



開發人員指南

AWS Key Management Service



AWS Key Management Service: 開發人員指南

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

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

Table of Contents

AWS Key Management Service	1
為什麼要使用 AWS KMS ?	1
AWS KMS 在 中 AWS 區域	2
AWS KMS 定價	2
AWS KMS 服務層級協議	2
存取 AWS Key Management Service	3
AWS 管理主控台	3
使用 AWS KMS 主控台所需的許可	3
AWS Command Line Interface	4
AWS KMS REST API	4
AWS SDKs	4
使用 AWS SDKs	4
AWS Encryption SDK	5
AWS KMS 最終一致性	6
混合式後量子 TLS	6
關於後量子 TLS	7
使用方式	8
設定混合式後量子 TLS	8
進一步了解	10
AWS KMS 透過 VPC 端點連線至	10
建立的 VPC 端點 AWS KMS	11
連線至 VPC 端點	12
使用 VPC 端點控制對 AWS KMS 資源的存取	13
記錄使用 VPC 端點的 AWS KMS 請求	16
雙堆疊端點	17
無法透過 IPv6 提供的功能	18
概念	19
簡介	19
設計目標	20
AWS KMS keys	21
客戶自管金鑰	23
AWS 受管金鑰	24
AWS 擁有的金鑰	24
AWS KMS key 階層	25

金鑰識別符 (KeyID)	26
非對稱金鑰。	29
HMAC 金鑰	31
ML-DSA 金鑰	32
多區域金鑰	33
匯入的金鑰資料	41
CloudHSM 金鑰存放區中的 KMS 金鑰	47
外部金鑰存放區中的 KMS 金鑰	49
AWS KMS 密碼編譯基本概念	51
熵和隨機數字產生	51
對稱金鑰操作 (僅限加密)	52
非對稱金鑰操作 (加密、數位簽章和簽章驗證)	52
金鑰衍生函數	52
AWS KMS 內部使用數位簽章	53
封套加密	53
密碼編譯操作	54
支援的演算法	56
密碼編譯演算法	56
KMS 金鑰存取和許可	59
KMS 金鑰政策	59
KMS 金鑰授權	60
金鑰政策	60
建立金鑰政策	61
預設金鑰政策	67
檢視金鑰政策	80
變更金鑰政策	83
AWS 服務的許可	85
IAM 政策	86
允許多個 IAM 主體存取 KMS 金鑰	88
IAM 政策的最佳實務	88
在 IAM 政策陳述式中指定 KMS 金鑰	90
範例	93
資源控制政策	99
授權	101
授予概念	103
最佳實務	106

控制對授予的存取	107
建立授予	108
檢視授予	116
使用授予字符	117
淘汰和撤銷授予	118
條件索引鍵	119
AWS 全域條件索引鍵	119
AWS KMS 條件索引鍵	122
AWS KMS 已驗證平台的條件索引鍵	185
最低權限許可	190
實作最低權限的許可	191
屬性型存取控制 (ABAC)	194
的 ABAC 條件金鑰 AWS KMS	195
標籤或別名？	197
對適用於 的 ABAC 進行故障診斷 AWS KMS	198
角色型存取控制 (RBAC)	202
跨帳戶存取權	204
步驟 1：在本機帳戶中新增金鑰政策陳述式	206
步驟 2：在外部帳戶中新增 IAM 政策	209
允許使用外部 KMS 金鑰搭配 AWS 服務	210
在其他帳戶中使用 KMS 金鑰	210
控制對多區域金鑰的存取	211
多區域金鑰的授權基礎知識	212
授權多區域主要管理員和使用者	213
判斷存取權	217
檢查金鑰政策	217
檢查 IAM 政策	220
檢查授與	222
加密內容	223
加密內容規則	224
政策中的加密內容	224
授與中的加密內容	225
記錄加密內容	226
儲存加密內容	226
測試您的許可	226
什麼是 DryRun？	227

利用 API 指定 DryRun	228
故障診斷 AWS KMS 許可	228
範例 1：拒絕使用者存取其 中的 KMS 金鑰 AWS 帳戶	230
範例 2：使用者擔任的角色具有在不同 中使用 KMS 金鑰的許可 AWS 帳戶	231
詞彙表	235
身分驗證	235
Authorization	235
使用身分驗證	235
使用政策管理存取權	236
AWS KMS 資源	237
建立 KMS 金鑰	239
建立 KMS 金鑰的許可	241
選擇要建立的 KMS 金鑰類型	241
建立對稱加密 KMS 金鑰	243
建立非對稱 KMS 金鑰	248
建立 HMAC KMS 金鑰	253
建立多區域主索引鍵	257
建立多區域複本金鑰	262
步驟 1：選擇複本區域	262
步驟 2：建立複本金鑰	263
使用匯入的金鑰材料建立 KMS 金鑰	268
匯入金鑰材料的許可	268
匯入金鑰資料需求	270
步驟 1：建立 AWS KMS key 不含金鑰材料的	272
步驟 2：下載包裝公有金鑰及匯入字符	275
步驟 3：加密金鑰材料	283
步驟 4：匯入金鑰材料	291
在 金鑰存放區中建立 KMS AWS CloudHSM 金鑰	297
在 CloudHSM 金鑰存放區中建立新的 KMS 金鑰	298
在外部金鑰存放區中建立 KMS 金鑰	304
外部金鑰存放區中 KMS 金鑰的要求	305
在外部金鑰存放區中建立新的 KMS 金鑰	306
識別和檢視金鑰	313
尋找金鑰 ID 和金鑰 ARN	313
存取和列出 KMS 金鑰詳細資訊	315
識別不同的金鑰類型	323

識別非對稱 KMS 金鑰	324
識別 HMAC KMS 金鑰	325
識別多區域 KMS 金鑰	325
使用匯入的金鑰材料識別 KMS 金鑰	326
識別金鑰存放區中的 KMS AWS CloudHSM 金鑰	326
識別外部金鑰存放區中的 KMS 金鑰	327
自訂您的主控台檢視	328
排序和篩選您的 KMS 金鑰	328
自訂您的 KMS 金鑰資料表	331
在 金鑰存放區中尋找 KMS 金鑰和 AWS CloudHSM 金鑰材料	332
在 金鑰存放區中尋找 KMS AWS CloudHSM 金鑰	333
尋找 金鑰存放區的所有 AWS CloudHSM 金鑰	334
尋找 金鑰的 KMS AWS CloudHSM 金鑰	336
尋找 KMS AWS CloudHSM 金鑰的金鑰	341
啟用和停用金鑰	344
輪換金鑰	346
為什麼要輪換 KMS 金鑰？	348
金鑰輪換的運作方式	348
啟用自動金鑰輪換	352
停用自動金鑰輪換	354
執行隨需金鑰輪換	356
啟動隨需金鑰輪換（主控台）	356
啟動隨需金鑰輪換 (AWS KMS API)	357
列出輪換和金鑰資料	358
列出輪換和金鑰材料（主控台）	359
列出輪換和金鑰材料 (AWS KMS API)	359
手動輪換金鑰	360
變更一組多區域金鑰中的主金鑰	363
更新主要區域	364
刪除金鑰	368
關於等待期	369
特殊考量	369
控制金鑰刪除的存取權	372
允許金鑰管理員排程和取消金鑰刪除	372
排程金鑰刪除	374
取消刪除金鑰	376

建立警示	377
判斷 KMS 金鑰的過去用量	378
檢查 KMS 金鑰許可，以判斷潛在用量的範圍	379
檢查 AWS CloudTrail 日誌以判斷實際用量	379
刪除匯入的金鑰材料	382
產生資料金鑰	385
建立資料金鑰	385
使用資料金鑰進行密碼編譯操作的運作方式	386
使用資料金鑰來加密資料	386
使用資料金鑰來解密資料	387
無法使用的 KMS 金鑰如何影響資料金鑰	388
產生資料金鑰對	390
建立資料金鑰對	390
使用資料金鑰對進行密碼編譯操作的運作方式	391
使用資料金鑰對加密資料	392
使用資料金鑰對解密資料	392
使用資料金鑰對簽署訊息	393
使用資料金鑰對驗證簽章	394
衍生具有資料金鑰對的共用秘密	395
使用公有金鑰執行離線操作	396
下載公開金鑰的特殊考量	396
下載公有金鑰	397
離線操作範例	399
離線衍生共用秘密	399
使用 ML-DSA 金鑰對進行離線驗證	401
使用 SM2 金鑰對進行離線驗證 (僅限中國區域)	403
監控金鑰	408
監控工具	409
自動化工具	409
手動工具	409
使用 記錄 AWS CloudTrail	410
在 CloudTrail 中尋找 AWS KMS 日誌項目	411
從線索排除 AWS KMS 事件	412
AWS KMS 日誌項目的範例	413
使用 CloudWatch 監控金鑰	496
AWS KMS 指標和維度	496

為匯入金鑰材料過期建立 CloudWatch 警示	504
為外部金鑰存放區建立 CloudWatch 警示	505
使用 Amazon EventBridge 監控金鑰	509
KMS CMK Rotation (KMS CMK 輪換)	509
KMS Imported Key Material Expiration (KMS 匯入的金鑰資料過期)	510
KMS CMK Deletion (KMS CMK 的刪除)	511
別名	512
別名的運作方式	512
控制對別名的存取	515
kms:CreateAlias	515
kms:ListAliases	516
kms:UpdateAlias	517
kms>DeleteAlias	518
限制別名許可	519
建立別名	521
尋找別名名稱和別名 ARN	523
更新別名	527
刪除別名	529
使用別名來控制對 KMS 金鑰的存取	530
kms:RequestAlias	531
kms:ResourceAliases	532
了解如何在應用程式中使用別名	533
在 AWS CloudTrail 日誌中尋找別名	535
Tags (標籤)	537
控制對標籤的存取	538
標記政策中的許可	539
限制標籤許可	541
新增標籤	542
建立 KMS 金鑰時新增標籤	542
將標籤新增至現有的 KMS 金鑰	544
編輯標籤	545
移除標籤	546
檢視標籤	547
使用標籤來控制對 KMS 金鑰的存取	548
金鑰存放區	552
AWS KMS 標準金鑰存放區	552

AWS KMS 具有匯入金鑰材料的標準金鑰存放區	552
AWS KMS 自訂金鑰存放區	553
AWS CloudHSM 金鑰存放區	554
外部金鑰存放區	554
AWS CloudHSM 金鑰存放區	554
AWS CloudHSM 金鑰存放區概念	557
控制對 AWS CloudHSM 金鑰存放區的存取	559
建立 AWS CloudHSM 金鑰存放區	560
檢視 AWS CloudHSM 金鑰存放區	566
編輯 AWS CloudHSM 金鑰存放區設定	569
連接 AWS CloudHSM 金鑰存放區	572
中斷連接 AWS CloudHSM 金鑰存放區	576
刪除 AWS CloudHSM 金鑰存放區	578
對自訂金鑰存放區進行故障診斷	581
外部金鑰存放區	594
外部金鑰存放區概念	597
外部金鑰存放區的運作方式	605
控制對外部金鑰存放區的存取	606
選擇代理連線選項	611
建立外部金鑰存放區	622
編輯外部金鑰存放區屬性	633
檢視外部金鑰存放區	639
監控外部金鑰存放區	644
連接和中斷連接外部金鑰存放區	654
刪除外部金鑰存放區	662
外部金鑰存放區故障診斷	664
安全	685
資料保護	685
保護金鑰資料	686
資料加密	687
網際網路隱私權	688
身分與存取管理	689
AWS 受管政策	689
服務連結角色	692
日誌記錄和監控	697
法規遵循驗證	698

合規和安全文件	698
進一步了解	699
恢復能力	699
區域隔離	700
多租用戶設計	700
中的彈性最佳實務 AWS KMS	700
基礎設施安全性	701
實體主機的隔離	702
配額	703
資源配額	703
AWS KMS keys : 100,000 個	704
每個 KMS 金鑰的別名 : 50	704
每個 KMS 金鑰的授予 : 50,000	704
自訂金鑰存放區資源配額 : 10	705
隨需輪換 : 25	705
請求配額	705
請求每個 AWS KMS API 操作的配額	706
套用請求配額	713
密碼編譯操作的共用配額	714
代表您進行的 API 請求	715
跨帳戶請求	715
自訂金鑰存放區請求配額	715
調節請求	716
程式碼範例	718
基本概念	719
您好 AWS KMS	720
了解基本概念	724
動作	797
案例	961
使用資料表加密	962
中的密碼編譯認證支援 AWS KMS	964
如何對 進行經驗證的呼叫 AWS KMS	965
監控已驗證的請求	966
對 Nitro Enclaves 的監空請求	966
監控 NitroTPM 的請求	971
加密 AWS 服務	976

Amazon Elastic Block Store (Amazon EBS)	976
Amazon EBS 加密	977
使用 KMS 金鑰和資料金鑰	977
Amazon EBS 加密內容	978
偵測 Amazon EBS 失敗	978
使用 AWS CloudFormation 建立加密的 Amazon EBS 磁碟區	979
Amazon EMR	979
加密 EMR 檔案系統 (EMRFS) 上的資料	980
加密叢集節點之儲存磁碟區上的資料	982
加密內容	983
Amazon Redshift	983
Amazon Redshift 加密	984
加密內容	984
參考資料	986
金鑰狀態參考	986
金鑰狀態和 KMS 金鑰類型	987
金鑰狀態資料表	988
金鑰類型參考	994
金鑰類型資料表	995
特殊功能資料表	1001
金鑰規格參考	1008
SYMMETRIC_DEFAULT 金鑰規格	1010
RSA 金鑰規格	1011
橢圓曲線金鑰規格	1014
HMAC KMS 金鑰的金鑰規格	1016
ML-DSA 金鑰規格	1017
SM2 金鑰規格 (僅限中國區域)	1017
許可參考	1018
資料欄描述	1062
AWS KMS 內部操作	1064
網域和網域狀態	1064
內部通訊安全	1068
多區域金鑰的複寫程序	1070
耐久性保護	1071
文件歷史紀錄	1072
最近更新	1072

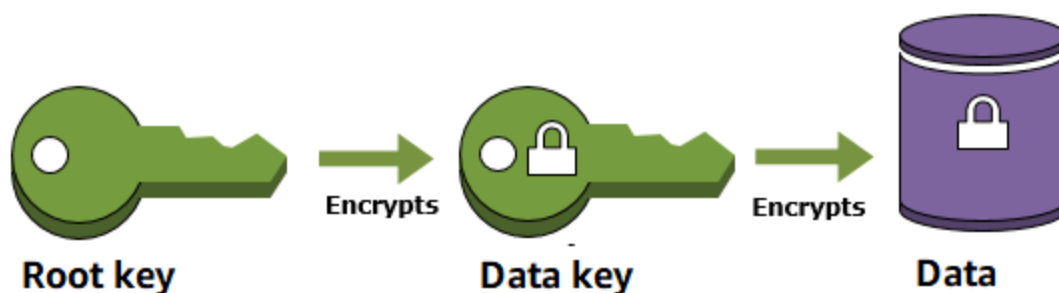
舊版更新	1077
.....	mlxxx

AWS Key Management Service

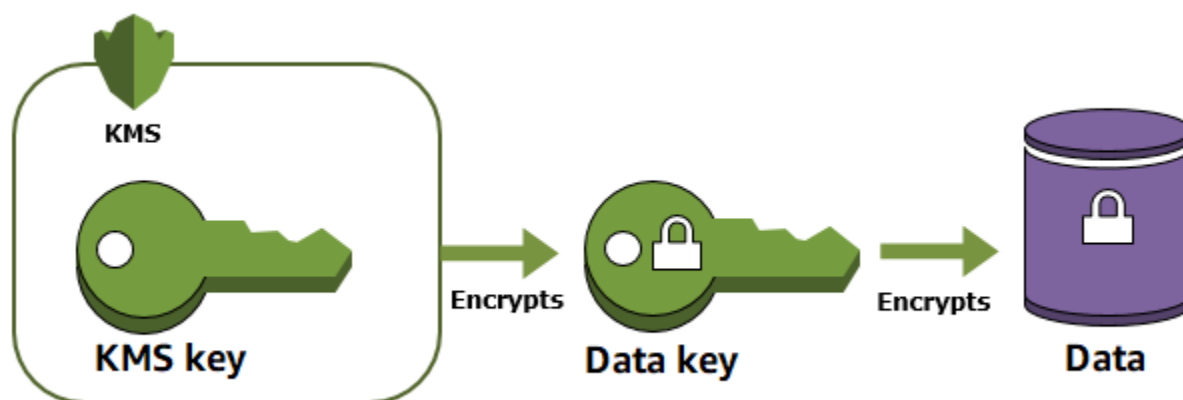
AWS Key Management Service (AWS KMS) 是一種 AWS 受管服務，可讓您輕鬆地建立和控制用於加密和簽署資料的金鑰。您在 中建立 AWS KMS keys 的 AWS KMS 受到 [FIPS 140-3 安全層級 3 驗證的硬體安全模組 \(HSM\)](#) 保護。它們永遠不會保持 AWS KMS 未加密狀態。若要使用或管理您的 KMS 金鑰，您可以與 互動 AWS KMS。

為什麼要使用 AWS KMS ？

當您加密資料時，您需要保護您的加密金鑰。如果您加密金鑰，則需要保護其加密金鑰。最後，您必須保護階層中保護資料的最高層級加密金鑰（稱為根金鑰）。這就是 AWS KMS。



AWS KMS 保護您的根金鑰。KMS 金鑰完全在其中建立、管理、使用和刪除 AWS KMS。他們永遠不會讓服務處於未加密狀態。若要使用或管理您的 KMS 金鑰，請呼叫 AWS KMS。



此外，您可以在 中建立和管理[金鑰政策](#) AWS KMS，確保只有信任的使用者才能存取 KMS 金鑰。

AWS KMS 在 中 AWS 區域

AWS KMS 支援 AWS 區域 的 會列在[AWS Key Management Service 端點和配額](#)中。如果 支援的 中 不支援 AWS KMS 功能 AWS 區域 AWS KMS ，則會在功能的相關主題中說明區域差異。

AWS KMS 定價

如同其他 AWS 產品，使用 AWS KMS 不需要合約或最低購買金額。如需 AWS KMS 定價的詳細資訊，請參閱 [AWS Key Management Service 定價](#)。

AWS KMS 服務層級協議

AWS Key Management Service 由定義我們服務可用性政策的服務[層級協議](#)提供支援。

存取 AWS Key Management Service

您可以透過 AWS KMS 下列方式使用：

主題

- [AWS 管理主控台](#)
- [AWS Command Line Interface](#)
- [AWS KMS REST API](#)
- [AWS SDKs](#)
- [搭配 AWS SDK 使用此服務](#)
- [AWS Encryption SDK](#)
- [AWS KMS 最終一致性](#)
- [搭配 使用混合式後量子 TLS AWS KMS](#)
- [AWS KMS 透過 VPC 端點連線至](#)
- [雙堆疊端點支援](#)

AWS 管理主控台

主控台是用於管理 AWS KMS 和資源的 Web 型使用者介面 AWS。如果您已註冊 AWS 帳戶，您可以登入 AWS 管理主控台 並從 AWS KMS AWS 管理主控台 首頁選擇 來存取 AWS KMS 主控台。

使用 AWS KMS 主控台所需的許可

若要使用 AWS KMS 主控台，使用者必須擁有一組最低許可，以允許他們使用其中 AWS KMS 的資源 AWS 帳戶。除了這些 AWS KMS 許可，使用者還必須具有列出 IAM 使用者和 IAM 角色清單的許可。如果您建立比最基本必要許可更嚴格的 IAM 政策，則對於採取該 IAM 政策的使用者而言，AWS KMS 主控台就無法如預期運作。

對於允許使用者唯讀存取 AWS KMS 主控台所需的最低許可，請參閱[允許使用者在 AWS KMS 主控台中檢視 KMS 金鑰](#)。

若要允許使用者使用 AWS KMS 主控台來建立和管理 KMS 金鑰，請將 AWSKeyManagementServicePowerUser 受管政策連接至使用者，如中所述[AWS 的 受管政策 AWS Key Management Service](#)。

對於透過 [AWS SDKsAWS Command Line Interface](#)、或 使用 AWS KMS API 的使用者，您不需要允許最低主控台許可[AWS Tools for PowerShell](#)。但是，您需要授予這些使用者使用 API 的許可。如需詳細資訊，請參閱[許可參考](#)。

AWS Command Line Interface

您可以使用這些 AWS CLI 工具在系統的命令列發出命令或建置指令碼，以執行 AWS（包括 AWS KMS）任務。

如需 AWS KMS 透過 使用 的詳細資訊 AWS CLI，請參閱 [AWS CLI 命令參考](#)

AWS KMS REST API

的架構設計 AWS KMS 為程式設計語言中立。REST API 是 的 HTTP 界面 AWS KMS。透過 REST API，您可以使用標準 HTTP 請求來建立、擷取和刪除金鑰。

如需使用 的詳細資訊 AWS KMS REST API，請參閱 [AWS Key Management Service API 參考](#)

AWS SDKs

AWS 提供 SDKs（軟體開發套件），其中包含適用於常見程式設計語言和平台（Java、JavaScript、C、Python 等）的程式庫和範本程式碼。AWS SDKs 提供便捷的方式來建立對 AWS KMS 和 的程式設計存取 AWS。AWS KMS 是一項 REST 服務。您可以使用 AWS SDK AWS KMS 程式庫將請求傳送至，該程式庫會包裝基礎 AWS KMS REST API 並簡化您的程式設計任務。如需有關的資訊 AWS SDKs，包括如何下載和安裝它們，請參閱[在上建置的工具 AWS](#)。

[AWS KMS 使用 AWS SDKs 程式碼範例](#) 提供 AWS KMS 透過 使用 的良好起點 AWS SDKs。

搭配 AWS SDK 使用此服務

AWS 軟體開發套件 (SDKs) 適用於許多熱門的程式設計語言。每個 SDK 都提供 API、程式碼範例和說明文件，讓開發人員能夠更輕鬆地以偏好的語言建置應用程式。

SDK 文件	代碼範例
適用於 C++ 的 AWS SDK	適用於 C++ 的 AWS SDK 程式碼範例
AWS CLI	AWS CLI 程式碼範例

SDK 文件	代碼範例
適用於 Go 的 AWS SDK	適用於 Go 的 AWS SDK 程式碼範例
適用於 Java 的 AWS SDK	適用於 Java 的 AWS SDK 程式碼範例
適用於 JavaScript 的 AWS SDK	適用於 JavaScript 的 AWS SDK 程式碼範例
適用於 Kotlin 的 AWS SDK	適用於 Kotlin 的 AWS SDK 程式碼範例
適用於 .NET 的 AWS SDK	適用於 .NET 的 AWS SDK 程式碼範例
適用於 PHP 的 AWS SDK	適用於 PHP 的 AWS SDK 程式碼範例
AWS Tools for PowerShell	AWS Tools for PowerShell 程式碼範例
適用於 Python (Boto3) 的 AWS SDK	適用於 Python (Boto3) 的 AWS SDK 程式碼範例
適用於 Ruby 的 AWS SDK	適用於 Ruby 的 AWS SDK 程式碼範例
適用於 Rust 的 AWS SDK	適用於 Rust 的 AWS SDK 程式碼範例
適用於 SAP ABAP 的 AWS SDK	適用於 SAP ABAP 的 AWS SDK 程式碼範例
適用於 Swift 的 AWS SDK	適用於 Swift 的 AWS SDK 程式碼範例

可用性範例

找不到所需的內容嗎？請使用本頁面底部的提供意見回饋連結申請程式碼範例。

AWS Encryption SDK

AWS Encryption SDK 是在應用程式中實作用戶端加密的工具。它不提供 KMS 的完整存取權，而是與整合 AWS KMS，或可在不參考 KMS 金鑰的情況下用作獨立 SDK。程式庫適用於 Java、JavaScript、C、Python 和其他程式設計語言。

如需詳細資訊，請參閱《AWS Encryption SDK 開發人員指南》<https://docs.aws.amazon.com/encryption-sdk/latest/developer-guide/introduction.html>。

AWS KMS key 政策和 IAM 政策

AWS KMS 最終一致性

由於系統的分散式性質，AWS KMS API 遵循[最終一致性](#)模型。因此，您執行的後續命令可能不會立即看到 AWS KMS 資源的變更。

當您執行 AWS KMS API 呼叫時，在變更全面可用之前，可能會有短暫的延遲 AWS KMS。變更傳播到整個系統通常需要不到幾秒鐘的時間，但在某些情況下可能需要幾分鐘。在此期間，您可能會收到非預期的錯誤，例如 `NotFoundException` 或 `InvalidStateException`。例如，`NotFoundException` 如果您 `GetParametersForImport` 在呼叫後立即呼叫，AWS KMS 可能會傳回 `CreateKey`。

建議您在 AWS KMS 用戶端上設定重試策略，以便在短暫的等待期間之後自動重試操作。如需詳細資訊，請參閱與叫在 AWS SDK 與工具參考指南的[重試操作](#)。

對於授權相關的 API 呼叫，您可以[使用授權權杖](#)來避免任何潛在的延遲，並立即在授權中使用權限。如需詳細資訊，請參閱[最終一致性模式 \(授權\)](#)。

搭配使用混合式後量子 TLS AWS KMS

AWS Key Management Service (AWS KMS) 支援 Transport Layer Security (TLS) 網路加密通訊協定的混合式後量子金鑰交換選項。連線至 AWS KMS API 端點時，您可使用此 TLS 選項。這些選用的混合式後量子金鑰交換功能至少與現今使用的 TLS 加密功能同樣安全，且還能提供其他長期安全優勢。不過，與現今使用的傳統金鑰交換通訊協定相較之下，這些功能會影響延遲和輸送量。

您傳送至 AWS Key Management Service (AWS KMS) 的資料受到 Transport Layer Security (TLS) 連線提供的加密保護。AWS KMS 支援可用於 TLS 工作階段的傳統密碼套件會使得暴力破解攻擊在現今的科技下變得毫無用武之地。不過如果大規模量子運算在未來變得可行，那麼用於 TLS 金鑰交換機制的傳統密碼套件將容易遭受這些攻擊影響。如果您開發的應用程式依賴於透過 TLS 連線傳遞的資料的長期機密性，您應該考慮在大規模量子電腦可供使用之前遷移到量子後密碼編譯的計劃。AWS 正在為未來做好準備，我們也希望您做好準備。

為了保護今天加密的資料免受潛在的未來攻擊，AWS 正與密碼編譯社群參與開發量子抗性或後量子演算法。我們已在 中實作混合式後量子金鑰交換密碼套件 AWS KMS，結合傳統和後量子元素，以確保您的 TLS 連線至少與傳統密碼套件一樣強大。

這些混合密碼套件可在大多數 [AWS 區域](#) 的生產工作負載上使用。不過，由於混合式密碼套件的效能特性和頻寬需求與傳統金鑰交換機制不同，我們建議您在不同條件下[在 AWS KMS API 呼叫上測試它們](#)。

意見回饋

我們總是歡迎您提供意見回饋，並參加我們的開放原始碼儲存庫。我們特別想要了解您的基礎設施如何與此新版的 TLS 流量互動。

- 若要提供有關此主題的意見回饋，請使用本頁右上角的意見回饋連結。
- 我們正在 GitHub 上的 [s2n-tls](#) 儲存庫中以開放原始碼開發這些混合式密碼套件。若要提供密碼套件可用性的意見回饋，或分享新的測試條件或結果，請在 s2n-tls 儲存庫中[建立問題](#)。
- 我們正在撰寫程式碼範例，以在 [aws-kms-pq-tls-example](#) GitHub 儲存庫 AWS KMS 中使用混合式後量子 TLS 搭配。若要詢問有關設定 HTTP 用戶端或 AWS KMS 用戶端以使用混合密碼套件的問題或分享想法，請在儲存aws-kms-pq-tls-example庫中[建立問題](#)。

支援的 AWS 區域

適用於 的後量子 TLS AWS KMS 可在所有 AWS 區域 AWS KMS 支援的 中使用。

如需每個 AWS KMS 端點的清單 AWS 區域，請參閱 中的[AWS Key Management Service 端點和配額](#)Amazon Web Services 一般參考。如需有關 FIPS 端點的詳細資訊，請參閱《Amazon Web Services 一般參考》的 [FIPS 端點](#)。

關於 TLS 中的混合式後量子金鑰交換

AWS KMS 支援混合式後量子金鑰交換密碼套件。您可以使用 Linux 系統上的 AWS SDK for Java 2.x 和 AWS Common Runtime 來設定使用這些密碼套件的 HTTP 用戶端。然後，每當您使用 HTTP 用戶端連線到 AWS KMS 端點時，都會使用混合式密碼套件。

此 HTTP 客戶端使用 [s2n-tls](#)，這是 TLS 協定的開源實作。s2n-tls 使用的混合式密碼套件僅針對金鑰交換實作，而非針對直接資料加密。在金鑰交換時，用戶端與伺服器會計算其用於加密與解密線路上資料的金鑰。

s2n-tls 使用的演算法是一種混合式演算法，結合了 [Elliptic Curve Diffie-Hellman](#) (ECDH)，這是目前在 TLS 中使用的傳統金鑰交換演算法，與 [Module-Lattice-Based金鑰封裝機制](#) (ML-KEM)、公有金鑰加密和金鑰建立演算法，美國國家標準技術研究所 (NIST) [已指定為第一個標準](#)量子後金鑰協議演算法。此混合會獨立使用各演算法，以產生金鑰。然後以密碼編譯方式結合兩個金鑰。使用 s2n-tls，您可以將 [HTTP 用戶端設定為](#)偏好後量子 TLS，將具有 ML-KEM 的 ECDH 置於偏好清單的首位。為了確保相容性，傳統金鑰交換演算法仍包含在偏好設定清單中，但偏好設定順序較低。

搭配 使用混合式後量子 TLS AWS KMS

您可以使用混合式後量子 TLS 來進行呼叫 AWS KMS。設定 HTTP 用戶端測試環境時，請注意下列資訊：

傳輸中加密

s2n-tls 中的混合式密碼套件僅用於傳輸中加密。它們會在資料從用戶端傳輸到 AWS KMS 端點時保護您的資料。AWS KMS 不會使用這些密碼套件來加密 下的資料 AWS KMS keys。

