Analyzer - 政策驗證 (AWS 新聞部落格)

- 使用 AWS CloudFormation 巨集對範本執行自訂處理 (AWS CloudFormation 文件)
- 使用 Python 建置 Lambda 函數 (AWS Lambda 文件)

AWS Security Hub CSPM 與 Jira 軟體雙向整合

Joaquin Rinaudo，Amazon Web Services

總結

此解決方案支援 AWS Security Hub CSPM 和 Jira 之間的雙向整合。使用此解決方案，您可以自動和手動從 Security Hub CSPM 調查結果建立和更新 Jira 票證。安全團隊可以使用此整合，向開發人員團隊通知需要採取動作的嚴重安全問題清單。

解決方案可讓您：

- 選取在 Jira 中自動建立或更新票證的 Security Hub CSPM 控制項。
- 在 Security Hub CSPM 主控台中，使用 Security Hub CSPM 自訂動作在 Jira 中手動呈報票證。
- 根據 中定義的 AWS 帳戶 標籤，在 Jira 中自動指派票證 AWS Organizations。如果未定義此標籤，則會使用預設指派者。
- 在 Jira 中自動隱藏標記為誤報或接受風險的 Security Hub CSPM 調查結果。
- 在 Security Hub CSPM 中封存 Jira 票證的相關調查結果時自動關閉 Jira 票證。
- 當 Security Hub CSPM 問題清單再次發生時，重新開啟 Jira 票證。

Jira 工作流程

解決方案使用自訂 Jira 工作流程，可讓開發人員管理和記錄風險。當問題通過工作流程時，雙向整合可確保 Jira 票證和 Security Hub CSPM 調查結果的狀態在兩個服務中的工作流程之間同步。此工作流程是 Dinis Cruz 授予 Apache License 2.0 版授權的 SecDevOps 風險工作流程的衍生產品。 https://www.apache.org/licenses/LICENSE-2.0我們建議新增 Jira 工作流程條件，以便只有安全團隊的成員可以變更票證狀態。

如需此解決方案自動產生的 Jira 票證範例，請參閱此模式的其他資訊區段。

先決條件和限制

先決條件

- 如果您想要在多帳戶 AWS 環境中部署此解決方案：
  - 您的多帳戶環境為作用中並由 管理 AWS Organizations。
  - 您的 上已啟用 Security Hub CSPM AWS 帳戶。

  - 在 中 AWS Organizations，您已指定 Security Hub CSPM 管理員帳戶。
  - 您的跨帳戶 AWS Identity and Access Management (IAM) 角色具有 AWS Organizations 管理帳戶的`AWSOrganizationsReadOnlyAccess`許可。
  - （選用）您已 AWS 帳戶 使用 標記您的 `SecurityContactID`。此標籤用於將 Jira 票證指派給定義的安全聯絡人。
- 如果您想要在單一 中部署此解決方案 AWS 帳戶：
  - 您有作用中的 AWS 帳戶。
  - 您的 上已啟用 Security Hub CSPM AWS 帳戶。
- Jira 資料中心執行個體

> **Important**
>
> 此解決方案支援使用 Jira Cloud。不過，Jira Cloud 不支援匯入 XML 工作流程，因此您需要在 Jira 中手動重新建立工作流程。您可以在 GitHub 儲存庫中找到轉換和狀態。

- Jira 中的管理員許可
- 下列其中一個 Jira 字符：
  - 若為 Jira Enterprise，則為個人存取字符 (PAT)。如需詳細資訊，請參閱使用個人存取字符 (Atlassian 支援）。
  - 若為 Jira Cloud，則為 Jira API 字符。如需詳細資訊，請參閱管理 API 字符 (Atlassian 支援）。

Architecture

本節說明各種情況下解決方案的架構，例如開發人員和安全工程師決定接受風險或決定修正問題時。

案例 1：開發人員解決問題

1. Security Hub CSPM 會根據指定的安全控制產生問題清單，例如AWS 基礎安全最佳實務標準中的問題清單。
2. 與調查結果相關聯的 Amazon CloudWatch 事件，而 `CreateJIRA`動作會啟動 AWS Lambda 函數。
3. Lambda 函數會使用其組態檔案和調查結果`GeneratorId`的 欄位來評估是否應呈報調查結果。
4. Lambda 函數會判斷應該呈報的問題清單，並在 AWS Organizations AWS 管理`SecurityContactID`帳戶中從 取得帳戶標籤。此 ID 與開發人員相關聯，並用作 Jira 票證的受指派者 ID。

5. Lambda 函數會使用存放在 中的登入資料 AWS Secrets Manager ，在 Jira 中建立票證。Jira 通知開發人員。
6. 開發人員處理基礎安全調查結果，並在 Jira 中將票證的狀態變更為 TEST FIX。
7. Security Hub CSPM 會將問題清單更新為 ARCHIVED，並產生新的事件。此事件會導致 Lambda 函數自動關閉 Jira 票證。

案例 2：開發人員決定接受風險

1. Security Hub CSPM 會根據指定的安全控制產生問題清單，例如AWS 基礎安全最佳實務標準中的問題清單。
2. 與調查結果相關聯的 CloudWatch 事件和動作會CreateJIRA啟動 Lambda 函數。
3. Lambda 函數會使用其組態檔案和調查結果GeneratorId的 欄位來評估是否應呈報調查結果。
4. Lambda 函數會判斷應該呈報的問題清單，並在 AWS Organizations AWS 管理SecurityContactID帳戶中從 取得帳戶標籤。此 ID 與開發人員相關聯，並用作 Jira 票證的受指派者 ID。
5. Lambda 函數會使用存放在 Secrets Manager 中的登入資料，在 Jira 中建立票證。Jira 通知開發人員。
6. 開發人員決定接受風險，並在 Jira 中將票證的狀態變更為 AWAITING RISK ACCEPTANCE。
7. 安全工程師會檢閱請求，並找到適當的業務理由。安全工程師將 Jira 票證的狀態變更為 ACCEPTED RISK。這會關閉 Jira 票證。
8. CloudWatch 每日事件會啟動重新整理 Lambda 函數，以識別已關閉的 Jira 票證，並將其相關的 Security Hub CSPM 調查結果更新為 SUPPRESSED。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和區域的生命週期進行管理。
- Amazon CloudWatch Events 可協助您監控 AWS 資源的系統事件，方法是使用規則來比對事件，並將其路由至函數或串流。

- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。
- AWS Secrets Manager 可協助您將程式碼中的硬式編碼憑證 (包括密碼) 取代為 Secrets Manager 的 API 呼叫，以便透過程式設計方法來擷取機密。
- AWS Security Hub CSPM 提供 中安全狀態的完整檢視 AWS。它還可協助您根據安全產業標準和最佳實務來檢查 AWS 環境。

程式碼儲存庫

此模式的程式碼可在 aws-securityhub-jira-software-integration 儲存庫的 GitHub 上取得。它包含此解決方案的範例程式碼和 Jira 工作流程。

史詩

設定 Jira

任務	Description	所需的技能
匯入工作流程。	身為 Jira 中的管理員，請將 `issue-workflow.xml` 檔案匯入您的 Jira 資料中心執行個體。如果您使用 Jira Cloud，則需要根據 `assets/jira-cloud-transitions.png` 和 `assets/jira-cloud-status.png` 檔案建立工作流程。 您可以在 GitHub 的 aws-securityhub-jira-software-integration 儲存庫中找到檔案。如需說明，請參閱使用 XML 建立工作流程 (Jira 文件) 。	Jira 管理員

任務	Description	所需的技能
啟用並指派工作流程。	工作流程會處於非作用中狀態，直到您將其指派給工作流程結構描述為止。然後，您將工作流程結構描述指派給專案。 1. 針對您的專案，請確定您已識別專案的問題類型方案。您可以建立新的問題類型，或從現有的問題類型中選取，例如 Bug。 2. 根據啟用工作流程 (Jira 文件） 中的指示，將匯入的工作流程指派給工作流程結構描述。 3. 根據將工作流程結構描述與專案建立關聯中的指示，將工作流程結構描述指派給專案 (Jira 文件）。	Jira 管理員

設定解決方案參數

任務	Description	所需的技能
設定解決方案參數。	1. 在 conf 資料夾中，開啟 `params_prod.shfile` 。 2. 提供下列參數的值： • `ORG_ACCOUNT_ID` – AWS Organizations 管理帳戶的帳戶 ID。解決方案會讀取帳戶標籤，並將票證指派給這些 AWS 帳戶	AWS 系統管理員

任務	Description	所需的技能
	標籤中定義的特定安全聯絡人。 • ORG_ROLE – 用來存取 AWS Organizations 管理帳戶的 IAM 角色名稱。此角色必須具有 OrganizationsReadOnlyAccess 許可。 • EXTERNAL_ID – 如果您使用外部 ID 擔任 中定義的 IAM 角色，則為選用參數ORG_ROLE。如需詳細資訊，請參閱如何使用外部 ID (IAM 文件）。 • JIRA_DEFAULT_ASSIGNEE – 這是所有安全問題的 Jira 預設受指派者。如果帳戶未正確標記或無法擔任角色，則會使用此預設指派者。 • JIRA_INSTANCE – Jira 端點的 HTTPS 地址，格式如下： team-<team-id>.atlassian.net/ • JIRA_PROJECT_KEY – 用來建立票證的 Jira 專案金鑰名稱，例如 SEC或 TEST。此專案必須已存在於 Jira 中。 • ISSUE_TYPE – 指派給 Jira 中專案的問題類型結	

任務	Description	所需的技能
	構描述名稱，例如 Bug或 `Security Issue`。 • `REGIONS` – 您要部署此解決方案的 AWS 區域程式碼清單，例如 `eu-west-1` 。 3. 儲存並關閉解決方案參數檔案。	
識別您要自動化的問題清單。	1. 開啟 Security Hub CSPM 主控台。 2. 在 Security Hub CSPM 導覽窗格中，選擇問題清單。 3. 選擇問題清單標題。 4. 選擇問題清單 ID。這會顯示問題清單的完整 JSON。 5. 在 JSON 中，複製 `GeneratorId` 欄位中的字串。此值為AWS 安全性調查結果格式 (ASFF)。例如，`aws-foundational-security-best-practices/v/1.0.0/S3.1` 對應至安全控制 S3.1 S3 封鎖公開存取設定的問題清單應啟用。 6. 重複這些步驟，直到您複製要自動化的任何問題清單的所有`GeneratorID` 值為止。	

任務	Description	所需的技能
將問題清單新增至組態檔案。	1. 在 src/code 中，開啟 `config.jsonconfig` 檔案。 2. 將您先前案例擷取`GeneratorID` 的值貼到 `default` 參數，並使用逗號分隔每個 ID。 3. 儲存並關閉 組態檔案。 下列程式碼範例顯示自動化 `aws-foundational-security-best-practices/v/1.0.0/SNS.1` 和 `aws-foundational-security-best-practices/v/1.0.0/S3.1` 問題清單。 `{ "Controls" : { "eu-west-1": [ "arn:aws:securityhub:::ruleset/cis-aws-foundations-benchmark/v/1.2.0/rule/1.22" ], "default": [ aws-foundational-security-best-practices/v/1.0.0/SNS.1, aws-foundational-security-best-practices/v/1.0.0/S3.1 ] }`	AWS 系統管理員

任務	Description	所需的技能
	} **Note** 您可以選擇為每個問題清單自動化不同的問題清單 AWS 區域。協助防止重複問題清單的最佳實務是選取單一區域，以自動建立與 IAM 相關的控制項。	

部署整合

任務	Description	所需的技能
部署整合。	在命令列終端機中，輸入下列命令： `./deploy.sh prod`	AWS 系統管理員
將 Jira 登入資料上傳至 Secrets Manager。	1. 開啟 Secrets Manager 主控台。 2. 在 Secrets (秘密)，請選擇 Store a new secret (存放新秘密)。 3. 針對機密類型，選擇其他類型的機密。 4. 如果您使用的是 Jira Enterprise，對於金鑰/值對，請執行下列動作： • 在第一列`auth`中，在索引鍵方塊中輸入 ，然後在值	AWS 系統管理員

任務	Description	所需的技能
	方塊中輸入 `token_auth` 。 • 新增第二列，`token`在金鑰方塊中輸入 ，然後在值方塊中輸入您的個人存取字符。 如果您使用的是 Jira Cloud，對於金鑰/值對，請執行下列動作： • 在第一列`auth`中，在索引鍵方塊中輸入 ，然後在值方塊中輸入 `basic_auth` 。 • 新增第二列，`token`在金鑰方塊中輸入 ，然後在值方塊中輸入您的 API 字符。 • 新增第三列，`email`在金鑰方塊中輸入 ，然後在值方塊中輸入您的電子郵件地址。 5. 選擇下一步。 6. 對於秘密名稱，輸入 `Jira-Token` ，然後在頁面底部選擇下一步。 7. 在 Secret rotation (秘密輪換) 中，保留 Disable automatic rotation (停用自動輪換)，然後在頁面底部選擇 Next (下一步)。	

任務	Description	所需的技能
	8. 在 Review (檢閱) 頁面上，檢閱秘密詳細資訊，然後選擇 Store (存放)。	
建立 Security Hub CSPM 自訂動作。	1. 對於每個 AWS 區域，在 AWS Command Line Interface (AWS CLI) 中使用 create-action-target 命令來建立名為 的 Security Hub CSPM 自訂動作`CreateJiraIssue` 。 `aws securityhub create-action-target --name "CreateJiraIssue" \ --description "Create ticket in JIRA" \ --id "CreateJiraIssue" --region $<aws-region>` 2. 開啟 Security Hub CSPM 主控台。 3. 在 Security Hub CSPM 導覽窗格中，選擇問題清單。 4. 在問題清單中，選取您要呈報的問題清單。 5. 在動作功能表中，選擇 `CreateJiraIssue` 。	AWS 系統管理員

相關資源

- AWS 適用於 Jira Service Management 的服務管理連接器

- AWS 基礎安全最佳實務標準

其他資訊

Jira 票證的範例

發生指定的 Security Hub CSPM 調查結果時，此解決方案會自動建立 Jira 票證。票證包含下列資訊：

- 標題 – 標題會以下列格式識別安全問題：

```
AWS Security Issue :: <AWS account ID> :: <Security Hub finding title>
```

- 描述 – 票證的描述區段說明與調查結果相關聯的安全控制，包括 Security Hub CSPM 主控台中調查結果的連結，並提供如何處理 Jira 工作流程中安全問題的簡短描述。

以下是自動產生 Jira 票證的範例。

Title	AWS 安全問題：：012345678912：：Lambda.1 Lambda 函數政策應禁止公開存取。
Description	問題是什麼？ 我們在您負責的 AWS 帳戶 012345678912 中偵測到安全調查結果。 此控制項會檢查連接至 Lambda 資源的 AWS Lambda 函數政策是否禁止公開存取。如果 Lambda 函數政策允許公開存取，則控制項會失敗。 <連結至 Security Hub CSPM 調查結果> 我需要如何處理票證？ • 存取帳戶並驗證組態。將票證移至「已配置修正」，以確認處理票證。修正後，移至測試修正，讓安全性驗證問題已解決。 • 如果您認為應該接受風險，請將其移至「等待風險接受」。這需要由安全工程師審核。

- 如果您認為是誤報，請將其轉換為「標記為誤報」。這將由安全工程師審核，並相應地重新開啟/關閉。

使用 EC2 Image Builder 和 Terraform 建置強化容器映像的管道

Mike Saintcross 和 Andrew Ranes，Amazon Web Services

總結

此模式會建置 EC2 Image Builder 管道，產生強化的 Amazon Linux 2 基礎容器映像。Terraform 用作基礎設施即程式碼 (IaC) 工具，用於設定和佈建用於建立強化容器映像的基礎設施。配方可協助您部署已根據 Red Hat Enterprise Linux (RHEL) 7 STIG 第 3 版第 7 版 – 媒體強化的 Docker 型 Amazon Linux 2 容器映像。（請參閱 EC2 Image Builder 文件的 Linux STIG 元件一節中的 STIG-Build-Linux-Medium 2022.2.1 版。）這稱為黃金容器映像。

組建包含兩個 Amazon EventBridge 規則。當 Amazon Inspector 調查結果為高或嚴重時，一個規則會啟動容器映像管道，以便取代不安全映像。此規則需要同時啟用 Amazon Inspector 和 Amazon Elastic Container Registry (Amazon ECR) 增強型掃描。另一個規則會在成功將映像推送至 Amazon ECR 儲存庫後，將通知傳送至 Amazon Simple Queue Service (Amazon SQS) 佇列，以協助您使用最新的容器映像。

> **Note**
>
> Amazon Linux 2 即將結束支援。如需詳細資訊，請參閱 Amazon Linux 2 FAQs。

先決條件和限制

先決條件

- 您可以在其中部署基礎設施的 AWS 帳戶。
- 安裝 AWS Command Line Interface (AWS CLI) 以設定本機部署的 AWS 登入資料。
- 遵循 Terraform 文件中的指示下載和設定 Terraform。
- Git（如果您是從本機電腦佈建）。
- AWS 帳戶中的角色，可用來建立 AWS 資源。
- .tfvars 檔案中定義的所有變數。 或者，您可以在套用 Terraform 組態時定義所有變數。

限制

- 此解決方案會建立 Amazon Virtual Private Cloud (Amazon VPC) 基礎設施，其中包含 NAT 閘道和網際網路閘道，以便從其私有子網路進行網際網路連線。您無法使用 VPC 端點，因為 AWS Task Orchestrator 和 Executor (AWSTOE) 的引導程序會從網際網路安裝 AWS CLI 第 2 版。

產品版本

- Amazon Linux 2
- AWS CLI 1.1 版或更新版本

Architecture

目標技術堆疊

此模式會建立 43 個資源，包括：

- 兩個 Amazon Simple Storage Service (Amazon S3) 儲存貯體：一個用於管道元件檔案，另一個用於伺服器存取和 Amazon VPC 流程日誌
- Amazon ECR 儲存庫
- 虛擬私有雲端 (VPC)，其中包含公有子網路、私有子網路、路由表、NAT 閘道和網際網路閘道
- EC2 Image Builder 管道、配方和元件
- 容器映像
- 用於映像加密的 AWS Key Management Service (AWS KMS) 金鑰
- SQS 佇列
- 三個角色：一個用於執行 EC2 Image Builder 管道、一個用於 EC2 Image Builder 的執行個體描述檔，以及一個用於 EventBridge 規則
- 兩個 EventBridge 規則

Terraform 模組結構

如需原始程式碼，請參閱 GitHub 儲存庫 Terraform EC2 Image Builder Container Hardening Pipeline。

```
### components.tf
### config.tf
### dist-config.tf
```

```
### files
#   ###assumption-policy.json
### hardening-pipeline.tfvars
### image.tf
### infr-config.tf
### infra-network-config.tf
### kms-key.tf
### main.tf
### outputs.tf
### pipeline.tf
### recipes.tf
### roles.tf
### sec-groups.tf
### trigger-build.tf
### variables.tf
```

模組詳細資訊

- components.tf 包含上傳/files目錄內容的 Amazon S3 上傳資源。您也可以在這裡以模組化方式新增自訂元件 YAML 檔案。
- /files 包含定義 中所用元件.yml的檔案components.tf。
- image.tf 包含基礎映像作業系統的定義。您可以在此修改不同基礎映像管道的定義。
- infr-config.tf 和 dist-config.tf包含啟動和分配映像所需的最低 AWS 基礎設施的資源。
- infra-network-config.tf 包含要部署容器映像的最小 VPC 基礎設施。
- hardening-pipeline.tfvars 包含要在套用時間使用的 Terraform 變數。
- pipeline.tf 在 Terraform 中建立和管理 EC2 Image Builder 管道。
- recipes.tf 您可以在其中指定不同的元件混合，以建立容器配方。
- roles.tf 包含 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體描述檔和管道部署角色的 AWS Identity and Access Management (IAM) 政策定義。
- trigger-build.tf 包含 EventBridge 規則和 SQS 佇列資源。

目標架構

圖表說明下列工作流程：

1. EC2 Image Builder 會使用定義的配方來建置容器映像，這會安裝作業系統更新，並將 RHEL Medium STIG 套用至 Amazon Linux 2 基礎映像。

2. 強化的映像會發佈至私有 Amazon ECR 登錄檔，而 EventBridge 規則會在映像成功發佈時傳送訊息至 SQS 佇列。
3. 如果 Amazon Inspector 設定為增強型掃描，則會掃描 Amazon ECR 登錄檔。
4. 如果 Amazon Inspector 為映像產生嚴重或高嚴重性問題清單，EventBridge 規則會觸發 EC2 Image Builder 管道再次執行，並發佈新強化的映像。

自動化和擴展

- 此模式說明如何在您的電腦上佈建基礎設施和建置管道。不過，它旨在大規模使用。您可以在多帳戶環境中使用它們，例如 AWS Control Tower with Account Factory for Terraform environment，而不是在本機部署 Terraform 模組。在這種情況下，您應該使用後端狀態 S3 儲存貯體來管理 Terraform 狀態檔案，而不是在本機管理組態狀態。
- 針對擴展用途，請從 Control Tower 或登陸區域帳戶模型將解決方案部署至一個中央帳戶，例如共用服務或通用服務帳戶，並授予取用者帳戶存取 Amazon ECR 儲存庫和 AWS KMS 金鑰的許可。如需設定的詳細資訊，請參閱 re：Post 文章如何允許次要帳戶在 Amazon ECR 映像儲存庫中推送或提取映像？ 例如，在帳戶販賣機或 Account Factory for Terraform 中，為每個帳戶基準或帳戶自訂基準新增許可，以提供對該 Amazon ECR 儲存庫和加密金鑰的存取權。
- 部署容器映像管道之後，您可以使用 元件等 EC2 Image Builder 功能進行修改，這可協助您將更多元件封裝到 Docker 建置中。
- 用於加密容器映像的 AWS KMS 金鑰應該在映像要使用的帳戶之間共用。
- 您可以複製整個 Terraform 模組並修改下列`recipes.tf`屬性，以新增對其他映像的支援：
  - 修改`parent_image = "amazonlinux:latest"`為另一個影像類型。
  - 修改 `repository_name` 以指向現有的 Amazon ECR 儲存庫。這會建立另一個管道，將不同的父系映像類型部署到現有的 Amazon ECR 儲存庫。

工具

工具

- Terraform (IaC 佈建）
- Git （如果在本機佈建）
- AWS CLI 第 1 版或第 2 版 （如果在本機佈建）

程式碼

此模式的程式碼位於 GitHub 儲存庫 Terraform EC2 Image Builder Container Hardening Pipeline 中。若要使用範例程式碼，請依照下一節中的指示進行。

史詩

佈建 基礎設施

任務	Description	所需的技能

設定本機登入資料。	設定您的 AWS 臨時登入資料。
	
	1. 查看是否已安裝 AWS CLI：
	
	``
	$ aws --version
	aws-cli/1.16.249
	 Python/3.6.8...
	``
	
	• AWS CLI 版本應為 1.1 或更新版本。
	• 如果找不到 命令，請安裝 AWS CLI。
	2. 執行`aws configure` 並提供下列值：
	
	``
	$ aws configure
	AWS Access Key ID
	 [*************xxxx
	]: <Your AWS access
	 key ID>
	AWS Secret Access Key
	 [**************xxx
	x]: <Your AWS secret
	 access key>
	Default region name:
	 [us-east-1]: <Your
	 desired Region for
	 deployment>
	``
		AWS DevOps

任務	Description	所需的技能
	`Default output format [None]: <Your desired output format>`	

複製儲存庫。	1. 複製此模式隨附的儲存庫。您可以使用 HTTPS 或 Secure Shell (SSH)。
	
	HTTPS：
	
	`git clone https://github.com/aws-samples/terraform-ec2-image-builder-container-hardening-pipeline`
	
	SSH：
	
	`git clone git@github.com:aws-samples/terraform-ec2-image-builder-container-hardening-pipeline.git`
	2. 導覽至包含此解決方案的本機目錄：
	
	`cd terraform-ec2-image-builder-container-hardening-pipeline`
		AWS DevOps

任務	Description	所需的技能

更新變數。	更新 `hardening-pipeline.tfvars` 檔案中的變數，以符合您的環境和所需的組態。您必須提供自己的 `account_id` 。不過，您也應該修改其餘的變數，以符合所需的部署。所有變數都是必要的。
	
	`account_id = "<DEPLOYMENT-ACCOUNT-ID>"`
	`aws_region = "us-east-1"`
	`vpc_name = "example-hardening-pipeline-vpc"`
	`kms_key_alias = "image-builder-container-key"`
	`ec2_iam_role_name = "example-hardening-instance-role"`
	`hardening_pipeline_role_name = "example-hardening-pipeline-role"`
	`aws_s3_ami_resources_bucket = "example-hardening-ami-resources-bucket-0123"`
	`image_name = "example-hardening-al2-container-image"`
	`ecr_name = "example-hardening-container-repo"`
	`recipe_version = "1.0.0"`
	`ebs_root_vol_size = 10`
		AWS DevOps

任務	Description	所需的技能
	以下是每個變數的描述： • account_id – 您要部署解決方案的 AWS 帳戶號碼。 • aws_region – 您要部署解決方案的 AWS 區域。 • vpc_name – VPC 基礎設施的名稱。 • kms_key_alias – EC2 Image Builder 基礎設施組態要使用的 AWS KMS 金鑰名稱。 • ec2_iam_role_name – 將用作 EC2 執行個體描述檔的角色名稱。 • hardening_pipeline_role_name – 將用於部署強化管道的角色名稱。 • aws_s3_ami_resources_bucket – S3 儲存貯體的名稱，將託管建置管道和容器映像所需的所有檔案。 • image_name – 容器映像名稱。此值必須介於 3 到 50 個字元之間，且應僅包含英數字元和連字號。 • ecr_name – 要存放容器映像的 Amazon ECR 登錄檔名稱。	

任務	Description	所需的技能
	• `recipe_version` – 映像配方的版本。預設值為 1.0.0。 • `ebs_root_vol_size` – Amazon Elastic Block Store (Amazon EBS) 根磁碟區的大小（以 GB 為單位）。預設值為 10 GB。	
初始化 Terraform。	更新變數值後，您可以初始化 Terraform 組態目錄。初始化組態目錄會下載並安裝組態中定義的 AWS 提供者。 `terraform init` 您應該會看到一則訊息，指出 Terraform 已成功初始化，並識別已安裝的提供者版本。	AWS DevOps
部署基礎設施並建立容器映像。	使用以下命令，使用 `.tfvars` 檔案中定義的變數來初始化、驗證 Terraform 模組，並將模組套用至環境： `terraform init && terraform validate && terraform apply -var-file *.tfvars -auto-approve`	AWS DevOps

任務	Description	所需的技能
自訂容器。	您可以在 EC2 Image Builder 部署管道和初始配方之後，建立新的容器配方版本。 您可以在 EC2 Image Builder 中新增任何可用的 31+ 個元件，以自訂容器建置。如需詳細資訊，請參閱 EC2 Image Builder 文件中建立新版本容器配方的元件一節。	AWS 管理員

驗證資源

任務	Description	所需的技能

驗證 AWS 基礎設施佈建。	成功完成第一個 Terraform `apply`命令後，如果您要在本機佈建，您應該會在本機機器的終端機中看到此程式碼片段：
	
	`Apply complete!`
	`Resources: 43 added, 0`
	`changed, 0 destroyed.`
		AWS DevOps

驗證個別 AWS 基礎設施資源。	若要驗證已部署的個別資源，如果您在本機佈建，您可以執行下列命令： `terraform state list` 此命令會傳回 43 個資源的清單。	AWS DevOps

移除資源

任務	Description	所需的技能
移除基礎設施和容器映像。	使用完 Terraform 組態後，您可以執行下列命令來移除資源： `terraform init && terraform validate && terraform destroy -var-file *.tfvars -auto-approve`	AWS DevOps

疑難排解

問題	解決方案
驗證供應商登入資料時發生錯誤	當您從本機電腦執行 Terraform `apply`或`destroy`命令時，您可能會遇到類似以下的錯誤： `Error: configuring Terraform AWS Provider: error validating provider credentials: error calling sts:GetCallerIdentity: operation error STS: GetCallerIdentity, https response error StatusCode: 403, RequestID: 123456a9-fbc1-40ed-b8d8-513d0133ba7f, api error InvalidClientTokenId: The security token included in the request is invalid.` 此錯誤是由本機電腦組態中使用的登入資料的安全字符過期所造成。 若要解決錯誤，請參閱 AWS CLI 文件中的設定和檢視組態設定。

相關資源

- Terraform EC2 映像建置器容器強化管道 (GitHub 儲存庫）
- EC2 Image Builder 文件
- 適用於 Terraform 的 AWS Control Tower 帳戶工廠 (AWS 部落格文章）
- 後端狀態 S3 儲存貯體 (Terraform 文件）
- 安裝或更新最新版本的 AWS CLI (AWS CLI 文件）
- 下載 Terraform

使用 Terraform 在 AWS Organizations 中集中管理 IAM 存取金鑰

Aarti Rajput、Chintamani Aphale、T.V.R.L.Phani Kumar Dadi、Pratap Kumar Nanda、Pradip kumar Pandey 和 Mayuri Shinde、Amazon Web Services

總結

強制執行金鑰和密碼的安全規則是每個組織 的重要任務。其中一個重要規則是定期輪換 AWS Identity and Access Management (IAM) 金鑰，以強制執行安全性。每當團隊想要從 AWS 命令列界面 (AWS CLI) 或從 AWS 外部應用程式存取 AWS 時，通常會在本機建立和設定 AWS 存取金鑰。若要在整個組織中維持強大的安全性，在滿足要求之後或定期變更或刪除舊的安全金鑰。管理組織中多個帳戶間金鑰輪換的程序既耗時又繁瑣。此模式使用 Account Factory for Terraform (AFT) 和 AWS 服務，協助您自動化輪換程序。

模式提供以下優點：

- 從中央位置管理組織中所有帳戶的存取金鑰 IDs 和私密存取金鑰。
- 自動輪換 `AWS_ACCESS_KEY_ID`和 `AWS_SECRET_ACCESS_KEY`環境變數。
- 如果使用者登入資料洩露，則強制執行續約。

模式使用 Terraform 來部署 AWS Lambda 函數、Amazon EventBridge 規則和 IAM 角色。EventBridge 規則會定期執行，並呼叫 Lambda 函數，根據使用者存取金鑰的建立時間列出它們。如果先前的金鑰早於您定義的輪換期間（例如 45 天），其他 Lambda 函數會建立新的存取金鑰 ID 和私密存取金鑰，並使用 Amazon Simple Notification Service (Amazon SNS) 和 Amazon Simple Email Service (Amazon SES) 通知安全管理員。秘密是在該使用者的 AWS Secrets Manager 中建立的，舊的秘密存取金鑰是存放在 Secrets Manager 中，並且已設定存取舊金鑰的許可。為了確保不再使用舊的存取金鑰，會在非作用中期間（例如 60 天，也就是在我們的範例中輪換金鑰後 15 天）之後停用。在非作用中緩衝期間（例如，在我們的範例中輪換金鑰後 90 天或 45 天）之後，舊的存取金鑰會從 AWS Secrets Manager 中刪除。如需詳細的架構和工作流程，請參閱架構一節。

先決條件和限制

- 使用 AWS Control Tower (3.1 版或更新版本）為組織建置的登陸區域
- 使用三個帳戶設定的 Account Factory for Terraform (AFT)：
  - 組織管理帳戶會從中央位置管理整個組織。
  - AFT 管理帳戶託管 Terraform 管道，並將基礎設施部署到部署帳戶中。
  - 部署帳戶會部署此完整的解決方案，並從中央位置管理 IAM 金鑰。

- Terraform 0.15.0 版或更新版本，用於在部署帳戶中佈建基礎設施。
- 在 Amazon Simple Email Service (Amazon SES中設定的電子郵件地址。
- （建議）若要增強安全性，請在虛擬私有雲端 (VPC) 內的私有子網路（部署帳戶）內部署此解決方案。您可以在自訂變數時提供 VPC 和子網路的詳細資訊（請參閱 Epics 區段中的程式碼管道自訂參數)。

Architecture

AFT 儲存庫

此模式使用 Account Factory for Terraform (AFT) 建立所有必要的 AWS 資源和程式碼管道，以在部署帳戶中部署資源。程式碼管道在兩個儲存庫中執行：

- 全域自訂包含將跨向 AFT 註冊的所有帳戶執行的 Terraform 程式碼。
- 帳戶自訂包含將在部署帳戶中執行的 Terraform 程式碼。

資源詳細資訊

AWS CodePipeline 任務會在部署帳戶中建立下列資源：

- AWS EventBridge 規則和設定的規則
- `account-inventory` Lambda 函數
- `IAM-access-key-rotation` Lambda 函數
- `Notification` Lambda 函數
- 包含電子郵件範本的 Amazon Simple Storage Service (Amazon S3) 儲存貯體
- 必要的 IAM 政策

架構

此圖展示了以下要點：

1. EventBridge 規則每 24 小時呼叫`account-inventory`一次 Lambda 函數。
2. `account-inventory` Lambda 函數會查詢 AWS Organizations 以取得所有 AWS 帳戶 IDs、帳戶名稱和帳戶電子郵件的清單。

3. account-inventoryLambda 函數會為每個 AWS 帳戶啟動 IAM-access-key-auto-rotation Lambda 函數，並將中繼資料傳遞給它以進行其他處理。
4. IAM-access-key-auto-rotation Lambda 函數使用擔任的 IAM 角色來存取 AWS 帳戶。Lambda 指令碼會對帳戶中的所有使用者及其 IAM 存取金鑰執行稽核。
5. 部署 IAM-access-key-auto-rotation Lambda 函數時，IAM 金鑰輪換閾值（輪換期間）會設定為環境變數。如果修改輪換期間，則會使用更新的 環境變數重新部署 IAM-access-key-auto-rotation Lambda 函數。您可以設定參數來設定輪換期間、舊金鑰的非作用中期間，以及刪除舊金鑰之後的非作用中緩衝區（請參閱 Epics 區段中的程式碼管道自訂參數)。
6. IAM-access-key-auto-rotation Lambda 函數會根據存取金鑰的組態來驗證存取金鑰的存留期。如果 IAM 存取金鑰的存留期未超過您定義的輪換期間，Lambda 函數不會採取進一步動作。
7. 如果 IAM 存取金鑰的存留期已超過您定義的輪換期間，IAM-access-key-auto-rotationLambda 函數會建立新的金鑰並輪換現有的金鑰。
8. Lambda 函數會將舊金鑰儲存在 Secrets Manager 中，並將許可限制為存取金鑰偏離安全標準的使用者。Lambda 函數也會建立資源型政策，僅允許指定的 IAM 主體存取和擷取秘密。
9. IAM-access-key-rotation Lambda 函數會呼叫 Notification Lambda 函數。
10. Notification Lambda 函數會查詢電子郵件範本的 S3 儲存貯體，並動態產生具有相關活動中繼資料的電子郵件訊息。
11. Notification Lambda 函數會呼叫 Amazon SES 以進行進一步的動作。
12. Amazon SES 會傳送電子郵件到帳戶擁有者的電子郵件地址，其中包含相關資訊。

工具

AWS 服務

- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。此模式需要 IAM 角色和許可。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- AWS Secrets Manager 可協助您以 API 呼叫 Secrets Manager，以程式設計方式擷取秘密，取代程式碼中的硬式編碼登入資料，包括密碼。
- Amazon Simple Email Service (Amazon SES) 可協助您使用自己的電子郵件地址和網域來傳送和接收電子郵件。

其他工具

- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。

程式碼儲存庫

此模式的說明和程式碼可在 GitHub IAM 存取金鑰輪換儲存庫中找到。您可以在 AWS Control Tower 中央部署帳戶中部署程式碼，從中央位置管理金鑰輪換。

最佳實務

- 對於 IAM，請參閱 IAM 文件中的安全最佳實務。
- 如需金鑰輪換，請參閱 IAM 文件中的更新存取金鑰準則。

史詩

設定來源檔案

任務	Description	所需的技能

複製儲存庫。	1. 複製 IAM 存取金鑰輪換 GitHub 儲存庫：
	`$ git clone https://github.com/aws-samples/centralized-iam-key-management-aws-organizations-terraform.git`
	2. 確認儲存庫的本機副本包含三個資料夾：
	`$ cd Iam-Access-keys-Rotation`
	`$ ls`
	`org-account-customization`
	`global-account-customization`
		DevOps 工程師

任務	Description	所需的技能
	`account-custom ization`	

設定帳號

任務	Description	所需的技能

設定引導帳戶。	作為 AFT 引導程序的一部分，您應該在本機電腦上有一個名為 `aft-bootstrap` 的資料夾。
	1. 手動將所有 Terraform 檔案從本機 GitHub org-account-customization 資料夾複製到您的`aft-bootstrap` 資料夾。
	2. 執行 Terraform 命令以在 AWS Control Tower 管理帳戶中設定全域跨帳戶角色：
	`$ cd aft-bootstrap`
	`$ terraform init`
	`$ terraform apply –auto-approve`
		DevOps 工程師

設定全域自訂。	作為 AFT 資料夾設定的一部分，您應該在本機電腦上有一個名為 `aft-global-customizations` 的資料夾。 1. 手動將所有 Terraform 檔案從本機 GitHub global-account-customization 資料	DevOps 工程師

任務	Description	所需的技能

	夾複製到您的`aft-global-customizations/terraform` 資料夾。
	2. 將程式碼推送至 AWS CodeCommit：
	`$ git add *`
	`$ git commit -m "message"`
	`$ git push`	

設定帳戶自訂。	作為 AFT 資料夾設定的一部分，您在本機電腦上有一個名為 `aft-account-customizations` 的資料夾。
	1. 使用您提供的帳號建立資料夾。
	2. 手動將所有 Terraform 檔案從本機 GitHub 帳戶自訂資料夾複製到您的`aft-account-customizations/<vended account>/terraform` 資料夾。
	3. 將程式碼推送至 AWS CodeCommit：
	`$ git add *`
	`$ git commit -m "message"`
	`$ git push`
		DevOps 工程師

自訂程式碼管道的參數

任務	Description	所需的技能
為所有帳戶自訂非 Terraform 程式碼管道參數。	在 `aft-global-customizations/terraform/` 資料夾中建立名為 `input.auto.tfvars` 的檔案，並提供所需的輸入資料。如需預設值，請參閱 GitHub 儲存庫中的 檔案。	DevOps 工程師
自訂部署帳戶的程式碼管道參數。	在 `aft-account-customizations/<AccountName>/terraform/` 資料夾中建立名為 `input.auto.tfvars` 的檔案，並將程式碼推送至 AWS CodeCommit。將程式碼推送至 AWS CodeCommit 會自動啟動程式碼管道。 根據組織的需求指定參數的值，包括下列項目（請參閱 Github 儲存庫中的 檔案以取得預設值）： • `s3_bucket_name` – 電子郵件範本的唯一儲存貯體名稱。 • `s3_bucket_prefix` – S3 儲存貯體內的資料夾名稱。 • `admin_email_address` – 應接收通知之管理員的電子郵件地址。 • `org_list_account` – 管理帳戶的帳號。	DevOps 工程師

任務	Description	所需的技能
	• `rotation_period` – 金鑰應該從作用中輪換為非作用中的天數。 • `inactive_period` – 輪換金鑰應停用的天數。此值必須大於 的值`rotation_period` 。 • `inactive_buffer` – 輪換與停用金鑰之間的寬限期。 • `recovery_grace_period` – 停用和刪除金鑰之間的寬限期。 • `dry_run_flag` – 如果您想要將通知傳送給管理員進行測試，而不輪換金鑰，請將 設為 true。 • `store_secrets_in_central_account` – 如果您想要將秘密存放在部署帳戶中，請將 設為 true。如果變數設定為 false （預設），則秘密會存放在成員帳戶中。 • `credential_replication_region` – 您要部署 Lambda 函數的 AWS 區域，以及電子郵件範本的 S3 儲存貯體。 • `run_lambda_in_vpc` – 設為 true 以在 VPC 內執行 Lambda 函數。	

任務	Description	所需的技能
	• vpc_id – 部署帳戶的 VPC ID，如果您想要在 VPC 內執行 Lambda 函數。 • vpc_cidr – 部署帳戶的 CIDR 範圍。 • subnet_id – 部署帳戶的子網路 IDs。 • create_smtp_endpoint – 如果您想要啟用電子郵件端點，請將 設定為 true。	

驗證金鑰輪換

任務	Description	所需的技能
驗證解決方案。	1. 從 AWS 管理主控台登入部署帳戶。 2. 開啟 IAM 主控台，並檢查使用者登入資料（存取金鑰 IDs和私密金鑰）是否依指定輪換。 3. 輪換 IAM 金鑰之後，請確認下列事項： • 舊值存放在 AWS Secrets Manager 中。 • 秘密名稱的格式為 Account_<account ID>_User_<username>_AccessKey 。 • 您在 admin_email_address 參數中指	DevOps 工程師

任務	Description	所需的技能
	定的使用者會收到有關金鑰輪換的電子郵件通知。	

擴展解決方案

任務	Description	所需的技能
自訂電子郵件通知日期。	如果您想要在特定日期停用存取金鑰之前傳送電子郵件通知，您可以使用這些變更來更新 `IAM-access-key-auto-rotation` Lambda 函數： 1. 定義名為 的變數`notify-period` 。 2. 在 中新增`if`條件，`main.py`再停用金鑰： `If (keyage>rotation-period-notify-period){ send_to_notifier(context, aws_account_id, account_name, resource_owner, resource_actions[resource_owner], dryrun, config.emailTemplateAudit) }`	DevOps 工程師

疑難排解

問題	解決方案
`account-inventory` Lambda 任務在列出帳戶`AccessDenied` 時失敗。	如果您遇到此問題，您必須驗證許可： 1. 登入新付費的帳戶，開啟 Amazon CloudWatch 主控台，然後檢視 CloudWatch 日誌群組 `/aws/lambda/account-inventory-lambda` 。 2. 在最新的 CloudWatch 日誌中，識別導致存取遭拒問題的帳號。 3. 登入 AWS Control Tower 管理帳戶並確認`allow-list-account` 已建立角色。 4. 如果角色不存在，請使用 `terraform apply`命令重新執行 Terraform 程式碼。 5. 選擇信任帳戶索引標籤，並驗證同一帳戶是否受信任。

相關資源

- Terraform 建議實務 (Terraform 文件)
- IAM 中的安全最佳實務 (IAM 文件)
- 金鑰輪換的最佳實務 (IAM 文件)

檢查 Amazon CloudFront 分佈是否有存取記錄、HTTPS 和 TLS 版本

SaiJeevan Devireddy 和 Bijesh Bal，Amazon Web Services

總結

此模式會檢查 Amazon CloudFront 分佈，以確保其使用 HTTPS、使用 Transport Layer Security (TLS) 1.2 版或更新版本，以及啟用存取記錄。CloudFront 是由 Amazon Web Services (AWS) 提供的服務，可加速將靜態和動態 Web 內容，例如 .html、.css、.js 和映像檔案分發給使用者。CloudFront 透過稱為節點的資料中心全球網路交付您的內容。當使用者請求您使用 CloudFront 提供的內容時，請求會被路由到可提供最低延遲 (時間延遲) 的節點，以便能以最佳的效能發佈內容。

此模式提供 AWS Lambda 函數，會在 Amazon CloudWatch Events 偵測到 CloudFront API 呼叫 CreateDistribution、CreateDistributionWithTags 或 UpdateDistribution 時啟動。Lambda 函數中的自訂邏輯會評估在 AWS 帳戶中建立或更新的所有 CloudFront 分佈。如果偵測到下列違規，它會使用 Amazon Simple Notification Service (Amazon SNS) 傳送違規通知：

- 全域檢查：
  - 自訂憑證不使用 TLS 1.2 版
  - 已停用記錄以進行分佈
- 原始伺服器檢查：
  - 原始伺服器未設定 TLS 1.2 版
  - 允許在 HTTPS 以外的通訊協定上與原始伺服器通訊
- 行為檢查：
  - HTTPS 以外的通訊協定允許預設行為通訊
  - 允許在 HTTPS 以外的通訊協定上進行自訂行為通訊

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 您想要接收違規通知的電子郵件地址

限制

- 除非已對分佈進行更新，否則此安全控制不會檢查現有 Cloudfront 分佈。
- CloudFront 被視為全球服務，且與特定 AWS 區域無關。不過，適用於全球服務的 Amazon CloudWatch Logs 和 AWS Cloudtrail API 記錄會位於美國東部（維吉尼亞北部）區域 (us-east-1)。因此，此 CloudFront 的安全控制必須在 中部署和維護us-east-1。此單一部署會監控 CloudFront 的所有分佈。請勿在任何其他 AWS 區域中部署安全控制。（在其他區域中部署會導致無法啟動 CloudWatch Events 和 Lambda 函數，而且沒有 SNS 通知。）
- 此解決方案已透過 CloudFront Web 內容分發進行廣泛的測試。它不涵蓋即時傳訊通訊協定 (RTMP) 串流分佈。

Architecture

目標技術堆疊

- Lambda 函式
- SNS 主題
- Amazon EventBridge 規則

目標架構

自動化和擴展

- 如果您使用的是 AWS Organizations，則可以使用 AWS Cloudformation StackSets 將連接的範本部署到您要監控的多個帳戶。

工具

AWS 服務

- AWS CloudFormation – CloudFormation 是一項服務，可協助您使用基礎設施做為程式碼來建立模型和設定 AWS 資源。
- Amazon EventBridge – EventBridge 可從您自己的應用程式、軟體即服務 (SaaS) 應用程式和 AWS 服務提供即時資料串流，將該資料路由到 Lambda 函數等目標。
- AWS Lambda – Lambda 支援執行程式碼，無需佈建或管理伺服器。

- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種高度可擴展的物件儲存服務，可用於各種儲存解決方案，包括網站、行動應用程式、備份和資料湖。
- Amazon SNS – Amazon SNS 會協調和管理發佈者和用戶端之間的訊息傳遞或傳送，包括 Web 伺服器和電子郵件地址。訂閱者會收到發佈到所訂閱主題的所有訊息，且某一主題的所有訂閱者均會收到相同訊息。

Code

附加的程式碼包括：

- 包含 Lambda 程式碼的 .zip 檔案 (index.py：//)
- 您執行以部署 Lambda 程式碼的 CloudFormation 範本 (.yml 檔案)

史詩

上傳安全控制

任務	Description	所需的技能
為 Lambda 程式碼建立 S3 儲存貯體。	在 Amazon S3 主控台上，使用不包含正斜線的唯一名稱建立 S3 儲存貯體。S3 儲存貯體名稱全域唯一，且命名空間由所有 AWS 帳戶共用。您的 S3 儲存貯體必須位於您計劃部署 Lambda 程式碼的區域中。	雲端架構師
將 Lambda 程式碼上傳至 S3 儲存貯體。	將附件區段中提供的 Lambda 程式碼 (cloudfront_ssl_log_lambda.zip 檔案) 上傳到您在上一個步驟中建立的 S3 儲存貯體。	雲端架構師

部署 CloudFormation 範本

任務	Description	所需的技能
部署 CloudFormation 範本。	在 AWS CloudFormation 主控台上，在與 S3 儲存貯體相同的 AWS 區域中，部署附件區段中提供的 CloudFormation 範本 (cloudfront-ssl-logging.yml)。	雲端架構師
指定 S3 儲存貯體名稱。	針對 S3 儲存貯體參數，指定您在第一個 epic 中建立的 S3 儲存貯體名稱。	雲端架構師
指定 Lambda 檔案的 Amazon S3 金鑰名稱。	針對 S3 金鑰參數，指定 SAmazon S3 S3 位置。請勿包含正斜線 （例如，您可以輸入 lambda.zip 或 control/lambda.zip)。	雲端架構師
提供通知電子郵件地址。	對於通知電子郵件參數，請提供您要接收違規通知的電子郵件地址。	雲端架構師
定義記錄層級。	針對 Lambda 記錄層級參數，定義 Lambda 函數的記錄層級。請選擇下列其中一個值： • INFO 以取得應用程式進度的詳細資訊訊息。 • 取得仍可允許應用程式繼續執行之錯誤事件相關資訊的錯誤。 • 警告 以取得潛在有害情況的相關資訊。	雲端架構師

確認訂閱

任務	Description	所需的技能
確認訂閱。	成功部署 CloudFormation 範本後，會建立新的 SNS 主題，並將訂閱訊息傳送至您提供的電子郵件地址。您必須確認此電子郵件訂閱，才能接收違規通知。	雲端架構師

相關資源

- AWS CloudFormation 資訊
- 在 AWS CloudFormation 主控台上建立堆疊 (CloudFormation 文件)
- CloudFront 記錄 (CloudFront 文件)
- Amazon S3 資訊
- AWS Lambda 資訊

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

為企業應用程式選擇 Amazon Cognito 身分驗證流程

Michael Daehnert 和 Fabian Jahnke，Amazon Web Services

總結

Amazon Cognito 為 Web 和行動應用程式提供身分驗證、授權和使用者管理。它為聯合身分的身分驗證提供了有益的功能。若要啟動並執行，技術架構師需要決定如何使用這些功能。

Amazon Cognito 支援驗證請求的多個流程。這些流程會定義您的使用者如何驗證其身分。使用哪個身分驗證流程的決定取決於您應用程式的特定需求，並且可能會變得複雜。此模式可協助您決定哪個身分驗證流程最適合您的企業應用程式。其假設您已具備 Amazon Cognito、OpenID Connect (OIDC) 和聯合身分的基本知識，並引導您完成有關不同聯合身分驗證流程的詳細資訊。

此解決方案適用於技術決策者。它可協助您了解不同的身分驗證流程，並將其映射到您的應用程式需求。技術主管應收集必要的洞見，以啟動 Amazon Cognito 整合。由於企業組織主要專注於 SAML 聯合，因此此模式包含具有 SAML 聯合的 Amazon Cognito 使用者集區的描述。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 具有 Amazon Cognito 完整存取權的 AWS Identity and Access Management (IAM) 角色和許可
- （選用）存取您的身分提供者 (IdP)，例如 Microsoft Entra ID、Active Directory Federation Service (AD FS) 或 Okta
- 應用程式的高階專業知識
- Amazon Cognito、OpenID Connect (OIDC) 和聯合的基本知識

限制

- 此模式著重於 Amazon Cognito 使用者集區和身分提供者。如需 Amazon Cognito 身分集區的相關資訊，請參閱其他資訊一節。

Architecture

使用下表協助您選擇身分驗證流程。本節提供每個流程的詳細資訊。

您需要machine-to-machine的身分驗證嗎？	您的應用程式是否為在伺服器上轉譯前端的 Web 應用程式？	您的應用程式是單頁應用程式 (SPA) 還是行動型前端應用程式？	您的應用程式是否需要「保持我登入」功能的重新整理權杖？	前端是否提供瀏覽器型重新導向機制？	建議的 Amazon Cognito 流程
是	否	否	否	否	用戶端登入資料流程
否	是	否	是	是	授權碼流程
否	否	是	是	是	授權碼流程與程式碼交換的驗證金鑰 (PKCE)
否	否	否	否	否	資源擁有者密碼流程*

* 只有在絕對必要時，才應使用資源擁有者密碼流程。如需詳細資訊，請參閱此模式中的資源擁有者密碼流程一節。

用戶端登入資料流程

用戶端登入資料流程是 Amazon Cognito 流程的最短流程。如果系統或服務彼此通訊而沒有任何使用者互動，則應該使用它。請求系統使用用戶端 ID 和用戶端秘密來擷取存取字符。由於這兩個系統皆可在沒有使用者互動的情況下運作，因此不需要額外的同意步驟。

此圖展示了以下要點：

1. 應用程式 1 會將具有用戶端 ID 和用戶端秘密的身分驗證請求傳送至 Amazon Cognito 端點，並擷取存取字符。
2. 應用程式 1 會針對每次後續對應用程式 2 的呼叫使用此存取字符。
3. 應用程式 2 會使用 Amazon Cognito 驗證存取權杖。

應該使用此流程：

- 對於沒有使用者互動的應用程式之間的通訊

不應使用此流程：

- 對於任何可能進行使用者互動的通訊

授權碼流程

授權碼流程適用於傳統 Web 型身分驗證。在此流程中，後端會處理所有字符交換和儲存。瀏覽器型用戶端不會看到實際字符。此解決方案用於以 .NET Core、Jakarta Faces 或 Jakarta Server Pages (JSP) 等架構撰寫的應用程式。

授權碼流程是以重新導向為基礎的流程。用戶端必須能夠與 Web 瀏覽器或類似用戶端互動。用戶端會重新導向至身分驗證伺服器，並對此伺服器進行身分驗證。如果用戶端驗證成功，則會將其重新導向回伺服器。

此圖展示了以下要點：

1. 用戶端會將請求傳送至 Web 伺服器。
2. Web 伺服器會使用 HTTP 302 狀態碼，將用戶端重新導向至 Amazon Cognito。用戶端會自動遵循此重新導向至設定的 IdP 登入。
3. IdP 會檢查 IdP 端的現有瀏覽器工作階段。如果不存在，使用者會提供使用者名稱和密碼來收到驗證的提示。IdP 會以 SAML 字符回應 Amazon Cognito。
4. Amazon Cognito 使用 JSON Web 權杖 (JWT) 傳回成功，特別是程式碼權杖。Web 伺服器會呼叫 /oauth2/token 來交換存取字符的程式碼字符。Web 伺服器會將用戶端 ID 和用戶端秘密傳送至 Amazon Cognito 進行驗證。
5. 存取權杖會用於後續每次呼叫其他應用程式。
6. 其他應用程式會使用 Amazon Cognito 驗證存取權杖。

應該使用此流程：

- 如果使用者能夠與 Web 瀏覽器或用戶端互動。應用程式程式碼會在伺服器上執行和轉譯，以確保不會向瀏覽器公開任何秘密。

不應使用此流程：

- 對於單頁應用程式 (SPAs) 或行動應用程式，因為它們是在用戶端上轉譯的，不應使用用戶端秘密。

使用 PKCE 的授權碼流程

使用 Code Exchange (PKCE) 的驗證金鑰的授權碼流程應用於單頁應用程式和行動應用程式。它是隱含流程的後繼者，並且使用 PKCE 更安全。PKCE 是公有用戶端 OAuth 2.0 授權碼授予的延伸。PKCE 可防止兌換攔截的授權碼。

此圖展示了以下要點：

1. 應用程式會建立程式碼驗證程式和程式碼挑戰。這些是明確定義的唯一值，會傳送到 Amazon Cognito 以供日後參考。
2. 應用程式會呼叫 Amazon Cognito 的 /oauth2/authorization 端點。它會自動將使用者重新導向至設定的 IdP 登入。
3. IdP 會檢查現有工作階段。如果不存在，使用者會提供使用者名稱和密碼來收到驗證的提示。IdP 會以 SAML 字符回應 Amazon Cognito。
4. 在 Amazon Cognito 以程式碼字符傳回成功後，Web 伺服器會呼叫 /oauth2/token 以交換程式碼字符做為存取字符。
5. 存取權杖會用於後續每次呼叫其他應用程式。
6. 其他應用程式會使用 Amazon Cognito 驗證存取權杖。

應該使用此流程：

- 對於 SPAs 或行動應用程式

不應使用此流程：

- 如果應用程式後端處理身分驗證

資源擁有者密碼流程

資源擁有者密碼流程適用於沒有重新導向功能的應用程式。其建置方式是在您自己的應用程式中建立登入表單。登入是透過 CLI 或 SDK 呼叫在 Amazon Cognito 上檢查，而不是依賴重新導向流程。此身分驗證流程中無法聯合，因為聯合需要瀏覽器型重新導向。

此圖展示了以下要點：

1. 使用者在應用程式提供的登入表單中輸入其登入資料。
2. AWS Command Line Interface (AWS CLI) 會對 Amazon Cognitoadmin-initiated-auth呼叫。

> **Note**
>
> 或者，您可以使用 AWS SDKs，而不是 AWS CLI。

3. Amazon Cognito 會傳回存取字符。
4. 存取權杖會用於後續每次呼叫其他應用程式。
5. 其他應用程式會使用 Amazon Cognito 驗證存取權杖。

應該使用此流程：

- 將使用直接身分驗證邏輯的現有用戶端（例如基本存取身分驗證或摘要存取身分驗證）遷移至 OAuth 時，請將儲存的登入資料轉換為存取字符

不應使用此流程：

- 如果您想要使用聯合身分
- 如果您的應用程式支援重新導向

工具

AWS 服務

- Amazon Cognito 為 Web 和行動應用程式提供身分驗證、授權和使用者管理。

其他工具

- JSON Web 權杖 (JWT) 偵錯工具是一種以 Web 為基礎的 JWT 驗證工具。

史詩

評估您的應用程式

任務	Description	所需的技能
定義身分驗證需求。	根據您的特定身分驗證要求評估您的應用程式。	應用程式開發人員、應用程式架構師
將需求與身分驗證流程保持一致。	在架構區段中，使用決策表和每個流程的說明來選擇您的 Amazon Cognito 身分驗證流程。	應用程式開發人員、一般 AWS、應用程式架構師

設定 Amazon Cognito 使用者集區

任務	Description	所需的技能
建立使用者集區。	1. 登入 AWS 管理主控台，然後開啟 Amazon Cognito 主控台。 2. 建立新的 Cognito 使用者集區。如需說明，請參閱 Amazon Cognito 使用者集區。 3. 視需要更新使用者集區設定和屬性。例如，設定使用者集區的密碼政策。尚未建立應用程式用戶端。	一般 AWS
（選用）設定身分提供者。	1. 在 Amazon Cognito 使用者集區中建立 SAML 身分提供者。如需說明，請參閱在使用者集區中新增和管理 SAML 身分提供者。	一般 AWS、聯合管理員

任務	Description	所需的技能
	2. 將您的第三方 SAML 身分提供者設定為使用 Amazon Cognito 使用者集區的聯合。如需詳細資訊，請參閱設定第三方 SAML 身分提供者。如果您使用的是 AD FS，請參閱使用 Amazon Cognito 使用者集區為 Web 應用程式建立 AD FS 聯合 (AWS 部落格文章）。	
建立應用程式用戶端。	1. 為使用者集區建立應用程式用戶端。如需說明，請參閱建立應用程式用戶端。注意下列事項： • 視需要變更設定，例如權杖過期。 • 如果您的身分驗證流程不需要用戶端秘密，請清除產生用戶端秘密核取方塊。 2. 選擇應用程式用戶端設定，透過 SAML 型 IdP 變更其與使用者集區登入（使用者名稱和密碼）或聯合登入的整合。 3. 定義 URLs 並視需要定義 OAuth 流程或範圍，以啟用您的 IdP。	一般 AWS