相反地，當在 KMS 金鑰下 AWS KMS 加密您的資料時，它會使用對稱密碼編譯搭配 256 位元金鑰和 Galois 計數器模式 (AES-GCM) 演算法中的進階加密標準，該演算法已具有量子防護能力。在理論上，未來針對以 256 位元 AES-GCM 金鑰建立的加密文字所做的大規模量子運算攻擊，會將金鑰的有效安全性降低到 128 位元。此安全層級足以對 AWS KMS 加密文字進行暴力破解攻擊。

支援的系統

目前僅支援在 Linux 系統上使用 s2n-tls 中的混合式密碼套件。此外，這些加密套件僅在支援 AWS 常用執行時間 (如 AWS SDK for Java 2.x) 的軟體開發套件中獲得支援。如需範例，請參閱 [設定混合式後量子 TLS](#)。

AWS KMS 端點

AWS KMS 在所有端點上支援混合式後量子 TLS，包括 [FIPS 140-3 驗證的端點](#)。

設定混合式後量子 TLS

在此程序中，為 AWS 通用執行期 HTTP 用戶端新增 Maven 相依性。接下來，設定偏好後量子 TLS 的 HTTP 用戶端。然後，建立使用 HTTP AWS KMS 用戶端的用戶端。

若要查看設定與搭配 AWS KMS 使用混合式後量子 TLS 的完整可用範例，請參閱 [aws-kms-pq-tls-example](#) 儲存庫。

Note

已作為預覽版提供的 AWS 通用執行期 HTTP 用戶端，已於 2023 年 2 月正式推出。在該版本中，`tlsCipherPreference` 類別和 `tlsCipherPreference()` 方法參數由 `postQuantumTlsEnabled()` 方法參數取代。如果您在預覽期間使用此範例，則需要更新程式碼。

1. 將 AWS Common Runtime 用戶端新增至 Maven 相依性。我們建議使用最新的可用版本。

例如，此陳述式會將 2.30.22 版本的 AWS Common Runtime 用戶端新增至 Maven 相依性。

```
<dependency>
  <groupId>software.amazon.awssdk</groupId>
  <artifactId>aws-crt-client</artifactId>
  <version>2.30.22</version>
</dependency>
```

2. 若要啟用混合式後量子密碼套件，請將 AWS SDK for Java 2.x 新增至您的專案並初始化。然後如下列範例所示在 HTTP 用戶端啟用混合式後量子密碼套件。

此程式碼使用 `postQuantumTlsEnabled()` 方法參數來設定 [AWS 常用的執行時間 HTTP 用戶端](#)，其偏好建議的混合式後量子密碼套件 ECDH 搭配 ML-KEM。然後，它會使用設定的 HTTP 用戶端來建置 AWS KMS 非同步用戶端的執行個體 [KmsAsyncClient](#)。完成此程式碼之後，`KmsAsyncClient` 執行個體上的所有 [AWS KMS API](#) 請求都會使用混合式後量子 TLS。

```
// Configure HTTP client
SdkAsyncHttpClient awsCrtHttpClient = AwsCrtAsyncHttpClient.builder()
    .postQuantumTlsEnabled(true)
    .build();

// Create the AWS KMS async client
KmsAsyncClient kmsAsync = KmsAsyncClient.builder()
    .httpClient(awsCrtHttpClient)
    .build();
```

3. 使用混合式後量子 TLS 測試您的 AWS KMS 呼叫。

當您在設定的 AWS KMS 用戶端上呼叫 AWS KMS API 操作時，您的呼叫會使用混合式後量子 TLS 傳輸至 AWS KMS 端點。若要測試您的組態，請呼叫 AWS KMS API，例如 [ListKeys](#)。

```
ListKeysResponse keys = kmsAsync.listKeys().get();
```

測試您的混合式後量子 TLS 組態

考慮在您的 呼叫的應用程式上使用混合密碼套件執行下列測試 AWS KMS。

- 檢視您應用程式進行之 AWS KMS API 呼叫的 CloudTrail 日誌項目中的 `tlsDetails` 區段。keyExchange 欄位應提及混合演算法，例如 X25519MLKEM768。如需範例，請參閱 [透過後量子 TLS 連線使用標準對稱加密金鑰進行解密](#)。
- 使用混合式後量子 TLS 執行基準測試。混合金鑰交換會增加 TLS 交握中某些訊息的大小和處理時間，但在大多數情況下，整體效能影響應該是不可接受的。
- 請嘗試從不同位置進行連線。根據請求採用的網路路徑，您可能會發現具有深度封包檢查 (DPI) 的舊版中繼主機、代理或防火牆會封鎖請求。這可能是因為在 TLS 交握的 `ClientHello` 部分中使用新的金鑰交換群組，或來自較大的金鑰交換訊息。如果您無法解決這些問題，請與您的安全團隊或 IT 管理員合作，更新相關組態並解除封鎖新的 TLS 金鑰交換群組。

進一步了解 中的後量子 TLS AWS KMS

如需在 中使用混合式後量子 TLS 的詳細資訊 AWS KMS，請參閱下列資源。

- 若要了解 的後量子密碼編譯 AWS，包括部落格文章和研究論文的連結，請參閱[後量子密碼編譯](#)。
- 如需 s2n-tls 的資訊，請參閱 [全新開放原始碼 TLS 實作 s2n-tls 簡介](#) 和使用 [s2n-tls](#)。
- 如需有關 AWS 通用執行期 HTTP 用戶端的資訊，請參閱《AWS SDK for Java 2.x 開發人員指南》中的[設定 AWS CRT 型 HTTP 用戶端](#)。
- 如需國家標準技術研究 (NIST) 的後量子加密法專案的資訊，請參閱[後量子加密法](#)。
- 如需有關 NIST 後量子密碼學標準化的資訊，請參閱[後量子密碼學標準化](#)。

AWS KMS 透過 VPC 端點連線至

您可以透過虛擬私有雲端 (VPC) AWS KMS 中的私有介面端點直接連線至 。當您使用介面 VPC 端點時，VPC 與 之間的通訊 AWS KMS 會完全在 AWS 網路中執行。

AWS KMS 支援採用 技術的 Amazon Virtual Private Cloud (Amazon VPC) 端點[AWS PrivateLink](#)。每個 VPC 端點皆會由一個或多個具私有 IP 地址[彈性網路界面](#) (ENI) 來表示，而該界面位於 VPC 子網路中。

介面 VPC 端點會直接將您的 VPC 連接到 ，AWS KMS 無需網際網路閘道、NAT 裝置、VPN 連接或 AWS Direct Connect 連線。VPC 中的執行個體不需要公有 IP 地址即可與之通訊 AWS KMS。

區域

AWS KMS 在 支援的所有 中支援 VPC 端點和 VPC AWS 區域 [AWS KMS](#) 端點政策。

AWS KMS VPC 端點的考量事項

在您設定介面 VPC 端點之前 AWS KMS，請檢閱《AWS PrivateLink 指南》中的[介面端點屬性和限制](#)主題。

AWS KMS 對 VPC 端點的支援包括下列項目。

- 您可以使用 VPC 端點從 VPC 中呼叫所有 [AWS KMS API 操作](#)。
- 您可以建立連線至 AWS KMS 區域端點或 [AWS KMS FIPS 端點的介面 VPC 端點](#)。
- 您可以使用 AWS CloudTrail 日誌，透過 VPC 端點稽核 KMS 金鑰的使用。如需詳細資訊，請參閱[記錄使用 VPC 端點的 AWS KMS 請求](#)。

主題

- [建立的 VPC 端點 AWS KMS](#)
- [連線至 AWS KMS VPC 端點](#)
- [使用 VPC 端點控制對 AWS KMS 資源的存取](#)
- [記錄使用 VPC 端點的 AWS KMS 請求](#)

建立的 VPC 端點 AWS KMS

您可以使用 Amazon VPC AWS KMS 主控台或 Amazon VPC API 來建立的 VPC 端點。請依照下列程序，使用下列其中一個值[建立介面端點](#)。

- 若要為 建立 VPC 端點 AWS KMS，請使用下列服務名稱：

```
com.amazonaws.region.kms
```

例如，在美國西部 (奧勒岡) 區域 (us-west-2)，服務名稱為：

```
com.amazonaws.us-west-2.kms
```

- 若要建立連接 [AWS KMS FIPS 端點](#) 的 VPC 端點，請採用下列服務名稱：

```
com.amazonaws.region.kms-fips
```

例如，在美國西部 (奧勒岡) 區域 (us-west-2)，服務名稱為：

```
com.amazonaws.us-west-2.kms-fips
```

若要更輕鬆使用 VPC 端點，您可以為 VPC 端點啟用[私有 DNS 名稱](#)。如果您選擇 Enable DNS Name (啟用 DNS 名稱) 選項，標準 AWS KMS DNS 主機名稱會解析為您的 VPC 端點。例如，`https://kms.us-west-2.amazonaws.com` 會解析為連接至服務名稱 `com.amazonaws.us-west-2.kms` 的 VPC 端點。

此選項可讓您更輕鬆使用 VPC 端點。根據預設，AWS SDKs 和 AWS CLI 會使用標準 AWS KMS DNS 主機名稱，因此您不需要在應用程式和命令中指定 VPC 端點 URL。

如需詳細資訊，請參閱《AWS PrivateLink 指南》中的[透過介面端點存取服務](#)。

連線至 AWS KMS VPC 端點

您可以使用 AWS SDK AWS CLI、或 `awscli`，AWS KMS 透過 VPC 端點連線至 AWS Tools for PowerShell。若要指定 VPC 端點，請使用它的 DNS 名稱。

例如，此 `list-keys` 命令會使用 `endpoint-url` 參數來指定 VPC 端點。若要使用如下的命令，請將範例 VPC 端點 ID 換成您帳戶中的 ID。

```
$ aws kms list-keys --endpoint-url https://vpce-1234abcd5678c90a-09p7654s-us-east-1a.ec2.us-east-1.vpce.amazonaws.com
```

所需的許可

若要讓使用 VPC 端點的 AWS KMS 請求成功，委託人需要兩個來源的許可：

- [金鑰政策](#)、[IAM 政策](#)，或[授予](#)必須為委託人授予對資源 (KMS 金鑰或別名) 呼叫操作的許可。
- VPC 端點政策必須授予委託人許可，才能使用端點提出請求。

例如，金鑰政策可能會授予委託人對特定 KMS 金鑰呼叫 [Decrypt](#) 的許可。不過，VPC 端點政策可能不允許該委託人透過使用端點對該 KMS 金鑰呼叫 `Decrypt`。

或者，VPC 端點政策可能允許委託人使用端點呼叫對某些 KMS 金鑰呼叫 [DisableKey](#)。但是，如果委託人沒有來自金鑰政策、IAM 政策或授予的許可，則請求會失敗。

您可以在建立端點時建立 VPC 端點政策，並且可以隨時變更 VPC 端點政策。使用 VPC 管理主控台，或 [CreateVpcEndpoint](#) 或 [ModifyVpcEndpoint](#) 操作。您也可以[使用 AWS CloudFormation 範本](#)建立和變更 VPC 端點政策。如需有關如何使用 VPC 管理主控台的說明，請參閱《AWS PrivateLink 指南》中的[建立介面端點](#)和[修改介面端點](#)。

私有主機名稱

如果您在建立 VPC 端點時啟用私有主機名稱，則不需要在 CLI 命令或應用程式組態中指定 VPC 端點 URL。標準 AWS KMS DNS 主機名稱會解析為您的 VPC 端點。AWS CLI 和 SDKs 預設使用此主機名稱，因此您可以開始使用 VPC 端點連線到 AWS KMS 區域端點，而無需變更指令碼和應用程式中的任何內容。

若要使用私有主機名稱，您 VPC 的 `enableDnsHostnames` 和 `enableDnsSupport` 屬性必須設為 `true`。如需設定這些屬性，請使用 [ModifyVpcAttribute](#) 操作。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的[檢視和更新 VPC 的 DNS 屬性](#)。

使用 VPC 端點控制對 AWS KMS 資源的存取

當請求來自 VPC 或使用 VPC 端點時，您可以控制對 AWS KMS 資源和操作的存取。為此，請使用[金鑰政策](#)或[IAM 政策](#)的下列其中一個[全域條件索引鍵](#)。

- 使用 `aws:sourceVpce` 條件索引鍵，以根據 VPC 端點來授予或限制存取。
- 使用 `aws:sourceVpc` 條件索引鍵，以根據託管私有端點的 VPC 來授予或限制存取。

Note

根據 VPC 端點建立金鑰政策和 IAM 政策時，請務必謹慎執行。如果政策陳述式要求請求來自特定 VPC 或 VPC 端點，則來自代表您使用 AWS KMS 資源之整合 AWS 服務的請求可能會失敗。如需協助，請參閱[在具有 AWS KMS 許可的政策中使用 VPC 端點條件](#)。

此外，當請求來自[Amazon VPC 端點](#)時，`aws:sourceIP` 條件索引鍵無效。若要限制對 VPC 端點的請求，請使用 `aws:sourceVpce` 或 `aws:sourceVpc` 條件金鑰。如需詳細資訊，請參閱《AWS PrivateLink 指南》中的[VPC 端點和 VPC 端點服務的身分與存取管理](#)

您可以使用這些全域條件金鑰來控制對 AWS KMS keys (KMS 金鑰)、別名和操作的存取，例如不依賴任何特定資源的[CreateKey](#)。

例如，以下範例金鑰政策只在請求使用指定 VPC 端點時，才允許使用者執行某些密碼編譯操作。當使用者向提出請求時 AWS KMS，請求中的 VPC 端點 ID 會與政策中的 `aws:sourceVpce` 條件索引鍵值進行比較。如果不相符，則會拒絕請求。

若要使用這類政策，請將預留位置 AWS 帳戶 ID 和 VPC 端點 IDs 取代為帳戶的有效值。

JSON

```
{
  "Id": "example-key-1",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "EnableIAMpolicies",
      "Effect": "Allow",
      "Principal": {"AWS": ["111122223333"]},
      "Action": ["kms:*"],
      "Resource": "*"
    },
    {
      "Sid": "Restrict usage to my VPC endpoint",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:sourceVpc": "vpce-1234abcd5678c90a"
        }
      }
    }
  ]
}
```

您也可以使用 `aws:sourceVpc` 條件索引鍵，以根據 VPC 端點所在的 VPC 來限制對 KMS 金鑰的存取權。

以下範例金鑰政策只在命令來自 `vpc-12345678` 時，才允許這些命令管理 KMS 金鑰。此外，它只在命令來自 `vpc-2b2b2b2b` 時，才允許命令將 KMS 金鑰用於密碼編譯操作。如果應用程式在一個 VPC 中執行，但您使用第二個隔離的 VPC 來執行管理功能，您可能會使用如下的政策。

若要使用這類政策，請將預留位置 AWS 帳戶 ID 和 VPC 端點 IDs 取代為帳戶的有效值。

JSON

```
{
  "Id": "example-key-2",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAdministrativeActionsFromVPC",
      "Effect": "Allow",
      "Principal": {
        "AWS": "111122223333"
      },
      "Action": [
        "kms:Create*",
        "kms:Enable*",
        "kms:Put*",
        "kms:Update*",
        "kms:Revoke*",
        "kms:Disable*",
        "kms>Delete*",
        "kms:TagResource",
        "kms:UntagResource"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:sourceVpc": "vpc-12345678"
        }
      }
    },
    {
      "Sid": "AllowKeyUsageFromVPC2b2b2b2b",
      "Effect": "Allow",
      "Principal": {
        "AWS": "111122223333"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:GenerateDataKey*"
      ],
```



```

        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:sourceVpc": "vpc-2b2b2b2b"
            }
        }
    },
    {
        "Sid": "AllowReadActionsEverywhere",
        "Effect": "Allow",
        "Principal": {
            "AWS": "111122223333"
        },
        "Action": [
            "kms:Describe*",
            "kms:List*",
            "kms:Get*"
        ],
        "Resource": "*"
    }
]
}

```

記錄使用 VPC 端點的 AWS KMS 請求

AWS CloudTrail 會記錄使用 VPC 端點的所有操作。當請求 AWS KMS 使用 VPC 端點時，VPC 端點 ID 會出現在記錄請求的 [AWS CloudTrail 日誌](#) 項目中。您可以使用端點 ID 稽核 AWS KMS VPC 端點的使用。

不過，您的 CloudTrail 日誌不包含其他帳戶中的主體所請求的操作，或對其他帳戶中 KMS 金鑰和別名所請求 AWS KMS 的操作。此外，為了保護您的 VPC，被 VPC 端點政策拒絕，但在其他情況下已允許的請求不會記錄在 [AWS CloudTrail](#)。

例如，此範例日誌項目中會記錄使用 VPC 端點的 [GenerateDataKey](#) 請求。vpcEndpointId 欄位出現在日誌項目結尾。

```

{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",

```

```

    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "accountId": "111122223333",
    "userName": "Alice"
  },
  "eventTime": "2018-01-16T05:46:57Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GenerateDataKey",
  "awsRegion": "eu-west-1",
  "sourceIPAddress": "172.01.01.001",
  "userAgent": "aws-cli/1.14.23 Python/2.7.12 Linux/4.9.75-25.55.amzn1.x86_64
botocore/1.8.27",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "numberOfBytes": 128
  },
  "responseElements": null,
  "requestID": "a9fff0bf-fa80-11e7-a13c-afcabbff2f04c",
  "eventID": "77274901-88bc-4e3f-9bb6-acf1c16f6a7c",
  "readOnly": true,
  "resources": [
    {
      "ARN": "arn:aws:kms:eu-west-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "accountId": "111122223333",
      "type": "AWS::KMS::Key"
    }
  ],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333",
  "vpcEndpointId": "vpce-1234abcd5678c90a"
}

```

雙堆疊端點支援

AWS KMS 提供支援 IPv4 和 IPv6 用戶端的雙堆疊公有端點。雙堆疊端點可讓用戶端 AWS KMS 使用 IPv4 或 IPv6 地址與通訊。如需 AWS KMS 端點的詳細資訊，請參閱[AWS Key Management Service 端點和配額](#)。

的 AWS KMS 雙堆疊公有端點 `https://kms.your-region.api.aws` 支援 IPv4 和 IPv6 用戶端。AWS KMS 也可以使用透過 IPv4 和 IPv6 從虛擬私有雲端 (VPC) 私下存取 AWS PrivateLink。如需為建立私有介面 VPC 端點的詳細資訊 AWS KMS，請參閱[AWS KMS 透過 VPC 端點連線至](#)。

如需 VPCs IPv6 定址的詳細資訊，請參閱 [《Amazon Virtual Private Cloud 使用者指南》中的 Amazon VPC 的運作方式](#)。Amazon Virtual Private Cloud 如需如何將 VPC 設定為雙堆疊模式的詳細資訊，請參閱 [《Amazon Virtual Private Cloud 使用者指南》中的 VPCs 和子網路的 IP 定址](#)。

無法透過 IPv6 提供的功能

AWS KMS 無法透過 IPv6 與 AWS CloudHSM 金鑰存放區或外部金鑰存放區通訊。此限制不會阻止您透過 IPv6 AWS KMS APIs。

AWS KMS 概念

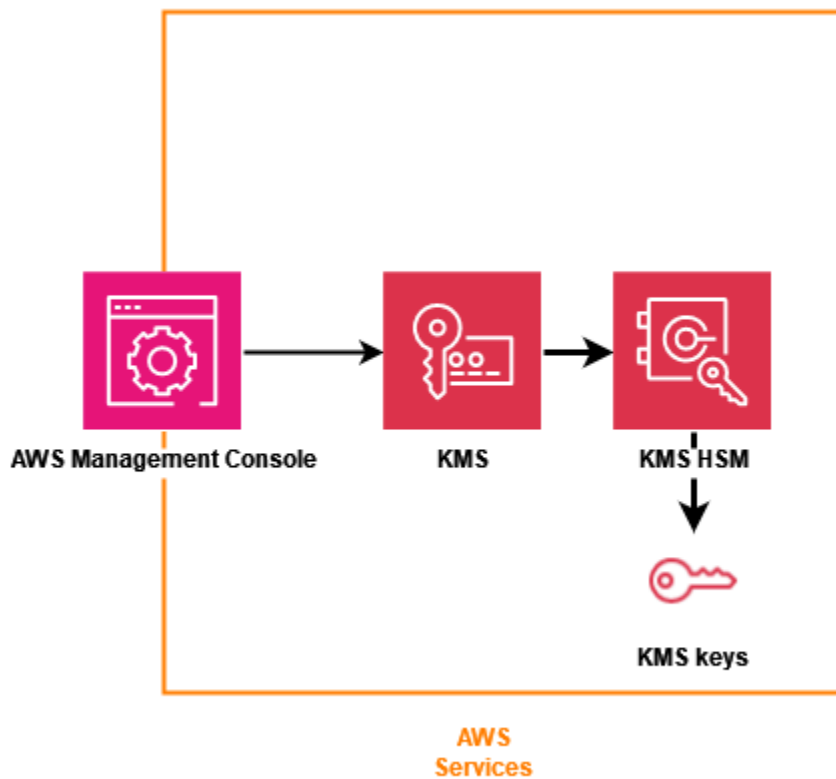
了解 AWS Key Management Service (AWS KMS) 中使用的基本術語和概念，以及它們如何協同運作以協助保護您的資料。

簡介 AWS KMS

AWS Key Management Service (AWS KMS) 提供 Web 界面來產生和管理密碼編譯金鑰，並做為保護資料的密碼編譯服務提供者。AWS KMS 提供與 AWS 服務整合的傳統金鑰管理服務 AWS，透過集中式管理和稽核，提供客戶金鑰的一致檢視。

AWS KMS 包含透過 AWS 管理主控台的 Web 界面、命令列界面和 RESTful API 操作，以請求 FIPS 140-3 驗證的分散式硬體安全模組 (HSMs) 的密碼編譯操作。AWS KMS HSM 是一種多晶片獨立硬體密碼編譯設備，旨在提供專用密碼編譯函數，以滿足的安全性和可擴展性要求 AWS KMS。您可以在作為 AWS KMS keys 管理的金鑰下建立自己的 HSM 式密碼編譯階層。這些金鑰只能在處理密碼編譯請求所需的必要時間內在 HSM 上和記憶體內提供。您可以建立多個 KMS 金鑰，每個金鑰都以其金鑰 ID 表示。只有在每位客戶管理的 AWS IAM 角色和帳戶下，才能建立、刪除或使用客戶受管 KMS 金鑰來加密、解密、簽署或驗證資料。您可以藉由建立連接至金鑰的政策，來定義誰可以管理和/或使用 KMS 金鑰的存取控制項。這些政策可讓您為每個 API 操作定義金鑰的應用程式特定用途。

此外，大多數 AWS 服務都支援使用 KMS 金鑰加密靜態資料。此功能可讓客戶透過控制存取 KMS 金鑰的方式和時間，來控制 AWS 服務存取加密資料的方式和時間。



AWS KMS 是一種分層服務，由面向 Web 的 AWS KMS 主機和 HSMs 層組成。這些分層主機的分組會形成 AWS KMS 堆疊。所有對的請求 AWS KMS 都必須透過 Transport Layer Security 通訊協定 (TLS) 提出，並在 AWS KMS 主機上終止。AWS KMS 主機只允許 TLS 搭配提供完美轉送私密的密碼套件。AWS KMS 會使用所有其他 AWS API 操作可用的相同登入資料和政策機制 AWS Identity and Access Management (IAM) 來驗證和授權您的請求。

AWS KMS 設計目標

AWS KMS 的設計符合下列要求。

耐久性

密碼編譯金鑰的耐久性旨在與中最高的耐久性服務相同 AWS。單一密碼編譯金鑰可以加密長時間累積的大量資料。

值得信賴

金鑰的使用受到您定義和管理之存取控制政策的保護。沒有任何機制可匯出純文字 KMS 金鑰。密碼編譯金鑰的機密性至關重要。若要對 HSM 執行管理動作，需要有多名 Amazon 員工擁有對以仲裁為基礎之存取控制項的角色特定存取權。

低延遲和高輸送量

AWS KMS 在適合供其他 服務使用的延遲和輸送量層級提供密碼編譯操作 AWS。

獨立區域

AWS 為需要限制不同區域中資料存取的客戶提供獨立區域。可以在 AWS 區域內隔離金鑰用量。

隨機數字的安全來源

由於強式密碼編譯取決於真正不可預測的隨機數字產生，AWS KMS 提供高品質且經過驗證的隨機數字來源。

稽核

AWS KMS 在 AWS CloudTrail 日誌中記錄密碼編譯金鑰的使用和管理。您可以使用 AWS CloudTrail 日誌來檢查密碼編譯金鑰的使用，包括 AWS 服務代表您使用金鑰。

為了實現這些目標，AWS KMS 系統包含一組管理「網域」的 AWS KMS 運算子和服務主機運算子（統稱為「運算子」）。網域是區域定義的一組 AWS KMS 伺服器、HSMs 和運算子。每個 AWS KMS 運算子都有硬體字串，其中包含用於驗證其動作的私有和公有金鑰對。HSM 具有額外的私有和公有金鑰對，可建立保護 HSM 狀態同步的加密金鑰。

AWS KMS keys

您為在自己的密碼編譯應用程式中使用而建立和管理的 KMS 金鑰類型稱為客戶受管金鑰。客戶受管金鑰也可以與使用 KMS 金鑰的 AWS 服務搭配使用，以代您加密服務存放的資料。對於希望完全控制金鑰生命週期和用量的客戶，建議使用客戶受管金鑰。您的帳戶中有客戶受管金鑰需要每月成本。此外，請求使用和/或管理金鑰會產生使用成本。如需詳細資訊，請參閱 [AWS Key Management Service 定價](#)。

在某些情況下，客戶可能希望 AWS 服務加密其資料，但他們不希望管理金鑰的額外負荷，也不想支付金鑰的費用。AWS 受管金鑰是存在於您帳戶中的 KMS 金鑰，但只能在特定情況下使用。具體而言，它只能在您操作 AWS 的服務內容中使用，並且只能由帳戶中存在金鑰的主體使用。您無法管理有關這些金鑰生命週期或許可的任何內容。當您在 AWS 服務中使用加密功能時，您可能會看到 AWS 受管金鑰；它們使用「aws<service code>」形式的別名。例如，aws/ebs 金鑰只能用來加密 EBS 磁碟區，而且只能用於與金鑰相同帳戶中的 IAM 主體所使用的磁碟區。請考慮 AWS 受管金鑰 縮小範圍的，僅供您帳戶中的使用者用於您帳戶中的資源。您無法 AWS 受管金鑰 與其他 帳戶共用在 下加密的資源。雖然 AWS 受管金鑰 可以免費存在於您的帳戶中，但指派給金鑰 AWS 的服務會向您收取任何使用此金鑰類型的費用。

AWS 受管金鑰 是自 2021 年起不再為新 AWS 服務建立的舊版金鑰類型。反之，新的（和舊版）AWS 服務預設會使用所謂的 AWS 擁有的金鑰來加密客戶資料。是一種 AWS 擁有的金鑰 KMS 金鑰，位於由 AWS 服務管理的帳戶中，因此服務運算子能夠管理其生命週期和使用許可。透過使用 AWS 擁有的金鑰，AWS 服務可以透明地加密您的資料，並允許輕鬆跨帳戶或跨區域共用資料，而無需擔心金鑰許可。AWS 擁有的金鑰 用於 encryption-by-default 工作負載，提供更輕鬆、更自動化的資料保護。由於這些金鑰是由擁有和管理 AWS，因此您不需要支付其存在或其用量的費用，您無法變更其政策，也無法稽核這些金鑰的活動，也無法刪除它們。在控制很重要時使用客戶受管金鑰，但在便利性最重要 AWS 擁有的金鑰 時使用。

	客戶受管金鑰	AWS 受管金鑰	AWS 擁有的金鑰
金鑰政策	由客戶獨家控制	由服務控制；可由客戶檢視	完全控制且只能由加密資料的 AWS 服務檢視
日誌	CloudTrail 客戶追蹤或事件資料存放區	CloudTrail 客戶追蹤或事件資料存放區	客戶無法檢視
生命週期管理	客戶管理輪換、刪除和區域位置	AWS KMS 管理輪換（每年）、刪除和區域位置	AWS 服務 管理輪換、刪除和區域位置
定價	金鑰存在的每月費用（每小時按比例）。也會針對金鑰使用量收費	不收取月費；但呼叫者需要支付這些金鑰上的 API 使用費	不向客戶收費

您建立的 KMS 金鑰是[客戶受管金鑰](#)。使用 KMS 金鑰來加密您的服務資源的 AWS 服務 通常會為您建立金鑰。在您的 AWS 帳戶中 AWS 服務 建立的 KMS 金鑰為 [AWS 受管金鑰](#)。在服務帳戶中 AWS 服務 建立的 KMS 金鑰為 [AWS 擁有的金鑰](#)。

KMS 金鑰類型	可以檢視 KMS 金鑰中繼資料	可以管理 KMS 金鑰	僅用於我的 AWS 帳戶	自動旋轉	定價
客戶受管金鑰	是	是	是	選用。	月費 (按小時比例計算)

KMS 金鑰類型	可以檢視 KMS 金鑰中繼資料	可以管理 KMS 金鑰	僅用於我的 AWS 帳戶	自動旋轉	定價
					每次使用費
AWS 受管金鑰	是	否	是	必要。每年（約 365 天）。	沒有月費 使用費（有些會為您 AWS 服務支付此費用）
AWS 擁有的金鑰	否	否	否	AWS 服務管理輪換策略	沒有費用

[AWS 與整合的服務 AWS KMS](#)在對 KMS 金鑰的支援方面有所不同。有些 AWS 服務預設會使用 AWS 擁有的金鑰或來加密您的資料 AWS 受管金鑰。有些 AWS 服務支援客戶受管金鑰。其他服務 AWS 支援所有類型的 KMS 金鑰，讓您能夠輕鬆使用 AWS 擁有的金鑰、的可見性 AWS 受管金鑰，或控制客戶受管金鑰。如需 AWS 服務提供的加密選項詳細資訊，請參閱《使用者指南》中的靜態加密主題或服務開發人員指南。

客戶自管金鑰

您建立的 KMS 金鑰是客戶受管金鑰。客戶受管金鑰是您建立、擁有和管理 AWS 帳戶之 中的 KMS 金鑰。您可以完全控制這些 KMS 金鑰，包括建立和維護其[金鑰政策](#)、[IAM 政策和授予](#)、[啟用和停用](#)這些項目、[輪換其密碼編譯材料](#)、[新增標籤](#)、[建立參考 KMS 金鑰的別名](#)，以及[排程 KMS 金鑰供刪除](#)。

客戶受管金鑰會出現在 AWS KMS 的 AWS 管理主控台 客戶受管金鑰頁面。如要明確識別客戶受管金鑰，請使用 [DescribeKey](#) 操作。對於客戶受管金鑰，DescribeKey 回應的 KeyManager 欄位值是 CUSTOMER。

您可以在密碼編譯操作中使用您的客戶受管金鑰，並在 AWS CloudTrail 日誌中稽核其使用情形。此外，許多與 [AWS KMS 整合的 AWS 服務](#)可讓您指定客戶受管金鑰，以保護為您存放和管理的資料。

客戶受管金鑰會衍生每月費用，以及超出免費方案部分的使用費用。它們會計入您帳戶的 AWS KMS [配額](#)。如需詳細資訊，請參閱 [AWS Key Management Service 定價和配額](#)。

AWS 受管金鑰

AWS 受管金鑰 是帳戶中的 KMS 金鑰，由[AWS 與 整合 AWS KMS](#)的服務代表您建立、管理和使用。

有些 AWS 服務可讓您選擇 AWS 受管金鑰 或客戶受管金鑰，以保護該服務中的資源。一般而言，除非您需要控制保護 資源的加密金鑰，否則 AWS 受管金鑰 是不錯的選擇。您不需要建立或維護金鑰或其金鑰政策，而且 AWS 受管金鑰永遠沒有月費。

您有權[檢視帳戶中的 AWS 受管金鑰](#)、[檢視其金鑰政策](#)，以及[稽核其在 日誌中的使用](#)情形。AWS CloudTrail 不過，您無法變更任何屬性 AWS 受管金鑰、輪換它們、變更其金鑰政策，或排定刪除它們。此外，您無法直接 AWS 受管金鑰 在密碼編譯操作中使用；建立它們的服務會代表您使用它們。

組織中[的資源控制政策](#)不適用於 AWS 受管金鑰。

AWS 受管金鑰 會出現在 AWS 管理主控台的 AWS 受管金鑰頁面上 AWS KMS。您也可以 AWS 受管金鑰 依別名識別，其格式為 `aws/service-name`，例如 `aws/redshift`。若要明確識別 AWS 受管金鑰，請使用 [DescribeKey](#) 操作。對於 AWS 受管金鑰，DescribeKey 回應的 KeyManager 欄位值是 AWS。

所有 AWS 受管金鑰 都會每年自動輪換。您無法變更此輪換排程。

Note

在 2022 年 5 月，的輪換排程 AWS 受管金鑰 從每三年（約 1,095 天）AWS KMS 變更為每年（約 365 天）。

不收取月費 AWS 受管金鑰。超過免費方案可能需要支付使用費，但有些 AWS 服務會為您支付這些費用。如需詳細資訊，請參閱該服務使用者指南或開發人員指南中的靜態加密主題。如需詳細資訊，請參閱 [AWS Key Management Service 定價](#)。

AWS 受管金鑰 不會計入您帳戶每個區域中 KMS 金鑰數量的資源配額。但是，如果代表您帳戶中的委託人使用，則會將 KMS 金鑰計入請求配額中。如需詳細資訊，請參閱[配額](#)。

AWS 擁有的金鑰

AWS 擁有的金鑰 是 AWS 服務擁有和管理用於多個的 KMS 金鑰集合 AWS 帳戶。雖然 AWS 擁有的金鑰 不在您的 中 AWS 帳戶，但 AWS 服務可以使用 AWS 擁有的金鑰 來保護您帳戶中的資源。

有些 AWS 服務可讓您選擇 AWS 擁有的金鑰 或客戶受管金鑰。一般而言，除非您需要稽核或控制保護資源的加密金鑰，否則 AWS 擁有的金鑰 是不錯的選擇。AWS 擁有的金鑰 完全免費（無需每月費用

或使用費)，它們不會計入您帳戶的[AWS KMS 配額](#)，而且易於使用。您不需要建立或維護金鑰或其金鑰政策。

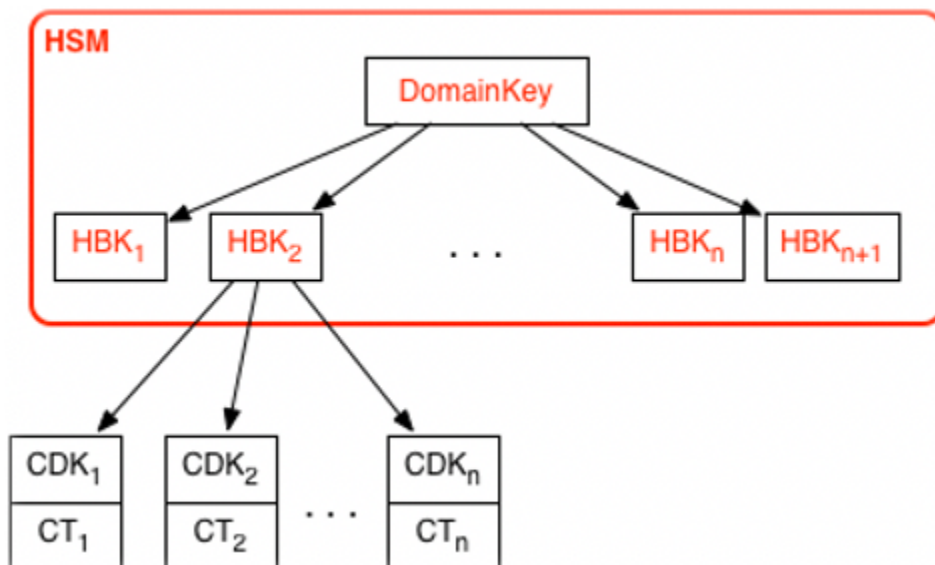
的輪換會因服務 AWS 擁有的金鑰 而異。如需特定 輪換的相關資訊 AWS 擁有的金鑰，請參閱 服務使用者指南或開發人員指南中的靜態加密主題。

AWS KMS key 階層

您的金鑰階層從最上層邏輯金鑰 開始 AWS KMS key。KMS 金鑰代表頂層金鑰材料的容器，並且在 AWS 服務命名空間內使用 Amazon Resource Name (ARN) 進行專屬定義。ARN 包括專屬產生的金鑰識別符，金鑰 ID。KMS 金鑰是根據使用者透過 提出的請求所建立 AWS KMS。接收時，AWS KMS 會請求建立要放置在 KMS 金鑰容器中的初始 HSM 備份金鑰 (HBK)。HBK 是在網域中的 HSM 上產生的，並且設計為永遠不會以純文字形式從 HSM 匯出。相反地，HBK 會在受 HSM 管理的網域金鑰下加密匯出。這些匯出的 HBK 稱為匯出的金鑰字符 (EKT)。

EKT 會匯出至高度耐用、低延遲的儲存體。例如，假設您收到邏輯 KMS 金鑰的 ARN。這代表您的金鑰階層或密碼編譯內容的頂端。您可以在帳戶中建立多個 KMS 金鑰，並在 KMS 金鑰上設定政策，就像任何其他 AWS 具名資源一樣。

在特定 KMS 金鑰的階層中，HBK 可以被視為 KMS 金鑰的版本。當您想要輪換 KMS 金鑰時 AWS KMS，會建立新的 HBK，並與 KMS 金鑰建立關聯，做為 KMS 金鑰的作用中 HBK。系統會保留較舊的 HBK，可用來解密和驗證先前受保護的資料。但是，只有作用中的密碼編譯金鑰可以用來保護新的資訊。



您可以透過 提出請求 AWS KMS，以使用 KMS 金鑰直接保護資訊，或請求受 KMS 金鑰保護的其他 HSM 產生的金鑰。這些金鑰稱為客戶資料金鑰 (CDK)。CDK 可以傳回加密為加密文字 (CT)，純文

字，或兩者兼而有之。在 KMS 金鑰下加密的所有物件（客戶提供的資料或 HSM 產生的金鑰）只能透過呼叫在 HSM 上解密 AWS KMS。

傳回的加密文字或解密的承載永遠不會存放在其中 AWS KMS。資訊會透過至 AWS KMS 的 TLS 連線傳回給您。這也適用於 AWS 服務代表您進行的呼叫。

金鑰階層和特定金鑰屬性會出現在下表中。

金鑰	Description	生命週期
網域金鑰	僅在 HSM 記憶體中用來包裝 KMS 金鑰 (HSM 備份金鑰) 版本的 256 位元 AES-GCM 金鑰。	每日輪換 ¹
HSM 備份金鑰	256 位元對稱金鑰或 RSA 或橢圓曲線私有金鑰，用於保護客戶資料和金鑰，並以網域金鑰加密存放。一個或多個 HSM 備份金鑰由 KMS 金鑰組成，用 keyId 表示。	每年輪換 ² (選用組態)
衍生的加密金鑰	僅在 HSM 記憶體中用來加密客戶資料和金鑰的 256 位元 AES-GCM 金鑰。從每個加密的 HBK 衍生。	每次加密使用一次，並在解密時重新產生
客戶資料金鑰	以純文字和加密文字從 HSM 匯出之使用者定義的對稱或非對稱金鑰。 在 HSM 備份金鑰下加密，並透過 TLS 通道傳回給授權的使用者。	輪換和使用由應用程式控制

¹ AWS KMS 可能不時放寬網域金鑰輪換至最多每週一次，以考慮網域管理和組態任務。

² AWS KMS 代表您 AWS 受管金鑰 建立和管理的預設 會每年自動輪換。

金鑰識別符 (KeyID)

金鑰識別符形同 KMS 金鑰的名稱，可幫助您在主控台中識別您的 KMS 金鑰。您可以使用它們指出要在 AWS KMS API 操作、金鑰政策、IAM 政策和授予中使用的 KMS 金鑰。金鑰識別符的值與 KMS 金鑰相關聯的金鑰資料完全無關。

AWS KMS 定義數個金鑰識別符。當您建立 KMS 金鑰時，AWS KMS 會產生金鑰 ARN 和金鑰 ID，這是 KMS 金鑰的屬性。當您建立**別名**時，會根據您定義的別名名稱 AWS KMS 產生別名 ARN。您可以在和 AWS KMS API 中檢視金鑰 AWS 管理主控台 和別名識別符。

在 AWS KMS 主控台中，您可以依金鑰 ARN、金鑰 ID 或別名名稱檢視和篩選 KMS 金鑰，並依金鑰 ID 和別名名稱排序。如需在主控台中尋找金鑰識別符的說明，請參閱[the section called “尋找金鑰 ID 和金鑰 ARN”](#)。

在 AWS KMS API 中，您用來識別 KMS 金鑰的參數會命名為 `KeyId` 或變化，例如 `TargetKeyId` 或 `DestinationKeyId`。不過，這些參數的數值並不限於金鑰 ID。有些參數可以接受任何有效的金鑰識別符。如需每個參數值的相關資訊，請參閱 AWS Key Management Service API 參考中的參數描述。

Note

使用 AWS KMS API 時，請注意您使用的金鑰識別符。不同的 API 需要不同的金鑰識別符。一般而言，請使用您任務適用之最完整且最實用的金鑰識別符。

AWS KMS 支援下列金鑰識別符。

金鑰 ARN

金鑰 ARN 是 KMS 金鑰的 Amazon Resource Name (ARN)。它是 KMS 金鑰唯一、完全合格的識別符。金鑰 ARN 包含 AWS 帳戶、區域和金鑰 ID。如需尋找 KMS 金鑰的金鑰 ARN 說明，請參閱[the section called “尋找金鑰 ID 和金鑰 ARN”](#)。

金鑰 ARN 的格式如下：

```
arn:<partition>:kms:<region>:<account-id>:key/<key-id>
```

以下是單一區域 KMS 金鑰的範例金鑰 ARN。

```
arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab
```

[多區域金鑰](#)之金鑰 ARN 的 *key-id* 元素以 `mrk-` 字首開頭。以下是多區域金鑰的範例金鑰 ARN。

```
arn:aws:kms:us-west-2:111122223333:key/mrk-1234abcd12ab34cd56ef1234567890ab
```

金鑰 ID

金鑰 ID 可唯一地識別帳戶和區域內的 KMS 金鑰。如需尋找 KMS 金鑰的金鑰 ID 說明，請參閱 [the section called “尋找金鑰 ID 和金鑰 ARN”](#)。

以下是單一區域 KMS 金鑰的範例金鑰 ID。

```
1234abcd-12ab-34cd-56ef-1234567890ab
```

[多區域金鑰](#)的金鑰 ID 以 `mrk-` 字首開頭。以下是多區域金鑰的範例金鑰 ID。

```
mrk-1234abcd12ab34cd56ef1234567890ab
```

別名 ARN

別名 ARN 是別名的 Amazon Resource Name AWS KMS (ARN)。它是別名及其代表的 KMS 金鑰唯一、完全合格的識別符。別名 ARN 包含 AWS 帳戶、區域和別名名稱。

別名 ARN 在任何時候可識別一個特定的 KMS 金鑰。不過，因為您可以變更與別名相關聯的 KMS 金鑰，所以別名 ARN 可以在不同的時間識別不同的 KMS 金鑰。如需尋找 KMS 金鑰的別名 ARN 說明，請參閱 [尋找 KMS 金鑰的別名名稱和別名 ARN](#)。

別名 ARN 的格式如下：

```
arn:<partition>:kms:<region>:<account-id>:alias/<alias-name>
```

以下是虛構 ExampleAlias 的 ARN 別名。

```
arn:aws:kms:us-west-2:111122223333:alias/ExampleAlias
```

別名名稱

別名名稱是最多 256 個字元的字串。其可唯一地識別帳戶和區域內關聯的 KMS 金鑰。在 AWS KMS API 中，別名名稱一律以開頭 `alias/`。如需尋找 KMS 金鑰之別名名稱的說明，請參閱 [尋找 KMS 金鑰的別名名稱和別名 ARN](#)。

別名的格式如下：

```
alias/<alias-name>
```

例如：

```
alias/ExampleAlias
```

別名名稱的 `aws/` 字首預留給 [AWS 受管金鑰](#)。您無法使用此字首建立別名。例如，Amazon Simple Storage Service (Amazon S3) AWS 受管金鑰 的別名名稱如下。

```
alias/aws/s3
```

中的非對稱金鑰 AWS KMS

非對稱 KMS 金鑰表示以數學方式相關的公有金鑰和私有金鑰對。您可以將公有金鑰提供給任何人，即使不受信任的人也可以，但私有金鑰則一定要保管好。

在非對稱 KMS 金鑰中，私有金鑰是在 中建立 AWS KMS，絕不會保持 AWS KMS 未加密狀態。若要使用私有金鑰，您必須呼叫 AWS KMS。您可以透過呼叫 AWS KMS API 操作 AWS KMS，在 中使用公有金鑰。或者，您可以 [下載公有金鑰](#) 並在外部使用 AWS KMS。

如果您的使用案例需要無法呼叫的使用者 AWS 在 外部加密 AWS KMS，非對稱 KMS 金鑰是不錯的選擇。不過，如果您要建立 KMS 金鑰來加密您在 AWS 服務中存放或管理的資料，請使用對稱加密 KMS 金鑰。[AWS 與 整合的 服務 AWS KMS](#) 只會使用對稱加密 KMS 金鑰來加密您的資料。這些服務不支援使用非對稱 KMS 金鑰進行加密。

使用 簽署大於 4 KB 的訊息時 AWS KMS，您必須在簽署 AWS KMS 之前對 外部的訊息進行雜湊。AWS KMS 提供三種處理訊息輸入 `MessageType` 的選項：RAW 純文字訊息（其中 AWS KMS 執行雜湊）、預先雜湊訊息 DIGEST（其中 AWS KMS 略過雜湊步驟），以及 EXTERNAL_MU 特別是輸入為 64 位元組代表 μ 值的 ML-DSA KMS 金鑰規格。對於超過 4 KB 限制的大型訊息，請在外部雜湊訊息，並在 calling AWS KMS [Sign](#) and AWS KMS [Verify](#) 操作時使用 [MessageType: DIGEST](#)（或 [MessageType: EXTERNAL_MU](#) ML-DSA KMS 金鑰）。

AWS KMS 支援多種類型的非對稱 KMS 金鑰。

RSA KMS 金鑰

具有 RSA 金鑰對的 KMS 金鑰，用於加密和解密或簽署和驗證（但不能同時進行兩者）。AWS KMS 支援多種金鑰長度，以滿足不同的安全需求。

如需 AWS KMS 支援 RSA KMS 金鑰之加密和簽署演算法的技術詳細資訊，請參閱 [RSA 金鑰規格](#)。

橢圓曲線 (ECC) KMS 金鑰

具有橢圓曲線金鑰對的 KMS 金鑰，用於簽署和驗證或衍生共用秘密（但不能同時產生兩者）。AWS KMS 支援數個常用的曲線。

如需 AWS KMS 支援 ECC KMS 金鑰之簽署演算法的技術詳細資訊，請參閱[橢圓曲線金鑰規格](#)。

ML-DSA KMS 金鑰

具有 ML-DSA 金鑰對的 KMS 金鑰，用於簽署和驗證。ML-DSA 是由美國國家標準技術研究所 (NIST) 開發的量子後密碼編譯標準，可防止量子運算帶來的安全威脅。ML-DSA 是從 RSA 或橢圓曲線數位簽章演算法轉換為量子後安全密碼編譯的組織建議使用的數位簽章演算法。

AWS KMS 支援多種金鑰長度，以滿足不同的安全需求。如需 AWS KMS 支援 ML-DSA KMS 金鑰之簽署演算法的技術詳細資訊，請參閱[ML-DSA 金鑰規格](#)。

SM2 KMS 金鑰（僅限中國區域）

具有 SM2 金鑰對的 KMS 金鑰，用於加密和解密、簽署和驗證，或衍生共用秘密（您必須選擇一個[Key usage](#)類型）。

如需 AWS KMS 支援 SM2 KMS 金鑰的加密和簽署演算法技術詳細資訊（僅限中國區域），請參閱[SM2 金鑰規格](#)。

如需有關如何選擇非對稱金鑰組態的說明，請參閱[選擇要建立的 KMS 金鑰類型](#)。

區域

AWS 區域 AWS KMS 支援的所有 都支援非對稱 KMS 金鑰和非對稱資料金鑰對。

進一步了解

- 若要建立非對稱 KMS 金鑰，請參閱[建立非對稱 KMS 金鑰](#)。
- 若要建立多區域非對稱 KMS 金鑰，請參閱[建立多區域主索引鍵](#)。
- 若要了解如何使用非對稱 KMS 金鑰簽署訊息並驗證簽章，請參閱 AWS 安全部落格中的[使用 AWS KMS 新的非對稱金鑰功能進行數位簽署](#)。
- 若要了解刪除非對稱 KMS 金鑰的特殊考量，請參閱[Deleting asymmetric KMS keys](#)。
- 若要識別和檢視非對稱 KMS 金鑰，請參閱[識別非對稱 KMS 金鑰](#)。

中的 HMAC 金鑰 AWS KMS

雜湊訊息驗證碼 (HMAC) KMS 金鑰是對稱金鑰，用於在 AWS KMS 中產生和驗證 HMAC。與每個 HMAC KMS 金鑰關聯的唯一金鑰資料提供了 HMAC 演算法所需的私密金鑰。您可以將 HMAC KMS 金鑰與 [GenerateMac](#) 和 [VerifyMac](#) 操作搭配使用來驗證 AWS KMS 之中的資料完整性和真實性。

HMAC 演算法結合了密碼編譯雜湊函數和共享私密金鑰。它們接受訊息和私密金鑰 (如 HMAC KMS 金鑰中的金鑰資料)，並傳回一個唯一且固定大小的代碼或標籤。如果訊息中即使只有一個字元更改，或者私密金鑰不相同，則產生的標籤將完全不同。透過要求私密金鑰，HMAC 還提供了真實性；如果沒有私密金鑰，就無法產生相同的 HMAC 標籤。HMAC 有時被稱為對稱簽章，因為它們的運作方式類似於數位簽章，但使用單一金鑰進行簽署和驗證。

AWS KMS 使用的 HMAC KMS 金鑰和 HMAC 演算法符合 [RFC 2104](#) 中定義的產業標準。AWS KMS [GenerateMac](#) 操作會產生標準 HMAC 標籤。HMAC KMS 金鑰是在根據 [FIPS 140-3 密碼編譯模組驗證計劃](#) (中國 (北京) 和中國 (寧夏) 區域除外) 認證的 AWS KMS 硬體安全模組中產生，絕不會保持 AWS KMS 未加密狀態。若要使用 HMAC KMS 金鑰，必須呼叫 AWS KMS。

您可使用 HMAC KMS 金鑰來確定訊息的真實性，例如 JSON Web 字符 (JWT)、字符化的信用卡資訊或提交的密碼。它們還可以用作安全金鑰衍生函數 (KDF)，特別是在需要確定性金鑰的應用程式中。

HMAC KMS 金鑰提供優於應用程式軟體 HMACs 的優勢，因為金鑰材料完全在內部產生和使用 AWS KMS，取決於您在金鑰上設定的存取控制。

Tip

最佳實務是建議您限制任何簽名機制 (包括 HMAC) 的有效時間。這可以阻止攻擊者使用簽署的訊息反覆建立有效性攻擊或在訊息被取代很長時間後反覆建立有效性攻擊。HMAC 標籤不包含時間戳記，但您可以在字符或訊息中包含時間戳記，以幫助您偵測何時重新整理 HMAC。

支援的密碼編譯操作

HMAC KMS 金鑰僅支援 [GenerateMac](#) 和 [VerifyMac](#) 密碼編譯操作。您無法使用 HMAC KMS 金鑰來加密資料或簽署訊息，或在 HMAC 操作中使用任何其他類型的 KMS 金鑰。使用 [GenerateMac](#) 操作時，您可以提供最多 4,096 個位元組的訊息、一個 HMAC KMS 金鑰以及與 HMAC 金鑰規格相容的 MAC 演算法，[GenerateMac](#) 會運算 HMAC 標籤。若要驗證 HMAC 標籤，您必須提供 HMAC 標籤以及 [GenerateMac](#) 用來運算原始 HMAC 標籤的相同訊息、HMAC KMS 金鑰以及 MAC 演算法。[VerifyMac](#) 操作會運算 HMAC 標籤並驗證它是否與提供的 HMAC 標籤相同。如果輸入和運算得出的 HMAC 標籤不相同，驗證會失敗。

HMAC KMS 金鑰不支援[金鑰自動輪換](#)，而且您無法在[自訂金鑰存放區](#)建立 HMAC KMS 金鑰。

如果您要建立 KMS 金鑰來加密 AWS 服務中的資料，請使用對稱加密金鑰。您無法使用 HMAC KMS 金鑰。

區域

所有 支援的 都 AWS KMS 支援 HMAC KMS AWS 區域 金鑰。

進一步了解

- 若要建立 HMAC KMS 金鑰，請參閱 [建立 HMAC KMS 金鑰](#)。
- 若要建立多區域 HMAC KMS 金鑰，請參閱 [中的多區域金鑰 AWS KMS](#)。
- 若要檢查 AWS KMS 主控台為 HMAC KMS 金鑰設定的預設金鑰政策差異，請參閱 [the section called “允許金鑰使用者在密碼編譯操作中使用 KMS 金鑰”](#)。
- 若要識別和檢視 HMAC KMS 金鑰，請參閱 [識別 HMAC KMS 金鑰](#)。
- 若要了解有關使用 HMAC 建立 JSON Web 字符的詳細資訊，請參閱《AWS 安全部落格》中的「[如何在 AWS KMS內保護 HMAC](#)」。
- 在官方 AWS Podcast 上收聽 Podcast：[AWS Key Management Service的 HMAC 簡介](#)。

中的 ML-DSA 金鑰 AWS KMS

AWS Key Management Service (AWS KMS) 支援用於量子後密碼編譯簽章的 Module-Lattice 數位簽章演算法 (ML-DSA)。此實作遵循[聯邦資訊處理標準 \(FIPS\) 204 標準](#)，以協助防範未來的量子運算威脅。會在 FIPS 140-3 安全層級 3 驗證的硬體安全模組中 AWS KMS 建立並保護所有 ML-DSA 金鑰和簽章操作。為了協助平衡安全性與效能，中的 ML-DSA 透過不同的金鑰規格 AWS KMS 提供三種不同的安全層級：ML_DSA_44、ML_DSA_65 和 ML_DSA_87。

AWS KMS 支援使用 訊息類型對高達 4 KB RAW的訊息進行非對稱金鑰簽章。對於較大的訊息，您必須外部運算用於 ML-DSA 簽署的 64 位元組訊息表示 μ ，如 NIST FIPS 204 第 6.2 節所定義。在 AWS KMS [Sign](#) 操作中使用EXTERNAL_MU訊息類型來指定此預先處理的 64 位元組訊息。外部計算的 μ 所產生的簽章與使用相同訊息和私有金鑰RAW時的簽章相同。請注意，此簽署不同於 NIST FIPS 204 第 5.4 節中的「雜湊前」ML-DSA 或 HashML-DSA。

如需使用 ML-DSA 和 EXTERNAL_MU 訊息類型的詳細資訊，請參閱 [ML-DSA 金鑰規格](#)。

如需使用 ML-DSA 和 EXTERNAL_MU 訊息類型的範例，請參閱 [使用 ML-DSA 金鑰對進行離線驗證](#)。

中的多區域金鑰 AWS KMS

AWS KMS 支援多區域金鑰，這些金鑰 AWS KMS keys 位於不同的 AWS 區域中，可以互換使用 – 就像您在多個區域中擁有相同的金鑰一樣。每組相關的多區域金鑰都具有相同的金鑰材料和[金鑰 ID](#)，因此您可以在一個中加密資料 AWS 區域，並在不同的中解密資料，AWS 區域而無需重新加密或進行跨區域呼叫 AWS KMS。

如同所有 KMS 金鑰，多區域金鑰永遠不會保持 AWS KMS 未加密狀態。您可以建立對稱或非對稱多區域金鑰以進行加密或簽署，或是建立 HMAC 多區域金鑰以產生和驗證 HMAC 標籤，或是建立[含有匯入金鑰資料或 AWS KMS 產生之金鑰資料的多區域金鑰](#)。您必須獨立管理每個多區域金鑰，包括建立別名和標籤、設定其金鑰政策和授予，以及選擇性地將其啟用和停用。您可以在所有可以使用單一區域金鑰執行的密碼編譯操作中使用多區域金鑰。

多區域金鑰是靈活且強大的解決方案，適用於許多常見的資料安全案例。

災難復原

在備份和復原架構中，多區域金鑰可讓您在不斷的情況下處理加密的資料，即使發生 AWS 區域中斷也一樣。備份區域中維護的資料可以在備份區域中解密，備份區域中新加密的資料可以在主要區域 (當該區域還原時) 中解密。

全域資料管理

全球營運的企業需要全球分散式資料，這些資料可一致地跨 AWS 區域提供。您可以在資料所在的所有區域中建立多區域金鑰，然後將金鑰當作單一區域金鑰使用，以避免跨區域呼叫的延遲，或是在每個區域中不同金鑰下重新加密資料的成本。

分散式簽署應用程式

需要跨區域簽章功能的應用程式可以使用多區域非對稱簽署金鑰，在不同 AWS 區域中一致且重複地產生相同的數位簽章。

如果您將憑證鏈結與單一全域信任存放區 (針對單一根憑證授權機構 (CA)) 和根 CA 簽署的區域中繼 CA，則不需要多區域金鑰。不過，如果您的系統不支援中繼 CA (例如應用程式簽署)，則可以使用多區域金鑰來為區域認證提供一致性。

跨越多個區域的主動-主動應用程式

某些工作負載和應用程式可以跨越主動-主動架構中的多個區域。對於這些應用程式，透過為針對可能跨區域邊界移動之資料的同時加密和解密操作提供相同的金鑰材料，多區域金鑰可以降低複雜性。

您可以搭配用戶端加密程式庫使用多區域金鑰，例如 [AWS Encryption SDK](#)、[AWS 資料庫加密 SDK](#) 和 [Amazon S3 用戶端加密](#)。

[AWS 與 整合以進行靜態加密或數位簽章的服務 AWS KMS](#)，目前會將多區域金鑰視為單一區域金鑰。其可能會重新包裝或重新加密在區域之間移動的資料。例如，Amazon S3 跨區域複寫會在目的地區域的 KMS 金鑰下解密和重新加密資料，即使複寫受多區域金鑰保護的物件也一樣。

多區域金鑰不適用於全域。您建立多區域主要金鑰，然後將其複寫到您在 [AWS 分割區](#) 內選取的區域中。然後，您可以獨立管理每個區域中的多區域金鑰。AWS KMS 永遠不會代表您自動將多區域金鑰建立或複寫到任何區域中。[AWS 受管金鑰](#) 是 AWS 服務在帳戶中為您建立的 KMS 金鑰，一律是單一區域金鑰。

在中國區域中，您可以使用多區域金鑰功能，在中國區域分割區 () 中複寫 KMS 金鑰 `aws-cn`。例如，您可以將金鑰從中國（北京）區域複寫到中國（寧夏）區域，或反向複寫。透過將金鑰從一個中國區域複寫到另一個中國區域，即表示您同意使用 AWS Key Management Service 目的地區域的，並遵守目的地區域的所有適用協議條款。您無法將金鑰從北京和寧夏區域複寫到中國區域分割區 AWS 以外的區域。同樣地，您無法將金鑰從中國區域分區以外的區域複寫到北京和寧夏區域。

您無法將現有的單一區域金鑰轉換為多區域金鑰。此設計可確保所有受現有單一區域金鑰保護的資料都維持相同的資料落地和資料主權屬性。

對於大多數資料安全需求，區域資源的區域隔離和容錯能力使標準 AWS KMS 單一區域金鑰成為最適合的解決方案。不過，當您需要跨多個區域加密或簽署用戶端應用程式中的資料時，多區域金鑰可能就是解決方案。

區域

AWS 區域 支援的所有 AWS KMS 都支援多區域金鑰。

定價和配額

一組相關多區域金鑰中的每個金鑰都會計為一個 KMS 金鑰，用於定價和配額。[AWS KMS 配額](#) 會針對帳戶的每個區域個別計算。在每個區域中使用和管理多區域金鑰會計為該區域的配額。