將應用程式與 Amazon Cognito 整合

任務	Description	所需的技能
Exchange Amazon Cognito 整合詳細資訊。	根據您的身分驗證流程，與應用程式共用 Amazon Cognito 資訊，例如使用者集區 ID 和應用程式用戶端 ID。	應用程式開發人員，一般 AWS
實作 Amazon Cognito 身分驗證。	這取決於您選擇的身分驗證流程、您的程式設計語言，以及您正在使用的架構。如需一些開始使用的連結，請參閱相關資源一節。	應用程式開發人員

相關資源

AWS 文件

- 使用者集區身分驗證流程
- 驗證 JSON Web 權杖
- 使用 Amazon Cognito 身分集區從 ASP.NET Core 應用程式存取 AWS 服務
- 架構和SDKs：
  - Amazon Amplify 身分驗證
  - Amazon Cognito 身分提供者範例 （適用於 Java 的 AWS 開發套件 2.x 文件）
  - 使用 Amazon Cognito （適用於 .NET 的 AWS 開發套件文件） 驗證使用者

AWS 部落格文章

- 使用 Cookie 的 Authorization@Edge：保護您的 Amazon CloudFront 內容不受未經驗證的使用者下載
- 使用 Amazon Cognito 使用者集區為 Web 應用程式建置 AD FS 聯合

實作合作夥伴

- 身分驗證解決方案的 AWS 合作夥伴

其他資訊

常見問答集

為什麼隱含流程已棄用？

自 OAuth 2.1 架構發行以來，基於安全考量，隱含流程會標示為已棄用。或者，請使用授權碼流程搭配架構一節所述的 PKCE。

如果 Amazon Cognito 不提供我需要的一些功能，該怎麼辦？

AWS 合作夥伴提供不同的身分驗證和授權解決方案整合。如需詳細資訊，請參閱 AWS 合作夥伴的身分驗證解決方案。

Amazon Cognito 身分集區流程呢？

Amazon Cognito 使用者集區和聯合身分用於身分驗證。Amazon Cognito 身分集區透過請求臨時 AWS 登入資料，用於授權 AWS 資源存取。此模式不會討論身分集區的 ID 字符和存取字符交換。如需詳細資訊，請參閱 Amazon Cognito 使用者集區和身分集區與常見 Amazon Cognito 案例之間的差異。Amazon Cognito

後續步驟

此模式提供 Amazon Cognito 身分驗證流程的概觀。下一步，需要選擇應用程式程式設計語言的詳細實作。多種語言提供 SDKs 和架構，您可以搭配 Amazon Cognito 使用。如需實用參考，請參閱相關資源一節。

使用 AWS CloudFormation Guard 政策建立 AWS Config 自訂規則

Andrew Lok、Nicole Brown、Kailash Havildar 和 Tanya Howell，Amazon Web Services

總結

AWS Config 規則可協助您評估 AWS 資源及其目標組態狀態。 AWS Config 規則有兩種類型： 受管和自訂。您可以使用 AWS Lambda 函數或使用policy-as-codeAWS CloudFormation Guard(GitHub) 建立自訂規則。

使用 Guard 建立的規則提供比受管規則更精細的控制，而且通常比完全自訂的 Lambda 規則更容易設定。此方法可讓工程師和架構師能夠建置規則，而不需要知道 Python、NodeJS 或 Java，而這些都是透過 Lambda 部署自訂規則的必要項目。

此模式提供可行的範本、程式碼範例和部署方法，協助您使用 Guard 來採用自訂規則。透過使用此模式，管理員可以使用 AWS Config 來建置具有組態項目屬性的自訂合規規則。例如，開發人員可以針對 AWS Config 組態項目使用 Guard 政策，以持續監控已部署 AWS 和非AWS 資源的狀態、偵測規則違規，並自動啟動修復。

目標

讀取此模式後，您應該能夠：

- 了解 Guard 政策程式碼如何與服務互動 AWS Config 。
- 部署案例 1，這是使用 Guard 語法來驗證加密磁碟區合規性的 AWS Config 自訂規則。此規則會驗證磁碟機正在使用中，並驗證磁碟機類型為 gp3。
- 部署案例 2，這是使用 Guard 語法來驗證 Amazon GuardDuty 合規性的 AWS Config 自訂規則。此規則會驗證 GuardDuty 記錄器是否已啟用 Amazon Simple Storage Service (Amazon S3) 保護和 Amazon Elastic Kubernetes Service (Amazon EKS) 保護。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- AWS Config，在 中設定 AWS 帳戶

限制

- Guard 自訂規則只能查詢目標組態項目 JSON 記錄中的鍵值對

Architecture

您可以將 Guard 語法套用到 AWS Config 規則做為自訂 policy. AWS Config captures 每個指定資源的階層式 JSON。 AWS Config 組態項目的 JSON 包含鍵/值對。這些屬性會用在 Guard 語法中，做為指派給其對應值的變數。

以下是 Guard 語法的說明。使用組態項目 JSON 中的變數，並在前面加上%字元。

```
# declare variable
let <variable name> = <'value'>

# create rule and assign condition and policy
    rule <rule name> when
        <CI json key> == <"CI json value"> {
            <top level CI json key>.<next level CI json key> == %<variable name>
        }
```

案例 1：Amazon EBS 磁碟區

案例 1 部署使用 Guard 語法來驗證加密磁碟區合規性的 AWS Config 自訂規則。此規則會驗證磁碟機正在使用中，並驗證磁碟機類型為 gp3。

以下是案例 1 的 AWS Config 組態項目範例。此組態項目中有三個鍵值對，用作 Guard 政策中的變數：`volumeencryptionstatus`、 `volumestatus`和 `volumetype`。此外，`resourceType`金鑰會用作 Guard 政策中的篩選條件。

```
{
  "version": "1.3",
  "accountId": "111111111111",
  "configurationItemCaptureTime": "2023-01-15T19:04:45.402Z",
  "configurationItemStatus": "ResourceDiscovered",
  "configurationStateId": "4444444444444",
  "configurationItemMD5Hash": "",
  "arn": "arn:aws:ec2:us-west-2:111111111111:volume/vol-222222222222",
  "resourceType": "AWS::EC2::Volume",
  "resourceId": "vol-222222222222",
  "awsRegion": "us-west-2",
  "availabilityZone": "us-west-2b",
  "resourceCreationTime": "2023-01-15T19:03:22.247Z",
```

```
  "tags": {},
  "relatedEvents": [],
  "relationships": [
    {
      "resourceType": "AWS::EC2::Instance",
      "resourceId": "i-33333333333333333",
      "relationshipName": "Is attached to Instance"
    }
  ],
  "configuration": {
    "attachments": [
      {
        "attachTime": "2023-01-15T19:03:22.000Z",
        "device": "/dev/xvda",
        "instanceId": "i-33333333333333333",
        "state": "attached",
        "volumeId": "vol-222222222222",
        "deleteOnTermination": true,
        "associatedResource": null,
        "instanceOwningService": null
      }
    ],
    "availabilityZone": "us-west-2b",
    "createTime": "2023-01-15T19:03:22.247Z",
    "encrypted": false,
    "kmsKeyId": null,
    "outpostArn": null,
    "size": 8,
    "snapshotId": "snap-55555555555555555",
    "state": "in-use",
    "volumeId": "vol-222222222222",
    "iops": 100,
    "tags": [],
    "volumeType": "gp2",
    "fastRestored": null,
    "multiAttachEnabled": false,
    "throughput": null,
    "sseType": null
  },
  "supplementaryConfiguration": {}
}
```

以下是使用 Guard 語法來定義案例 1 中變數和規則的範例。於下列範例中：

- 前三行使用 `let`命令定義變數。 系統會為他們指派衍生自組態項目屬性的名稱和值。
- 當條件相依性尋找符合 的`resourceType`鍵值對時，`compliancecheck`規則區塊會新增 `AWS::EC2::Volume`。如果找到相符項目，則規則會繼續進行其餘 JSON 屬性，並在下列三個條件下尋找相符項目：`state`、 `encrypted`和 `volumeType`。

```
let volumestatus = 'available'
let volumetype = 'gp3'
let volumeencryptionstatus = true

    rule compliancecheck when
        resourceType == "AWS::EC2::Volume" {
            configuration.state == %volumestatus
            configuration.encrypted == %volumeencryptionstatus
            configuration.volumeType == %volumetype
        }
```

如需實作此自訂規則的完整 Guard 自訂政策，請參閱 GitHub 程式碼儲存庫中的 awsconfig-guard-cft.yaml 或 awsconfig-guard-tf-ec2vol.json。如需在 Guard 中部署此自訂政策的 HashiCorp Terraform 程式碼，請參閱程式碼儲存庫中的 awsconfig-guard-tf-example.json。

案例 2：GuardDuty 合規

案例 2 部署使用 Guard 語法來驗證 Amazon GuardDuty 合規性的 AWS Config 自訂規則。此規則會驗證 GuardDuty 記錄器是否已啟用 Amazon S3 保護和 Amazon EKS 保護。它也會驗證每 15 分鐘發佈一次 GuardDuty 調查結果。此案例可以部署到組織 (in AWS Organizations) AWS 區域 中的所有 AWS 帳戶 和 。

以下是案例 2 的 AWS Config 組態項目範例。此組態項目中有三個鍵/值對，用作 Guard 政策中的變數：`FindingPublishingFrequency`、 `S3Logs`和 `Kubernetes`。此外，`resourceType`金鑰會用作政策中的篩選條件。

```
{
  "version": "1.3",
  "accountId": "111111111111",
  "configurationItemCaptureTime": "2023-11-27T13:34:28.888Z",
  "configurationItemStatus": "OK",
  "configurationStateId": "7777777777777",
  "configurationItemMD5Hash": "",
  "arn": "arn:aws:guardduty:us-
west-2:111111111111:detector/66666666666666666666666666666666",
```

```
  "resourceType": "AWS::GuardDuty::Detector",
  "resourceId": "66666666666666666666666666666666",
  "resourceName": "66666666666666666666666666666666",
  "awsRegion": "us-west-2",
  "availabilityZone": "Regional",
  "resourceCreationTime": "2020-02-17T02:48:04.511Z",
  "tags": {},
  "relatedEvents": [],
  "relationships": [],
  "configuration": {
    "Enable": true,
    "FindingPublishingFrequency": "FIFTEEN_MINUTES",
    "DataSources": {
      "S3Logs": {
        "Enable": true
      },
      "Kubernetes": {
        "AuditLogs": {
          "Enable": true
        }
      }
    },

    "Id": "66666666666666666666666666666666",
    "Tags": []
  },
  "supplementaryConfiguration": {
    "CreatedAt": "2020-02-17T02:48:04.511Z"
  }
}
```

以下是使用 Guard 語法定義案例 2 中變數和規則的範例。於下列範例中：

- 前三行使用 `let`命令定義變數。 系統會為他們指派衍生自組態項目屬性的名稱和值。
- 當條件相依性尋找符合 的`resourceType`鍵值對時，`compliancecheck`規則區塊會新增 `AWS::GuardDuty::Detector`。如果找到相符項目，則規則會繼續進行其餘的 JSON 屬性，並在下列三個條件下尋找相符項目：`S3Logs.Enable`、 `Kubernetes.AuditLogs.Enable`和 `FindingPublishingFrequency`。

```
let s3protection = true
let kubernetesprotection = true
```

```
let publishfrequency = 'FIFTEEN_MINUTES'

    rule compliancecheck when
        resourceType == "AWS::GuardDuty::Detector" {
            configuration.DataSources.S3Logs.Enable == %s3protection
            configuration.DataSources.Kubernetes.AuditLogs.Enable ==
 %kubernetesprotection
            configuration.FindingPublishingFrequency == %publishfrequency
        }
```

如需實作此自訂規則的完整 Guard 自訂政策，請參閱 GitHub 程式碼儲存庫中的 awsconfig-guard-cft-gd.yaml。如需在 Guard 中部署此自訂政策的 HashiCorp Terraform 程式碼，請參閱程式碼儲存庫中的 awsconfig-guard-tf-gd.json。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和區域的生命週期中進行管理。
- AWS Config 提供 中資源的詳細檢視 AWS 帳戶 及其設定方式。它可協助您識別資源彼此之間的關係，以及其組態如何隨著時間而改變。

其他工具

- HashiCorp Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。

程式碼儲存庫

此模式的程式碼可在 AWS Config GitHub AWS CloudFormation Guard GitHub 搭配儲存庫中使用。此程式碼儲存庫包含此模式所述的兩種案例的範例。

史詩

建立 AWS Config 自訂規則

任務	Description	所需的技能
（選用）選取規則的鍵/值對。	如果您要定義自訂 Guard 政策，請完成以下步驟。如果您針對案例 1 或 2 使用其中一個範例政策，請略過這些步驟。 1. 登入 AWS 管理主控台 並開啟 AWS Config 主控台。 2. 在左側導覽中，選擇資源。 3. 在資源庫存中，選擇要為其建立 AWS Config 自訂規則的資源類型。 4. 請選擇檢視詳細資料。 5. 選擇檢視組態項目 (JSON)。本節展開以顯示 JSON 格式的組態項目。 6. 識別您要為其建置 AWS Config 自訂規則的鍵/值對。	AWS 管理員、安全工程師
建立自訂規則。	使用您先前識別的鍵/值對，或使用其中一個提供的範例 Guard 政策，遵循建立 AWS Config 自訂政策規則中的指示來建立自訂規則。	AWS 管理員、安全工程師
驗證自訂規則。	執行下列其中一項作業來驗證自訂 Guard 規則： • 在 AWS Command Line Interface () 中輸入下列命令 AWS CLI。	AWS 管理員、安全工程師

任務	Description	所需的技能
	`cfn-guard validate -r guard-s3.guard -d s3bucket-prod-pass.json` • 遵循 Detective 模式中使用AWS Config 規則評估您的資源以部署規則的指示 AWS Config。確認 Guard 語法與目標帳戶或檔案中的對應資源正確相符。	

疑難排解

問題	解決方案

在 外部測試 Guard 政策 AWS Config	單元測試可以在本機裝置或整合開發環境 (IDE) 中完成，例如 AWS Cloud9 IDE。若要執行單位測試，請執行下列動作：
	1. 安裝 AWS CloudFormation Guard CLI 及其相依性。
	2. 將 JSON 格式 CI 範例儲存為 .json 檔案到您的工作站。
	3. 將 GuardDuty 政策儲存為 .guard 檔案到您的工作站。
	4. 在 Guard CLI 中，輸入下列命令，使用 Guard 政策驗證範例 JSON 檔案。
	`cfn-guard validate \`
	`-r guard-s3.guard \`
	`-d s3bucket-prod-pass.json`

問題	解決方案
偵錯 AWS Config 自訂規則	在您的 Guard 政策中，將`EnableDebugLogDelivery` 值變更為 `true`。預設值為 `false`。日誌訊息存放在 Amazon CloudWatch 中。

相關資源

AWS 文件

- 建立 AWS Config 自訂政策規則 (AWS Config 文件）
- 撰寫 AWS CloudFormation Guard 規則 (Guard 文件）

AWS 部落格文章和研討會

- Introducing AWS CloudFormation Guard 2.0 (AWS 部落格文章）

其他資源

- AWS CloudFormation Guard (GitHub)
- AWS CloudFormation Guard CLI 文件 (GitHub)

從多個 建立 Prowler 安全性問題清單的合併報告 AWS 帳戶

Mike Virgilio、Jay Durga 和 Andrea Di Fabio，Amazon Web Services

總結

Prowler (GitHub) 是一種開放原始碼命令列工具，可協助您評估、稽核和監控 Amazon Web Services (AWS) 帳戶是否符合安全最佳實務。在此模式中，您會在組織中的集中式 AWS 帳戶 中部署 Prowler，由 管理 AWS Organizations，然後使用 Prowler 對組織中的所有帳戶執行安全評估。

雖然有許多方法可以部署和利用 Prowler 進行評估，但此解決方案是專為快速部署、組織或定義目標帳戶中的所有帳戶的完整分析，以及安全調查結果的可存取報告而設計。在此解決方案中，當 Prowler 完成組織中所有帳戶的安全評估時，它會合併結果。它也會篩選掉任何預期的錯誤訊息，例如與限制相關的錯誤，這些限制會阻止 Prowler 在佈建的帳戶中掃描 Amazon Simple Storage Service (Amazon S3) 儲存貯體 AWS Control Tower。篩選的合併結果會以 Microsoft Excel 範本報告，此範本包含於此模式。您可以使用此報告來識別組織中安全控制的潛在改進。

此解決方案的設計考量如下：

- AWS CloudFormation 範本可減少在此模式中部署 AWS 資源所需的工作量。
- 您可以在部署時調整 CloudFormation 範本和 prowler_scan.sh 指令碼中的參數，為您的環境自訂範本。
- 透過平行處理 AWS 帳戶、彙總結果、使用建議補救措施的合併報告，以及自動產生的視覺化效果，來最佳化生產者評估和報告速度。
- 使用者不需要監控掃描進度。評估完成後，系統會透過 Amazon Simple Notification Service (Amazon SNS) 主題通知使用者，以便他們可以擷取報告。
- 報告範本可協助您僅讀取和評估整個組織的相關結果。

先決條件和限制

先決條件

- AWS 帳戶 用於託管安全服務和工具的 ，作為組織的成員帳戶進行管理 AWS Organizations。在此模式中，此帳戶稱為安全帳戶。
- 在安全帳戶中，您必須擁有具有傳出網際網路存取權的私有子網路。如需說明，請參閱《Amazon Virtual Private Cloud (Amazon VPC) 文件》中的 VPC 與私有子網路中的伺服器以及 NAT。您可以使用公有子網路中佈建的 AnNAT 閘道來建立網際網路存取。

- 存取 AWS Organizations 管理帳戶或具有 CloudFormation 委派管理員許可的帳戶。如需說明，請參閱 CloudFormation 文件中的註冊委派管理員。
- 啟用 AWS Organizations 和 CloudFormation 之間的受信任存取。如需說明，請參閱 CloudFormation 文件中的使用 啟用受信任存取 AWS Organizations。

限制

- 目標 AWS 帳戶 必須以組織身分管理 AWS Organizations。如果您未使用 AWS Organizations，您可以更新您環境的 IAM-ProwlerExecRole.yaml CloudFormation 範本和 prowler_scan.sh 指令碼。反之，您可以提供您要執行指令碼 AWS 帳戶 IDs 和區域的清單。
- CloudFormation 範本旨在將 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體部署在具有傳出網際網路存取權的私有子網路中。 AWS Systems Manager 代理程式 (SSM 代理程式） 需要傳出存取權才能到達 AWS Systems Manager 服務端點，而您需要傳出存取權才能複製程式碼儲存庫並安裝相依性。如果您想要使用公有子網路，您必須修改 prowler-resources.yaml 範本，將彈性 IP 地址與 EC2 執行個體建立關聯。

產品版本

- Prowler 4.0 版或更新版本

Architecture

圖表顯示下列程序：

1. 使用 的 Session Manager 功能 AWS Systems Manager，使用者會向 EC2 執行個體進行身分驗證，並執行 prowler_scan.sh 指令碼。此 shell 指令碼會執行步驟 2–8。
2. EC2 執行個體會擔任 `ProwlerEC2Role` IAM 角色，授予存取 S3 儲存貯體和在組織中其他帳戶中擔任 IAM `ProwlerExecRole` 角色的許可。
3. EC2 執行個體會在組織的管理帳戶中擔任 `ProwlerExecRole` IAM 角色，並產生組織中的帳戶清單。
4. EC2 執行個體會擔任組織成員帳戶中的 `ProwlerExecRole` IAM 角色 （ 在架構圖表中稱為工作負載帳戶)，並在每個帳戶中執行安全評估。調查結果會以 CSV 和 HTML 檔案形式存放在 EC2 執行個體上。

> **Note**
>
> HTML 檔案是 Prowler 評估的輸出。由於 HTML 的性質，它們不會在此模式中直接串連、處理或使用。不過，這些對於個別帳戶報告檢閱可能很有用。

5. EC2 執行個體會處理所有 CSV 檔案，以移除已知的預期錯誤，並將剩餘的問題清單合併為單一 CSV 檔案。
6. EC2 執行個體會將個別帳戶結果和彙總結果封裝為 zip 檔案。
7. EC2 執行個體會將 zip 檔案上傳至 S3 儲存貯體。
8. EventBridge 規則會偵測檔案上傳，並使用 Amazon SNS 主題傳送電子郵件給使用者，通知他們評估已完成。
9. 使用者從 S3 儲存貯體下載 zip 檔案。使用者將結果匯入 Excel 範本並檢閱結果。

工具

AWS 服務

- Amazon Elastic Compute Cloud (Amazon EC2) 在 AWS 雲端中提供可擴展的運算容量。您可以視需要啟動任意數量的虛擬伺服器，，並快速進行擴展或縮減。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如， AWS Lambda 函數、使用 API 目的地的 HTTP 呼叫端點，或其他事件匯流排 AWS 帳戶。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Systems Manager 可協助您管理在 中執行的應用程式和基礎設施 AWS 雲端。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。此模式使用 Systems Manager 功能的 Session Manager。

其他工具

- Prowler 是一種開放原始碼命令列工具，可協助您評估、稽核和監控您的帳戶是否符合 AWS 安全最佳實務和其他安全架構和標準。

程式碼儲存庫

此模式的程式碼可透過 Prowler 儲存庫在 GitHub 多帳戶安全評估中取得。 https://github.com/aws-samples/multi-account-security-assessment-via-prowler程式碼儲存庫包含下列檔案：

- prowler_scan.sh – 此 bash 指令碼用於 AWS 帳戶平行啟動多個 Prowler 安全評估。如 Prowler-resources.yaml CloudFormationtemplate 中所定義，此指令碼會自動部署到 EC2 執行個體上的 `usr/local/prowler` 資料夾。
- Prowler-Resources.yaml – 您可以使用此 CloudFormation 範本在組織的安全帳戶中建立堆疊。此範本會部署此帳戶所有必要的資源，以支援解決方案。此堆疊必須部署在 IAM-ProwlerExecRole.yaml 範本之前。我們不建議您將這些資源部署在託管關鍵生產工作負載的 帳戶中。

> **Note**
>
> 如果刪除並重新部署此堆疊，您必須重建`ProwlerExecRole`堆疊集，才能在 IAM 角色之間重建跨帳戶相依性。

- IAM-ProwlerExecRole.yaml – 您可以使用此 CloudFormation 範本建立堆疊集，在組織中的所有帳戶中部署 `ProwlerExecRole` IAM 角色，包括 管理帳戶。
- prowler-report-template.xlsm – 您可以使用此 Excel 範本來處理 Prowler 調查結果。報告中的樞紐分析表提供搜尋功能、圖表和合併調查結果。

史詩

準備部署

任務	Description	所需的技能
複製程式碼儲存庫。	1. 在命令列界面中，將工作目錄變更為您要存放範例檔案的位置。 2. 輸入以下命令：	AWS DevOps

任務	Description	所需的技能
	`git clone https://github.com/aws-samples/multi-account-security-assessment-via-prowler.git`	
檢閱範本。	1. 在複製的儲存庫中，開啟 Prowler-Resources.yaml 和 IAM-ProwlerExecRole.yaml 檔案。 2. 檢閱這些範本建立的資源，並視需要調整您環境的範本。如需詳細資訊，請參閱 CloudFormation 文件中的使用範本。 3. 儲存並關閉 Prowler-Resources.yaml 和 IAM-ProwlerExecRole.yaml 檔案。	AWS DevOps

建立 CloudFormation 堆疊

任務	Description	所需的技能
在安全帳戶中佈建資源。	使用 prowler-resources.yaml 範本，您可以建立 CloudFormation 堆疊，以部署安全帳戶中所有必要的資源。如需說明，請參閱 CloudFormation 文件中的建立堆疊。部署此範本時，請注意下列事項： 1. 在指定範本頁面上，選擇範本已就緒，然後上傳	AWS DevOps

任務	Description	所需的技能
	prowler-resources.yaml 檔案。 2. 在指定堆疊詳細資訊頁面上的堆疊名稱方塊中，輸入 `Prowler-Resources` 。 3. 在參數區段中，輸入下列內容： • `VPCId` – 選取帳戶中的 VPC。 • `SubnetId` — 選取具有網際網路存取權的私有子網路。 注意：如果您選取公有子網路，將不會指派公有 IP 地址給 EC2 執行個體，因為 CloudFormation 範本預設不會佈建和連接彈性 IP 地址。 • `InstanceType` – 根據平行評估的數量選取執行個體大小： • 針對 10，選擇 `r6i.large` 。 • 針對 12，選擇 `r6i.xlarge` 。 • 針對 14–18，選擇 `r6i.2xlarge` 。 • `InstanceImageId` – 保留 Amazon Linux 的預設值。	

任務	Description	所需的技能
	• KeyPairName – 如果您使用 SSH 進行存取，請指定現有金鑰對的名稱。 • PermittedSSHInbound – 如果您使用 SSH 進行存取，請指定允許的 CIDR 區塊。如果您未使用 SSH，請保留預設值 127.0.0.1 。 • BucketName – 預設值為 prowler-output-<accountID>-<region> 。您可以視需要修改此項目。如果您指定自訂值，帳戶 ID 和區域會自動附加到指定的值。 • EmailAddress – 在 Prowler 完成評估並將 .zip 檔案上傳至 S3 儲存貯體時，指定 Amazon SNS 通知的電子郵件地址。 注意：SNS 訂閱組態必須在 Prowler 完成評估之前確認，否則不會傳送通知。 • IAMProwlerEC2Role – 除非您的命名慣例需要此 IAM 角色的不同名稱，否則請保留預設值。 • IAMProwlerExecRole – 保留預設值，除非在部署 IAM-ProwlerExecRol	

任務	Description	所需的技能
	e.yaml 檔案時將使用另一個名稱。 • `Parallelism` – 指定要執行的平行評估數目。請確定 `InstanceType` 參數中的 值支援此數量的平行評估。 • `FindingOutput` – 如果您想要排除傳遞結果，請選取 `FailOnly`。這可大幅減少輸出大小，並專注於可能需要解決的檢查。如果您想要包含傳遞結果，請選取 `FailAndPass` 。 4. 在檢閱頁面上，選取下列資源需要功能：【AWS::IAM::Role】，然後選擇建立堆疊。 5. 堆疊成功建立後，在 CloudFormation 主控台的輸出索引標籤上，複製 `ProwlerEC2Role` Amazon Resource Name (ARN)。您稍後在部署 IAM-ProwlerExecRole.yaml 檔案時使用此 ARN。	

任務	Description	所需的技能
在成員帳戶中佈建 IAM 角色。	在 AWS Organizations 管理帳戶或具有 CloudFormation 委派管理員許可的帳戶中，使用 IAM-ProwlerExecRole.yaml 範本建立 CloudFormation 堆疊集。堆疊集會在組織中的所有成員帳戶中部署 `ProwlerExecRole` IAM 角色。如需說明，請參閱 CloudFormation 文件中的建立具有服務管理許可的堆疊集。部署此範本時，請注意下列事項： 1. 在準備範本下，選擇範本已就緒，然後上傳 IAM-ProwlerExecRole.yaml 檔案。 2. 在指定 StackSet 詳細資訊頁面上，將堆疊集命名為 `IAM-ProwlerExecRole` 。 3. 在參數區段中，輸入下列內容： • `AuthorizedARN` – 輸入您在建立`Prowler-Resources` 堆疊時複製的 `ProwlerEC2Role` ARN。 • `ProwlerExecRoleName` – 保留預設值，`ProwlerExecRole` 除非在部署 Prowler-Resources.yaml 檔案時使用另一個名稱。	AWS DevOps

任務	Description	所需的技能
	4. 在 Permissions (許可) 下，選擇 Service-managed permissions (服務管理許可)。 5. 在設定部署選項頁面的部署目標下，選擇部署到組織並接受所有預設值。 注意：如果您想要同時將堆疊部署到所有成員帳戶，請將並行帳戶上限和容錯能力設定為高值，例如 100。 6. 在部署區域下，選擇部署 Prowler AWS 區域 EC2 執行個體的 。由於 IAM 資源是全域而非區域性，因此這會在所有作用中區域中部署 IAM 角色。 7. 在檢閱頁面上，選取我確認 AWS CloudFormation 可能會使用自訂名稱建立 IAM 資源，然後選擇建立 StackSet。 8. 監控堆疊執行個體索引標籤（適用於個別帳戶狀態）和操作索引標籤（適用於整體狀態），以判斷部署何時完成。	

任務	Description	所需的技能
在管理帳戶中佈建 IAM 角色。	使用 IAM-ProwlerExecRole.yaml 範本，您可以建立 CloudFormation 堆疊，在組織的管理帳戶中部署 `ProwlerExecRole` IAM 角色。您先前建立的堆疊集不會在管理帳戶中部署 IAM 角色。如需說明，請參閱 CloudFormation 文件中的建立堆疊。部署此範本時，請注意下列事項： 1. 在指定範本頁面上，選擇範本已就緒，然後上傳 IAM-ProwlerExecRole.yaml 檔案。 2. 在指定堆疊詳細資訊頁面上的堆疊名稱方塊中，輸入 `IAM-ProwlerExecRole` 。 3. 在參數區段中，輸入下列內容： • `AuthorizedARN` – 輸入您在建立`Prowler-Resources` 堆疊時複製的 `ProwlerEC2Role` ARN。 • `ProwlerExecRoleName` – 保留預設值，`ProwlerExecRole` 除非在部署 Prowler-Resources.yaml 檔案時使用另一個名稱。 4. 在檢閱頁面上，選取下列資源需要功能：【AWS::IAM	AWS DevOps

任務	Description	所需的技能
	::Role】，然後選擇建立堆疊。	

執行 Prowler 安全評估

任務	Description	所需的技能
執行掃描。	1. 登入組織中的安全帳戶。 2. 使用 Session Manager，連線至您先前佈建的 Prowler EC2 執行個體。如需說明，請參閱使用 Session Manager 連線至 Linux 執行個體。如果您無法連線，請參閱此模式的故障診斷一節。 3. 導覽至 `usr/local/prowler`，然後開啟 prowler_scan.sh 檔案。 4. 視需要檢閱和修改此指令碼中的可調整參數和變數。如需自訂選項的詳細資訊，請參閱指令碼開頭的註解。 例如，您可以修改指令碼以指定 AWS 區域 要掃描 AWS 帳戶 IDs 或 ，或者參考包含這些參數的外部檔案，而不是從管理帳戶取得組織中所有成員帳戶的清單。 5. 儲存並關閉 prowler_scan.sh 檔案。	AWS 管理員

任務	Description	所需的技能

	6. 輸入下列命令：這會執行 prowler_scan.sh 指令碼。
	
	`sudo -i`
	`screen`
	`cd /usr/local/prowler`
	`./prowler_scan.sh`
	
	注意下列事項：
	• 螢幕命令允許指令碼在連線逾時或您失去主控台存取權時繼續執行。
	• 掃描開始後，您可以按下 Ctrl+A D 強制分離螢幕。畫面分離，您可以關閉執行個體連線，並允許評估繼續進行。
	• 若要繼續分離的工作階段，請連線至執行個體，輸入 `sudo -i`，然後輸入 `screen -r`。
	• 若要監控個別帳戶評估的進度，您可以導覽至 `usr/local/prowler`目錄，然後輸入命令 `tail -f output/stdout-<account-id>`。
	7. 等待 Prowler 在所有帳戶中完成掃描。指令碼會同時評估多個帳戶。在所有帳戶中完成評估時，如果您在部署 Prowler-Resources.yaml 檔	

任務	Description	所需的技能
	案時指定電子郵件地址，則會收到通知。	
擷取 Prowler 調查結果。	1. 從 `prowler-output-<accountID>-<region>` 儲存貯體下載 `prowler-output-<assessDate>.zip` 檔案。如需說明，請參閱 Amazon S3 文件中的下載物件。Amazon S3 2. 刪除儲存貯體中的所有物件，包括您下載的檔案。這是成本最佳化的最佳實務，並確保您可以隨時刪除 `Prowler-Resources` CloudFormation 堆疊。如需說明，請參閱 Amazon S3 文件中的刪除物件。	一般 AWS
停止 EC2 執行個體。	若要在執行個體閒置時防止計費，請停止執行 Prowler 的 EC2 執行個體。如需說明，請參閱 Amazon EC2 文件中的停止和啟動執行個體。	AWS DevOps

建立問題清單的報告

任務	Description	所需的技能
匯入問題清單。	1. 在 Excel 中，開啟 prowler-report-template.xlsx 檔案，然後選擇 Prowler CSV 工作表。	一般 AWS

任務	Description	所需的技能
	2. 刪除所有範例資料，包括標頭列。如果您被詢問是否刪除與要移除的資料相關聯的查詢，請選擇否。 刪除查詢可能會影響 Excel 範本中樞紐分析表的功能。 3. 從 S3 儲存貯體擷取您下載的 zip 檔案內容。 4. 在 Excel 中，開啟 prowler-fullorgresults-accessdeniedfiltered.txt。我們建議您使用此檔案，因為最常見的不可動作錯誤已移除，例如與嘗試掃描 AWS Control Tower 資源相關的`Access Denied`錯誤。如果您想要未篩選的問題清單，請改為開啟 prowler-fullorgresults.txt 檔案。 5. 選取欄 A。 6. 如果您使用的是 Windows，請輸入 Ctrl+C，或者如果您使用的是 MacOS，請輸入 Cmd+C。 這會將所有資料複製到剪貼簿。 7. 在 Excel 報告範本的 Prowler CSV 工作表上，選取儲存格 A1。 8. 如果您使用的是 Windows，請輸入 Ctrl+V，或者如果您使用的是 MacOS，請輸入 Cmd+V。 這會將問題清單貼到報告中。	

任務	Description	所需的技能
	9. 確認已選取包含貼上資料的所有儲存格。如果沒有，請選取欄 A。 10.在資料索引標籤上，選擇文字轉資料欄。 11.在精靈中，執行下列動作： • 針對步驟 1，選擇分隔。 • 對於步驟 2，對於分隔符號，選擇分號。在資料預覽窗格中，確認資料已分隔為資料欄。 • 針對步驟 3，選擇完成。 12.確認文字資料跨多個資料欄分隔。 13.使用新名稱儲存 Excel 報告。 14.搜尋並刪除問題清單中的任何`Access Denied`錯誤。如需如何以程式設計方式移除這些項目的說明，請參閱其他資訊一節中的以程式設計方式移除錯誤。	

任務	Description	所需的技能
完成報告。	1. 選擇調查結果工作表，然後選取儲存格 A17。此儲存格是樞紐分析表的標頭。 2. 在功能區中，在 PivotTable 工具下，選擇分析，然後在重新整理下，選擇全部重新整理。這會使用新資料集更新樞紐分析表。 3. 根據預設，Excel 不會正確顯示 AWS 帳戶 數字。若要修正數字格式，請執行下列動作： • 在問題清單工作表上，開啟欄 A 的內容（按一下滑鼠右鍵）選單，然後選擇格式化儲存格。 • 選擇數字，然後在小數點中，輸入 0。 • 選擇確定。 注意：如果 AWS 帳戶 數字以一個或多個零開頭，則 Excel 會自動移除零。如果您在報告中看到少於 12 位數的帳號，則缺少的數字在號碼開頭為零。 4.（選用）您可以摺疊欄位，讓問題清單更容易閱讀。請執行下列操作： • 在問題清單工作表上，如果您將游標移至資料列 18 和 19 之間的行（關鍵標頭和第一個問題清單之	一般 AWS

任務	Description	所需的技能
	間的空格），則游標圖示會變更為指向下方的小箭頭。 • 按一下以選取所有問題清單欄位。 • 開啟內容（按一下滑鼠右鍵）選單，尋找展開/收合，然後選擇收合。 5. 如需評估的詳細資訊，請檢閱調查結果、嚴重性和通過失敗工作表。	

（選用）更新 Prowler 或程式碼儲存庫中的資源

任務	Description	所需的技能

更新 Prowler。	如果您想要將 Prowler 更新至最新版本，請執行下列動作：
	1. 使用 Session Manager 連線至適用於 Prowler 的 EC2 執行個體。如需說明，請參閱使用 Session Manager 連線至 Linux 執行個體。
	2. 輸入以下命令。
	`sudo -i`
	`pip3 install --upgrade prowler`
		一般 AWS

更新 prowler_scan.sh 指令碼。	如果您想要將 prowler_scan.sh 指令碼更新為儲存庫中的最新版本，請執行下列動作：	一般 AWS

任務	Description	所需的技能

	1. 使用 Session Manager 連線至適用於 Prowler 的 EC2 執行個體。如需說明，請參閱使用 Session Manager 連線至 Linux 執行個體。
	2. 輸入以下命令。
	`sudo -i`
	3. 導覽至 Prowler 指令碼目錄。
	`cd /usr/local/prowler`
	4. 輸入下列命令來存放本機指令碼，讓您可以將自訂變更合併到最新版本。
	`git stash`
	5. 輸入下列命令以取得指令碼的最新版本。
	`git pull`
	6. 輸入下列命令，將自訂指令碼與最新版本的指令碼合併。
	`git stash pop`
	Note
	您可能會收到與 GitHub 儲存庫中未本	

任務	Description	所需的技能
	機產生的任何檔案相關的警告，例如問題清單報告。只要 prowler_scan.sh 顯示本機銷毀的變更會重新合併，您就可以忽略這些變更。	

(選用) 清除

任務	Description	所需的技能
刪除所有部署的資源。	您可以在帳戶中保留部署的資源。如果您在 EC2 執行個體未使用時將其關閉，並保持 S3 儲存貯體空白，這可降低維護資源以供未來掃描的成本。 如果您想要取消佈建所有資源，請執行下列動作： 1. 刪除管理帳戶中佈建的`IAM-ProwlerExecRole` 堆疊。如需說明，請參閱 CloudFormation 文件中的刪除堆疊。 2. 刪除組織管理帳戶或委派管理員帳戶中佈建的`IAM-ProwlerExecRole` 堆疊集。如需說明，請參閱 CloudFormation 文件中的刪除堆疊集。 3. 刪除 `prowler-output` S3 儲存貯體中的所有物件。	AWS DevOps

任務	Description	所需的技能
	如需說明，請參閱 Amazon S3 文件中的刪除物件。 4. 刪除在安全帳戶中佈建的`Prowler-Resources` 堆疊。如需說明，請參閱 CloudFormation 文件中的刪除堆疊。	

疑難排解

問題	解決方案
無法使用 Session Manager 連線至 EC2 執行個體。	SSM Agent 必須能夠與 Systems Manager 端點通訊。請執行下列操作： 1. 驗證部署 EC2 執行個體的子網路是否具有網際網路存取權。 2. 重新啟動 EC2 執行個體。
部署堆疊集時，CloudFormation 主控台會提示您使用 `Enable trusted access with AWS Organizations to use service-managed permissions` 。	這表示 AWS Organizations 和 CloudFormation 之間尚未啟用受信任的存取。部署服務受管堆疊集需要信任的存取權。選擇 按鈕以啟用受信任的存取。如需詳細資訊，請參閱 CloudFormation 文件中的啟用受信任存取。

相關資源

AWS 文件

- 在 上實作安全控制 AWS (AWS 方案指引)

其他資源

- Prowler (GitHub)

其他資訊

以程式設計方式移除錯誤

如果結果包含Access Denied錯誤，您應該從調查結果中移除它們。這些錯誤通常是由於外部影響許可導致 Prowler 無法評估特定資源。例如，某些檢查在檢閱透過 佈建的 S3 儲存貯體時失敗 AWS Control Tower。您可以以程式設計方式擷取這些結果，並將篩選結果儲存為新檔案。

下列命令會移除包含單一文字字串 （模式） 的資料列，然後將結果輸出至新檔案。

- 針對 Linux 或 MacOS (Grep)

```
grep -v -i "Access Denied getting bucket" myoutput.csv > myoutput_modified.csv
```

- 適用於 Windows (PowerShell)

```
Select-String -Path myoutput.csv -Pattern 'Access Denied getting bucket' -NotMatch >
 myoutput_modified.csv
```

下列命令會移除符合多個文字字串的資料列，然後將結果輸出至新檔案。

- 針對 Linux 或 MacOS （在字串之間使用逸出管道）

```
grep -v -i 'Access Denied getting bucket\|Access Denied Trying to Get' myoutput.csv >
 myoutput_modified.csv
```

- 對於 Windows （字串之間使用逗號）

```
Select-String -Path myoutput.csv -Pattern 'Access Denied getting bucket', 'Access
 Denied Trying to Get' -NotMatch > myoutput_modified.csv
```

報告範例

下圖是合併 Prowler 調查結果報告中調查結果工作表的範例。

下圖是合併 Prowler 調查結果報告中通過失敗工作表的範例。（根據預設，傳遞結果會從輸出中排除。）

下圖是合併 Prowler 調查結果報告中嚴重性工作表的範例。

使用 和 AWS CDK CloudFormation 部署和管理 AWS Control Tower 控制項

Iker Reina Fuente 和 Ivan Girardi，Amazon Web Services

總結

此模式說明如何使用 AWS CloudFormation 和 AWS Cloud Development Kit (AWS CDK) 實作和管理預防性、偵測性和主動性 AWS Control Tower 控制作為基礎設施即程式碼 (IaC)。控制項（也稱為護欄) 是一種高階規則，可為您的整體 AWS Control Tower 環境提供持續的控管。例如，您可以使用控制項來要求記錄 ， AWS 帳戶 然後在發生特定安全相關事件時設定自動通知。

AWS Control Tower 可協助您實作預防性、偵測性和主動性控制，以控管您的 AWS 資源並監控多個資源的合規性 AWS 帳戶。每個控制項都會強制執行單一規則。在此模式中，您可以使用提供的 IaC 範本來指定要在環境中部署的控制項。

AWS Control Tower 控制項適用於整個組織單位 (OU)，而控制項會影響 OU AWS 帳戶 中的每個單位。因此，當使用者在您登陸區域的任何帳戶中執行任何動作時，該動作會受到管理 OU 的控制。

實作 AWS Control Tower 控制項有助於為您的 AWS 登陸區域建立強大的安全基礎。透過使用此模式透過 CloudFormation 將控制項部署為 IaC AWS CDK，您可以標準化登陸區域中的控制項，並更有效率地部署和管理它們。此解決方案使用 cdk_nag 在部署期間掃描 AWS CDK 應用程式。此工具會檢查應用程式是否符合 AWS 最佳實務。

若要將 AWS Control Tower 控制項部署為 IaC，您也可以使用 HashiCorp Terraform 而非 AWS CDK。如需詳細資訊，請參閱使用 Terraform 部署和管理 AWS Control Tower 控制項。

目標對象

對於具有 CloudFormation AWS Control Tower和 經驗的使用者 AWS CDK，建議使用此模式 AWS Organizations。

先決條件和限制

先決條件

- 作為 AWS Organizations 和 AWS Control Tower 登陸區域中的組織進行主動 AWS 帳戶 管理。如需說明，請參閱 AWS Control Tower 文件中的入門。

- AWS Command Line Interface (AWS CLI)，已安裝並設定。
- 節點套件管理員 (npm)，已安裝並設定 AWS CDK。
- 的先決條件 AWS CDK。
- 部署帳戶中擔任現有 AWS Identity and Access Management (IAM) 角色的許可。
- 在組織管理帳戶中擔任 IAM 角色的許可，可用於引導 AWS CDK。角色必須具有修改和部署 CloudFormation 資源的許可。如需詳細資訊，請參閱 AWS CDK 文件中的引導。
- 在組織的管理帳戶中建立 IAM 角色和政策的許可。如需詳細資訊，請參閱 IAM 文件中的存取 IAM 資源所需的許可。

限制

- 此模式提供從部署帳戶到組織管理帳戶跨 AWS 帳戶部署此解決方案的說明。基於測試目的，您可以直接在管理帳戶中部署此解決方案，但不會明確提供此組態的指示。
- 對於 AWS Control Tower 控制項，此模式需要使用以下格式的全域識別符：

```
arn:<PARTITION>:controlcatalog:::control/<CONTROL_CATALOG_OPAQUE_ID>
```

此模式的先前版本使用不再支援的區域識別符。我們建議您從區域識別符遷移至全域識別符。全域識別符可協助您管理控制項，並擴展您可以使用的控制項數量。

> **Note**
>
> 在大多數情況下， 的值<PARTITION>為 aws。

產品版本

- AWS Control Tower 3.2 版或更新版本
- Python 3.9 版或更新版本
- npm 8.9.0 版或更新版本

Architecture

本節提供此解決方案的高階概觀，以及範例程式碼所建立的架構。下圖顯示部署在 OU 中各種帳戶的控制項。

AWS Control Tower 控制項會根據其行為及其指引進行分類。

控制行為有三種主要類型：

1. 預防性控制旨在防止動作發生。這些會在 中使用服務控制政策 SCPs) 或資源控制政策 RCPs) 實作 AWS Organizations。預防性控制的狀態為強制執行或未啟用。所有 都支援預防性控制 AWS 區域。
2. Detective 控制項旨在偵測發生的特定事件，並記錄動作 AWS CloudTrail。這些是使用 AWS Config 規則實作的。偵測性控制項的狀態為明確、違規或未啟用。Detective 控制項僅適用於 AWS 區域 支援的控制項 AWS Control Tower。
3. 主動控制掃描由 佈建的資源， AWS CloudFormation 並檢查這些資源是否符合您的公司政策和目標。不會佈建不合規的資源。這些會使用AWS CloudFormation 勾點實作。主動控制的狀態為 PASS、FAIL 或 SKIP。

控制指導是指如何將每個控制套用至 OUs 的建議實務。 AWS Control Tower 提供三種類型的指導：強制性、強烈建議和選擇性。控制項的指引與其行為無關。如需詳細資訊，請參閱控制行為和指引。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端 基礎設施。Toolkit AWS CDK是與您的 AWS CDK 應用程式互動的主要工具。
- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。
- AWS Config 提供 中資源的詳細檢視 AWS 帳戶 及其設定方式。它可協助您識別資源彼此之間的關係，以及其組態如何隨著時間而改變。
- AWS Control Tower 可協助您設定和管理 AWS 多帳戶環境，並遵循規範性最佳實務。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。

其他工具

- cdk_nag 是一種開放原始碼工具，使用規則套件的組合來檢查 AWS CDK 應用程式是否符合最佳實務。

- npm 是在 Node.js 環境中執行的軟體登錄檔，用於共用或借用套件和管理私有套件的部署。
- Python 是一種一般用途的電腦程式設計語言。

程式碼儲存庫

此模式的程式碼可在 GitHub 部署控制項中使用儲存庫。 AWS Control TowerAWS CDK您可以使用 cdk.json 檔案與 AWS CDK 應用程式互動，並使用 package.json 檔案安裝 npm 套件。

最佳實務

- 遵循最低權限原則 (IAM 文件）。此模式中提供的範例 IAM 政策和信任政策包含所需的最低許可，而且在管理帳戶中建立的 AWS CDK 堆疊受限於這些許可。
- 遵循AWS Control Tower 管理員的最佳實務 (AWS Control Tower 文件）。
- 遵循使用 （文件） 開發和部署雲端基礎設施的最佳實務 AWS CDK。AWS CDK
- 引導 時 AWS CDK，自訂引導範本以定義政策和信任的帳戶，這些帳戶應該能夠讀取和寫入管理帳戶中的任何資源。如需詳細資訊，請參閱自訂引導。
- 使用程式碼分析工具掃描產生的 CloudFormation 範本，例如 cfn_nag。cfn-nag 工具會在 CloudFormation 範本中尋找可能表示基礎設施不安全的模式。您也可以使用 CloudFormation-include 模組，使用 cdk-nag 來檢查 CloudFormation 範本。

史詩

準備啟用控制項

任務	Description	所需的技能
在管理帳戶中建立 IAM 角色。	1. 在管理帳戶中建立具有其他資訊區段中 IAM 政策中定義許可的 IAM 政策。如需說明，請參閱 IAM 文件中的建立 IAM 政策。請記下政策的 Amazon Resource Name (ARN)。以下是範例 ARN。 `arn:aws:iam::<MANAGEMENT-ACCOUNT-ID>`	DevOps 工程師，一般 AWS

任務	Description	所需的技能

	<pre>:policy/<POLICY-NA
	ME></pre>
	2. 在管理帳戶中建立 IAM 角色、連接您在上一個步驟中建立的 IAM 許可政策，以及附加其他資訊區段中的信任政策中的自訂信任政策。如需說明，請參閱《IAM 文件》中的使用自訂信任政策建立角色。以下是新角色的範例 ARN。
	<pre>arn:aws:iam::
	<MANAGEMENT-ACCOUN
	T-ID>:role/<ROLE-N
	AME></pre>	

任務	Description	所需的技能
引導 AWS CDK。	1. 在管理帳戶中，擔任具有引導許可的角色 AWS CDK。 2. 輸入下列命令，取代下列命令： • <MANAGEMENT-ACCOUNT-ID> 是組織的管理帳戶的 ID。 • <AWS-CONTROL-TOWER-REGION> 是部署 AWS 區域 的 AWS Control Tower 。如需區域代碼的完整清單，請參閱《AWS 一般參考》中的區域端點。 • <DEPLOYMENT-ACCOUNT-ID> 是部署帳戶的 ID。 • <DEPLOYMENT-ROLE-NAME> 是您使用部署帳戶的 IAM 角色名稱。 • <POLICY-NAME> 是您在管理帳戶中建立的政策名稱。 `$ npx cdk bootstrap aws://<MANAGEMENT-ACCOUNT-ID>/<AWS-CONTROL-TOWER-REGION> \ --trust arn:aws:iam::<DEPLOYMENT-ACCOUNT-ID>:role/<DEPLOYMENT-ROLE-NAME> \`	DevOps 工程師、一般 AWS、Python

任務	Description	所需的技能
	`--cloudformation-execution-policies arn:aws:iam::<MANAGEMENT-ACCOUNT-ID>:policy/<POLICY-NAME>`	
複製儲存庫。	在 bash shell 中，輸入下列命令。這會複製使用 GitHub 儲存庫的部署 AWS Control Tower 控制項 AWS CDK。 `git clone https://github.com/aws-samples/aws-control-tower-controls-cdk.git`	DevOps 工程師，一般 AWS

任務	Description	所需的技能
編輯 AWS CDK 組態檔案。	1. 在複製的儲存庫中，開啟 constants.py 檔案。 2. 在 ACCOUNT_ID 參數中，輸入管理帳戶的 ID。 3. 在 <AWS-CONTROL-TOWER-REGION> 參數中，輸入部署 AWS 區域 AWS Control Tower 所在的 。 4. 在 ROLE_ARN 參數中，輸入您在管理帳戶中建立之角色的 ARN。 5. 開啟 AWS Control Tower 文件中的所有全域識別碼。 6. 在 JSON 格式清單中，找到您要實作的控制項，然後複製其全域識別符（也稱為 {CONTROL_CATALOG_OPAQUE_ID} 值）。例如，AWS-GR_AUDIT_BUCKET_ENCRYPTION_ENABLED 控制項的全域識別符為 k4izcjxhukijhajp6ks5mjxk 。 7. 在 GUARDRAILS_CONFIGURATION 區段的 Enable-Control 參數中，輸入您複製的全域識別符。以雙引號輸入識別符，並以逗號分隔多個識別符。 8. 在 GUARDRAILS_CONFIGURATION 區	DevOps 工程師，一般 AWS

任務	Description	所需的技能
	段的 `OrganizationalUnitIds` 參數中，輸入您要啟用控制項的組織單位 ID，例如 `ou-1111-11111111` 。以雙引號輸入 ID，並以逗號分隔多個 IDs。如需如何擷取 OU IDs 的詳細資訊，請參閱檢視 OU 的詳細資訊。 9. 儲存並關閉 constants.py 檔案。如需更新 constants.py 檔案的範例，請參閱此模式的其他資訊一節。	

在管理帳戶中啟用控制項

任務	Description	所需的技能
擔任部署帳戶中的 IAM 角色。	在部署帳戶中，擔任具有在管理帳戶中部署 AWS CDK 堆疊許可的 IAM 角色。如需在 中擔任 IAM 角色的詳細資訊 AWS CLI，請參閱在 中使用 IAM 角色 AWS CLI。	DevOps 工程師，一般 AWS
啟動環境。	如果您使用的是 Linux 或 MacOS： 1. 輸入下列命令來建立虛擬環境。 `$ python3 -m venv .venv`	DevOps 工程師，一般 AWS

任務	Description	所需的技能

	2. 虛擬環境建立後，請輸入下列命令來啟用它。
	`$ source .venv/bin/activate`
	如果您使用的是 Windows：
	1. 輸入下列命令以啟用虛擬環境。
	`% .venv\Scripts\activate.bat`	

安裝相依性。	啟用虛擬環境後，請輸入下列命令來執行 install_deps.sh 指令碼。此指令碼會安裝必要的相依性。 `$ ./scripts/install_deps.sh`	DevOps 工程師、一般 AWS、Python

部署堆疊。	輸入下列命令來合成和部署 CloudFormation 堆疊。
	`$ npx cdk synth`
	`$ npx cdk deploy`
		DevOps 工程師、一般 AWS、Python

相關資源

AWS 文件

- 關於控制項 (AWS Control Tower 文件)
- 控制項程式庫 (AWS Control Tower 文件)

- AWS CDK 工具組命令 (AWS CDK 文件)
- 使用 Terraform 部署和管理 AWS Control Tower 控制項 (AWS 方案指引)

其他資源

- Python

其他資訊

範例 constants.py 檔案

以下是更新 constants.py 檔案的範例。此範例會啟用 AWS-GR_DISALLOW_CROSS_REGION_NETWORKING 控制項 （ 全域 ID：dvuaav61i5cnfazfelmvn9m6k) 和 AWS-GR_SUBNET_AUTO_ASSIGN_PUBLIC_IP_DISABLED 控制項 （ 全域 ID：50z1ot237wl8u1lv5ufau6qqo)。如需全域 IDs 的清單，請參閱 AWS Control Tower 文件中的所有全域識別碼。

```
ACCOUNT_ID = 111122223333
AWS_CONTROL_TOWER_REGION = us-east-2
ROLE_ARN = "arn:aws:iam::111122223333:role/CT-Controls-Role"
GUARDRAILS_CONFIGURATION = [
    {
        "Enable-Control": {
            "dvuaav61i5cnfazfelmvn9m6k": {  # AWS-GR_DISALLOW_CROSS_REGION_NETWORKING
                "Parameters": {
                    "ExemptedPrincipalArns": ["arn:aws:iam::111122223333:role/
RoleName"]
                },
                "Tags": [{"key": "Environment", "value": "Production"}]
            },
            ...
        },
        "OrganizationalUnitIds": ["ou-1111-11111111", "ou-2222-22222222"...],
    },
    {
        "Enable-Control": {
            "50z1ot237wl8u1lv5ufau6qqo",  # AWS-
GR_SUBNET_AUTO_ASSIGN_PUBLIC_IP_DISABLED
            ...
```

```
        },
        "OrganizationalUnitIds": ["ou-2222-22222222"...],
    },
]
```

IAM 政策

下列範例政策允許在將 AWS CDK 堆疊從部署帳戶部署至管理帳戶時，啟用或停用 AWS Control Tower 控制項所需的最低動作。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "controltower:EnableControl",
                "controltower:DisableControl",
                "controltower:GetControlOperation",
                "controltower:ListEnabledControls",
                "organizations:AttachPolicy",
                "organizations:CreatePolicy",
                "organizations:DeletePolicy",
                "organizations:DescribeOrganization",
                "organizations:DescribeOrganizationalUnit",
                "organizations:DetachPolicy",
                "organizations:ListAccounts",
                "organizations:ListAWSServiceAccessForOrganization",
                "organizations:ListChildren",
                "organizations:ListOrganizationalUnitsForParent",
                "organizations:ListParents",
                "organizations:ListPoliciesForTarget",
                "organizations:ListRoots",
                "organizations:UpdatePolicy",
                "ssm:GetParameters"
            ],
            "Resource": "*"
        }
    ]
}
```

信任政策

下列自訂信任政策允許部署帳戶中的特定 IAM 角色擔任管理帳戶中的 IAM 角色。取代以下項目：

- <DEPLOYMENT-ACCOUNT-ID> 是部署帳戶的 ID
- <DEPLOYMENT-ROLE-NAME> 是部署帳戶中的角色名稱，允許 在管理帳戶中擔任該角色

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::<DEPLOYMENT-ACCOUNT-ID>:role/<DEPLOYMENT-ROLE-
NAME>"
            },
            "Action": "sts:AssumeRole",
            "Condition": {}
        }
    ]
}
```

使用 Terraform 部署和管理 AWS Control Tower 控制項

Iker Reina Fuente 和 Ivan Girardi，Amazon Web Services

總結

此模式說明如何使用 AWS Control Tower 控制項、HashiCorp Terraform 和基礎設施作為程式碼 (IaC) 來實作和管理預防性、偵測性和主動性安全控制。控制項 （也稱為護欄) 是一種高階規則，可為您的整體 AWS Control Tower 環境提供持續的控管。例如，您可以使用控制項來要求記錄 ， AWS 帳戶 然後在發生特定安全相關事件時設定自動通知。

AWS Control Tower 可協助您實作預防性、偵測性和主動性控制，以控管您的 AWS 資源並監控多個資源的合規性 AWS 帳戶。每個控制項都會強制執行單一規則。在此模式中，您可以使用提供的 IaC 範本來指定要在環境中部署的控制項。

AWS Control Tower 控制項適用於整個組織單位 (OU)，而控制項會影響 OU AWS 帳戶 中的每個單位。因此，當使用者在您登陸區域的任何帳戶中執行任何動作時，該動作會受到管理 OU 的控制。

實作 AWS Control Tower 控制項有助於為您的 AWS 登陸區域建立強大的安全基礎。透過使用此模式透過 Terraform 將控制項部署為 IaC，您可以標準化登陸區域中的控制項，並更有效率地部署和管理它們。

若要將 AWS Control Tower 控制項部署為 IaC，您也可以使用 AWS Cloud Development Kit (AWS CDK) 而非 Terraform。如需詳細資訊，請參閱使用 和 部署和管理 AWS Control Tower 控制項 AWS CDKAWS CloudFormation。

目標對象

對於具有 AWS Control Tower、Terraform 和 經驗的使用者，建議使用此模式 AWS Organizations。

先決條件和限制

先決條件

- 作為 AWS Organizations 和 AWS Control Tower 登陸區域中的組織進行主動 AWS 帳戶 管理。如需說明，請參閱 AWS Control Tower 文件中的入門。
- AWS Command Line Interface (AWS CLI)，已安裝並設定。
- 管理帳戶中具有部署此模式許可的 AWS Identity and Access Management (IAM) 角色。如需必要許可和範例政策的詳細資訊，請參閱此模式額外資訊區段中 IAM 角色的最低權限許可。
- 在管理帳戶中擔任 IAM 角色的許可。

- Terraform CLI，已安裝 (Terraform 文件）。
- Terraform AWS Provider，已設定 (Terraform 文件）。
- Terraform 後端，已設定 (Terraform 文件）。

限制

- 對於 AWS Control Tower 控制項，此模式需要使用下列格式的全域識別符：

```
arn:<PARTITION>:controlcatalog:::control/<CONTROL_CATALOG_OPAQUE_ID>
```

此模式的先前版本使用不再支援的區域識別符。我們建議您從區域識別符遷移到全域識別符。全域識別符可協助您管理控制項，並擴展您可以使用的控制項數量。

> **Note**
>
> 在大多數情況下， 的值<PARTITION>為 aws。

產品版本

- AWS Control Tower 3.2 版或更新版本
- Terraform 1.5 版或更新版本
- Terraform AWS Provider 4.67 版或更新版本

Architecture

本節提供此解決方案的高階概觀，以及範例程式碼所建立的架構。下圖顯示部署在 OU 中各種帳戶的控制項。

AWS Control Tower 控制項會根據其行為及其指引進行分類。

控制行為有三種主要類型：

1. 預防性控制旨在防止動作發生。這些會在 中使用服務控制政策 SCPs) 或資源控制政策 RCPs) 實作 AWS Organizations。預防性控制的狀態為強制執行或未啟用。所有 都支援預防性控制 AWS 區域。

2. Detective 控制項旨在偵測發生的特定事件，並記錄動作 AWS CloudTrail。這些是使用 AWS Config 規則實作的。偵測性控制項的狀態為明確、違規或未啟用。Detective 控制項僅適用於 AWS 區域 支援的控制項 AWS Control Tower。
3. 主動控制掃描由 佈建的資源， AWS CloudFormation 並檢查這些資源是否符合您的公司政策和目標。不會佈建不合規的資源。這些會使用AWS CloudFormation 勾點實作。主動控制的狀態為 PASS、FAIL 或 SKIP。

控制指引是將每個控制項套用至 OUs 的建議實務。 AWS Control Tower 提供三種類型的指引：強制性、強烈建議和選擇性。控制項的指引與其行為無關。如需詳細資訊，請參閱控制行為和指引。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和區域的生命週期中管理資源。
- AWS Config 提供 中資源的詳細檢視 AWS 帳戶 及其設定方式。它可協助您識別資源彼此之間的關係，以及其組態如何隨著時間而改變。
- AWS Control Tower 可協助您設定和管理 AWS 多帳戶環境，並遵循規範最佳實務。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。

其他工具

- HashiCorp Terraform 是一種基礎設施即程式碼 (IaC) 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。

程式碼儲存庫

此模式的程式碼可在 GitHub 部署中使用，並使用 Terraform 儲存庫管理 AWS Control Tower 控制項。

最佳實務

- 用於部署此解決方案的 IAM 角色應遵循最低權限 (IAM 文件） 原則。
- 遵循AWS Control Tower 管理員的最佳實務 (AWS Control Tower 文件）。

史詩

在管理帳戶中啟用控制項

任務	Description	所需的技能
複製儲存庫。	在 bash shell 中，輸入下列命令。這會從 GitHub 使用 Terraform 儲存庫複製部署和管理 AWS Control Tower 控制項。 `git clone https://github.com/aws-samples/aws-control-tower-controls-terraform.git`	DevOps 工程師
編輯 Terraform 後端組態檔案。	1. 在複製的儲存庫中，開啟 backend.tf 檔案。 2. 編輯 檔案以設定 Terraform 後端組態。您在此檔案中定義的組態取決於您的環境。如需詳細資訊，請參閱後端組態 (Terraform 文件）。 3. 儲存並關閉 backend.tf 檔案。	DevOps 工程師，Terraform
編輯 Terraform 提供者組態檔案。	1. 在複製的儲存庫中，開啟 provider.tf 檔案。 2. 編輯 檔案以設定 Terraform 提供者組態。如需詳細資訊，請參閱提供者組態 (Terraform 文件）。將 AWS 區域 設定為可使用 AWS Control Tower API 的區域。	DevOps 工程師，Terraform

任務	Description	所需的技能
	3. 儲存並關閉 provider.tf 檔案。	

任務	Description	所需的技能
編輯組態檔案。	1. 在複製的儲存庫中，開啟 variables.tfvars 檔案。 2. 開啟 AWS Control Tower 文件中的所有全域識別碼。 3. 在 JSON 格式清單中，找到您要實作的控制項，然後複製其全域識別符（也稱為 {CONTROL_CATALOG_OPAQUE_ID} 值）。例如，AWS-GR_AUDIT_BUCKET_ENCRYPTION_ENABLED 控制項的全域識別符為 `k4izcjxhukijhajp6ks5mjxk` 。 4. 在 `controls`區段的 `control_names` 參數中，輸入您複製的全域識別碼。 5. 在 `controls`區段的 `organizational_unit_ids` 參數中，輸入您要啟用控制項的組織單位 ID，例如 `ou-1111-11111111` 。以雙引號輸入 ID，並以逗號分隔多個 IDs。如需如何擷取 OU IDs 的詳細資訊，請參閱檢視 OU 的詳細資訊。 6. 儲存並關閉 variables.tfvars 檔案。如需已更新 variables.tfvars 檔案的範例，請參閱此模式的其他資訊一節。	DevOps 工程師、一般 AWS、Terraform

任務	Description	所需的技能
擔任管理帳戶中的 IAM 角色。	在管理帳戶中，擔任具有部署 Terraform 組態檔案許可的 IAM 角色。如需所需許可和範例政策的詳細資訊，請參閱其他資訊區段中的 IAM 角色的最低權限許可。如需在 中擔任 IAM 角色的詳細資訊 AWS CLI，請參閱在 中使用 IAM 角色 AWS CLI。	DevOps 工程師，一般 AWS

部署組態檔案。	1. 輸入下列命令來初始化 Terraform。
	`$ terraform init -upgrade`
	2. 輸入下列命令，以預覽相較於目前狀態的變更。
	`$ terraform plan -var-file="variables.tfvars"`
	3. 檢閱 Terraform 計劃中的組態變更，並確認您想要在組織中實作這些變更。
	4. 輸入下列命令來部署資源。
	`$ terraform apply -var-file="variables.tfvars"`
		DevOps 工程師、一般 AWS、Terraform

（選用）在 AWS Control Tower 管理帳戶中停用控制項

任務	Description	所需的技能
執行 destroy 命令。	輸入下列命令以移除此模式部署的資源。 `$ terraform destroy -var-file="variables.tfvars"`	DevOps 工程師、一般 AWS、Terraform

疑難排解

問題	解決方案
`Error: creating ControlTower Control ValidationException: Guardrail <control ID> is already enabled on organizational unit <OU ID>` 錯誤	您嘗試啟用的控制項已在目標 OU 中啟用。如果使用者透過、透過 AWS Control Tower 或透過 手動啟用控制項 AWS 管理主控台，則可能會發生此錯誤 AWS Organizations。若要部署 Terraform 組態檔案，您可以使用下列其中一個選項。 選項 1：更新 Terraform 目前狀態檔案 您可以將資源匯入至 Terraform 目前狀態檔案。當您重新執行`apply`命令時，Terraform 會略過此資源。執行下列動作，將資源匯入至目前的 Terraform 狀態： 1. 在 AWS Control Tower 管理帳戶中，輸入下列命令來擷取 OUs 的 Amazon Resource Name (ARNs) 清單，其中 `<root-ID>` 是組織根。如需擷取此 ID 的詳細資訊，請參閱檢視根的詳細資訊。

問題	解決方案

	`aws organizations list-organizational-units-for-parent --parent-id <root-ID>`
	2. 針對上一個步驟傳回的每個 OU，輸入下列命令，其中 <OU-ARN>是 OU 的 ARN。
	`aws controltower list-enabled-controls --target-identifier <OU-ARN>`
	3. 複製 ARNs 並在所需的模組中執行 Terraform 匯入，使其包含在 Terraform 狀態中。如需說明，請參閱匯入 (Terraform 文件）。
	4. 在 Epics 區段中重複部署組態中的步驟。
	
	選項 2：停用控制項
	
	如果您在非生產環境中工作，您可以在 主控台中停用控制項。在 Epics 區段中重複部署組態中的步驟，以重新啟用它。此方法不建議用於生產環境，因為有一段時間會停用控制項。如果您想要在生產環境中使用此選項，您可以實作暫時控制，例如暫時套用 SCP AWS Organizations。

相關資源

AWS 文件

- 關於控制項 (AWS Control Tower 文件）
- 控制項程式庫 (AWS Control Tower 文件）
- 使用 和 AWS CDK （方案指引） 部署和管理 AWS Control Tower 控制項 AWS CloudFormationAWS

其他資源

- Terraform
- Terraform CLI 文件

其他資訊

範例 variables.tfvars 檔案

以下是已更新 variables.tfvars 檔案的範例。此範例會啟用 AWS-GR_ENCRYPTED_VOLUMES 控制項（全域 ID：503uicglhjkokaajywfpt6ros) 和 AWS-GR_SUBNET_AUTO_ASSIGN_PUBLIC_IP_DISABLED 控制項（全域 ID：50z1ot237wl8u1lv5ufau6qqo)。如需全域 IDs 的清單，請參閱 AWS Control Tower 文件中的所有全域識別碼。

下列範例也會啟用具有 CT.S3.PV.5（全域 ID：7mo7a2h2ebsq71l8k6uzr96ou) 和 CT.SECRETSMANAGER.PV.1（全域 ID：）等參數的控制項。 dvhe47fxg5o6lryqrq9g6sxg4如需具有參數的控制項清單，請參閱 AWS Control Tower 文件中的具有參數的控制項。

```
controls = [
    {
        control_names = [
            "503uicglhjkokaajywfpt6ros", # AWS-GR_ENCRYPTED_VOLUMES
            ...
        ],
        organizational_unit_ids = ["ou-1111-11111111", "ou-2222-22222222"...],
    },
    {
        control_names = [
            "50z1ot237wl8u1lv5ufau6qqo", # AWS-GR_SUBNET_AUTO_ASSIGN_PUBLIC_IP_DISABLED
            ...
        ],
        organizational_unit_ids = ["ou-1111-11111111"...],
    },
]

controls_with_params = [
  {
    control_names = [
      { "7mo7a2h2ebsq71l8k6uzr96ou" = { # CT.S3.PV.5
        parameters = {
          "ExemptedPrincipalArns" : ["arn:aws:iam::*:role/RoleName"],
          "ExemptedResourceArns" : [],
```

```
        }
      } },
      { "dvhe47fxg5o6lryqrq9g6sxg4" = { # CT.SECRETSMANAGER.PV.1
        parameters = {
          "ExemptedPrincipalArns" : ["arn:aws:iam::*:role/RoleName"],
        }
      } },
      ...
    ],
    organizational_unit_ids = ["ou-1111-11111111"...]
  },
  {
    control_names = [
      { "dvuaav61i5cnfazfelmvn9m6k" = { # AWS-GR_DISALLOW_CROSS_REGION_NETWORKING
        parameters = {
          "ExemptedPrincipalArns" : ["arn:aws:iam::*:role/RoleName"],
        }
      } },
      { "41ngl8m5c4eb1myoz0t707n7h" = { # AWS-GR_DISALLOW_VPC_INTERNET_ACCESS
        parameters = {
          "ExemptedPrincipalArns" : ["arn:aws:iam::*:role/RoleName"],
        }
      } },
      ...
    ],
    organizational_unit_ids = ["ou-2222-22222222"...]
  }
]
```

IAM 角色的最低權限許可

此模式需要您在管理帳戶中擔任 IAM 角色。最佳實務是擔任具有暫時許可的角色，並根據最低權限原則限制許可。下列範例政策允許啟用或停用 AWS Control Tower 控制項所需的最低動作。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "controltower:EnableControl",
                "controltower:DisableControl",
                "controltower:GetControlOperation",
```

```
                "controltower:ListEnabledControls",
                "organizations:AttachPolicy",
                "organizations:CreatePolicy",
                "organizations:DeletePolicy",
                "organizations:DescribeOrganization",
                "organizations:DetachPolicy",
                "organizations:ListAccounts",
                "organizations:ListAWSServiceAccessForOrganization",
                "organizations:ListChildren",
                "organizations:ListOrganizationalUnitsForParent",
                "organizations:ListParents",
                "organizations:ListPoliciesForTarget",
                "organizations:ListRoots",
                "organizations:UpdatePolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

使用 Terraform 部署 AWS WAF 解決方案的安全自動化

Rahul Sharad Gaikwad 醫生和 Tamilselvan P，Amazon Web Services

總結

AWS WAF 是一種 Web 應用程式防火牆，可讓您使用自訂規則，在 Web 存取控制清單 (ACLs) 中定義和部署，協助保護應用程式免受常見入侵。設定 AWS WAF 規則可能具有挑戰性，尤其是對於沒有專用安全團隊的組織。為了簡化此程序，Amazon Web Services (AWS) 提供適用於 解決方案的安全自動化 AWS WAF，其會自動部署具有一組 AWS WAF 規則的單一 Web ACL，以篩選 Web 型攻擊。在 Terraform 部署期間，您可以指定要包含哪些保護功能。部署此解決方案之後， 會 AWS WAF 檢查現有 Amazon CloudFront 分佈或 Application Load Balancer 的 Web 請求，並封鎖不符合規則的任何請求。

解決方案的 安全自動化 AWS WAF 可以 AWS CloudFormation 根據 安全自動化 AWS WAF 實作指南中的說明使用 部署。此模式為使用 HashiCorp Terraform 作為其偏好基礎設施作為程式碼 (IaC) 工具來佈建和管理其雲端基礎設施的組織提供了替代部署選項。當您部署此解決方案時，Terraform 會自動套用雲端中的變更，並部署和設定 AWS WAF 設定和保護功能。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- AWS Command Line Interface (AWS CLI) 2.4.25 版或更新版本，已安裝並設定必要的許可。如需詳細資訊，請參閱入門 (AWS CLI 文件）。
- 已安裝並設定 Terraform 1.1.9 版或更新版本。如需詳細資訊，請參閱安裝 Terraform (Terraform 文件）。

Architecture

目標架構

此模式會部署 AWS WAF 解決方案的安全自動化。如需目標架構的詳細資訊，請參閱《 安全自動化 AWS WAF 實作指南》中的架構概觀。如需此部署中 AWS Lambda 自動化、應用程式日誌剖析器、AWS WAF 日誌剖析器、IP 清單剖析器和存取處理常式的詳細資訊，請參閱《 安全自動化 AWS WAF 實作指南》中的元件詳細資訊。

Terraform 部署

當您執行 時`terraform apply`，Terraform 會執行下列動作：

1. Terraform 會根據 testing.tfvars 檔案的輸入建立 AWS Identity and Access Management (IAM) 角色和 Lambda 函數。
2. Terraform 會根據 testing.tfvars 檔案的輸入建立 AWS WAF ACL 規則和 IP 集。
3. Terraform 會根據 testing.tfvars 檔案的輸入建立 Amazon Simple Storage Service (Amazon S3) 儲存貯體、Amazon EventBridge 規則、 AWS Glue 資料庫資料表和 Amazon Athena 工作群組。
4. Terraform 部署 AWS CloudFormation 堆疊以佈建自訂資源。
5. Terraform 會根據 testing.tfvars 檔案的指定輸入建立 Amazon API Gateway 資源。

自動化和擴展

您可以使用此模式為多個 建立 AWS WAF 規則 AWS 區域 ， AWS 帳戶 並在整個 AWS 雲端 環境中部署適用於 AWS WAF 解決方案的安全自動化。

工具

AWS 服務

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- AWS WAF 是一種 Web 應用程式防火牆，可協助您監控轉送至受保護 Web 應用程式資源的 HTTP 和 HTTPS 請求。

其他服務

- Git 是一種開放原始碼的分散式版本控制系統。
- HashiCorp Terraform 是一種命令列界面應用程式，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。

程式碼儲存庫

此模式的程式碼可在 GitHub AWS WAF Automation Using Terraform 儲存庫中使用。

最佳實務

- 將靜態檔案放在不同的 Amazon S3 儲存貯體中。

- 避免硬式編碼變數。
- 限制自訂指令碼的使用。
- 採用命名慣例。

史詩

設定您的本機工作站

任務	Description	所需的技能
安裝 Git。	遵循入門 (Git 網站） 中的指示，在本機工作站上安裝 Git。	DevOps 工程師
複製儲存庫。	在本機工作站上，輸入下列命令來複製程式碼儲存庫： `git clone https://github.com/aws-samples/aws-waf-automation-terraform-samples.git`	DevOps 工程師
更新變數。	1. 輸入下列命令，導覽至複製的目錄： `cd terraform-aws-waf-automation` 2. 在任何文字編輯器中，開啟 testing.tfvars 檔案。 3. 更新 testing.tfvars 檔案中變數的值。 4. 儲存並關閉檔案。	DevOps 工程師

使用 Terraform 佈建目標架構

任務	Description	所需的技能
初始化 Terraform 組態。	輸入下列命令來初始化包含 Terraform 組態檔案的工作目錄： `terraform init`	DevOps 工程師
預覽 Terraform 計劃。	輸入以下命令。Terraform 會評估組態檔案，以判斷宣告資源的目標狀態。然後，它會比較目標狀態與目前狀態，並建立計劃： `terraform plan -var-file ="testing.tfvars"`	DevOps 工程師
驗證計劃。	檢閱計劃並確認其已在您的目標中設定所需的架構 AWS 帳戶。	DevOps 工程師
部署解決方案。	1. 輸入下列命令以套用計劃： `terraform apply -var-file="testing .tfvars"` 2. 輸入 `yes` 以確認。Terraform 會建立、更新或銷毀基礎設施，以達到組態檔案中宣告的目標狀態。如需序列的詳細資訊，請參閱此模式的架構區段中的 Terraform 部署。	DevOps 工程師

驗證和清除

任務	Description	所需的技能
驗證變更。	1. 在 Terraform 主控台中，確認輸出符合預期的結果。 2. 登入 AWS 管理主控台。 3. 確認 Terraform 主控台中的輸出已成功部署在您的 中 AWS 帳戶。	DevOps 工程師
（選用）清除基礎設施。	如果您想要移除此解決方案所做的所有資源和組態變更，請執行下列動作： 1. 在 Terraform 主控台中，輸入下列命令： `terraform destroy -var-file="testing.tfvars"` 2. 輸入 `yes` 以確認。	DevOps 工程師

疑難排解

問題	解決方案
`WAFV2 IPSet: WAFOptimisticLockException 錯誤`	如果您在執行 `terraform destroy`命令時 收到此錯誤，則必須手動刪除 IP 集。如需說明，請參閱刪除 IP 集 (AWS WAF 文件）。

相關資源

AWS 參考

- 適用於 AWS WAF 實作的安全自動化指南

- 的安全自動化 AWS WAF(AWS 解決方案程式庫）
- 安全自動化 AWS WAF 常見問答集

Terraform 參考

- Terraform 後端組態
- Terraform AWS 提供者 - 文件和使用
- Terraform AWS 提供者 (GitHub 儲存庫）

部署可同時偵測多個程式碼交付項目中安全問題的管道

Benjamin Morris、Tim Hahn、Sapeksh Madan、Dina Odum 和 Isaiah Schisler，Amazon Web Services

總結

Simple Code Scanning Pipeline (SCSP) 提供兩鍵式建立程式碼分析管道，以平行執行業界標準的開放原始碼安全工具。這可讓開發人員檢查程式碼的品質和安全性，而無需安裝工具，甚至了解如何執行它們。這可協助您減少程式碼交付項目中的漏洞和錯誤設定。它也可以減少您的組織安裝、研究和設定安全工具所花費的時間。

在 SCSP 之前，使用此特定工具套件掃描程式碼需要開發人員尋找、手動安裝和設定軟體分析工具。即使在本機安裝，all-in-one工具都需要設定 Docker 容器才能執行。不過，透過 SCSP，一組業界標準程式碼分析工具會自動在 中執行 AWS 雲端。透過此解決方案，您可以使用 Git 推送程式碼交付項目，然後您會收到視覺化輸出，其中包含安全檢查失敗的at-a-glance。

先決條件和限制

- 作用中 AWS 帳戶
- 您想要掃描安全性問題的一或多個程式碼交付項目
- AWS Command Line Interface (AWS CLI)，已安裝並設定
- Python 3.0 版或更新版本和 pip 9.0.3 版或更新版本，已安裝
- Git，已安裝
- 在本機工作站上安裝 git-remote-codecommit

Architecture

目標技術堆疊

- AWS CodeCommit 儲存庫
- AWS CodeBuild 專案
- AWS CodePipeline 管道
- Amazon Simple Storage Service (Amazon S3) 儲存貯體
- AWS CloudFormation 範本

目標架構

靜態程式碼分析的 SCSP 是 DevOps 專案，旨在提供可交付程式碼的安全意見回饋。

1. 在 中 AWS 管理主控台，登入目標 AWS 帳戶。確認您位於 AWS 區域 您要部署管道的 中。
2. 使用程式碼儲存庫中的 CloudFormation 範本來部署 SCSP 堆疊。這會建立新的 CodeCommit 儲存庫和 CodeBuild 專案。

> **Note**
>
> 做為替代部署選項，您可以在堆疊部署期間提供儲存庫的 Amazon Resource Name (ARN) 做為參數，以使用現有的 CodeCommit 儲存庫。

3. 將儲存庫複製到本機工作站，然後將任何檔案新增至複製儲存庫中的個別資料夾。
4. 使用 Git 將檔案新增、遞交和推送至 CodeCommit 儲存庫。
5. 推送至 CodeCommit 儲存庫會啟動 CodeBuild 任務。CodeBuild 專案使用安全工具掃描程式碼交付項目。
6. 檢閱管道的輸出。發現錯誤層級問題的安全工具會導致管道中的動作失敗。修正這些錯誤或將其隱藏為誤報。在 CodePipeline 中的動作詳細資訊或管道的 S3 儲存貯體中檢閱工具輸出的詳細資訊。S3

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和區域的生命週期中管理資源。
- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。

其他工具

如需 SCSP 用來掃描程式碼交付項目的完整工具清單，請參閱 GitHub 中的 SCSP 讀我檔案。

程式碼儲存庫

此模式的程式碼可在 GitHub 的簡易程式碼掃描管道 (SCSP) 儲存庫中使用。

史詩

部署 SCSP

任務	Description	所需的技能
建立 CloudFormation 堆疊。	1. 登入 AWS 管理主控台。 2. 在 主控台中，確認您位於您要部署解決方案的目標區域中。如需詳細資訊，請參閱選擇區域。 3. 選擇以下連結。這會在 CloudFormation 中開啟快速建立堆疊精靈。 https://console.aws.amazon.com/cloudformation/home?#/stacks/create/review?templateURL=https://proservetools.s3.amazonaws.com/cft/scsp-pipeline-stack.template.json&stackName=SimpleCodeScanPipeline 4. 在快速建立堆疊精靈上，檢閱堆疊的參數設定，並視需要針對您的使用案例進行任何修改。 5. 選擇 I acknowledge that AWS CloudFormation might create IAM resources (我知道 AWS CloudFormation 可能會建立 IAM 資源)，然後	AWS DevOps、AWS 管理員

任務	Description	所需的技能
	選擇 Create stack (建立堆疊)。 這會建立 CodeCommit 儲存庫、CodePipeline 管道、數個 CodeBuild 任務定義和 S3 儲存貯體。建置執行和掃描結果會複製到此儲存貯體。完全部署 CloudFormation 堆疊之後，SCSP 即可使用。	

使用管道

任務	Description	所需的技能
檢查掃描的結果。	1. 在 Amazon S3 主控台的儲存貯體中，選擇 simplecodescanpipeline-deleteresourcespipelinereso 儲存貯體。 2. 選擇 scan_results 目錄，然後選擇具有最新掃描日期戳記的資料夾。 3. 檢閱此資料夾中的日誌檔案，以檢閱管道中使用的安全工具偵測到的任何問題。發現錯誤層級問題的安全工具會導致管道中的`failed`動作。如果這些是誤報，則需要修正或禁止。	應用程式開發人員、AWS DevOps

任務	Description	所需的技能
	Note 您也可以在 CodePipeline 主控台的動作詳細資訊區段中，檢視工具輸出的詳細資訊（適用於通過和失敗的掃描）。	

疑難排解

問題	解決方案
未掃描 HashiCorp Terraform 或 AWS CloudFormation 檔案。	請確定 Terraform (.tf) 和 CloudFormation (.yml、.yaml 或 .json) 檔案放置在複製 CodeCommit 儲存庫的適當資料夾中。
`git clone` 命令正在失敗。	請確定您已安裝 `git-remote-codecommit` 且您的 CLI 有權存取具有讀取 CodeCommit 儲存庫許可的 AWS 登入資料。
並行錯誤，例如 `Project-level concurrent build limit cannot exceed the account-level concurrent build limit of 1 。`	在 CodePipeline 主控台中選擇發行變更按鈕，以重新執行管道。這是管道執行的前幾次中最常見的已知問題。

相關資源

提供有關 SCSP 專案的意見回饋。

其他資訊

常見問答集

SCSP 專案是否與自動化安全協助程式 (ASH) 相同？

否。當您想要使用容器執行程式碼掃描工具的 CLI 工具時，請使用 ASH。自動化安全協助程式 (ASH) 是一種工具，旨在降低新程式碼、基礎設施或 IAM 資源組態中發生安全違規的可能性。ASH 是一種命令列公用程式，可在本機執行。本機使用需要在系統上安裝和操作容器環境。

當您想要比 ASH 更簡單的設定管道時，請使用 SCSP。SCSP 不需要本機安裝。SCSP 旨在個別在管道中執行檢查，並依工具顯示結果。SCSP 也可透過設定 Docker 來避免許多額外負荷，而且與作業系統 (OS) 無關。

SCSP 僅適用於安全團隊嗎？

否，任何人都可以部署管道，以判斷其程式碼的哪些部分未通過安全檢查。例如，非安全使用者可以使用 SCSP 檢查程式碼，然後再與其安全團隊進行檢閱。

如果我使用其他類型的儲存庫，例如 GitLab、GitHub 或 Bitbucket，是否可以使用 SCSP？

您可以設定本機 git 儲存庫，以指向兩個不同的遠端儲存庫。例如，您可以複製現有的 GitLab 儲存庫、建立 SCSP 執行個體（視需要指定 CloudFormation、Terraform 和 AWS Config 規則開發套件 (AWS RDK) 資料夾），然後使用 `git remote add upstream <SCSPGitLink>` 來指向 SCSP CodeCommit 儲存庫的本機儲存庫。這可讓程式碼變更先傳送至 SCSP、經過驗證，然後在進行任何額外的更新以解決問題清單、推送至 GitLab、GitHub 或 Bitbucket 儲存庫之後。如需多個遠端的詳細資訊，請參閱推送遞交至其他 Git 儲存庫 (AWS 部落格文章）。

> **Note**
>
> 請注意偏離，例如避免透過 Web 介面進行變更。

貢獻和新增您自己的動作

SCSP 設定會維護為 GitHub 專案，其中包含 SCSP AWS Cloud Development Kit (AWS CDK) 應用程式的原始程式碼。若要將其他檢查新增至管道， AWS CDK 應用程式需要更新，然後合成或部署到管道將執行 AWS 帳戶 的目標。若要這樣做，請先複製 SCSP GitHub 專案，然後在 `lib` 資料夾中尋找堆疊定義檔案。

如果您想要新增其他檢查， AWS CDK 程式碼中的 StandardizedCodeBuildProject類別會讓新增動作變得非常直接。提供名稱、描述和 install 或 build命令。 會使用合理的預設值來 AWS CDK 建立 CodeBuild 專案。除了建立建置專案之外，您還需要在建置階段將其新增至 CodePipeline 動作。設計新的檢查時，FAIL如果掃描工具偵測到問題或無法執行， 動作應該是 。PASS 如果掃描工具未偵測到任何問題，則 動作應該是 。如需設定工具的範例，請檢閱 Bandit動作的程式碼。

如需預期輸入和輸出的詳細資訊，請參閱儲存庫文件。

如果您新增自訂動作，則需要使用 cdk deploy或 部署 SCSPcdk synth + CloudFormation deploy。這是因為快速建立堆疊 CloudFormation 範本是由儲存庫擁有者維護。

使用 部署公有子網路的偵測屬性型存取控制 AWS Config

Amazon Web Services，Alberto Menendez

摘要

分散式邊緣網路架構依賴於與虛擬私有雲端 (VPCs) 中的工作負載一起執行的網路邊緣安全性。相較於更常見、集中的方法，這提供了前所未有的可擴展性。雖然在工作負載帳戶中部署公有子網路可以帶來好處，但它也會帶來新的安全風險，因為它會增加攻擊面。我們建議您僅在這些 VPCs 的公有子網路中部署 Elastic Load Balancing 資源，例如 Application Load Balancer 或 NAT 閘道。在專用公有子網路中使用負載平衡器和 NAT 閘道，可協助您針對傳入和傳出流量實作精細的控制。

我們建議您同時實作預防性和偵測性控制，以限制可在公有子網路中部署的資源類型。如需使用屬性型存取控制 (ABAC) 部署公有子網路預防性控制的詳細資訊，請參閱部署公有子網路的預防性屬性型存取控制。雖然對大多數情況都有效，但這些預防性控制可能無法解決所有可能的使用案例。因此，此模式以 ABAC 方法為基礎，可協助您設定有關部署在公有子網路中不合規資源的提醒。解決方案會檢查彈性網路介面是否屬於公有子網路中不允許的資源。

為了達成此目的，此模式使用AWS Config 自訂規則和 ABAC。自訂規則會在建立或修改彈性網路界面時處理其組態。在高階，此規則會執行兩個動作來判斷網路界面是否合規：

1. 若要判斷網路界面是否在規則範圍內，規則會檢查子網路是否具有指出其為公有子網路的特定AWS標籤。例如，此標籤可能是 `IsPublicFacing=True`。
2. 如果網路界面部署在公有子網路中，則規則會檢查 AWS 服務 建立此資源的 。如果資源不是 Elastic Load Balancing 資源或 NAT 閘道，則會將該資源標記為不合規。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- AWS Config，在工作負載帳戶中設定
- 在工作負載帳戶中部署所需資源的許可
- 具有公有子網路的 VPC
- 正確套用標籤以識別目標公有子網路
- （選用）中的組織 AWS Organizations

- （選用）中央安全帳戶，其為 AWS Config 和 的委派管理員 AWS Security Hub CSPM

Architecture

目標架構

此圖展示了以下要點：

1. 部署或修改彈性網路界面資源 (`AWS::EC2::NetworkInterface`) 時， 會 AWS Config 擷取事件和組態。
2. AWS Config 會將此事件與用來評估組態的自訂規則相符。
3. 系統會叫用與此自訂規則相關聯的 AWS Lambda 函數。函數會評估資源並套用指定的邏輯，以判斷資源組態是 `COMPLIANT`、 `NON_COMPLIANT`或 `NOT_APPLICABLE`。
4. 如果資源判斷為 `NON_COMPLIANT`， 會透過 Amazon Simple Notification Service (Amazon SNS) AWS Config 傳送提醒。

> 注意：如果此帳戶是 中的成員帳戶 AWS Organizations，您可以透過 AWS Config 或 將合規資料傳送至中央安全帳戶 AWS Security Hub CSPM。

Lambda 函數評估邏輯

下圖顯示 Lambda 函數套用的邏輯，以評估彈性網路界面的合規性。

自動化和擴展

此模式是一種偵測性解決方案。您也可以使用修補規則來補充它，以自動解決任何不合規的資源。如需詳細資訊，請參閱使用 AWS Config 規則修復不合規資源。

您可以透過以下方式擴展此解決方案：

- 強制套用您建立的對應 AWS 標籤，以識別面向公有的子網路。如需詳細資訊，請參閱 AWS Organizations 文件中的標記政策。
- 設定將 AWS Config 自訂規則套用至組織中每個工作負載帳戶的中央安全帳戶。如需詳細資訊，請參閱大規模自動化組態合規 AWS (AWS 部落格文章）。

- AWS Config 與 整合， AWS Security Hub CSPM 以大規模擷取、集中和通知 。如需詳細資訊，請參閱 AWS Security Hub CSPM 文件中的設定 AWS Config。

工具

- AWS Config 提供 中資源的詳細檢視 AWS 帳戶 及其設定方式。它可協助您識別資源彼此之間的關係，以及其組態如何隨著時間而改變。
- Elastic Load Balancing 會將傳入的應用程式或網路流量分散到多個目標。例如，您可以將流量分散到一或多個可用區域中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、容器和 IP 地址。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。

最佳實務

如需開發自訂 AWS Config 規則的更多範例和最佳實務，請參閱 GitHub 上的官方AWS Config 規則儲存庫。

史詩

部署解決方案

任務	Description	所需的技能
建立 Lambda 函數。	1. 登入 AWS 管理主控台，然後開啟 AWS Lambda 主控台。 2. 在 Functions (函數) 頁面上，選擇 Create function (建立函數)。 3. 選取從頭開始撰寫。	一般 AWS

任務	Description	所需的技能
	4. 在基本資訊窗格中，針對函數名稱輸入名稱。 5. 針對執行時期，選擇 Python 3.12。 6. 將架構設定為 x86_64。 7. 選擇建立函數。 8. 選擇 程式碼 標籤。 9. 在檔案總管中，選擇 lambda_function.py。 10將此模式額外資訊區段中提供的範例程式碼貼到 lambda_function.py 索引標籤。自訂範本程式碼以識別`evaluate_change_notification_compliance` 函數中的任何自訂評估邏輯。 11選擇部署。	

任務	Description	所需的技能
將許可新增至 Lambda 函數的執行角色。	1. 在導覽視窗中，選擇函數。 2. 選擇您剛建立的函數。 3. 選擇 組態 ，然後選擇 許可 。 4. 選擇角色名稱以在 AWS Identity and Access Management (IAM) 主控台中開啟角色。 5. 在許可政策下，選擇新增許可，然後選擇建立內嵌政策。 6. 選擇 JSON。 7. 將下列政策貼到政策編輯器中。這可讓 Lambda 函數： • 取得子網路標籤的詳細資訊。 • 將合規結果傳回至 AWS Config。 `{ "Version": "2012-10-17", "Statement": [ { "Action": [ "config:PutEvaluations", "ec2:DescribeSubnets" ], "Resource": "*",`	一般 AWS

任務	Description	所需的技能
	<pre>"Effect": "Allow" }] }</pre> 8. 選擇下一步。 9. 輸入政策名稱，然後選擇 Create policy (建立政策)。	
擷取 Lambda 函數 Amazon Resource Name (ARN)。	1. 開啟 Lambda 主控台。 2. 在導覽視窗中，選擇函數。 3. 選擇您剛建立的函數。 4. 在函數概觀區段的函數 ARN 下，複製 值。	一般 AWS

任務	Description	所需的技能
建立 AWS Config 自訂規則。	1. 開啟 AWS Config 主控台。 2. 在 Rules (規則) 頁面，選擇 Add rule (新增規則)。 3. 在指定規則類型頁面上，選擇建立自訂 Lambda 規則，然後選擇下一步。 4. 在設定規則頁面上，執行下列動作： a. 輸入名稱和描述。 b. 針對AWS Lambda 函數 ARN，貼上您先前複製的 ARN。 c. 針對 觸發類型，選擇 組態有所變更時。 d. 針對變更範圍，選取資源。 e. 針對資源類型，選擇 AWS EC2 NetworkInterface。 f. 選擇下一步。 5. 在檢閱和建立頁面上，驗證您的規則，然後選擇儲存。	一般 AWS

任務	Description	所需的技能
設定通知。	1. 請遵循建立 Amazon SNS 主題中的指示，以建立 Amazon SNS 主題。 2. 請遵循訂閱 Amazon SNS 主題中的指示，設定接收 Amazon SNS 主題通知的端點。 3. 遵循 如何在 AWS 資源不合規時使用 來設定不合規資源的自訂 Amazon EventBridge 規則時收到通知 AWS Config。 EventBridge	一般 AWS

測試解決方案

任務	Description	所需的技能
建立合規資源。	1. 使用下列指示，在公有子網路中建立其中一個支援的 資源： • 建立 NAT 閘道 • Network Load Balancers 入門 • 建立 Application Load Balancer 2. 建立資源之後， AWS Config 自訂規則會評估與資源相關聯的彈性網路介面。它將這些網路介面標記為 `COMPLIANT` 。您可以 AWS Config 依照下列步驟檢視 中的資源：	一般 AWS

任務	Description	所需的技能
	a. 開啟 AWS Config 主控台。 b. 在規則頁面上，選擇您的規則。 c. 在規則詳細資訊頁面上，前往頁面底部。 d. 在範圍內的資源下，選取合規。確認您看到已建立之網路介面IDs。 e. 如需網路介面組態的詳細資訊，請選擇資源 ID。	

任務	Description	所需的技能
建立不合規的資源。	1. 使用下列指示在公有子網路中建立不合規的資源： • 啟動 Amazon EC2 執行個體 • 建立 Amazon Relational Database Service (Amazon RDS) 資料庫執行個體 • 建立 VPC 端點 2. 建立資源之後， AWS Config 自訂規則會評估與資源相關聯的彈性網路介面。它將這些網路介面標記為 `NON_COMPLIANT` 。您可以 AWS Config 依照下列步驟檢視 中的資源： a. 開啟 AWS Config 主控台。 b. 在規則頁面上，選擇您的規則。 c. 在規則詳細資訊頁面上，前往頁面底部。 d. 在範圍內的資源下，選取NonCompliant。確認您看到已建立之網路介面 IDs。 e. 如需網路介面組態的詳細資訊，請選擇資源 ID。 3. 確認您在 Amazon SNS 中設定的端點收到通知。	一般 AWS

任務	Description	所需的技能
建立不適用的資源。	1. 在私有子網路中，建立任何需要彈性網路界面的資源。 2. 建立資源之後， AWS Config 自訂規則會評估與資源相關聯的彈性網路介面。它將這些網路介面標記為 `NOT_APPLICABLE` 。這些資源不會在 AWS Config 主控台中顯示。	一般 AWS

相關資源

AWS 文件

- 設定 AWS Config
- AWS Config 自訂規則
- 的 ABAC AWS
- 部署公有子網路的預防性屬性型存取控制

其他 AWS 資源

- 大規模自動化組態合規 AWS
- 具有 Gateway Load Balancer 的分散式檢查架構

其他資訊

以下是為示範目的而提供的 Lambda 函數範例。

```
import boto3
import json
import os

# Init clients
config_client = boto3.client('config')
```

```
ec2_client = boto3.client('ec2')

def lambda_handler(event, context):

    # Init values
    compliance_value = 'NOT_APPLICABLE'
    invoking_event = json.loads(event['invokingEvent'])
    configuration_item = invoking_event['configurationItem']

    status = configuration_item['configurationItemStatus']
    eventLeftScope = event['eventLeftScope']

    # First check if the event configuration applies. Ex. resource event is not delete
    if (status == 'OK' or status == 'ResourceDiscovered') and not eventLeftScope:
        compliance_value = evaluate_change_notification_compliance(configuration_item)


    config_client.put_evaluations(
       Evaluations=[
           {
               'ComplianceResourceType': invoking_event['configurationItem']
['resourceType'],
               'ComplianceResourceId': invoking_event['configurationItem']
['resourceId'],
               'ComplianceType': compliance_value,
               'OrderingTimestamp': invoking_event['configurationItem']
['configurationItemCaptureTime']
           },
       ],
       ResultToken=event['resultToken'])

# Function with the logs to evaluate the resource
def evaluate_change_notification_compliance(configuration_item):
    is_in_scope = is_in_scope_subnet(configuration_item['configuration']['subnetId'])

    if (configuration_item['resourceType'] != 'AWS::EC2::NetworkInterface') or not
 is_in_scope:
        return 'NOT_APPLICABLE'

    else:
        alb_condition = configuration_item['configuration']['requesterId'] in ['amazon-
elb']
        nlb_condition = configuration_item['configuration']['interfaceType'] in
 ['network_load_balancer']
```

```
        nat_gateway_condition = configuration_item['configuration']['interfaceType'] in
 ['nat_gateway']

        if alb_condition or nlb_condition or nat_gateway_condition:
            return 'COMPLIANT'
    return 'NON_COMPLIANT'

# Function to check if elastic network interface is in public subnet
def is_in_scope_subnet(eni_subnet):

    subnet_description = ec2_client.describe_subnets(
        SubnetIds=[eni_subnet]
    )

    for subnet in subnet_description['Subnets']:
        for tag in subnet['Tags']:
            if tag['Key'] == os.environ.get('TAG_KEY') and tag['Value'] ==
 os.environ.get('TAG_VALUE'):
                return True

    return False
```

部署公有子網路的預防性屬性型存取控制

Joel Alfredo Nunez Gonzalez 和 Samuel Ortega Sancho，Amazon Web Services

總結

在集中式網路架構中，檢查和邊緣虛擬私有雲端 (VPCs) 會集中所有傳入和傳出流量，例如往返網際網路的流量。不過，這可能會產生瓶頸，或導致達到 AWS 服務配額的限制。與更常見、集中的方法相比，在其 VPCs中的工作負載中部署網路邊緣安全性提供了前所未有的可擴展性。這稱為分散式邊緣架構。

雖然在工作負載帳戶中部署公有子網路可以帶來好處，但它也會帶來新的安全風險，因為它會增加攻擊面。我們建議您僅在這些 VPCs 的公有子網路中部署 Elastic Load Balancing (ELB) 資源，例如 Application Load Balancer 或 NAT 閘道。在專用公有子網路中使用負載平衡器和 NAT 閘道，可協助您針對傳入和傳出流量實作精細的控制。

屬性型存取控制 (ABAC) 是根據使用者屬性建立精細許可的做法，例如部門、任務角色和團隊名稱。如需詳細資訊，請參閱 ABAC for AWS。ABAC 可以為工作負載帳戶中的公有子網路提供防護機制。這有助於應用程式團隊保持敏捷，而不會影響基礎設施的安全性。

此模式說明如何透過 AWS Organizations 中的服務控制政策 (SCP) 和 AWS Identity and Access Management (IAM) 中的政策來實作 ABAC，以協助保護公有子網路的安全。 AWS Organizations 您可以將 SCP 套用到組織的成員帳戶或組織單位 (OU)。這些 ABAC 政策允許使用者在目標子網路中部署 NAT 閘道，並防止他們部署其他 Amazon Elastic Compute Cloud (Amazon EC2) 資源，例如 EC2 執行個體和彈性網路介面。

先決條件和限制

先決條件

- AWS Organizations 中的組織
- 對 AWS Organizations 根帳戶的管理存取權
- 在組織中，用於測試 SCP 的作用中成員帳戶或 OU

限制

- 此解決方案中的 SCP 不會阻止使用服務連結角色的 AWS 服務在目標子網路中部署資源。這些服務的範例包括 Elastic Load Balancing (ELB)、Amazon Elastic Container Service (Amazon ECS) 和

Amazon Relational Database Service (Amazon RDS)。如需詳細資訊，請參閱 AWS Organizations 文件中的 SCP 對許可的影響。實作安全控制以偵測這些例外狀況。

Architecture

目標技術堆疊

- 套用至 AWS Organizations 中 AWS 帳戶或 OU 的 SCP
- 下列 IAM 角色：
  - `AutomationAdminRole` – 用於修改子網路標籤，並在實作 SCP 後建立 VPC 資源
  - `TestAdminRole` – 用來測試 SCP 是否阻止其他 IAM 主體執行為 預留的動作，包括具有管理存取權的主體 `AutomationAdminRole`

目標架構

1. 您可以在目標帳戶中建立 `AutomationAdminRole` IAM 角色。此角色具有管理聯網資源的許可。請注意此角色獨有的下列許可：
   - 此角色可以建立 VPCs和公有子網路。
   - 此角色可以修改目標子網路的標籤指派。
   - 此角色可以管理自己的許可。
2. 在 AWS Organizations 中，您將 SCP 套用到目標 AWS 帳戶或 OU。如需範例政策，請參閱此模式中的其他資訊。
3. CI/CD 管道中的使用者或工具可以擔任`AutomationAdminRole`角色，將`SubnetType`標籤套用至目標子網路。
4. 透過擔任其他 IAM 角色，組織中的授權 IAM 主體可以管理目標子網路中的 NAT 閘道，以及 AWS 帳戶中其他允許的聯網資源，例如路由表。使用 IAM 政策授予這些許可。如需詳細資訊，請參閱 Amazon VPC 的身分和存取管理。

自動化和擴展

為了協助保護公有子網路，必須套用對應的 AWS 標籤。套用 SCP 後，NAT 閘道是授權使用者可以在具有 `SubnetType:IFA`標籤的子網路中建立的唯一 Amazon EC2 資源類型。(IFA 表示面向網際網路的資產。) SCP 可防止建立其他 Amazon EC2 資源，例如執行個體和彈性網路介面。我們建議您使用

擔任該AutomationAdminRole角色的 CI/CD 管道來建立 VPC 資源，以便將這些標籤正確套用至公有子網路。

工具

AWS 服務

- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 AWS 帳戶合併到您建立並集中管理的組織。在 AWS Organizations 中，您可以實作服務控制政策 (SCPs)，這是一種可用來管理組織中許可的政策類型。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路類似於您在自己的資料中心內操作的傳統網路，具有使用可擴展的 AWS 基礎設施的優勢。

史詩

套用 SCP

任務	Description	所需的技能
建立測試管理員角色。	在目標 AWS 帳戶中建立名為 TestAdminRole 的 IAM 角色。將 AdministratorAccess AWS 受管 IAM 政策連接至新角色。如需說明，請參閱《IAM 文件》中的建立角色以將許可委派給 IAM 使用者。	AWS 管理員
建立自動化管理員角色。	1. 在目標 AWS 帳戶中建立名為 AutomationAdminRole 的 IAM 角色。 2. 將 AdministratorAccess AWS 受管 IAM 政策連接至新角色。	AWS 管理員

任務	Description	所需的技能
	以下是信任政策的範例，您可以用來測試帳戶中的角色111122223333 。 `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "AWS": [ "arn:aws:iam::111122223333:root" ] }, "Action": "sts:AssumeRole", "Condition": {} } ] }`	
建立並連接 SCP。	1. 使用其他資訊區段中提供的範例程式碼，建立安全控制政策。如需說明，請參閱 AWS Organizations 文件中的建立 SCP。 2. 將 SCP 連接至目標 AWS 帳戶或 OU。如需說明，請參閱 AWS Organizations 文件中的連接和分離服務控制政策。	AWS 管理員

測試 SCP

任務	Description	所需的技能
建立 VPC 或子網路。	1. 擔任目標 AWS 帳戶中`TestAdminRole` 的角色。 2. 嘗試在現有 VPC 中建立 VPC 或新的公有子網路。如需說明，請參閱 Amazon VPC 文件中的建立 VPC、子網路和其他 VPC 資源。您不應該能夠建立這些資源。 3. 擔任`AutomationAdminRole` 角色，然後重試上一個步驟。現在，您應該能夠建立聯網資源。	AWS 管理員
管理標籤。	1. 擔任目標 AWS 帳戶中`TestAdminRole` 的角色。 2. 將`SubnetType:IFA` 標籤新增至可用的公有子網路。您應該可以新增此標籤。如需如何透過 AWS Command Line Interface (AWS CLI) 新增標籤的說明，請參閱 AWS CLI 命令參考中的 create-tags。 3. 在不變更您的登入資料的情況下，嘗試修改指派給此子網路的`SubnetType:IFA` 標籤。您不應該能夠修改此標籤。	AWS 管理員

任務	Description	所需的技能
	4. 擔任AutomationAdminRole 角色，然後重試先前的步驟。此角色應該能夠新增和修改此標籤。	
在目標子網路中部署資源。	1. 擔任TestAdminRole 角色。 2. 對於具有 SubnetType:IFA 標籤的公有子網路，請嘗試建立 EC2 執行個體。如需說明，請參閱 Amazon EC2 文件中的啟動執行個體。在此子網路中，您應該無法建立、修改或刪除 NAT 閘道以外的任何 Amazon EC2 資源。 3. 在相同的子網路中，建立 NAT 閘道。如需說明，請參閱 Amazon VPC 文件中的建立 NAT 閘道。您應該能夠在此子網路中建立、修改或刪除 NAT 閘道。	AWS 管理員
管理 AutomationAdminRole 角色。	1. 擔任TestAdminRole 角色。 2. 嘗試修改AutomationAdminRole 角色。如需說明，請參閱 IAM 文件中的修改角色。您不應該能夠修改此角色。 3. 擔任AutomationAdminRole 角色，然後重試上一個步驟。現在，您應該能夠修改角色。	AWS 管理員

清除

任務	Description	所需的技能
清除已部署的資源。	1. 從 AWS 帳戶或 OU 分離 SCP。如需說明，請參閱 AWS Organizations 文件中的分離 SCP。 2. 刪除 SCP。如需說明，請參閱刪除 SCP (AWS Organizations 文件）。 3. 刪除`AutomationAdminRole` 角色和`TestAdminRole` 角色。如需說明，請參閱 IAM 文件中的刪除角色。 4. 刪除您為此解決方案建立的所有聯網資源，例如 VPCs 和子網路。	AWS 管理員

相關資源

AWS 文件

- 連接和分離 SCPs
- 建立、更新和刪除 SCPs
- 使用 AWS Config 部署公有子網路的偵測屬性型存取控制
- 偵測性控制
- 服務授權參考
- 標記 AWS 資源
- 什麼是 ABAC for AWS？

其他 AWS 參考

- 在 AWS Organizations 中使用服務控制政策保護用於授權的資源標籤 (AWS 部落格文章）

其他資訊

下列服務控制政策是您可以用來在組織中測試此方法的範例。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyVPCActions",
      "Effect": "Deny",
      "Action": [
        "ec2:CreateVPC",
        "ec2:CreateRoute",
        "ec2:CreateSubnet",
        "ec2:CreateInternetGateway",
        "ec2:DeleteVPC",
        "ec2:DeleteRoute",
        "ec2:DeleteSubnet",
        "ec2:DeleteInternetGateway"
      ],
      "Resource": [
        "arn:aws:ec2:*:*:*"
      ],
      "Condition": {
        "StringNotLike": {
          "aws:PrincipalARN": ["arn:aws:iam::*:role/AutomationAdminRole"]
        }
      }
    },
    {
      "Sid": "AllowNATGWOnIFASubnet",
      "Effect": "Deny",
      "NotAction": [
        "ec2:CreateNatGateway",
        "ec2:DeleteNatGateway"
      ],
      "Resource": [
        "arn:aws:ec2:*:*:subnet/*"
      ],
      "Condition": {
        "ForAnyValue:StringEqualsIfExists": {
          "aws:ResourceTag/SubnetType": "IFA"
        },
```