支援的 KMS 金鑰類型

您可以建立以下類型的多區域 KMS 金鑰：

- 對稱加密 KMS 金鑰

- 非對稱 KMS 金鑰
- HMAC KMS 金鑰
- 包含匯入金鑰資料的 KMS 金鑰

您無法在自訂金鑰存放區建立多區域金鑰。

進一步了解

- 若要了解如何控制對多區域 KMS 金鑰的存取，請參閱 [控制對多區域金鑰的存取](#)。
- 若要建立任何類型的多區域主要 KMS 金鑰，請參閱 [建立多區域主索引鍵](#)。
- 若要建立多區域複本 KMS 金鑰，請參閱 [建立多區域複本金鑰](#)。
- 若要更新主要區域，請參閱 [變更一組多區域金鑰中的主金鑰](#)。
- 若要識別和檢視多區域 KMS 金鑰，請參閱 [識別 HMAC KMS 金鑰](#)。
- 若要了解刪除多區域 KMS 金鑰的特殊考量，請參閱 [Deleting multi-Region keys](#)。

術語與概念

下列術語和概念與多區域金鑰搭配使用。

多區域金鑰

多區域金鑰是在不同 AWS 區域中具有相同金鑰 ID 和金鑰材料 (以及其他[共用屬性](#)) 的一組 KMS 金鑰之一。每個多區域金鑰都是功能完善的 KMS 金鑰，可完全獨立於其相關的多區域金鑰之外使用。由於所有相關的多區域金鑰都具有相同的金鑰 ID 和金鑰材料，因此它們可以互通，也就是說，任何 中的任何相關多區域金鑰 AWS 區域 都可以解密任何其他相關多區域金鑰加密的加密文字。

您在建立 KMS 金鑰時會設定其多區域屬性。您無法在現有金鑰上變更多區域屬性。您無法將單一區域金鑰轉換為多區域金鑰，或將多區域金鑰轉換為單一區域金鑰。若要將現有的工作負載移至多區域案例，您必須重新加密資料，或使用新的多區域金鑰建立新的簽章。

多區域金鑰可以是[對稱或非對稱](#)，而且可以使用 AWS KMS 金鑰材料或[匯入的金鑰材料](#)。您無法在[自訂金鑰存放區](#)建立多區域金鑰。

在一組相關的多區域金鑰中，任何時候都只有一個[主要金鑰](#)。您可以在其他 AWS 區域建立該主要金鑰的[複本金鑰](#)。您也可以[更新主要區域](#)，它會將主要金鑰變更為複本金鑰，並將指定的複本金鑰變更為主要金鑰。不過，每個 只能維護一個主索引鍵或複本索引鍵 AWS 區域。所有區域必須在相同的 [AWS 分割區](#) 中。

您可以在相同或不同的 AWS 區域中擁有多組相關的多區域金鑰。雖然相關的多區域金鑰可互通操作，但不相關的多區域金鑰無法互通操作。

主索引鍵

多區域主金鑰是 KMS 金鑰，可複寫至相同分割區 AWS 區域 中的其他。每組多區域金鑰只有一個主要金鑰。

主要金鑰與複本金鑰有下列幾點不同之處：

- 只有主要金鑰可以[複寫](#)。
- 主要金鑰是其[複本金鑰](#)之[共用屬性](#)的來源，包括金鑰資料和金鑰 ID。
- 您可以僅針對主要金鑰啟用和停用[自動金鑰輪換](#)。
- 您可以隨時[排程刪除主要金鑰](#)。但在刪除其所有複本金鑰之前 AWS KMS，不會刪除主要金鑰。

不過，主要和複本金鑰在任何密碼編譯屬性中都沒有差異。您可以互換使用主要金鑰及其複本金鑰。

您不需要複寫主要金鑰。您可以像使用任何 KMS 金鑰一樣使用它，並在有用時對其進行複寫。不過，由於多區域金鑰與單一區域金鑰具有不同的安全屬性，建議您只在計劃複寫時建立多區域金鑰。

複本金鑰

多區域複本金鑰是 KMS 金鑰，其[金鑰 ID](#) 和金鑰材料與其[主要金鑰](#)和相關複本金鑰相同，但存在於不同的 AWS 區域。

複本金鑰是功能完善的 KMS 金鑰，具有自己的金鑰政策、授予、別名、標籤和其他屬性。它不是主要金鑰或任何其他金鑰的指標複本。您可以使用複本金鑰，即使其主要金鑰和所有相關的複本金鑰已停用。您也可以將複本金鑰轉換為主要金鑰，將主要金鑰轉換為複本金鑰。建立後，複本金鑰僅依賴其主要金鑰進行[金鑰輪換](#)和[更新主要區域](#)。

主要和複本金鑰在任何密碼編譯屬性中都沒有差異。您可以互換使用主要金鑰及其複本金鑰。透過主要金鑰或複本金鑰加密的資料可以透過相同的金鑰或任何相關的主要金鑰或複本金鑰進行解密。

複寫

您可以將多區域[主索引鍵](#)複寫到相同分割區 AWS 區域 中的不同。當您這樣做時，會在指定的區域中 AWS KMS 建立多區域[複本金鑰](#)，其[金鑰 ID](#) 和其他[共用屬性](#)與其主金鑰相同。對於具有 AWS_KMS 原始伺服器的 KMS 金鑰，會 AWS KMS 安全地跨區域界限傳輸金鑰材料，並將其與新的複本金鑰建立關

聯，全部都在其中 AWS KMS。對於具有EXTERNAL原始伺服器的 KMS 金鑰，您必須匯入匯入主要區域金鑰的相同金鑰材料，以個別連接複本區域金鑰。

共用屬性

共用屬性是與其複本金鑰共用之多區域主要金鑰的屬性。會 AWS KMS 建立具有與主要金鑰相同共用屬性值的複本金鑰。然後，它會定期將主要金鑰的共用屬性值同步至其複本金鑰。您無法在複本金鑰上設定這些屬性。

以下是多區域金鑰的共用屬性。

- [金鑰 ID](#) – ([金鑰 ARN](#) 的 Region 元素有所不同。)
- [金鑰材料](#) — 一組相關多區域金鑰中的主要和複本金鑰共用相同的金鑰材料。對於金鑰材料是由 AWS KMS (AWS_KMS原始伺服器) 產生的多區域金鑰，會在複本建立時或透過自動或隨需輪換建立新的金鑰材料時 AWS KMS，安全地將所有金鑰材料從主要 傳輸到每個複本。對於具有匯入金鑰材料 (EXTERNAL 原始伺服器) 的多區域金鑰，會從主金鑰 AWS KMS 同步金鑰材料識別符，但您必須將金鑰材料單獨匯入每個複本金鑰。
- [金鑰資料來源](#)
- [金鑰規格](#)和加密演算法
- [金鑰用途](#)
- [自動金鑰輪換](#) – 您可以僅針對主要金鑰啟用和停用自動金鑰輪換。使用共用金鑰材料的所有版本建立新的複本金鑰。如需詳細資訊，請參閱[Rotating multi-Region keys](#)。
- [隨需輪換](#) — 您只能在主索引鍵上執行隨需輪換。對於金鑰材料由 AWS KMS (AWS_KMS 原始伺服器) 產生的多區域金鑰，會使用共用金鑰材料的所有版本 AWS KMS 建立複本金鑰。對於具有匯入金鑰材料 (EXTERNAL 原始伺服器) 的多區域金鑰，會將金鑰材料 ID 和金鑰材料描述從主要金鑰 AWS KMS 傳播到複本金鑰，但不傳播金鑰材料。您必須個別將正確的金鑰材料匯入每個複本金鑰。如需詳細資訊，請參閱[Rotating multi-Region keys](#)。

您也可以將相關多區域金鑰的主要和複本指定視為共用屬性。當您[建立新的複本金鑰](#)或[更新主金鑰](#)時，會將變更 AWS KMS 同步至所有相關多區域金鑰。完成這些變更後，所有相關的多區域金鑰都會準確地列出其主要金鑰和複本金鑰。

多區域金鑰的所有其他屬性都是獨立的屬性，包括金鑰描述、[金鑰政策](#)、[授予](#)、[啟用和停用的金鑰狀態](#)、[別名](#)和[標籤](#)。您可以在所有相關的多區域金鑰上為這些屬性設定相同的值，但是如果您變更獨立屬性的值，則 AWS KMS 不會將其同步。

您可以追蹤多區域金鑰之共用屬性的同步。在您的 AWS CloudTrail 日誌中，尋找[SynchronizeMultiRegionKey](#) 事件。

多區域金鑰的安全考量

僅在您需要 AWS KMS 多區域金鑰時使用。多區域金鑰為在 AWS 區域 之間移動加密資料的工作負載或需要跨區域存取的工作負載提供靈活且可擴展的解決方案。如果您必須跨區域共用、移動或備份受保護的資料，或者需要建立在不同區域中運作之應用程式的相同數位簽章，請考慮使用多區域金鑰。

不過，建立多區域金鑰的程序會在 AWS KMS 內跨 AWS 區域 邊界移動您的金鑰材料。由多區域金鑰產生的加密文字可能會由多個地理位置中的多個相關金鑰進行解密。區域隔離的服務和資源也有顯著的優勢。每個 AWS 區域 都是隔離的，且與其他區域各自獨立。區域提供容錯能力、穩定性和恢復能力，也可降低延遲。它們可讓您建立冗餘資源，這些資源會保持可用且不受其他區域運行中斷的影響。在 AWS KMS 中，它們也確保每個加密文字只能由一個金鑰解密。

多區域金鑰也會引發新的安全考量：

- 使用多區域金鑰，控制存取和強制執行資料安全政策會比較複雜。您需要確保會在多個隔離區域的金鑰上對政策進行一致地稽核。而且您需要使用政策來強制執行邊界，而不是依賴單獨的金鑰。

例如，您需要針對資料設定政策條件，以防止一個區域的薪資團隊能夠讀取不同區域的薪資資料。此外，您必須使用存取控制來防止出現以下案例：一個區域中的多區域金鑰保護一個租用戶的資料，而另一個區域中的相關多區域金鑰保護不同租用戶的資料。

- 跨區域稽核金鑰也比較複雜。使用多區域索引鍵，您需要檢查和協調多個區域的稽核活動，以完全了解受保護資料的金鑰活動。
- 符合資料落地規定可能會更加複雜。透過隔離區域，您可以確保資料落地和資料主權的合規性。指定區域中的 KMS 金鑰只能解密該區域中的敏感資料。在一個區域中加密的資料可以在任何其他區域中保持處於完全保護且無法存取的狀態。

若要使用多區域金鑰驗證資料落地和資料主權，您需要實作存取政策和跨多個區域編譯 AWS CloudTrail 事件。

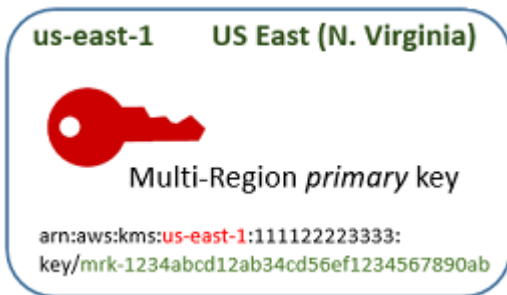
為了讓您更輕鬆地管理多區域金鑰的存取控制，複寫多區域金鑰的許可 ([kms:ReplicateKey](#)) 會與建立金鑰的標準許可 ([kms:CreateKey](#)) 分開。此外，AWS KMS 支援多區域金鑰的多種政策條件，包括 `kms:MultiRegion`，允許或拒絕建立、使用或管理多區域金鑰的許可，以及 `kms:ReplicaRegion`，這會限制可複寫多區域金鑰的區域。如需詳細資訊，請參閱[控制對多區域金鑰的存取](#)。

多區域金鑰的運作方式

首先在 AWS 區域 AWS KMS 支援的 中建立對稱或非對稱[多區域主索引鍵](#)，例如美國東部（維吉尼亞北部）。只有在建立金鑰時，您才會決定是單一區域金鑰還是多區域金鑰；之後就無法變更此屬性。與

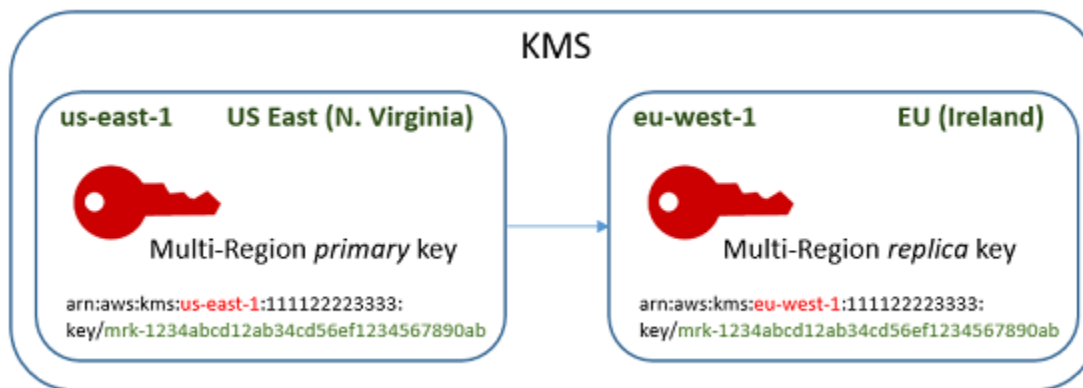
任何 KMS 金鑰一樣，您可以設定多區域金鑰的金鑰政策，並建立授予，以及新增分類和授權的別名和標籤。(這些是獨立屬性，並未與其他金鑰共用或同步。) 您可以在密碼編譯操作中使用多區域主要金鑰進行加密或簽署。

您可以在 AWS KMS 主控台中[建立多區域主金鑰](#)，或使用 [CreateKey](#) API，並將 `MultiRegion` 參數設定為 `true`。請注意，多區域金鑰具有開頭為 `mrk-` 的獨特金鑰 ID。您可以使用 `mrk-` 字首，以程式設計方式識別 MRK。



如果您選擇，您可以將多區域主索引鍵複寫到相同AWS 分割區 AWS 區域中的一個或多個不同，例如歐洲（愛爾蘭）。當您這樣做時，AWS KMS 會在指定的區域中建立複本金鑰，其金鑰 ID 和其他共用屬性與主金鑰相同。其結果是兩個相關多區域金鑰（主要金鑰和複本金鑰）可以互換使用。

您可以在 AWS KMS 主控台或使用 [ReplicateKey](#) API [建立多區域複本金鑰](#)。



由此產生的多區域複本金鑰是功能齊全的 KMS 金鑰，具有與主要金鑰相同的共用屬性。在所有其他方面，它是獨立的 KMS 金鑰，具有自己的描述、金鑰政策、授予、別名和標籤。啟用或停用多區域金鑰不會影響相關的多區域金鑰。您可以在密碼編譯操作中獨立使用主要金鑰和複本金鑰，或協調其使用。例如，您可以使用美國東部（維吉尼亞北部）區域的主要金鑰來加密資料、將資料移至歐洲（愛爾蘭）區域，然後使用複本金鑰來解密資料。

相關的多區域金鑰具有相同的金鑰 ID。其金鑰 ARN (Amazon Resource Name) 僅在區域字段中不同。例如，多區域主要金鑰和複本金鑰可能具有下列範例金鑰 ARN。金鑰 ID (金鑰 ARN 中的最後一個元素) 是相同的。兩個金鑰都有多區域金鑰的獨特金鑰 ID，開頭為 `mrk-`。

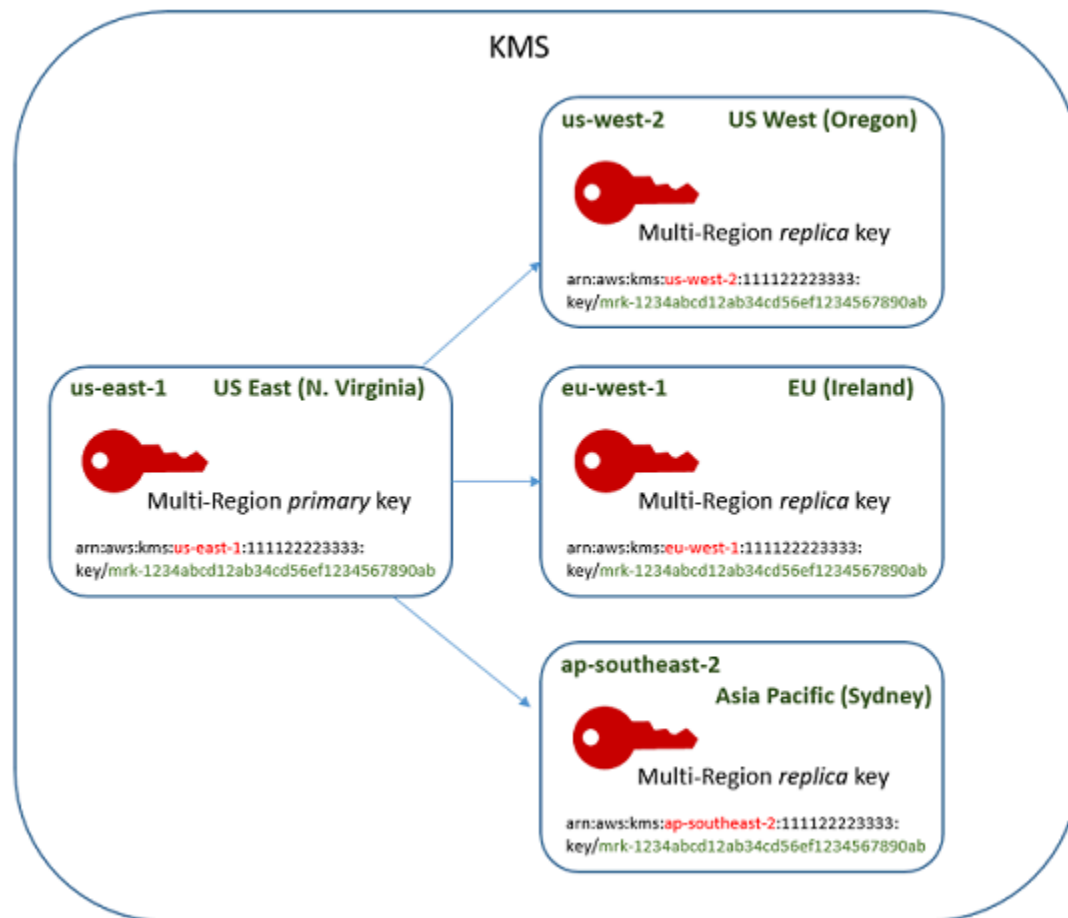

```

Primary key: arn:aws:kms:us-east-1:111122223333:key/mrk-1234abcd12ab34cd56ef12345678990ab
Replica key: arn:aws:kms:eu-west-1:111122223333:key/mrk-1234abcd12ab34cd56ef12345678990ab

```

具有相同的金鑰 ID 才能交互操作。加密時，會將 KMS 金鑰的金鑰 ID AWS KMS 繫結至加密文字，以便只能使用該 KMS 金鑰或具有相同金鑰 ID 的 KMS 金鑰解密加密文字。這項功能也讓相關的多區域金鑰易於識別，並且可以更輕鬆地對其進行互換使用。例如，在應用程式中使用時，您可以透過其共用金鑰 ID 來引用相關的多區域金鑰。然後，如有必要，請指定區域或 ARN 來區分這些金鑰。

隨著您的資料需求變更，您可以將主索引鍵複寫到相同分割區 AWS 區域中的其他，例如美國西部（奧勒岡）和亞太區域（雪梨）。其結果是具有相同金鑰材料和金鑰 ID 的四個相關多區域金鑰，如下圖所示。您可以獨立管理金鑰。對於具有匯入金鑰材料的多區域金鑰，您必須負責個別將金鑰材料匯入每個相關金鑰。您可以獨立或以協調的方式使用這些金鑰。例如，您可以使用亞太區域（雪梨）中的複本金鑰來加密資料、將資料移至美國西部（奧勒岡），然後使用美國西部（奧勒岡）的複本金鑰來解密資料。



多區域金鑰的其他考量包括下列各項。

同步共用屬性 — 如果多區域金鑰的**共用屬性**變更，AWS KMS 會自動將變更從**主金鑰**同步到其所有**複本金鑰**。您無法請求或強制同步共用屬性。會為您 AWS KMS 偵測和同步所有變更。不過，您可以使用 CloudTrail 日誌中的 [SynchronizeMultiRegionKey](#) 事件稽核同步。

例如，如果您在具有AWS_KMS原始伺服器的對稱多區域主金鑰上啟用自動金鑰輪換，會將該設定AWS KMS 複製到其所有複本金鑰。輪換金鑰材料時，所有相關多區域金鑰之間的輪換會同步，因此其仍然具有相同的當前金鑰材料，並存取所有舊版的金鑰材料。如果您建立新的複本金鑰，則該金鑰具有與所有相關多區域金鑰相同的當前金鑰材料，並可存取所有舊版金鑰材料。如需詳細資訊，請參閱[Rotating multi-Region keys](#)。

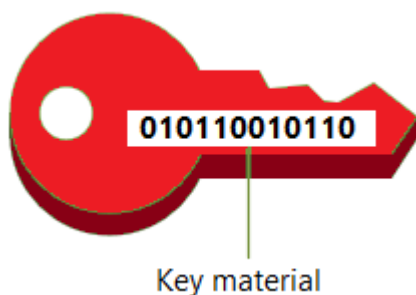
變更主要金鑰 – 每組多區域金鑰必須只有一個主要金鑰。**主要金鑰**是唯一可以複寫的金鑰。它也是其複本金鑰共用屬性的來源。但是您可以將主要金鑰變更為複本，並將其中一個複本金鑰升級為主要金鑰。您可以這樣做，以便您可以從特定區域刪除多區域主要金鑰，或者在更接近專案管理員的區域找到主要金鑰。如需詳細資訊，請參閱[變更一組多區域金鑰中的主金鑰](#)。

刪除多區域金鑰 — 如同所有 KMS 金鑰，您必須先排程刪除多區域金鑰，才能將其 AWS KMS 刪除。當金鑰正在等待刪除時，您無法在任何密碼編譯操作中使用金鑰。不過，在刪除其所有複本金鑰之前，AWS KMS 不會刪除多區域主金鑰。如需詳細資訊，請參閱[Deleting multi-Region keys](#)。

匯入金鑰的 AWS KMS 金鑰材料

您可以使用您提供的金鑰材料來建立 AWS KMS keys (KMS 金鑰)。

KMS 金鑰是資料金鑰的邏輯表示法。KMS 金鑰的中繼資料包含用於執行密碼編譯操作的金鑰材料 ID。根據預設，當您[建立 KMS 金鑰](#)時，AWS KMS 會產生該 KMS 金鑰的金鑰材料。但是，您可以建立不含金鑰材料的 KMS 金鑰，然後將您自己的金鑰材料匯入該 KMS 金鑰。這項功能通常稱為「使用自有金鑰 (BYOK)」。



 Note

AWS KMS 不支援解密由 外部對稱加密 KMS AWS KMS 金鑰加密的任何加密文字 AWS KMS，即使加密文字是在具有匯入金鑰材料的 KMS 金鑰下加密。AWS KMS 不會發佈此任務所需的加密文字格式，而且格式可能會變更，恕不另行通知。

當您使用匯入的金鑰材料時，您仍需對金鑰材料負責，同時 AWS KMS 允許 使用其副本。您可能因為以下一個或多個原因而選擇這樣做：

- 為證明您採用符合要求的熵來源產生金鑰資料。
- 將自有基礎設施的金鑰材料與 AWS 服務搭配使用 AWS KMS，並使用 管理其中該金鑰材料的生命週期 AWS。
- 在 中使用現有、建立良好的金鑰 AWS KMS，例如用於程式碼簽署、PKI 憑證簽署和憑證鎖定應用程式的金鑰
- 若要設定 中金鑰材料的過期時間，AWS 並[手動將其刪除](#)，但也要讓它在未來再次可用。反之，[排程金鑰刪除](#)需要等候 7 到 30 天，超過此期間將無法恢復已刪除的 KMS 金鑰。
- 擁有金鑰材料的原始副本，並在金鑰材料的完整生命週期期間將其保留在 之外 AWS，以獲得額外的耐用性和災難復原。
- 對於非對稱金鑰和 HMAC 金鑰，匯入會建立相容且可互通的金鑰，可在 內外操作 AWS。

支援的 KMS 金鑰類型

AWS KMS 支援下列 KMS 金鑰類型的匯入金鑰材料。您無法匯入金鑰資料至[自訂金鑰存放區](#)的 KMS 金鑰。

- [對稱加密 KMS 金鑰](#)
- [非對稱 KMS 金鑰 \(ML-DSA 金鑰除外\)](#)
- [HMAC KMS 金鑰](#)
- 所有支援類型的[多區域金鑰](#)。

區域

支援的所有 都支援匯入 AWS 區域 的 AWS KMS 金鑰材料。

在中國區域中，對稱加密 KMS 金鑰的金鑰材料需求與其他區域不同。如需詳細資訊，請參閱[步驟 3：加密金鑰材料](#)。

進一步了解

- 若要使用匯入的金鑰材料建立 KMS 金鑰，請參閱 [使用匯入的金鑰材料建立 KMS 金鑰](#)。
- 若要建立警示，以便在 KMS 金鑰中匯入的金鑰材料接近其過期時間時通知您，請參閱 [為匯入金鑰材料過期建立 CloudWatch 警示](#)。
- 若要將金鑰材料重新匯入 KMS 金鑰，請參閱 [重新匯入金鑰材料](#)。
- 若要將新的金鑰材料匯入 KMS 金鑰以進行隨需輪換，請參閱 [匯入新的金鑰資料](#)和 [執行隨需金鑰輪換](#)。
- 若要使用匯入的金鑰材料來識別和檢視 KMS 金鑰，請參閱 [使用匯入的金鑰材料識別 KMS 金鑰](#)。
- 若要了解使用匯入金鑰材料刪除 KMS 金鑰的特殊考量，請參閱 [Deleting KMS keys with imported key material](#)。

匯入金鑰資料的特殊考量

在您決定將金鑰材料匯入至 之前 AWS KMS，您應該了解匯入金鑰材料的下列特性。

您可以產生關鍵材料

您有責任使用符合您安全要求的隨機來源產生金鑰材料。

您必須為可用性和耐久性負責

AWS KMS 旨在讓匯入的金鑰材料保持高可用性。但 AWS KMS 不會將匯入金鑰材料的耐久性維持在與 AWS KMS 產生的金鑰材料相同的層級。如需詳細資訊，請參閱[保護匯入的金鑰資料](#)。

您可以刪除金鑰材料

您可以從 KMS 金鑰中[刪除匯入的金鑰材料](#)，立即使 KMS 金鑰無法使用。此外，當您將金鑰材料匯入 KMS 金鑰時，您可以決定金鑰是否會過期，並[設定其過期時間](#)。當過期時間到達時，AWS KMS [會刪除金鑰材料](#)。如果沒有金鑰材料，即無法在任何密碼編譯操作中使用 KMS 金鑰。若要還原金鑰，您必須將相同的金鑰材料重新匯入至金鑰。

您無法變更非對稱的金鑰材料和 HMAC 金鑰

當您將金鑰材料匯入 KMS 金鑰，KMS 金鑰將永久關聯到該金鑰材料。您可以[重新匯入相同的金鑰材料](#)，但您不能將不同的金鑰材料匯入 KMS 金鑰。此外，您無法為具有匯入金鑰材料的任何 KMS 金鑰[啟用自動金鑰輪換](#)。不過，您可以[手動輪換具有匯入金鑰材料的 KMS 金鑰](#)。

您可以在對稱加密金鑰上執行隨需輪換

具有匯入金鑰材料的對稱加密金鑰支援隨需輪換。您可以將[多個金鑰材料匯入](#)這些金鑰，並使用[隨需輪換](#)來更新目前的金鑰材料。目前的金鑰材料同時用於加密和解密，但其他（非目前）金鑰材料只能用於解密。

您無法變更金鑰資料來源

設計用於匯入金鑰材料的 KMS 金鑰具有無法變更的 EXTERNAL [來源](#)值。您無法轉換匯入金鑰材料的 KMS 金鑰，以使用任何其他來源的金鑰材料，包括 AWS KMS。同樣地，您無法將 KMS 金鑰與 AWS KMS 金鑰材料轉換為專為匯入金鑰材料設計的金鑰材料。

您無法匯出金鑰資料

您無法匯出您匯入的任何金鑰材料。AWS KMS 無法以任何形式將匯入的金鑰材料傳回給您。您必須在 外部保留匯入金鑰材料的副本 AWS，最好是在金鑰管理器中，例如硬體安全模組 (HSM)，以便在刪除金鑰材料或金鑰材料過期時重新匯入金鑰材料。

您可利用匯入金鑰資料來建立多區域金鑰。

具匯入金鑰資料的多區域擁有匯入金鑰資料的 KMS 金鑰功能，且可在 AWS 區域之間互通。若要利用匯入金鑰資料來建立多區域金鑰，您必須匯入相同金鑰資料至主要 KMS 金鑰以及每個複本金鑰。如需匯入多區域金鑰之金鑰資料的詳細資訊，請參閱。

非對稱金鑰與 HMAC 金鑰為可攜式且可互通

您可以在 外部使用您的非對稱金鑰材料和 HMAC 金鑰材料 AWS，與具有相同匯入金鑰材料的 AWS KMS 金鑰互通。

與演算法中使用的 KMS 金鑰不可分割繫結的 AWS KMS 對稱加密文字不同，AWS KMS 會使用標準 HMAC 和非對稱格式進行加密、簽署和產生 MAC。因此，這些金鑰為可攜式，並且支援傳統委付金鑰案例。

當您的 KMS 金鑰已匯入金鑰材料時，您可以使用 外部匯入的金鑰材料 AWS 來執行下列操作。

- HMAC 金鑰 — 您可利用匯入金鑰資料來驗證 HMAC KMS 金鑰所產生的 HMAC 標籤。您也可以將 HMAC KMS 金鑰與匯入的金鑰材料搭配使用，以驗證由外部金鑰材料產生的 HMAC 標籤 AWS。
- 非對稱加密金鑰 — 您可以使用 外部的私有非對稱加密金鑰 AWS，以對應的公有金鑰解密 KMS 金鑰加密的加密文字。您也可以使用非對稱 KMS 金鑰來解密在 外部產生的非對稱加密文字 AWS。

- **非對稱簽署金鑰** — 您可以使用非對稱簽署 KMS 金鑰搭配匯入的金鑰材料，來驗證外部私有簽署金鑰所產生的數位簽章 AWS。您也可以使用 外部的非對稱公有簽署金鑰 AWS 來驗證非對稱 KMS 金鑰所產生的簽章。
- **非對稱金鑰協議金鑰** — 您可以使用非對稱金鑰協議 KMS 金鑰搭配匯入的金鑰材料，以衍生與外部對等的共用秘密 AWS。