```
        "StringNotLike": {
          "aws:PrincipalARN": ["arn:aws:iam::*:role/AutomationAdminRole"]
        }
      }
    },
    {
      "Sid": "DenyChangesToAdminRole",
      "Effect": "Deny",
      "NotAction": [
        "iam:GetContextKeysForPrincipalPolicy",
        "iam:GetRole",
        "iam:GetRolePolicy",
        "iam:ListAttachedRolePolicies",
        "iam:ListInstanceProfilesForRole",
        "iam:ListRolePolicies",
        "iam:ListRoleTags"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AutomationAdminRole"
      ],
      "Condition": {
        "StringNotLike": {
          "aws:PrincipalARN": ["arn:aws:iam::*:role/AutomationAdminRole"]
        }
      }
    }
  ]
}
```

偵測具有即將到期 CA 憑證的 Amazon RDS 和 Aurora 資料庫執行個體

Stephen DiCato 和 Eugene Shifer，Amazon Web Services

總結

作為安全最佳實務，建議您加密應用程式伺服器和關聯式資料庫之間傳輸中的資料。您可以使用 SSL 或 TLS 來加密與資料庫（資料庫）執行個體或叢集的連線。這些通訊協定有助於在應用程式和資料庫之間提供機密性、完整性和真實性。資料庫使用由憑證授權單位 (CA) 發行的伺服器憑證，並用於執行伺服器身分驗證。SSL 或 TLS 透過驗證憑證的數位簽章並確保憑證未過期，來驗證憑證的真偽。

在 中 AWS 管理主控台，Amazon Relational Database Service (Amazon RDS) 和 Amazon Aurora 會提供需要憑證更新的資料庫執行個體通知。不過，若要檢查這些通知，您必須登入每個 ， AWS 帳戶並導覽至每個 中的服務主控台 AWS 區域。如果您需要評估在 中以組織身分管理的許多 之間的憑證有效性 AWS 帳戶 ，此任務會變得更加複雜AWS Organizations。

透過將基礎設施佈建為此模式中提供的程式碼 (IaC)，您可以偵測 AWS 帳戶 或 AWS 組織中所有 Amazon RDS 和 Aurora 資料庫執行個體的即將到期 CA 憑證。AWS CloudFormation 範本會佈建 AWS Config 規則、 AWS Lambda 函數和必要的許可。您可以將它部署到單一帳戶做為堆疊，也可以將它部署到整個 AWS 組織中做為堆疊集。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 如果您要部署到單一 AWS 帳戶：
  - 請確定您具有建立 CloudFormation 堆疊的許可。
  - 在目標帳戶中啟用 AWS Config 。
  - （選用）在目標帳戶中啟用 AWS Security Hub CSPM 。
- 如果您要部署到 AWS 組織：
  - 請確定您具有建立 CloudFormation 堆疊集的許可。
  - 透過 AWS Organizations 整合啟用 Security Hub CSPM。
  - 在您部署此解決方案的帳戶中啟用 AWS Config 。
  - 將 指定 AWS 帳戶 為 AWS Config 和 Security Hub CSPM 的委派管理員。

限制

- 如果您要部署到未啟用 Security Hub CSPM 的個別帳戶，您可以使用 AWS Config 來評估問題清單。
- 如果您要部署到沒有 AWS Config 和 Security Hub CSPM 委派管理員的組織，您必須登入個別成員帳戶才能檢視問題清單。
- 如果您使用 AWS Control Tower 來管理組織中的帳戶，請使用 Customizations for(CfCT) 在此模式中部署 IaC。 AWS Control Tower CfCT 使用 CloudFormation AWS Control Tower 主控台將從護欄建立組態偏離，並要求您重新註冊組織單位 (OUs) 或受管帳戶。
- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱服務端點和配額頁面，然後選擇服務的連結。

Architecture

部署至個別 AWS 帳戶

下列架構圖顯示資源在單一 AWS 中的部署 AWS 帳戶。其實作方式是直接透過 CloudFormation 主控台使用 CloudFormation 範本。如果啟用 Security Hub CSPM，您可以在 AWS Config 或 Security Hub CSPM 中檢視結果。如果 Security Hub CSPM 未啟用，您只能在 AWS Config 主控台中檢視結果。

圖表顯示下列步驟：

1. 您可以建立 CloudFormation 堆疊。這會部署 Lambda 函數和 AWS Config 規則。規則和函數都會設定在 AWS Config 和 日誌中發佈資源評估所需的 AWS Identity and Access Management (IAM) 許可。
2. 此 AWS Config 規則以偵測評估模式運作，每 24 小時執行一次。
3. Security Hub CSPM 會收到所有 AWS Config 問題清單。
4. 您可以根據帳戶的組態 AWS Config，在 Security Hub CSPM 或 中檢視問題清單。

部署到 AWS 組織

下圖顯示透過 AWS Organizations 和 管理的多個帳戶的憑證過期評估 AWS Control Tower。您可以透過 CfCT 部署 CloudFormation 範本。評估結果集中在委派管理員帳戶中的 Security Hub CSPM 中。圖表中描述的 AWS CodePipeline 工作流程顯示 CfCT 部署期間發生的背景步驟。

圖表顯示下列步驟：

1. 根據 CfCT 的組態，在管理帳戶中，您可以將 IaC 推送到 AWS CodeCommit 儲存庫，或將 IaC 的壓縮 (ZIP) 檔案上傳到 Amazon Simple Storage Service (Amazon S3) 儲存貯體。
2. CfCT 管道會解壓縮檔案、執行 cfn-nag (GitHub) 檢查，並將其部署為 CloudFormation 堆疊集。
3. 根據 CfCT 資訊清單檔案中指定的組態，CloudFormation StackSets 會將堆疊部署到個別帳戶或指定的 OUs。這會在目標帳戶中部署 Lambda 函數和 AWS Config 規則。規則和函數都會設定在 AWS Config 和 日誌中發佈資源評估所需的 IAM 許可。
4. 此 AWS Config 規則以偵測評估模式運作，每 24 小時執行一次。
5. AWS Config 將所有問題清單轉送至 Security Hub CSPM。
6. Security Hub CSPM 調查結果會在委派的管理員帳戶中彙總。
7. 您可以在委派管理員帳戶中檢視 Security Hub CSPM 中的調查結果。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和區域的生命週期進行管理。
- AWS Config 提供 中資源的詳細檢視 AWS 帳戶 及其設定方式。它可協助您識別資源彼此之間的關係，以及其組態如何隨著時間而改變。An AWS Config rule 會定義您的理想資源組態設定，並可 AWS Config 評估您的 AWS 資源是否符合規則中的條件。
- AWS Control Tower 可協助您設定和管理 AWS 多帳戶環境，並遵循規範最佳實務。自訂 AWS Control Tower (CfCT) 可協助您自訂 AWS Control Tower 登陸區域，並保持符合 AWS 最佳實務。自訂是使用 CloudFormation 範本和服務控制政策 (SCPs實作。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。
- AWS Security Hub CSPM 提供 中安全狀態的完整檢視 AWS。它還可協助您根據安全產業標準和最佳實務來檢查 AWS 環境。

其他工具

- Python 是一種一般用途的電腦程式設計語言。

程式碼儲存庫

此模式的程式碼可在具有即將到期 CA 憑證儲存庫的 GitHub Detect Amazon RDS 執行個體中使用。https://github.com/aws-samples/config-rds-ca-expiry

最佳實務

我們建議您遵守下列資源中的最佳實務：

- 使用 的組織單位最佳實務 AWS Organizations (AWS 雲端操作和遷移部落格)
- 在 AWS Control Tower 上使用 建立初始基礎的指導 AWS (AWS 解決方案程式庫)
- 建立和修改 AWS Control Tower 資源的指引 (AWS Control Tower 文件)
- CfCT 部署考量 (AWS Control Tower 文件)

史詩

檢閱解決方案和程式碼

任務	Description	所需的技能
決定您的部署策略。	檢閱解決方案和程式碼，以判斷如何將其部署到您的 AWS 環境。決定您要部署到單一帳戶或 AWS 組織。	應用程式擁有者，一般 AWS
複製儲存庫。	輸入下列命令以複製具有即將過期 CA 憑證儲存庫的 Detect Amazon RDS 執行個體。 `git clone https://github.com/aws-samples/config-rds-ca-expiry.git`	應用程式開發人員、應用程式擁有者
驗證 Python 版本。	1. 導覽至複製儲存庫中的最上層目錄。	應用程式開發人員、應用程式擁有者

任務	Description	所需的技能
	`cd config-rds-ca-expiry` 2. 開啟 config-rds-ca-expiry.yaml。 3. 在 `CertExpirationCheckLambdaFunction` 資源中，確認 Python 版本與您的目標相容 AWS 區域。根據預設，此函數會使用 Python 3.12。如需詳細資訊，請參閱 AWS Lambda 新增對 Python 3.12 的支援。如有必要，請更新 Python 版本。 4. 儲存並關閉 config-rds-ca-expiry.yaml。	

部署解決方案

任務	Description	所需的技能
部署 CloudFormation 範本。	將 CloudFormation 範本部署到您的 AWS 環境。執行以下任意一項： • 如果您要部署到單一 AWS 帳戶，請遵循建立堆疊中的指示。 • 如果您要部署到非 管理的組織 AWS Control Tower，請遵循建立堆疊集中的指示。	應用程式開發人員、AWS 管理員、一般 AWS

任務	Description	所需的技能
	• 如果您要部署到由 管理的組織 AWS Control Tower，請參閱建置您自己的自訂中的說明。	
驗證部署。	在 CloudFormation 主控台中，確認堆疊或堆疊集已成功部署。	AWS 管理員、應用程式擁有者

檢閱問題清單

任務	Description	所需的技能
檢視 AWS Config 規則調查結果。	在 Security Hub CSPM 中，執行下列動作以檢視個別問題清單： 1. 開啟 Security Hub CSPM 主控台。 2. 在導覽窗格中，選擇調查結果。 3. 在新增篩選條件方塊中，新增下列篩選條件： • 合規狀態為 FAILED • 標題為 rds-has-expiring-ca 4. 選擇套用。 在 Security Hub CSPM 中，執行下列動作以檢視依 分組的問題清單總數 AWS 帳戶： 1. 開啟 Security Hub CSPM 主控台。	AWS 管理員、AWS 系統管理員、雲端管理員

任務	Description	所需的技能
	2. 在導覽窗格中，選擇 Insights。 3. 選擇 Create insight (建立洞見)。 4. 選取洞見的分組屬性： a. 選擇搜尋方塊以顯示篩選條件選項。 b. 選擇 Group by (分組依據)。 c. 選取 AwsAccountId。 d. 選擇套用。 5. 在新增篩選條件方塊中，新增下列篩選條件： • 標題為 `rds-has-expiring-ca` • 合規狀態為 `FAILED` 6. 選擇 Create insight (建立洞見)。 7. 輸入 Insight 名稱，然後選擇建立 Insight。 在 中 AWS Config，若要檢視問題清單，請遵循 AWS Config 文件中檢視合規資訊和評估結果中的指示。	

疑難排解

問題	解決方案
CloudFormation 堆疊集建立或刪除失敗	部署 AWS Control Tower 時，它會強制執行必要的護欄，並取得 AWS Config 對彙總器和規則的控制。這包括防止透過 CloudFormation 進行任何直接變更。若要正確部署或移除此 CloudFormation 範本，包括所有相關資源，您必須使用 CfCT。
CfCT 無法刪除 CloudFormation 範本	如果在資訊清單檔案中進行必要的變更並移除範本檔案後CloudFormation 範本仍存在，請確認資訊清單檔案包含 `enable_stack_set_deletion` 參數，且值設定為 `false`。如需詳細資訊，請參閱 CfCT 文件中的刪除堆疊集。

相關資源

- 使用 SSL/TLS 加密與資料庫執行個體或叢集的連線 (Amazon RDS 文件)
- AWS Config 自訂規則 (AWS Config 文件)

使用 Step Functions 透過 IAM Access Analyzer 動態產生 IAM 政策

Thomas Scott、Koen van Blijderveen、Adil El Kanabi 和 Rafal Pawlaszek，Amazon Web Services

總結

最低權限是授予執行任務所需最低許可的安全性最佳實務。在已處於作用中狀態的 Amazon Web Services (AWS) 帳戶中實作最低權限存取可能具有挑戰性，因為您不想透過變更使用者許可來意外封鎖使用者執行其任務。您必須先了解帳戶使用者正在執行的動作和資源，才能實作 AWS Identity and Access Management (IAM) 政策變更。

此模式旨在協助您套用最低權限存取原則，而不會封鎖或降低團隊生產力。它說明如何使用 IAM Access Analyzer，並根據目前在帳戶中執行的動作，為您的角色 AWS Step Functions 動態產生up-to-date IAM 政策。新政策旨在允許目前的活動，但會移除任何不必要的提升權限。您可以定義允許和拒絕規則來自訂產生的政策，而解決方案會整合您的自訂規則。

此模式包含使用 AWS Cloud Development Kit (AWS CDK) 或 HashiCorp CDK for Terraform (CDKTF) 實作解決方案的選項。然後，您可以使用持續整合和持續交付 (CI/CD) 管道，將新政策與角色建立關聯。如果您有多帳戶架構，您可以在任何要為角色產生更新 IAM 政策的帳戶中部署此解決方案，以提高整個 AWS 雲端 環境的安全性。

先決條件和限制

先決條件

- 啟用 AWS CloudTrail 線索 AWS 帳戶 的作用中 。
- 下列項目的 IAM 許可：
  - 建立和部署 Step Functions 工作流程。如需詳細資訊，請參閱 的動作、資源和條件索引鍵 AWS Step Functions （步驟函數文件）。
  - 建立 AWS Lambda 函數。如需詳細資訊，請參閱執行角色和使用者許可 (Lambda 文件）。
  - 建立 IAM 角色。如需詳細資訊，請參閱建立角色以將許可委派給 IAM 使用者 (IAM 文件）。
- npm 已安裝。如需詳細資訊，請參閱下載並安裝 Node.js 和 npm (npm 文件）。
- 如果您要使用 AWS CDK （選項 1) 部署此解決方案：
  - AWS CDK 工具組，已安裝和設定。如需詳細資訊，請參閱安裝 AWS CDK(AWS CDK 文件）。
- 如果您要使用 CDKTF 部署此解決方案 （選項 2)：
  - CDKTF，已安裝並設定。如需詳細資訊，請參閱安裝 CDK for Terraform (CDKTF 文件）。
  - Terraform，已安裝並設定。如需詳細資訊，請參閱入門 (Terraform 文件）。

- AWS Command Line Interface (AWS CLI) 已在本機為您的 安裝和設定 AWS 帳戶。如需詳細資訊，請參閱安裝或更新最新版本的 AWS CLI(AWS CLI 文件）。

限制

- 此模式不會將新的 IAM 政策套用至角色。在此解決方案結束時，新的 IAM 政策會存放在 儲存 AWS CodeCommit 庫中。您可以使用 CI/CD 管道，將政策套用至帳戶中的角色。

Architecture

目標架構

1. 定期排程的 Amazon EventBridge 事件規則會啟動 Step Functions 工作流程。您可以將此再生排程定義為設定此解決方案的一部分。
2. 在 Step Functions 工作流程中，Lambda 函數會產生日期範圍，用於分析 CloudTrail 日誌中的帳戶活動。
3. 下一個工作流程步驟會呼叫 IAM Access Analyzer API 來開始產生政策。
4. IAM Access Analyzer 會使用您在設定期間指定之角色的 Amazon Resource Name (ARN)，分析 CloudTrail 日誌中指定日期速率內的活動。根據活動，IAM Access Analyzer 會產生 IAM 政策，僅允許角色在指定日期範圍內使用的動作和服務。當此步驟完成時，此步驟會產生任務 ID。
5. 下一個工作流程步驟會每 30 秒檢查一次任務 ID。偵測到任務 ID 時，此步驟會使用任務 ID 呼叫 IAM Access Analyzer API 並擷取新的 IAM 政策。IAM Access Analyzer 會以 JSON 檔案的形式傳回政策。
6. 下一個工作流程步驟會將 <IAM 角色名稱>/policy.json 檔案放入 Amazon Simple Storage Service (Amazon S3) 儲存貯體。您可以在設定此解決方案時定義此 S3 儲存貯體。
7. Amazon S3 事件通知會啟動 Lambda 函數。
8. Lambda 函數會從 S3 儲存貯體擷取政策、整合您在 allow.json 和 deny.json 檔案中定義的自訂規則，然後將更新的政策推送至 CodeCommit。您可以在設定此解決方案的過程中定義 CodeCommit 儲存庫、分支和資料夾路徑。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端 基礎設施。
- AWS CDK Toolkit 是命令列雲端開發套件，可協助您與 AWS Cloud Development Kit (AWS CDK) 應用程式互動。
- AWS CloudTrail 可協助您稽核 的控管、合規和營運風險 AWS 帳戶。
- AWS CodeCommit 是一種版本控制服務，可協助您私下存放和管理 Git 儲存庫，而無需管理您自己的來源控制系統。
- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。此模式使用 IAM Access Analyzer 功能來分析 CloudTrail 日誌，以識別 IAM 實體 （使用者或角色） 所使用的動作和服務，然後根據該活動產生 IAM 政策。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Step Functions 是一種無伺服器協同運作服務，可協助您結合 AWS Lambda 函數和其他 AWS 服務 來建置業務關鍵型應用程式。在此模式中，您會使用 Step Functions AWS 中的 SDK 服務整合，從工作流程呼叫服務 API 動作。

其他工具

- CDK for Terraform (CDKTF) 可協助您使用 Python 和 Typescript 等常見程式設計語言，將基礎設施定義為程式碼 (IaC)。
- Lerna 是一種建置系統，用於管理和發佈來自相同儲存庫的多個 JavaScript 或 TypeScript 套件。
- Node.js 是一種事件驅動的 JavaScript 執行期環境，旨在建置可擴展的網路應用程式。
- npm 是在 Node.js 環境中執行的軟體登錄檔，用於共用或借用套件和管理私有套件的部署。

程式碼儲存庫

此模式的程式碼可在 GitHub 自動化 IAM Access Analyzer 角色政策產生器儲存庫中取得。

史詩

準備部署

任務	Description	所需的技能
複製儲存庫。	下列命令會複製自動 IAM Access Analyzer 角色政策產生器 (GitHub) 儲存庫。 `git clone https://github.com/aws-samples/automated-iam-access-analyzer.git`	應用程式開發人員
安裝 Lerna。	下列命令會安裝 Lerna。 `npm i -g lerna`	應用程式開發人員

設定相依性。	下列命令會安裝儲存庫的相依性。
	`cd automated-iam-access-analyzer/`
	`npm install && npm run bootstrap`
		應用程式開發人員

建置程式碼。	下列命令會測試、建置和準備 Lambda 函數的 zip 套件。
	`npm run test:code`
	`npm run build:code`
	`npm run pack:code`
		應用程式開發人員

建置 建構。	下列命令會為 AWS CDK 和 CDKTF 建置基礎設施合成應用程式。	

任務	Description	所需的技能
	`npm run build:infra`	
設定任何自訂許可。	在複製儲存庫的儲存庫資料夾中，編輯 allow.json 和 denied.json 檔案，以定義角色的任何自訂許可。如果 allow.json 和 denied.json 檔案包含相同的許可，則會套用拒絕許可。	AWS 管理員、應用程式開發人員

選項 1 – 使用 部署解決方案 AWS CDK

任務	Description	所需的技能
部署 AWS CDK 堆疊。	下列命令會透過 部署基礎設施 AWS CloudFormation。定義下列參數： • `<NAME_OF_ROLE>` – 您要為其建立新政策之 IAM 角色的 ARN。 • `<TRAIL_ARN>` – 存放角色活動的 CloudTrail 追蹤 ARN。 • `<CRON_EXPRESSION_TO_RUN_SOLUTION>` – 定義政策再生排程的 Cron 表達式。Step Functions 工作流程會依此排程執行。 • `<TRAIL_LOOKBACK>` – 評估角色許可時要回顧追蹤的期間，以天為單位。	應用程式開發人員

任務	Description	所需的技能

	`cd infra/cdk`
	`cdk deploy –-parameters roleArn=<NAME_OF_ROLE> \`
	`–-parameters trailArn=<TRAIL_ARN> \`
	`--parameters schedule=<CRON_EXPRESSION_TO_RUN_SOLUTION> \`
	`[ --parameters trailLookBack=<TRAIL_LOOKBACK> ]`
	
	Note
	方括號表示選用參數。	

（選用）等待新政策。	如果追蹤不包含角色的合理歷史活動量，請等到您確信 IAM Access Analyzer 有足夠的記錄活動來產生準確的政策。如果角色在帳戶中已處於作用中狀態足夠一段時間，則可能不需要此等待期間。	AWS 管理員
手動檢閱產生的政策。	在您的 CodeCommit 儲存庫中，檢閱產生的 <ROLE_ARN>.json 檔案，確認允許和拒絕許可適合該角色。	AWS 管理員

選項 2 – 使用 CDKTF 部署解決方案

任務	Description	所需的技能
合成 Terraform 範本。	下列命令會合成 Terraform 範本。	應用程式開發人員

任務	Description	所需的技能
	`lerna exec cdktf synth --scope @aiaa/tfm`	

任務	Description	所需的技能

部署 Terraform 範本。	下列命令會導覽至包含 CDKTF 定義基礎設施的目錄。
	`cd infra/cdktf`
	下列命令會在目標中部署基礎設施 AWS 帳戶。定義下列參數：
	• `<account_ID>` – 目標帳戶的 ID。
	• `<region>` - 目標 AWS 區域。
	• `<selected_role_ARN>` – 您要為其建立新政策之 IAM 角色的 ARN。
	• `<trail_ARN>` – 存放角色活動的 CloudTrail 追蹤 ARN。
	• `<schedule_expression>` – 定義政策再生排程的 Cron 表達式。Step Functions 工作流程會依此排程執行。
	• `<trail_look_back>` – 評估角色許可時要回顧追蹤的期間，以天為單位。
	`TF_VAR_accountId=<account_ID> \ TF_VAR_region=<region> \ TF_VAR_roleArns=<selected_role_ARN> \`
		應用程式開發人員

任務	Description	所需的技能
	TF_VAR_trailArn=<trail_ARN> \ TF_VAR_schedule=<schedule_expression> \ [TF_VAR_trailLookBack=<trail_look_back>] \ cdktf deploy **Note** 方括號表示選用參數。	
（選用）等待新政策。	如果追蹤不包含角色的合理歷史活動量，請等到您確信 IAM Access Analyzer 有足夠的記錄活動來產生準確的政策。如果角色在帳戶中已處於作用中狀態足夠一段時間，則可能不需要此等待期間。	AWS 管理員
手動檢閱產生的政策。	在您的 CodeCommit 儲存庫中，檢閱產生的 <ROLE_ARN>.json 檔案，確認允許和拒絕許可適合該角色。	AWS 管理員

相關資源

AWS resources

- IAM Access Analyzer 端點和配額
- 設定 AWS CLI
- 開始使用 AWS CDK
- 最低權限許可

其他資源

- 適用於 Terraform 的 CDK (Terraform 網站)

使用 AWS CloudFormation 範本有條件地啟用 Amazon GuardDuty

Ram Kandaswamy，Amazon Web Services

總結

AWS CloudFormation是基礎設施即程式碼 (IaC) 工具，可協助您透過以範本為基礎的部署來管理 AWS 資源。CloudFormation 通常用於管理 AWS 資源。使用它來啟用 AWS 服務，例如 Amazon GuardDuty，可能會帶來獨特的挑戰。GuardDuty 是一種威脅偵測服務，可持續監控您的 AWS 帳戶 是否有惡意活動和未經授權的行為。與可多次建立的典型資源不同，GuardDuty 是一種服務，每個帳戶和 都需要啟用一次 AWS 區域。傳統 CloudFormation 條件僅支援靜態值比較，因此很難檢查 GuardDuty 等服務的目前狀態。如果您嘗試在已啟用的帳戶中透過 CloudFormation 啟用 GuardDuty，堆疊部署會失敗。這可能會為管理多帳戶環境的 DevOps 團隊帶來營運挑戰。

此模式引入了此挑戰的解決方案。它使用由 AWS Lambda函數支援的 CloudFormation 自訂資源來執行動態狀態檢查。條件式邏輯只會在尚未啟用的情況下啟用 GuardDuty。它使用堆疊輸出來記錄 GuardDuty 狀態以供未來參考。

透過遵循此模式，您可以自動化整個 AWS 基礎設施的 GuardDuty 部署，同時保持乾淨、可預測的 CloudFormation 堆疊操作。這種方法對以下組織特別有用：

- AWS 帳戶 透過 IaC 管理多個
- 大規模實作安全服務
- 需要等冪基礎設施部署
- 自動化安全服務部署

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 具有建立、更新和刪除 CloudFormation 堆疊許可的 AWS Identity and Access Management (IAM) 角色
- AWS Command Line Interface (AWS CLI)，已安裝並設定

限制

如果 AWS 帳戶 或 的 GuardDuty 已手動停用 AWS 區域，則此模式不會為該目標帳戶或區域啟用 GuardDuty。

Architecture

目標技術堆疊

模式使用 CloudFormation 做為基礎設施的程式碼 (IaC)。您可以使用由 Lambda 函數支援的 CloudFormation 自訂資源來實現動態服務啟用功能。

目標架構

下列高階架構圖顯示透過部署 CloudFormation 範本來啟用 GuardDuty 的程序：

1. 您可以部署 CloudFormation 範本來建立 CloudFormation 堆疊。
2. 堆疊會建立 IAM 角色和 Lambda 函數。
3. Lambda 函數會擔任 IAM 角色。
4. 如果目標上尚未啟用 GuardDuty AWS 帳戶，Lambda 函數會啟用它。

自動化和擴展

您可以使用 AWS CloudFormation StackSet 功能將此解決方案擴展到多個 AWS 帳戶 和 AWS 區域。如需詳細資訊，請參閱 CloudFormation 文件中的使用 AWS CloudFormation StackSets。

工具

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您 AWS 服務 透過命令列 shell 中的命令與 互動。
- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和區域的生命週期中管理資源。
- Amazon GuardDuty 是一項持續的安全監控服務，可分析和處理日誌，以識別您 AWS 環境中非預期和可能未經授權的活動。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。

史詩

建立 CloudFormation 範本並部署堆疊

任務	Description	所需的技能
將程式碼儲存在 Amazon S3 中。	1. 在此模式的其他資訊區段中複製 Python 程式碼。 2. 在文字編輯器中貼上程式碼。 3. 儲存檔案為 `index.py`。 4. 將 檔案上傳到 Amazon Simple Storage Service (Amazon S3) 儲存貯體。如需說明，請參閱 Amazon S3 文件中的上傳物件。Amazon S3	AWS DevOps
建立 CloudFormation 範本。	1. 開啟 CloudFormation 主控台。 2. 在左側導覽窗格中，選擇 Infrastructure Composer。 3. 如果您沒有看到空白畫布，請建立新的專案。 4. 將 AWS Lambda 函數拖放到畫布上。 5. 在範本檢視中，驗證 Lambda 函數和日誌群組是否存在。 6. 將 Lambda 函數`Runtime`的修改為最新版本的 Python。 7. 驗證 `Handler` 屬性的值為 `index.lambda_handler`.	AWS DevOps

任務	Description	所需的技能

	8. 將 CodeUri 屬性設定為上傳 Python 程式碼的 Amazon S3 位置。範例值為 `s3://amzn-s3-demo-bucket/key-name` 。
	9. 新增資源的Policies屬性。遵循安全最佳實務，提供允許 CloudFormation 和 GuardDuty 動作的最低權限存取許可。例如，您可以將 AWSLambda Execute 受管政策用於 日誌，並將 自訂`iam:CreateServiceLinkedRole` 用於 GuardDuty。
	10.導覽至範本檢視，將自訂資源定義新增至程式碼片段。
	`CheckResourceExist:`
	`  Type: 'Custom::LambdaCustomResource'`
	`  Properties:`
	`  ServiceToken: !GetAtt Function.Arn`
	11.將範本儲存為 `sample.yaml` 。	

任務	Description	所需的技能

建立 CloudFormation 堆疊。	1. 在 中 AWS CLI，輸入下列命令。這會使用 `sample.yaml` 檔案建立新的 CloudFormation 堆疊。如需詳細資訊，請參閱 CloudFormation 文件中的建立堆疊。
	`aws cloudformation create-stack \ --stack-name guardduty-cf-stack \ --template-body file://sample.yaml`
	2. 確認下列值出現在 中 AWS CLI，表示堆疊已成功建立。建立堆疊所需的時間可能有所不同。
	`"StackStatus": "CREATE_COMPLETE",`
		AWS DevOps

驗證是否已為 啟用 GuardDuty AWS 帳戶。	1. 登入 AWS 管理主控台 並開啟 GuardDuty 主控台。 2. 確認 GuardDuty 服務已啟用。	雲端管理員、AWS 管理員
設定其他帳戶或區域。	根據您的使用案例，使用 CloudFormation StackSet 功能將此解決方案擴展到多個 AWS 帳戶 和 AWS 區域。如需詳細資訊，請參閱 CloudFormation 文件中的使用 AWS CloudFormation StackSets。	雲端管理員、AWS 管理員

相關資源

參考

- AWS CloudFormation 文件
- AWS Lambda 資源類型參考
- CloudFormation 資源類型： AWS：IAM：：Role
- CloudFormation 資源類型： AWS：GuardDuty：：Detector
- 使用 擷取任何 AWS 服務屬性的四種方式 AWS CloudFormation （部落格文章）

教學課程和影片

- 使用 簡化您的基礎設施管理 AWS CloudFormation （教學課程）
- 使用 Amazon GuardDuty 和 AWS Security Hub CSPM 來保護多個帳戶 (AWS re：Invent 2020)
- 撰寫的最佳實務 AWS CloudFormation(AWS re：Invent 2019)
- 上的威脅偵測 AWS：Amazon GuardDuty 簡介 (AWS re：Inforce 2019)

其他資訊

Python 程式碼

```
import boto3
import os
import json
from botocore.exceptions import ClientError
import cfnresponse
guardduty=boto3.client('guardduty')
cfn=boto3.client('cloudformation')
def lambda_handler(event, context):
    print('Event: ', event)
    if 'RequestType' in event:
      if event['RequestType'] in ["Create","Update"]:
          enabled=False
          try:
            response=guardduty.list_detectors()
            if "DetectorIds" in response and len(response["DetectorIds"])>0:
              enabled="AlreadyEnabled"
            elif "DetectorIds" in response and len(response["DetectorIds"])==0:
```

```
            cfn_response=cfn.create_stack(
              StackName='guardduty-cfn-stack',
              TemplateBody='{ "AWSTemplateFormatVersion": "2010-09-09",
   "Description": "Guard duty creation template",    "Resources":
{ "IRWorkshopGuardDutyDetector": {  "Type": "AWS::GuardDuty::Detector",
"Properties": {   "Enable": true  }   } } }'
              )
            enabled="True"
        except Exception as e:
            print("Exception: ",e)
        responseData = {}
        responseData['status'] = enabled
        cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
"CustomResourcePhysicalID" )
    elif event['RequestType'] == "Delete":
        cfn_response=cfn.delete_stack(
                StackName='guardduty-cfn-stack')
        cfnresponse.send(event, context, cfnresponse.SUCCESS, {})
```

在 Amazon RDS for SQL Server 中啟用透明資料加密

Ranga Cherukuri，Amazon Web Services

總結

此模式說明如何在 SQL Server 的 Amazon Relational Database Service (Amazon RDS) 中實作透明資料加密 (TDE)，以加密靜態資料。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- Amazon RDS for SQL Server 資料庫執行個體

產品版本

Amazon RDS 目前支援下列 SQL Server 版本和版本的 TDE：

- SQL Server 2016 Enterprise Edition
- SQL Server 2017 Enterprise Edition
- SQL Server 2019 Standard 和 Enterprise Editions
- SQL Server 2022 Standard 和 Enterprise Edition

如需支援版本和版本的最新資訊，請參閱 Amazon RDS 文件中的 SQL Server 中對透明資料加密的支援。

Architecture

技術堆疊

- Amazon RDS for SQL Server

架構

工具

- Microsoft SQL Server Management Studio (SSMS) 是管理 SQL Server 基礎設施的整合環境。它提供使用者介面和一組工具，其中包含與 SQL Server 互動的豐富指令碼編輯器。

史詩

在 Amazon RDS 主控台中建立選項群組

任務	Description	所需的技能
開啟 Amazon RDS 主控台。	登入 AWS 管理主控台並開啟 Amazon RDS 主控台。	開發人員，DBA
建立選項群組。	在導覽窗格中，選擇選項群組、建立群組。選擇 sqlserver-ee 做為資料庫引擎，然後選取引擎版本。	開發人員，DBA
新增 TRANSPARENT_DATA_ENCRYPTION 選項。	編輯您建立的選項群組，並新增名為 的選項TRANSPARENT_DATA_ENCRYPTION 。	開發人員，DBA

將選項群組與資料庫執行個體關聯

任務	Description	所需的技能
選擇資料庫執行個體。	在 Amazon RDS 主控台的導覽窗格中，選擇資料庫，然後選擇您要與選項群組建立關聯的資料庫執行個體。	開發人員，DBA
將資料庫執行個體與選項群組建立關聯。	選擇修改，然後使用選項群組設定，將 SQL Server 資料庫執行個體與您先前建立的選項群組建立關聯。	開發人員，DBA

任務	Description	所需的技能
套用變更。	視需要立即或在下一個維護時段套用變更。	開發人員，DBA

取得憑證名稱。	使用下列查詢取得預設憑證名稱。
	`USE [master]`
	`GO`
	`SELECT name FROM sys.certificates WHERE name LIKE 'RDSTDECertificate%'`
	`GO`
		開發人員，DBA

建立資料庫加密金鑰

任務	Description	所需的技能
使用 SSMS 連線至 Amazon RDS for SQL Server 資料庫執行個體。	如需說明，請參閱 Microsoft 文件中的使用 SSMS。	開發人員，DBA

使用預設憑證建立資料庫加密金鑰。	使用您先前取得的預設憑證名稱來建立資料庫加密金鑰。使用下列 T-SQL 查詢來建立資料庫加密金鑰。您可以指定 AES_256 演算法，而不是 AES_128。
	`USE [Databasename]`
	`GO`
	`CREATE DATABASE ENCRYPTION KEY`
	`WITH ALGORITHM = AES_128`
	`ENCRYPTION BY SERVER CERTIFICATE [certificatename]`
		開發人員，DBA

任務	Description	所需的技能
	`GO`	

在資料庫上啟用加密。	使用下列 T-SQL 查詢來啟用資料庫加密。
	`ALTER DATABASE [Database Name]`
	`SET ENCRYPTION ON`
	`GO`
		開發人員，DBA

檢查加密的狀態。	使用下列 T-SQL 查詢來檢查加密狀態。 `SELECT DB_NAME(database_id) AS DatabaseName, encryption_state, percent_complete FROM sys.dm_database_encryption_keys`	開發人員，DBA

相關資源

- 支援 SQL Server 中的透明資料加密 (Amazon RDS 文件）
- 使用選項群組 (Amazon RDS 文件）
- 修改 Amazon RDS 資料庫執行個體 (Amazon RDS 文件）
- SQL Server 的透明資料加密 (Microsoft 文件）
- 使用 SSMS (Microsoft 文件）

監控和修復 AWS KMS 金鑰的排程刪除

Mikesh Khanal 和 Ramya Pulipaka，Amazon Web Services

總結

在 Amazon Web Services (AWS) 雲端上，刪除 AWS Key Management Services (AWS KMS) 金鑰可能會導致資料遺失。刪除會移除與 AWS KMS 金鑰相關聯的金鑰材料和所有中繼資料，而且無法復原。刪除 AWS KMS 金鑰後，您無法再解密在該 AWS KMS 金鑰下加密的資料，以便無法復原資料。

此模式會設定監控，並在應用程式或使用者排定刪除 AWS KMS 金鑰時收到通知。如果您收到通知，建議您取消刪除 AWS KMS 金鑰，並重新考慮刪除該金鑰的決定。模式使用 AWS Systems Manager 自動化 Runbook AWSConfigRemediation-CancelKeyDeletion 來協助取消刪除 AWS KMS 金鑰。

> **Note**
>
> 模式的 CloudFormation 範本必須部署在您想要監控 AWS KMS 金鑰刪除的所有 AWS 區域中。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 了解下列 AWS 服務：
  - Amazon EventBridge
  - AWS KMS
  - Amazon Simple Notification Service (Amazon SNS)
  - AWS Systems Manager

限制

- 任何解決方案的自訂都需要了解 AWS CloudFormation 範本和此模式中使用的 AWS 服務。
- 目前，此解決方案使用預設事件匯流排，可根據需求自訂。如需自訂事件匯流排的詳細資訊，請參閱 AWS 文件。

Architecture

目標技術堆疊

- Amazon EventBridge
- AWS KMS
- Amazon SNS
- AWS Systems Manager
- 使用下列項目進行自動化：
  - AWS 命令列界面 (AWS CLI) 或 AWS 開發套件
  - AWS CloudFormation 堆疊

目標架構

1. 已排程刪除 AWS KMS 金鑰。
2. EventBridge 規則會評估排程刪除事件。
3. EventBridge 規則會參與 Amazon SNS 主題。
4. EventBridge 規則會啟動 Systems Manager 自動化和 Runbook。
5. Runbook 會取消刪除。

自動化和擴展

CloudFormation 堆疊會部署此解決方案運作所需的所有必要資源和服務。模式可以在單一帳戶中獨立執行，或使用多個獨立帳戶或組織的 AWS CloudFormation StackSets 執行。

```
aws cloudformation create-stack --stack-name  <stack-name>\
    --template-body file://<Full-Path-of-file> \
    --parameters ParameterKey=,ParameterValue= \
    --capabilities CAPABILITY_NAMED_IAM
```

工具

工具

- AWS CloudFormation – AWS CloudFormation 是一項服務，可協助您建立模型和設定 Amazon Web Services 資源，以減少管理這些資源的時間，並有更多時間專注於在 AWS 上執行的應用程式。您可以使用 CloudFormation 範本在 AWS 區域的 AWS 帳戶中建立堆疊。範本說明您想要的所有 AWS 資源，而 CloudFormation 會為您佈建和設定這些資源。
- AWS CLI – AWS 命令列界面 (AWS CLI) 是一種開放原始碼工具，可讓您在命令列 Shell 中使用命令與 AWS 服務互動。
- Amazon EventBridge – Amazon EventBridge 是一種無伺服器事件匯流排服務，可將您的應用程式與來自各種來源的資料連線。EventBridge 會從您自己的應用程式和 AWS 服務提供即時資料串流，並將該資料路由到 AWS Lambda 等目標。EventBridge 可簡化建置事件驅動型架構的程序。
- AWS KMS – AWS Key Management Service (AWS KMS) 是一種受管服務，用於建立和控制 AWS KMS 金鑰，這是用來加密資料的加密金鑰。
- AWS SDKs – AWS 工具包含SDKs，可讓您以您選擇的程式設計語言在 AWS 上開發和管理應用程式。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一種受管服務，可將訊息從發佈者交付給訂閱者（也稱為生產者和消費者）。發佈者透過製作並傳送訊息到主題 (其為邏輯存取點和通訊管道) 與訂閱者進行非同步的通訊。
- AWS Systems Manager – AWS Systems Manager 是一種 AWS 服務，可用來檢視和控制 AWS 上的基礎設施。使用 Systems Manager 主控台，您可以自動化 AWS 資源的操作任務。Systems Manager 透過掃描您的受管執行個體並報告 (或採取修正動作) 其偵測的任何政策違規，協助您保持安全與合規。

Code

- 已連接專案的 `alerting_ct_logs.yaml` CloudFormation 範本。

史詩

準備 AWS 帳戶

任務	Description	所需的技能
安裝和設定 AWS CLI。	安裝 AWS CLI 第 2 版。然後設定身分的安全登入資料設定、預設輸出格式，以及 AWS	開發人員、安全工程師

任務	Description	所需的技能
	CLI 用來與 AWS 互動的預設 AWS 區域。 身分必須具有執行任務所需的許可。	

部署 AWS CloudFormation 範本

任務	Description	所需的技能
下載 CloudFormation 範本。	將附件下載至您電腦上的本機路徑，並解壓縮`alerting_ct_logs.yaml` 範本檔案。	開發人員、安全工程師
部署 範本。	在已設定 AWS 帳戶設定檔的終端機視窗中，執行下列命令。 `aws cloudformation create-stack --stack-name <stack_name> \ --capabilities <Value> \ --template-body file://<Full_Path> \ --parameters ParameterKey=DestinationEmailAddress,ParameterValue=<Value> \ ParameterKey=SNSTopicName,ParameterValue=<Value> \ ParameterKey=EnableRemediation ,ParameterValue=<Value> \ ParameterKey=AutomationAssumeRole,`	開發人員、安全工程師

任務	Description	所需的技能
	`ParameterValue=<Value>` 在下一個步驟中，輸入範本參數的值。	
完成範本參數。	輸入參數的必要值。 • `DestinationEmailAddress` – 排定刪除 AWS KMS 金鑰時接收提醒的電子郵件地址。 • `SNSTopicName` – Amazon SNS 主題的名稱。 • `EnableRemediation` – 使用 Systems Manager Runbook 取消排程的金鑰刪除。允許的值為 `true` 和 `false`。 • `AutomationAssumeRole` – 角色的 Amazon Resource Name (ARN)，允許 Systems Manager 自動化代表您執行動作。如需詳細資訊，請參閱 AWSConfigRemediation-CancelKeyDeletion 文件中的必要 IAM 許可一節。 • `Capabilities` – 若要讓 AWS CloudFormation 建立堆疊，您必須明確確認您的堆疊範本包含特定功能。	開發人員、安全工程師

確認訂閱

任務	Description	所需的技能
確認訂閱。	檢查您的電子郵件收件匣，然後在您從 Amazon SNS 收到的電子郵件訊息中選擇確認訂閱。Web 瀏覽器視窗將開啟並顯示訂閱確認和您的訂閱 ID。	開發人員、安全工程師

相關資源

參考

- 建立 AWS 服務的規則
- 建立 Amazon CloudWatch 警示，以偵測待刪除的 AWS KMS 金鑰使用情況

教學課程和影片

- 如何開始使用 Amazon EventBridge
- 深入了解 Amazon EventBridge (AWS Online Tech Talks)

AWS 研討會

- 使用 EventBridge 規則

其他資訊

下列程式碼提供延伸解決方案的範例，以監控和通知您任何 AWS 服務的任何變更。這些範例包括預先定義的模式和自訂模式。如需詳細資訊，請參閱 EventBridge 中的事件和事件模式。

```
EventPattern:
        source:
        - aws.kms
        detail-type:
        - AWS API Call via CloudTrail
        detail:
```