如您匯入相同金鑰資料至位於相同 AWS 區域的不同 KMS 金鑰，則這些金鑰也可互通。若要在不同中建立可互通的 KMS 金鑰 AWS 區域，請使用匯入的金鑰材料建立多區域金鑰。

RSA 私有金鑰

- AWS KMS 要求匯入的 RSA 私有金鑰具有符合 [FIPS 186-5 第 A. 1.3 節](#)所述測試的主要因素。其他軟體或裝置可能會使用不同的演算法來驗證 RSA 私有金鑰的這些主要因素。在極少數情況下，可能無法接受使用其他演算法驗證的金鑰 AWS KMS。

對稱加密金鑰為不可攜式且不可互通

AWS KMS 產生的對稱加密文字不是可攜式或互通性。AWS KMS 不會發佈可攜性所需的對稱加密文字格式，而且格式可能會變更，恕不另行通知。

- AWS KMS 即使您使用已匯入的金鑰材料 AWS，也無法解密您在 外部加密的對稱加密文字。
- AWS KMS 不支援解密 AWS KMS 外部的任何對稱加密文字 AWS KMS，即使加密文字是在具有匯入金鑰材料的 KMS 金鑰下加密。
- 具相同匯入金鑰資料的 KMS 金鑰無法互通。AWS KMS 產生每個 KMS 金鑰特定加密文字的對稱加密文字。此密文格式可保證僅加密該資料的 KMS 金鑰可解密。

此外，您無法使用 [AWS Encryption SDK](#)或 [Amazon S3 用戶端加密](#)等任何 AWS 工具來解密 AWS KMS 對稱加密文字。

因此，您無法將金鑰與匯入的金鑰材料搭配使用，以支援金鑰託管安排，其中具有金鑰材料條件存取的授權第三方可以解密外部的某些加密文字 AWS KMS。若要支援金鑰委付，請使用 [AWS Encryption SDK](#) 在獨立於 AWS KMS 的金鑰下將您的訊息加密。

保護匯入的金鑰資料

您匯入的金鑰資料在傳輸過程及靜態狀態都受到保護。在匯入金鑰材料之前，您可以使用在 FIPS 140-3 密碼編譯模組驗證計畫下驗證的 AWS KMS 硬體安全模組 (HSMs) 中產生的 RSA 金鑰對公有金鑰來加密 (或「包裝」) 金鑰材料。 <https://csrc.nist.gov/projects/cryptographic-module-validation-program/certificate/4884> 您可利用包裝公有金鑰直接加密金鑰資料，或利用 AES 對稱金鑰來加密金鑰資料，然後利用 RSA 公有金鑰加密 AES 對稱金鑰。

收到時，會使用 AWS KMS HSM 中對應的私有金鑰 AWS KMS 來解密金鑰材料，並在僅存在於 HSM 揮發性記憶體中的 AES 對稱金鑰下重新加密。您的金鑰資料永遠不會以純文字形式離開 HSM。它只會在使用中且僅在 AWS KMS HSMs 內解密。

與匯入的金鑰資料搭配運用的 KMS 金鑰取決於您在 KMS 金鑰設定的[存取控制政策](#)。此外，您可利用[別名與標籤](#)來識別及[控制 KMS 金鑰的存取權](#)。您可以使用等服務[來啟用和停用](#)金鑰、[檢視](#)和[監控](#)金鑰 AWS CloudTrail。

然而，您要維護金鑰資料的唯一故障安全副本。為了獲得此額外控制，您必須負責匯入金鑰材料的耐用性和整體可用性。AWS KMS 旨在讓匯入金鑰材料保持高可用性。但 AWS KMS 不會將匯入金鑰材料的耐久性維持在與 AWS KMS 產生的金鑰材料相同的層級。

這種持久性差異在以下情況具有意義：

- 當您[為匯入的金鑰材料設定過期時間](#)時，會在金鑰材料過期後 AWS KMS 刪除金鑰材料。AWS KMS 不會刪除 KMS 金鑰或其中繼資料。您可以[建立 Amazon CloudWatch 警示](#)，在匯入的金鑰資料臨近其到期日期時通知您。

您無法刪除為 KMS 金鑰 AWS KMS 產生的金鑰材料，也無法將 AWS KMS 金鑰材料設定為過期。

- 當您[手動刪除匯入的金鑰材料](#)時，會 AWS KMS 刪除金鑰材料，但不會刪除 KMS 金鑰或其中繼資料。相反地，[排程金鑰刪除](#)需要 7 到 30 天的等待期間，之後會 AWS KMS 永久刪除 KMS 金鑰、其中繼資料及其金鑰材料。
- 萬一發生影響 AWS KMS（例如總停電）的特定全區域故障，AWS KMS 無法自動還原您匯入的金鑰材料。不過，AWS KMS 可以還原 KMS 金鑰及其中繼資料。

您必須將外部匯入金鑰材料的副本保留 AWS 在您控制的系統中。建議您將匯入金鑰資料的可匯出複本儲存在金鑰管理系統，例如 HSM。最佳實務是，您應該將 KMS 金鑰 ARN 和 AWS KMS 產生的金鑰材料 ID 的參考與可匯出的金鑰材料複本一起存放。如匯入的金鑰資料遭到刪除或到期，則在您重新匯入相同金鑰資料之前，其關聯的 KMS 金鑰將無法運用。如匯入的金鑰資料永久遺失，則利用 KMS 金鑰加密的任何密文均無法復原。

Important

對稱加密金鑰可以有多個與其相關聯的金鑰材料。一旦您刪除其中一個金鑰資料，或其中一個金鑰資料過期（除非已刪除或即將過期的金鑰資料為 PENDING_ROTATION 或），整個 KMS 金鑰就會變得無法使用 PENDING_MULTI_REGION_IMPORT_AND_ROTATION。您必須重新匯入與此類金鑰相關聯的任何過期或刪除的金鑰資料，金鑰才能用於密碼編譯操作。

CloudHSM 金鑰存放區中的 KMS 金鑰

您可以在 AWS CloudHSM 金鑰存放區 AWS KMS keys 中建立、檢視、管理、使用和排程刪除。您使用的處理程序非常類似於對其他 KMS 金鑰所用的程序。唯一的差別是您在建立 KMS AWS CloudHSM 金鑰時指定金鑰存放區。然後，在與金鑰存放區相關聯的叢集中 AWS CloudHSM，為 KMS 金鑰 AWS KMS 建立不可擷取的 AWS CloudHSM 金鑰材料。當您在金鑰存放區中使用 KMS AWS CloudHSM 金鑰時，[密碼編譯操作](#)會在叢集的 HSMs 中執行。

支援的功能

除了本節中討論的程序之外，您還可以使用金鑰存放區中的 KMS AWS CloudHSM 金鑰執行下列動作：

- 使用金鑰政策、IAM 政策和授予，以[授權存取](#) KMS 金鑰。
- [啟用和停用](#) KMS 金鑰。
- 指派[標籤](#)並建立[別名](#)，使用屬性型存取控制 (ABAC) 授權對 KMS 金鑰的存取。
- 使用 KMS 金鑰執行下列密碼編譯操作：
 - [加密](#)
 - [解密](#)
 - [GenerateDataKey](#)
 - [GenerateDataKeyWithoutPlaintext](#)
 - [ReEncrypt](#)

自訂金鑰存放區中不支援產生非對稱資料金鑰對、[GenerateDataKeyPair](#) 和 [GenerateDataKeyPairWithoutPlaintext](#) 的操作。

- 將 KMS 金鑰用於與 [AWS KMS 整合的 AWS 服務](#)，且這些服務支援客戶受管金鑰。
- 追蹤您的 KMS 金鑰在 [AWS CloudTrail 日誌](#) 和 [Amazon CloudWatch 監控工具](#) 中的使用情況。

不支援的功能

- AWS CloudHSM 金鑰存放區僅支援對稱加密 KMS 金鑰。您無法在金鑰 AWS CloudHSM 存放區中建立 HMAC KMS 金鑰、非對稱 KMS 金鑰或非對稱資料金鑰對。
- 您無法將[金鑰材料匯入](#)金鑰存放區中的 KMS AWS CloudHSM 金鑰。AWS KMS 會產生叢集中 AWS CloudHSM KMS 金鑰的金鑰材料。
- 您無法啟用或停用金鑰存放區中 KMS 金鑰的 AWS CloudHSM 金鑰材料[自動輪換](#)。

在 金鑰存放區中使用 KMS AWS CloudHSM 金鑰

當您在請求中使用 KMS 金鑰時，請依其 ID 或別名識別 KMS 金鑰；您不需要指定 AWS CloudHSM 金鑰存放區或 AWS CloudHSM 叢集。回應包含為針對任何對稱加密 KMS 金鑰所傳回的相同欄位。

不過，當您在 AWS CloudHSM 金鑰存放區中使用 KMS 金鑰時，密碼編譯操作會完全在與 AWS CloudHSM 金鑰存放區相關聯的 AWS CloudHSM 叢集內執行。操作會使用叢集內與您選擇的 KMS 金鑰相關聯的金鑰材料。

但必須符合以下條件，才有可能這樣做。

- KMS 金鑰的[金鑰狀態](#)必須是 Enabled。若要尋找金鑰狀態，請使用 [AWS KMS 主控台](#) 中的 Status (狀態) 欄位或 [DescribeKey](#) 回應中的 KeyState 欄位。
- AWS CloudHSM 金鑰存放區必須連接到其 AWS CloudHSM 叢集。其 [AWS KMS 主控台](#) 中的 Status (狀態) 或 [DescribeCustomKeyStores](#) 回應中的 ConnectionState 必須為 CONNECTED。
- 與自訂金鑰存放區相關聯的 AWS CloudHSM 叢集必須至少包含一個作用中的 HSM。若要尋找叢集中的作用中 HSMs 數量，請使用 [AWS KMS 主控台](#)、AWS CloudHSM 主控台或 [DescribeClusters](#) 操作。
- AWS CloudHSM 叢集必須包含 KMS 金鑰的金鑰材料。如果已從叢集刪除金鑰資料，或 HSM 是從不含金鑰資料的備份所建立，則密碼編譯操作會失敗。

如果不符合這些條件，密碼編譯操作會失敗，並 AWS KMS 傳回 KMSInvalidStateException 例外狀況。一般而言，您只需要[重新連接 AWS CloudHSM 金鑰存放區](#)。如需其他說明，請參閱[如何修正失效的 KMS 金鑰](#)。

在 AWS CloudHSM 金鑰存放區中使用 KMS 金鑰時，請注意每個 AWS CloudHSM 金鑰存放區中的 KMS 金鑰共用[密碼編譯操作的自訂金鑰存放區請求配額](#)。如果您超過配額，會 AWS KMS 傳回 ThrottlingException。如果與 AWS CloudHSM 金鑰存放區相關聯的 AWS CloudHSM 叢集正在處理許多命令，包括與 AWS CloudHSM 金鑰存放區無關的命令，您可能會 ThrottlingException 以更低的速率取得。如果您收到任何請求的 ThrottlingException，請降低請求速率，然後再試一次命令。如需自訂金鑰存放區配額的詳細資訊，請參閱 [自訂金鑰存放區請求配額](#)。

進一步了解

- 若要進一步了解 AWS CloudHSM 金鑰存放區，請參閱 [AWS CloudHSM 金鑰存放區](#)。
- 若要在 金鑰存放區中建立 KMS AWS CloudHSM 金鑰，請參閱 [在 金鑰存放區中建立 KMS AWS CloudHSM 金鑰](#)。

- 若要識別和檢視 金鑰存放區中的 KMS AWS CloudHSM 金鑰，請參閱 [識別金鑰存放區中的 KMS AWS CloudHSM 金鑰](#)。
- 若要在 金鑰存放區中尋找 KMS 金鑰和 AWS CloudHSM 金鑰材料，請參閱 [在 金鑰存放區中尋找 KMS 金鑰和 AWS CloudHSM 金鑰材料](#)。
- 若要了解刪除 AWS CloudHSM 金鑰存放區中 KMS 金鑰的特殊考量，請參閱[從 AWS CloudHSM 金鑰存放區刪除 KMS 金鑰](#)。

外部金鑰存放區中的 KMS 金鑰

若要在外部金鑰存放區中建立、檢視、管理、使用和排程 KMS 金鑰的刪除，您可以使用與用於其他 KMS 金鑰非常相似的程序。但是，當您在外部金鑰存放區中建立 KMS 金鑰時，您會指定[外部金鑰存放區](#)和[外部金鑰](#)。當您在外部金鑰存放區中使用 KMS 金鑰時，外部金鑰管理器將使用指定的外部金鑰執行[加密和解密操作](#)。

AWS KMS 無法在您的外部金鑰管理器中建立、檢視、更新或刪除任何密碼編譯金鑰。AWS KMS 絕不會直接存取您的外部金鑰管理器或任何外部金鑰。所有密碼編譯操作的請求都會由您的[外部金鑰存放區代理](#)進行協調。若要在外部金鑰存放區中使用 KMS 金鑰，託管 KMS 金鑰的外部金鑰存放區必須[連接](#)至其外部金鑰存放區代理。

支援的功能

除了本節所討論的程序，您還可以對外部金鑰存放區中的 KMS 金鑰執行下列操作：

- 使用[金鑰政策](#)、[IAM 政策](#)和[授予](#)，以控制對 KMS 金鑰的存取。
- [啟用和停用](#) KMS 金鑰。這些動作不會影響外部金鑰管理器中的外部金鑰。
- 指派[標籤](#)並建立[別名](#)，使用[屬性型存取控制](#) (ABAC) 授權對 KMS 金鑰的存取。
- 使用 KMS 金鑰執行下列密碼編譯操作：
 - [加密](#)
 - [解密](#)
 - [GenerateDataKey](#)
 - [GenerateDataKeyWithoutPlaintext](#)
 - [ReEncrypt](#)

自訂金鑰存放區中不支援產生非對稱資料金鑰對、[GenerateDataKeyPair](#) 和 [GenerateDataKeyPairWithoutPlaintext](#) 的操作。

- 將 KMS 金鑰用於[與 AWS KMS整合的AWS 服務](#)，且這些服務支援[客戶受管金鑰](#)。

不支援的功能

- 外部金鑰存放區僅支援[對稱加密 KMS 金鑰](#)。您無法在外部金鑰存放區中建立 HMAC KMS 金鑰或非對稱 KMS 金鑰。
- 外部金鑰存放區中的 KMS 金鑰不支援 [GenerateDataKeyPair](#) 和 [GenerateDataKeyPairWithoutPlaintext](#)。
- 您無法使用 [AWS::KMS::Key CloudFormation 範本](#) 在外部金鑰存放區中建立外部金鑰存放區或 KMS 金鑰。
- 外部金鑰存放區不支援[多區域金鑰](#)。
- 外部金鑰存放區不支援具有[匯入金鑰材料](#)的 KMS 金鑰。
- 外部金鑰存放區中的 KMS 金鑰不支援[自動金鑰輪換](#)。

在外部金鑰存放區中使用 KMS 金鑰

在請求中使用 KMS 金鑰時，請按照其[金鑰 ID、金鑰 ARN、別名或別名 ARN](#) 來識別 KMS 金鑰。您不需要指定外部金鑰存放區。回應包含為針對任何對稱加密 KMS 金鑰所傳回的相同欄位。不過，當您在外部金鑰存放區中使用 KMS 金鑰時，外部金鑰管理器將使用與 KMS 金鑰關聯的外部金鑰來執行加密和解密操作。

為了確保外部金鑰存放區中 KMS 金鑰加密的加密文字至少與標準 KMS 金鑰加密的任何加密文字一樣安全，AWS KMS 會使用[雙重加密](#)。資料會先 AWS KMS 使用 AWS KMS 金鑰材料在 中加密。然後，您的外部金鑰管理器會使用 KMS 金鑰的外部金鑰對其進行加密。若要解密雙重加密的密文，您的外部金鑰管理器會先使用 KMS 金鑰的外部金鑰對密文進行解密。然後，它會 AWS KMS 使用 KMS AWS KMS 金鑰的金鑰材料在 中解密。

但必須符合以下條件，才有可能這樣做。

- KMS 金鑰的[金鑰狀態](#)必須是 Enabled。若要尋找金鑰狀態，請參閱 [AWS KMS 主控台](#) 中客戶受管金鑰的 Status (狀態) 欄位或 [DescribeKey](#) 回應中的 KeyState 欄位。
- 託管 KMS 金鑰的外部金鑰存放區必須連接至其[外部金鑰存放區代理](#)，也就是說，外部金鑰存放區的[連接狀態](#)必須為 CONNECTED。

您可以在 AWS KMS 主控台的外部金鑰存放區頁面或 [DescribeCustomKeyStores](#) 回應中檢視連線狀態。外部金鑰存放區的連接狀態也顯示在 AWS KMS 主控台 KMS 金鑰的詳細資訊頁面上。在詳細資訊頁面上，選擇 Cryptographic configuration (密碼編譯組態) 索引標籤，並查看 Custom key store (自訂金鑰存放區) 區段中的 Connection state (連接狀態) 欄位。

如果連接狀態為 DISCONNECTED，則必須先將其連接。如果連接狀態為 FAILED，則您必須解決問題，中斷連接外部金鑰存放區，然後進行連接。如需說明，請參閱[連接和中斷連接外部金鑰存放區](#)。

- 外部金鑰存放區代理必須能夠找到外部金鑰。
- 必須啟用外部金鑰，而且必須執行加密和解密。

外部金鑰的狀態獨立於 KMS 金鑰的[金鑰狀態變更](#)，並不受其影響，包括啟用和停用 KMS 金鑰。同樣，停用或刪除外部金鑰不會變更 KMS 金鑰的金鑰狀態，但使用關聯 KMS 金鑰的密碼編譯操作將會失敗。

如果不符合這些條件，密碼編譯操作會失敗，並 AWS KMS 傳回 `KMSInvalidStateException` 例外狀況。您可能需要[重新連接外部金鑰存放區](#)，或使用外部金鑰管理器工具來重新設定或修復外部金鑰。如需其他說明，請參閱[the section called “外部金鑰存放區故障診斷”](#)。

在外部金鑰存放區中使用 KMS 金鑰時，請注意每個外部金鑰存放區中的 KMS 金鑰會針對密碼編譯操作共用[自訂金鑰存放區請求配額](#)。如果您超過配額，會 AWS KMS 傳回 `ThrottlingException`。如需自訂金鑰存放區配額的詳細資訊，請參閱[自訂金鑰存放區請求配額](#)。

進一步了解

- 若要進一步了解外部金鑰存放區，請參閱[外部金鑰存放區](#)。
- 若要進一步了解外部金鑰存放區中的金鑰材料，請參閱[外部金鑰](#)。
- 若要在外部金鑰存放區中建立 KMS 金鑰，請參閱[在外部金鑰存放區中建立 KMS 金鑰](#)。
- 若要識別和檢視外部金鑰存放區中的 KMS 金鑰，請參閱[識別外部金鑰存放區中的 KMS 金鑰](#)。
- 若要了解在外部金鑰存放區中刪除 KMS 金鑰的特殊考量，請參閱[從外部金鑰存放區刪除 KMS 金鑰](#)。

AWS KMS 密碼編譯基本概念

AWS KMS 使用可設定的密碼編譯演算法，讓系統可以快速從一個核准的演算法或模式遷移到另一個演算法。針對安全屬性和效能，已從聯邦資訊處理標準 (經 FIPS 核准) 演算法中選取初始預設的密碼編譯演算法集。

如需偏好和可接受的密碼編譯演算法的詳細資訊，請參閱[支援的密碼編譯演算法](#)。

熵和隨機數字產生

AWS KMS 金鑰產生會在 AWS KMS HSMs 中執行。HSM 會實作混合式隨機數字產生器，其使用[採用 AES-256 的 NIST SP800-90A 決定性隨機位元產生器](#)。它植入了具有 384 位元熵的非決定性隨機位元產生器，並使用額外的熵進行更新，以便在每次呼叫加密材料時提供預測阻力。

對稱金鑰操作 (僅限加密)

HSM 中使用的所有對稱金鑰加密命令都會使用[進階加密標準 \(AES\)](#)，採用 256 位元金鑰的 [Galois 計數器模式 \(GCM\)](#)。要解密的類似呼叫使用反函數。

AES-GCM 是經過驗證的加密配置。除了加密純文字以產生加密文字，它還會針對加密文字和需要身分驗證的任何其他資料 (另外驗證的資料或 AAD) 運算身分驗證標籤。身分驗證標籤有助於確保資料來自所宣稱的來源，而且加密文字和 AAD 尚未修改。

通常，AWS 會省略在我們的描述中包含 AAD，特別是在參考資料金鑰加密時。這些情況下的周圍文字會暗示，在要加密純文字和要保護純文字 AAD 之間會進行分區的要加密結構。

AWS KMS 提供您將金鑰材料匯入的選項，AWS KMS key 而不是依賴 AWS KMS 產生金鑰材料。此匯入的金鑰材料可以使用 [RSAES-OAEP](#) 加密，以在傳輸到 AWS KMS HSM 期間保護金鑰。RSA 金鑰對會在 AWS KMS HSMs 上產生。匯入的金鑰材料會在 AWS KMS HSM 上解密，並在服務儲存之前在 AES-GCM 下重新加密。

非對稱金鑰操作 (加密、數位簽章和簽章驗證)

AWS KMS 支援將非對稱金鑰操作用於加密、數位簽章和金鑰協議操作。非對稱金鑰操作依賴數學上相關的公有金鑰和私有金鑰對，可用於加密和解密、簽署和簽章驗證，或衍生共用秘密。私有金鑰永遠不會讓 AWS KMS 處於未加密狀態。您可以呼叫 AWS KMS API 操作 AWS KMS，在中使用公有金鑰，或下載公有金鑰並在外部使用 AWS KMS。

AWS KMS 支援下列非對稱密碼。

- RSA-OAEP (用於加密) 與 RSA-PSS 和 RSA-PKCS-#1-v1_5 (用於簽署和驗證) – 針對不同的安全要求，支援 RSA 金鑰長度 (以位元為單位)：2048、3072 和 4096。
- 橢圓曲線 (ECC) – 用於簽署和驗證或衍生共用秘密，但不能同時用於兩者。支援 ECC 曲線：NIST P256, P384, P521、SECP 256k1、Ed25519。
- ML-DSA – 用於簽署和驗證。支援的 ML-DSA 金鑰規格為：ML_DSA_44、ML_DSA_65 和 ML_DSA_87。
- SM2 (僅限中國區域) – 用於加密和解密、簽署和驗證，或衍生共用秘密，但您必須選擇一個金鑰用量。支援用於加密的 SM2PKE 和用於簽署的 SM2DSA。

金鑰衍生函數

金鑰衍生函數用於從初始秘密或金鑰衍生其他金鑰。AWS KMS 使用金鑰衍生函數 (KDF) 來衍生下每個加密的每個呼叫金鑰 AWS KMS key。所有 KDF 操作均會使用 [計數器模式中的 KDF](#) (使用 HMAC

[[FIPS197](#)] 與 SHA256 [[FIPS180](#)]。256 位元衍生金鑰與 AES-GCM 搭配使用，以加密或解密客戶資料和金鑰。

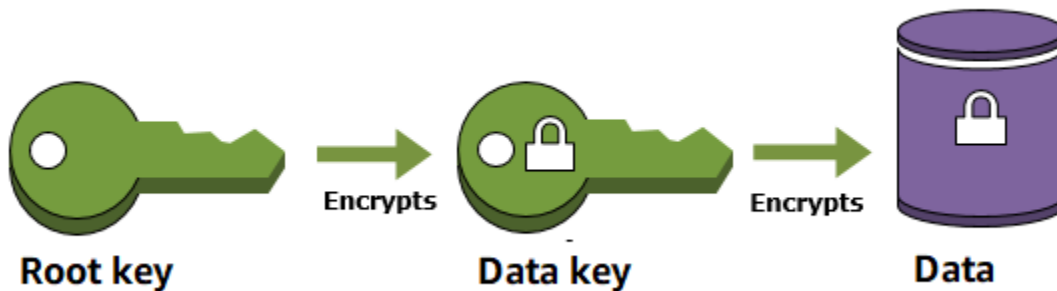
AWS KMS 內部使用數位簽章

數位簽章也可用來對命令和 AWS KMS 實體之間的通訊進行身分驗證。所有服務實體都有橢圓曲線數位簽章演算法 (ECDSA) 金鑰對。其執行[密碼編譯訊息語法 \(CMS\) 中橢圓曲線密碼編譯 \(ECC\) 演算法的使用](#)和 X9.62-2005：金融服務業的公有金鑰密碼編譯：橢圓曲線數位簽章演算法 (ECDSA) 中所定義的 ECDSA。實體使用[聯邦資訊處理標準出版物 FIPS PUB 180-4 \(稱為 SHA384\)](#) 中所定義的安全雜湊演算法。金鑰於曲線 secp384r1 (NIST-P384) 產生。

封套加密

加密資料後，您的資料會受到保護，但您需要保護您的加密金鑰。一個策略是把它加密。封套加密是使用資料金鑰來加密純文字資料，然後再透過另一個金鑰來加密資料金鑰的實務做法。

您甚至可以透過另一個加密金鑰來加密資料加密金鑰，並透過另一個加密金鑰來加密該加密金鑰。但是，最終必須有一個金鑰保留在純文字中，以便您可以解密金鑰和您的資料。這個最上層的純文字金鑰加密金鑰稱為根金鑰。



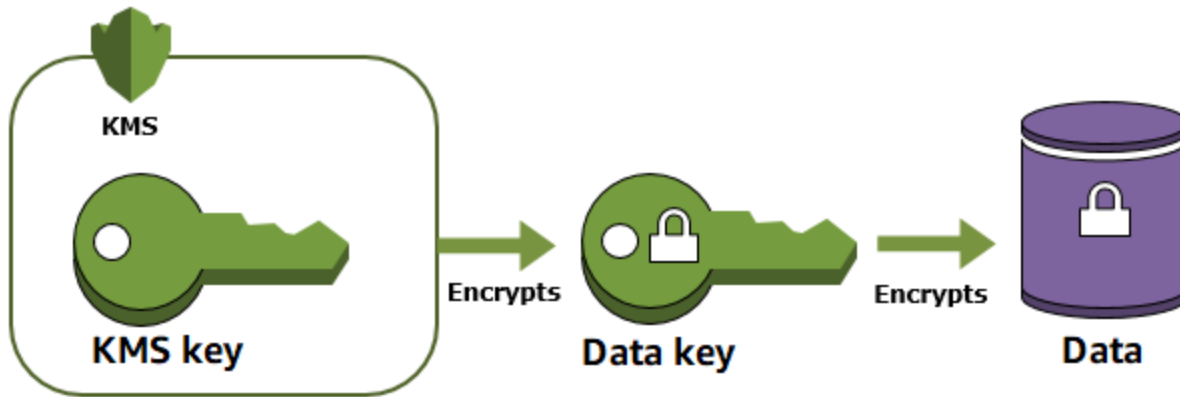
AWS KMS 可協助您安全地存放和管理加密金鑰，以保護您的加密金鑰。存放在 中的根金鑰 AWS KMS，稱為 AWS KMS keys，永遠不會讓 AWS KMS [FIPS 140-3 安全層級 3 驗證的硬體安全模組](#)處於未加密狀態。若要使用 KMS 金鑰，您必須呼叫 AWS KMS。

在許多密碼編譯系統中使用的基本結構是信封加密。信封加密使用兩個或多個密碼編譯金鑰來保護訊息。通常，一個金鑰是從較長期的靜態金鑰 k ，另一個金鑰是每條訊息金鑰 $msgKey$ (這是為了加密訊息而產生的)。信封是透過加密訊息： $ciphertext = Encrypt(msgKey, message)$ 形成的。然後訊息金鑰使用長期靜態金鑰： $encKey = Encrypt(k, msgKey)$ 進行加密。最後，將這兩個值 ($encKey, ciphertext$) 封裝成單一結構或信封加密訊息。

收件人 (可存取 k) 可以先解密加密的金鑰，然後解密訊息，以開啟封住的訊息。

AWS KMS 可讓您管理這些長期靜態金鑰，並自動化資料的信封加密程序。

除了 AWS KMS 服務內提供的加密功能之外，[AWS Encryption SDK](#) 還提供用戶端信封加密程式庫。您可以使用這些程式庫來保護您的資料和用來加密該資料的加密金鑰。



封套加密提供多種優勢：

- 保護資料金鑰

加密資料金鑰時，您不需要擔心如何存放加密的資料金鑰，因為這份資料金鑰本質上已獲得加密保護。加密的資料金鑰可以安全地跟加密資料一起存放。

- 在多個金鑰下加密相同的資料

加密操作可能極為耗時，特別是要加密的資料屬於大型物件時，更為明顯。這時您可以捨棄使用不同金鑰來多次重新加密原始資料的做法，改成只重新加密負責保護原始資料的資料金鑰。

- 結合多種演算法的優勢

一般而言，比起公開金鑰演算法，對稱金鑰演算法速度較快，產生的加密文字較小。但是，公開金鑰演算法本質上就會區隔角色，因此金鑰管理較為方便。封套加密可讓您結合每個策略的優點。

密碼編譯操作

在中 AWS KMS，密碼編譯操作是使用 KMS 金鑰保護資料的 API 操作。由於 KMS 金鑰保留在內 AWS KMS，您必須呼叫 AWS KMS 在密碼編譯操作中使用 KMS 金鑰。

若要使用 KMS 金鑰執行密碼編譯操作，請使用 AWS SDKs、AWS Command Line Interface (AWS CLI) 或 AWS Tools for PowerShell。您無法在 AWS KMS 主控台中執行密碼編譯操作。如需以數種程式設計語言呼叫密碼編譯操作的範例，請參閱[AWS KMS 使用 AWS SDKs 程式碼範例](#)。

下表列出 AWS KMS 密碼編譯操作。它也會顯示操作中使用之 KMS 金鑰的金鑰類型和[金鑰使用情形](#)需求。

作業	Key type	金鑰用途
解密	對稱或不對稱	ENCRYPT_DECRYPT
DeriveSharedSecret	非對稱	KEY_AGREEMENT
加密	對稱或不對稱	ENCRYPT_DECRYPT
GenerateDataKey	對稱	ENCRYPT_DECRYPT
GenerateDataKeyPair	對稱 [1] 不支援自訂金鑰存放區中的 KMS 金鑰。	ENCRYPT_DECRYPT
GenerateDataKeyPairWithoutPlaintext	對稱 [1] 不支援自訂金鑰存放區中的 KMS 金鑰。	ENCRYPT_DECRYPT
GenerateDataKeyWithoutPlaintext	對稱	ENCRYPT_DECRYPT
GenerateMac	HMAC	GENERATE_VERIFY_MAC
GenerateRandom	N/A。此操作不會使用 KMS 金鑰。	N/A
ReEncrypt	對稱或不對稱	ENCRYPT_DECRYPT
符號	非對稱	SIGN_VERIFY
確認	非對稱	SIGN_VERIFY
VerifyMac	HMAC	GENERATE_VERIFY_MAC

[1] 產生受對稱加密 KMS 金鑰保護的非對稱資料金鑰對。

如需密碼編譯操作許可的詳細資訊，請參閱[the section called “許可參考”](#)。

為了讓所有使用者 AWS KMS 都能回應並發揮高功能，會在每秒呼叫的密碼編譯操作數量上 AWS KMS 建立配額。如需詳細資訊，請參閱[the section called “密碼編譯操作的共用配額”](#)。

支援的密碼編譯演算法

密碼編譯演算法

下表摘要說明跨其服務 AWS 部署以保護您的資料的密碼編譯演算法、密碼、模式和金鑰大小。它們不應被視為 中所有可用密碼編譯選項的詳盡清單 AWS。演算法分為兩個類別：

- 偏好的演算法符合 AWS 安全性和效能標準。
- 可接受的演算法可用於某些應用程式的相容性，但不建議使用。

非對稱密碼編譯

下表列出加密、金鑰協議和數位簽章支援的非對稱演算法。

Type	演算法	狀態
加密	RSA-OAEP (2048 或 3072 位元模數)	可接受
加密	HPKE (P-256 或 P-384、HKF 和 AES-GCM)	可接受
金鑰協議	ML-KEM-768 或 ML-KEM-1024	偏好 (抗量子)
金鑰協議	ECDH(E) 搭配 P-384	可接受
金鑰協議	ECDH(E) 搭配 P-256, P-521 或 X25519	可接受

Type	演算法	狀態
金鑰協議	ECDH(E) 搭配 brainpool P256r1, brainpoolP384r1 或 brainpoolP512r1	可接受
Signatures (簽章)	ML-DSA-65 或 ML-DSA-87	偏好 (抗量子)
Signatures (簽章)	SLH-DSA	偏好 (抗量子軟體/韌體簽署)
Signatures (簽章)	ECDSA 搭配 P-384	可接受
Signatures (簽章)	ECDSA 搭配 P-256, P-521 或 Ed25519	可接受
Signatures (簽章)	RSA-2048 或 RSA-3072	可接受

對稱密碼編譯

下表列出加密、已驗證加密和金鑰包裝支援的對稱演算法。

Type	演算法	狀態
驗證加密	AES-GCM-256	偏好
驗證加密	AES-GCM-128	可接受
驗證加密	ChaCha20/Poly1305	可接受
加密模式	AES-XTS-256 (用於區塊儲存)	偏好
加密模式	AES-CBC/CTR (未驗證模式)	可接受
金鑰包裝	AES-GCM-256	偏好
金鑰包裝	具有 256 位元金鑰的 AES-KW 或 AES-KWP	可接受

密碼編譯函數

下表列出雜湊、金鑰衍生、訊息身分驗證和密碼雜湊支援的演算法。

Type	演算法	狀態
雜湊	SHA2-384	偏好
雜湊	SHA2-256	可接受
雜湊	SHA3	可接受
金鑰衍生	HKDF_Expand 或 HKDF 搭配 SHA2-256	偏好
金鑰衍生	使用 HMAC-SHA2-256 的計數器模式 KDF	可接受
訊息驗證碼	HMAC-SHA2-384	偏好
訊息驗證碼	HMAC-SHA2-256	可接受
訊息驗證碼	KMAC	可接受
密碼雜湊	使用 SHA384 加密	偏好
密碼雜湊	PBKDF2	可接受

如需部署在 中的密碼編譯演算法詳細資訊 AWS，請參閱[密碼編譯演算法和 AWS 服務](#)。

KMS 金鑰存取和許可

若要使用 AWS KMS，您必須擁有 AWS 可用於驗證請求的登入資料。登入資料必須包含存取 AWS 資源 AWS KMS keys 和**[別名的許可](#)**。除非明確提供該許可且從未拒絕，否則任何 AWS 委託人都沒有任何 KMS 金鑰的許可。沒有能使用或管理 KMS 金鑰的隱含或自動許可。

若要控制對 KMS 金鑰的存取，您可以使用以下政策機制。

- **[金鑰政策](#)** – 每個 KMS 金鑰都有一個金鑰政策。它是控制對 KMS 金鑰之存取的主要機制。您可以單獨使用金鑰政策來控制存取，這表示在單一文件 (金鑰政策) 中定義 KMS 金鑰的完整存取範圍。如需使用金鑰政策的詳細資訊，請參閱**[金鑰政策](#)**。
- **[IAM 政策](#)** – 您可以使用 IAM 政策搭配金鑰政策和授權，來控制對 KMS 金鑰的存取。透過這種方式控制存取，可讓您管理 IAM 中 IAM 身分的所有許可。若要使用 IAM 政策來允許存取 KMS 金鑰，金鑰政策必須明確允許。如需使用 IAM 政策的詳細資訊，請參閱**[IAM 政策](#)**。
- **[授權](#)** – 您可以使用授權搭配金鑰政策和 IAM 政策，來允許對 KMS 金鑰的存取。透過這種方式控制存取，可讓您在金鑰政策中允許存取 KMS 金鑰，並且允許身分將其存取權委派給其他人。如需使用授權的詳細資訊，請參閱**[在中授予 AWS KMS](#)**。

KMS 金鑰政策

管理 AWS KMS 資源存取權的主要方法是使用 政策。政策是描述哪些主體可以存取哪些資源的文件。連接至 IAM 身分的政策稱為以身分為基礎的政策 (或 IAM 政策)，而連接至其他資源類型的政策稱為資源政策。KMS 金鑰的資源 AWS KMS 政策稱為金鑰政策。

所有 KMS 金鑰都擁有金鑰政策。如果您未提供，會為您 AWS KMS 建立一個。AWS KMS 使用**[的預設金鑰政策](#)**會因您在 AWS KMS 主控台中建立金鑰或使用 AWS KMS API 而有所不同。我們建議您編輯預設金鑰政策，以符合組織對**[最低權限許可](#)**的要求。

如果金鑰和 IAM 主體位於相同 AWS 帳戶中，您可以單獨使用金鑰政策來控制存取，這表示 KMS 金鑰的完整存取範圍是在單一文件中定義 (金鑰政策)。不過，當一個帳戶中的發起人必須存取不同帳戶中的金鑰時，您無法單獨使用金鑰政策來授予存取權。在跨帳戶案例中，IAM 政策必須連接到發起人的使用者或角色，明確允許發起人進行 API 呼叫。

您也可以將 IAM 政策與金鑰政策和授權結合使用，以控制對 KMS 金鑰的存取。若要使用 IAM 政策來控制對 KMS 金鑰的存取，金鑰政策必須授予帳戶使用 IAM 政策的許可。您可以指定**[啟用 IAM 政策的金鑰政策陳述](#)**式，也可以在金鑰政策中明確**[指定允許的主體](#)**。

撰寫政策時，請確定您擁有強式控制，限制誰可以執行下列動作：

- 更新、建立和刪除 IAM 和 KMS 金鑰政策
- 從使用者、角色和群組連接和分離 IAM 政策
- 從 KMS 金鑰連接和分離 KMS 金鑰政策

KMS 金鑰授權

除了 IAM 和金鑰政策之外，AWS KMS 還支援[授予](#)。授予提供靈活且強大的方法來委派許可。您可以使用授予，對 AWS 帳戶中的 IAM 主體或其他 AWS 帳戶中的 IAM 主體發出有時間限制的 KMS 金鑰存取。如果您在建立政策時不知道委託人的名稱，或者需要存取的委託人經常變更，我們建議您發出有時間限制的存取。[承授者主體](#)可以位於與 KMS 金鑰相同的帳戶中，也可以位於不同的帳戶中。如果委託人和 KMS 金鑰位於不同的帳戶中，則除了授予之外，您還必須指定 IAM 政策。授予需要額外的管理，因為您必須呼叫 API 來建立授予，並在不再需要授予時淘汰或撤銷授予。

中的金鑰政策 AWS KMS

金鑰政策是 的資源政策 AWS KMS key。金鑰政策是控制對 KMS 金鑰之存取的主要方式。每個 KMS 金鑰都必須只有一個金鑰政策。金鑰政策中的陳述式決定誰有使用 KMS 金鑰的許可以及可以使用它的方式。您也可以使用 [IAM 政策](#)和[授予](#)來控制對 KMS 金鑰的存取，但每個 KMS 金鑰都必須有金鑰政策。

除非在金鑰政策、IAM 政策或授予中明確允許且從未拒絕 KMS 金鑰，否則任何 AWS 委託人，包括帳戶根使用者或金鑰建立者，都沒有任何許可。

除非金鑰政策明確允許，否則您不能使用 IAM 政策來允許存取 KMS 金鑰。如果沒有金鑰政策的許可，允許許可的 IAM 政策將不起作用。(您可以使用 IAM 政策，在未經金鑰政策許可的情況下拒絕對 KMS 金鑰的許可。) 預設金鑰政策會啟用 IAM 政策。若要在金鑰政策中啟用 IAM 政策，請新增 [允許存取 AWS 帳戶 並啟用 IAM 政策](#) 中所述的政策陳述式。

與全域 IAM 政策不同，金鑰政策是區域性的。金鑰政策僅控制對同一區域中 KMS 金鑰的存取。它對其他區域中的 KMS 金鑰沒有影響。

主題

- [建立金鑰政策](#)
- [預設金鑰政策](#)
- [檢視金鑰政策](#)

- [變更金鑰政策](#)
- [金鑰政策中 AWS 服務的許可](#)

建立金鑰政策

您可以在 AWS KMS 主控台或使用 AWS KMS API 操作建立和管理金鑰政策，例如 [CreateKey](#)、[ReplicateKey](#) 和 [PutKeyPolicy](#)。

當您在 AWS KMS 主控台中建立 KMS 金鑰時，主控台會逐步解說根據[主控台的預設金鑰政策建立金鑰政策](#)的步驟。使用 [CreateKey](#) 或 [ReplicateKey](#) API 時，若不指定金鑰政策，這些 API 便會[為以程式設計方式建立之金鑰，套用預設金鑰政策](#)。當您採用 [PutKeyPolicy](#) API 時，必須指定金鑰政策。

每個政策文件可以擁有一個或多個政策陳述式。以下範例顯示具有一個政策陳述式的有效金鑰政策文件。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DescribePolicyStatement",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:user/Alice"
      },
      "Action": "kms:DescribeKey",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:KeySpec": "SYMMETRIC_DEFAULT"
        }
      }
    }
  ]
}
```

主題

- [金鑰政策格式](#)

- [金鑰政策中的元素](#)
- [範例金鑰政策](#)

金鑰政策格式

金鑰政策文件必須符合以下規則：

- 最大為 32 KB (32,768 個位元組)
- 金鑰政策陳述式中的 Sid 元素可包含空格。(IAM 政策文件的 Sid 元素中禁止使用空格。)

金鑰政策文件僅可包含以下字元：

- 可印出的 ASCII 字元
- 基本拉丁字母和拉丁字母補充-1 字元集中的可印出字元
- Tab (\u0009)、換行字元 (\u000A) 和歸位字元 (\u000D) 特殊字元

金鑰政策中的元素

金鑰政策文件必須擁有下列元素：

版本

指定金鑰政策文件版本。將版本設定為 2012-10-17 (最新版本)。

陳述式

包含政策陳述式。金鑰政策文件必須至少包含一個陳述式。

每個金鑰政策陳述式最多可包含六個元素。需要 Effect、Principal、Action 和 Resource 元素。

Sid

(選用) 陳述式識別符 (Sid) 為可用來描述陳述式的任意字串。金鑰政策中的 Sid 可包含空格。(不得在 IAM 政策 Sid 元素中包含空格。)

Effect

(必要) 決定是允許還是拒絕政策陳述式中的許可。有效值為 Allow 或 Deny。如果您未明確允許存取 KMS 金鑰，將會隱含拒絕存取。您也可以明確拒絕存取 KMS 金鑰。您可以這樣做以確保使用者無法存取 CMK，即使其他政策允許存取。

Principal

(必要) [主體](#)是取得政策陳述式中指定許可的身分。您可以在金鑰政策中將 AWS 帳戶 IAM 使用者、IAM 角色和部分 AWS 服務指定為主體。IAM [使用者群組](#)在任何政策類型中都不是有效的主體。

星號值，例如 "AWS": "*" 代表所有帳戶的所有 AWS 身份。

Important

除非採用[條件](#)來限制金鑰政策，否則請勿在任何允許許可的金鑰政策陳述式將主體設為星號 (*)。除非另一個政策陳述式明確拒絕 KMS 金鑰，否則星號會提供每個 AWS 帳戶許可中的每個身分。其他 中的使用者只要在自己的帳戶中擁有對應的許可，AWS 帳戶就可以使用您的 KMS 金鑰。

Note

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的安全性最佳實務](#)。

若金鑰政策陳述式中的主體是以 `arn:aws:iam::111122223333:root` 表達的 [AWS 帳戶主體](#)，則政策陳述式不會向任何 IAM 主體授予許可。而是提供使用 IAM 政策的 AWS 帳戶許可，以委派金鑰政策中指定的許可。(雖然帳戶識別符中使用了「根」，但採用 `arn:aws:iam::111122223333:root` 格式的主體不代表[AWS 帳戶根使用者](#)。然而，帳戶主體代表帳戶及其管理員，包括帳戶根用戶。)

當委託人是另一個 AWS 帳戶 或其委託人時，只有在 區域中使用 KMS 金鑰和金鑰政策啟用帳戶時，許可才有效。如需相關資訊了解哪些區域預設為未啟用 (「選擇加入區域」)，請參閱《AWS 一般參考》 [的管理 AWS 區域](#)。

若要允許不同的 AWS 帳戶 或其委託人使用 KMS 金鑰，您必須在金鑰政策和另一個帳戶中的 IAM 政策中提供許可。如需詳細資訊，請參閱[允許其他帳戶中的使用者使用 KMS 金鑰](#)。

Action

(必要) 指定要允許或拒絕的 API 操作。例如，`kms:Encrypt`動作對應至 AWS KMS [Encrypt](#) 操作。您可以在政策陳述式中列出多個動作。如需詳細資訊，請參閱[許可參考](#)。

Note

如果必要的 Action 元素在金鑰政策陳述式中遺失，則政策陳述式沒有任何效果。沒有 Action 元素的金鑰政策陳述式不適用於任何 KMS 金鑰。

當金鑰政策陳述式缺少其 Action 元素時，AWS KMS 主控台會正確報告錯誤，但 [CreateKey](#) 和 [PutKeyPolicy](#) APIs 會成功，即使政策陳述式無效。

資源

(必要) 在金鑰政策中，Resource 元素的值是 "*"，表示「此 KMS 金鑰」。星號 ("*") 會識別金鑰政策所連接的 KMS 金鑰。

Note

如果必要的 Resource 元素在金鑰政策陳述式中遺失，則政策陳述式沒有任何效果。沒有 Resource 元素的金鑰政策不適用於任何 KMS 金鑰。

當金鑰政策陳述式缺少其 Resource 元素時，AWS KMS 主控台會正確報告錯誤，但 [CreateKey](#) 和 [PutKeyPolicy](#) APIs 會成功，即使政策陳述式無效。

條件

(選用) 條件則會指定金鑰政策生效必須符合的需求。使用條件時，AWS 可以評估 API 請求的內容，以判斷政策陳述式是否適用。

若要指定條件，您可以使用預先定義的條件索引鍵。AWS KMS 支援 [AWS 全域條件索引鍵](#) 和 [AWS KMS 條件索引鍵](#)。為了支援屬性型存取控制 (ABAC)，AWS KMS 提供條件金鑰，以根據標籤和別名控制對 KMS 金鑰的存取。如需詳細資訊，請參閱 [適用於的 ABAC AWS KMS](#)。

條件的格式為：

```
"Condition": {"condition operator": {"condition key": "condition value"}}
```

例如：

```
"Condition": {"StringEquals": {"kms:CallerAccount": "111122223333"}}
```

如需 AWS 政策語法的詳細資訊，請參閱 [《AWS IAM 使用者指南》中的 IAM 政策參考](#)。

範例金鑰政策

下列範例顯示對稱加密 KMS 金鑰的完整金鑰政策。在閱讀本章中的關鍵政策概念時，您可以將其用作參考。此金鑰政策將先前[預設金鑰政策](#)一節的範例政策陳述式結合至單一金鑰政策，來完成以下任務：

- 允許範例 AWS 帳戶 111122223333 完整存取 KMS 金鑰。它允許帳戶及其管理員 (包括帳戶根使用者 (緊急情況)) 在帳戶中使用 IAM 政策，以允許對 KMS 金鑰的存取。
- 允許 ExampleAdminRole IAM 角色管理 KMS 金鑰。
- 允許 ExampleUserRole IAM 角色使用 KMS 金鑰。

JSON

```
{
  "Id": "key-consolepolicy",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "EnableIAMUserPermissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "AllowKeyAdministratorsAccess",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/ExampleAdminRole"
      },
      "Action": [
        "kms:Create*",
        "kms:Describe*",
        "kms:Enable*",
        "kms:List*",
        "kms:Put*",
        "kms:Update*",
        "kms:Revoke*",
        "kms:Disable*",
        "kms:Get*",

```

```

        "kms:Delete*",
        "kms:TagResource",
        "kms:UntagResource",
        "kms:ScheduleKeyDeletion",
        "kms:CancelKeyDeletion",
        "kms:RotateKeyOnDemand"
    ],
    "Resource": "*"
},
{
    "Sid": "AllowKeyUse",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/ExampleUserRole"
    },
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
    "Resource": "*"
},
{
    "Sid": "AllowAttachmentPersistentResources",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/ExampleUserRole"
    },
    "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
    ],
    "Resource": "*",
    "Condition": {
        "Bool": {
            "kms:GrantIsForAWSResource": "true"
        }
    }
}
]

```

```
}
```

預設金鑰政策

在您建立 KMS 金鑰時，可以指定新 KMS 金鑰的金鑰政策。如果您未提供，會為您 AWS KMS 建立一個。AWS KMS 使用的預設金鑰政策會因您在 AWS KMS 主控台中建立金鑰或使用 AWS KMS API 而有所不同。

當您以程式設計方式建立 KMS 金鑰時的預設金鑰政策

當您使用 [AWS KMS API](#) 以程式設計方式建立 KMS 金鑰（包括使用 [AWS SDKs](#) [AWS Command Line Interface](#) 或 [AWS Tools for PowerShell](#)），而且您未指定金鑰政策時，會 AWS KMS 套用非常簡單的預設金鑰政策。此預設金鑰政策有一個政策陳述式，提供 AWS 帳戶擁有 KMS 金鑰的使用 IAM 政策的許可，以允許存取 KMS 金鑰上的所有 AWS KMS 操作。如需有關此政策陳述式的詳細資訊，請參閱 [允許存取 AWS 帳戶 並啟用 IAM 政策](#)。

使用 建立 KMS 金鑰時的預設金鑰政策 AWS 管理主控台

當您使用 [建立 KMS 金鑰 AWS 管理主控台](#) 時，金鑰政策會從 [允許存取 AWS 帳戶 並啟用 IAM 政策](#) 的政策陳述式開始。主控台接著會新增 [金鑰管理員陳述式](#)、[金鑰使用者陳述式](#)，以及（對於大多數金鑰類型）允許主體將 KMS 金鑰與其他 [AWS 服務](#) 搭配使用的陳述式。您可以使用 AWS KMS 主控台的功能來指定 IAM 使用者、IAM 角色，以及 AWS 帳戶 誰是金鑰管理員，以及誰是金鑰使用者（或兩者）。

許可

- [允許存取 AWS 帳戶 並啟用 IAM 政策](#)
- [允許金鑰管理員來管理 KMS 金鑰](#)
- [允許金鑰使用者使用 KMS 金鑰](#)
 - [允許金鑰使用者在密碼編譯操作中使用 KMS 金鑰](#)
 - [允許金鑰使用者搭配 AWS 服務使用 KMS 金鑰](#)

允許存取 AWS 帳戶 並啟用 IAM 政策

以下預設金鑰政策陳述式至關重要。

- 它可讓 AWS 帳戶 擁有 KMS 金鑰的 完整存取 KMS 金鑰。

與其他 AWS 資源政策不同，AWS KMS 金鑰政策不會自動將許可授予帳戶或其任何身分。若要向帳戶管理員授予許可，金鑰政策必須包含提供此許可的明確陳述式。

- 除金鑰政策外，它還允許帳戶使用 IAM 政策來允許對 KMS 金鑰的存取。

如果沒有此許可，允許存取金鑰的 IAM 政策將無效，但拒絕存取金鑰的 IAM 政策仍然有效。

- 它透過向帳戶管理員 (包括帳戶根使用者) 授予無法刪除存取控制許可，來降低金鑰變得無法管理的風險。

以下金鑰政策陳述式是以程式設計方式建立的 KMS 金鑰的完整預設金鑰政策。這是 AWS KMS 在主控台中建立之 KMS 金鑰的預設金鑰政策中的第一個政策陳述式。

```
{
  "Sid": "Enable IAM User Permissions",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:root"
  },
  "Action": "kms:*",
  "Resource": "*"
}
```

允許 IAM 政策允許對 KMS 金鑰的存取。

上述金鑰政策陳述式提供擁有金鑰許可 AWS 帳戶 的使用 IAM 政策以及金鑰政策，以允許 KMS 金鑰上的所有動作 (kms:*)。

此金鑰政策陳述式的主體為[帳戶主體](#)，由以下格式的 ARN 表示：arn:aws:iam::**account-id**:root。帳戶主體代表 AWS 帳戶及其管理員。

若金鑰政策陳述式中的主體為帳戶主體，則政策陳述式不會向任何 IAM 主體授予使用 KMS 金鑰的許可。而是允許帳戶使用 IAM 政策，來委派政策陳述式中指定的許可。此預設金鑰政策陳述式允許帳戶使用 IAM 政策，來委派 KMS 金鑰上的所有動作許可 (kms:*)。

降低 KMS 金鑰變得無法管理的風險。

與其他 AWS 資源政策不同，AWS KMS 金鑰政策不會自動將許可授予帳戶或其任何委託人。向包括[帳戶主體](#)在內的任何主體授予許可，必須使用明確授予許可的金鑰政策陳述式。您無需授予帳戶主體或任何主體存取 KMS 金鑰的許可。然而，向帳戶主體授予存取許可，能夠幫助您防止金鑰變得無法管理。

例如，假設您建立的金鑰政策僅向一個使用者授予存取 KMS 金鑰的許可。如果您隨後刪除該使用者，金鑰將變得無法管理，您必須[聯絡 AWS 支援](#)，以重新取得對 KMS 金鑰的存取。

上面顯示的金鑰政策陳述式提供[帳戶主體](#)控制金鑰的許可。帳戶主體代表 AWS 帳戶及其管理員，包括[帳戶根使用者](#)。帳戶根使用者是唯一不能刪除的主體，除非您刪除 AWS 帳戶。IAM 最佳實務不鼓勵代表帳戶根使用者採取行動，但在緊急情況下除外。然而，如果刪除具有 KMS 金鑰存取許可的所有其他使用者和角色，則可能需要充當帳戶根使用者。

允許金鑰管理員來管理 KMS 金鑰

透過主控台建立的預設金鑰政策可讓您選擇帳戶中的 IAM 使用者和角色，並將其設為金鑰管理員。此陳述式稱為金鑰管理員陳述式。金鑰管理員有權管理 KMS 金鑰，但無權在[密碼編譯操作](#)中使用 KMS 金鑰。當您在預設檢視或政策檢視中建立 KMS 金鑰時，您可以新增 IAM 使用者和角色至金鑰管理員的清單。

Warning

由於金鑰管理員具有變更金鑰政策並建立授予的許可，因此他們可以授予自己和此政策中未指定的其他 AWS KMS 許可。

擁有管理標籤和別名許可的主體也可以控制對 KMS 金鑰的存取。如需詳細資訊，請參閱[適用於的 ABAC AWS KMS](#)。

Note

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 中的安全性最佳實務](#)。

以下範例顯示 AWS KMS 主控台之預設檢視中的金鑰管理員陳述式。

Key policy

Tags

Key policy

Switch to policy view

Key administrators

Choose the IAM users and roles who can administer this key through the KMS API. You might need to add additional permissions for the users or roles to administer this key from this console. [Learn more](#)

Add

Remove

<

1

>

☐	Name	Path	Type
☐	ExampleAdminRole	/	Role

Key deletion

☒ Allow key administrators to delete this key

以下是 AWS KMS 主控台之政策檢視中的範例金鑰管理員陳述式。此金鑰管理員陳述式適用於單一區域對稱加密 KMS 金鑰。

Note

AWS KMS 主控台會將金鑰管理員新增至陳述式識別碼下的金鑰政策"Allow access for Key Administrators"。修改此陳述式識別符可能會影響主控台顯示您對陳述式所做的更新的方式。

```
{
  "Sid": "Allow access for Key Administrators",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/ExampleAdminRole"},
  "Action": [
    "kms:Create*",
  ]
}
```

```
"kms:Describe*",
"kms:Enable*",
"kms:List*",
"kms:Put*",
"kms:Update*",
"kms:Revoke*",
"kms:Disable*",
"kms:Get*",
"kms:Delete*",
"kms:TagResource",
"kms:UntagResource",
"kms:ScheduleKeyDeletion",
"kms:CancelKeyDeletion",
"kms:RotateKeyOnDemand"
],
"Resource": "*"
}
```

最常見的 KMS 金鑰 (即單一區域對稱加密 KMS 金鑰) 的預設金鑰管理員陳述式允許以下許可。如需有關每個許可的詳細資訊，請參閱 [AWS KMS 許可](#)。

當您使用 AWS KMS 主控台建立 KMS 金鑰時，主控台會將您指定的使用者和角色新增至金鑰管理員陳述式中的 Principal 元素。

這些許可中有許多都包含萬用字元 (*)，允許以指定動詞開頭的所有許可。因此，當 AWS KMS 新增 API 操作時，會自動允許金鑰管理員使用這些操作。您不需要更新您的主要政策，即可包含新的操作。如果您想要將金鑰管理員限制在一組固定的 API 操作中，您可以[變更金鑰政策](#)。

kms:Create*

允許 [kms:CreateAlias](#) 和 [kms:CreateGrant](#)。(kms:CreateKey 許可僅在 IAM 政策中有效。)

kms:Describe*

允許 [kms:DescribeKey](#) 需要 kms:DescribeKey 許可才能檢視 AWS 管理主控台中 KMS 金鑰的金鑰詳細資訊頁面。

kms:Enable*

允許 [kms:EnableKey](#) 如果是對稱加密 KMS 金鑰，其亦允許 [kms:EnableKeyRotation](#)。

kms:List*

允許 [kms:ListGrants](#)、[kms:ListKeyPolicies](#) 和 [kms:ListResourceTags](#)。(在 AWS 管理主控台中檢視 KMS 金鑰所需的 `kms:ListAliases` 和 `kms:ListKeys` 許可，只在 IAM 政策中有效。)

kms:Put*

允許 [kms:PutKeyPolicy](#) 此許可允許金鑰管理員變更此 KMS 金鑰的金鑰政策。

kms:Update*

允許 [kms:UpdateAlias](#) 和 [kms:UpdateKeyDescription](#)。對於多區域金鑰，它允許此 KMS 金鑰上的 [kms:UpdatePrimaryRegion](#)。

kms:Revoke*

允許 [kms:RevokeGrant](#)，它允許金鑰管理員[刪除授權](#)，即使他們不是授權中的[淘汰主體](#)。

kms:Disable*

允許 [kms:DisableKey](#) 如果是對稱加密 KMS 金鑰，其亦允許 [kms:DisableKeyRotation](#)。

kms:Get*

允許 [kms:GetKeyPolicy](#) 和 [kms:GetKeyRotationStatus](#)。對於具有匯入金鑰材料的 KMS 金鑰，它允許 [kms:GetParametersForImport](#)。對於非對稱 KMS 金鑰，它允許 [kms:GetPublicKey](#)。需要 `kms:GetKeyPolicy` 許可才能檢視 AWS 管理主控台中 KMS 金鑰的金鑰政策。

kms:Delete*

允許 [kms>DeleteAlias](#) 對於具有匯入金鑰材料的金鑰，它允許 [kms>DeleteImportedKeyMaterial](#)。`kms:Delete*` 許可不允許金鑰管理員刪除 KMS 金鑰 (`ScheduleKeyDeletion`)。

kms:TagResource

允許 [kms:TagResource](#)，可讓金鑰管理員將標籤新增至 KMS 金鑰。由於標籤也可用來控制對 KMS 金鑰的存取，此許可讓管理員可允許或拒絕對 KMS 金鑰的存取。如需詳細資訊，請參閱[適用於的 ABAC AWS KMS](#)。

kms:UntagResource

允許 [kms:UntagResource](#)，可讓金鑰管理員從 KMS 金鑰刪除標籤。由於標籤可用來控制對金鑰的存取，因此此許可可讓管理員允許或拒絕對 KMS 金鑰的存取。如需詳細資訊，請參閱[適用於的 ABAC AWS KMS](#)。

kms:ScheduleKeyDeletion

允許 [kms:ScheduleKeyDeletion](#)，可讓金鑰管理員[刪除此 KMS 金鑰](#)。若要刪除此許可，請清除 Allow key administrators to delete this key (允許金鑰管理員刪除此金鑰) 選項。

kms:CancelKeyDeletion

允許 [kms:CancelKeyDeletion](#)，可讓金鑰管理員[取消刪除此 KMS 金鑰](#)。若要刪除此許可，請清除 Allow key administrators to delete this key (允許金鑰管理員刪除此金鑰) 選項。

kms:RotateKeyOnDemand

允許 [kms:RotateKeyOnDemand](#)，這可讓金鑰管理員[執行此 KMS 金鑰中金鑰材料的隨需輪換](#)。

AWS KMS 當您建立特殊用途金鑰時，會將下列許可新增至預設金鑰管理員陳述式。

kms:ImportKeyMaterial

[kms:ImportKeyMaterial](#) 許可允許金鑰管理員將金鑰材料匯入 KMS 金鑰。只有當您[建立不含金鑰資料的 KMS 金鑰](#)時，此許可才會包含在金鑰政策中。

kms:ReplicateKey

[kms:ReplicateKey](#) 許可允許金鑰管理員[在不同區域中建立多區域主要金鑰的複本](#) AWS。只有當您建立多區域主要金鑰或複本金鑰時，此許可才會包含在金鑰政策中。

kms:UpdatePrimaryRegion

[kms:UpdatePrimaryRegion](#) 許可允許金鑰管理員[將多區域複本金鑰變更為多區域主要金鑰](#)。只有當您建立多區域主要金鑰或複本金鑰時，此許可才會包含在金鑰政策中。

允許金鑰使用者使用 KMS 金鑰

主控台為 KMS 金鑰建立的預設金鑰政策可讓您選擇帳戶中和外部的 IAM 使用者和 IAM 角色 AWS 帳戶，並使其成為金鑰使用者。

主控台會將兩個政策陳述式新增至金鑰使用者的金鑰政策。

- [直接使用 KMS 金鑰](#) – 第一個金鑰政策陳述式給予金鑰使用者直接針對該類型 KMS 金鑰所有受支援的[密碼編譯操作](#)使用 KMS 金鑰的許可。

- 將 [KMS 金鑰與 AWS 服務](#) 搭配使用 — 第二個政策陳述式提供金鑰使用者許可，允許與整合 AWS 的服務代表他們 AWS KMS 使用 KMS 金鑰來保護資源，例如 Amazon S3 儲存貯體和 Amazon DynamoDB 資料表。

您可以在建立 KMS 金鑰時 AWS 帳戶，將 IAM 使用者、IAM 角色和其他新增至金鑰使用者清單。您也可以使用主控台的金鑰政策預設檢視來編輯清單，如下圖所示。金鑰政策的預設檢視位於金鑰詳細資訊頁面。如需允許其他中的使用者 AWS 帳戶使用 KMS 金鑰的詳細資訊，請參閱 [允許其他帳戶中的使用者使用 KMS 金鑰](#)。

 Note

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的安全性最佳實務](#)。

Key users

The following IAM users and roles can use this key for cryptographic operations. They can also allow AWS services that are integrated with KMS to use the key on their behalf. [Learn more](#) 

AddRemove

< **1** >

☐	Name	Path	Type
☐	ExampleRole	/	Role

Other AWS accounts

- arn:aws:iam::444455556666:root

單一區域對稱的預設金鑰使用者陳述式允許下列許可。如需有關每個許可的詳細資訊，請參閱 [AWS KMS 許可](#)。

當您使用 AWS KMS 主控台建立 KMS 金鑰時，主控台會將您指定的使用者和角色新增至每個金鑰使用者陳述式中的 Principal 元素。

 Note

AWS KMS 主控台會將金鑰使用者新增至陳述式識別碼 "Allow use of the key" 和 下的金鑰政策 "Allow attachment of persistent resources"。修改這些陳述式識別符可能會影響主控台顯示您對陳述式所做的更新。

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {"AWS": [
    "arn:aws:iam::111122223333:role/ExampleRole",
    "arn:aws:iam::444455556666:root"
  ]},
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:DescribeKey"
  ],
  "Resource": "*"
},
{
  "Sid": "Allow attachment of persistent resources",
  "Effect": "Allow",
  "Principal": {"AWS": [
    "arn:aws:iam::111122223333:role/ExampleRole",
    "arn:aws:iam::444455556666:root"
  ]},
  "Action": [
    "kms:CreateGrant",
    "kms:ListGrants",
    "kms:RevokeGrant"
  ],
  "Resource": "*",
  "Condition": {"Bool": {"kms:GrantIsForAWSResource": true}}
}
```