```
        eventSource:
        - kms.amazonaws.com
        eventName:
        - ScheduleKeyDeletion
```

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用 Security Hub CSPM 識別 中的公 AWS Organizations 有 Amazon S3 儲存貯體

Mourad Cherfaoui、Arun Chandapillai 和 Parag Nagwekar，Amazon Web Services

總結

此模式說明如何建置機制，以識別您 AWS Organizations 帳戶中的公有 Amazon Simple Storage Service (Amazon S3) 儲存貯體。此機制的運作方式是使用 AWS 中基礎安全最佳實務 (FSBP) 標準的控制項 AWS Security Hub CSPM 來監控 Amazon S3 儲存貯體。您可以使用 Amazon EventBridge 來處理 Security Hub CSPM 問題清單，然後將這些問題清單發佈至 Amazon Simple Notification Service (Amazon SNS) 主題。組織中的利益相關者可以訂閱主題，並立即收到有關調查結果的電子郵件通知。

根據預設，新的 Amazon S3 儲存貯體及其物件不允許公開存取。您可以在必須根據組織需求修改預設 Amazon S3 組態的情況下使用此模式。例如，這可能是您有一個 Amazon S3 儲存貯體託管公開網站或檔案的情況，網際網路上的每個人都必須能夠從 Amazon S3 儲存貯體讀取。

Security Hub CSPM 通常部署為中央服務，以合併所有安全性問題清單，包括與安全標準和合規要求相關的問題清單。您可以使用其他 來偵測公 AWS 服務 有 Amazon S3 儲存貯體，但此模式會使用具有最少組態的現有 Security Hub CSPM 部署。

先決條件和限制

先決條件

- 具有專用 Security Hub CSPM 管理員帳戶的 AWS 多帳戶設定
- Security Hub CSPM 和 AWS Config，在您要監控 AWS 區域 的 中啟用

> **Note**
>
> 如果您想要從單一彙總區域監控多個區域，則必須在 Security Hub CSPM 中啟用跨區域彙總。

- 存取和更新 Security Hub CSPM 管理員帳戶的使用者許可、對組織中所有 Amazon S3 儲存貯體的讀取存取權，以及關閉公有存取權的許可 （如果需要）

Architecture

下圖顯示使用 Security Hub CSPM 來識別公有 Amazon S3 儲存貯體的架構。

圖表顯示下列工作流程：

1. Security Hub CSPM 會使用 FSBP 安全標準的 S3.2 和 S3.3 控制項來監控所有 AWS Organizations 帳戶（包括管理員帳戶）中 Amazon S3 儲存貯體的組態，並在儲存貯體設定為公有時偵測問題清單。
2. Security Hub CSPM 管理員帳戶會從所有成員帳戶存取問題清單（包括 S3.2 和 S3.3 的問題清單）。
3. Security Hub CSPM 會自動將所有新調查結果和現有調查結果的所有更新作為 Security Hub CSPM 調查結果 - 匯入的事件傳送至 EventBridge。這包括來自管理員和成員帳戶的問題清單事件。
4. EventBridge 規則會篩選來自 S3.2 和 S3.3 的問題清單，其 為 `ComplianceStatus` `FAILED`，工作流程狀態為 `NEW`，而 `RecordState`為 `ACTIVE`。
5. 規則使用事件模式來識別事件，並在符合時將其傳送至 Amazon SNS 主題。
6. Amazon SNS 主題會將事件傳送給其訂閱者（例如透過電子郵件）。
7. 指定接收電子郵件通知的安全分析師會檢閱有問題的 Amazon S3 儲存貯體。
8. 如果儲存貯體已核准公開存取，安全分析師會將 Security Hub CSPM 中對應調查結果的工作流程狀態設定為 `SUPPRESSED`。否則，分析師會將狀態設定為 `NOTIFIED`。這可消除 Amazon S3 儲存貯體的未來通知，並減少通知雜訊。
9. 如果工作流程狀態設定為 `NOTIFIED`，安全分析師會與儲存貯體擁有者一起檢閱調查結果，以判斷公有存取是否合理且符合隱私權和資料保護要求。調查會導致移除儲存貯體的公有存取權或核准公有存取權。在後一種情況下，安全分析師會將工作流程狀態設定為 `SUPPRESSED`。

> **Note**
>
> 架構圖同時適用於單一區域和跨區域彙總部署。在圖表中的帳戶 A、B 和 C 中，如果啟用跨區域彙總，Security Hub CSPM 可以屬於與管理員帳戶相同的區域，也可以屬於不同的區域。

工具

- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。EventBridge 可從您自己的應用程式、軟體即服務 (SaaS) 應用程式和 提供即時資料串流

AWS 服務。如果資料符合使用者定義的規則，EventBridge 會將該資料路由到目標，例如 Amazon SNS 主題和 AWS Lambda 函數。

- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。訂閱者會收到發佈到所訂閱主題的所有訊息，且某一主題的所有訂閱者均會收到相同訊息。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Security Hub CSPM 提供 中安全狀態的完整檢視 AWS。Security Hub CSPM 也可協助您根據安全產業標準和最佳實務來檢查 AWS 環境。Security Hub CSPM 會從跨 AWS 帳戶、 服務和支援的第三方合作夥伴產品收集安全資料，然後協助分析安全趨勢並識別最高優先順序的安全問題。

史詩

設定 Security Hub CSPM 帳戶

任務	Description	所需的技能
在 AWS Organizations 帳戶中啟用 Security Hub CSPM。	若要在您要監控 Amazon S3 儲存貯體的組織帳戶中啟用 Security Hub CSPM，請參閱 Security Hub CSPM 文件中的指定 Security Hub CSPM 管理員帳戶（主控台）和管理屬於組織的成員帳戶。	AWS 管理員
（選用）啟用跨區域彙總。	如果您想要從單一區域監控多個區域中的 Amazon S3 儲存貯體，請設定跨區域彙總。	AWS 管理員
啟用 S3FSBP 安全標準的 S3.2 和 S3.3 控制項。	您必須為 S3FSBP 安全標準啟用 S3.2 和 S3.3 控制項。 1. 若要啟用 S3.2 控制項，請遵循 Security Hub CSPM 文件中【S3.2】 S3 儲存貯體應禁止公開讀取存取的指示。	AWS 管理員

任務	Description	所需的技能
	2. 若要啟用 S3.3 控制項，請遵循 Security Hub CSPM 文件中 【3】 S3 儲存貯體應禁止公有寫入存取的指示。	

設定環境

任務	Description	所需的技能
設定 Amazon SNS 主題和電子郵件訂閱。	1. 登入 AWS 管理主控台 並開啟 Amazon SNS 主控台。 2. 在導覽窗格中選擇 Topics (主題)，然後選擇 Create topic (建立主題)。 3. 針對類型，選擇標準。 4. 針對名稱，輸入主題的名稱（例如 public-s3-buckets)。 5. 請選擇建立主題。 6. 在主題的訂閱索引標籤上，選擇建立訂閱。 7. 關於通訊協定，請選擇電子郵件。 8. 針對端點，輸入將接收通知的電子郵件地址。您可以使用 AWS 管理員、IT 專業人員或 Infosec 專業人員的電子郵件地址。 9. 選擇建立訂閱。若要建立其他電子郵件訂閱，請視需要重複步驟 6–8。	AWS 管理員
設定 EventBridge 規則。	1. 開啟 EventBridge 主控台。	AWS 管理員

任務	Description	所需的技能
	2. 在入門區段中，選取 EventBridge 規則，然後選擇建立規則。 3. 在定義規則詳細資訊頁面上，針對名稱輸入規則的名稱 （例如 public-s3-buckets)。選擇下一步。 4. 在事件模式區段中，選擇編輯模式。 5. 複製下列程式碼，將它貼到事件模式程式碼編輯器，然後選擇下一步。 `{ "source": ["aws.securityhub"], "detail-type": ["Security Hub Findings - Imported"], "detail": { "findings": { "Compliance": { "Status": ["FAILED"] }, "RecordState": ["ACTIVE"], "Workflow": { "Status": ["NEW"] }, "ProductFields": { "ControlId": ["S3.2", "S3.3"] } }`	

任務	Description	所需的技能
	<pre> } }</pre> 6. 在選取目標 (Select target) 頁面上，針對選取目標，選取 SNS 主題做為目標，然後選取您先前建立的主題。 7. 選擇下一步，再次選擇下一步，然後選擇建立規則。	

疑難排解

問題	解決方案
我的 Amazon S3 儲存貯體已啟用公有存取，但我沒有收到電子郵件通知。	這可能是因為儲存貯體是在另一個區域中建立的，並且未在 Security Hub CSPM 管理員帳戶中啟用跨區域彙總。若要解決此問題，請在 Amazon S3 儲存貯體目前所在的區域中啟用跨區域彙總或實作此模式的解決方案。

相關資源

- 什麼是 AWS Security Hub CSPM？ (Security Hub CSPM 文件）
- AWS 基礎安全最佳實務 (FSBP) 標準 (Security Hub CSPM 文件）
- AWS Security Hub CSPM 多帳戶啟用指令碼 (AWS Labs)
- Amazon S3 的安全最佳實務 (Amazon S3 文件）

其他資訊

用於監控公有 Amazon S3 儲存貯體的工作流程

下列工作流程說明如何監控組織中的公有 Amazon S3 儲存貯體。工作流程假設您已完成此模式的設定 Amazon SNS 主題和電子郵件訂閱案例中的步驟。

1. 當 Amazon S3 儲存貯體設定為公開存取時，您會收到電子郵件通知。
   - 如果儲存貯SUPPRESSED體已核准公開存取，請在 Security Hub CSPM 管理員帳戶中將對應調查結果的工作流程狀態設為 。這可防止 Security Hub CSPM 為此儲存貯體發出進一步通知，並可以消除重複的提醒。
   - 如果儲存貯體未核准公開存取，請將 Security Hub CSPM 管理員帳戶中對應調查結果的工作流程狀態設定為 NOTIFIED。這可防止 Security Hub CSPM 從 Security Hub CSPM 發出此儲存貯體的進一步通知，並可以消除雜訊。
2. 如果儲存貯體可能包含敏感資料，請立即關閉公開存取，直到檢閱完成為止。如果您關閉公有存取，則 Security Hub CSPM 會將工作流程狀態變更為 RESOLVED。然後，儲存貯體的電子郵件通知會停止。
3. 尋找將儲存貯體設定為公有 （例如，使用 AWS CloudTrail) 並開始檢閱的使用者。檢閱會導致移除儲存貯體的公有存取權或核准公有存取權。如果已核准公開存取，請將對應調查結果的工作流程狀態設定為 SUPPRESSED。

在 Microsoft Sentinel 中擷取和分析 AWS 安全日誌

Ivan Girardi 和 Sebastian Wenzel，Amazon Web Services

總結

此模式說明如何將 AWS 安全日誌，例如 AWS CloudTrail 日誌、Amazon CloudWatch Logs 資料、Amazon VPC Flow Logs 資料和 Amazon GuardDuty 調查結果，自動擷取至 Microsoft Sentinel。如果您的組織使用 Microsoft Sentinel 做為安全資訊和事件管理 (SIEM) 系統，這可協助您集中監控和分析日誌，以偵測與安全相關的事件。一旦日誌可供使用，它們會在不到 5 分鐘內自動交付到 Amazon Simple Storage Service (Amazon S3) 儲存貯體。這可協助您快速偵測 AWS 環境中的安全事件。

Microsoft Sentinel 會以表格格式擷取 CloudTrail 日誌，其中包含記錄事件時的原始時間戳記。擷取日誌的結構可透過在 Microsoft Sentinel 中使用 Kusto 查詢語言來啟用查詢功能。

模式會部署監控和提醒解決方案，可在不到 1 分鐘內偵測擷取失敗。它還包含外部 SIEM 可以監控的通知系統。您可以使用 AWS CloudFormation 在記錄帳戶中部署所需的資源。

目標對象

建議具有 AWS Organizations CloudFormation AWS Control Tower、 AWS Identity and Access Management (IAM) 和 AWS Key Management Service () 相關經驗的使用者使用此模式AWS KMS。

先決條件和限制

先決條件

以下是部署此解決方案的先決條件：

- 在 中以組織身分管理 AWS 帳戶 的作用中 AWS Organizations ，是 AWS Control Tower 登陸區域的一部分。組織應包含用於記錄的專用帳戶。如需說明，請參閱 AWS Organizations 文件中的建立和設定組織。
- CloudTrail 追蹤會記錄整個組織的事件，並將日誌存放在記錄帳戶中的 Amazon S3 儲存貯體中。如需說明，請參閱建立組織的追蹤。
- 在記錄帳戶中，擔任具有下列許可之現有 IAM 角色的許可：
  - 部署在提供的 CloudFormation 範本中定義的資源。
  - 部署提供的 CloudFormation 範本。
  - 如果使用客戶受管 AWS KMS 金鑰加密日誌，請修改金鑰政策。

- AWS Command Line Interface (AWS CLI)，已安裝並設定。
- 訂閱使用 Microsoft Sentinel 的 Microsoft Azure 帳戶。
- 啟用和設定 Microsoft Sentinel。如需說明，請參閱 Microsoft Sentinel 文件中的啟用 Microsoft Sentinel 和初始功能和內容。
- 符合設定 Microsoft Sentinel S3 連接器的先決條件。

限制

- 此解決方案會將安全日誌從記錄帳戶中的 Amazon S3 儲存貯體轉送至 Microsoft Sentinel。未明確提供如何將日誌傳送至 Amazon S3 的說明。
- 此模式提供 AWS Control Tower 在登陸區域中部署的指示。不過， AWS Control Tower 不需要使用 。
- 此解決方案相容於 Amazon S3 記錄儲存貯體受到服務控制政策 (SCPs) 限制的環境，例如不允許在日誌存檔中變更 AWS Control Tower 已建立的 Amazon S3 儲存貯體的儲存貯體政策。
- 此模式提供轉送 CloudTrail 日誌的指示，但您可以調整此解決方案來傳送 Microsoft Sentinel 支援的其他日誌，例如來自 CloudWatch Logs、Amazon VPC Flow Logs 和 GuardDuty 的日誌。
- 這些指示使用 AWS CLI 部署 CloudFormation 範本，但您也可以使用 AWS 管理主控台。如需說明，請參閱使用 AWS CloudFormation 主控台。如果您使用 主控台部署堆疊，請將堆疊部署在 AWS 區域 與記錄儲存貯體相同的 中。
- 此解決方案會部署 Amazon Simple Queue Service (Amazon SQS) 佇列來傳送 Amazon S3 通知。佇列包含的訊息包含 Amazon S3 儲存貯體中上傳物件的路徑，而非實際資料。佇列使用 SSE-SQS 加密協助保護訊息的內容。如果您想要使用 SSE-KMS 加密 SQS 佇列，您可以使用客戶管理的 KMS 金鑰。如需詳細資訊，請參閱 Amazon SQS 中的靜態加密。

Architecture

本節提供範本程式碼建立之架構的高階概觀。下圖顯示日誌帳戶中部署的資源，以便將日誌從現有的 Amazon S3 儲存貯體擷取到 Microsoft Sentinel。

架構圖顯示下列資源互動：

1. 在記錄帳戶中，Microsoft Sentinel 透過 OpenID Connect (OIDC) 擔任 IAM 角色，以存取特定 Amazon S3 儲存貯體和 Amazon SQS 佇列中的日誌。
2. Amazon Simple Notification Service (Amazon SNS) 和 Amazon S3 AWS KMS 用於加密。

3. 每當 Amazon S3 收到新日誌時，就會傳送通知訊息至 Amazon SQS 佇列。
4. Microsoft Sentinel 會檢查 Amazon SQS 是否有新訊息。Amazon SQS 佇列使用 SSE-SQS 加密。訊息保留期設定為 14 天。
5. Microsoft Sentinel 會從 Amazon SQS 佇列提取訊息。訊息包含已上傳 Amazon S3 物件的路徑。Microsoft Sentinel 會將這些物件從 Amazon S3 儲存貯體擷取至 Microsoft Azure 帳戶。
6. CloudWatch 警示會監控 Amazon SQS 佇列。如果未在 5 分鐘內從 Amazon SQS 佇列接收和刪除訊息，則會啟動傳送電子郵件的 Amazon SNS 通知。

AWS Control Tower 可協助您設定基礎組織單位 (OU) 結構，並將 CloudTrail 日誌集中在記錄帳戶中。它還實作強制性 SCPs 來保護記錄儲存貯體。

我們已在 AWS Control Tower 登陸區域中提供目標架構，但這並非嚴格要求。在此圖中，管理帳戶中的資源反映了 的 AWS Control Tower 部署和 CloudTrail 追蹤，可記錄整個組織的事件。

此模式著重於日誌帳戶中資源的部署。如果存放在 AWS Control Tower 登陸區域中 Amazon S3 中的日誌使用客戶受管 KMS 金鑰加密，則您必須更新金鑰政策，以允許 Microsoft Sentinel 解密日誌。在 AWS Control Tower 登陸區域中，您可以從管理帳戶管理金鑰政策，這是建立金鑰的位置。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和區域的生命週期中管理資源。
- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您在其上執行的應用程式。
- AWS Control Tower 可協助您設定和管理 AWS 多帳戶環境，並遵循最佳實務。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。
- Amazon Simple Queue Service (Amazon SQS) 提供安全、耐用且可用的託管佇列，可協助您整合和分離分散式軟體系統和元件。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他工具

- Microsoft Sentinel 是一種雲端原生 SIEM 系統，可提供安全協同運作、自動化和回應 (SOAR)。

程式碼儲存庫

此模式的程式碼可在 GitHub 擷取中取得，並分析 Microsoft Sentinel 儲存庫中的 AWS 安全日誌。

最佳實務

- 遵循最低權限原則 (IAM 文件）。
- 遵循AWS Control Tower 管理員的最佳實務 (AWS Control Tower 文件）。
- 遵循AWS CloudFormation 最佳實務 (CloudFormation 文件）。
- 使用程式碼分析工具，例如 cfn_nag，掃描產生的 CloudFormation 範本。cfn_nag 工具透過搜尋模式來識別 CloudFormation 範本中的潛在安全問題。

史詩

將 Microsoft Sentinel 連線至 Amazon S3

任務	Description	所需的技能
準備 Microsoft Sentinel S3 連接器。	1. 在 Microsoft Sentinel 中，選擇資料連接器。 2. 從資料連接器圖庫中，選擇 Amazon Web Services S3。 **Note** 如果您沒有看到連接器，請從 Microsoft Sentinel 中的 Content Hub 安裝 Amazon Web Services 解決方案。	DevOps 工程師，一般 AWS

任務	Description	所需的技能
	3. 在連接器的詳細資訊窗格中，選擇開啟連接器頁面。 4. 在組態區段中，複製外部 ID。您稍後需要此 ID。	

部署 CloudFormation 堆疊

任務	Description	所需的技能
複製儲存庫。	在 bash shell 中，輸入下列命令。這會複製擷取和分析 Microsoft Sentinel 儲存庫中的 AWS 安全日誌。 `git clone https://github.com/aws-samples/ingest-and-analyze-aws-security-logs-in-microsoft-sentinel.git`	DevOps 工程師，一般 AWS
擔任記錄帳戶中的 IAM 角色。	在記錄帳戶中，擔任具有部署 CloudFormation 堆疊許可的 IAM 角色。如需在 中擔任 IAM 角色的詳細資訊 AWS CLI，請參閱在 中使用 IAM 角色 AWS CLI。	DevOps 工程師，一般 AWS
部署堆疊。	若要部署 CloudFormation 堆疊，請輸入下列命令： • `<Bucket name>` 是記錄 Amazon S3 儲存貯體的名稱。	DevOps 工程師，一般 AWS

任務	Description	所需的技能
	• <Sentinel external ID> 是 Microsoft Sentinel 中 Amazon S3 連接器的外部 ID。 • <Email address> 是您想要接收通知的有效電子郵件地址。 • <Customer managed key ARN> 是客戶受管 KMS 金鑰的 Amazon Resource Name (ARN)。只有在日誌使用客戶受管 KMS 金鑰加密時，才提供此參數。 • <Suffix> 是選用參數，可避免資源名稱衝突。 • <ARN for the OIDC provider> 如果 OIDC 提供者已存在，則為其 ARN。如果您未提供此參數，CloudFormation 會建立 OIDC 供應商。 ⚠ Important 如果使用 Microsoft Code Defender 監控 AWS 組織，則 Microsoft 的 OIDC 供應商已部署。您必須提供此參數和現有提供者的 ARN。	

任務	Description	所需的技能
	`aws cloudformation deploy --stack-name cloudtrail-sentinel-integration \ --no-fail-on-empty-changeset \ --template-file template.yml \ --capabilities CAPABILITY_IAM CAPABILITY_NAMED_IAM CAPABILITY_AUTO_EXPAND \ --parameter-overrides \ ControlTowerS3BucketName="<Bucket name>" \ AzureWorkspaceID="<Sentinel external ID>" \ EmailAddress="<Email address>" \ KMSKeyArn="<Customer managed key ARN>" \ Suffix="<Suffix to avoid name conflicts>" \ OIDCProviderArn="<ARN for the OIDC provider>"`	
複製輸出。	從 CloudFormation 堆疊的輸出中，複製 `SentinelRoleArn` 和 的值`SentinelSQS` 。您稍後使用這些值來完成 Microsoft Sentinel 中的組態。	DevOps 工程師，一般 AWS

任務	Description	所需的技能
修改金鑰政策。	如果您不是使用客戶受管 KMS 金鑰來加密 Amazon S3 儲存貯體中的日誌，則可以略過此步驟。 如果日誌使用客戶受管 KMS 金鑰加密，請修改金鑰政策以授予 Microsoft Sentinel 解密日誌的許可。金鑰政策範例如下。如果 KMS 金鑰位於另一個 中，則此範例政策允許跨帳戶存取 AWS 帳戶。 `{ "Version": "2012-10-17", "Id": "key-policy", "Statement": [ ... { "Sid": "Grant access to decrypt", "Effect": "Allow", "Principal": { "AWS": "<SentinelRoleArn>" }, "Action": "kms:Decrypt", "Resource": "<KeyArn>" } ] }`	DevOps 工程師，一般 AWS

在 Microsoft Sentinel 中設定連接器

任務	Description	所需的技能
完成 Microsoft Sentinel 中的組態。	1. 在 Microsoft Sentinel 中，選擇資料連接器。 2. 從資料連接器圖庫中，選擇 Amazon Web Services S3。 3. 在連接器的詳細資訊窗格中，選擇開啟連接器頁面。 4. 在組態區段中，執行下列動作： a. 在要新增的角色中，輸入您複製`SentinelRoleArn` 的值。 b. 在 SQS URL 中，輸入您複製`SentinelSQS` 的值。 c. 在目的地資料表清單中，選擇 `AWSCloudTrail` 。 5. 選擇 Add Connection (新增連線)。	DevOps 工程師
將 Amazon S3 事件通知傳送至 Amazon SQS。	遵循使用 Amazon S3 主控台啟用和設定事件通知中的指示，以設定 Amazon S3 記錄儲存貯體將事件通知傳送至 Amazon SQS 佇列。如果已為整個組織設定 CloudTrail，則此儲存貯體中的日誌具有字首 `<OrgID>/AWSLogs/<OrgID>/` ，其中 `<OrgID>`是	DevOps 工程師，一般 AWS

任務	Description	所需的技能
	組織 ID。如需詳細資訊，請參閱檢視組織的詳細資訊。	
確認日誌已擷取。	1. 等到 Microsoft Sentinel 中擷取日誌。這可能需要幾分鐘的時間。 2. 在 Microsoft Sentinel 中，開啟 Amazon S3 Data Connector 頁面，然後執行下列動作： • 確認 Amazon S3 Data Connector 狀態為 `Connected` 。 • 檢查資料接收圖形中的資料磁碟區。 如需檢查資料連接器活動的詳細資訊，請參閱 Microsoft 文件中的資料連接器。	DevOps 工程師

驗證解決方案

任務	Description	所需的技能
比較 CloudWatch 和 Sentinel 日誌。	在 的預設組態中 AWS Control Tower，CloudTrail 日誌會傳送至 Amazon CloudWatch，並存放在 AWS Control Tower 管理帳戶中。如需詳細資訊，請參閱在 中記錄和監控 AWS Control Tower。使用下列步驟來確認日誌會自動擷取至 Microsoft Sentinel： 1. 開啟 CloudWatch 主控台。	DevOps 工程師，一般 AWS

任務	Description	所需的技能

	2. 在導覽窗格中，選擇 Logs (日誌)，然後選擇 Logs Insights (日誌洞察)。
	3. 對於選取日誌群組 （選取），選取存放 CloudTrail 日誌的日誌群組，例如 `aws-controltower/CloudTrailLogs` 。
	4. 在查詢編輯器方塊中，輸入 `fields eventID`。
	5. 選擇 Run query (執行查詢)。
	6. 選擇匯出結果，然後選擇將資料表複製到剪貼簿 (CSV)。
	7. 將結果貼到文字編輯器。
	8. 變更輸出的格式，以便在 Microsoft Sentinel 查詢中使用。以下是使用 Kusto 查詢語言的範例：
	`AWSCloudTrail`
	`| where AwsEventId in ('aa08b5fe-3bfb-391a-a14e-5fcebe14dab2', '9decd805-269c-451c-b75b-762f5dce59f9')`
	9. 在 Microsoft Sentinel 中，開啟 Amazon S3 Data Connector 頁面。在接收到的資料圖表旁，選擇前往記錄分析。	

任務	Description	所需的技能
	10.在查詢編輯器方塊中，輸入查詢，然後選擇執行。 11.在 Microsoft Sentinel 和 CloudWatch 中，確認項目數量相同。視需要調整時間範圍。	

相關資源

AWS 文件和資源

- AWS CLI 命令參考 (AWS CLI 文件）
- 選擇性設定 AWS KMS keys (AWS Control Tower 文件）
- Amazon SQS 中的靜態加密 (Amazon SQS 文件）
- 如何防止郵寄清單收件人取消訂閱 Amazon SNS 主題電子郵件清單中的每個人？ (AWS 知識中心）

Microsoft 文件

- 將 Microsoft Sentinel 連線至 Amazon Web Services 以擷取 AWS 服務日誌資料
- Microsoft Sentinel 中的 Kusto 查詢語言

使用 AWS CodePipeline 和 Amazon Bedrock 以程式碼形式管理 AWS Organizations 政策

Andre Cavalcante 和 Mariana Pessoa de Queiroz，Amazon Web Services

總結

您可以在 中使用授權政策 AWS Organizations 來集中設定和管理成員帳戶中主體和資源的存取權。服務控制政策 SCPs) 定義組織中 AWS Identity and Access Management (IAM) 角色和使用者的最大可用許可。資源控制政策 RCPs) 會定義組織中資源可用的最大許可。

此模式可協助您將 SCPs 和 RCPs 管理為透過持續整合和持續部署 (CI/CD) 管道部署的基礎設施即程式碼 (IaC)。透過使用 AWS CloudFormation 或 Hashicorp Terraform 來管理這些政策，您可以減輕與建立和維護多個授權政策相關的負擔。

此模式包含下列功能：

- 您可以使用資訊清單檔案 (`scp-management.json` 和) 建立、刪除和更新授權政策`rcp-management.json`。
- 您使用護欄而非政策。您可以在資訊清單檔案中定義護欄及其目標。
- 使用 AWS CodePipeline AWS CodeBuild 和 的管道會合併和最佳化資訊清單檔案中的護欄。對於資訊清單檔案中的每個陳述式，管道會將護欄合併為單一 SCP 或 RCP，然後將其套用至定義的目標。
- AWS Organizations 會將政策套用至您的目標。目標可以是 AWS 帳戶、組織單位 (OU)、環境 （這是您在 `environments.json` 檔案中定義的一組帳戶或 OUs)，或共用AWS 標籤的一組帳戶。
- Amazon Bedrock 會讀取管道日誌並摘要所有政策變更。
- 管道需要手動核准。核准者可以檢閱 Amazon Bedrock 準備的執行摘要，以協助他們了解變更。

先決條件和限制

先決條件

- 以組織形式管理 AWS 帳戶 的多個 AWS Organizations。如需詳細資訊，請參閱建立組織。
- SCP 和 RCP 功能已在 中啟用 AWS Organizations。如需詳細資訊，請參閱啟用政策類型。
- 已安裝 Terraform 1.9.8 版或更新版本。
- 如果您不是透過 Terraform 管道部署此解決方案，則 Terraform 狀態檔案必須存放在部署政策管理管道 AWS 帳戶 的 中的 Amazon Simple Storage Service (Amazon S3) 儲存貯體中。

- 已安裝 Python 3.13.3 版或更新版本。

限制

- 您無法使用此模式來管理在此 CI/CD 管道之外建立的 SCPs 或 RCPs。不過，您可以透過管道重新建立現有的政策。如需詳細資訊，請參閱此模式額外資訊區段中的將現有政策遷移至管道。
- 每個帳戶中的帳戶、OUs 和政策數量受 的配額和服務限制約束 AWS Organizations。
- 此模式無法用於在 中設定管理政策 AWS Organizations，例如備份政策、標籤政策、聊天應用程式政策或宣告政策。

Architecture

下圖顯示政策管理管道及其相關聯資源的工作流程。

該圖顯示以下工作流程：

1. 使用者將變更遞交至遠端儲存庫主分支中的 `scp-management.json`或 `rcp-management.json` 資訊清單檔案。
2. `main` 分支的變更會在其中啟動管道 AWS CodePipeline。
3. CodePipeline 會啟動 `Validate-Plan` CodeBuild 專案。此專案使用遠端儲存庫中的 Python 指令碼來驗證政策和政策資訊清單檔案。此 CodeBuild 專案會執行下列動作：
   a. 檢查 SCP 和 RCP 資訊清單檔案是否包含唯一的陳述式 IDs(`Sid`)。
   b. 使用 `scp-policy-processor/main.py`和 `rcp-policy-processor/main.py` Python 指令碼，將護欄資料夾中的護欄串連為單一 RCP 或 SCP 政策。它結合了具有相同 `Resource`、`Action`和 的護欄`Condition`。
   c. 使用 AWS Identity and Access Management Access Analyzer 驗證最終的最佳化政策。如果有任何問題清單，管道會停止。
   d. 建立 Terraform 用來建立資源的 `scps.json`和 `rcps.json` 檔案。
   e. 執行 `terraform plan`命令，這會建立 Terraform 執行計畫。
4. （選用） `Validate-Plan` CodeBuild 專案使用`bedrock-prompt/prompt.py`指令碼將提示傳送至 Amazon Bedrock。您可以在 `bedrock-prompt/prompt.txt` 檔案中定義提示。Amazon Bedrock 使用 Anthropic Claude Sonnet 3.5，透過分析 Terraform 和 Python 日誌來產生提議變更的摘要。

5. CodePipeline 使用 Amazon Simple Notification Service (Amazon SNS) 主題，以通知核准者必須檢閱變更。如果 Amazon Bedrock 產生變更摘要，通知會包含此摘要。
6. 政策核准者核准 CodePipeline 中的動作。如果 Amazon Bedrock 產生變更摘要，核准者可以在核准之前檢閱 CodePipeline 中的摘要。
7. CodePipeline 會啟動 `Apply` CodeBuild 專案。此專案使用 Terraform 來套用 RCP 和 SCP 變更 AWS Organizations。

與此架構相關聯的 IaC 範本也會部署支援政策管理管道的下列資源：

- 用於存放 CodePipeline 成品和指令碼的 Amazon S3 儲存貯體，例如 `scp-policy-processor/main.py`和 `bedrock-prompt/prompt.py`
- 加密此解決方案所建立資源的 AWS Key Management Service (AWS KMS) 金鑰

工具

AWS 服務

- Amazon Bedrock 是一項全受管 AI 服務，可透過統一 API 使用許多高效能的基礎模型。
- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。
- 適用於 Python (Boto3) 的 AWS SDK 是一種軟體開發套件，可協助您整合 Python 應用程式、程式庫或指令碼 AWS 服務。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

其他工具

- HashiCorp Terraform 是一種 IaC 工具，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。

程式碼儲存庫

此模式的程式碼可在 organizations-policy-pipeline GitHub 儲存庫中使用。以下是 sample-repository 資料夾中包含的金鑰檔案：

- 在 environments 資料夾中， environments.json包含環境清單。環境是一組目標，它們可以包含 AWS 帳戶 IDs或組織單位 OUs)。
- 在 rcp-management資料夾中：
  - guardrails 資料夾包含 RCPs 的個別護欄
  - policies 資料夾包含個別 RCPs
  - rcp-management.json 資訊清單檔案可協助您管理 RCP 護欄、完整 RCPs 及其相關聯的目標。
- 在 scp-management資料夾中：
  - guardrails 資料夾包含 SCPs 的個別護欄
  - policies 資料夾包含個別 SCPs
  - scp-management.json 資訊清單檔案可協助您管理 SCP 護欄、完整 SCPs 及其相關聯的目標。
- utils 資料夾包含的指令碼可協助您遷移目前的 SCPs 和 RCPs，以便您可以透過管道管理它們。如需詳細資訊，請參閱此模式的其他資訊一節。

最佳實務

- 在您設定管道之前，建議您確認尚未達到 AWS Organizations 配額的限制。
- 我們建議您僅將 AWS Organizations 管理帳戶用於必須在該帳戶中執行的任務。如需詳細資訊，請參閱 管理帳戶的最佳實務。

史詩

設定目標帳戶

任務	Description	所需的技能
建立 儲存庫。	建立安全操作團隊將從中管理政策的儲存庫。使用其中一個 AWS CodeConnections 支援的第三方儲存庫提供者。	DevOps 工程師

任務	Description	所需的技能
委派政策管理。	將 AWS Organizations 政策的管理委派給您要部署管道的成員帳戶。如需說明，請參閱使用 建立資源型委派政策 AWS Organizations。如需範例政策，請參閱此模式額外資訊區段中的以資源為基礎的委派政策範例。	AWS 管理員
（選用）啟用基礎模型。	如果您想要產生政策變更的摘要，請在 AWS 帳戶 您要部署管道的 中，啟用 Amazon Bedrock 中 Anthropic Claude 3.5 Sonnet 基礎模型的存取權。如需說明，請參閱新增或移除對 Amazon Bedrock 基礎模型的存取權。	一般 AWS

部署管道的資源

任務	Description	所需的技能
複製儲存庫。	輸入下列命令，從 GitHub 複製 organizations-policy-pipeline 儲存庫： `git clone https://github.com/aws-samples/organizations-policy-pipeline.git`	DevOps 工程師
定義您的部署方法。	1. 在複製的儲存庫中，開啟 `variables.tf` 檔案。	DevOps 工程師

任務	Description	所需的技能
	2. 針對 `project_name` ，輸入您要套用至已部署資源名稱的字首。 3. 針對 `provider_type` ，輸入遠端儲存庫的提供者。檔案中會提供有效值。 4. 針對 `full_repository_name` ，輸入遠端儲存庫的名稱。 5. 針對 `branch_name` ，輸入您要用來部署政策的 Git 分支名稱。此分支中的推送或合併會啟動管道。一般而言，這是主要分支。 6. 針對 `terraform_version` ，輸入您正在使用的 Terraform 版本。 7. 對於 `enable_bedrock` ，`true`如果您希望 Amazon Bedrock 摘要說明變更，請輸入 。`false` 如果您不想產生變更的摘要，請輸入 。 8. 針對 `tags`，輸入您要指派為已部署資源標籤的鍵/值對。 9. 儲存並關閉 `variables.tf` 檔案。	

任務	Description	所需的技能

部署管道。	1. 輸入下列命令來建立計劃並檢閱變更：
	`terraform plan`
	2. 輸入下列命令以套用計劃並建立管道基礎設施：
	`terraform apply`
		DevOps 工程師，Terraform

連接遠端儲存庫。	在上一個步驟中，Terraform 建立了與第三方儲存庫的 CodeConnections 連線。在AWS 開發人員工具主控台中，將連線的狀態從 變更為 `PENDING` `AVAILABLE` 。如需說明，請參閱更新待定連線。	AWS DevOps
訂閱 Amazon SNS 主題。	Terraform 已建立 Amazon SNS 主題。訂閱 主題的端點並確認訂閱，讓核准者收到管道中待核准動作的通知。如需說明，請參閱建立 Amazon SNS 主題的訂閱。	一般 AWS

定義您的護欄和政策

任務	Description	所需的技能
填入遠端儲存庫。	從複製的儲存庫，將`sample-repository` 資料夾的內容複製到遠端儲存庫。這包括 `environments` 、`scp-`	DevOps 工程師

任務	Description	所需的技能
	`management` 、 `rcp-management` 和 `utils` 資料夾。	

任務	Description	所需的技能
定義您的環境。	1. 在 environments 資料夾中，開啟 environments.json 檔案。這是您為 RCPs AWS 帳戶 和 SCPs 定義目標和 OUs 的檔案。 2. 刪除範例環境。 3. 以下列格式新增您的目標環境： `[ { "ID": "<environment-name>", "Target": [ "<ou-name>:<ou-id>", "<account-name>:<account-id>" ] } ]` 其中： • <environment-name> 是您指派給 OUs和 AWS 帳戶群組的名稱。您可以在資訊清單檔案中使用此名稱來定義要套用政策的位置。 • <ou-name> 是目標 OU 的名稱。 • <ou-id> 是目標 OU 的 ID。	DevOps 工程師

任務	Description	所需的技能
	• <account-name> 是目標的名稱 AWS 帳戶。 • <account-id> 是目標的 ID AWS 帳戶。 如需範例，請參閱原始程式碼儲存庫。 4. 儲存並關閉 environments.json 檔案。	

任務	Description	所需的技能
定義您的護欄。	1. 導覽至遠端儲存庫中的 `rcp-management/guardrails` 資料夾。這是您定義 RCP 資訊清單檔案護欄的資料夾。每個護欄都必須在個別檔案中。護欄檔案可以包含一或多個陳述式。 **Note** 您可以在 SCPs 和 RCPs 的資訊清單檔案中的多個陳述式中使用相同的護欄。如果您修改護欄，任何包含此護欄的政策都會受到影響。 2. 刪除從原始程式碼儲存庫複製的任何護欄範例。 3. 建立新的 .json 檔案，並提供描述性名稱。 4. 開啟您建立的 .json 檔案。 5. 以下列格式定義護欄： `[ { "Sid": "<guardrail-name>", "Effect": "<effect-value>", "Action": [ "<action-name>" ],`	DevOps 工程師

任務	Description	所需的技能
	<pre>"Resource": "<resource-arn>", "Condition": { "<condition-operator>": { "<condition-key>": ["<condition-value>"] } } }]</pre> 其中： • <guardrail-name> 是護欄的唯一名稱。此名稱不能用於任何其他護欄。 • <effect-value> 必須是 Allow或 Deny。如需詳細資訊，請參閱效果。 • <action-name> 必須是服務支援之動作的有效名稱。如需詳細資訊，請參閱動作。 • <resource-arn> 是護欄套用的資源的 Amazon Resource Name (ARN)。您也可以使用萬用字元，例如 *或 ?。如需詳細資訊，請參閱 資源。 • <condition-operator> 是有效的條件運算	

任務	Description	所需的技能
	子。如需詳細資訊，請參閱條件運算子。 • <condition-key> 是有效的全域條件內容索引鍵或服務特定內容索引鍵。如需詳細資訊，請參閱 條件。 • <condition-value> 是條件中用來評估護欄是否套用的特定值。如需詳細資訊，請參閱 條件。 如需 RCP 護欄的範例，請參閱原始程式碼儲存庫。 6. 儲存並關閉 .json 檔案。 7. 重複這些步驟，視需要建立任意數量的 RCP 護欄。 8. 在 scp-management/guardrails 資料夾中重複這些步驟，視需要為 SCPs 建立任意數量的護欄。如需 SCP 護欄範例，請參閱原始程式碼儲存庫。	

任務	Description	所需的技能
定義您的政策。	1. 導覽至遠端儲存庫中的 `rcp-management/policies` 資料夾。這是您為 RCP 資訊清單檔案定義完整政策的資料夾。每個政策必須是個別檔案。 **Note** 如果您修改此資料夾中的政策，政策變更會影響套用此政策的任何帳戶或 OUs，如清單檔案所定義。 2. 刪除從原始程式碼儲存庫複製的任何範例政策。 3. 建立新的 .json 檔案，並提供描述性名稱。 4. 開啟您建立的 .json 檔案。 5. 定義 RCP。如需 RCPs範例，請參閱原始程式碼儲存庫，或參閱 AWS Organizations 文件中的資源控制政策範例。 6. 儲存並關閉 .json 檔案。 7. 重複這些步驟，視需要建立任意數量RCPs。 8. 在 `scp-management/policies` 資料夾中重複這些步驟，視需要建立任意數量SCPs。如需 SCPs範例，請參閱原始程式碼儲存庫，或參閱 AWS Organizat	DevOps 工程師

任務	Description	所需的技能
	ions 文件中的服務控制政策範例。	

使用資訊清單檔案來管理政策

任務	Description	所需的技能
設定資訊清單檔案。	1. 在 `rcp-management` 資料夾中，開啟 `rcp-management.json` 檔案。這是您定義哪些 RCP 護欄和完整 RCPs 適用於目標環境的檔案。如需此檔案的範例，請參閱原始程式碼儲存庫。 2. 刪除範例陳述式。 3. 以下列格式新增陳述式： `[ { "SID": "<statement-name>", "Target": { "Type": "<target-type>", "ID": "<target-name>" }, "Guardrails": [ "<guardrail-name>" ], "Policy": "<policy-name>", "Comments": "<comment-text>" }`	DevOps 工程師

任務	Description	所需的技能

	]
	
	其中：
	• <statement-name> 是陳述式的唯一名稱。
	• <target-type> 是您要套用政策的目標類型。有效值為 Account、OU、Environment 或 Tag。
	• <target-name> 是您要套用政策的目標識別符。輸入下列其中之一：
	  • 對於 AWS 帳戶，輸入識別符為 <account-name>:<account-id> 。
	  • 對於 OU，將識別符輸入為 <OU-name>:<ou-id> 。
	  • 針對環境，輸入您在 environments.json 檔案中定義的唯一名稱。
	  • 對於標籤，將鍵值對輸入為 <tag-key>:<tag-value> 。
	• <guardrail-name> 是您在 rcp-management/guardrails 資料夾中定義的 RCP 護欄的唯一名稱。您可以在此元素中新增多個護欄。	

任務	Description	所需的技能
	如果您不想套用護欄，您可以將此欄位保留空白。 • `<policy-name>` 是您在 `rcp-management/policies` 資料夾中定義的 RCP 的唯一名稱。您只能在此元素中新增一個政策。如果您不想套用政策，您可以將此欄位保留空白。 • `<comment-text>` 是您可以輸入用於文件用途的描述。管道處理期間不會使用此欄位。如果您不想新增註解，您可以將此欄位保留空白。 4. 重複這些步驟，視需要新增任意數量的陳述式，為您的組織設定 RCPs。 5. 儲存並關閉 `rcp-management.json` 檔案。 6. 在 `scp-management` 資料夾中，重複 `scp-management.json` 檔案中的這些步驟。這是您定義哪些 SCP 護欄和完整 SCPs 適用於目標環境的檔案。如需此檔案的範例，請參閱原始程式碼儲存庫。	

任務	Description	所需的技能
啟動管道。	遞交變更並推送至您在 `variables.tf` 檔案中定義的遠端儲存庫分支。一般而言，這是`main`分支。CI/CD 管道會自動啟動。如果有任何管道錯誤，請參閱此模式的故障診斷一節。	DevOps 工程師
核准變更。	`Validate-Plan` CodeBuild 專案完成後，政策核准者會透過您先前設定的 Amazon SNS 主題收到通知。請執行下列操作： 1. 開啟通知訊息。 2. 如果可用，請檢閱政策變更的摘要。 3. 遵循 CodePipeline 中的核准或拒絕核准動作中的指示。	一般 AWS、政策核准者
驗證部署。	1. 登入委派管理員帳戶中的 AWS Organizations 主控台 AWS Organizations。 2. 在服務控制政策頁面上，確認您建立SCPs 已列出。 3. 選擇透過管道管理的 SCP，並確認它適用於預期目標。 4. 在資源控制政策頁面上，確認您建立RCPs 已列出。 5. 選擇透過管道管理的 RCP，並確認它適用於預期目標。	一般 AWS

疑難排解

問題	解決方案
管道Validate-Plan 階段中的資訊清單檔案錯誤	如果 scp-management.json 或 檔案有任何錯誤，「資訊清單檔案驗證與計劃階段的管道錯誤」訊息會顯示在管道輸出中rcp-management.json 。可能的錯誤包括不正確的環境名稱、重複SIDs 或無效的欄位或值。請執行下列操作： 1. 請遵循檢視建置詳細資訊 AWS CodeBuild中的指示。 2. 在建置日誌中，尋找驗證錯誤。錯誤會提供導致建置失敗之原因的詳細資訊。 3. 更新對應的 .json 檔案。 4. 遞交更新的檔案並推送至遠端儲存庫。管道會重新啟動。 5. 監控狀態以確認驗證錯誤已解決。
管道Validate-Plan 階段中的 IAM Access Analyzer 調查結果	如果護欄或政策定義中有任何錯誤，「在驗證與計劃階段期間 IAM Access Analyzer 中尋找」訊息會顯示在管道輸出中。此模式使用 IAM Access Analyzer 來驗證最終政策。請執行下列操作： 1. 請遵循檢視建置詳細資訊 AWS CodeBuild中的指示。 2. 在建置日誌中，尋找 IAM Access Analyzer 驗證錯誤。錯誤會提供導致建置失敗之原因的詳細資訊。如需調查結果類型的詳細資訊，請參閱 IAM 政策驗證檢查參考。 3. 更新護欄或政策對應的 .json 檔案。 4. 遞交更新的檔案並推送至遠端儲存庫。管道會重新啟動。

問題	解決方案
	5. 監控狀態以確認驗證錯誤已解決。

相關資源

- JSON 政策元素參考 (IAM 文件)
- 資源控制政策 (AWS Organizations 文件)
- 服務控制政策 (AWS Organizations 文件)
- 新增或移除對 Amazon Bedrock 基礎模型的存取權 (Amazon Bedrock 文件)
- 在 CodePipeline (CodePipeline 文件) 中核准或拒絕核准動作 CodePipeline

其他資訊

以資源為基礎的委派政策範例

以下是 的範例資源型委派政策 AWS Organizations。它允許委派的管理帳戶管理組織的 SCPs RCPs。在下列範例政策中，將 <MEMBER_ACCOUNT_ID>取代為您部署政策管理管道的帳戶 ID。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DelegationToAudit",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::<MEMBER_ACCOUNT_ID>:root"
      },
      "Action": [
        "organizations:ListTargetsForPolicy",
        "organizations:CreatePolicy",
        "organizations:DeletePolicy",
        "organizations:AttachPolicy",
        "organizations:DetachPolicy",
        "organizations:DisablePolicyType",
        "organizations:EnablePolicyType",
        "organizations:UpdatePolicy",
        "organizations:DescribeEffectivePolicy",
        "organizations:DescribePolicy",
```

```
        "organizations:DescribeResourcePolicy"
      ],
      "Resource": "*"
    }
  ]
}
```

將現有政策遷移至管道

如果您有要透過此管道遷移和管理的現有 SCPs 或 RCPs，您可以使用程式碼儲存庫 `sample-repository/utils` 資料夾中的 Python 指令碼。這些指令碼包括：

- `check-if-scp-exists-in-env.py` – 此指令碼會檢查指定的政策是否適用於您在 `environments.json` 檔案中定義的特定環境中的任何目標。輸入下列命令來執行此指令碼：

  ```
  python3 check-if-scp-exists-in-env.py \
     --policy-type <POLICY_TYPE> \
     --policy-name <POLICY_NAME> \
     --env-id <ENV_ID>
  ```

  在此命令中取代以下內容：

  - <POLICY_TYPE> 是 `scp` 或 `rcp`
  - <POLICY_NAME> 是 SCP 或 RCP 的名稱
  - <ENV_ID> 是您在 `environments.json` 檔案中定義的環境 ID
- `create-environments.py` – 此指令碼會根據您環境中目前的 SCPs和 RCPs 建立 environment.json 檔案。它不包括透過 部署的政策 AWS Control Tower。輸入下列命令來執行此指令碼，其中 <POLICY_TYPE>為 `scp`或 `rcp`：

  ```
  python create-environments.py --policy-type <POLICY_TYPE>
  ```
- `verify-policies-capacity.py` – 此指令碼會檢查您定義的每個環境，以判斷每個 AWS Organizations 政策相關配額的剩餘容量。您可以定義要在 `environments.json` 檔案中檢查的環境。輸入下列命令來執行此指令碼，其中 <POLICY_TYPE>為 `scp`或 `rcp`：

  ```
  python verify-policies-capacity.py --policy-type <POLICY_TYPE>
  ```

使用 以程式碼形式管理 AWS IAM Identity Center 許可集 AWS CodePipeline

Andre Cavalcante 和 Claison Amorim，Amazon Web Services

總結

AWS IAM Identity Center 可協助您集中管理所有 AWS 帳戶 和應用程式的單一登入 (SSO) 存取。您可以在 IAM Identity Center 中建立和管理使用者身分，也可以連接現有的身分來源，例如 Microsoft Active Directory 網域或外部身分提供者 (IdP)。IAM Identity Center 提供統一的管理體驗，透過使用許可集來定義、自訂和指派精細 AWS 的環境存取。許可集適用於來自 IAM Identity Center 身分存放區或外部 IdP 的聯合身分使用者和群組。

此模式可協助您在以組織身分管理的多帳戶環境中，將 IAM Identity Center 許可集管理為程式碼 AWS Organizations。透過此模式，您可以達成下列目標：

- 建立、刪除和更新許可集
- 建立、更新或刪除對目標 AWS 帳戶、組織單位 (OUs) 或組織根目錄的許可集指派。

若要以程式碼形式管理 IAM Identity Center 許可和指派，此解決方案會部署使用 AWS CodeBuild 和的持續整合和持續交付 (CI/CD) 管道 AWS CodePipeline。您可以在存放在遠端儲存庫的 JSON 範本中管理許可集和指派。當 Amazon EventBridge 規則偵測到儲存庫的變更，或偵測到目標 OU 中帳戶的修改時，就會啟動 AWS Lambda 函數。Lambda 函數會啟動 CI/CD 管道，以更新 IAM Identity Center 中的許可集和指派。

先決條件和限制

先決條件

- 以組織身分管理的多帳戶環境 AWS Organizations。如需詳細資訊，請參閱建立組織。
- IAM Identity Center，已啟用並使用身分來源設定。如需詳細資訊，請參閱 IAM Identity Center 文件中的入門。
- 註冊為下列委派管理員的成員帳戶 AWS 服務：
  - IAM Identity Center – 如需說明，請參閱 IAM Identity Center 文件中的註冊成員帳戶。
  - AWS Organizations – 如需說明，請參閱 的委派管理員 AWS Organizations。此帳戶必須具有列出和描述帳戶和 OUs許可。

> **Note**
>
> 您必須使用與這兩個服務的委派管理員相同的帳戶。

- 在 IAM Identity Center 委派管理員帳戶和組織的管理帳戶中部署 AWS CloudFormation 堆疊的許可。如需詳細資訊，請參閱 CloudFormation 文件中的控制存取。
- IAM Identity Center 委派管理員帳戶中的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。您可以將成品程式碼上傳到此儲存貯體。如需說明，請參閱 Amazon S3 文件中的建立儲存貯體。
- 組織的管理帳戶的帳戶 ID。如需說明，請參閱尋找您的 AWS 帳戶 ID。
- 原始碼主機中的儲存庫，例如 GitHub。

限制

- 此模式無法用來管理或指派單一帳戶環境或非以組織身分管理之帳戶的許可集 AWS Organizations。
- 部署後無法修改許可集名稱、指派 IDs 和 IAM Identity Center 主體類型和 IDs。
- 此模式可協助您建立和管理自訂許可。您無法使用此模式來管理或指派預先定義的許可。
- 此模式無法用來管理組織管理帳戶的許可集。

Architecture

目標架構

該圖顯示以下工作流程：

1. 使用者進行下列其中一個變更：
   - 對遠端儲存庫進行一或多個變更，例如 GitHub
   - 在 中修改 OU 中的帳戶 AWS Organizations
2. 如果使用者將遠端儲存庫的變更遞交至主分支，則管道會開始。

   如果使用者修改了 OU 中的帳戶，則 `MoveAccount` EventBridge 規則會偵測變更，並在組織的管理帳戶中啟動 Lambda 函數。
3. 啟動的 Lambda 函數會在 CodePipeline 中啟動 CI/CD 管道。
4. CodePipeline 會啟動 `TemplateValidation` CodeBuild 專案。`TemplateValidation` CodeBuild 專案使用遠端儲存庫中的 Python 指令碼來驗證許可集範本。CodeBuild 驗證下列項目：

- 許可集名稱是唯一的。
- 指派陳述式 IDs(`Sid`) 是唯一的。
- `CustomPolicy` 參數中的政策定義和有效。（此驗證使用 AWS Identity and Access Management Access Analyzer。）
- 受管政策的 Amazon Resource Name (ARNs) 有效。

5. Deploy CodeBuild 專案中的`PermissionSet`動作群組使用 適用於 Python (Boto3) 的 AWS SDK 來刪除、建立或更新 IAM Identity Center 中的許可集。只有具有 `SSOPipeline:true`標籤的許可集會受到影響。透過此管道管理的所有許可集都有此標籤。
6. Deploy CodeBuild 專案中的`Assignments`動作群組使用 Terraform 來刪除、建立或更新 IAM Identity Center 中的指派。Terraform 後端狀態檔案存放在相同帳戶中的 Amazon S3 儲存貯體中。
7. CodeBuild 會更新 IAM Identity Center 中的許可集和指派。

自動化和擴展

由於多帳戶環境中的所有新帳戶都會移至 中的特定組織單位 AWS Organizations，因此此解決方案會自動執行所需的許可集，並授予您在指派範本中指定的所有帳戶。不需要額外的自動化或擴展動作。

在大型環境中，對 IAM Identity Center 的 API 請求數量可能會導致此解決方案執行速度變慢。Terraform 和 Boto3 會自動管理限流，以將效能降低降至最低。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。
- AWS CodeBuild 是一種全受管建置服務，可協助您編譯原始程式碼、執行單元測試，並產生準備好部署的成品。
- AWS CodePipeline 可協助您快速建模和設定軟體版本的不同階段，並自動化持續發行軟體變更所需的步驟。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如， AWS Lambda 函數、使用 API 目的地的 HTTP 呼叫端點，或其他事件匯流排 AWS 帳戶。
- AWS IAM Identity Center 可協助您集中管理所有 AWS 帳戶 和雲端應用程式的單一登入 (SSO) 存取。

- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。
- 適用於 Python (Boto3) 的 AWS SDK 是一種軟體開發套件，可協助您整合 Python 應用程式、程式庫或指令碼 AWS 服務。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

程式碼儲存庫

此模式的程式碼可在 aws-iam-identity-center-pipeline 儲存庫中使用。儲存庫中的範本資料夾包含許可集和指派的範例範本。它還包含用於在目標帳戶中部署 CI/CD 管道 AWS 和資源的 AWS CloudFormation 範本。

最佳實務

- 在您開始修改許可集和指派範本之前，建議您為組織規劃許可集。考慮許可應該是什麼、許可集應該套用哪些帳戶或 OUs，以及許可集應該影響哪些 IAM Identity Center 委託人（使用者或群組）。部署後無法修改許可集名稱、關聯 IDs 和 IAM Identity Center 主體類型和 IDs。
- 遵守最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱《 AWS Identity and Access Management (IAM) 文件》中的授予最低權限和安全最佳實務。

史詩

規劃許可集和指派

任務	Description	所需的技能
複製儲存庫。	在 bash shell 中，輸入下列命令。這會從 GitHub 複製 aws-iam-identity-center-pipeline 儲存庫。 `git clone https://github.com/aws-samples/aws-iam-identity-center-pipeline.git`	DevOps 工程師

任務	Description	所需的技能
定義許可集。	1. 在複製的儲存庫中，導覽至 `templates/permissionsets` 資料夾，然後開啟其中一個可用的範本。 2. 在 `Name` 參數中，輸入許可集的名稱。此值必須是唯一的，且無法在部署後變更。 3. 在 `Description` 參數中，簡短描述許可集，例如其使用案例。 4. 在 `SessionDuration` 參數中，指定使用者可登入的時間長度 AWS 帳戶。使用 ISO-8601 持續時間格式（維基百科），例如 `PT4H` 4 小時。如果未定義任何值，IAM Identity Center 中的預設值為 1 小時。 5. 在 `RelayState` 參數中，指定 URL，提供最適合使用者角色的主控台快速存取。 6. 自訂許可集中的政策。下列所有參數皆為選用，可在部署後修改。您必須至少使用其中一個參數，才能定義許可集中的政策： • 在 `ManagedPolicies` 參數中，輸入您要指派之任何AWS 受管政策ARNs。 • 在 `CustomerManagedPolicies` 參數中，輸入您要指派的任	DevOps 工程師

任務	Description	所需的技能
	何客戶受管政策的名稱。請勿使用 ARN。 • 在 PermissionBoundary 參數中，執行下列動作來指派許可界限： • 如果您使用 AWS 受管政策做為許可界限，請在 PolicyType 中輸入 AWS，並在 Policy中輸入政策的 ARN。 • 如果您使用客戶受管政策做為許可界限，請在 PolicyType 中輸入 Customer，並在 Policy中輸入政策的名稱。請勿使用 ARN。 • 在 CustomPolicy 參數中，定義您要指派的任何自訂 JSON 格式政策。如需 JSON 政策結構的詳細資訊，請參閱 JSON 政策概觀。 7. 儲存並關閉許可集範本。我們建議您使用與許可集名稱相符的名稱來儲存檔案。 8. 重複此程序，為您的組織建立所需的任意數量許可集，並刪除不需要的任何範例範本。	

任務	Description	所需的技能
定義指派。	1. 在複製的儲存庫中，導覽至 `templates/assignments` 資料夾，然後開啟 `iam-identitycenter-assigments.json` 。此檔案說明如何將許可集指派給 AWS 帳戶 或 OUs。 2. 在 `SID` 參數中，輸入指派的識別符。此值必須是唯一的，且無法在部署後修改。 3. 在 `Target` 參數中，定義您要套用許可集的帳戶或組織。有效值為帳戶 IDs、OUs 或 `root`。 會將許可集`root`指派給組織中的所有成員帳戶，管理帳戶除外。以雙引號輸入值，並以逗號分隔多個值。帳戶 IDs 和 OUs應遵循 模式： `{{account_name}}:{{account_id}}` 或 `{{ou_name}}:{{ou_id}}` 。如果您想要以遞迴方式將許可指派給巢狀 OUs，請使用結尾有萬用字元的 OU 模式。範例：`{{ou_name}}:{{ou_id}}:*` 4. 在 `PrincipalType` 參數中，輸入將受許可集影響的 IAM Identity Center 主體類型。有效值為 USER 或 GROUP。部署後無法修改此值。	DevOps 工程師

任務	Description	所需的技能
	5. 在 `PrincipalID` 參數中，輸入將受許可集影響的 IAM Identity Center 身分存放區中的使用者或群組名稱。部署後無法修改此值。 6. 在 `PermissionSetName` 參數中，輸入您要指派的許可集名稱。 7. 重複步驟 2–6，以建立此檔案中所需的任意數量指派。一般而言，每個許可集都有一個指派。刪除任何不需要的範例指派。 8. 儲存並關閉 `iam-identitycenter-assigments.json` 檔案。	

部署許可集和指派

任務	Description	所需的技能
在 IAM Identity Center 委派管理員帳戶中部署資源。	1. 在 IAM Identity Center 委派管理員帳戶中，開啟 CloudFormation 主控台。 2. 部署 `iam-identitycenter-pipeline.yaml` 範本。為堆疊提供清晰且描述性的名稱，並依照指示更新參數。如需說明，請參閱 CloudFormation 文件中的建立堆疊。	DevOps 工程師
在 AWS Organizations 管理帳戶中部署資源。	1. 登入組織的管理帳戶。	DevOps 工程師

任務	Description	所需的技能
	2. 開啟 AWS CloudFormation 主控台。 3. 在導覽列中，選擇目前顯示的 名稱 AWS 區域。然後選擇 us-east-1 區域。此為必要區域，以便 MoveAccount EventBridge 規則可以偵測與組織變更相關聯的 AWS CloudTrail 事件。 4. 部署 iam-identitycenter-organization 範本。為堆疊提供清晰且描述性的名稱，並依照指示更新參數。如需說明，請參閱 CloudFormation 文件中的建立堆疊。	
完成遠端儲存庫設定。	將 AWS CodeConnections 連線的狀態從 變更為 PENDING AVAILABLE 。此連線是在您部署 CloudFormation 堆疊時建立的。如需說明，請參閱 CodeConnections 文件中的更新待定連線。	DevOps 工程師
將檔案上傳至遠端儲存庫。	將您從aws-samples 儲存庫下載並在先前步驟中編輯的所有檔案上傳至遠端儲存庫。main 分支的變更會啟動管道，這會建立或更新許可集和指派。	DevOps 工程師

更新許可集和指派

任務	Description	所需的技能
更新許可集和指派。	當 `MoveAccount` Amazon EventBridge 規則偵測到組織中帳戶的修改時，CI/CD 管道會自動啟動和更新許可集。例如，如果您將 帳戶新增至指派 JSON 檔案中指定的 OU，則 CI/CD 管道會將許可集套用至新帳戶。 如果您想要修改已部署的許可集和指派，請更新 JSON 檔案，然後將其遞交至遠端儲存庫。 使用 CI/CD 管道管理先前部署的許可集和關聯時，請注意下列事項： • 如果您變更許可集的名稱，CI/CD 管道會刪除原始許可集並建立新的許可集。 • 此管道只會管理具有 `SSOPipeline:true` 標籤的許可集。 • 您可以在儲存庫的相同資料夾中擁有多個許可集和指派範本。 • 如果您刪除範本，管道會刪除指派或許可集。 • 如果您刪除整個指派 JSON 區塊，管道會從 IAM Identity Center 刪除指派。	DevOps 工程師

任務	Description	所需的技能
	• 您無法刪除指派給 的許可集 AWS 帳戶。首先，您必須取消指派許可集。	

疑難排解

問題	解決方案
存取遭拒錯誤	確認您擁有部署 CloudFormation 範本所需的許可，以及其中定義的資源。如需詳細資訊，請參閱 CloudFormation 文件中的控制存取。
驗證階段中的管道錯誤	如果許可集或指派範本中有任何錯誤，就會出現此錯誤。 1. 在 CodeBuild 中，檢視建置詳細資訊。 2. 在建置日誌中，尋找驗證錯誤，該錯誤提供導致建置失敗之原因的詳細資訊。 3. 更新許可集或指派範本，然後將其遞交至儲存庫。 4. CI/CD 管道會重新啟動 CodeBuild 專案。監控狀態以確認驗證錯誤已解決。

相關資源

- 許可集 (IAM Identity Center 文件)

使用 AWS Secrets Manager 管理登入資料

Durga Prasad Cheepuri，Amazon Web Services

總結

此模式會逐步引導您使用 AWS Secrets Manager 來動態擷取 Java Spring 應用程式的資料庫登入資料。

以往當您建立從資料庫擷取資訊的自訂應用程式時，通常必須內嵌登入資料 (秘密)，才可直接存取應用程式中的資料庫。輪換登入資料時，您必須花時間更新應用程式以使用新的登入資料，然後分發更新的應用程式。如果您有多個共用登入資料的應用程式，且您遺漏更新其中一個，則應用程式會失敗。由於這種風險，許多使用者選擇不定期輪換其登入資料，這實際上取代了另一個登入資料的風險。

Secrets Manager 可讓您將程式碼中的硬式編碼登入資料（包括密碼）取代為 API 呼叫，以程式設計方式擷取秘密。這有助於確保檢查程式碼的人員不會洩露秘密，因為秘密根本不存在。您也可以設定 Secrets Manager 根據您指定的排程自動輪換秘密。這可讓您將長期秘密取代為短期秘密，這有助於大幅降低入侵的風險。如需詳細資訊，請參閱 AWS Secrets Manager 文件。

先決條件和限制

先決條件

- 可存取 Secrets Manager 的 AWS 帳戶
- Java Spring 應用程式

Architecture

來源技術堆疊

- 具有存取資料庫之程式碼的 Java Spring 應用程式，具有從 application.properties 檔案管理的資料庫登入資料。

目標技術堆疊

- Java Spring 應用程式具有存取資料庫的程式碼，並在 Secrets Manager 中管理資料庫登入資料。application.properties 檔案會將秘密保留給 Secrets Manager。

Secrets Manager 與應用程式的整合

工具

- Secrets Manager – AWS Secrets Manager 是一種 AWS 服務，可讓您更輕鬆地管理秘密。秘密可能是資料庫憑證、密碼、第三方 API 金鑰，甚至是任意文字。您可以使用 Secrets Manager 主控台、Secrets Manager 命令列界面 (CLI) 或 Secrets Manager API 和 SDKs存取。

史詩

在 Secrets Manager 中存放秘密

任務	Description	所需的技能
將資料庫登入資料儲存為 Secrets Manager 中的秘密。	遵循 Secrets Manager 文件中建立秘密的步驟，將 Amazon Relational Database Service (Amazon RDS) 或其他資料庫憑證儲存為 Secrets Manager 中的秘密。 https://docs.aws.amazon.com/secretsmanager/latest/userguide/manage_create-basic-secret.html	系統管理員
設定 Spring 應用程式存取 Secrets Manager 的許可。	根據 Java Spring 應用程式如何使用 Secrets Manager 來設定適當的許可。若要控制對秘密的存取，請根據 Secrets Manager 文件所提供的資訊建立政策，請參閱使用身分型政策 (IAM 政策） 和 ABAC for Secrets Manager 和使用資源型政策 for Secrets Manager 一節。請遵循 Secrets Manager 文件中擷取秘密值一節中的步驟。	系統管理員

更新 Spring 應用程式

任務	Description	所需的技能
新增 JAR 相依性以使用 Secrets Manager。	如需詳細資訊，請參閱其他資訊一節。	Java 開發人員
將秘密的詳細資訊新增至 Spring 應用程式。	使用秘密名稱、端點和 AWS 區域更新 application.properties 檔案。如需範例，請參閱其他資訊一節。	Java 開發人員
在 Java 中更新資料庫登入資料擷取程式碼。	在應用程式中，更新擷取資料庫登入資料的 Java 程式碼，以從 Secrets Manager 擷取這些詳細資訊。如需範例程式碼，請參閱其他資訊一節。	Java 開發人員

相關資源

- AWS Secrets Manager 文件
- 針對 Secrets Manager 使用身分型政策 (IAM 政策） 和 ABAC
- 針對 Secrets Manager 使用資源型政策
- 範例程式碼

其他資訊

新增使用 Secrets Manager 的 JAR 相依性

Maven：

```
<groupId>com.amazonaws</groupId>
    <artifactId>aws-java-sdk-secretsmanager</artifactId>
    <version>1.11. 355 </version>
```

Gradle：

```
compile group: 'com.amazonaws', name: 'aws-java-sdk-secretsmanager', version:
 '1.11.355'
```

使用秘密的詳細資訊更新 application.properties 檔案

```
spring.aws.secretsmanager.secretName=postgres-local
spring.aws.secretsmanager.endpoint=secretsmanager.us-east-1.amazonaws.com
spring.aws.secretsmanager.region=us-east-1
```

在 Java 中更新資料庫登入資料擷取程式碼

```
String  secretName  =  env.getProperty("spring.aws.secretsmanager.secretName");
String  endpoints  =  env.getProperty("spring.aws.secretsmanager.endpoint");
String  AWS Region  =  env.getProperty("spring.aws.secretsmanager.region");
AwsClientBuilder.EndpointConfiguration  config  =  new
  AwsClientBuilder.EndpointConfiguration(endpoints, AWS Region);
AWSSecretsManagerClientBuilder  clientBuilder  =
  AWSSecretsManagerClientBuilder.standard();
clientBuilder.setEndpointConfiguration(config);
AWSSecretsManager  client  =  clientBuilder.build();

ObjectMapper  objectMapper  =  new  ObjectMapper();

JsonNode  secretsJson  =  null;

ByteBuffer  binarySecretData;

GetSecretValueRequest  getSecretValueRequest  =  new
  GetSecretValueRequest().withSecretId(secretName);

GetSecretValueResult  getSecretValueResponse  =  null;

try  {
     getSecretValueResponse  =  client.getSecretValue(getSecretValueRequest);
    }

catch  (ResourceNotFoundException  e)  {
     log.error("The requested secret "  +  secretName  +  " was not found");
    }

catch  (InvalidRequestException  e)  {
```

```
     log.error("The request was invalid due to: "  +  e.getMessage());
     }

catch  (InvalidParameterException  e)  {
     log.error("The request had invalid params: "  +  e.getMessage());
     }
if  (getSecretValueResponse  ==  null)  {
     return  null;
     }  // Decrypted secret using the associated KMS key // Depending on whether the
 secret was a string or binary, one of these fields will be populated


String secret = getSecretValueResponse.getSecretString();

if (secret != null) {
     try {
                secretsJson  =  objectMapper.readTree(secret);
           }

     catch  (IOException  e)  {
                log.error("Exception while retrieving secret values: "  +
  e.getMessage());
           }
}

else  {
     log.error("The Secret String returned is null");

     return null;

     }
     String  host  =  secretsJson.get("host").textValue();
     String  port  =  secretsJson.get("port").textValue();
     String  dbname  =  secretsJson.get("dbname").textValue();
     String  username  =  secretsJson.get("username").textValue();
     String  password  =  secretsJson.get("password").textValue();
}
```

監控 Amazon ElastiCache 叢集的靜態加密

Abhishek Agawane，Amazon Web Services

總結

Amazon ElastiCache 是一種 Amazon Web Services (AWS) 服務，可提供高效能、可擴展且符合成本效益的快取解決方案，用於在雲端中分發記憶體內資料存放區或快取環境。它會從高輸送量和低延遲的記憶體內資料存放區擷取資料。此功能使其成為即時使用案例的熱門選擇，例如快取、工作階段存放區、遊戲、地理空間服務、即時分析和佇列。ElastiCache 提供 Redis 和 Memcached 資料存放區，兩者都提供低於毫秒的回應時間。

資料加密有助於防止未經授權的使用者讀取 Redis 叢集及其相關聯快取儲存系統上可用的敏感資料。這包括儲存至持久性媒體的資料，稱為靜態資料，以及可在用戶端與快取伺服器之間透過網路傳輸時攔截的資料，稱為傳輸中的資料。

您可以在建立複寫群組時，將 `AtRestEncryptionEnabled`參數設定為 ，以啟用 ElastiCache (Redis OSS) 的靜態加密`true`。啟用此參數時，它會在同步、備份和交換操作期間加密磁碟，並加密存放在 Amazon Simple Storage Service (Amazon S3) 中的備份。您無法在現有的複寫群組上啟用靜態加密。當您建立複寫群組時，您可以透過下列兩種方式啟用靜態加密：

- 透過選擇預設選項，該選項使用服務受管靜態加密。
- 使用客戶受管金鑰並提供來自 () 的金鑰 ID 或 Amazon Resource Name AWS Key Management Service (ARN)AWS KMS。

此模式提供監控 API 呼叫的安全性控制項，並在`CreateReplicationGroup`操作上產生 Amazon EventBridge Events 事件。此事件會呼叫執行 Python 指令碼的 AWS Lambda 函數。函數會從事件 JSON 輸入取得複寫群組 ID，並執行下列檢查，以判斷是否有未加密的叢集：

- 檢查`AtRestEncryptionEnabled`金鑰是否存在。
- 如果`AtRestEncryptionEnabled`存在， 會檢查值，以查看其是否為 `true`。
- 如果 `AtRestEncryptionEnabled`值設定為 `false`， 會使用 Amazon Simple Notification Service (Amazon SNS) 通知，設定追蹤違規的變數，並將違規訊息傳送至您提供的電子郵件地址。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 上傳所提供 Lambda 程式碼的 S3 儲存貯體。
- 您想要接收違規通知的電子郵件地址。
- 已啟用 ElastiCache 記錄，以存取所有 API 日誌。

限制

- 此偵測性控制是區域性的，必須部署在您要監控 AWS 區域 的每個 中。
- 控制項支援在虛擬私有雲端 (VPC) 中執行的複寫群組。
- 控制項支援執行下列節點類型的複寫群組：
  - R7g, R6gd, R6g, R5, R4, R3
  - M7g, M6g, M5, M4, M3
  - T4g、T3、T2
  - C7gn

產品版本

- 支援 ElastiCache (Redis OSS) 3.2.6 版或更新版本，以及 Valkey 7.2 版或更新版本

Architecture

工作流程架構

1. 使用者透過 AWS 管理主控台、 AWS Command Line Interface (AWS CLI) 或 API 呼叫啟動 ElastiCache 複寫群組。
2. 呼叫 `CreateReplicationGroup` API 時，ElastiCache 會產生 EventBridge 事件。
3. EventBridge 規則會觸發並呼叫 Lambda 函數以進行合規檢查。
4. Lambda 函數會處理事件，並檢查 ElastiCache 叢集上是否啟用靜態加密。
5. 如果偵測到加密違規，Lambda 函數會將通知訊息發佈至 SNS 主題。
6. Amazon SNS 會將加密合規違規的電子郵件通知傳送給管理員。

自動化和擴展

- 如果您使用的是 AWS Organizations，則可以使用 AWS CloudFormation StackSets 將此範本部署到您要監控的多個帳戶中。

工具

AWS 服務

- Amazon ElastiCache 可讓您在 中輕鬆設定、管理和擴展分散式記憶體內快取環境 AWS 雲端。它提供高效能、可調整大小且符合成本效益的記憶體內快取，同時消除與部署和管理分散式快取環境相關的複雜性。ElastiCache 可與 Redis 和 Memcached 引擎搭配使用。
- AWS CloudFormation 可協助您建立和設定 AWS 資源的模型、快速且一致地佈建資源，以及在整個生命週期中進行管理。您可以使用範本來描述您的資源及其相依性，並將它們一起啟動和設定為堆疊，而不是個別管理資源。您可以管理和佈建跨多個 和 的堆疊 AWS 帳戶 AWS 區域。
- Amazon EventBridge 提供近乎即時的系統事件串流，描述 AWS 資源的變更。EventBridge 會在操作變更發生時得知，並視需要採取修正動作，方法是傳送訊息以回應環境、啟用 函數、進行變更，以及擷取狀態資訊。
- AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只會在需要時執行程式碼，並自動從每天的幾個請求擴展到每秒數千個請求。只需為使用的運算時間支付費用，一旦未執行程式碼，就會停止計費。
- Amazon SNS 會協調和管理發佈者和用戶端之間的訊息傳送，包括 Web 伺服器和電子郵件地址。訂閱者會收到發佈到所訂閱主題的所有訊息，且某一主題的所有訂閱者均會收到相同訊息。

Code

此模式的程式碼可在 GitHub Monitor Amazon ElastiCache 叢集中用於靜態加密儲存庫。如需如何使用儲存庫中檔案的資訊，請參閱 Epics 一節。

最佳實務

部署

- 在部署此控制項之前，請確定 AWS CloudTrail 正在記錄 ElastiCache API 呼叫。
- 這是區域性控制項；在您使用 ElastiCache 的每個 AWS 區域 中部署控制項。
- 在開發/測試環境中驗證解決方案，再將其部署到生產環境。

安全性

- 若要增強對加密金鑰的控制，請使用客戶受管 KMS 金鑰。
- Review AWS Identity and Access Management (IAM) 許可，以確保 Lambda 執行角色的最低權限存取。
- 設定無效字母佇列中訊息的提醒。

操作

- 設定適當的日誌保留，以平衡合規需求與成本。
- 調校 Lambda 的預留並行，以根據您的 ElastiCache 建立頻率進行調整。
- 訂閱多個電子郵件地址至 Amazon SNS 以取得團隊通知。

監控

- 檢閱 Amazon CloudWatch 警示，以確保警示閾值符合您的操作需求。
- 定期監控 Lambda 指標執行持續時間和錯誤率。
- 定期稽核違規，以檢閱加密合規通知。

史詩

部署安全控制

任務	Description	所需的技能
從 GitHub 下載程式碼。	從 GitHub 複製或下載程式碼儲存庫。儲存庫包含 檔案`index.py`和 `elasticache_encryption_at_rest.yml` 。	雲端架構師
建立 Lambda 部署套件。	從 Python 程式碼建立兩個 .zip 檔案： • 建立 `ElastiCache-EncryptionAtRest.zip` ，其中包含 `index.py`。	雲端架構師

任務	Description	所需的技能
	• 使用下列命令： `zip ElastiCache-EncryptionAtRest.zip index.py`	
將程式碼上傳至 S3 儲存貯體。	1. 建立新的 S3 儲存貯體或使用現有的 S3 儲存貯體上傳 Lambda 程式碼。 2. 壓縮 Lambda 程式碼 (`index.py`) 並命名為 `ElastiCache-EncryptionAtRest.zip` 。 3. 將 .zip 檔案上傳至 S3 儲存貯體。此儲存貯體必須與您要評估 AWS 區域 的資源位於相同的 中。	雲端架構師
部署 CloudFormation 範本。	在與 S3 儲存貯體 AWS 區域 相同的 中開啟CloudFormation 主控台，並部署程式碼儲存庫中提供`elasticache_encryption_at_rest.yml` 的檔案。在下一個 Epic 中，提供範本參數的值。	雲端架構師

完成 CloudFormation 範本中的參數

任務	Description	所需的技能
提供 S3 儲存貯體名稱。	輸入您在第一個特徵中建立或選取的 S3 儲存貯體名稱。此 S3 儲存貯體包含 Lambda 程式碼的 .zip 檔案，且必須與要評	雲端架構師

任務	Description	所需的技能
	估的 CloudFormation 範本和資源 AWS 區域 位於相同的 中。	
提供 S3 金鑰。	提供 Lambda 程式碼 .zip 檔案在 S3 儲存貯體中的位置，不帶正斜線 （例如 `ElasticCache-EncryptionAtRest.zip` 或 `controls/ElasticCache-EncryptionAtRest.zip`)。	雲端架構師
提供電子郵件地址。	提供您要接收違規通知的作用中電子郵件地址。	雲端架構師
指定記錄層級。	指定記錄層級和詳細程度。 • `Info` 會指定應用程式進度的詳細資訊訊息，且應僅用於偵錯。 • `Error` 指定仍然可以允許應用程式繼續執行的錯誤事件。 • `Warning` 指定潛在的有害情況。	雲端架構師

確認訂閱

任務	Description	所需的技能
確認電子郵件訂閱。	當 CloudFormation 範本成功部署時，它會傳送訂閱訊息到您提供的電子郵件地址。若要接收通知，您必須確認此電子郵件訂閱。	雲端架構師

疑難排解

問題	解決方案
未觸發 Lambda 函數	徵狀：建立或修改 ElastiCache 叢集後CloudWatch 中沒有日誌。 解決方案： • 驗證 EventBridge 規則狀態： `aws events describe-rule --name <rule-name>` • 確認 Lambda 資源型政策允許 EventBridge 調用。 • 確認 EventBridge 事件模式符合 ElastiCache API 呼叫。
沒有電子郵件通知	徵狀：Lambda 函數已成功執行，但您不會收到任何電子郵件通知。 解決方案： • 透過電子郵件確認連結確認您的 Amazon SNS 訂閱。 • 檢查垃圾郵件資料夾是否有 Amazon SNS 電子郵件。 • 在 Lambda 環境變數中驗證 Amazon SNS 主題的 Amazon Resource Name (ARN)：`OUTBOUND_TOPIC_ARN` • 手動測試 Amazon SNS： `aws sns publish --topic-arn <arn> --message "Test"`
許可問題	徵狀：存取 Lambda 函數 CloudWatch 日誌中的拒絕錯誤。 CloudWatch

問題	解決方案
	解決方案： • 確認 Lambda 執行角色具有所需的 ElastiCache 讀取許可。 • 確認 AWS KMS 金鑰政策包含 Lambda 執行角色。 • 確定 Amazon SNS 主題政策允許 Lambda 發佈。

相關資源

- 從 CloudFormation 主控台建立堆疊 (CloudFormation 文件)
- ElastiCache (Redis OSS) 中的靜態加密 (ElastiCache 文件)

監控 IAM 根使用者活動

JJ Sung 和 Mostefa Brougui，Amazon Web Services

總結

每個 Amazon Web Services (AWS) 帳戶都有根使用者。作為 AWS Identity and Access Management (IAM) 的安全最佳實務，我們建議您使用根使用者來完成只有根使用者可以執行的任務。如需完整清單，請參閱《AWS 帳戶管理參考指南》中的需要根使用者憑證的任務。由於根使用者可以完整存取您的所有 AWS 資源和帳單資訊，因此建議您不要使用此帳戶並監控任何活動，這可能表示根使用者憑證已遭到入侵。

使用此模式，您可以設定監控 IAM 根使用者的事件驅動架構。此模式會設定中hub-and-spoke解決方案，以監控多個 AWS 帳戶、輻條帳戶，並將管理和報告集中在中樞帳戶單一帳戶中。

使用 IAM 根使用者憑證時，Amazon CloudWatch 和 AWS CloudTrail 會分別在日誌和追蹤中記錄活動。在輪輻帳戶中，Amazon EventBridge 規則會將事件傳送至中樞帳戶中的中央事件匯流排。在中樞帳戶中，EventBridge 規則會將事件傳送至 AWS Lambda 函數。函數使用 Amazon Simple Notification Service (Amazon SNS) 主題來通知您根使用者活動。

在此模式中，您會使用 AWS CloudFormation 範本在輪輻帳戶中部署監控和事件處理服務。您可以使用 HashiCorp Terraform 範本，在中樞帳戶中部署事件管理和通知服務。