允許金鑰使用者在密碼編譯操作中使用 KMS 金鑰

金鑰使用者有權直接在 KMS 金鑰支援的所有[密碼編譯操作](#)中使用 KMS 金鑰。他們也可以使用 [DescribeKey](#) 操作，在 AWS KMS 主控台或使用 AWS KMS API 操作取得 KMS 金鑰的詳細資訊。

根據預設，AWS KMS 主控台會將下列範例中的金鑰使用者陳述式新增至預設金鑰政策。由於政策陳述式支援不同的 API 操作，因此對稱加密 KMS 金鑰、HMAC KMS 金鑰、用於公有金鑰加密的非對稱 KMS 金鑰、用於簽署和驗證的非對稱 KMS 金鑰的政策陳述式中的動作都會略有不同。

對稱加密 KMS 金鑰

主控台會將下列陳述式新增至對稱加密 KMS 金鑰的金鑰政策。

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/ExampleKeyUserRole"},
  "Action": [
    "kms:Decrypt",
    "kms:DescribeKey",
    "kms:Encrypt",
    "kms:GenerateDataKey*",
    "kms:ReEncrypt*"
  ],
  "Resource": "*"
}
```

HMAC KMS 金鑰

主控台會將下列陳述式新增至 HMAC KMS 金鑰的金鑰政策。

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/ExampleKeyUserRole"},
  "Action": [
    "kms:DescribeKey",
    "kms:GenerateMac",
    "kms:VerifyMac"
  ],
  "Resource": "*"
}
```

公有金鑰加密的非對稱 KMS 金鑰

主控台會將下列陳述式新增至金鑰使用情形為 Encrypt and decrypt (加密和解密) 之非對稱 KMS 金鑰的金鑰政策。

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleKeyUserRole"
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:DescribeKey",
    "kms:GetPublicKey"
  ],
  "Resource": "*"
}
```

用於簽署和驗證的非對稱 KMS 金鑰

主控台會將下列陳述式新增至金鑰使用情形為 Sign and verify (簽署和驗證) 之非對稱 KMS 金鑰的金鑰政策。

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/ExampleKeyUserRole"},
  "Action": [
    "kms:DescribeKey",
    "kms:GetPublicKey",
    "kms:Sign",
    "kms:Verify"
  ],
  "Resource": "*"
}
```

用於衍生共用秘密的非對稱 KMS 金鑰

主控台會將下列陳述式新增至具有金鑰協議金鑰用量的非對稱 KMS 金鑰的金鑰政策。

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/ExampleKeyUserRole"},
  "Action": [
    "kms:DescribeKey",
    "kms:GetPublicKey",
    "kms:DeriveSharedSecret"
  ],
  "Resource": "*"
}
```

這些陳述式中的動作會提供金鑰使用者下列許可。

[kms:Encrypt](#)

允許金鑰使用者使用此 KMS 金鑰加密資料。

[kms:Decrypt](#)

允許金鑰使用者使用此 KMS 金鑰解密資料。

[kms:DeriveSharedSecret](#)

允許金鑰使用者使用此 KMS 金鑰衍生共用秘密。

[kms:DescribeKey](#)

允許金鑰使用者取得此 KMS 金鑰的資訊，包括其識別符、建立日期和金鑰狀態。它還允許金鑰使用者在 AWS KMS 主控台中顯示 KMS 金鑰的詳細資訊。

kms:GenerateDataKey*

允許金鑰使用者為用戶端密碼編譯操作請求對稱資料金鑰或非對稱資料金鑰對。主控台使用 * 萬用字元表示允許下列 API 操作：[GenerateDataKey](#)、[GenerateDataKeyWithoutPlaintext](#)、[GenerateDataKeyPair](#) 和 [GenerateDataKeyPairWithoutPlaintext](#)。這些許可只有在加密資料金鑰的對稱 KMS 金鑰上才有效。

[kms:GenerateMac](#)

允許金鑰使用者使用 HMAC KMS 金鑰來產生 HMAC 標籤。

[kms:GetPublicKey](#)

允許金鑰使用者下載非對稱 KMS 金鑰的公有金鑰。與您共用此公有金鑰的對象可以加密 外部的資料 AWS KMS。不過，這些加密文字只能透過呼叫 AWS KMS 中的 [Decrypt](#) 操作進行解密。

[kms:ReEncrypt*](#)

允許金鑰使用者重新加密原本使用此 KMS 金鑰加密的資料，或使用此 KMS 金鑰重新加密之前加密過的資料。[ReEncrypt](#) 操作需要同時存取來源和目標 KMS 金鑰。若要完成此動作，您可以允許來源 KMS 金鑰的 `kms:ReEncryptFrom` 許可和目標 KMS 金鑰的 `kms:ReEncryptTo` 許可。但為簡單起見，主控台允許兩種 KMS 金鑰都使用 `kms:ReEncrypt*` (使用 * 萬用字元)。

[kms:Sign](#)

允許金鑰使用者使用此 KMS 金鑰簽署訊息。

[kms:Verify](#)

允許金鑰使用者使用此 KMS 金鑰驗證簽章。

[kms:VerifyMac](#)

允許金鑰使用者使用 HMAC KMS 金鑰來驗證 HMAC 標籤。

允許金鑰使用者搭配 AWS 服務使用 KMS 金鑰

主控台預設金鑰政策也會為金鑰使用者提供所需的授予許可，以保護使用 `grant.services` AWS 的服務中的資料。AWS 服務通常會使用 `grant` 來取得使用 KMS 金鑰的特定有限許可。

此金鑰政策陳述式允許金鑰使用者建立、檢視和撤銷 KMS 金鑰的授予，但僅限於授予操作請求來自 [AWS 與整合的服務 AWS KMS](#) 時。[kms:GrantIsForAWSResource](#) 政策條件不允許使用者直接呼叫這些授予操作。當金鑰使用者允許時，AWS 服務可以代表使用者建立授予，以允許服務使用 KMS 金鑰來保護使用者的資料。

金鑰使用者需要這些授權許可，才能使用其 KMS 金鑰與整合服務，但這些許可仍有不足。金鑰使用者也需要有許可才能使用整合服務。如需有關讓使用者存取與整合之 AWS 服務的詳細資訊 AWS KMS，請參閱整合服務的文件。

```
{
  "Sid": "Allow attachment of persistent resources",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/ExampleKeyUserRole"},
  "Action": [
```

```
"kms:CreateGrant",
"kms:ListGrants",
"kms:RevokeGrant"
],
"Resource": "*",
"Condition": {"Bool": {"kms:GrantIsForAWSResource": true}}
}
```

例如，金鑰使用者可以透過下列方式在 KMS 金鑰上使用這些許可。

- 將此 KMS 金鑰搭配 Amazon Elastic Block Store (Amazon EBS) 和 Amazon Elastic Compute Cloud (Amazon EC2) 使用，將加密的 EBS 磁碟區連接到 EC2 執行個體。金鑰使用者隱含提供 Amazon EC2 使用 KMS 金鑰將加密磁碟區連接到執行個體的許可。如需詳細資訊，請參閱[Amazon Elastic Block Store \(Amazon EBS\) 如何使用 AWS KMS](#)。
- 將此 KMS 金鑰搭配 Amazon Redshift 使用，以啟動加密叢集。金鑰使用者隱含提供 Amazon Redshift 使用 KMS 金鑰啟動加密叢集並建立加密快照的許可。如需詳細資訊，請參閱[Amazon Redshift 如何使用 AWS KMS](#)。
- 使用此 KMS 金鑰搭配其他[與 AWS KMS 整合且使用授權的 AWS 服務](#)，來建立、管理或使用加密資源搭配這些服務。

預設金鑰政策允許金鑰使用者將其許可授予所有使用授權的整合服務。不過，您可以建立自訂金鑰政策，將許可限制為指定的 AWS 服務。如需詳細資訊，請參閱 [kms:ViaService](#) 條件金鑰。

檢視金鑰政策

您可以使用 AWS KMS 主控台或 AWS KMS API [AWS 受管金鑰](#) 中的 [GetKeyPolicy](#) 操作，檢視 AWS KMS [帳戶中客戶受管金鑰](#) 或的金鑰政策。您無法使用這些技術檢視不同 AWS 帳戶中 KMS 金鑰的金鑰政策。

若要進一步了解 AWS KMS 金鑰政策，請參閱 [中的金鑰政策 AWS KMS](#)。若要了解如何判斷哪些使用者和角色可以存取 KMS 金鑰，請參閱 [the section called “判斷存取權”](#)。

使用 AWS KMS 主控台

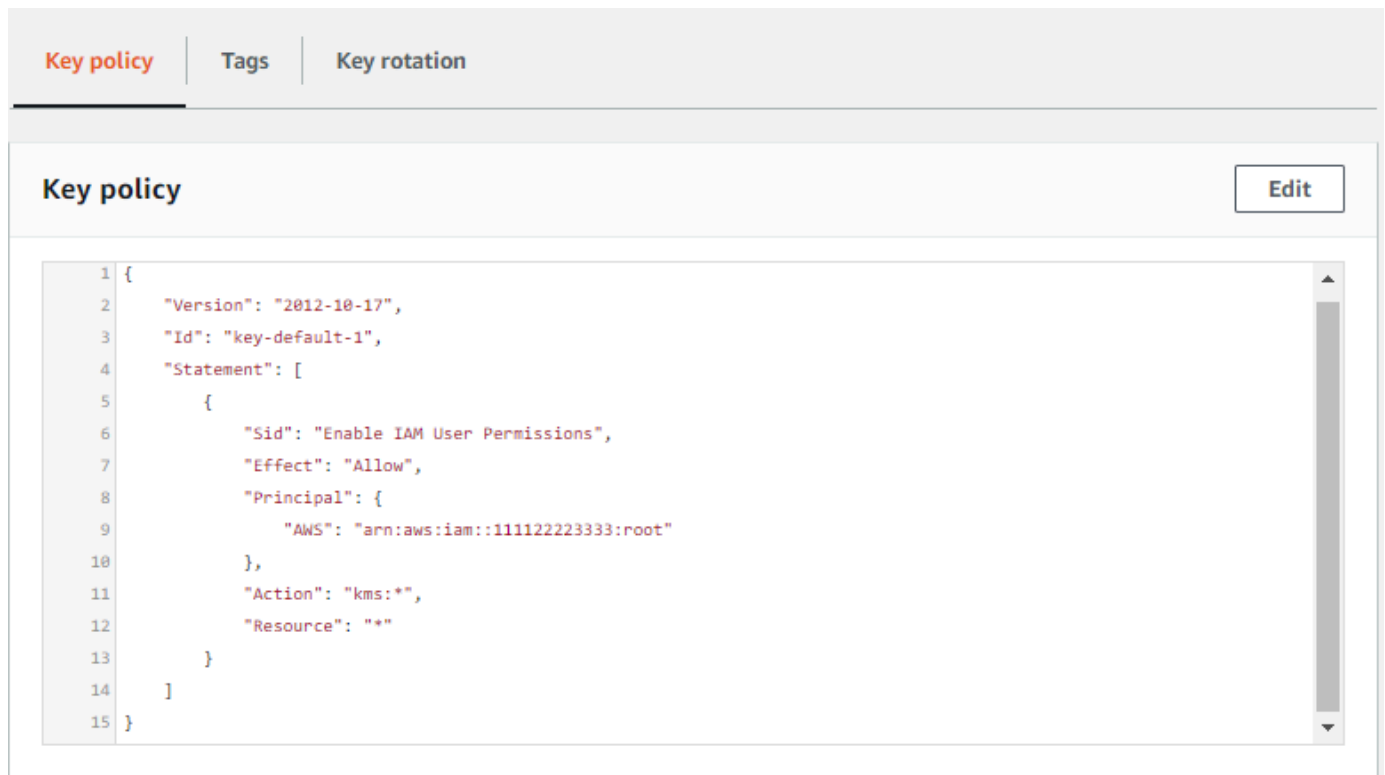
授權的使用者可以在 AWS 管理主控台的 Key policy (金鑰政策) 索引標籤上檢視 [AWS 受管金鑰](#) 或 [客戶受管金鑰](#) 的金鑰政策。

若要在 中檢視 KMS 金鑰的金鑰政策 AWS 管理主控台，您必須具有 [kms:ListAliases](#)、[kms:DescribeKey](#) 和 [kms:GetKeyPolicy](#) 許可。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 若要檢視帳戶中為您 AWS 建立和管理的金鑰，請在導覽窗格中選擇AWS 受管金鑰。若要檢視您所建立及管理帳戶中的金鑰，請在導覽窗格中選擇Customer managed keys (客戶受管金鑰)。
4. 在 KMS 金鑰清單中，選擇您要檢查之 KMS 金鑰的別名或金鑰 ID。
5. 選擇 Key policy (金鑰政策) 標籤。

在 Key policy (金鑰政策) 索引標籤中，您可能會看到金鑰政策文件。這是「政策檢視」。在金鑰政策陳述式中，您可以看到由金鑰政策授予 KMS 金鑰存取權的主體，也會看到他們可以執行的動作。

下列範例顯示[預設金鑰政策](#)的政策檢視。



或者，如果您在 中建立 KMS 金鑰 AWS 管理主控台，您會看到包含金鑰管理員、金鑰刪除和金鑰使用者區段的預設檢視。若要查看金鑰政策文件，請選擇 Switch to policy view (切換至政策檢視)。

下列範例顯示[預設金鑰政策](#)的預設檢視。

Key policy

Tags

Key rotation

Key policy

Switch to policy view

Key administrators

Choose the IAM users and roles who can administer this key through the KMS API. You might need to add additional permissions for the users or roles to administer this key from this console. [Learn more](#)

Add

Remove

Q

< 1 >

	Name	Path	Type
Empty Resources			
No resources to display			

Key deletion

☐

 Allow key administrators to delete this key

Key users

The following IAM users and roles can use this key to encrypt and decrypt data from within applications and when using AWS services integrated with KMS. [Learn more](#)

Add

Remove

Q

< 1 >

	Name	Path	Type
Empty Resources			
No resources to display			

使用 AWS KMS API

若要取得 KMS 金鑰的金鑰政策 AWS 帳戶，請使用 API 中的 AWS KMS [GetKeyPolicy](#) 操作。您無法使用此操作檢視不同帳戶中的金鑰政策。

下列範例使用 AWS Command Line Interface (AWS CLI) 中的 [get-key-policy](#) 命令，但您可以使用任何 AWS SDK 提出此請求。

請注意，雖然 `default` 是唯一的有效值，但 `PolicyName` 參數是必要的。此外，這個命令請求使用較易檢視的文字輸出，而不是 JSON。

執行此命令之前，請將範例金鑰 ID 更換成您帳戶的有效 ID。

```
$ aws kms get-key-policy --key-id 1234abcd-12ab-34cd-56ef-1234567890ab --policy-name default --output text
```

回應應該類似下面其中一個項目，會傳回[預設的金鑰政策](#)。

JSON

```
{
  "Version": "2012-10-17",
  "Id" : "key-consolepolicy-3",
  "Statement" : [ {
    "Sid" : "EnableIAMUserPermissions",
    "Effect" : "Allow",
    "Principal" : {
      "AWS" : "arn:aws:iam::111122223333:root"
    },
    "Action" : "kms:*",
    "Resource" : "*"
  } ]
}
```

變更金鑰政策

您可以使用 AWS 管理主控台 或 [PutKeyPolicy](#) 操作 AWS 帳戶 來變更 中 KMS 金鑰的金鑰政策。您無法使用這些技術變更不同 AWS 帳戶中 KMS 金鑰的金鑰政策。

變更金鑰政策時，請注意以下規則：

- 您可以檢視 [AWS 受管金鑰](#) 或 [客戶受管金鑰](#) 的金鑰政策，但只能變更客戶受管金鑰的金鑰政策。的政策 AWS 受管金鑰 是由在您帳戶中建立 KMS 金鑰 AWS 的服務所建立和管理。您無法檢視或變更 [AWS 擁有的金鑰](#) 的金鑰政策。
- 您可以在金鑰政策 AWS 帳戶 中新增或移除 IAM 使用者、IAM 角色和 ，並變更這些主體允許或拒絕的動作。如需在金鑰政策中指定主體和許可之方式的詳細資訊，請參閱[金鑰政策](#)。

- 您不能新增 IAM 群組到金鑰政策，但您可以新增多個 IAM 使用者和 IAM 角色。如需詳細資訊，請參閱[允許多個 IAM 主體存取 KMS 金鑰](#)。
- 如果您將新增至金鑰政策 AWS 帳戶的外部，您還必須在外部帳戶中使用 IAM 政策，將許可授予這些帳戶中的 IAM 使用者、群組或角色。如需詳細資訊，請參閱[允許其他帳戶中的使用者使用 KMS 金鑰](#)。
- 產生的金鑰政策文件不能超過 32 KB (32,768 位元組)。

如何變更金鑰政策

變更金鑰政策有三種方法，如以下章節所述。

主題

- [使用 AWS 管理主控台 預設檢視](#)
- [使用 AWS 管理主控台 政策檢視](#)
- [使用 AWS KMS API](#)

使用 AWS 管理主控台 預設檢視

您可以使用主控台，以稱為預設檢視的圖形界面來變更金鑰政策。

如果以下步驟不符合您在主控台中看到的步驟，可能表示此金鑰政策不是使用主控台所建立的。也可能表示主控台的預設檢視不支援修改後的金鑰政策。在這種情況下，請遵循[使用 AWS 管理主控台 政策檢視](#)或[使用 AWS KMS API](#) 中的步驟進行。

1. 請依 [使用 AWS KMS 主控台](#) 所述，檢視客戶受管金鑰的金鑰政策。(您無法變更的金鑰政策 AWS 受管金鑰。)
2. 決定進行哪些變更。
 - 若要新增或移除[金鑰管理員](#)，以及允許或不允許金鑰管理員[刪除 KMS 金鑰](#)，請使用頁面上的 Key administrators (金鑰管理員) 區段中的控制項。金鑰管理員負責管理 KMS 金鑰，包括啟用和停用它、設定金鑰政策，以及[啟用金鑰輪換](#)。
 - 若要新增或移除[金鑰使用者](#)，以及允許或不允許外部 AWS 帳戶使用 KMS 金鑰，請使用頁面金鑰使用者區段中的控制項。金鑰使用者可以在[密碼編譯操作](#)中使用 KMS 金鑰，例如加密、解密、重新加密和產生資料金鑰。

使用 AWS 管理主控台 政策檢視

您可以使用主控台的「政策檢視」來變更金鑰政策文件。

1. 請依 [使用 AWS KMS 主控台](#) 所述，檢視客戶受管 KMS 金鑰的金鑰政策。（您無法變更的金鑰政策 AWS 受管金鑰。）
2. 在金鑰政策區段中，選擇切換為政策檢視。
3. 選擇編輯。
4. 決定進行哪些變更。
 - 若要新增陳述式，請選擇新增陳述式。然後，您可以從陳述式建置器面板中列出的選項中選取新金鑰政策陳述式的動作、主體和條件，或手動輸入政策陳述式元素。
 - 若要從您的金鑰政策中移除陳述式，請選取陳述式，然後選擇移除。檢閱選取的政策陳述式，並確認您想要將其移除。如果您決定不想繼續移除陳述式，請選擇取消。
 - 若要編輯現有的金鑰政策陳述式，請選取陳述式。然後，您可以使用陳述式建置器面板來選擇您要修改的特定元素，或手動編輯陳述式。
5. 選擇儲存變更。

使用 AWS KMS API

您可以使用 [PutKeyPolicy](#) 操作來變更中 KMS 金鑰的金鑰政策 AWS 帳戶。您無法對其他 AWS 帳戶中的 KMS 金鑰使用此 API。

1. 使用 [GetKeyPolicy](#) 操作取得現有的金鑰政策文件，然後將金鑰政策文件儲存至檔案。如需多種程式設計語言的範例程式碼，請參閱[GetKeyPolicy 搭配 AWS SDK 或 CLI 使用](#)。
2. 在您偏好的文字編輯器中開啟金鑰政策文件、編輯金鑰政策文件，然後儲存檔案。
3. 使用 [PutKeyPolicy](#) 操作將更新的金鑰政策文件套用到 KMS 金鑰。如需多種程式設計語言的範例程式碼，請參閱[PutKeyPolicy 搭配 AWS SDK 或 CLI 使用](#)。

如需將金鑰政策從一個 KMS 金鑰複製到另一個的範例，請參閱《AWS CLI 命令參考》中的[GetKeyPolicy 範例](#)。

金鑰政策中 AWS 服務的許可

許多 AWS 服務會使用 AWS KMS keys 來保護他們管理的資源。當服務使用 [AWS 擁有的金鑰](#) 或 [AWS 受管金鑰](#) 時，服務會建立並維護這些 KMS 金鑰的金鑰政策。

但是，當您透過 AWS 服務使用[客戶受管金鑰](#)時，您需要設定並維護金鑰政策。該金鑰政策必須允許服務具有代表您保護資源所需的最低許可。建議您遵循最低權限原則：僅授予服務所需的許可。您可以藉由了解服務需要哪些權限並使用[AWS 全域條件金鑰](#)和[AWS KMS 條件金鑰](#)來調整許可，有效地完成這個操作。

若要尋找服務對客戶受管金鑰所需的許可，請參閱服務的加密文件。下列清單包含部分 服務文件的連結：

- AWS CloudTrail 許可 - [設定 CloudTrail 的 AWS KMS 金鑰政策](#)
- Amazon Elastic Block Store 許可 - [Amazon EC2 使用者指南](#)和 [Amazon EC2 使用者指南](#)
- AWS Lambda 許可 - [Lambda 的靜態資料加密](#)
- Amazon Q 許可 - [Amazon Q 的資料加密](#)
- Amazon Relational Database Service 許可 - [AWS KMS 金鑰管理](#)
- AWS Secrets Manager 許可 - [授權使用 KMS 金鑰](#)
- Amazon Simple Queue Service 許可 - [Amazon SQS 金鑰管理](#)

搭配 使用 IAM 政策 AWS KMS

您可以使用 IAM 政策，以及[金鑰政策](#)、[授權](#)和[VPC 端點政策](#)，來控制對 AWS KMS keys 的存取 AWS KMS。

Note

若要使用 IAM 政策來控制 KMS 金鑰的存取，KMS 金鑰的金鑰政策必須授予帳戶使用 IAM 政策的許可。特別是，金鑰政策必須包含[啟用 IAM 政策的政策陳述式](#)。

本節說明如何使用 IAM 政策來控制對 AWS KMS 操作的存取。如需 IAM 許可的一般資訊，請參閱《[IAM 使用者指南](#)》。

所有 KMS 金鑰都必須有金鑰政策。IAM 政策是選用的。若要使用 IAM 政策來控制 KMS 金鑰的存取，KMS 金鑰的金鑰政策必須授予帳戶使用 IAM 政策的許可。特別是，金鑰政策必須包含[啟用 IAM 政策的政策陳述式](#)。

IAM 政策可以控制對任何 AWS KMS 操作的存取。與金鑰政策不同，IAM 政策可以控制對多個 KMS 金鑰的存取，並為數個相關 AWS 服務的操作提供許可。但是 IAM 政策對於控制操作的存取 (例如[CreateKey](#)) 特別有用，因為它們不涉及任何特定 KMS 金鑰而無法由金鑰政策控制。

如果您 AWS KMS 透過 Amazon Virtual Private Cloud (Amazon VPC) 端點存取，您也可以使用 VPC 端點政策來限制在使用端點時對 AWS KMS 資源的存取。例如，使用 VPC 端點時，您可能只允許中的委託人 AWS 帳戶存取您的客戶受管金鑰。如需詳細資訊，請參閱 [VPC 端點政策](#)。

如需撰寫及格式化 JSON 政策文件的說明，請參閱《IAM 使用者指南》中的 [IAM JSON 政策參考](#)。

您可以下方式來使用 IAM 政策：

- 連接許可政策到聯合或跨帳戶許可的角色 – 您可以連接 IAM 政策到 IAM 角色來啟用聯合身分、允許跨帳戶許可，或提供應用程式在 EC2 執行個體上執行的許可。如需 IAM 角色各種使用案例的詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 角色](#)。
- 連接許可政策到使用者或群組 – 您可以連接允許使用者或使用者群組呼叫 AWS KMS 操作的政策。不過，IAM 最佳實務建議您盡可能使用具有臨時憑證的身分，例如 IAM 角色。

下列範例顯示具有 AWS KMS 許可的 IAM 政策。此政策允許其連接的 IAM 身分列出所有 KMS 金鑰和別名。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:ListKeys",
      "kms:ListAliases"
    ],
    "Resource": "*"
  }
}
```

與所有 IAM 政策一樣，此政策沒有 Principal 元素。將 IAM 政策連接到 IAM 身分時，該身分會取得政策中指定的許可。

如需顯示所有 AWS KMS API 動作及其適用的資源的資料表，請參閱 [許可參考](#)。

允許多個 IAM 主體存取 KMS 金鑰

IAM 群組在金鑰政策中不是有效的主體。若要允許多個使用者和角色存取 KMS 金鑰，請執行下列其中一項操作：

- 使用 IAM 角色作為金鑰政策中的主體。多個授權使用者可以視需要擔任該角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 角色](#)。

雖然您可以在金鑰政策中列出多個 IAM 使用者，但不建議使用此做法，因為這會要求您在每次授權使用者清單變更時更新金鑰政策。此外，IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的安全性最佳實務](#)。

- 使用 IAM 政策，將許可授予給 IAM 群組。為此，請確保金鑰政策包含的陳述式 [可啟用 IAM 政策以允許存取 KMS 金鑰](#)，建立一個允許存取 KMS 金鑰的 [IAM 政策](#)，然後 [將該政策連接到包含該授權 IAM 使用者的 IAM 群組](#)。透過此方式，您不需要在授權的使用者清單變更時，隨之變更任何政策。相反地，您只需要從適當的 IAM 群組新增或移除這些使用者。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 使用者群組](#)

如需 AWS KMS 金鑰政策和 IAM 政策如何搭配運作的詳細資訊，請參閱 [故障診斷 AWS KMS 許可](#)。

IAM 政策的最佳實務

保護對的存取對於所有 AWS 資源的安全 AWS KMS keys 至關重要。KMS 金鑰可用來保護您的 AWS 帳戶中許多最敏感的資源。花時間設計控制 KMS 金鑰存取的 [金鑰政策](#)、IAM 政策、[授權](#)和 VPC 端點政策。

在控制 KMS 金鑰存取的 IAM 政策陳述式中，使用[最低權限政策](#)。僅為 IAM 主體提供他們必須使用或管理之 KMS 金鑰所需的許可。

下列最佳實務適用於控制 AWS KMS 金鑰和別名存取的 IAM 政策。如需一般 IAM 政策最佳實務指南，請參閱《IAM 使用者指南》中的 [IAM 安全最佳實務](#)。