先決條件和限制

先決條件

1. 在 AWS 環境中部署 AWS 資源的許可。
2. 部署 CloudFormation 堆疊集的許可。如需詳細資訊，請參閱堆疊集操作的先決條件 (CloudFormation 文件）。
3. 安裝 Terraform 並準備好使用。如需詳細資訊，請參閱入門 – AWS (Terraform 文件）。
4. 每個輻條帳戶中的現有線索。如需詳細資訊，請參閱 AWS CloudTrail 入門 (CloudTrail 文件）。
5. 線索已設定為將事件傳送至 CloudWatch Logs。如需詳細資訊，請參閱將事件傳送至 CloudWatch Logs (CloudTrail 文件）。
6. 您的中樞和輻條帳戶必須由 AWS Organizations 管理。

Architecture

下圖說明實作的建置區塊。

1. 使用 IAM 根使用者憑證時，CloudWatch 和 CloudTrail 會分別在日誌和追蹤中記錄活動。
2. 在輪輻帳戶中，EventBridge 規則會將事件傳送至中樞帳戶中的中央事件匯流排。
3. 在中樞帳戶中，EventBridge 規則會將事件傳送至 Lambda 函數。
4. Lambda 函數使用 Amazon SNS 主題，通知您根使用者活動。

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶和區域的生命週期進行管理。
- AWS CloudTrail 可協助您稽核 AWS 帳戶的控管、合規和營運風險。
- Amazon CloudWatch Logs 可協助您集中所有系統、應用程式和 AWS 服務的日誌，以便您可以監控日誌並將其安全地存檔。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如，AWS Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他 AWS 帳戶中的事件匯流排。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一種運算服務，可協助您執行程式碼，而無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。

其他工具和服務

- Terraform 是一種 CLI 應用程式，可透過使用組態檔案形式的程式碼來佈建和管理雲端基礎設施和資源。

程式碼儲存庫

此模式的原始程式碼和範本可在 GitHub 儲存庫中使用。此模式提供兩種範本：

- 包含您在中樞帳戶中部署之資源的 Terraform 範本
- 您在輪輻帳戶中部署為堆疊集執行個體的 CloudFormation 範本

儲存庫的整體結構如下。

```
.
 |__README.md
 |__spoke-stackset.yaml
 |__hub.tf
 |__root-activity-monitor-module
     |__main.tf  # contains Terraform code to deploy resources in the Hub account
     |__iam      # contains IAM policies JSON files
         |__ lambda-assume-policy.json          # contains trust policy of the IAM role
 used by the Lambda function
         |__ lambda-policy.json                 # contains the IAM policy attached to
 the IAM role used by the Lambda function
     |__outputs  # contains Lambda function zip code
```

Epics 區段提供部署範本的step-by-step說明。

史詩

將資源部署至中樞帳戶

任務	Description	所需的技能
複製範本程式碼儲存庫。	1. 開啟 AWS IAM 根使用者活動監視器儲存庫。 2. 在程式碼索引標籤的檔案清單上方，選擇程式碼，然後複製 HTTPS URL。 3. 在命令列界面中，將工作目錄變更為您要存放範例檔案的位置。 4. 輸入以下命令：	一般 AWS

任務	Description	所需的技能
	`git clone <repoURL>`	
更新 Terraform 範本。	1. 擷取您的組織 ID。如需說明，請參閱從管理帳戶檢視組織的詳細資訊 (AWS Organizations 文件）。 2. 在複製的儲存庫中，開啟 `hub.tf`。 3. 為您的環境使用適當的值更新以下內容： • `OrganizationId` – 新增您的組織 ID。 • `SNSTopicName` – 新增 Amazon SNS 主題的名稱。 • `SNSSubscriptions` – 新增應傳送 Amazon SNS 通知的電子郵件。 • `Region` – 新增您要部署資源的 AWS 區域碼。例如 `eu-west-1` 。 • `Tags` – 新增標籤。如需詳細資訊，請參閱標記 AWS 資源 (AWS 一般參考）。 4. 儲存並關閉 `hub.tf` 檔案。	一般 AWS

任務	Description	所需的技能

將資源部署至 AWS 中樞帳戶。	1. 在 Terraform 命令列界面中，導覽至複製儲存庫的根資料夾，然後輸入下列命令。
	`terraform init && terraform plan`
	2. 檢閱輸出並確認您想要建立描述的資源。
	3. 輸入以下命令。
	`terraform apply`
	4. 出現提示時，輸入 以確認部署yes。
		一般 AWS

將資源部署到您的發言帳戶

任務	Description	所需的技能
部署 CloudFormation 範本。	1. 登入 AWS 管理主控台，然後開啟 CloudFormation 主控台。 2. 從導覽窗格選擇 StackSets。 3. 選擇 StackSets 頁面上方的 Create StackSet (建立 StackSet)。 4. 在許可下，選擇服務受管許可。CloudFormation 會自動設定部署到 AWS Organizations 管理的目標帳戶所需的許可。	一般 AWS

任務	Description	所需的技能
	5. 在準備範本下，請選擇範本已準備就緒。 6. 在指定範本下，選擇上傳範本檔案。 7. 選擇選擇檔案，然後在複製的儲存庫中選取 `spoke-stackset.yaml` 。 8. 選擇下一步。 9. 在指定 StackSet 詳細資訊頁面上，輸入堆疊集的名稱。 10.在參數下，輸入中樞帳戶的帳戶 ID，然後選擇下一步。 11.在設定 StackSet 選項頁面的標籤下，新增您的標籤。 12.在執行組態下，選擇非作用中，然後選擇下一步。 13.在設定部署選項頁面上，指定您要部署堆疊集的組織單位和區域，然後選擇下一步。 14.在檢閱頁面上，選取我確認 AWS CloudFormation 可能會建立 IAM 資源，然後選擇提交。CloudFormation 會開始部署您的堆疊集。 如需詳細資訊和說明，請參閱建立堆疊集 (CloudFormation 文件) 。	

（選用）測試通知

任務	Description	所需的技能
使用根使用者憑證。	1. 使用根使用者憑證登入輔條帳戶或中樞帳戶。 2. 確認您指定的電子郵件帳戶收到 Amazon SNS 通知。	一般 AWS

相關資源

- 安全最佳實務 (IAM 文件）
- 使用 StackSets (CloudFormation 文件）
- 入門 (Terraform 文件）

其他資訊

Amazon GuardDuty 是一項持續的安全監控服務，可分析和處理日誌，以識別 AWS 環境中未預期和可能未經授權的活動。作為此解決方案的替代方案，如果您已啟用 GuardDuty，則可以在使用根使用者憑證時提醒您。GuardDuty 調查結果為 `Policy:IAMUser/RootCredentialUsage`，預設嚴重性為低。如需詳細資訊，請參閱管理 Amazon GuardDuty 調查結果。

建立 IAM 使用者時傳送通知

Mansi Suratwala 和 Sergiy Shevchenko，Amazon Web Services

總結

在 Amazon Web Services (AWS) 上，您可以使用此模式來部署 AWS CloudFormation 範本，以便在建立 AWS Identity and Access Management (IAM) 使用者時自動接收通知。

使用 IAM，您可以安全地管理對 AWS 服務和資源的存取。您可以建立和管理 AWS 使用者和群組，並使用許可來允許和拒絕這些使用者和群組存取 AWS 資源。

CloudFormation 範本會建立 Amazon CloudWatch Events 事件和 AWS Lambda 函數。事件使用 AWS CloudTrail 來監控在 AWS 帳戶中建立的任何 IAM 使用者。如果建立使用者，CloudWatch Events 事件會啟動 Lambda 函數，這會傳送 Amazon Simple Notification Service (Amazon SNS) 通知給您，通知您新的使用者建立事件。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 建立並部署的 AWS CloudTrail 追蹤

限制

- AWS CloudFormation 範本必須`CreateUser`僅針對 部署。

Architecture

目標技術堆疊

- IAM
- AWS CloudTrail
- Amazon CloudWatch Events
- AWS Lambda
- Amazon Simple Storage Service (Amazon S3)

- Amazon SNS

目標架構

自動化和擴展

您可以針對不同的 AWS 區域和帳戶多次使用 AWS CloudFormation 範本。您只需要在每個區域或帳戶中執行一次。若要自動部署到多個帳戶，請使用 AWS CloudFormation StackSets。CloudFormation 範本將能夠在每個帳戶中部署所有必要的資源。

工具

工具

- IAM – AWS Identity and Access Management (IAM) 是一種 Web 服務，可協助您安全地控制對 AWS 資源的存取。您可以使用 IAM 來控制能通過身分驗證 (登入) 和授權使用資源的 (具有許可) 的人員。
- AWS CloudFormation – AWS CloudFormation 可協助您建立模型和設定 Amazon Web Services 資源，以減少管理這些資源的時間，並有更多時間專注於在 AWS 中執行的應用程式。您可以建立範本來描述您想要的所有 AWS 資源，CloudFormation 會為您佈建和設定這些資源。
- AWS CloudTrail – AWS CloudTrail 可協助您管理 AWS 帳戶的控管、合規以及操作和風險稽核。使用者、角色或 AWS 服務採取的動作會在 CloudTrail 中記錄為事件。事件包括在 AWS 管理主控台、AWS 命令列界面，以及 AWS SDKs和 APIs 中採取的動作。
- Amazon CloudWatch Events – Amazon CloudWatch Events 提供near-real-time的系統事件串流，說明 AWS 資源的變更。
- AWS Lambda – AWS Lambda 是一種運算服務，支援執行程式碼，無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是網際網路的儲存體。您可以使用 Amazon S3 隨時從 Web 任何地方存放和擷取任意資料量。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一種受管服務，使用 Lambda、HTTP、電子郵件、行動推播通知和行動文字訊息 (SMS) 提供訊息傳遞。

Code

專案的 .zip 檔案可作為附件使用。

史詩

建立 Lambda 指令碼的 S3 儲存貯體

任務	Description	所需的技能
定義 S3 儲存貯體。	開啟 Amazon S3 主控台，然後選擇或建立 S3 儲存貯體。此 S3 儲存貯體將託管 Lambda 程式碼 .zip 檔案。S3 儲存貯體名稱不能包含正斜線。	雲端架構師

將 Lambda 程式碼上傳至 S3 儲存貯體

任務	Description	所需的技能
上傳 Lambda 程式碼。	將附件區段中提供的 Lambda 程式碼 .zip 檔案上傳至您定義的 S3 儲存貯體。	雲端架構師

部署 CloudFormation 範本

任務	Description	所需的技能
部署 CloudFormation 範本。	在 CloudFormation 主控台上，部署做為此模式附件提供的 CloudFormation `createIAMuser.yaml` 範本。在下一個 Epic 中，提供範本參數的值。	雲端架構師

完成 CloudFormation 範本中的參數

任務	Description	所需的技能
提供 S3 儲存貯體名稱。	輸入您在第一個特徵中建立或選擇的 S3 儲存貯體名稱。	雲端架構師
提供 S3 金鑰。	在您的 S3 儲存貯體中提供 Lambda 程式碼 .zip 檔案的位置，不要加上斜線 （例如 `<directory>/<file-name>.zip` ）。	雲端架構師
提供電子郵件地址。	提供作用中的電子郵件地址以接收 Amazon SNS 通知。	雲端架構師
定義記錄層級。	定義 Lambda 函數的記錄層級和頻率。 會`Info`指定應用程式進度的詳細資訊訊息。 會`Error`指定仍可允許應用程式繼續執行的錯誤事件。 會`Warning`指定可能有害的情況。	雲端架構師

確認訂閱

任務	Description	所需的技能
確認訂閱。	當範本成功部署時，它會傳送訂閱電子郵件訊息到提供的電子郵件地址。若要接收通知，您必須確認此電子郵件訂閱。	雲端架構師

相關資源

- 建立追蹤

- 建立 S3 儲存貯體
- 將檔案上傳至 S3 儲存貯體
- 部署 CloudFormation 範本
- 建立 IAM 使用者
- 使用 AWS CloudTrail 建立在 AWS API 呼叫上觸發的 CloudWatch Events 規則 CloudTrail

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

使用服務控制政策，防止帳戶層級的網際網路存取

Sergiy Shevchenko、Sean O'Sullivan 和 Victor Mazeo Whitaker，Amazon Web Services

總結

組織經常想要限制應保持私有之帳戶資源的網際網路存取。在這些帳戶中，虛擬私有雲端 (VPCs) 中的資源不應以任何方式存取網際網路。許多組織選擇集中式檢查架構。對於集中式檢查架構中的東西 (VPC-to-VPC) 流量，您需要確保輻條帳戶及其資源無法存取網際網路。對於南北（網際網路輸出和內部部署）流量，您想要僅允許透過檢查 VPC 存取網際網路。

此模式使用服務控制政策 (SCP) 來協助防止網際網路存取。您可以在帳戶或組織單位 (OU) 層級套用此 SCP。SCP 會阻止下列項目來限制網際網路連線：

- 建立或連接允許直接網際網路存取 VPC 的 IPv4 或 IPv6 網際網路閘道
- 建立或接受可能允許透過另一個 VPC 間接網際網路存取的 VPC 互連連線
- 建立或更新可能允許直接網際網路存取 VPC 資源的AWS Global Accelerator組態

先決條件和限制

先決條件

- 以組織 AWS 帳戶 身分管理的一或多個 AWS Organizations。
- 在 中啟用所有功能 AWS Organizations。
- SCPs會在組織中啟用。
- 許可：
  - 存取組織的管理帳戶。
  - 建立 SCPs。如需最低許可的詳細資訊，請參閱建立 SCP。
  - 將 SCP 連接到目標帳戶或組織單位 OUs)。如需最低許可的詳細資訊，請參閱連接和分離服務控制政策。

限制

- SCP 不會影響管理帳戶中的使用者或角色。它們只會影響組織中的成員帳戶。
- SCPs僅影響由屬於組織一部分的帳戶管理的 AWS Identity and Access Management (IAM) 使用者和角色。如需詳細資訊，請參閱 SCP 對許可的影響。

工具

AWS 服務

- AWS Organizations 是一種帳戶管理服務，可協助您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。在此模式中，您會在 中使用服務控制政策 SCPs) AWS Organizations。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。

最佳實務

在您的組織中建立此 SCP 之後，請務必經常更新它，以解決可能影響網際網路存取的任何新 AWS 服務 或 功能。

史詩

建立並連接 SCP

任務	Description	所需的技能
建立 SCP。	1. 登入 AWS Organizations 主控台。您必須登入組織的管理帳戶。 2. 在左側窗格中，選擇政策。 3. 在政策頁面上，選擇服務控制政策。 4. 在 Service control policies (服務控制政策) 頁面上，選擇 Create policy (建立政策)。 5. 在建立新的服務控制政策頁面上，輸入政策名稱和選用的政策描述。 6. （選用）將AWS 標籤新增至您的政策。	AWS 管理員

任務	Description	所需的技能
	7. 在 JSON 編輯器中，刪除預留位置政策。 8. 將以下 政策貼到 JSON 編輯器。 `{ "Version": "2012-10-17", "Statement": [{ "Action": ["ec2:AttachInternetGateway", "ec2:CreateInternetGateway", "ec2:CreateVpcPeeringConnection", "ec2:AcceptVpcPeeringConnection", "ec2:CreateEgressOnlyInternetGateway"], "Resource": "*", "Effect": "Deny" }, { "Action": ["globalaccelerator:Create*", "globalaccelerator:Update*"], "Resource": "*", "Effect": "Deny" }`	

任務	Description	所需的技能
	<pre>] }</pre> 9. 選擇建立政策。	
連接 SCP。	1. 在服務控制政策頁面上，選擇您建立的政策。 2. 在 Targets (目標) 索引標籤上，選擇 Attach (連接)。 3. 選取您要連接政策的 OU 或帳戶。您可能需要展開 OUs 才能尋找您想要的 OU 或帳戶。 4. 選擇連接政策。	AWS 管理員

相關資源

- AWS Organizations 文件
- 服務控制政策 (SCP)
- 使用 AWS Gateway Load Balancer 和 （ 部落格文章 ） 的集中式檢查架構 AWS Transit GatewayAWS

使用 PowerShell 匯出 AWS IAM Identity Center 身分及其指派的報告

Jorge Pava、Frank Allotta、Manideep Reddy Gillela 和 ChadMiles、Amazon Web Services

總結

當您使用 AWS IAM Identity Center (AWS Single Sign-On 的後繼者） 集中管理所有 Amazon Web Services (AWS) 帳戶和雲端應用程式的單一登入 (SSO) 存取時，透過 AWS 管理主控台報告和稽核這些指派可能會很繁瑣且耗時。如果您報告使用者或群組在數十個或數百個 AWS 帳戶中的許可，尤其如此。

對於許多使用者而言，檢視此資訊的理想工具會位於試算表應用程式中，例如 Microsoft Excel。這可協助您篩選、搜尋和視覺化整個組織的資料，並由 AWS Organizations 管理。

此模式說明如何使用 AWS Tools for PowerShell 在 IAM Identity Center 中產生 SSO 身分組態的報告。報告會格式化為 CSV 檔案，其中包含身分名稱（委託人）、身分類型（使用者或群組）、身分可存取的帳戶，以及許可集。產生此報告後，您可以在偏好的應用程式中開啟報告，視需要搜尋、篩選和稽核資料。下圖顯示試算表應用程式中的範例資料。

重要：由於此報告包含敏感資訊，因此強烈建議您安全地存放它，並僅在need-to-know的基礎上共用它。

先決條件和限制

先決條件

- IAM Identity Center 和 AWS Organizations，已設定並啟用。
- PowerShell，已安裝並設定。如需詳細資訊，請參閱安裝 PowerShell (Microsoft 文件）。
- 已安裝和設定的 AWS Tools for PowerShell。基於效能考量，強烈建議您安裝名為的 AWS Tools for PowerShell 模組化版本`AWS.Tools`。每個 AWS 服務都由其個別的小型模組支援。在 PowerShell shell 中，輸入下列命令來安裝此模式所需的模組：`AWS.Tools.Installer`、`SSOAdmin`、 `Organizations`和 `IdentityStore`。

```
Install-Module AWS.Tools.Installer
```

```
Install-AWSToolsModule -Name Organizations, SSOAdmin, IdentityStore
```

如需詳細資訊，請參閱在 Windows 上安裝 AWS.Tools 或在 Linux 或 macOS 上安裝 AWS.Tools (AWS Tools for PowerShell 文件）。如果您在安裝模組時收到錯誤，請參閱此模式的故障診斷一節。

- AWS 命令列界面 (AWS CLI) 或 AWS 開發套件先前必須使用工作登入資料進行設定，方法為執行下列其中一項：
  - 使用 AWS CLI `aws configure` 如需詳細資訊，請參閱快速組態 (AWS CLI 文件）。
  - 設定 AWS CLI 或 AWS 雲端開發套件 (AWS CDK)，透過 AWS Identity and Access Management (IAM) 角色取得暫時存取權。如需詳細資訊，請參閱取得 CLI 存取的 IAM 角色登入資料 (IAM Identity Center 文件）。
- AWS CLI 的具名設定檔，其已儲存 IAM 主體的登入資料：
  - 可存取 AWS Organizations 管理帳戶或 IAM Identity Center 的委派管理員帳戶
  - `AWSSSOReadOnly` 和 `AWSSSODirectoryReadOnly` AWS 受管政策是否已套用

  如需詳細資訊，請參閱使用具名設定檔 (AWS CLI 文件） 和 AWS 受管政策 (IAM 文件）。

限制

- 目標 AWS 帳戶必須以 AWS Organizations 中的組織管理。

產品版本

- 對於所有作業系統，建議您使用 PowerShell 7.0 版或更新版本。

Architecture

目標架構

1. 使用者在 PowerShell 命令列中執行指令碼。
2. 指令碼會假設 AWS CLI 的具名設定檔。這會授予 IAM Identity Center 的存取權。
3. 指令碼會從 IAM Identity Center 擷取 SSO 身分組態。
4. 指令碼會在儲存指令碼的本機工作站上的相同目錄中產生 CSV 檔案。

工具

AWS 服務

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- AWS IAM Identity Center 可協助您集中管理所有 AWS 帳戶和雲端應用程式的單一登入 (SSO) 存取。
- AWS Tools for PowerShell 是一組 PowerShell 模組，可協助您從 PowerShell 命令列對 AWS 資源執行指令碼操作。

其他工具

- PowerShell 是在 Windows、Linux 和 macOS 上執行的 Microsoft 自動化和組態管理程式。

史詩

產生報告

任務	Description	所需的技能
準備指令碼。	1. 在此模式的其他資訊區段中複製 PowerShell 指令碼。 2. 在 `Param`區段中，為您的 AWS 環境定義下列變數的值： • `OutputFile` – 報告的檔案名稱。 • `ProfileName` – 您要用來產生報告的 AWS CLI 命名設定檔。 • `Region` – 部署 IAM Identity Center 的 AWS 區域。如需區域及其代碼的完整清單，請參閱區域端點。	雲端管理員

任務	Description	所需的技能
	3. 儲存檔案名稱為 的指令碼SSO-Report.ps1 。	

執行指令碼。	建議您使用下列命令，在 PowerShell shell 中執行自訂指令碼。
	`.\SSO-Report.ps1`
	或者，您也可以輸入下列命令，從另一個 shell 執行指令碼。
	`pwsh .\SSO-Report.ps1`
	指令碼會在與指令碼檔案相同的目錄中產生 CSV 檔案。
		雲端管理員

分析報告資料。	輸出 CSV 檔案具有 AccountName、 PermissionSet、 Principal 和 Type 標頭。在您偏好的試算表應用程式中開啟此檔案。您可以建立資料表來篩選和排序輸出。	雲端管理員

疑難排解

問題	解決方案
The term 'Get-<parameter>' is not recognized as the name of a cmdlet, function, script file, or operable program. 錯誤	未安裝 AWS Tools for PowerShell 或其模組。在 PowerShell shell 中，輸入下列命令來安裝適用於 PowerShell 的 AWS 工具，以及此模式所需的模組：AWS.Tools.Installer 、SSOAdmin、 Organizations 和 IdentityStore 。

問題	解決方案
	`Install-Module AWS.Tools.Installer` `Install-AWSToolsModule -Name Organizations, SSOAdmin, IdentityStore`
`No credentials specified or obtained from persisted/shell defaults` 錯誤	在 Epics 區段的準備指令碼中，確認您已正確輸入 `ProfileName` 和 `Region`變數。確定具名設定檔中的設定和登入資料有足夠的許可來管理 IAM Identity Center。
`Authenticode Issuer …` 安裝 AWS.Tools 模組時發生錯誤	將 `-SkipPublisherCheck` 參數新增至 `Install-AWSToolsModule` 命令的結尾。

`Get-ORGAccountList : Assembly AWSSDK.SSO could not be found or loaded.` 錯誤	指定具名 AWS CLI 設定檔、將 AWS CLI 設定為使用 IAM Identity Center 驗證使用者，並將 AWS CLI 設定為自動擷取重新整理的身分驗證字符時，可能會發生此錯誤。若要解決此錯誤，請執行下列動作：
	1. 輸入下列命令以確認已安裝 SSO和 SSOOIDC模組。
	`Install-AWSToolsModule SSO, SSOOIDC`
	2. 將以下幾行插入`param()`區塊下方的指令碼。
	`Import-Module AWS.Tools.SSO`
	`Import-Module AWS.Tools.SSOOIDC`

相關資源

- 組態設定存放在哪裡？ (AWS CLI 文件)
- 設定 AWS CLI 以使用 AWS IAM Identity Center (AWS CLI 文件)

- 使用具名設定檔 (AWS CLI 文件)

其他資訊

在下列指令碼中，判斷是否需要更新下列參數的值：

- 如果您在 AWS CLI 中使用具名設定檔來存取已設定 IAM Identity Center 的帳戶，請更新 `$ProfileName`值。
- 如果 IAM Identity Center 部署在與您的 AWS CLI 或 AWS SDK 組態預設區域不同的 AWS 區域中，請更新 `$Region`值以使用部署 IAM Identity Center 的區域。
- 如果這兩種情況都不適用，則不需要更新指令碼。

```
param (
    # The name of the output CSV file
    [String] $OutputFile  = "SSO-Assignments.csv",
    # The AWS CLI named profile
    [String] $ProfileName = "",
    # The AWS Region in which IAM Identity Center is configured
    [String] $Region      = ""
)
$Start = Get-Date; $OrgParams = @{}
If ($Region){ $OrgParams.Region = $Region}
if ($ProfileName){$OrgParams.ProfileName = $ProfileName}
$SSOParams   = $OrgParams.Clone(); $IdsParams = $OrgParams.Clone()
$AccountList = Get-ORGAccountList @OrgParams | Select-Object Id, Name
$SSOinstance = Get-SSOADMNInstanceList @OrgParams
$SSOParams['InstanceArn']       = $SSOinstance.InstanceArn
$IdsParams['IdentityStoreId']   = $SSOinstance.IdentityStoreId
$PSsets       = @{}; $Principals   = @{}
$Assignments  = @(); $AccountCount = 1; Write-Host ""
foreach ($Account in $AccountList) {
    $Duration = New-Timespan -Start $Start -End (Get-Date) | ForEach-Object
 {[Timespan]::New($_.Days, $_.Hours, $_.Minutes, $_.Seconds)}
    Write-Host "`r$Duration - Account $AccountCount of $($AccountList.Count)
 (Assignments:$($Assignments.Count))        " -NoNewline
    $AccountCount++
    foreach ($PS in Get-SSOADMNPermissionSetsProvisionedToAccountList -AccountId
 $Account.Id @SSOParams) {
        if (-not $PSsets[$PS]) {$PSsets[$PS] = (Get-SSOADMNPermissionSet @SSOParams -
PermissionSetArn $PS).Name;$APICalls++}
```

```
        $AssignmentsResponse = Get-SSOADMNAccountAssignmentList @SSOParams -
PermissionSetArn $PS -AccountId $Account.Id
        if ($AssignmentsResponse.NextToken) {$AccountAssignments =
 $AssignmentsResponse.AccountAssignments}
        else {$AccountAssignments = $AssignmentsResponse}
        While ($AssignmentsResponse.NextToken) {
            $AssignmentsResponse = Get-SSOADMNAccountAssignmentList @SSOParams -
PermissionSetArn $PS -AccountId $Account.Id -NextToken $AssignmentsResponse.NextToken
            $AccountAssignments += $AssignmentsResponse.AccountAssignments}
        foreach ($Assignment in $AccountAssignments) {
            if (-not $Principals[$Assignment.PrincipalId]) {
                $AssignmentType = $Assignment.PrincipalType.Value
                $Expression     = "Get-IDS"+$AssignmentType+" @IdsParams -"+
$AssignmentType+"Id "+$Assignment.PrincipalId
                $Principal      = Invoke-Expression $Expression
                if ($Assignment.PrincipalType.Value -eq "GROUP")
 { $Principals[$Assignment.PrincipalId] = $Principal.DisplayName }
                else { $Principals[$Assignment.PrincipalId] = $Principal.UserName }
            }
            $Assignments += [PSCustomObject]@{
                AccountName     = $Account.Name
                PermissionSet   = $PSsets[$PS]
                Principal       = $Principals[$Assignment.PrincipalId]
                Type            = $Assignment.PrincipalType.Value}
        }
    }
}
$Duration = New-Timespan -Start $Start -End (Get-Date) | ForEach-Object
 {[Timespan]::New($_.Days, $_.Hours, $_.Minutes, $_.Seconds)}
Write-Host "`r$($AccountList.Count) accounts done in $Duration. Outputting result to
 $OutputFile"
$Assignments | Sort-Object Account | Export-CSV -Path $OutputFile -Force
```

使用 根據 IP 地址或地理位置限制存取 AWS WAF

Louis Hourcade，Amazon Web Services

總結

AWS WAF 是一種 Web 應用程式防火牆，可協助保護 Web 應用程式和 APIs，防範可能影響可用性、危及安全性或消耗過多資源的常見 Web 入侵和機器人。中的 Web 存取控制清單 (Web ACLs) AWS WAF 可讓您控制流量到達應用程式的方式。在 Web ACL 中，您可以新增規則或規則群組，這些規則或規則群組旨在允許合法流量、控制機器人流量，以及封鎖常見的攻擊模式。如需詳細資訊，請參閱如何 AWS WAF 運作。

您可以將下列類型的規則與 AWS WAF Web ACLs建立關聯：

- 受管規則群組 – AWS 受管規則團隊和 AWS Marketplace 賣方提供預先設定的規則集。有些受管規則群組旨在協助保護特定類型的 Web 應用程式。其他則針對已知威脅或常見漏洞提供廣泛的保護。
- 自訂規則和自訂規則群組 – 您也可以建立規則和規則群組，以自訂對 Web 應用程式和 APIs存取。例如，您可以根據特定 IP 地址清單或國家/地區清單來限制流量。

透過使用此模式和相關聯的程式碼儲存庫，您可以使用 AWS Cloud Development Kit (AWS CDK) 部署具有自訂規則的 AWS WAF Web ACL。 ACLs 這些規則會根據最終使用者的 IP 地址或地理位置，限制對 Web 應用程式資源的存取。您也可以選擇性地連接數個受管規則群組。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 部署 AWS WAF 資源的許可
- AWS CDK，在您的帳戶中安裝和設定
- Git，已安裝

限制

- 您只能在 AWS WAF 可用的 AWS 區域 中使用此模式。如需區域可用性，請參閱AWS 服務 依區域。

工具

AWS 服務

- AWS Cloud Development Kit (AWS CDK) 是一種軟體開發架構，可協助您在程式碼中定義和佈建 AWS 雲端 基礎設施。
- AWS WAF 是一種 Web 應用程式防火牆，可協助您監控轉送至受保護 Web 應用程式資源的 HTTP 和 HTTPS 請求。

程式碼儲存庫

此模式的程式碼可在 GitHub IP 和含儲存庫的地理位置限制 AWS WAF中使用。此程式碼會部署兩個 AWS WAF Web ACLs。第一個是區域性 Web ACL，適用於 Amazon API Gateway 資源。第二個是 Amazon CloudFront 資源的全域 Web ACL。兩個 Web ACLs都包含下列自訂規則：

- `IPMatch` 會封鎖來自不允許 IP 地址的請求。
- `GeoMatch` 會封鎖來自不允許國家/地區的請求。

在部署期間，您可以選擇將所有下列受管規則群組連接至 Web ACLs：

- 核心規則集 (CRS) – 此規則群組包含通常適用於 Web 應用程式的規則。它有助於防止各種漏洞遭到利用，包括 OWASP 出版物中所述的一些高風險和常見漏洞，例如 OWASP 前 10 名。
- 管理員保護 – 此規則群組包含的規則可協助您封鎖對公開管理頁面的外部存取。
- 已知錯誤輸入 – 此規則群組有助於封鎖已知無效且與漏洞利用或探索相關聯的請求模式。
- Amazon IP 評價清單 – 此規則群組包含以 Amazon 內部威脅情報為基礎的規則。它可協助您封鎖通常與機器人或其他威脅相關聯的 IP 地址。
- Linux 作業系統受管規則群組 – 此規則群組可協助封鎖與利用 Linux 漏洞相關的請求模式，包括 Linux 特定的本機檔案包含 (LFI) 攻擊。
- SQL 資料庫受管規則群組 – 此規則群組有助於封鎖與利用 SQL 資料庫相關聯的請求模式，例如 SQL Injection 攻擊。

史詩

設定 AWS WAF Web ACLs

任務	Description	所需的技能
複製儲存庫。	輸入下列命令，將 IP 和地理位置限制與 AWS WAF 儲存庫複製到您的本機工作站： `git clone https://github.com/aws-samples/ip-and-geolocation-restriction-with-waf-cdk.git`	Git

設定規則。	1. 在複製的儲存庫中，開啟 app.py 檔案。
	2. 修改下列變數的值以自訂規則：
	`aws_acccount = "AWS_ACCOUNT"`
	`region = "AWS_REGION"`
	`ip_list = ["CIDR_RANGE_1", "CIDR_RANGE_2"]`
	`geo_list = ["COUNTRY_CODE_1", "COUNTRY_CODE_2"]`
	`aws_managed_rules = True`
	其中：
	• `aws_account` 是目標的 ID AWS 帳戶。
		一般 AWS、Python

任務	Description	所需的技能
	• `region` 是 API Gateway 資源 AWS 區域 的 Web ACL 目標。 **Note** CloudFront 資源的 Web ACL 是全域的，並將部署在 `us-east-1` 區域中。 • `ip_list` 是允許存取的 CIDR 範圍清單。 • `geo_list` 是允許存取的國家/地區清單。如需有效值，請參閱 AWS WAF 文件。 • `aws_managed_rules` 控制是否將受管規則群組新增至 Web ACL。如果此值為 `True`，則會新增這些值。如果此值為 `False`，則會排除這些值。 3. 儲存並關閉 app.py 檔案。	

引導和部署程式碼

任務	Description	所需的技能
引導您的 AWS 環境。	如果尚未完成，您需要先引導 AWS 環境，才能部署 AWS CDK 應用程式。	一般 AWS

任務	Description	所需的技能

	1. 在 AWS CDK CLI 中，輸入下列命令來引導 us-east-1 區域：
	`cdk bootstrap aws://<account-id>/us-east-1`
	2. 如果您要在 以外的區域中部署 API Gateway 的 Web ACLus-east-1 ，請輸入下列命令來引導目標區域：
	`cdk bootstrap aws://<account-id>/<region>`	

部署 AWS CDK 應用程式。	1. 輸入下列命令來部署 AWS CDK 應用程式： `cdk deploy --all` 2. 等待 AWS CloudFormation 堆疊部署完成。	一般 AWS

驗證部署

任務	Description	所需的技能
確認 Web ACLs已成功部署。	1. 登入 AWS 管理主控台，然後開啟 AWS WAF 主控台。 2. 在導覽窗格中，選擇 Web ACL。 3. 在 清單中 AWS 區域，選擇全域 (CloudFront)。	一般 AWS

任務	Description	所需的技能
	4. 確認已部署新的 CloudFront Web ACL，並確認它具有您定義的 IP 地址和地理位置規則。此 Web ACL 的預設名稱為 `WebACLCloudfront-<ID>` 。 5. 在 清單中 AWS 區域，選擇您部署堆疊的區域。 6. 確認已部署 API Gateway 資源的新 Web ACL。確認它具有您定義的 IP 地址和地理位置規則。此 Web ACL 的預設名稱為 `WebACLApiGW-<ID>` 。	
（選用）將 Web ACLs 與您的資源建立關聯。	將 AWS WAF Web ACLs 與您的 AWS 資源建立關聯，例如 Application Load Balancer、API Gateway 或 CloudFront 分佈。如需說明，請參閱將 Web ACL 與資源建立關聯或取消關聯AWS。https://docs.aws.amazon.com/waf/latest/developerguide/web-acl-associating-aws-resource.html如需範例，請參閱 AWS CDK 文件中的類別 CfnWebACL Association （建構）。	一般 AWS

清除資源

任務	Description	所需的技能
刪除堆疊。	1. 取消 Web ACL 與任何 AWS 資源的關聯。如需說明，請參閱 AWS WAF 文件。 2. 在 AWS CDK CLI 中，輸入下列命令來刪除 AWS CDK 應用程式。 `cdk destroy --all`	一般 AWS

相關資源

- API 參考 (AWS CDK 文件）
- aws-cdk-lib.aws_wafv2 模組 (AWS CDK 文件）
- 使用 Web ACLs(AWS WAF 文件）
- 管理您自己的規則群組 (AWS WAF 文件）
- 規則 (AWS WAF 文件）

使用 git-secrets 掃描 Git 儲存庫是否有敏感資訊和安全問題

Saurabh Singh，Amazon Web Services

總結

此模式說明如何使用 AWS Labs 的開放原始碼 git-secrets 工具掃描 Git 來源儲存庫，並尋找可能包含敏感資訊的程式碼，例如使用者密碼或 AWS 存取金鑰，或有任何其他安全性問題。

`git-secrets` 掃描遞交、遞交訊息和合併，以防止敏感資訊，例如秘密新增至您的 Git 儲存庫。例如，如果合併歷史記錄中的遞交、遞交訊息或任何遞交符合您其中一個已設定、禁止的規則表達式模式，遞交會遭到拒絕。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 需要安全性掃描的 Git 儲存庫
- 已安裝 Git 用戶端 (2.37.1 版及更新版本）

Architecture

目標架構

- Git
- `git-secrets`

工具

- git-secrets 是一種工具，可防止您將敏感資訊遞交至 Git 儲存庫。
- Git 是一種開放原始碼分散式版本控制系統。

最佳實務

- 一律透過包含所有修訂來掃描 Git 儲存庫：

```
git secrets --scan-history
```

史詩

連線至 EC2 執行個體

任務	Description	所需的技能
使用 SSH 連線至 EC2 執行個體。	使用 SSH 和金鑰對檔案連線至 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。 如果您要掃描本機電腦上的儲存庫，可以略過此步驟。	一般 AWS

安裝 Git

任務	Description	所需的技能
安裝 Git。	使用 命令安裝 Git： `yum install git -y` 如果您使用的是本機電腦，您可以為特定作業系統版本安裝 Git 用戶端。如需詳細資訊，請參閱 Git 網站。	一般 AWS

複製來源儲存庫並安裝 git-secrets

任務	Description	所需的技能
複製 Git 來源儲存庫。	若要複製您要掃描的 Git 儲存庫，請從主目錄中選擇 Git 複製命令。	一般 AWS
複製 git-secrets。	複製 `git-secrets` Git 儲存庫。 `git clone https://github.com/awslabs/git-secrets.git` 將`git-secrets` 某個位置放在 中，PATH 讓 Git 在您執行時收取它`git-secrets` 。	一般 AWS
安裝 git-secrets。	對於 Unix 和變體 (Linux/macOS)： 您可以使用 `install`的目標 `Makefile`（在 `git-secrets` 儲存庫中提供）來安裝工具。您可以使用 `PREFIX`和 `MANPREFIX` 變數來自訂安裝路徑。 `make install` 用於 Windows： 執行儲存`git-secrets` 庫中提供的 PowerShell `install.ps1` 指令碼。此指令碼會將安裝檔案複製到安裝目錄 (`%USERPROFILE`	一般 AWS

任務	Description	所需的技能

	%/.git-secrets 預設為），並將目錄新增至目前的使用者 PATH。
	`PS > ./install.ps1`
	對於 Homebrew (macOS 使用者）：
	執行：
	`brew install git-secrets`	

掃描 git 程式碼儲存庫

任務	Description	所需的技能
前往來源儲存庫。	切換到您要掃描的 Git 儲存庫的目錄： `cd my-git-repository`	一般 AWS
註冊 AWS 規則集 (Git 掛鉤）。	若要設定 git-secrets 在每個遞交上掃描您的 Git 儲存庫，請執行 命令： `git secrets --register-aws`	一般 AWS
掃描儲存庫。	執行下列命令以開始掃描您的儲存庫： `git secrets --scan`	一般 AWS

任務	Description	所需的技能

檢閱輸出檔案。	如果在 Git 儲存庫中找到漏洞，工具會產生輸出檔案。例如：
	
	`example.sh:4:AWS_SECRET_ACCESS_KEY = *********`
	
	`[ERROR] Matched one or more prohibited patterns`
	
	`Possible mitigations:`
	`- Mark false positives as allowed using: git config --add secrets.allowed ...`
	`- Mark false positives as allowed by adding regular expressions to .gitallowed at repository's root directory`
	`- List your configured patterns: git config --get-all secrets.patterns`
	`- List your configured allowed patterns: git config --get-all secrets.allowed`
	`- List your configured allowed patterns in .gitallowed at repository's root directory`
	`- Use --no-verify if this is a one-time false positive`
		一般 AWS

相關資源

- git-secrets 工具

使用 Transfer Family、Amazon Cognito 和 GuardDuty 保護檔案傳輸

Manoj Kumar，Amazon Web Services

總結

此解決方案可協助您使用 透過 SFTP 伺服器安全地傳輸檔案 AWS Transfer Family。其中包括透過 Amazon GuardDuty 功能 Malware Protection for S3 的自動化惡意軟體掃描功能。它專為需要安全地與外部各方交換檔案，並在處理之前驗證所有傳入檔案是否掃描惡意軟體的組織而設計。

此模式隨附的基礎設施即程式碼 (IaC) 範本可協助您部署下列項目：

- 透過 使用 Amazon Cognito 身分驗證的安全 SFTP 伺服器 AWS Lambda
- Amazon Simple Storage Service (Amazon S3) 儲存貯體，用於已掃描惡意軟體的上傳和傳入檔案
- 虛擬私有雲端 (VPC) 型架構，具有跨多個可用區域的公有和私有子網路
- 輸入和輸出流量的 IP 型存取控制，具有可設定的允許和拒絕清單
- 透過 GuardDuty 自動化惡意軟體掃描
- 透過 Amazon EventBridge 和 Lambda 以掃描結果為基礎的智慧型檔案路由
- 透過 Amazon Simple Notification Service (Amazon SNS) 即時通知安全事件
- 透過 AWS Key Management Service (AWS KMS) 加密 Amazon S3 儲存貯體和 Lambda 環境變數
- 無需網際網路暴露即可存取的 Amazon Virtual Private Cloud (Amazon VPC) 端點
- 透過 Amazon CloudWatch 整合進行全面記錄

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- AWS Identity and Access Management (IAM) 中執行此模式中所描述動作的許可，包括部署佈建 IAM 角色的 AWS CloudFormation 範本
- GuardDuty，在目標帳戶中啟用
- S3 的惡意軟體防護，在目標帳戶中啟用
- 服務配額可讓您在目標帳戶中建立下列項目：

  - 一個 VPC
  - 一個私有子網路
  - 一個公有子網路
  - 三個彈性 IP 地址
  - 足夠的 Lambda 並行限制
- 安全相關通知的有效電子郵件地址
- （選用）您要允許或拒絕的 IP 地址或 CIDR 範圍清單
- （選用）AWS Command Line Interface (AWS CLI)，已安裝並設定

限制

- S3 的惡意軟體防護受到配額限制，例如檔案大小上限。如需詳細資訊，請參閱《GuardDuty 文件》中的 S3 惡意軟體防護配額和 Amazon S3 功能的支援能力。
- 此解決方案僅使用 Amazon Cognito 使用者名稱和密碼身分驗證。此範本不支援憑證型或其他身分驗證方法。根據預設，此解決方案不會設定多重要素驗證 (MFA)。
- 解決方案只會透過安全群組實作 IP 型存取控制。

Architecture

下列架構圖顯示在此模式中部署的資源。此解決方案使用 Amazon Cognito 進行使用者身分驗證和授權。 AWS Transfer Family SFTP 伺服器用於檔案上傳。檔案存放在 Amazon S3 儲存貯體中，Amazon GuardDuty 會掃描檔案是否有惡意軟體。如果偵測到惡意軟體，Amazon SNS 會傳送電子郵件通知。

該圖顯示以下工作流程：

1. 使用者連線到其中的 SFTP 伺服器端點 AWS Transfer Family。這會使用 Amazon Cognito 使用者集區啟動身分驗證程序。
2. Lambda 函數會啟動身分驗證和授權程序，並使用 Amazon Cognito 驗證使用者的登入資料。
3. Lambda 函數會傳回 `UploadBucket` Amazon S3 儲存貯體做為主目錄。使用者擔任 Transfer Family 伺服器的 IAM 角色，Lambda 函數會通知使用者他們已成功驗證。
4. 使用者將檔案上傳至 Transfer Family SFTP 伺服器。檔案會存放在 `UploadBucket` Amazon S3 儲存貯體中。

5. GuardDuty 會掃描檔案是否有惡意軟體。潛在的掃描結果為 NO_THREATS_FOUND、THREATS_FOUND、ACCESS_DENIED、 UNSUPPORTED和 FAILED。如需範例結果，請參閱 GuardDuty 文件中的 S3 物件掃描結果。
6. EventBridge 規則會偵測掃描結果事件。
7. EventBridge 會啟動檔案路由 Lambda 函數。
8. Lambda 函數會處理事件，並根據掃描結果篩選檔案，如下所示：
   - 具有NO_THREATS_FOUND掃描結果的檔案會傳送至 CleanBucket Amazon S3 儲存貯體。
   - 具有THREATS_FOUND掃描結果的檔案會傳送至 MalwareBucket Amazon S3 儲存貯體。
   - 具有UNSUPPORTED掃描結果的檔案會傳送至 ErrorBucket Amazon S3 儲存貯體。
   - 具有ACCESS_DENIED掃描結果的檔案會傳送至 ErrorBucket Amazon S3 儲存貯體。
   - 具有FAILED掃描結果的檔案會傳送至 ErrorBucket Amazon S3 儲存貯體。

   所有檔案都會使用 加密 AWS KMS key。
9. 如果檔案已傳送至 MalwareBucket Amazon S3 儲存貯體，Lambda 函數會啟動 Amazon SNS 主題。Amazon SNS 主題會將電子郵件通知傳送至您設定的電子郵件地址。

工具

AWS 服務

- Amazon CloudWatch 可協助您 AWS 即時監控 AWS 資源的指標，以及您在 上執行的應用程式。
- Amazon Cognito 為 Web 和行動應用程式提供身分驗證、授權和使用者管理。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如， AWS Lambda 函數、使用 API 目的地的 HTTP 調用端點，或其他事件匯流排 AWS 帳戶。
- Amazon GuardDuty 是一種持續的安全監控服務，可分析和處理日誌，以識別您 AWS 環境中非預期和可能未經授權的活動。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。

- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS Transfer Family 可協助您透過 SFTP、FTPS 或 FTP 通訊協定將檔案傳入和傳出 AWS 儲存服務。
- Amazon Virtual Private Cloud (Amazon VPC) 可協助您在已定義的虛擬網路中啟動 AWS 資源。此虛擬網路與您在自己的資料中心中操作的傳統網路相似，且具備使用 AWS可擴展基礎設施的優勢。

程式碼儲存庫

此模式的程式碼可在 GitHub AWS Transfer Family 和 GuardDuty 惡意軟體掃描解決方案儲存庫中使用。

最佳實務

提供的 CloudFormation 範本旨在整合許多 AWS 最佳實務，例如 IAM 角色和政策的最低權限許可、靜態和傳輸中的加密，以及自動金鑰輪換。對於生產環境，請考慮實作下列其他建議：

- 為 Amazon Cognito 使用者啟用 MFA
- 實作AWS Shield分散式拒絕服務 (DDoS) 保護
- 設定 AWS Config以進行持續合規監控
- 實作 AWS CloudTrail進行全面的 API 記錄
- 設定 Amazon GuardDuty 進行惡意軟體掃描以外的威脅偵測
- 實作 AWS Security Hub CSPM以進行集中式安全管理
- 使用 AWS Secrets Manager 進行登入資料管理
- 使用流量鏡像實作網路流量監控
- 為 Amazon S3 中的敏感資料探索和保護設定 Amazon Macie Amazon S3
- 實作定期安全性評估和滲透測試
- 建立正式的事件回應計劃
- 實作所有元件的自動化修補
- 為管理員執行定期安全訓練
- AWS Organizations 設定多帳戶安全管理

史詩

部署 資源

任務	Description	所需的技能
複製儲存庫。	輸入下列命令，將 AWS Transfer Family 和 GuardDuty 惡意軟體掃描解決方案儲存庫複製到您的本機工作站： `git clone https://github.com/aws-samples/sample-secure-transfer-family-code.git`	應用程式開發人員、DevOps 工程師
建立 CloudFormation 堆疊。	1. 遵循 CloudFormation 文件中的指示，透過 CloudFormation 主控台或透過 建立堆疊AWS CLI。設定下列參數： • `UploadBucketName` - 上傳儲存貯體名稱 • `CleanBucketName` - 清除檔案儲存貯體名稱 • `MalwareBucketName` - 惡意軟體檔案儲存貯體名稱 • `ErrorBucketName` - 錯誤檔案儲存貯體名稱 • `SecurityTeamEmail` - 提醒的電子郵件地址 • `VpcCIDR` - VPC 的 CIDR 範圍	雲端管理員、DevOps 工程師

任務	Description	所需的技能
	• PublicSubnet1CIDR - 公有子網路 1 的 CIDR 範圍 • PublicSubnet2CIDR - 公有子網路 2 的 CIDR 範圍 • PrivateSubnet1CIDR - 私有子網路 1 的 CIDR 範圍 • PrivateSubnet2CIDR - 私有子網路 2 的 CIDR 範圍 • AllowedIPAddresses - 允許 IP 地址或 CIDR 範圍 • EnableEgressRules – 如果您想要啟用安全群組的輸出規則，請輸入 true。如果您想要停用輸出規則，請輸入 false。 2. 等待堆疊部署完成。 3. 在堆疊建立輸出中，記下 SFTP 伺服器端點。您稍後將需要這個值。	

設定 資源

任務	Description	所需的技能
開啟惡意軟體保護。	1. 登入 AWS 管理主控台 並開啟 GuardDuty 主控台。	雲端管理員、AWS 管理員

任務	Description	所需的技能
	2. 在導覽窗格中，選擇 S3 的惡意軟體防護。 3. 在受保護的儲存貯體區段中，選擇啟用。 4. 在輸入 S3 儲存貯體詳細資訊下，輸入 Amazon S3 儲存貯體名稱。或者，選擇瀏覽 S3 以選取儲存貯體。選擇 `UploadBucket` Amazon S3 儲存貯體。 5. 在字首下，選取 S3 儲存貯體中的所有物件。 6. 在標記掃描的物件下，選取標記物件。 7. 在服務存取區段中，選擇建立並使用新的服務角色。 8. 選擇啟用。	
將使用者新增至使用者集區。	將一或多個使用者新增至 Amazon Cognito 使用者集區。如需說明，請參閱 Amazon Cognito 文件中的管理使用者集區中的使用者。 Amazon Cognito	雲端管理員、AWS 管理員

測試 SFTP 伺服器

任務	Description	所需的技能
連線至 SFTP 伺服器端點。	1. 輸入下列命令以取得 SFTP 伺服器端點，其中 `<stack-name>` 是 CloudFormation 堆疊的名稱：	應用程式開發人員、雲端管理員、雲端架構師、DevOps 工程師

任務	Description	所需的技能
	``` TRANSFER_ENDPOINT= $(aws cloudformation describe-stacks \ --stack-name <stack-name> \ --query 'Stacks[0].Outputs[?OutputK ey==`TransferServe rEndpoint`].Output Value' \ --output text) ``` 2. 輸入下列命令以連線至 SFTP 伺服器： ``` sftp user@comp any.com@$TRANSFER_ ENDPOINT ``` 3. 驗證您收到下列提示： ``` sftp> ``` 這表示您現在可以輸入命令來執行檔案傳輸。	

疑難排解

問題	解決方案
使用者身分驗證失敗	1. 確認使用者存在於 Amazon Cognito 使用者集區中。 2. 驗證使用者已確認並啟用。 3. 確定使用者提供正確的密碼。 4. 檢查 CloudWatch Logs 中是否有錯誤。

問題	解決方案
	如需可協助您執行這些故障診斷步驟的 AWS CLI 命令清單，請參閱其他資訊一節中的用於故障診斷的實用命令。
SFTP 身分驗證失敗	1. 確認使用者存在於 Amazon Cognito 使用者集區中。 2. 驗證使用者已確認並啟用。 3. 確定使用者提供正確的密碼。 4. 確認 AWS Transfer Family 伺服器的安全群組中允許 IP 地址。 5. 檢查身分驗證 Lambda 函數日誌中的錯誤。 6. 確認 的 IAM 角色 AWS Transfer Family 具有正確的許可。 如需可協助您執行這些故障診斷步驟的 AWS CLI 命令清單，請參閱其他資訊一節中的用於故障診斷的實用命令。
檔案上傳存取遭拒	1. 確認 AWS KMS 金鑰許可已正確設定。 2. 檢查 Amazon S3 儲存貯體政策。 3. 確認已正確設定 IAM 角色許可。 4. 確認 VPC 端點已正確設定 5. 確認 的 IAM 角色 AWS Transfer Family 具有正確的許可。 如需可協助您執行這些故障診斷步驟的 AWS CLI 命令清單，請參閱其他資訊一節中的用於故障診斷的實用命令。

問題	解決方案
無惡意軟體掃描	1. 在帳戶中啟用 GuardDuty。 2. 啟用 S3 的惡意軟體防護。 3. 確認 Amazon EventBridge 規則處於作用中狀態。 4. 檢查檔案路由 Lambda 函數日誌中的錯誤。 5. 確認已在 Amazon SNS 中正確設定通知。 如需可協助您執行這些故障診斷步驟的 AWS CLI 命令清單，請參閱其他資訊區段中的用於故障診斷的實用命令。
Lambda 函數錯誤	1. 檢查 VPC 組態。 2. 確認 VPC 端點已正確設定。 3. 確認 的 IAM 角色 AWS Transfer Family 具有正確的許可。 4. 檢查 CloudWatch Logs 中是否有錯誤。 5. 確認 AWS KMS 金鑰許可已正確設定。 如需可協助您執行這些故障診斷步驟的 AWS CLI 命令清單，請參閱其他資訊區段中的用於故障診斷的實用命令。

相關資源

- Transfer Family Web 應用程式 (Transfer Family 文件)

其他資訊

用於故障診斷的實用命令

檢查 CloudFormation 堆疊的狀態：

```
aws cloudformation describe-stacks \
  --stack-name <STACK_NAME>
```

列出 Amazon Cognito 使用者集區中的所有使用者：

```
aws cognito-idp list-users \
  --user-pool-id <USER_POOL_ID>
```

檢視 Lambda 函數的日誌：

```
aws logs describe-log-groups \
  --log-group-name-prefix /aws/lambda/
```

檢查 GuardDuty 的狀態：

```
aws guardduty list-detectors
```

檢查安全群組規則：

```
aws ec2 describe-security-groups \
  --group-ids <SECURITY_GROUP_ID> \
  --output table
```

檢查 AWS Transfer Family 伺服器的狀態：

```
aws transfer describe-server \
  --server-id <SERVER_ID>
```

列出 Amazon S3 儲存貯體中的所有檔案：

```
aws s3 ls s3://<BUCKET_NAME>/ \
  --recursive
```

檢查 EventBridge 規則的狀態：

```
aws events describe-rule \
  --name <RULE_NAME>
```

使用 Amazon Macie 保護 CloudWatch Logs 中的敏感資料

Anisha Salunkhe、Omar Franco 和 David Guardiola，Amazon Web Services

總結

此模式說明如何使用 Amazon Macie 透過實作全面的安全監控工作流程，自動偵測 Amazon CloudWatch Logs 日誌群組中的敏感資料。解決方案使用 Amazon Data Firehose 將 CloudWatch Logs 項目串流至 Amazon Simple Storage Service (Amazon S3)。Macie 會定期掃描此儲存貯體是否有個人身分識別資訊 (PII)、財務資料和其他敏感內容。基礎設施是透過佈建所有必要 AWS 服務 和 組態的 AWS CloudFormation 範本進行部署。

CloudWatch Logs 通常包含的應用程式資料可能會不小心包含敏感的使用者資訊。這可能會產生合規和安全性風險。傳統日誌監控方法缺乏自動化敏感資料偵測功能。這可能會導致難以即時識別和回應潛在的資料暴露。

此模式透過自動偵測和提醒記錄系統中的敏感資料，協助安全團隊和合規主管維護資料機密性。此解決方案可透過 Amazon Simple Notification Service (Amazon SNS) 通知啟用主動事件回應，並自動將敏感資料隔離到安全的 Amazon S3 儲存貯體。您可以自訂偵測模式，並將工作流程與您現有的安全操作程序整合。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 建立 CloudFormation 堆疊的許可
- 您要監控的 CloudWatch Logs 日誌群組
- 接收來自 Amazon SNS 通知的作用中電子郵件地址
- AWS CloudShell 的存取權限
- （選用）存取已安裝和設定的 AWS Command Line Interface (AWS CLI)

限制

- Macie 受限於服務配額。如需詳細資訊，請參閱 Macie 文件中的 Macie 配額。

Architecture

目標架構

下圖顯示使用 Macie 檢查 CloudWatch Logs 日誌項目是否有敏感資料的工作流程。

工作流程會顯示下列步驟：

1. CloudWatch Logs 日誌群組會產生受訂閱篩選條件約束的日誌。
2. 訂閱篩選條件會將日誌轉送至 Amazon Data Firehose。
3. 當日誌通過 Amazon Data Firehose 交付串流時，會使用 AWS Key Management Service (AWS KMS) 金鑰加密日誌。
4. 交付串流會將日誌交付至 Amazon S3 中匯出的日誌儲存貯體。
5. Amazon EventBridge 會在每天凌晨 4 點啟動 AWS Lambda 函數，以啟動 Macie 掃描匯出日誌儲存貯體中的敏感資料。
6. 如果 Macie 識別儲存貯體中的敏感資料，Lambda 函數會從匯出的日誌儲存貯體中移除日誌，並使用 加密日誌 AWS KMS key。
7. Lambda 函數會隔離包含資料隔離儲存貯體中敏感資料的日誌。
8. 敏感資料的識別會啟動 Amazon SNS 主題。
9. Amazon SNS 會傳送電子郵件通知至您設定的電子郵件地址，其中包含敏感資料日誌的相關資訊。

部署的資源

CloudFormation 範本會在您的目標和 中部署下列資源 AWS 帳戶 AWS 區域：

- 兩個 Amazon S3 儲存貯體：
  - 用於存放 CloudWatch Logs 資料的匯出日誌儲存貯體
  - 用於存放敏感資訊的資料隔離儲存貯體
- 回應 Macie 調查結果的 Amazon EventBridge 規則
- 啟動事件並將日誌匯出至 Amazon S3 儲存貯體的AWS Lambda 函數
- Amazon SNS 主題和訂閱
- Amazon Data Firehose 串流

- Macie 工作階段
- Macie 自訂資料識別符
- CloudWatch Logs 訂閱篩選條件
- AWS KMS 金鑰，用於加密存放在儲存貯體中的日誌
- 解決方案的必要 AWS Identity and Access Management (IAM) 角色和政策

工具

AWS 服務

- AWS CloudFormation 可協助您設定 AWS 資源、快速且一致地佈建資源，以及在整個 AWS 帳戶 和生命週期中管理資源 AWS 區域。
- Amazon CloudWatch Logs 可協助您集中所有系統、應用程式的日誌， AWS 服務 以便您可以監控日誌並將其安全地存檔。
- Amazon Data Firehose 可協助您將即時串流資料交付至其他 AWS 服務自訂 HTTP 端點，以及受支援的第三方服務供應商所擁有的 HTTP 端點。
- Amazon EventBridge 是一種無伺服器事件匯流排服務，可協助您將應用程式與來自各種來源的即時資料連線。例如， AWS Lambda 函數、使用 API 目的地的 HTTP 調用端點或其他事件匯流排等來源 AWS 帳戶。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以協助保護您的資料。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需按使用的運算時間付費。
- Amazon Macie 可協助您探索敏感資料、提供資料安全風險的可見性，並實現對這些風險的自動化保護。
- Amazon Simple Notification Service (Amazon SNS) 可協助您協調和管理發佈者和用戶端之間的訊息交換，包括 Web 伺服器和電子郵件地址。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。

程式碼儲存庫

此模式的程式碼可在 GitHub sample-macie-for-securing-cloudwatch-logs 儲存庫中使用。

最佳實務

遵循 CloudFormation 文件中的CloudFormation 最佳實務。

史詩

部署解決方案

任務	Description	所需的技能
複製程式碼儲存庫。	輸入下列命令，將儲存庫複製到本機工作站： `git clone https://github.com/aws-samples/sample-macie-for-securing-cloudwatch-logs`	應用程式開發人員
（選用）編輯 CloudFormation 範本。	1. 開啟 main.yaml 檔案。 2. 執行下列任一動作來自訂範本： • 您可以重新命名資源。 • 在 `parameter` 區段中，您可以修改預設值。 • 您可以變更訂閱篩選條件模式。如需詳細資訊，請參閱 CloudWatch Logs 文件中的日誌群組層級訂閱篩選條件。 3. 儲存並關閉 main.yaml 檔案。	應用程式開發人員
選項 1 – 使用具有命令列參數的指令碼進行部署。	輸入下列命令，使用命令列參數來部署解決方案，其中 的值`true`僅在尚未啟用 Amazon	一般 AWS

任務	Description	所需的技能
	Macie 時enable-macie 為： `./scripts/test-macie-solution.sh --deploy-stack \ --stack-name <stack name> \ --email <email address> \ --enable-macie <true or false> \ --region <region> \ --resource-name <prefix for all resources> \ --bucket-name <bucket name>`	

任務	Description	所需的技能

選項 2 – 使用具有環境變數的指令碼進行部署。	1. 輸入下列命令來定義環境變數，其中 的值`true`僅在尚未啟用 Amazon Macie 時`ENABLE_MACIE` 為 ：
	`export STACK_NAME=<stack name>`
	`export SNS_EMAIL=<email address>`
	`export ENABLE_MACIE=<true or false>`
	`export REGION=<region>`
	`export RESOURCE_NAME=<prefix for all resources>`
	`export BUCKET_NAME=<bucket name>`
	2. 輸入下列命令以在部署之前驗證參數：
	`./scripts/test-macie-solution.sh \`
	`--validate-params \`
	`--email <email address> \`
	`--region <region>`
	3. 輸入下列命令來部署解決方案：
	`./scripts/test-macie-solution.sh --deploy-stack`
		一般 AWS

任務	Description	所需的技能
選項 3 – 使用 部署 AWS CLI。	輸入下列命令，使用 來部署解決方案 AWS CLI，其中 的值`true`僅在尚未啟用 Amazon Macie 時`EnableMacie` 為 ： `aws cloudformation create-stack \ --region us-east-1 \ --stack-name macie-for-securing-cloudwatch-logs \ --template-body file://app/main.yml \ --capabilities CAPABILITY_IAM \ --parameters \ ParameterKey=ResourceName,ParameterValue=<prefix for all resources> \ ParameterKey=BucketName,ParameterValue=<bucket name> \ ParameterKey=LogGroupName,ParameterValue=<path for log group> \ ParameterKey=SNSTopicEndpointEmail,ParameterValue=<email address> \ ParameterKey=EnableMacie,ParameterValue=<true or false>`	

任務	Description	所需的技能
選項 4 – 透過 部署 AWS 管理主控台。	1. 開啟 AWS CloudFormation 主控台。 2. 在畫面頂端的導覽列上，選擇要在其中 AWS 區域 建立堆疊的 。 3. 在堆疊頁面的右上角，選擇建立堆疊，並選擇使用新資源 (標準)。 4. 在建立堆疊頁面上，針對先決條件 - 準備範本，選擇選擇現有範本。 5. 在指定範本下，選擇上傳範本檔案，然後從您複製的儲存庫上傳 main.yaml 範本。 6. 選擇下一步。 7. 在指定堆疊詳細資訊頁面上的堆疊名稱方塊中，輸入堆疊名稱。 8. 在參數區段中，指定下列範本參數的值。 • `ResourceName` ：所有資源的字首 • `BucketName` ：Amazon S3 儲存貯體的唯一名稱 • `LogGroupName` ：CloudWatch Logs 的日誌群組名稱 • `SNSTopicEndpointEmail` ：通知的電子郵件地址	一般 AWS

任務	Description	所需的技能
	• `EnableMacie` ：`true`如果 Macie 尚未啟用，則設定為 • （選用） `Region`： AWS 區域 您要部署堆疊的 。 • （選用） `TemplatePath` ： CloudFormation 範本的路徑 9. 選擇下一步。 10針對功能，選擇我確認此範本可能會建立 IAM 資源，以指定您要在範本中使用 IAM 資源。 11選擇下一步。 12.在檢閱並建立頁面中，檢閱堆疊詳細資訊。 13選擇提交以啟動堆疊。	

任務	Description	所需的技能
監控部署狀態並確認部署。	1. 輸入下列命令來監控部署狀態： `./scripts/test-macie-solution.sh \ --deployment-status \ --stack-name <stack name>` **Note** 您也可以在新堆疊的事件索引標籤上監控堆疊建立的進度和狀態。如需詳細資訊，請參閱監控堆疊進度。 2. 當狀態變更為 CREATE_COMPLETE 時，請檢閱資源資訊的堆疊輸出。	一般 AWS
確認 Amazon SNS 訂閱。	請遵循 Amazon SNS 文件中的確認 Amazon SNS 訂閱中的指示，以確認您的 Amazon SNS 訂閱。	應用程式開發人員

測試解決方案

任務	Description	所需的技能
選項 1 – 使用自動報告進行測試。	如果您使用預設堆疊名稱，請輸入下列命令來測試解決方案：	一般 AWS

任務	Description	所需的技能

	<pre>./scripts/test-macie-solution.sh \
	   --full-test</pre>
	如果您使用自訂堆疊名稱，請輸入下列命令來測試解決方案：
	<pre>./scripts/test-macie-solution.sh \
	   --full-test \
	   --stack-name <stack name></pre>
	如果您使用自訂堆疊名稱和自訂參數，請輸入下列命令來測試解決方案：
	<pre>./scripts/test-macie-solution.sh --full-test \
	  --stack-name <stack name> \
	  --region <region> \
	  --log-group <log group path></pre>	

任務	Description	所需的技能

選項 2 – 使用目標驗證進行測試。	1. 輸入下列命令以使用敏感資訊產生測試資料：
	
	`./scripts/test-macie-solution.sh \ --generate-test-data \ --stack-name <stack name>`
	
	此命令會執行下列動作：
	• 建立包含實際敏感資料模式的 CloudWatch Logs 日誌項目，包括員工 IDs、專利 IDs、信用卡號碼、社會安全號碼和電子郵件地址
	• 產生敏感和非敏感日誌項目以進行全面測試
	• 提供測試資料產生程序的詳細記錄
	2. 輸入下列命令來驗證資料管道流程：
	
	`./scripts/test-macie-solution.sh \ --verify-pipeline \ --stack-name <stack name>`
	
	此命令會執行下列動作：
	• 確認 CloudWatch Logs 項目已串流至 Amazon Data Firehose
		一般 AWS

任務	Description	所需的技能

	• 使用適當的加密來驗證日誌資料是否已交付至 Amazon S3 儲存貯體
	• 檢查 Amazon S3 物件儲存體是否具有正確的字首結構
	• 驗證儲存物件的加密狀態
	• 監控資料流程計時，並提供處理等待期
	3. 輸入下列命令以啟動 Macie 分類任務：
	`./scripts/test-macie-solution.sh \ --trigger-macie-job \ --stack-name <stack name>`
	此命令會執行下列動作：
	• 透過 Lambda 函數手動觸發 Macie 分類任務
	• 監控任務執行狀態並提供意見回饋
	• 在執行之前驗證 Macie 服務可用性
	• 處理帳戶中未啟用 Macie 的情況
	• 提供詳細的任務執行結果
	4. 輸入下列命令來驗證警示和資料隔離：
	`./scripts/test-macie-solution.sh \`	

任務	Description	所需的技能
	--verify-alerts \ --stack-name <stack name> 此命令會執行下列動作： • 確認 EventBridge 規則已正確設定並處於作用中狀態 • 驗證 Amazon SNS 主題組態和訂閱狀態 • 檢查資料隔離儲存貯體的設定和許可 • 監控敏感資料移至隔離儲存貯體的情況 • 驗證完整的警示工作流程功能	

清除

任務	Description	所需的技能
選項 1 – 執行自動清除。	如果您使用預設堆疊名稱，請輸入下列命令來刪除堆疊： ./scripts/cleanup-macie-solution.sh \ --full-cleanup 如果您使用自訂堆疊名稱，請輸入下列命令來刪除堆疊： ./scripts/cleanup-macie-solution.sh \ --full-cleanup \	一般 AWS

任務	Description	所需的技能
	--stack-name <stack name> 如果您使用自訂堆疊名稱和自訂參數，請輸入下列命令來刪除堆疊： ./scripts/cleanup-macie-solution.sh \ --full-cleanup \ --stack-name <stack name> \ --region <region> \ --disable-macie <true or false>	

任務	Description	所需的技能

選項 2 – step-by-step清除。	1. 輸入下列命令以停止作用中的程序：
	
	`./scripts/cleanup-macie-solution.sh \ --stop-processes \ --stack-name <stack name>`
	
	此命令會執行下列動作：
	• 關閉 EventBridge 規則以防止新任務執行
	• 停止任何目前正在執行的 Macie 分類任務
	• 取消待處理的 Macie 任務執行
	• 清除佇列中任何待處理的 Amazon SNS 訊息
	• 提供每個已停止程序的狀態更新
	2. 輸入下列命令以清空 Amazon S3 儲存貯體：
	
	`./scripts/cleanup-macie-solution.sh \ --empty-buckets \ --stack-name <stack name>`
	
	此命令會執行下列動作：
	• 從 CloudWatch Logs 的儲存貯體移除所有物件
	• 從資料隔離 Amazon S3 儲存貯體移除所有物件
		一般 AWS

任務	Description	所需的技能

	• 刪除任何未完成的分段上傳
	• 如果啟用 Amazon S3 版本控制，則處理版本控制的物件
	• 提供物件計數和刪除進度更新
	3. 輸入下列命令來刪除 CloudFormation 堆疊：
	`./scripts/cleanup-macie-solution.sh \ --delete-stack \ --stack-name <stack name>`
	此命令會執行下列動作：
	• 啟動 CloudFormation 堆疊刪除程序
	• 使用即時狀態更新監控刪除進度
	• 使用詳細的錯誤報告來處理刪除失敗
	• 等待完全移除堆疊後再繼續
	• 提供堆疊事件歷史記錄以進行故障診斷
	4. 輸入下列命令以清除 Macie 資源：
	`./scripts/cleanup-macie-solution.sh \ --cleanup-macie \`	

任務	Description	所需的技能
	<pre>--stack-name <stack name></pre> 此堆疊會執行下列動作： • 移除解決方案建立的自訂資料識別符 • 清除任何剩餘的 Macie 任務成品和調查結果 • 停用由堆疊啟用的 Macie 工作階段 • 處理 Macie 資源與其他應用程式共用的情況 • 提供每個 Macie 元件的詳細清除狀態	

任務	Description	所需的技能

驗證清除。	1. 輸入下列命令以確認堆疊已刪除：
	`aws cloudformation describe-stacks \ --stack-name <stack name> \ --region <region>`
	2. 輸入下列命令以確認已刪除 Amazon S3 儲存貯體：
	`aws s3 ls | grep macie`
	3. 輸入下列命令來驗證 Macie 自訂資料識別符是否已移除：
	`aws macie2 list-custom-data-identifiers \ --region <region>`
	4. 輸入下列命令來檢查是否有任何剩餘的資源：
	`./scripts/cleanup-macie-solution.sh \ --verify-cleanup \ --stack-name <stack name>`
		一般 AWS

疑難排解

問題	解決方案
CloudFormation 堆疊狀態顯示 CREATE_FAILED。	CloudFormation 範本設定為將日誌發佈至 CloudWatch Logs。您可以在 中檢視日誌，AWS 管理主控台 這樣您就不必連線到 Amazon EC2 執行個體。如需詳細資訊，請參閱在 主控台中檢視 CloudFormation 日誌 (AWS 部落格文章）。
CloudFormation `delete-stack` 命令失敗。	某些資源在刪除之前必須先清空。例如，您必須刪除 Amazon S3 儲存貯體中的所有物件或移除 Amazon EC2 安全群組中的所有執行個體，才能刪除該儲存貯體或安全群組。如需詳細資訊，請參閱 Amazon S3 文件中的刪除堆疊失敗。
剖析參數時發生錯誤。	當您使用 AWS CLI 或 CloudFormation 主控台傳入值時，請新增引號。

相關資源

- 儲存體的架構最佳實務 (AWS 網站）
- 指標篩選條件、訂閱篩選條件、篩選條件日誌事件和 Live Tail 的篩選條件模式語法 (CloudWatch Logs 文件）
- 使用 Amazon CloudWatch 設計和實作記錄和監控 (AWS 方案指引）
- 故障診斷 CloudFormation(CloudFormation 文件）

從 AWS Network Firewall 傳送提醒到 Slack 頻道

Venki Srivatsav 和 Aromal Raj Jayarajan，Amazon Web Services

總結

此模式說明如何使用 Amazon Web Services (AWS) Network Firewall 搭配分散式部署模型來部署防火牆，以及如何將 AWS Network Firewall 產生的警示傳播到可設定的 Slack 頻道。

支付卡產業資料安全標準 (PCI DSS) 等合規標準要求您安裝和維護防火牆以保護客戶資料。在 AWS 雲端中，在這些合規要求的內容中，虛擬私有雲端 (VPC) 被視為與實體網路相同。您可以使用 Network Firewall 來監控 VPCs和 之間的網路流量，以保護在受合規標準規範VPCs 中執行的工作負載。Network Firewall 會在偵測到來自相同帳戶中其他 VPCs的未經授權存取時封鎖存取或產生提醒。不過，Network Firewall 支援有限數量的目的地來傳送提醒。這些目的地包括 Amazon Simple Storage Service (Amazon S3) 儲存貯體、Amazon CloudWatch 日誌群組和 Amazon Data Firehose 交付串流。這些通知的任何進一步動作都需要使用 Amazon Athena 或 Amazon Kinesis 進行離線分析。

此模式提供一種方法，可將 Network Firewall 產生的警示傳播到可設定的 Slack 頻道，以近乎即時的方式執行進一步的動作。您也可以將功能擴展到其他提醒機制，例如 PagerDuty、Jira 和電子郵件。（這些自訂超出此模式的範圍。）

先決條件和限制

先決條件

- Slack 頻道（請參閱 Slack 說明中心入門）
- 傳送訊息至頻道所需的權限
- 具有 API 字符的 Slack 端點 URL (選取您的應用程式，然後選擇傳入 Webhook 以查看其 URL；如需詳細資訊，請參閱 Slack API 文件中的建立傳入 Webhook)
- 工作負載子網路中的 Amazon Elastic Compute Cloud (Amazon EC2) 測試執行個體
- Network Firewall 中的測試規則
- 用來觸發測試規則的實際或模擬流量
- 存放要部署之來源檔案的 S3 儲存貯體

限制

- 目前，此解決方案僅支援單一無類別網域間路由 (CIDR) 範圍做為來源和目的地 IPs的篩選條件。

Architecture

目標技術堆疊

- 一個 VPC
- 四個子網路（兩個用於防火牆，兩個用於工作負載）
- 網際網路閘道
- 四個具有規則的路由表
- 做為提醒目的地的 S3 儲存貯體，設定儲存貯體政策和事件設定以執行 Lambda 函數
- 具有 執行角色的 Lambda 函數，用於傳送 Slack 通知
- 儲存 Slack URL 的 AWS Secrets Manager 秘密
- 具有警示組態的網路防火牆
- Slack 頻道

除了 Slack 頻道之外的所有元件都由 CloudFormation 範本和此模式隨附的 Lambda 函數佈建（請參閱程式碼區段）。

目標架構

此模式會設定具有 Slack 整合的分散式網路防火牆。此架構包含有兩個可用區域的 VPC。VPC 包含兩個受保護的子網路和兩個具有網路防火牆端點的防火牆子網路。透過建立防火牆政策和規則，即可監控進出受保護子網路的所有流量。網路防火牆設定為將所有警示放在 S3 儲存貯體中。此 S3 儲存貯體設定為在收到put事件時呼叫 Lambda 函數。Lambda 函數會從 Secrets Manager 擷取設定的 Slack URL，並將通知訊息傳送至 Slack 工作區。

如需此架構的詳細資訊，請參閱 AWS Network Firewall 的 AWS 部落格 postDeployment 模型。

工具

AWS 服務

- AWS Network Firewall 是 AWS 雲端中 VPCs 具狀態、受管的網路防火牆和入侵偵測和預防服務。您可以使用 Network Firewall 來篩選 VPC 周邊的流量，並保護 AWS 上的工作負載。
- AWS Secrets Manager 是一種用於憑證儲存和擷取的服務。使用 Secrets Manager，您可以使用以程式設計方式呼叫 Secrets Manager 擷取秘密的 API，取代程式碼中的硬式編碼登入資料，包括密碼。此模式使用 Secrets Manager 來存放 Slack URL。

- Amazon Simple Storage Service (Amazon S3) 是一種物件儲存服務。您可以使用 Amazon S3 隨時從 Web 任何地方存放和擷取任意資料量。此模式使用 Amazon S3 來存放 Lambda 函數的 CloudFormation 範本和 Python 指令碼。它也會使用 S3 儲存貯體做為網路防火牆警示目的地。
- AWS CloudFormation 可協助您建立模型和設定 AWS 資源、快速一致地佈建資源，以及在整個生命週期中進行管理。您可以使用範本來描述您的資源及其相依性，並將它們一起啟動和設定為堆疊，而不是個別管理資源。此模式使用 AWS CloudFormation 自動部署 Firewall Manager 的分散式架構。

Code

此模式的程式碼可在 GitHub 的 Network Firewall Slack 整合儲存庫中使用。在儲存庫的 `src` 資料夾中，您會找到：

- YAML 格式的一組 CloudFormation 檔案。您可以使用這些範本來佈建此模式的元件。
- 用來建立 Lambda 函數的 Python 來源檔案 (`slack-lambda.py`)。
- 用於上傳 Lambda 函數程式碼的 .zip 封存部署套件 (`slack-lambda.py.zip`)。

若要使用這些檔案，請遵循下一節中的指示。

史詩

設定 S3 儲存貯體

任務	Description	所需的技能
建立 S3 儲存貯體。	1. 登入 AWS 管理主控台，然後前往 https://console.aws.amazon.com/s3/ 開啟 Amazon S3 主控台。 2. 選擇或建立 S3 儲存貯體以託管程式碼。S3 儲存貯體名稱全域唯一，且命名空間由所有 AWS 帳戶共用。S3 儲存貯體名稱不能包含正斜線。我們建議您使用字首來組織此模式的程式碼。	應用程式開發人員、應用程式擁有者、雲端管理員

任務	Description	所需的技能
	如需詳細資訊，請參閱 Amazon S3 文件中的建立儲存貯體。	
上傳 CloudFormation 範本和 Lambda 程式碼。	1. 從此模式的 GitHub 儲存庫下載下列檔案： • `base.yml` • `igw-ingress-route.yml` • `slack-lambda.py` • `slackLambda.yml` • `decentralized-deployment.yml` • `protected-subnet-route.yml` • `slack-lambda.py.zip` 2. 將檔案上傳至您建立的 S3 儲存貯體。	應用程式開發人員、應用程式擁有者、雲端管理員

部署 CloudFormation 範本

任務	Description	所需的技能
啟動 CloudFormation 範本。	在與 S3 儲存貯體相同的 AWS 區域中開啟 AWS CloudFormation 主控台，並部署範本 `base.yml`。此範本會建立所需的 AWS 資源和 Lambda 函數，以將警示傳輸至 Slack 頻道。	應用程式開發人員、應用程式擁有者、雲端管理員

任務	Description	所需的技能
	如需部署 CloudFormation 範本的詳細資訊，請參閱 CloudFormation 文件中的在 AWS CloudFormation 主控台上建立堆疊。	
完成範本中的參數。	指定堆疊名稱並設定參數值。如需參數、其說明和預設值的清單，請參閱額外資訊區段中的 CloudFormation 參數。	應用程式開發人員、應用程式擁有者、雲端管理員
建立堆疊。	1. 根據您的環境需求檢閱堆疊詳細資訊並更新值。 2. 選擇建立堆疊以部署範本。	應用程式開發人員、應用程式擁有者、雲端管理員

驗證解決方案

任務	Description	所需的技能
測試部署。	使用 AWS CloudFormation 主控台或 AWS Command Line Interface (AWS CLI) 來驗證已建立目標技術堆疊區段中列出的資源。 如果 CloudFormation 範本無法成功部署，請檢查您為 `pAvailabilityZone1` 和 `pAvailabilityZone2` 參數提供的值。這些應該適用於您要部署解決方案的 AWS 區域。如需每個區域的可用區域清單，請參閱 Amazon EC2 文件中的區域和區域。	應用程式開發人員、應用程式擁有者、雲端管理員

任務	Description	所需的技能

測試功能。	1. 前往 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。
	
	2. 在其中一個受保護子網路中建立 EC2 執行個體。選擇要用作 HTTPS 伺服器的 Amazon Linux 2 AMI (HVM)。如需說明，請參閱 Amazon EC2 文件中的啟動執行個體。
	
	Note
	Amazon Linux 2 即將終止支援。如需詳細資訊，請參閱 Amazon Linux 2 FAQs。
	
	3. 使用下列使用者資料在 EC2 執行個體上安裝 Web 伺服器：
	
	`#!/bin/bash`
	`yum install httpd -y`
	`systemctl start httpd`
	`systemctl stop firewalld`
	`cd /var/www/html`
	`echo "Hello!! this is a NFW alert test page, 200 OK" > index.html`
	
	4. 建立下列網路防火牆規則：
	
	無狀態規則：
	
	`Source: 0.0.0.0/0`
		應用程式開發人員、應用程式擁有者、雲端管理員

任務	Description	所需的技能
	Destination 10.0.3.65 /32 (private IP of the EC2 instance) Action: Forward 狀態規則： Protocol: HTTP Source ip/port: Any / Any Destination ip/port: Any /Any 5. 取得您在步驟 3 中建立之 Web 伺服器的公有 IP。 6. 在瀏覽器中存取公有 IP。您應該會在瀏覽器中看到下列訊息： Hello!! this is a NFW alert test page, 200 OK 您也會在 Slack 頻道中收到通知。通知可能會延遲，取決於訊息的大小。基於測試目的，請考慮提供不太窄的 CIDR 篩選條件（例如，具有 /32 的 CIDR 值會被視為太窄，而 /8 會太寬）。如需詳細資訊，請參閱其他資訊中的篩選行為一節。	

相關資源

- AWS Network Firewall 的部署模型 (AWS 部落格文章)
- AWS Network Firewall 政策 (AWS 文件)
- Network Firewall Slack 整合 (GitHub 儲存庫)
- 建立 Slack 工作區 (Slack 協助中心)

其他資訊

CloudFormation 參數

參數	Description	預設或範例值
pVpcName	要建立的 VPC 名稱。	檢查
pVpcCidr	要建立之 VPC 的 CIDR 範圍。	10.0.0.0/16
pVpcInstanceTenancy	EC2 執行個體如何分散到實體硬體。選項為 default （共用租用 ） 或 dedicated （單一租用 ）。	預設
pAvailabilityZone1	基礎設施的第一個可用區域。	us-east-2a
pAvailabilityZone2	基礎設施的第二個可用區域。	us-east-2b
pNetworkFirewallSubnet1Cidr	第一個防火牆子網路的 CIDR 範圍 （最小 /28)。	10.0.1.0/24
pNetworkFirewallSubnet2Cidr	第二個防火牆子網路的 CIDR 範圍 （最小 /28)。	10.0.2.0/24
pProtectedSubnet1Cidr	第一個受保護 （工作負載 ） 子網路的 CIDR 範圍。	10.0.3.0/24
pProtectedSubnet2Cidr	第二個受保護 （工作負載 ） 子網路的 CIDR 範圍。	10.0.4.0/24

pS3BucketName	您上傳 Lambda 來源碼的現有 S3 儲存貯體名稱。	us-w2-yourname-lambda-functions
pS3KeyPrefix	您上傳 Lambda 原始碼的 S3 儲存貯體字首。	aod-test
pAWSSecretName4Slack	存放 Slack URL 的秘密名稱。	SlackEnpoint-Cfn
pSlackChannelName	您建立的 Slack 頻道名稱。	somename-notifications
pSlackUserName	Slack 使用者名稱。	Slack 使用者
pSecretKey	這可以是任何金鑰。我們建議您使用預設值。	webhookUrl
pWebHookUrl	Slack URL 的值。	https://hooks.slack.com/services/T???9T？？/A031885JRM7/9D4Y？？？？？？？
pAlertS3Bucket	做為網路防火牆警示目的地的 S3 儲存貯體名稱。系統會為您建立此儲存貯體。	us-w2-yourname-security-aod-alerts
pSecretTagName	秘密的標籤名稱。	AppName
pSecretTagValue	指定標籤名稱的標籤值。	LambdaSlackIntegration
pdestCidr	目的地 CIDR 範圍的篩選條件。如需詳細資訊，請參閱下一節：篩選行為。	10.0.0.0/16
pdestCondition	指出要排除或包含目的地比對的旗標。如需詳細資訊，請參閱下一節。有效值為 `include` 和 `exclude`。	包含
psrcCidr	要提醒的來源 CIDR 範圍篩選條件。如需詳細資訊，請參閱下一節。	118.2.0.0/16

psrcCondition	要排除或包含來源比對的旗標。如需詳細資訊，請參閱下一節。	包含

篩選行為

如果您尚未在 AWS Lambda 中設定任何篩選條件，所有產生的提醒都會傳送到您的 Slack 頻道。產生的警示來源和目的地 IPs 會與您部署 CloudFormation 範本時設定的 CIDR 範圍相符。如果找到相符項目，則會套用條件。如果來源或目的地落在設定的 CIDR 範圍內，且其中至少一個設定為條件 `include`，則會產生提醒。下表提供 CIDR 值、條件和結果的範例。

	設定的 CIDR	警示 IP	Configured	警示
來源	10.0.0.0/16	10.0.0.25	包含	是
目的地	100.0.0.0/16	202.0.0.13	包含	

	設定的 CIDR	警示 IP	Configured	警示
來源	10.0.0.0/16	10.0.0.25	排除	否
目的地	100.0.0.0/16	202.0.0.13	包含	

	設定的 CIDR	警示 IP	Configured	警示
來源	10.0.0.0/16	10.0.0.25	包含	是
目的地	100.0.0.0/16	100.0.0.13	包含	

	設定的 CIDR	警示 IP	Configured	警示
來源	10.0.0.0/16	90.0.0.25	包含	是
目的地	Null	202.0.0.13	包含	

	設定的 CIDR	警示 IP	Configured	警示
來源	10.0.0.0/16	90.0.0.25	包含	否
目的地	100.0.0.0/16	202.0.0.13	包含	

將自訂屬性傳送至 Amazon Cognito，並將其插入字符

Carlos Alessandro Ribeiro 和 Mauricio Mendoza，Amazon Web Services

總結

將自訂屬性傳送至 Amazon Cognito 身分驗證程序可為應用程式提供額外的內容、啟用更精細的存取控制，並讓您更輕鬆地管理使用者設定檔和身分驗證要求。這些功能在各種應用程式和案例中都很有用，可協助您改善應用程式的整體安全性和功能。

此模式顯示當應用程式需要為存取字符或身分 (ID) 字符提供額外的內容時，如何將自訂屬性傳送至 Amazon Cognito 身分驗證程序。您可以使用 Node.js 做為後端應用程式。應用程式會從 Amazon Cognito 使用者集區驗證使用者，並傳遞產生字符所需的自訂屬性。您可以使用 Amazon Cognito AWS Lambda 的觸發條件來自訂身分驗證程序，而無需進行主要程式碼自訂或大量工作。

> **Important**
>
> 此模式中的程式碼和範例不建議用於生產工作負載，因為它們僅用於示範用途。對於生產工作負載，用戶端需要額外的組態。使用此模式做為僅供試行或proof-of-concept參考。

先決條件和限制

先決條件

- 作用中 AWS 帳戶
- 建立和管理 Amazon Cognito 使用者集區和 AWS Lambda 函數的許可
- AWS Command Line Interface (AWS CLI)，已安裝和設定
- 支援 Node.js 的整合式開發環境 (IDE)
- Node.js 第 18 版或更新版本，已安裝
- npm 第 8 版或更新版本，已安裝
- TypeScript，已安裝

限制

- 此模式不適用於透過用戶端登入資料驗證流程進行應用程式整合。

- 產生金鑰前觸發條件只能新增或變更存取字符和身分字符的某些屬性。如需詳細資訊，請參閱 Amazon Cognito 文件中的產生字符前 Lambda 觸發程序。

Architecture

目標架構

下圖顯示此模式的目標架構。它也會顯示 Node.js 應用程式如何與後端搭配使用以更新資料庫。不過，後端資料庫更新超出此模式的範圍。

該圖顯示以下工作流程：

1. Node.js 應用程式會向 Amazon Cognito 使用者集區發出具有自訂屬性的存取字符。
2. Amazon Cognito 使用者集區會啟動預先金鑰產生 Lambda 函數，以自訂存取和 ID 權杖。
3. Node.js 應用程式會透過 Amazon API Gateway 進行 API 呼叫。

> **Note**
>
> 此架構中顯示的其他架構元件僅為 ，且超出此模式的範圍。

自動化和擴展

您可以使用 、、HashiCorp Terraform 或任何支援的基礎設施即程式碼 (IaC) 工具AWS CloudFormation，自動佈建 Amazon Cognito 使用者集區AWS Cloud Development Kit (AWS CDK)、AWS Lambda 函數、資料庫執行個體和其他資源。如果您想要擴展部署，請使用持續整合和持續交付 (CI/CD) 管道，這有助於防止與手動部署相關的錯誤。

工具

AWS 服務

- Amazon API Gateway 可協助您建立、發佈、維護、監控和保護任何規模的 REST、HTTP 和 WebSocket APIs。
- Amazon Cognito 為 Web 和行動應用程式提供身分驗證、授權和使用者管理。
- Amazon Elastic Container Service (Amazon ECS) 是快速、可擴展的容器管理服務，可協助您執行、停止和管理叢集上的容器。

- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。
- 適用於 JavaScript 的 AWS SDK 提供 JavaScript API AWS 服務。您可以使用它來建置 Node.js 或瀏覽器的程式庫或應用程式。

其他工具

- Node.js 是一種事件驅動的 JavaScript 執行期環境，專為建置可擴展的網路應用程式而設計。
- npm 是在 Node.js 環境中執行的軟體登錄檔，用於共用或借用套件和管理私有套件的部署。

最佳實務

我們建議您實作下列最佳實務：

- 秘密和敏感資料 – 請勿在應用程式中存放秘密或敏感資料。使用應用程式可以從中提取資料的外部系統，例如 AWS AppConfig、 AWS Secrets Manager或 AWS Systems Manager參數存放區。
- 標準化部署 – 使用 CI/CD 管道來部署您的應用程式。您可以使用 AWS CodeBuild和 等服務AWS CodePipeline。
- 權杖過期 – 設定存取權杖的簡短過期日期。
- 使用安全連線 – 用戶端應用程式與後端之間的所有通訊都應使用 SSL/TLS 加密。使用 AWS Certificate Manager (ACM) 來產生和管理 SSL/TLS 憑證，並使用 Amazon CloudFront 或 Elastic Load Balancing 來處理 SSL/TLS 終止。
- 驗證使用者輸入 – 確保所有使用者輸入都經過驗證，以防止注入攻擊和其他安全漏洞。使用輸入驗證程式庫和服務，例如 Amazon API Gateway 和 AWS WAF，以防止常見的攻擊向量。
- 使用 IAM 角色 – 使用 AWS Identity and Access Management (IAM) 角色來控制對 AWS 資源的存取，並確保只有獲得授權的使用者才能存取。遵循最低權限原則，並確保每個使用者只有執行其角色所需的許可。
- 使用密碼政策 – 設定符合您安全要求的密碼政策，例如長度下限、複雜性和過期時間。使用 Secrets Manager 或 AWS Systems Manager 參數存放區安全地存放和管理密碼。
- 啟用多重要素驗證 (MFA) – 為所有使用者啟用 MFA，以提供額外的安全層，並降低未經授權的存取風險。使用 AWS IAM Identity Center或 Amazon Cognito 來啟用 MFA 和其他身分驗證方法。
- 安全地存放敏感資訊 – 使用 AWS Key Management Service (AWS KMS) 或其他加密服務安全地存放敏感資訊，例如密碼和存取權杖。

- 使用強式身分驗證方法 – 若要提高身分驗證程序的安全性，請使用強式身分驗證方法，例如生物識別驗證或多重要素驗證。
- 監控可疑活動 – 使用 AWS CloudTrail 和其他監控工具來監控可疑活動和潛在的安全威脅。設定異常活動的自動提醒，並使用 Amazon GuardDuty 或 AWS Security Hub CSPM 來偵測潛在威脅。
- 定期檢閱和更新安全政策 – 定期檢閱和更新安全政策和程序，以確保它們符合您不斷變化的安全需求和最佳實務。使用 AWS Config 來追蹤和稽核安全政策和程序的變更。
- 自動化註冊 – 請勿啟用 Amazon Cognito 使用者集區的自動註冊。如需詳細資訊，請參閱使用 Amazon Cognito 使用者集區降低使用者註冊詐騙和 SMS 輸出的風險 (AWS 部落格文章）。

如需其他最佳實務，請參閱 Amazon Cognito 文件中的 Amazon Cognito 使用者集區的安全最佳實務。Amazon Cognito

史詩

設定 AWS 資源

任務	Description	所需的技能

建立使用者集區。	1. 在 CLI 終端機中，輸入下列命令來建立 Amazon Cognito 使用者集區：
	`aws cognito-idp create-user-pool \ --pool-name <MyUserPool>`
	2. 輸入下列命令來建立用戶端 ID 和秘密，以便與適用於 JavaScript 的 SDK 搭配使用：
	`aws cognito-idp create-user-pool-client \ --user-pool-id <UserPoolID> \`
		應用程式開發人員、AWS DevOps

任務	Description	所需的技能

	<pre>--client-name
	<MyNewClient> \
	--no-generate-secret \
	--explicit-auth-flows "USER_PASSWORD_AUTH"
	"ADMIN_NO_SRP_AUTH"</pre>
	如需如何在 中設定使用者集區的詳細資訊和指示 AWS 管理主控台，請參閱使用者集區入門和將更多功能和安全性選項新增至您的使用者集區。
	Tip
	若要降低成本，請使用 Essentials 計劃或 Lite 計劃來測試此模式。如需詳細資訊，請參閱 Amazon Cognito 定價。	

任務	Description	所需的技能
將使用者新增至使用者集區。	輸入下列命令，在 Amazon Cognito 使用者集區中建立一個使用者： `aws cognito-idp sign-up \ --client-id <ClientID> \ --username <jane@example.com> \ --password <PASSWORD> \ --user-attributes Name="email",Value="<jane@example.com>" Name="name",Value="<Jane>"`	應用程式開發人員、AWS DevOps

任務	Description	所需的技能
將應用程式用戶端新增至使用者集區。	1. 導覽至 Amazon Cognito 主控台。 2. 選擇 User pools (使用者集區)。 3. 選擇您先前建立的使用者集區。 4. 在定義應用程式下，選擇最適合您要為其建立身分驗證和授權服務之應用程式案例的應用程式類型。 5. 在為您的應用程式命名中，輸入描述性名稱或繼續使用預設名稱。 6. 在新增傳回 URL 下，輸入使用者完成身分驗證後應用程式的重新導向路徑。 7. 選擇建立您的應用程式。您可以在建立應用程式用戶端後設定進階選項。	AWS 系統管理員、AWS 管理員、AWS DevOps、應用程式開發人員

任務	Description	所需的技能
建立 Lambda 觸發程序以產生預先權杖。	1. 開啟 Lambda 主控台中的 函數頁面。 2. 選擇建立函數。 3. 選取從頭開始撰寫。 4. 在基本資訊窗格中，forFunction 名稱，輸入 `myPreTokenGenerationLambdaFunction` 。 5. ForRuntime，選擇 Node.js 22。 6. Leavearchitectureset tox86_64。 7. 選擇 Create function (建立函數)。 8. 在程式碼索引標籤的程式碼來源下，從此模式的其他資源區段輸入範例 Lambda 函數。此 Lambda 函數會將自訂屬性注入 ID 字符和存取字符。	AWS DevOps，應用程式開發人員

任務	Description	所需的技能
自訂使用者集區工作流程。	1. 導覽至 Amazon Cognito 主控台。 2. 選擇 User pools (使用者集區)。 3. 選擇您先前建立的使用者集區。 4. 在左側導覽窗格的身分驗證下，選擇延伸。 5. 選擇新增 Lambda 觸發程序。 6. 新增或編輯產生權杖前觸發程序。 7. 針對觸發類型，選擇身分驗證。 8. 在身分驗證下，選擇權杖產生前觸發條件。 9. 對於指派 Lambda 函數，選擇 `myPreTokenGenerationLambdaFunction` 。 10選擇新增 Lambda 觸發程序。 11選擇 aTrigger 事件版本基本功能 + 使用者身分的存取權杖自訂或基本功能 + 使用者和機器身分的存取權杖自訂。此設定會更新 Amazon Cognito 傳送至您函數的請求參數，以包含用於存取權杖自訂的欄位。	AWS DevOps，應用程式開發人員

任務	Description	所需的技能
	如需詳細資訊，請參閱 Amazon Cognito 文件中的使用 Lambda 觸發程序自訂使用者集區工作流程。	

建立 Node.js 應用程式

任務	Description	所需的技能

建立 Node.js 應用程式。	1. 在終端機中，輸入下列命令：
	`mkdir my-app`
	`cd my-app`
	`npm init -y`
	`npm install --save-dev typescript`
	`npm i --save-dev @types/node`
	`npm i amazon-cognito-identity-js`
	`npx tsc --init`
	`mkdir src`
	`touch src/index.ts`
	2. 修改 `package.json` 檔案以包含 TypeScript 編譯指令碼：
	`"scripts": { "start": "node src/index.js", "build": "tsc" },`
		應用程式開發人員

實作身分驗證邏輯。	1. 開啟 `index.ts` 檔案。	應用程式開發人員

任務	Description	所需的技能
	2. 貼上此模式的其他資源區段中提供的範例 Typescript。 3. 儲存並關閉 `index.ts` 檔案。 **Note** 您可以建立自己的 TypeScript 檔案，或修改您的使用案例所需的範例。	
設定環境變數和組態檔案。	在終端機中，輸入下列命令來建立環境變數： `export USERNAME= "<COGNITO_USER_NAME>" export PASSWORD= "<COGNITO_USER_PAS SWORD>" export USER_POOL _ID="<COGNITO_USER _ID>" export CLIENT_ID ="<COGNITO_CLIENT_ ID>"` **Important** 請勿硬式編碼秘密或公開您的登入資料。	應用程式開發人員

任務	Description	所需的技能

執行應用程式。	輸入下列命令來執行應用程式並確認其正常運作：
	`npm run build`
	`npm start`
		應用程式開發人員

確認自訂屬性已插入字符中。	使用 IDE 的偵錯功能來檢視存取和 ID 字符。確認已新增自訂屬性。如需範例權杖，請參閱此模式的其他資訊區段。	應用程式開發人員

疑難排解

問題	解決方案
嘗試驗證使用者時無效的用戶端 ID	當您使用用戶端 ID 搭配產生的用戶端秘密時，通常會發生此錯誤。您必須建立未連接秘密的用戶端 ID。如需詳細資訊，請參閱應用程式用戶端的應用程式特定設定。

相關資源

- 使用 Lambda 觸發來自訂使用者集區工作流程 (Amazon Cognito 文件）
- 權杖產生前 Lambda 觸發程序 (Amazon Cognito 文件）
- CognitoIdentityProviderClient (適用於 JavaScript 的 AWS SDK 文件）
- cognito-idp (AWS CLI 文件）

其他資訊

範例 TypeScript 檔案

下列程式碼範例是 TypeScript 檔案，使用 AWS SDK 將自訂屬性傳送至 Amazon Cognito 來叫用身分驗證程序：

```
import * as AmazonCognitoIdentity from "amazon-cognito-identity-js";

const userPoolId: string = process.env.USER_POOL_ID ?? '';
const clientId: string = process.env.CLIENT_ID ?? '';

const poolData = {
  UserPoolId: userPoolId,
  ClientId: clientId
};
const userPool = new AmazonCognitoIdentity.CognitoUserPool(poolData);

export const loginWithCognitoSDK = function (userName: string, password: string) {
  const authenticationDetails = new AmazonCognitoIdentity.AuthenticationDetails({
    Username: userName,
    Password: password,
    ClientMetadata: {
        customGroup: "MyCustomGroup",
        customApplicationData: "Custom data from a custom application"
    }
  });
  const userData = {
    Username: userName,
    Pool: userPool
  };

  const cognitoUser = new AmazonCognitoIdentity.CognitoUser(userData);

  // Authenticate the user using the authenticationDetails object
  cognitoUser.authenticateUser(authenticationDetails, {
    onSuccess: function (result: any) {},
    onFailure: function (err: any) {},
  });
}
loginWithCognitoSDK(process.env.USERNAME ?? '', process.env.PASSWORD ?? '');
```

此範例使用適用於 JavaScript 的 SDK 中的`AuthenticationDetails`模型來提供使用者名稱、密碼和 `ClientMetadada`。在 Amazon Cognito 中驗證之後，可以從存取和 ID 字符擷取用戶端中繼資料。

Lambda 函數範例

下列程式碼範例是連結至 Amazon Cognito 產生前字符的 Lambda 函數。它可協助您自訂 Amazon Cognito 使用的存取字符和 ID 字符。權杖會透過架構之間的整合傳遞。此範例包含名為 的自訂宣告屬性，customApplicationData以及名為 的自訂群組名稱MyCustomGroup：

```
export const handler = async(event, context, callback) => {
    event.response = {
        claimsOverrideDetails: {
            claimsToAddOrOverride: { customApplicationData:
 event.request.clientMetadata.customApplicationData },
            groupOverrideDetails: { groupsToOverride:
 [event.request.clientMetadata.customGroup] }
        }
    };
    callback(null, event);
};
```

存取字符範例

您可以解碼存取權杖，以視覺化已新增的自訂屬性。以下是存取字符的範例：

```
{
  "sub": "6daf331f-4451-48b4-abde-774579299204",
  "cognito:groups": [
    "MyCustomGroup"
  ],
  "iss": "https://cognito-idp.<REGION>.amazonaws.com/<USERPOOL_ID>",
  "client_id": "<YOUR_CLIENT_ID>",
  "origin_jti": "acff7e91-09f9-4fde-8eec-38b0f8c47cdc",
  "event_id": "c5113a9c-1f01-435b-9b73-a5cd3e88514e",
  "token_use": "access",
  "scope": "aws.cognito.signin.user.admin",
  "auth_time": 1677979246,
  "exp": 1677982846,
  "iat": 1677979246,
  "jti": "5c9c2708-a871-4428-bd9b-18ad261bea90",
  "username": "<USER_NAME>"
}
```

範例 ID 字符

您可以解碼存取權杖，以視覺化已新增的自訂屬性。以下是存取字符的範例：

```
{
```

```
  "sub": "6daf331f-4451-48b4-abde-774579299204",
  "cognito:groups": [
    "MyCustomGroup"
  ],
  "iss": "https://cognito-idp.<REGION>.amazonaws.com/<USERPOOL_ID>",
  "cognito:username": "<USER_NAME>",
  "origin_jti": "acff7e91-09f9-4fde-8eec-38b0f8c47cdc",
  "customApplicationData": "Custom data from a custom application",
  "aud": "<YOUR_CLIENT_ID>",
  "event_id": "c5113a9c-1f01-435b-9b73-a5cd3e88514e",
  "token_use": "id",
  "auth_time": 1677979246,
  "exp": 1677982846,
  "iat": 1677979246,
  "jti": "f7ca006b-f25b-44d2-a7a4-6e6423f4201f",
  "email": "<USER_EMAIL>"
}
```

使用 AWS Private CA 和 AWS RAM 簡化私有憑證管理

Everett Hinckley 和 Vivek Goyal，Amazon Web Services

總結

您可以使用 AWS Private Certificate Authority (AWS Private CA) 發行私有憑證，以驗證內部資源和簽署電腦程式碼。此模式提供 AWS CloudFormation 範本，可快速部署多層 CA 階層和一致的佈建體驗。或者，您可以使用 AWS Resource Access Manager (AWS RAM) 在 AWS Organizations 中的組織或組織單位 (OUs) 內安全地共用 CA，並在使用 AWS RAM 管理許可時集中 CA。每個帳戶中不需要私有 CA，因此這種方法可以節省您的成本。此外，您可以使用 Amazon Simple Storage Service (Amazon S3) 來存放憑證撤銷清單 (CRL) 和存取日誌。

此實作提供下列功能和優點：

- 使用 AWS Private CA 集中並簡化私有 CA 階層的管理。
- 將憑證和金鑰匯出至 AWS 和內部部署上的客戶受管裝置。
- 使用 AWS CloudFormation 範本以獲得快速部署和一致的佈建體驗。
- 建立私有根 CA 以及 1、2、3 或 4 個次級 CA 階層。
- 或者， 使用 AWS RAM 與組織或 OU 層級的其他帳戶共用終端實體次級 CA。
- 使用 AWS RAM 移除每個帳戶中私有 CA 的需求，以節省成本。
- 建立 CRL 的選用 S3 儲存貯體。
- 為 CRL 存取日誌建立選用的 S3 儲存貯體。

先決條件和限制

先決條件

如果您想要在 AWS Organizations 結構中共用 CA，請識別或設定下列項目：

- 用於建立 CA 階層和共用的安全帳戶。
- 用於測試的個別 OU 或帳戶。
- 在 AWS Organizations 管理帳戶中啟用共用。如需詳細資訊，請參閱 AWS RAM 文件中的在 AWS Organizations 中啟用資源共用。

限制

- CAs是區域資源。所有 CAs都位於單一 AWS 帳戶和單一 AWS 區域。
- 不支援使用者產生的憑證和金鑰。針對此使用案例，我們建議您自訂此解決方案以使用外部根 CA。
- 不支援公有 CRL 儲存貯體。我們建議您將 CRL 保持私有。如果需要 CRL 的網際網路存取，請參閱 AWS Private CA 文件中的使用 Amazon CloudFront 為 CRLs 提供啟用 S3 封鎖公開存取 (BPA) 功能的一節。
- 此模式實作單一區域方法。如果您需要多區域憑證授權單位，您可以在第二個 AWS 區域或內部部署中實作次級。這種複雜性超出此模式的範圍，因為實作取決於您的特定使用案例、工作負載磁碟區、相依性和需求。

Architecture

目標技術堆疊

- AWS Private CA
- AWS RAM
- Amazon S3
- AWS Organizations
- AWS CloudFormation

目標架構

此模式提供兩種共用給 AWS Organizations 的選項：

選項 1 — 在組織層級建立共享。組織中的所有帳戶都可以使用共用 CA 發行私有憑證，如下圖所示。

選項 2 — 在組織單位 (OU) 層級建立共享。只有指定 OU 中的帳戶可以使用共用 CA 發行私有憑證。例如，在下圖中，如果共享是在沙盒 OU 層級建立，則開發人員 1 和開發人員 2 都可以使用共用 CA 發行私有憑證。

工具

AWS 服務

- AWS Private CA – AWS Private Certificate Authority (AWS Private CA) 是用於發行和撤銷私有數位憑證的託管私有 CA 服務。它可協助您建立私有 CA 階層，包括根 CA 和次級 CAs，而無需操作內部部署 CA 的投資和維護成本。
- AWS RAM – AWS Resource Access Manager (AWS RAM) 可協助您在 AWS 帳戶以及 AWS Organizations 中的組織或 OUs 內安全地共用資源。若要減少多帳戶環境中的營運開銷，您可以建立資源，並使用 AWS RAM 跨帳戶共用該資源。
- AWS Organizations – AWS Organizations 是一種帳戶管理服務，可讓您將多個 AWS 帳戶合併到您建立並集中管理的組織。
- Amazon S3 – Amazon Simple Storage Service (Amazon S3) 是一種物件儲存服務。您可以使用 Amazon S3 隨時從 Web 任何地方存放和擷取任意資料量。此模式使用 Amazon S3 來存放憑證撤銷清單 (CRL) 和存取日誌。
- AWS CloudFormation – AWS CloudFormation 可協助您建立模型並設定 AWS 資源、快速一致地佈建資源，以及在整個生命週期中管理資源。您可以使用範本來描述您的資源及其相依性，並將它們一起啟動和設定為堆疊，而不是個別管理資源。此模式使用 AWS CloudFormation 自動部署多層 CA 階層。

Code

此模式的原始碼可在 GitHub 的 AWS Private CA 階層儲存庫中使用。儲存庫包含：

- AWS CloudFormation 範本 `ACMPCA-RootCASubCA.yaml`。您可以使用此範本來部署此實作的 CA 階層。
- 測試檔案是否有請求、匯出、描述和刪除憑證等使用案例。

若要使用這些檔案，請遵循 Epics 區段中的指示。

史詩

架構 CA 階層

任務	Description	所需的技能
收集憑證主體資訊。	收集憑證擁有者的憑證主體資訊：組織名稱、組織單位、國家/地區、州、地區和通用名稱。	雲端架構師、安全架構師、PKI 工程師

任務	Description	所需的技能
收集 AWS Organizations 的選用資訊。	如果 CA 是 AWS Organizations 結構的一部分，而且您想要在該結構內共用 CA 階層，請收集管理帳號、組織 ID 和選擇性的 OU ID（如果您只想與特定 OU 共用 CA 階層）。此外，判斷您要與之共用 CA 的 AWS Organizations 帳戶或 OUs。	雲端架構師、安全架構師、PKI 工程師
設計 CA 階層。	決定哪個帳戶將存放根 CA 和次級 CAs。決定階層在根憑證和終端實體憑證之間需要多少次級層級。如需詳細資訊，請參閱 AWS Private CA 文件中的設計 CA 階層。	雲端架構師、安全架構師、PKI 工程師
決定 CA 階層的命名和標記慣例。	決定 AWS 資源的名稱：根 CA 和每個次級 CA。決定應指派給每個 CA 的標籤。	雲端架構師、安全架構師、PKI 工程師
判斷所需的加密和簽署演算法。	判斷下列項目： • 您組織的公有金鑰加密演算法需求，您的 CA 在發行憑證時使用。預設值為 RSA_2048。 • CA 用於憑證簽署的金鑰演算法。預設值為 SHA256WITHRSA。	雲端架構師、安全架構師、PKI 工程師
判斷 CA 階層的憑證撤銷要求。	如果需要憑證撤銷功能，請為包含憑證撤銷清單 (CRL) 的 S3 儲存貯體建立命名慣例。	雲端架構師、安全架構師、PKI 工程師

任務	Description	所需的技能
判斷 CA 階層的記錄需求。	如果需要存取記錄功能，請為包含存取日誌的 S3 儲存貯體建立命名慣例。	雲端架構師、安全架構師、PKI 工程師
判斷憑證過期期間。	決定根憑證的過期日期（預設值為 10 年）、終端實體憑證（預設值為 13 個月）和次級 CA 憑證（預設值為 3 年）。次級 CA 憑證的過期時間應該早於階層中較高層級的 CA 憑證。如需詳細資訊，請參閱 AWS Private CA 文件中的管理私有 CA 生命週期。	雲端架構師、安全架構師、PKI 工程師

部署 CA 階層

任務	Description	所需的技能
完成 事前準備。	完成此模式的先決條件區段中的步驟。	雲端管理員、安全工程師、PKI 工程師
為各種角色建立 CA 角色。	1. 判斷 AWS IAM Identity Center (AWS Single Sign-On 的後繼者）中管理各種 CA 階層所需的 AWS Identity and Access Management (IAM) 角色或使用者類型，例如 RootCAAdmin、SubordinateCAAdmin 和 CertificateConsumer。 AWS Single Sign-On 2. 判斷分隔職責所需的政策精細程度。	雲端管理員、安全工程師、PKI 工程師

任務	Description	所需的技能
	3. 在 CA 階層所在的帳戶中，在 IAM Identity Center 中建立所需的 IAM 角色或使用者。	
部署 CloudFormation 堆疊。	1. 從此模式的 GitHub 儲存庫中，下載 AWSPCA-RootCASubCA.yaml 範本。 2. 從 AWS CloudFormation 主控台或 AWS 命令列界面 (AWS CLI) 部署範本。如需詳細資訊，請參閱 CloudFormation 文件中的使用堆疊。 3. 完成範本中的參數，包括組織名稱、OU 名稱、金鑰演算法、簽署演算法和其他選項。	雲端管理員、安全工程師、PKI 工程師

任務	Description	所需的技能
建構解決方案，以更新使用者受管資源使用的憑證。	整合式 AWS 服務的資源，例如 Elastic Load Balancing，會在過期前自動更新憑證。不過，使用者管理的資源，例如在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上執行的 Web 伺服器，需要另一個機制。 1. 判斷哪些使用者受管資源需要私有 CA 的終端實體憑證。 2. 規劃程序，以便在使用者受管資源和憑證過期時收到通知。如需 範例，請參閱下方： • 使用 AWS Config 受管規則 • 使用 Amazon CloudWatch 和 Amazon EventBridge 3. 撰寫自訂指令碼以更新使用者受管資源上的憑證，並將其與 AWS 服務整合以自動化更新。如需整合式 AWS 服務的詳細資訊，請參閱 ACM 文件中的與 AWS Certificate Manager 整合的服務。	雲端管理員、安全工程師、PKI 工程師

驗證並記錄 CA 階層

任務	Description	所需的技能
驗證選用的 AWS RAM 共用。	如果 CA 階層與 AWS Organizations 中的其他帳戶共用，請從 AWS 管理主控台登入其中一個帳戶，導覽至 AWS Private CA 主控台，並確認新建立的 CA 已共用至此帳戶。只有階層中最低層級的 CA 才會顯示，因為這是產生終端實體憑證的 CA。針對與 CA 共用的帳戶，重複取樣。	雲端管理員、安全工程師、PKI 工程師
使用憑證生命週期測試驗證 CA 階層。	在此模式的 GitHub 儲存庫中，找到生命週期測試。從 AWS CLI 執行測試以請求憑證、匯出憑證、描述憑證，以及刪除憑證。	雲端管理員、安全工程師、PKI 工程師
將憑證鏈匯入信任存放區。	對於要信任憑證的瀏覽器和其他應用程式，憑證的發行者必須包含在瀏覽器的信任存放區中，這是信任 CAs的清單。將新 CA 階層的憑證鏈新增至瀏覽器和應用程式的信任存放區。確認終端實體憑證是受信任的。	雲端管理員、安全工程師、PKI 工程師
建立 Runbook 以記錄 CA 階層。	建立 Runbook 文件來描述 CA 階層的架構、可請求終端實體憑證的帳戶結構、建置程序，以及發行終端實體憑證（除非您想要依子帳戶允許自助服務）、用量和追蹤等基本管理任務。	雲端管理員、安全工程師、PKI 工程師

相關資源

- 設計 CA 階層 (AWS Private CA 文件）
- 建立私有 CA (AWS Private CA 文件）
- 如何使用 AWS RAM 來共用您的 AWS Private CA 跨帳戶 (AWS 部落格文章）
- AWS Private CA 最佳實務 (AWS 部落格文章）
- 在 AWS Organizations 中啟用資源共用 (AWS RAM 文件）
- 管理私有 CA 生命週期 (AWS Private CA 文件）
- AWS Config 的 acm-certificate-expiration-check AWS Config(AWS Config 文件）
- AWS Certificate Manager 現在透過 Amazon CloudWatch 提供憑證過期監控 (AWS 公告）
- 與 AWS Certificate Manager 整合的服務 (ACM 文件）

其他資訊

匯出憑證時，請使用密碼編譯強式的複雜密碼，並與組織的資料外洩預防策略保持一致。

使用 Amazon Bedrock 代理程式和 簡化 Amazon EC2 合規管理 AWS Config

Anand Bukkapatnam Tirumala，Amazon Web Services

總結

此模式說明如何將 Amazon Bedrock 與 AWS Config 規則整合，以促進 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的合規管理。此方法使用進階生成式 AI 功能來提供符合 AWS Well-Architected Framework 的量身打造建議，以確保最佳執行個體類型選擇和系統效率。此模式的主要功能包括：

- 自動化合規監控： AWS Config 規則會根據所需執行個體類型的預先定義條件持續評估 EC2 執行個體。
- AI 驅動的建議：Amazon Bedrock 中的生成式 AI 模型會分析基礎設施模式。這些模型根據 AWS Well-Architected Framework 中概述的最佳實務提供智慧型改進建議。
- 修復：Amazon Bedrock 動作群組可實現自動化修復步驟，以快速解決不合規的執行個體，並將潛在的效能或成本效率降低到最低。
- 可擴展性和適應性：該解決方案旨在隨著您的基礎設施進行擴展，並適應不斷變化的雲端架構需求。
- 增強型安全建議：遵循 AWS Well-Architected 原則有助於改善安全狀態和系統效能。

您可以使用此模式做為藍圖，視需要使用 DevOps 實務，將您自己的生成式 AI 型基礎設施部署到多個環境中，並將變更降至最低。

先決條件和限制

先決條件

- 作用中 AWS 帳戶。
- 具有在 Amazon Simple Storage Service (Amazon S3) 儲存貯體、 AWS Config、 AWS Lambda 函數、Amazon Bedrock、IAM、Amazon CloudWatch Logs 和 Amazon EC2 中建立和管理資源許可的 AWS Identity and Access Management (IAM) 角色。
- 要標記為不合規的 EC2 執行個體。請勿將 `t2.small`類型用於此執行個體。
- Amazon Titan Text Embeddings V2 和 Anthropic Claude 3 Haiku 模型已在 中啟用 AWS 帳戶。若要為部署解決方案 AWS 區域 的 啟用模型存取，請參閱 Amazon Bedrock 文件中的新增或移除對 Amazon Bedrock 基礎模型的存取。

- Terraform，已安裝並設定。
- 在部署環境中安裝和設定的 AWS Command Line Interface (AWS CLI) v2。
- 已完成 Amazon Responsible AI 政策的檢閱。

限制

- 有些 AWS 服務 完全無法使用 AWS 區域。如需區域可用性，請參閱AWS 服務 依區域。如需特定端點，請參閱服務端點和配額，然後選擇服務的連結。
- 此解決方案已使用 Amazon Titan Text Embeddings V2 和 Claude 3 Haiku 模型進行測試。如果您偏好使用其他模型，您可以自訂 Terraform 程式碼，其會進行參數化以方便變更。
- 此解決方案不包含聊天歷史記錄功能，也不會儲存聊天。

Architecture

下圖顯示此模式的工作流程和架構元件。

工作流程包含下列步驟：

1. 使用者透過 Amazon Bedrock 聊天主控台與模型互動。使用者會提出問題，例如：
   - `What can you help me with?`
   - `List non-complaint resources`
   - `Suggest security best practices`
2. 如果模型已預先訓練，則會直接從其現有知識回應提示。否則，提示會移至 Amazon Bedrock 動作群組。
3. 動作群組使用 AWS PrivateLink進行安全的服務通訊，以到達 VPC 端點。
4. 請求會透過 Amazon Bedrock 服務的 VPC 端點到達 Lambda 函數。
5. Lambda 函數是主要執行引擎。根據請求， 函數會呼叫 API 以在 上執行動作 AWS 服務。它也會處理操作路由和執行。
6. Lambda 函數會呼叫 AWS Config 來判斷不合規的資源 （您建立做為先決條件的不合規 EC2 執行個體）。
7. AWS Config 會標記非投訴資源。此模式會部署 AWS Config desired-instance-type 規則，以尋找理想的 EC2 執行個體大小。

8. AWS Config 會提示使用者暫停或修復執行個體，並對 EC2 執行個體採取相應的動作。Amazon Bedrock 了解此傳回承載。
9. 使用者會在 Amazon Bedrock 聊天主控台上收到回應。

自動化和擴展

此解決方案使用 Terraform 做為基礎設施做為程式碼 (IaC) 工具，以輕鬆部署至 AWS 帳戶 ，並在多個帳戶中做為獨立公用程式運作。此方法可簡化管理並改善部署中的一致性。

工具

AWS 服務

- AWS Config 可讓您評估、稽核和評估 AWS 資源的組態，以實現合規和所需的設定。
- Amazon Bedrock 是一項全受管 AI 服務，可透過統一的 API 存取許多高效能的基礎模型。
- AWS Identity and Access Management (IAM) 透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Lambda 是一項運算服務，可協助您執行程式碼，無需佈建或管理伺服器。它只會在需要時執行程式碼並自動擴展，因此您只需支付使用的運算時間。

其他工具

- Git 是一種開放原始碼的分散式版本控制系統。
- Terraform 是 HashiCorp 的基礎設施即程式碼 (IaC) 工具，可協助您建立和管理雲端和內部部署資源。

程式碼儲存庫

此模式的程式碼可在 GitHub sample-awsconfig-bedrock-compliance-manager 儲存庫中使用。

最佳實務

- 遵循最低權限原則，並授予執行任務所需的最低許可。如需詳細資訊，請參閱 IAM 文件中的授予最低權限和安全最佳實務和使用案例。
- 定期監控 Lambda 執行日誌。如需詳細資訊，請參閱 Lambda 文件中的監控、偵錯和疑難排解 Lambda 函數，以及使用 AWS Lambda 函數的最佳實務。

史詩

部署解決方案

任務	Description	所需的技能
複製儲存庫。	若要複製此模式的儲存庫，請使用下列命令： `git clone "git@github.com:aws-samples/sample-awsconfig-bedrock-compliance-manager.git"`	AWS DevOps、建置主管、DevOps 工程師、雲端管理員
編輯環境變數。	在本機電腦上複製儲存庫的根目錄中，編輯 `terraform.tfvars` 檔案。檢閱標示為的預留位置`[XXXXX]`，並根據您的環境進行編輯。	AWS 系統管理員、AWS DevOps、DevOps 工程師、AWS 管理員

建立基礎設施。	1. 若要為此解決方案建立基礎設施，請執行下列命令：
	`terraform init`
	`terraform plan`
	2. 檢閱執行計畫，然後執行 命令：
	`terraform apply --auto-approve`
		AWS DevOps、DevOps 工程師、AWS 系統管理員、雲端管理員

使用代理程式

任務	Description	所需的技能

與客服人員聊天。	在上一個步驟中部署解決方案會部署 `security-bot-agent` ，這是具有聊天主控台的 Amazon Bedrock 代理程式。
	
	使用代理程式：
	
	1. AWS 管理主控台 使用具有使用 Amazon Bedrock 主控台之許可的 IAM 身分登入 。然後，開啟 Amazon Bedrock 主控台。
	2. 在導覽窗格中，選擇建置、客服人員。選取 security-bot-agent。
	3. 開始與客服人員交談。例如：
	
	`List non-complaint EC2 instances`
	
	和：
	
	`Replace these non-complaint EC2 instances with compliant instances`
		AWS DevOps、DevOps 工程師、AWS 系統管理員、雲端管理員

清除資源

任務	Description	所需的技能
刪除基礎設施和資源。	當您完成使用此解決方案的工作後，您可以執行 命令來刪除此模式建立的基礎設施： `terraform destroy --auto-approve`	AWS DevOps、DevOps 工程師、AWS 系統管理員、雲端管理員

疑難排解

問題	解決方案
客服人員行為問題	如需疑難排解資訊，請參閱 Amazon Bedrock 文件中的測試和疑難排解代理程式行為。
AWS Lambda 網路問題	如需詳細資訊，請參閱 Lambda 文件中的對 Lambda 中的聯網問題進行故障診斷。
IAM 許可	如需詳細資訊，請參閱 IAM 文件中的疑難排解 IAM。

相關資源

- Amazon Bedrock 代理程式
- 使用動作群組來定義代理程式要執行的動作 (Amazon Bedrock 文件)
- desired-instance-type 規則 (AWS Config 文件)
- AWS Config 運作方式 (AWS Config 文件)

使用 PowerShell 從 AWS IAM Identity Center 更新 AWS CLI 憑證

ChadMiles 和 Andy Bowen，Amazon Web Services

總結

如果您想要搭配 AWS 命令列界面 (AWS CLI)、AWS SDKs或 AWS 雲端開發套件 (AWS CDK) 使用 AWS IAM Identity Center (AWS Single Sign-On 的後續產品） 登入資料，您通常必須將登入資料從 IAM Identity Center 主控台複製並貼到命令列界面。此程序可能需要相當長的時間，而且必須針對每個需要存取的帳戶重複。

一個常見的解決方案是使用 AWS CLI `aws sso configure`命令。此命令會將啟用 IAM Identity Center 的設定檔新增至您的 AWS CLI 或 AWS 開發套件。不過，此解決方案的缺點是，您必須`aws sso login`對以這種方式設定的每個 AWS CLI 設定檔或帳戶執行 命令。

做為替代解決方案，此模式說明如何使用 AWS CLI 命名設定檔和適用於 PowerShell 的 AWS 工具，同時從單一 IAM Identity Center 執行個體存放和重新整理多個帳戶的登入資料。指令碼也會將 IAM Identity Center 工作階段資料存放在記憶體中，以重新整理登入資料，而無需再次登入 IAM Identity Center。

先決條件和限制

先決條件

- PowerShell，已安裝並設定。如需詳細資訊，請參閱安裝 PowerShell (Microsoft 文件）。
- 已安裝和設定的 AWS Tools for PowerShell。基於效能考量，強烈建議您安裝名為 的 AWS Tools for PowerShell 模組化版本`AWS.Tools`。每個 AWS 服務都由其個別的小型模組支援。在 PowerShell 提示中，輸入下列命令來安裝此模式所需的模組：`AWS.Tools.Installer`、 `SSO`和 `SSOIDC`。

  ```
  Install-Module AWS.Tools.Installer
  Install-AWSToolsModule SSO, SSOOIDC
  ```

  如需詳細資訊，請參閱在 Windows 上安裝 AWS.Tools 或在 Linux 或 macOS 上安裝 AWS.Tools。
- AWS CLI 或 AWS 開發套件先前必須使用工作登入資料進行設定，方法為執行下列其中一項：
  - 使用 AWS CLI `aws configure`命令。如需詳細資訊，請參閱快速組態 (AWS CLI 文件）。
  - 設定 AWS CLI 或 AWS CDK 以透過 IAM 角色取得暫時存取權。如需詳細資訊，請參閱取得 CLI 存取的 IAM 角色登入資料 (IAM Identity Center 文件）。

限制

- 此指令碼無法用於管道或全自動化解決方案。部署此指令碼時，您必須從 IAM Identity Center 手動授權存取。然後指令碼會自動繼續。

產品版本

- 對於所有作業系統，建議您使用 PowerShell 7.0 版或更新版本。

Architecture

您可以使用此模式中的指令碼，同時重新整理多個 IAM Identity Center 登入資料，也可以建立登入資料檔案，以搭配 AWS CLI、AWS SDKs或 AWS CDK 使用。

工具

AWS 服務

- AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可協助您透過命令列 shell 中的命令與 AWS 服務互動。
- AWS IAM Identity Center 可協助您集中管理所有 AWS 帳戶和雲端應用程式的單一登入 (SSO) 存取。
- AWS Tools for PowerShell 是一組 PowerShell 模組，可協助您從 PowerShell 命令列對 AWS 資源執行指令碼操作。

其他工具

- PowerShell 是在 Windows、Linux 和 macOS 上執行的 Microsoft 自動化和組態管理程式。

最佳實務

為每個 IAM Identity Center 執行個體保留一份此指令碼的副本。不支援將一個指令碼用於多個執行個體。

史詩

執行 SSO 指令碼

任務	Description	所需的技能
自訂 SSO 指令碼。	1. 在其他資訊區段中複製 SSO 指令碼。 2. 在 `Param`區段中，為您的 AWS 環境定義下列變數的值： • `DefaultRoleName` – 預設要使用的 IAM 角色或許可集。 • `Region` – 部署 IAM Identity Center 的 AWS 區域。如需區域及其代碼的完整清單，請參閱區域端點。 • `StartUrl` – 用來存取 IAM Identity Center 登入頁面的 URL。使用與指令碼中範例值相同的格式。 • `EnvironmentName` – 參考此指令碼複本的簡短名稱，用於在相同工作階段中執行多個指令碼複本時。 3. 在讀取 的第 10 行下`# Add your Account Information` ，編輯雜湊資料表中的下列值，以反映您的環境：	雲端管理員

任務	Description	所需的技能
	• `Profile` – 存放暫時登入資料的 AWS CLI 設定檔名稱。 • `AccountId` – 您要擷取登入資料的 AWS 帳戶 ID。 • `RoleName` – 您要使用的 IAM Identity Center 角色或許可集的名稱。您可以將此保留`$DefaultRoleName` 為您想要使用在 `Param`區段中定義的相同角色。 雜湊表中的每一行都必須以逗號結尾，最後一個除外。	

執行 SSO 指令碼。	建議您使用下列命令，在 PowerShell shell 中執行自訂指令碼。
	
	`./Set-AwsCliSsoCredentials.ps1`
	
	或者，您也可以輸入下列命令，從另一個 shell 執行指令碼。
	
	`pwsh Set-AwsCliSsoCredentials.ps1`
		雲端管理員

疑難排解

問題	解決方案
`No Access` 錯誤	您使用的 IAM 角色沒有存取您在 `RoleName` 參數中定義之角色或許可集的許可。更新您正在使用之角色的許可，或在指令碼中定義不同的角色或許可集。

相關資源

- 組態設定存放在哪裡？ (AWS CLI 文件)
- 設定 AWS CLI 以使用 AWS IAM Identity Center (AWS CLI 文件)
- 使用具名設定檔 (AWS CLI 文件)

其他資訊

SSO 指令碼

在下列指令碼中，將角括號 (<>) 中的預留位置取代為您自己的資訊，並移除角括號。

```
Set-AwsCliSsoCredentials.ps1
Param(
    $DefaultRoleName = '<AWSAdministratorAccess>',
    $Region          = '<us-west-2>',
    $StartUrl        = "<https://d-12345abcde.awsapps.com/start/>",
    $EnvironmentName = "<CompanyName>"
)
Try {$SsoAwsAccounts = (Get-Variable -name "$($EnvironmentName)SsoAwsAccounts" -Scope
 Global -ErrorAction 'SilentlyContinue').Value.Clone()}
Catch {$SsoAwsAccounts = $False}
if (-not $SsoAwsAccounts) { $SsoAwsAccounts = @(
# Add your account information in the list of hash tables below, expand as necessary,
 and do not forget the commas
    @{Profile = "<Account1>"      ; AccountId = "<012345678901 >"; RoleName =
 $DefaultRoleName },
    @{Profile = "<Account2>"      ; AccountId = "<123456789012>"; RoleName =
 "<AWSReadOnlyAccess>" }
)}
```

```
$ErrorActionPreference = "Stop"
if (-not (Test-Path ~\.aws))      { New-Item ~\.aws -type Directory }
if (-not (Test-Path ~\.aws\credentials)) { New-Item ~\.aws\credentials -type File }
$CredentialFile = Resolve-Path ~\.aws\credentials
$PsuedoCreds    = @{AccessKey =
 'AKAEXAMPLE123ACCESS';SecretKey='PsuedoS3cret4cceSSKey123PsuedoS3cretKey'} # Pseudo
 Creds, do not edit.
Try {$SSOTokenExpire = (Get-Variable -Scope Global -Name
 "$($EnvironmentName)SSOTokenExpire" -ErrorAction 'SilentlyContinue').Value} Catch
 {$SSOTokenExpire = $False}
Try {$SSOToken       = (Get-Variable -Scope Global -Name "$($EnvironmentName)SSOToken"
 -ErrorAction 'SilentlyContinue').Value }      Catch {$SSOToken       = $False}
if ( $SSOTokenExpire -lt (Get-Date) ) {
    $SSOToken = $Null
    $Client   = Register-SSOOIDCClient -ClientName cli-sso-client -ClientType public -
Region $Region @PsuedoCreds
    $Device   = $Client | Start-SSOOIDCDeviceAuthorization -StartUrl $StartUrl -Region
 $Region @PsuedoCreds
    Write-Host "A Browser window should open. Please login there and click ALLOW." -
NoNewline
    Start-Process $Device.VerificationUriComplete
    While (-Not $SSOToken){
        Try {$SSOToken = $Client | New-SSOOIDCToken -DeviceCode $Device.DeviceCode -
GrantType "urn:ietf:params:oauth:grant-type:device_code" -Region $Region @PsuedoCreds}
        Catch {If ($_.Exception.Message -notlike "*AuthorizationPendingException*")
{Write-Error $_.Exception} ; Start-Sleep 1}
    }
    $SSOTokenExpire = (Get-Date).AddSeconds($SSOToken.ExpiresIn)
    Set-Variable -Name "$($EnvironmentName)SSOToken" -Value $SSOToken -Scope Global
    Set-Variable -Name "$($EnvironmentName)SSOTokenExpire" -Value $SSOTokenExpire -
Scope Global
}
$CredsTime     = $SSOTokenExpire - (Get-Date)
$CredsTimeText = ('{0:D2}:{1:D2}:{2:D2} left on SSO Token' -f $CredsTime.Hours,
 $CredsTime.Minutes, $CredsTime.Seconds).TrimStart("0 :")
for ($i = 0; $i -lt $SsoAwsAccounts.Count; $i++) {
    if (([DateTimeOffset]::FromUnixTimeSeconds($SsoAwsAccounts[$i].CredsExpiration /
 1000)).DateTime -lt (Get-Date).ToUniversalTime()) {
        Write-host "`r
    `rRegistering Profile $($SsoAwsAccounts[$i].Profile)" -NoNewline
        $TempCreds = $SSOToken | Get-SSORoleCredential -AccountId
 $SsoAwsAccounts[$i].AccountId -RoleName $SsoAwsAccounts[$i].RoleName -Region $Region
 @PsuedoCreds
```

```
        [PSCustomObject]@{AccessKey = $TempCreds.AccessKeyId; SecretKey =
 $TempCreds.SecretAccessKey; SessionToken = $TempCreds.SessionToken
        } | Set-AWSCredential -StoreAs $SsoAwsAccounts[$i].Profile -ProfileLocation
 $CredentialFile
        $SsoAwsAccounts[$i].CredsExpiration = $TempCreds.Expiration
    }
}
Set-Variable -name "$($EnvironmentName)SsoAwsAccounts" -Value $SsoAwsAccounts.Clone() -
Scope Global
Write-Host "`r$($SsoAwsAccounts.Profile) Profiles registered, $CredsTimeText"
```

使用 Network Firewall 從傳出流量的伺服器名稱指示擷取 DNS 網域名稱

Kirankumar Chandrashekar，Amazon Web Services

總結

此模式說明如何使用 AWS Network Firewall 來收集傳出網路流量 HTTPS 標頭中伺服器名稱指示 (SNI) 提供的 DNS 網域名稱。Network Firewall 是一項受管服務，可讓您輕鬆部署 Amazon Virtual Private Cloud (Amazon VPC) 的重要網路保護，包括使用防火牆保護傳出流量的能力，該防火牆會封鎖不符合特定安全要求的封包。保護特定 DNS 網域名稱的傳出流量稱為傳出篩選，這是監控並可能限制傳出資訊從一個網路流向另一個網路的做法。

擷取通過 Network Firewall 的 SNI 資料後，您可以使用 Amazon CloudWatch Logs 和 AWS Lambda 將資料發佈至產生電子郵件通知的 Amazon Simple Notification Service (Amazon SNS) 主題。電子郵件通知包含伺服器名稱和其他相關的 SNI 資訊。此外，您可以使用此模式的輸出，透過防火牆規則，在 SNI 中依網域名稱允許或限制傳出流量。如需詳細資訊，請參閱 Network Firewall 文件中的在 AWS Network Firewall 中使用具狀態規則群組。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 在 Linux、macOS 或 Windows 上安裝和設定 AWS Command Line Interface (AWS CLI) 第 2 版。macOS
- Network Firewall，在 Amazon VPC 中設定和設定，並用於檢查傳出流量。您可以設定 Network Firewall 以使用下列任何 VPC 組態：
  - 具有網際網路閘道的簡單單一區域架構
  - 具有網際網路閘道的多區域架構
  - 具有網際網路閘道和 NAT 閘道的架構

Architecture

下圖顯示如何使用 Network Firewall 從傳出網路流量收集 SNI 資料，然後使用 CloudWatch Logs 和 Lambda 將該資料發佈至 SNS 主題。

該圖顯示以下工作流程：

1. Network Firewall 會從傳出網路流量的 HTTPS 標頭中的 SNI 資料收集網域名稱。
2. CloudWatch Logs 會監控 SNI 資料，並在傳出網路流量通過 Network Firewall 時叫用 Lambda 函數。
3. Lambda 函數會讀取 CloudWatch Logs 擷取的 SNI 資料，然後將該資料發佈至 SNS 主題。
4. SNS 主題會傳送電子郵件通知給您，其中包含 SNI 資料。

自動化和擴展

- 您可以使用 AWS CloudFormation 來建立此模式，方法是使用基礎設施做為程式碼。

技術堆疊

- Amazon CloudWatch Logs
- Amazon SNS
- Amazon VPC
- AWS Lambda
- AWS Network Firewall

工具

AWS 服務

- Amazon CloudWatch Logs – 您可以使用 Amazon CloudWatch Logs 從 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、AWS CloudTrail、Amazon Route 53 和其他來源監控、存放和存取您的日誌檔案。
- Amazon SNS – Amazon Simple Notification Service (Amazon SNS) 是一種受管服務，可將訊息從發佈者交付給訂閱者（也稱為生產者和消費者）。
- Amazon VPC – Amazon Virtual Private Cloud (Amazon VPC) 會佈建 AWS 雲端的邏輯隔離區段，您可以在您已定義的虛擬網路中啟動 AWS 資源。這個虛擬網路與您在資料中心中操作的傳統網路非常相似，且具備使用 AWS 可擴展基礎設施的優勢。
- AWS Lambda – AWS Lambda 是一種運算服務，可讓您執行程式碼，而無需佈建或管理伺服器。
- AWS Network Firewall – AWS Network Firewall 是一項受管服務，可讓您輕鬆為所有 Amazon VPCs 部署必要的網路保護。

史詩

建立 Network Firewall 的 CloudWatch 日誌群組

任務	Description	所需的技能
建立 CloudWatch 日誌群組。	1. 登入 AWS 管理主控台並開啟 CloudWatch 主控台。 2. 在導覽窗格中，選擇 Log groups (日誌群組)。 3. 選擇 Actions (動作)，然後選擇 Create log group (建立日誌群組)。 4. 輸入日誌群組名稱，然後選擇 Create log group (建立日誌群組)。 如需詳細資訊，請參閱 CloudWatch 文件中的使用日誌群組和日誌串流。	雲端管理員

建立 SNS 主題和訂閱

任務	Description	所需的技能
建立 SNS 主題。	若要建立 SNS 主題，請遵循 Amazon SNS 文件中的指示。	雲端管理員
訂閱 SNS 主題的端點。	若要將電子郵件地址訂閱為您建立的 SNS 主題的端點，請遵循 Amazon SNS 文件中的指示。針對通訊協定，選擇電子郵件/電子郵件 JSON。	雲端管理員

任務	Description	所需的技能
	Note 您也可以根據您的需求選擇不同的端點。	

在 Network Firewall 中設定記錄

任務	Description	所需的技能
啟用防火牆記錄。	1. 登入 AWS 管理主控台並開啟 Amazon VPC 主控台。 2. 在導覽窗格的 NETWORK FIREWALL 下，選擇防火牆。 3. 在防火牆區段中，選擇您要從 SNI 擷取傳出流量伺服器名稱的防火牆。 4. 選擇防火牆詳細資訊索引標籤，然後在記錄區段中選擇編輯。 5. 針對日誌類型，選取提醒。針對提醒，選取 CloudWatch 日誌群組的 ForLog 目的地。 CloudWatch 6. 對於 CloudWatch 日誌群組，搜尋並選擇您先前建立的日誌群組，然後選擇儲存。 如需使用 CloudWatch Logs 做為 Network Firewall 日誌目的地的詳細資訊，請參閱	雲端管理員

任務	Description	所需的技能
	Network Firewall 文件中的 Amazon CloudWatch Logs。	

在 Network Firewall 中設定具狀態規則

任務	Description	所需的技能
建立具狀態規則。	1. 登入 AWS 管理主控台並開啟 Amazon VPC 主控台。 2. 在導覽窗格的 NETWORK FIREWALL 下，選擇網路防火牆規則群組。 3. 選擇建立網路防火牆規則群組。 4. 在建立網路防火牆規則群組頁面上，針對規則群組類型，選擇具狀態規則群組。注意：如需詳細資訊，請參閱在 AWS Network Firewall 中使用具狀態規則群組。 5. 在具狀態規則群組區段中，輸入規則群組的名稱和描述。 6. 針對容量，設定您要允許狀態規則群組的最大容量（上限為 30，000)。注意：您無法在建立規則群組之後變更此設定。如需如何計算容量的資訊，請參閱在 AWS Network Firewall 中設定規則群組容量。如需最大	雲端管理員

任務	Description	所需的技能
	設定的資訊，請參閱 AWS Network Firewall 配額。 7. 針對具狀態規則群組選項，選取 5 個元組。 8. 在具狀態規則順序區段中，選擇預設。 9. 在規則變數區段中，保留預設值。 10.在新增規則區段中，選擇通訊協定的 TLS。對於來源，請選擇任何。對於來源連接埠，請選擇任何連接埠。對於目的地，請選擇任何。對於目的地連接埠，請選擇任何連接埠。針對流量方向，選擇轉送。針對動作，選擇提醒。選擇新增規則。 11選擇建立具狀態規則群組。	

任務	Description	所需的技能
將具狀態規則與 Network Firewall 建立關聯。	1. 登入 AWS 管理主控台並開啟 Amazon VPC 主控台。 2. 在導覽窗格中的 NETWORK FIREWALL 下，選擇防火牆。 3. 選擇您要從 SNI 擷取伺服器名稱用於傳出流量的防火牆。 4. 在具狀態規則群組區段中，選擇動作，然後選擇新增未受管具狀態規則群組。 5. 在新增未受管理的狀態規則群組頁面上，選取您先前建立的狀態規則群組，然後選擇新增狀態規則群組。	雲端管理員

建立 Lambda 函數以讀取日誌

任務	Description	所需的技能

建立 Lambda 函數的程式碼。	在可從 Network Firewall 讀取 CloudWatch Logs 事件用於傳出流量的整合開發環境 (IDE) 中，貼上下列 Python 3 程式碼，並以<SNS-topic-ARN>您的值取代 ：
	`import json`
	`import gzip`
	`import base64`
	`import boto3`
	`sns_client = boto3.cli`
	`ent('sns')`
		應用程式開發人員

任務	Description	所需的技能

	def lambda_handler(event, context):
	    decoded_event = json.loads(gzip.decompress(base64.b64decode(event['awslogs']['data'])))
	    body = '''
	    {filtermatch}
	    '''.format(
	        loggroup=decoded_event['logGroup'],
	        logstream=decoded_event['logStream'],
	        filtermatch=decoded_event['logEvents'][0]['message'],
	    )
	    print(body)
	    filterMatch = json.loads(body)
	    data = []
	    if 'http' in filterMatch['event']:
	        data.append(filterMatch['event']['http']['hostname'])
	    elif 'tls' in filterMatch['event']:
	        data.append(filterMatch['event']['tls']['sni'])
	    result = 'Domain accessed ' + 1*' ' + (data[0]) + 1*' ' 'via AWS Network Firewall ' + 1*' ' + (filterMatch['firewall_name'])	

任務	Description	所需的技能

	<pre> print(result)
	    message = {'ServerName': result}
	    send_to_sns = sns_client.publish(
	        TargetArn=<SNS-topic-ARN>,
	 #Replace with the SNS topic ARN
	        Message=json.dumps({'default': json.dumps(message),
	
	    'sms': json.dumps(message),
	
	   'email': json.dumps(message)}),
	        Subject='Server Name passed through the Network Firewall',
	        MessageStructure='json'
	    )</pre>
	此程式碼範例會剖析 CloudWatch Logs 內容，並擷取 SNI 在 HTTPS 標頭中提供的伺服器名稱。	

建立 Lambda 函數。	若要建立 Lambda 函數，請遵循 Lambda 文件中的指示，然後選擇 Python 3.9 for Runtime。	雲端管理員
將程式碼新增至 Lambda 函數。	若要將 Python 程式碼新增至您先前建立的 Lambda 函數，請遵循 Lambda 文件中的指示。	雲端管理員

任務	Description	所需的技能
新增 CloudWatch Logs 做為 Lambda 函數的觸發條件。	1. 登入 AWS 管理主控台並開啟 Lambda 主控台。 2. 在導覽窗格中，選擇函數，然後選擇您先前建立的函數。 3. 在函數概觀區段中，選擇新增觸發條件。 4. 在新增觸發頁面上的觸發組態區段中，選擇 CloudWatch Logs，然後選擇新增。 5. 針對日誌群組，選擇您先前建立的 CloudWatch 日誌群組。 6. 在篩選條件名稱中，輸入篩選條件的名稱。 7. 選擇新增。 8. 在函數頁面的組態索引標籤的觸發區段中，選取您剛新增的觸發，然後選擇啟用。 如需詳細資訊，請參閱 Lambda 文件中的搭配使用 Lambda 與 CloudWatch Logs。	雲端管理員

任務	Description	所需的技能
新增 SNS 發佈許可。	將 sns：Publish 許可新增至 Lambda 執行角色，以便 Lambda 可以進行 API 呼叫，將訊息發佈至 SNS。 1. 尋找您先前建立之 Lambda 函數的執行角色。 2. 將下列政策新增至您的 AWS Identity and Access Management (IAM) 角色： `{ "Version": "2012-10-17", "Statement": [ { "Sid": "AllowSNSPublish", "Effect": "Allow", "Action": [ "sns:GetTopicAttributes", "sns:Subscribe", "sns:Unsubscribe", "sns:Publish" ], "Resource": "*" } ] }`	雲端管理員

測試 SNS 通知的功能

任務	Description	所需的技能
透過 Network Firewall 傳送流量。	1. 傳送或等待 HTTPS 流量通過 Network Firewall。 2. 檢查流量通過 Network Firewall 時從 AWS 收到的 SNS 通知電子郵件。電子郵件包含傳出流量的 SNI 詳細資訊。例如，如果存取的網域名稱為 https://aws.amazon.com 且訂閱通訊協定為 EMAIL-JSON，則從上述 Lambda 程式碼產生的電子郵件將具有下列內容： `{ "Type": "Notification", "MessageId": "<messageID>", "TopicArn": "arn:aws:sns:us-west-2:123456789:testSNSTopic", "Subject": "Server Name passed through the Network Firewall", "Message": "{\"ServerName\": \"Domain 'aws.amazon.com' accessed via AWS Network Firewall 'AWS-Network-Firewall-Multi-AZ-firewall\"}",`	測試工程師

任務	Description	所需的技能
	<pre>"Timestamp": "2022-03-22T04:10:04.217Z", "SignatureVersion": "1", "Signature": "<Signature>", "SigningCertURL": "<SigningCertUrl>", "UnsubscribeURL": "<UnsubscribeURL>" }</pre> 然後，遵循 Amazon CloudWatch 文件中的指示，在 Amazon CloudWatch 中檢查 Network Firewall 警示日誌。提醒日誌會顯示下列輸出： <pre>{ "firewall_name": "AWS-Network-Firewall-Multi-AZ-firewall", "availability_zone": "us-east-2b", "event_timestamp": "<event timestamp>", "event": { "timestamp": "2021-03-22T04:10:04.214222+0000", "flow_id": <flow ID>, "event_type": "alert", "src_ip": "10.1.3.76",</pre>	

任務	Description	所需的技能
	"src_port": 22761, "dest_ip": "99.86.59.73", "dest_port": 443, "proto": "TCP", "alert": { "action": "allowed", "signature_id": 2, "rev": 0, "signature": "", "category": "", "severity": 3 }, "tls": { "subject": "CN=aws.amazon.com", "issuerdn": "C=US, O=Amazon, OU=Server CA 1B, CN=Amazon", "serial": "<serial number>", "fingerprint": "<fingerprint ID>", "sni": "aws.amazon.com", "version": "TLS 1.2", "notbefore": "2020-09-30T00:00:00", "notafter": "2021-09-23T12:00:00",	

任務	Description	所需的技能
	"ja3": {}, "ja3s": {} }, "app_proto": "tls" } }	

使用 Terraform 自動為組織啟用 Amazon GuardDuty

Aarthi Kannan，Amazon Web Services

總結

Amazon GuardDuty 會持續監控您的 Amazon Web Services (AWS) 帳戶，並使用威脅情報來識別 AWS 環境中的非預期和潛在惡意活動。手動為多個帳戶或組織、跨多個 AWS 區域或透過 AWS 管理主控台啟用 GuardDuty 可能會很麻煩。您可以使用基礎設施即程式碼 (IaC) 工具來自動化程序，例如可在雲端中佈建和管理多帳戶、多區域服務和資源的 Terraform。

AWS 建議使用 AWS Organizations 在 GuardDuty 中設定和管理多個帳戶。此模式會遵循該建議。這種方法的一個好處是，當新帳戶建立或新增至組織時，GuardDuty 將在所有支援區域的這些帳戶中自動啟用，而不需要手動介入。

此模式示範如何使用 HashiCorp Terraform 為組織中的三個或多個 Amazon Web Services (AWS) 帳戶啟用 Amazon GuardDuty。此模式隨附的範本程式碼會執行下列動作：

- 為 AWS Organizations 中目標組織目前成員的所有 AWS 帳戶啟用 GuardDuty
- 開啟 GuardDuty 中的自動啟用功能，這會為未來新增至目標組織的任何帳戶自動啟用 GuardDuty
- 可讓您選取要啟用 GuardDuty 的區域
- 使用組織的安全帳戶做為 GuardDuty 委派管理員
- 在記錄帳戶中建立 Amazon Simple Storage Service (Amazon S3) 儲存貯體，並設定 GuardDuty 發佈此儲存貯體中所有帳戶的彙總調查結果
- 根據預設，指派生命週期政策，在 365 天後將調查結果從 S3 儲存貯體轉移至 Amazon S3 Glacier Flexible Retrieval 儲存體

您可以手動執行此範本程式碼，也可以將其整合到您的持續整合和持續交付 (CI/CD) 管道中。

目標對象

對於具有 Terraform、Python、GuardDuty 和 AWS Organizations 經驗的使用者，建議使用此模式。

先決條件和限制

先決條件

- 作用中的 AWS 帳戶
- 組織是在 AWS Organizations 中設定，且至少包含下列三個帳戶：

  - 管理帳戶 – 這是您從中部署 Terraform 程式碼的帳戶，無論是獨立或作為 CI/CD 管道的一部分。Terraform 狀態也會存放在此帳戶中。
  - 安全帳戶 – 此帳戶用作 GuardDuty 委派管理員。如需詳細資訊，請參閱 GuardDuty 委派管理員的重要考量 (GuardDuty 文件）。
  - 記錄帳戶 – 此帳戶包含 GuardDuty 發佈所有成員帳戶彙總調查結果的 S3 儲存貯體。

  如需如何使用所需組態設定組織的詳細資訊，請參閱建立帳戶結構 (AWS Well-Architected 實驗室）。
- Amazon S3 儲存貯體和 Amazon DynamoDB 資料表，可做為遠端後端，將 Terraform 的狀態存放在管理帳戶中。如需針對 Terraform 狀態使用遠端後端的詳細資訊，請參閱 S3 後端 (Terraform 文件）。如需使用 S3 後端設定遠端狀態管理的程式碼範例，請參閱 remote-state-s3-backend (Terraform Registry)。請注意以下要求：
  - S3 儲存貯體和 DynamoDB 資料表必須位於相同的區域。
  - 建立 DynamoDB 資料表時，分割區索引鍵必須是 `LockID`（區分大小寫），而分割區索引鍵類型必須是字串。所有其他資料表設定必須處於其預設值。如需詳細資訊，請參閱關於主索引鍵和建立資料表 (DynamoDB 文件）。
- 用來存放 S3 儲存貯體存取日誌的 S3 儲存貯體，GuardDuty 會在其中發佈問題清單。如需詳細資訊，請參閱啟用 Amazon S3 伺服器存取記錄 (AmazonS3 文件）。如果您要部署到 AWS Control Tower 登陸區域，您可以為此重複使用日誌封存帳戶中的 S3 儲存貯體。
- 已安裝並設定 Terraform 0.14.6 版或更新版本。如需詳細資訊，請參閱入門 – AWS (Terraform 文件）。
- 已安裝並設定 Python 3.9.6 版或更新版本。如需詳細資訊，請參閱來源版本 (Python 網站）。
- 已安裝適用於 Python (Boto3) 的 AWS 開發套件。如需詳細資訊，請參閱安裝 (Boto3 文件）。
- jq 已安裝並設定。如需詳細資訊，請參閱下載 jq (jq 文件）。

限制

- 此模式支援 macOS 和 Amazon Linux 2 作業系統。此模式尚未經過測試可用於 Windows 作業系統。

> **Note**
>
> Amazon Linux 2 即將終止支援。如需詳細資訊，請參閱 Amazon Linux 2 FAQs。

- GuardDuty 不得在任何目標區域的任何帳戶中啟用。

- 此模式中的 IaC 解決方案不會部署先決條件。
- 此模式專為符合下列最佳實務的 AWS 登陸區域而設計：
  - 登陸區域是使用 AWS Control Tower 建立。
  - 個別 AWS 帳戶用於安全性和記錄。

產品版本

- Terraform 0.14.6 版或更新版本。範例程式碼已針對 1.2.8 版進行測試。
- Python 3.9.6 版或更新版本。

Architecture

本節提供此解決方案的高階概觀，以及範例程式碼所建立的架構。下圖顯示在單一 AWS 區域中跨組織中各種帳戶部署的資源。

1. Terraform 會在安全帳戶和記錄帳戶中建立 GuardDutyTerraformOrgRole AWS Identity and Access Management (IAM) 角色。
2. Terraform 會在記錄帳戶中的預設 AWS 區域中建立 S3 儲存貯體。此儲存貯體用作發佈目的地，以彙總組織中所有區域和所有帳戶的所有 GuardDuty 調查結果。Terraform 也會在安全帳戶中建立 AWS Key Management Service (AWS KMS) 金鑰，用於加密 S3 儲存貯體中的調查結果，並將 S3 儲存貯體中的調查結果自動封存至 S3 Glacier Flexible Retrieval 儲存體。
3. 從管理帳戶，Terraform 會將安全帳戶指定為 GuardDuty 的委派管理員。這表示安全帳戶現在會管理所有成員帳戶的 GuardDuty 服務，包括 管理帳戶。個別成員帳戶無法自行暫停或停用 GuardDuty。
4. Terraform 會在安全帳戶中為 GuardDuty 委派管理員建立 GuardDuty 偵測器。
5. 如果尚未啟用，Terraform 會在 GuardDuty 中啟用 S3 保護。如需詳細資訊，請參閱 Amazon GuardDuty 中的 Amazon S3 保護 Amazon GuardDuty (GuardDuty 文件）。
6. Terraform 會將組織中的所有目前作用中成員帳戶註冊為 GuardDuty 成員。
7. Terraform 會設定 GuardDuty 委派管理員，將彙總的問題清單從所有成員帳戶發佈到記錄帳戶中的 S3 儲存貯體。
8. Terraform 會針對您選擇的每個 AWS 區域重複步驟 3 到 7。

自動化和擴展

提供的範例程式碼已模組化，因此您可以將其整合到您的 CI/CD 管道以進行自動化部署。

工具

AWS 服務

- Amazon DynamoDB 是一項全受管 NoSQL 資料庫服務，可提供快速、可預期且可擴展的效能。
- Amazon GuardDuty 是一項持續的安全監控服務，可分析和處理日誌，以識別 AWS 環境中未預期和可能未經授權的活動。
- AWS Identity and Access Management (IAM) 可透過控制已驗證並獲授權使用的人員，協助您安全地管理對 AWS 資源的存取。
- AWS Key Management Service (AWS KMS) 可協助您建立和控制密碼編譯金鑰，以保護資料。
- AWS Organizations 是一種帳戶管理服務，可協助您將多個 AWS 帳戶合併到您建立並集中管理的組織。
- Amazon Simple Storage Service (Amazon S3) 是一種雲端型物件儲存服務，可協助您儲存、保護和擷取任何數量的資料。
- AWS SDK for Python (Boto3) 是一種軟體開發套件，可協助您整合 Python 應用程式、程式庫或指令碼與 AWS 服務。

其他工具和服務

- HashiCorp Terraform 是一種命令列界面應用程式，可協助您使用程式碼來佈建和管理雲端基礎設施和資源。
- Python 是一種一般用途的程式設計語言。
- jq 是一種命令列處理器，可協助您使用 JSON 檔案。

程式碼儲存庫

此模式的程式碼可在 GitHub 的 amazon-guardduty-for-aws-organizations-with-terraform 儲存庫中取得。

史詩

在組織中啟用 GuardDuty

任務	Description	所需的技能
複製儲存庫。	在 Bash shell 中，執行下列命令。在其他資訊區段中複製儲存庫，您可以複製包含 GitHub 儲存庫 URL 的完整命令。這會從 GitHub 複製 amazon-guardduty-for-aws-organizations-with-terraform 儲存庫。 `git clone <github-repository-url>`	DevOps 工程師
編輯 Terraform 組態檔案。	1. 在複製儲存庫的 `root`資料夾中，執行下列命令來複寫 configuration.json.sample 檔案。 `cp configuration.json.sample configuration.json` 2. 編輯新的 configuration.json 檔案，並定義下列每個變數的值： • `management_acc_id` – 管理帳戶的帳戶 ID。 • `delegated_admin_acc_id` – 安全帳戶的帳戶 ID。 • `logging_acc_id` – 記錄帳戶的帳戶 ID。	DevOps 工程師、一般 AWS、Terraform、Python

任務	Description	所需的技能
	• `target_regions` – 您想要啟用 GuardDuty 的 AWS 區域逗號分隔清單。 • `organization_id` – 您啟用 GuardDuty 之組織的 AWS Organizations ID。 • `default_region` – 您的 Terraform 狀態存放在管理帳戶中的區域。這是您為 Terraform 後端部署 S3 儲存貯體和 DynamoDB 資料表的相同區域。 • `role_to_assume_for_role_creation` – 您要指派給安全和記錄帳戶中新 IAM 角色的名稱。您可以在下一個故事中建立此新角色。Terraform 會擔任此角色，以在安全性和記錄帳戶中建立 `GuardDutyTerraformOrgRole` IAM 角色。 • `finding_publishing_frequency` – GuardDuty 將調查結果發佈至 S3 儲存貯體的頻率。 • `guardduty_findings_bucket_region` – 您想要為已發佈的問題清	

任務	Description	所需的技能
	單建立 S3 儲存貯體的偏好區域。 • `logging_acc_s3_bucket_name` – 已發佈問題清單之 S3 儲存貯體的偏好名稱。 • `security_acc_kms_key_alias` – 用於加密 GuardDuty 調查結果之金鑰的 AWS KMS 別名。 • `s3_access_log_bucket_name` – 預先存在的 S3 儲存貯體名稱，您要收集用於 GuardDuty 調查結果的 S3 儲存貯體的存取日誌。此儲存貯體應與 GuardDuty 調查結果儲存貯體位於相同的 AWS 區域。 • `tfm_state_backend_s3_bucket` – 儲存 Terraform 遠端後端狀態的既有 S3 儲存貯體名稱。 • `tfm_state_backend_dynamodb_table` – 預先存在的 DynamoDB 資料表名稱，用於鎖定 Terraform 狀態。 3. 儲存並關閉 組態檔案。	

任務	Description	所需的技能
為新的 IAM 角色產生 CloudFormation 範本。	此模式包含用於建立兩個 CloudFormation 範本的 IaC 解決方案。這些範本會建立 Terraform 在設定過程中使用的兩個 IAM 角色。這些範本遵循最低權限許可的安全最佳實務。 1. 在 Bash shell 的儲存庫`root`資料夾中，導覽至 `cfn-templates/` 。此資料夾包含具有 stubs 的 CloudFormation 範本檔案。 2. 執行下列命令。這會將 stubs 取代為您在 configuration.json 檔案中提供的值。 `bash scripts/replace_config_stubs.sh` 3. 確認已在 `cfn-templates/` 資料夾中建立下列 CloudFormation 範本： • management-account-role.yaml – 此檔案包含管理帳戶中 IAM 角色的角色定義和相關許可，其具有完成此模式所需的最低許可。 • role-to-assume-for-role-creation.yaml – 此檔案包含安全與記錄帳戶中 IAM 角色的角色定義和相關許可。Terraform 會擔任此	DevOps 工程師，一般 AWS

任務	Description	所需的技能
	角色，以便在這些帳戶中建立 GuardDutyTerraformOrgRole 角色。	
建立 IAM 角色。	遵循建立堆疊 (CloudFormation 文件） 中的指示，執行下列動作： 1. 在安全性和記錄帳戶中部署 role-to-assume-for-role-creation.yaml 堆疊。 2. 在管理帳戶中部署 management-account-role.yaml 堆疊。當您成功建立堆疊並查看CREATE_COMPLETE 堆疊狀態時，請在輸出中記下此新角色的 Amazon Resource Name (ARN)。	DevOps 工程師，一般 AWS
擔任管理帳戶中的 IAM 角色。	作為安全最佳實務，我們建議您在繼續之前擔任新的 management-account-role IAM 角色。在 AWS 命令列界面 (AWS CLI) 中，於其他資訊區段的擔任管理帳戶 IAM 角色中輸入 命令。	DevOps 工程師，一般 AWS

任務	Description	所需的技能
執行設定指令碼。	在儲存庫`root`資料夾中，執行下列命令來啟動設定指令碼。 `bash scripts/full-setup.sh` full-setup.sh 指令碼會執行下列動作： • 將所有組態值匯出為環境變數 • 為每個 Terraform 模組產生 backend.tf 和 terraform.tfvars 程式碼檔案 • 透過 AWS CLI 為組織中的 GuardDuty 啟用受信任存取。 • 將組織狀態匯入 Terraform 狀態 • 建立 S3 儲存貯體以在記錄帳戶中發佈問題清單 • 建立 AWS KMS 金鑰以加密安全帳戶中的問題清單 • 在所有選取的區域中啟用整個組織的 GuardDuty，如架構一節中所述	DevOps 工程師，Python

（選用）在組織中停用 GuardDuty

任務	Description	所需的技能
執行清除指令碼。	如果您使用此模式為組織啟用 GuardDuty，並想要停	DevOps 工程師、一般 AWS、Terraform、Python

任務	Description	所需的技能
	用 GuardDuty，請在儲存庫`root`資料夾中執行下列命令來啟動 cleanup-gd.sh 指令碼。 `bash scripts/cleanup-gd.sh` 此指令碼會停用目標組織中的 GuardDuty、移除任何已部署的資源，並在使用 Terraform 啟用 GuardDuty 之前，將組織還原至先前的狀態。 **Note** 此指令碼不會從本機和遠端後端移除 Terraform 狀態檔案或鎖定檔案。如果您需要這樣做，您必須手動執行這些動作。此外，此指令碼不會刪除匯入的組織或其管理的帳戶。GuardDuty 的信任存取權不會在清除指令碼中停用。	
移除 IAM 角色。	刪除使用 role-to-assume-for-role-creation.yaml 和 management-account-role.yaml CloudFormation 範本建立的堆疊。如需詳細資訊，請參閱刪除堆疊 (CloudFormation 文件）。	DevOps 工程師，一般 AWS

相關資源

AWS 文件

- 管理多個帳戶 (GuardDuty 文件)
- 授予最低權限 (IAM 文件)

AWS 行銷

- Amazon GuardDuty
- AWS Organizations

其他資源

- Terraform
- Terraform CLI 文件

其他資訊

複製儲存庫

執行下列命令來複製 GitHub 儲存庫。

```
git clone https://github.com/aws-samples/amazon-guardduty-for-aws-organizations-with-terraform
```

擔任管理帳戶 IAM 角色

若要在管理帳戶中擔任 IAM 角色，請執行下列命令。<IAM role ARN> 將 取代為 IAM 角色的 ARN。

```
export ROLE_CREDENTIALS=$(aws sts assume-role --role-arn <IAM role ARN> --role-session-name AWSCLI-Session --output json)
export AWS_ACCESS_KEY_ID=$(echo $ROLE_CREDENTIALS | jq .Credentials.AccessKeyId | sed 's/"//g')
export AWS_SECRET_ACCESS_KEY=$(echo $ROLE_CREDENTIALS | jq .Credentials.SecretAccessKey | sed 's/"//g')
```

```
export AWS_SESSION_TOKEN=$(echo $ROLE_CREDENTIALS | jq .Credentials.SessionToken | sed
 's/"//g')
```

使用 驗證 PCI DSS 4.0 的操作最佳實務 AWS Config

Tala Qraitem 和 Alex Goff，Amazon Web Services

總結

支付卡產業資料安全標準 (PCI DSS) 概述了必要的技術和操作通訊協定，以協助保護付款資料。PCI DSS 旨在鼓勵和增強支付卡帳戶的資料安全性。它還有助於全球採用一致的安全措施。雖然它專為具有支付卡帳戶資料的環境而設計，但您可以使用 PCI DSS 來協助防範威脅並保護付款生態系統中的其他元素。

PCI DSS 4.0 版已發佈，以因應不斷變化的需求、提供釐清或其他指引，並改善標準的結構和格式。如需變更的詳細資訊，請參閱從 PCI DSS 3.2.1 版到 4.0 版的變更摘要。

AWS Config 一致性套件是 AWS Config 規則和修補動作的集合，可協助您建立安全、操作或成本最佳化控管檢查。您可以在 AWS 帳戶 和 中將一致性套件部署為單一實體 AWS 區域，也可以在 中跨組織部署 AWS Organizations。

PCI DSS 4.0 版的一致性套件會增強並建置在 3.2.1 版的一致性套件上。一致性套件中的規則會對應至標準中的規則。如需詳細資訊，請參閱附件區段中提供的映射。您可以選擇此一致性套件的兩個版本：一個包含全域資源類型，另一個則排除它們。

> **Important**
>
> 一致性套件的設計並非完全確保符合特定的控管或合規標準。您有責任自行評估用量是否符合適用的法律和法規要求。

先決條件和限制

先決條件

- 具有作用中的 AWS 帳戶。
- 設定 AWS Config。
- 符合一致性套件的先決條件。
- 部署 PCI DSS 3.2.1 版一致性套件。
- 具有存取 AWS Config 和管理一致性套件的許可。如需範例政策，請參閱此模式的其他資訊一節。

限制

- 您的 AWS 帳戶 具有預設配額，先前稱為每個配額的限制 AWS 服務。除非另有說明，否則每個配額都是區域特定的。您可以為某些配額請求增加，但並非所有配額都可以增加。請務必熟悉AWS Config 服務限制，包括單一帳戶一致性套件和組織一致性套件的限制。
- 包含全域資源類型的此一致性套件版本僅適用於在 us-east-1區域中部署。
- 排除全域資源類型的此一致性套件版本僅適用於在下列區域中部署：
  - ap-east-1
  - ap-south-1
  - ap-northeast-2
  - ap-southeast-1
  - ap-southeast-2
  - ap-northeast-1
  - ca-central-1
  - eu-central-1
  - eu-west-1
  - eu-west-2
  - eu-west-3
  - eu-north-1
  - sa-east-1
  - us-east-2
  - us-west-1
  - us-west-2

工具

AWS 服務

- AWS Config 提供 中資源的詳細檢視 AWS 帳戶 及其設定方式。它可協助您識別資源彼此之間的關係，以及其組態隨著時間的變化。
- AWS Systems Manager 可協助您管理在 中執行的應用程式和基礎設施 AWS 雲端。它可簡化應用程式和資源管理、縮短偵測和解決操作問題的時間，並協助您大規模安全地管理 AWS 資源。

程式碼儲存庫

一致性套件位於AWS Config 一致性套件 GitHub 儲存庫中。此儲存庫包含下列與 PCI DSS 4.0 版相關的範本：

- Operational-Best-Practices-for-PCI-DSS-v4.0-including-global-resourcetypes.yaml
- Operational-Best-Practices-for-PCI-DSS-v4.0-excluding-global-resourcetypes.yaml

史詩

部署和管理一致性套件

任務	Description	所需的技能
下載一致性套件。	如果您要在 `us-east-1` 區域中部署一致性套件，請下載 Operational-Best-Practices-for-PCI-DSS-v4.0-including-global-resourcetypes.yaml 範本。 如果您要在不同區域中部署一致性套件，請下載 Operational-Best-Practices-for-PCI-DSS-v4.0-excluding-global-resourcetypes.yaml 範本。	DevOps 工程師
（選用）修改一致性套件。	您可以針對組織的獨特需求修改一致性套件範本。例如，您可以建立自訂修補動作。如需如何建立和修改範本的詳細資訊，請參閱 AWS Config 文件中的為自訂一致性套件建立範本。	一般 AWS
部署一致性套件。	如果您要在目標中部署 AWS 帳戶 或 AWS 區域，請遵循 AWS Config 文件中部署	一般 AWS

任務	Description	所需的技能
	一致性套件中的指示。您可以使用 AWS 管理主控台 或 AWS Command Line Interface (AWS CLI)。 如果您要在 中跨組織部署一致性套件 AWS Organizations，請遵循 AWS Systems Manager 文件中的使用快速設定部署 AWS Config 一致性套件中的指示。	
（選用）編輯一致性套件。	如果您想要編輯一致性套件，請遵循 AWS Config 文件中的編輯一致性套件中的指示。您可以使用 AWS 管理主控台 或 AWS CLI。	一般 AWS
（選用）刪除一致性套件。	如果您想要刪除一致性套件，請遵循 AWS Config 文件中刪除一致性套件中的指示。您可以使用 AWS 管理主控台 或 AWS CLI。	一般 AWS

相關資源

AWS resources

- 的一致性套件 AWS Config(AWS Config 文件）
- 使用快速設定部署一致性套件 AWS Config(Systems Manager 文件）
- (AWS 網站）上的 PCI DSS 合規 AWS
- PCI DSS 4.0 版 AWS（合規指南）

PCI DSS 資源

- PCI DSS 4.0 版資源中樞
- PCI 安全標準委員會文件庫
- 從 PCI DSS 3.2.1 版到 4.0 版的變更摘要

其他資訊

以下是範例 AWS Identity and Access Management (IAM) 政策，允許使用者存取 AWS Config 和管理一致性套件：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "config:PutConfigRule",
                "config:PutConformancePack",
                "config:DeleteConfigRule",
                "config:DeleteRemediationConfiguration",
                "config:DeleteConformancePack",
                "config:PutRemediationConfigurations",
                "config:BatchGetAggregateResourceConfig",
                "config:BatchGetResourceConfig",
                "config:Get*",
                "config:Describe*",
                "config:Deliver*",
                "config:List*",
                "config:Select*"
            ],
            "Resource": "*"
        }
    ]
}
```

附件

若要存取與本文件相關聯的其他內容，請解壓縮下列檔案： attachment.zip

更多模式

- 使用 Session Manager 和 Amazon EC2 Instance Connect 存取堡壘主機
- 使用 AWS Fargate、AWS PrivateLink 和 Network Load Balancer，在 Amazon ECS 上私下存取容器應用程式
- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon ECS 上私下存取容器應用程式
- 使用 AWS PrivateLink 和 Network Load Balancer 在 Amazon EKS 上私下存取容器應用程式
- 允許 EC2 執行個體寫入存取 AMS 帳戶中的 S3 儲存貯體
- 將儲存 AWS CodeCommit 庫與另一個帳戶中 AWS 帳戶 的 Amazon SageMaker AI Studio Classic 建立關聯
- 使用 Amazon Cognito 和 AWS Amplify UI 驗證現有的 React 應用程式使用者
- 使用 AWS Systems Manager 自動化新增或更新 Windows 登錄項目
- 使用 AWS CloudFormation 範本自動化 AWS Glue 中的加密強制執行 AWS CloudFormation
- 使用 Cloud Custodian 和 AWS CDK 將 Systems Manager 的 AWS 受管政策自動連接至 EC2 執行個體設定檔
- 自動加密現有和新的 Amazon EBS 磁碟區
- 使用 Cloud Custodian 封鎖對 Amazon RDS 的公開存取
- 使用 AWS Managed Microsoft AD 和內部部署 Microsoft Active Directory 集中 DNS 解析
- 實作集中式自訂 Checkov 掃描，以在部署 AWS 基礎設施之前強制執行政策
- 在啟動時檢查 EC2 執行個體是否有強制性標籤
- 設定對 Amazon DynamoDB 的跨帳戶存取權
- 使用 Application Load Balancer 在 Oracle WebLogic 上設定 Oracle JD Edwards EnterpriseOne 的 HTTPS 加密
- 為在 Amazon EKS 上執行的應用程式設定交互 TLS 身分驗證
- 使用 設定 Amazon RDS for Microsoft SQL Server 的 Windows 身分驗證 AWS Managed Microsoft AD
- 在 pgAdmin 中使用 SSH 通道進行連線
- 使用 AWS Amplify 建立 React 應用程式，並使用 Amazon Cognito 新增身分驗證
- 為多個 中的傳入網際網路存取建立 Network Access Analyzer 調查結果報告 AWS 帳戶
- 自訂 的 Amazon CloudWatch 提醒 AWS Network Firewall
- 在聊天應用程式自訂動作和 中使用 Amazon Q Developer 部署 ChatOps 解決方案來管理 SAST 掃描結果 CloudFormation

- 搭配 Kiro 和其他編碼助理使用 MCP 伺服器部署即時編碼安全驗證
- 記錄您的 AWS 登陸區域設計
- 在 Amazon RDS 中啟用 PostgreSQL 資料庫執行個體的加密連線
- 加密現有的 Amazon RDS for PostgreSQL 資料庫執行個體
- 在啟動時強制執行 Amazon RDS 資料庫的自動標記
- 在啟動時強制標記 Amazon EMR 叢集
- 確保在啟動時啟用對 Amazon S3 的 Amazon EMR 記錄
- 使用 Troposphere 產生包含 AWS Config 受管規則的 AWS CloudFormation 範本
- 當 AWS KMS 金鑰的金鑰狀態變更時，取得 Amazon SNS 通知
- 協助強制執行 DynamoDB 標記
- 當 Amazon Data Firehose 資源未使用 AWS KMS 金鑰加密時，識別和提醒
- 在 AWS Blu Age 現代化大型主機應用程式中實作 Microsoft Entra ID 型身分驗證
- 使用 Auth0 和 實作 Amazon WorkSpaces 的 SAML 2.Auth0驗證 AWS Managed Microsoft AD
- 從 SQL Server 遷移至 PostgreSQL 時，實作 PII 資料的 SHA1 雜湊
- 使用 AWS CDK 跨多個 AWS 區域、帳戶和 OUs 啟用 Amazon DevOps Guru，以改善營運效能
- 將 EC2 Windows 執行個體擷取並遷移至 AWS Managed Services 帳戶
- 使用 AWS DMS，以 SSL 模式將 Amazon RDS for Oracle 遷移至 Amazon RDS for PostgreSQL
- 將 ELK 堆疊遷移至 AWS 上的彈性雲端
- 在 上將 F5 BIG-IP 工作負載遷移至 F5 BIG-IP VE AWS 雲端
- 在沒有加密的情況下監控 Amazon Aurora 是否有執行個體
- 透過部署角色販賣機解決方案來佈建最低權限的 IAM 角色
- 使用信任的內容來保護和簡化 AWS 上 Db2 聯合資料庫中的使用者存取
- 使用 AWS Firewall Manager 和 Amazon Data Firehose 將 AWS WAF 日誌傳送至 Splunk
- 使用 Amazon CloudFront 在 Amazon S3 儲存貯體中透過 VPC 提供靜態內容
- 使用 cert-manager 和 Let's Encrypt 為 Amazon EKS 上的應用程式設定end-to-end加密
- 在 IAM 政策中使用使用者 IDs 進行存取控制和自動化
- 確認 ELB 負載平衡器需要終止 TLS
- 使用 Splunk 檢視 AWS Network Firewall 日誌和指標
- 使用 Amazon Quick Sight 視覺化所有 AWS 帳戶的 IAM 登入資料報告

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。