使用金鑰政策

盡可能在影響一個 KMS 金鑰的金鑰政策中提供許可，而不是在可套用至許多 KMS 金鑰的 IAM 政策中提供許可，包括其他 AWS 帳戶中的這些政策。這對 [kms:PutKeyPolicy](#) 和 [kms:ScheduleKeyDeletion](#) 等敏感許可特別重要，也適用於決定資料如何受到保護的密碼編譯操作。

限制 CreateKey 許可

僅對需要的主體授予建立金鑰 ([kms:CreateKey](#)) 的許可。建立 KMS 金鑰的主體也會設定其金鑰政策，讓他們可以授予自己和其他人使用和管理其所建立之 KMS 金鑰的許可。當您允許此許可時，請考慮使用[政策條件](#)對其進行限制。例如，您可以使用 [kms:KeySpec](#) 條件來限制對稱加密 KMS 金鑰的許可。

在 IAM 政策中指定 KMS 金鑰

最佳實務是在政策陳述式的 Resource 元素中指定許可套用至其中之每個 KMS 金鑰的[金鑰 ARN](#)。此實務會限制主體所需之 KMS 金鑰的許可。例如，此 Resource 元素只會列出主體需要使用的 KMS 金鑰。

```
"Resource": [
  "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321"
]
```

指定 KMS 金鑰不切實際時，請使用限制對信任 AWS 帳戶 和 區域中 KMS 金鑰存取 Resource 的值，例如 `arn:aws:kms:region:account:key/*`。或限制對信任之所有區域 (*) 中 KMS 金鑰的存取 AWS 帳戶，例如 `arn:aws:kms:*:account:key/*`。

您無法使用[金鑰 ID](#)、[別名名稱](#)，或[別名 ARN](#) 來代表 IAM 政策 Resource 欄位中的 KMS 金鑰。如果您指定別名 ARN，政策會套用至別名，而不是 KMS 金鑰。如需別名 IAM 政策的資訊，請參閱[控制對別名的存取](#)。

避免 IAM 政策中的 "Resource": "*"

明智地使用萬用字元 (*)。在金鑰政策中，Resource 元素中的萬用字元代表金鑰政策所連接的 KMS 金鑰。但是在 IAM 政策中，Resource 元素 ("Resource": "*") 中只有萬用字元會將許可套用至 AWS 帳戶 委託人帳戶有權使用的所有 KMS 金鑰。這可能包括[其他 中的 KMS 金鑰 AWS 帳戶](#)，以及委託人帳戶中的 KMS 金鑰。

例如，若要在另一個 中使用 KMS 金鑰 AWS 帳戶，委託人需要來自外部帳戶中 KMS 金鑰之金鑰政策的許可，以及來自自己帳戶中 IAM 政策的許可。假設某個任意帳戶提供了針對其 KMS 金鑰的 AWS 帳戶 [kms:Decrypt](#) 許可。如果是這樣，您帳戶中為所有 KMS 金鑰 ("Resource": "*") 提供角色 `kms:Decrypt` 許可的 IAM 政策將滿足需求的 IAM 部分。因此，擔任該角色的主體現在可以使用不可信任帳戶中的 KMS 金鑰來解密加密文字。其操作的項目會顯示在兩個帳戶的 CloudTrail 日誌中。

特別是，避免在允許下列 API 操作的政策陳述式中使用 "Resource": "*"。您可以在其他的 KMS 金鑰上呼叫這些操作 AWS 帳戶。

- [DescribeKey](#)
- [GetKeyRotationStatus](#)
- [密碼編譯操作](#)
([Encrypt](#)、[Decrypt](#)、[GenerateDataKey](#)、[GenerateDataKeyPair](#)、[GenerateDataKeyWithoutPlaintext](#)、[GenerateMac](#)、[VerifyMac](#))
- [CreateGrant](#)、[ListGrants](#)、[ListRetirableGrants](#)、[RetireGrant](#)、[RevokeGrant](#)

何時使用 "Resource": "*"

在 IAM 政策中，僅對需要的許可在 Resource 元素中使用萬用字元。只有下列許可需要 "Resource": "*" 元素。

- [kms:CreateKey](#)
- [kms:GenerateRandom](#)
- [kms:ListAliases](#)
- [kms:ListKeys](#)
- 自訂金鑰存放區的許可，例如 [kms:CreateCustomKeyStore](#) 和 [kms:ConnectCustomKeyStore](#)。

 Note

別名操作的許可 ([kms:CreateAlias](#)、[kms:UpdateAlias](#)、[kms>DeleteAlias](#)) 必須連接至別名和 KMS 金鑰。您可以使用 IAM 政策中的 "Resource": "*" 來表示別名和 KMS 金鑰，或在 Resource 元素中指定別名和 KMS 金鑰。如需範例，請參閱 [控制對別名的存取](#)。

本主題中的範例提供有關設計 KMS 金鑰之 IAM 政策的詳細資訊和指引。如需所有 AWS 資源的 IAM 最佳實務，請參閱 [《IAM 使用者指南》中的 IAM 中的安全最佳實務](#)。

在 IAM 政策陳述式中指定 KMS 金鑰

您可以使用 IAM 政策，允許主體使用或管理 KMS 金鑰。在政策陳述式的 Resource 元素中指定 KMS 金鑰。

- 若要在 IAM 政策陳述式中指定 KMS 金鑰，您必須使用其[金鑰 ARN](#)。您不能使用[金鑰 ID](#)、[別名名稱](#)或[別名 ARN](#) 來識別 IAM 政策陳述式中的 KMS 金鑰。

例如："Resource": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"

若要根據其別名來控制對 KMS 金鑰的存取，請使用 [kms:RequestAlias](#) 或 [kms:ResourceAliases](#) 條件索引鍵。如需詳細資訊，請參閱[適用於的 ABAC AWS KMS](#)。

僅在控制別名操作存取的政策陳述式中使用別名 ARN 作為資源，例如 [CreateAlias](#)、[UpdateAlias](#) 或 [DeleteAlias](#)。如需詳細資訊，請參閱[控制對別名的存取](#)。

- 若要在帳戶和區域中指定多個 KMS 金鑰，請在金鑰 ARN 的區域或資源 ID 位置中使用萬用字元 (*)。

例如，若要在帳戶的美國西部 (奧勒岡) 區域中指定所有 KMS 金鑰，請使用 "Resource": "arn:aws:kms:us-west-2:111122223333:key/*"。若要在帳戶的所有區域中指定所有 KMS 金鑰，請使用 "Resource": "arn:aws:kms:*:111122223333:key/*"。

- 若要代表所有 KMS 金鑰，請單獨使用萬用字元 ("*")。將此格式用於不使用任何特定 KMS 金鑰的操作，也就是 [CreateKey](#)、[GenerateRandom](#)、[ListAliases](#) 和 [ListKeys](#)。

在撰寫政策陳述式時，[最佳實務](#)是僅指定主體需要使用的 KMS 金鑰，而不是為其授予所有 KMS 金鑰的存取權。

例如，下列 IAM 政策陳述式允許主體僅針對政策陳述式 Resource 元素中列出的 KMS 金鑰呼叫 [DescribeKey](#)、[GenerateDataKey](#)、[Decrypt](#) 操作。透過金鑰 ARN 指定 KMS 金鑰 (這是最佳實務) 可確保許可僅限於指定的 KMS 金鑰。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:DescribeKey",
      "kms:GenerateDataKey",
      "kms:Decrypt"
    ],
    "Resource": [
      "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "arn:aws:kms:us-west-2:111122223333:key/01234abcd-12ab-34cd-56ef-1234567890ab"
    ]
  }
}
```

```
}
```

若要將許可套用至特定信任的所有 KMS 金鑰 AWS 帳戶，您可以在區域和金鑰 ID 位置中使用萬用字元 (*)。例如，下列政策陳述式可讓主體在兩個可信任範例帳戶中的所有 KMS 金鑰上呼叫指定的操作。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:DescribeKey",
      "kms:GenerateDataKey",
      "kms:GenerateDataKeyPair"
    ],
    "Resource": [
      "arn:aws:kms:*:111122223333:key/*",
      "arn:aws:kms*:444455556666:key/*"
    ]
  }
}
```

您也可以在 Resource 元素中單獨使用萬用字元 ("*")。因為它允許存取帳戶具有使用許可的所有 KMS 金鑰，所以建議主要用於沒有特定 KMS 金鑰的操作，以及 Deny 陳述式。您也可以在只允許較不敏感之唯讀操作的政策陳述式中使用它。若要判斷 AWS KMS 操作是否涉及特定 KMS 金鑰，請在中資料表的資源欄中尋找 KMS 金鑰值[the section called “許可參考”](#)。

例如，下列政策陳述式使用 Deny 效果，以禁止主體在任何 KMS 金鑰上使用指定的操作。它使用 Resource 元素中的萬用字元來代表所有 KMS 金鑰。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
```

```
"Action": [
    "kms:CreateKey",
    "kms:PutKeyPolicy",
    "kms:CreateGrant",
    "kms:ScheduleKeyDeletion"
],
"Resource": "*"
}
```

下列政策陳述式僅使用萬用字元來代表所有 KMS 金鑰。但它只允許不太敏感的唯一讀操作和不適用於任何特定 KMS 金鑰的操作。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:CreateKey",
      "kms:ListKeys",
      "kms:ListAliases",
      "kms:ListResourceTags"
    ],
    "Resource": "*"
  }
}
```

IAM 政策範例

在本節中，您可以找到允許各種 AWS KMS 動作之許可的 IAM 政策範例。

Important

以下政策中的一些許可僅在 KMS 金鑰的金鑰政策也允許時，才會獲得允許。如需詳細資訊，請參閱[許可參考](#)。

如需撰寫及格式化 JSON 政策文件的說明，請參閱《IAM 使用者指南》中的 [IAM JSON 政策參考](#)。

範例

- [允許使用者在 AWS KMS 主控台中檢視 KMS 金鑰](#)
- [允許使用者建立 KMS 金鑰](#)
- [允許使用者使用特定 中的任何 KMS 金鑰來加密和解密 AWS 帳戶](#)
- [允許使用者在特定 和 區域中使用任何 KMS 金鑰來加密 AWS 帳戶 和解密](#)
- [允許使用者使用特定 KMS 金鑰來加密和解密](#)
- [防止使用者停用或刪除任何 KMS 金鑰](#)

允許使用者在 AWS KMS 主控台中檢視 KMS 金鑰

下列 IAM 政策允許使用者唯讀存取 AWS KMS 主控台。具有這些許可的使用者可以檢視其 中的所有 KMS 金鑰 AWS 帳戶，但無法建立或變更任何 KMS 金鑰。

若要檢視 AWS 受管金鑰 和客戶受管金鑰頁面的 KMS 金鑰，主體需要 [kms:ListKeys](#)、[kms:ListAliases](#) 和 [tag:GetResources](#) 許可，即使金鑰沒有標籤或別名。其餘的許可 (特別是 [kms:DescribeKey](#)) 需要在 KMS 金鑰詳細資訊頁面上檢視選用的 KMS 金鑰表資料欄和資料。[iam:ListUsers](#) 和 [iam:ListRoles](#) 許可需要在預設檢視中顯示金鑰政策，而不會出現錯誤。若要檢視自訂金鑰存放區頁面上的資料和自訂金鑰存放區中 KMS 金鑰的詳細資訊，則主體也需要 [kms:DescribeCustomKeyStores](#) 許可。

如果您限制對特定 KMS 金鑰的使用者主控台存取，則主控台會針對不可見的每個 KMS 金鑰顯示錯誤。

此政策包含兩個政策陳述式。第一個政策陳述式中的 Resource 元素允許範例 AWS 帳戶所有區域中所有 KMS 金鑰上的指定許可。主控台檢視器不需要額外的存取權，因為 AWS KMS 主控台只會顯示主體帳戶中的 KMS 金鑰。即使他們有權在其他 中檢視 KMS 金鑰，也是如此 AWS 帳戶。其餘 AWS KMS 和 IAM 許可需要 "Resource": "*" 元素，因為它們不適用於任何特定的 KMS 金鑰。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ReadOnlyAccessForAllKMSKeysInAccount",
      "Effect": "Allow",
      "Action": [
```

```

        "kms:GetPublicKey",
        "kms:GetKeyRotationStatus",
        "kms:GetKeyPolicy",
        "kms:DescribeKey",
        "kms:ListKeyPolicies",
        "kms:ListResourceTags",
        "tag:GetResources"
    ],
    "Resource": "arn:aws:kms:*:111122223333:key/*"
},
{
    "Sid": "ReadOnlyAccessForOperationsWithNoKMSKey",
    "Effect": "Allow",
    "Action": [
        "kms:ListKeys",
        "kms:ListAliases",
        "iam:ListRoles",
        "iam:ListUsers"
    ],
    "Resource": "*"
}
]
}

```

允許使用者建立 KMS 金鑰

下列 IAM 政策可讓使用者建立各類型的 KMS 金鑰。Resource 元素的值是 `*`，* 因為 CreateKey 操作不使用任何特定 AWS KMS 資源 (KMS 金鑰或別名)。

若要限制使用者僅能使用特定類型的 KMS 金鑰，請使用 [kms:KeySpec](#)、[kms:KeyUsage](#) 和 [kms:KeyOrigin](#) 條件金鑰。

JSON

```

{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "kms:CreateKey",
        "Resource": "*"
    }
}

```


}

建立金鑰的主體可能需要一些相關許可。

- `kms:PutKeyPolicy` – 擁有 `kms:CreateKey` 許可的主體可以設定 KMS 金鑰的初始金鑰政策。不過，`CreateKey` 呼叫者必須有 [kms:PutKeyPolicy](#) 許可，這可讓其變更 KMS 金鑰政策，或者他們必須指定 `CreateKey` 的 `BypassPolicyLockoutSafetyCheck` 參數，但不建議這樣操作。`CreateKey` 呼叫者可以從 IAM 政策為 KMS 金鑰取得 `kms:PutKeyPolicy` 許可，或者其可以在所建立之 KMS 金鑰的金鑰政策中包含此許可。
- `kms:TagResource` – 若要為 `CreateKey` 操作中的 KMS 金鑰新增標籤，`CreateKey` 呼叫者必須有 IAM 政策中的 [kms:TagResource](#) 許可。在新 KMS 金鑰的金鑰政策中包含此許可並不足夠。但是，如果 `CreateKey` 呼叫者在初始金鑰政策中包含 `kms:TagResource`，則他們可以在建立 KMS 金鑰之後，在個別呼叫中新增標籤。
- `kms:CreateAlias` — 在 AWS KMS 主控台中建立 KMS 金鑰的委託人必須在 KMS 金鑰和別名上擁有 [kms:CreateAlias](#) 許可。(主控台進行兩個呼叫；一個至 `CreateKey` 和一個至 `CreateAlias`)。您必須在 IAM 政策中提供別名許可。您可以在金鑰政策、IAM 政策中提供 KMS 金鑰許可。如需詳細資訊，請參閱[控制對別名的存取](#)。

除了之外 `kms:CreateKey`，下列 IAM 政策還提供中所有 KMS 金鑰的 `kms:TagResource` 許可，AWS 帳戶以及帳戶所有別名的 `kms:CreateAlias` 許可。其中還包含一些只能在 IAM 政策中提供的有用唯讀許可。

此 IAM 政策並不包括 `kms:PutKeyPolicy` 許可或可在金鑰政策中設定的任何其他許可。[最佳實務](#)是在金鑰政策中設定這些許可，在其中它們會專門套用至一個 KMS 金鑰。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPermissionsForParticularKMSKeys",
      "Effect": "Allow",
      "Action": "kms:TagResource",
      "Resource": "arn:aws:kms:*:111122223333:key/*"
    },
    {
```

```

    "Sid": "IAMPermissionsForParticularAliases",
    "Effect": "Allow",
    "Action": "kms:CreateAlias",
    "Resource": "arn:aws:kms:*:111122223333:alias/*"
  },
  {
    "Sid": "IAMPermissionsForAllKMSKeys",
    "Effect": "Allow",
    "Action": [
      "kms:CreateKey",
      "kms:ListKeys",
      "kms:ListAliases"
    ],
    "Resource": "*"
  }
]
}

```

允許使用者使用特定 中的任何 KMS 金鑰來加密和解密 AWS 帳戶

下列 IAM 政策允許使用者使用 AWS 帳戶 111122223333 中的任何 KMS 金鑰來加密和解密資料。

JSON

```

{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:Encrypt",
      "kms:Decrypt"
    ],
    "Resource": "arn:aws:kms:*:111122223333:key/*"
  }
}

```

允許使用者在特定 和 區域中使用任何 KMS 金鑰來加密 AWS 帳戶 和解密

下列 IAM 政策允許使用者在美國西部（奧勒岡）區域中，使用 中的任何 AWS 帳戶 111122223333 KMS 金鑰來加密和解密資料。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:Encrypt",
      "kms:Decrypt"
    ],
    "Resource": [
      "arn:aws:kms:us-west-2:111122223333:key/*"
    ]
  }
}
```

允許使用者使用特定 KMS 金鑰來加密和解密

以下 IAM 政策允許使用者使用 Resource 元素中指定的兩個 KMS 金鑰來加密和解密資料。若要在 IAM 政策陳述式中指定 KMS 金鑰，您必須使用 KMS 金鑰的[金鑰 ARN](#)。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:Encrypt",
      "kms:Decrypt"
    ],
    "Resource": [
      "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "arn:aws:kms:us-west-2:111122223333:key/01234abc-d12a-b34c-d56e-f1234567890a"
    ]
  }
}
```

防止使用者停用或刪除任何 KMS 金鑰

以下 IAM 政策能防止使用者停用或刪除任何 KMS 金鑰，即使另一個 IAM 政策或金鑰政策允許這些許可。明確拒絕許可的政策會覆寫所有其他政策，即使那些政策明確允許相同的許可。如需詳細資訊，請參閱[故障診斷 AWS KMS 許可](#)。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "kms:DisableKey",
        "kms:ScheduleKeyDeletion"
      ],
      "Resource": "*"
    }
  ]
}
```

中的資源控制政策 AWS KMS

資源控制政策 (RCPs) 是一種組織政策，可用來對組織中 AWS 的資源強制執行預防性控制。RCPs 可協助您集中限制大規模對 AWS 資源的外部存取。RCPs 補充服務控制政策 (SCPs)。雖然 SCPs 可用來集中設定組織中 IAM 角色和使用者的最大許可，但 RCPs 可用來集中設定組織中 AWS 資源的最大許可。

您可以使用 RCPs 來管理組織中客戶受管 KMS 金鑰的許可。僅 RCPs 不足以授與許可給客戶受管金鑰。RCP 不會授予任何許可。RCP 會針對身分可對受影響帳戶中的資源採取的動作，定義許可護欄或設定限制。管理員仍然必須將身分型政策連接到 IAM 角色或使用者，或金鑰政策，以實際授予許可。

Note

組織中的資源控制政策不適用於 [AWS 受管金鑰](#)。

AWS 受管金鑰 是由 AWS 服務代表您建立、管理和使用，您無法變更或管理其許可。

進一步了解

- 如需 RCPs 的一般資訊，請參閱AWS Organizations 《使用者指南》中的[資源控制政策](#)。
- 如需如何定義 RCPs 的詳細資訊，包括範例，請參閱AWS Organizations 《使用者指南》中的 [RCP 語法](#)。

下列範例示範如何使用 RCP 來防止外部主體存取組織中的客戶受管金鑰。此政策只是範例，您應該量身打造它來滿足您的獨特業務和安全需求。例如，您可能想要自訂您的政策，以允許您的業務合作夥伴存取。如需詳細資訊，請參閱[資料周邊政策範例儲存庫](#)。

Note

kms:RetireGrant 許可在 RCP 中無效，即使 Action 元素指定星號 (*) 做為萬用字元。如需如何 kms:RetireGrant 判斷許可的詳細資訊，請參閱 [淘汰和撤銷授予](#)。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "RCPEnforceIdentityPerimeter",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "kms:*",
      "Resource": "*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:PrincipalOrgID": "my-org-id"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "false"
        }
      }
    }
  ]
}
```

下列範例 RCP 要求 AWS 服務主體只有在請求來自您的組織時，才能存取您的客戶受管 KMS 金鑰。此政策只會將控制項套用至aws:SourceAccount已存在的請求。這可確保不需要使用的服務整合aws:SourceAccount不會受到影響。如果請求內容中存在 aws:SourceAccount，則 Null條件會評估為 true，導致強制執行aws:SourceOrgID金鑰。

如需混淆代理人問題的詳細資訊，請參閱 [《IAM 使用者指南》中的混淆代理人問題](#)。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "RCPEnforceConfusedDeputyProtection",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "kms:*",
      "Resource": "*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceOrgID": "my-org-id"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
        },
        "Null": {
          "aws:SourceAccount": "false"
        }
      }
    }
  ]
}
```

在 中授予 AWS KMS

授權是一個政策工具，允許 [AWS 主體](#) 使用密碼編譯操作中的 KMS 金鑰。它也可以讓其檢視 KMS 金鑰 (DescribeKey)，並建立和管理授予。當授權存取 KMS 金鑰時，會考慮與 [金鑰政策](#) 和 [IAM 政策](#) 一

起授予。授予通常用於臨時許可，因為您可以建立授予、使用其許可並刪除授予，而無需變更金鑰政策或 IAM 政策。

與整合的 AWS 服務通常會使用授權 AWS KMS 來加密靜態資料。服務會代表帳戶中的使用者建立授予、使用其許可，並在其任務完成後立即淘汰授予。如需 AWS 服務使用授權方式的詳細資訊，請參閱服務使用者指南或開發人員指南中的靜態加密主題。

授予是非常靈活和有用的存取控制機制。當您建立 KMS 金鑰的授予時，只要授予中的所有指定條件都滿足，授予就會允許承授者主體呼叫 KMS 金鑰上指定的授予操作。

- 每個授予只允許存取一個 KMS 金鑰。您可以在不同的 AWS 帳戶中為 KMS 金鑰建立授予。
- 授予可以允許存取 KMS 金鑰，但不能拒絕存取。
- 每個授予都有一個[承授者主體](#)。承授者主體可以在 AWS 帳戶與 KMS 金鑰相同的 或不同帳戶中代表一或多個身分。
- 授予只能允許[授予操作](#)。授予操作必須由授予中的 KMS 金鑰支援。若您指定不支援的操作，則 [CreateGrant](#) 請求會失敗，`ValidationError` 為例外狀況。
- 承授者主體可以在不指定授予的情況下使用授予給予的許可，就如同許可來自金鑰政策或 IAM 政策一樣。不過，由於 AWS KMS API 遵循[最終一致性](#)模型，當您建立、淘汰或撤銷授予時，在變更全面可用之前，可能會有短暫的延遲 AWS KMS。若要立即使用授予中的許可，請[使用授予字符](#)。
- 授權主體可以刪除授予 (對其進行[淘汰](#)或[撤銷](#))。刪除授予會消除授予允許的所有許可。您不需要弄清楚要新增或移除哪些政策，即可復原授予。
- AWS KMS 會限制每個 KMS 金鑰的授予數量。如需詳細資訊，請參閱[每個 KMS 金鑰的授予：50,000](#)。

建立授予和給予他人建立授予的許可時，請小心謹慎。建立授予的許可具有安全影響，就像允許 [kms:PutKeyPolicy](#) 許可來設定政策。

- 具有為 KMS 金鑰 (`kms:CreateGrant`) 建立授予許可的使用者可以使用授予來允許使用者和角色使用 KMS 金鑰，包括 AWS 服務。委託人可以是您自己的身分 AWS 帳戶，也可以是不同帳戶或組織中的身分。
- 授予只能允許操作的子集 AWS KMS。您可以使用授予來允許主體檢視 KMS 金鑰、在密碼編譯操作中使用 KMS 金鑰，以及建立和淘汰授予。如需詳細資訊，請參閱[授予操作](#)。您也可以使用[授予限制條件](#)來限制授予對稱加密金鑰中的許可。
- 主體可以從金鑰政策或 IAM 政策取得建立授予的許可。針對 KMS 金鑰的任何[授與操作](#)，透過政策取得 `kms:CreateGrant` 許可的主體均可建立授與。這些主體不需擁有他們對金鑰授與的許可。當允許政策中的 `kms:CreateGrant` 許可時，您可以使用[政策條件](#)來限制此許可。

- 主體也可以從授予取得建立授予的許可。這些委託人只能委派其獲授予的許可，即使他們擁有政策的其他許可。如需詳細資訊，請參閱[授與 CreateGrant 許可](#)。

授予概念

若要有效地使用授予，您需要了解 AWS KMS 使用的條款和概念。

授予限制條件

限制授予中許可的條件。目前，AWS KMS 支援根據密碼編譯操作請求中的[加密內容](#)來授予限制。如需詳細資訊，請參閱[使用授予限制條件](#)。

授予 ID

用於 KMS 金鑰的授予唯一識別符。您可以使用授予 ID，與[金鑰識別符](#)一起，以識別 [RetireGrant](#) 或 [RevokeGrant](#) 請求。

授予操作

您可以在授予中允許 AWS KMS 的操作。如果您指定其他操作，則 [CreateGrant](#) 請求會失敗，Validation error 為例外狀況。這些也是接受[授予字串](#)的操作。如需有關這些許可的詳細資訊，請參閱 [AWS KMS 許可](#)。

這些授予操作實際上代表使用操作的許可。因此，對於 ReEncrypt 操作，您可以指定 ReEncryptFrom、ReEncryptTo，或兩者的 ReEncrypt*。

授予操作包括：

- 密碼編譯操作
 - [解密](#)
 - [DeriveSharedSecret](#)
 - [加密](#)
 - [GenerateDataKey](#)
 - [GenerateDataKeyPair](#)
 - [GenerateDataKeyPairWithoutPlaintext](#)
 - [GenerateDataKeyWithoutPlaintext](#)
 - [GenerateMac](#)
 - [ReEncryptFrom](#)
 - [ReEncryptTo](#)

- [符號](#)
- [確認](#)
- [VerifyMac](#)
- 其他操作
 - [CreateGrant](#)
 - [DescribeKey](#)
 - [GetPublicKey](#)
 - [RetireGrant](#)

您允許的授予作業必須由授予中的 KMS 金鑰支援。若您指定不支援的操作，則 [CreateGrant](#) 請求會失敗，`ValidationError` 為例外狀況。例如，對稱加密 KMS 金鑰的授予不允許 [Sign](#)、[Verify](#)、[GenerateMac](#) 或 [VerifyMac](#) 操作。非對稱 KMS 金鑰的授予不允許任何產生資料金鑰或資料金鑰對的操作。

授予字符

AWS KMS API 遵循[最終一致性](#)模型。當您建立授權時，在該變更適用於整個 AWS KMS 之前，可能會有短暫延遲。變更傳播到整個系統通常需要不到幾秒鐘的時間，但在某些情況下可能需要幾分鐘。如果您在整個系統中完全傳播授權之前嘗試使用授權，則可能會收到拒絕存取錯誤。授予字符可讓您參考授予並立即使用授予許可。

授予字符是唯一的，非秘密的，可變長度的，base64 編碼的字串，代表授予。您可以使用授予字符來識別任何[授予操作](#)中的授予。但是，由於字符值是雜湊摘要，因此它不會顯示有關授予的任何詳細資訊。

授權字符設計為只能在整個 AWS KMS 中完全傳播授權之後才能使用。在此之後，[承授者主體](#)可以使用授予中的許可，而不提供授予字符或任何其他授予的證據。您可以隨時使用授予字符，但一旦授予最終一致，AWS KMS 會使用授予來確定許可，而不是授予字符。

例如，下列命令會呼叫 [GenerateDataKey](#) 操作。它使用授予字符來表示授予呼叫者 (承授者主體) 許可來呼叫指定 KMS 金鑰上的 `GenerateDataKey`。

```
$ aws kms generate-data-key \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --key-spec AES_256 \
    --grant-token $token
```

您也可以使用授予字符來識別管理授予之操作中的授予。例如，[淘汰主體](#)可以在 [RetireGrant](#) 操作的呼叫中使用授予字符。

```
$ aws kms retire-grant \  
    --grant-token $token
```

CreateGrant 是傳回授予字符的唯一操作。您無法從任何其他 AWS KMS 操作或從 CreateGrant 操作的 [CloudTrail 日誌事件](#) 取得授予字符。[ListGrants](#) 和 [ListRetirableGrants](#) 操作會傳回 [授予 ID](#)，但不是授予字符。

如需詳細資訊，請參閱[使用授予字符](#)。

承授者主體

取得授予中所指定許可的身分。每個授予都有一個承授者主體，但承授者主體可以代表多個身分。

承授者委託人可以是任何 AWS 委託人，包括 AWS 帳戶（根）、[IAM 使用者](#)、[IAM 角色](#)、[聯合角色或使用者](#)，或擔任角色的使用者。承授者主體可以在與 KMS 金鑰相同的帳戶中，也可以在不同的帳戶中。不過，承授者主體不能是[服務主體](#)、[IAM 群組](#)，或 [AWS 組織](#)。

Note

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的安全性最佳實務](#)。

淘汰授予

終止授予。當您完成使用許可時，就會淘汰授予。

撤銷和淘汰授予都會刪除授予。但是，由授予中指定的主體完成淘汰動作。撤銷通常是由金鑰管理員完成。如需詳細資訊，請參閱[淘汰和撤銷授予](#)。

淘汰主體

可以[淘汰授予](#)的主體。您可以在授予中指定淘汰的主體，但不是必需的。淘汰委託人可以是任何 AWS 委託人，包括 AWS 帳戶 IAM 使用者、IAM 角色、聯合身分使用者和擔任的角色使用者。淘汰的主體可以在與 KMS 金鑰相同的帳戶中，也可以在不同的帳戶中。

Note

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的安全性最佳實務](#)。

除了授予中指定的淘汰委託人之外，授予也可以由建立授予 AWS 帳戶 的 淘汰。[承授者主體](#) 可以淘汰授予 (如果授予允許 `RetireGrant` 操作)。此外，AWS 帳戶 或 是淘汰主體 AWS 帳戶 的 可以將淘汰授予的許可委派給相同 中的 IAM 主體 AWS 帳戶。如需詳細資訊，請參閱[淘汰和撤銷授予](#)。

撤銷 (授予)

終止授予。您撤銷授予，以主動拒絕授予允許的許可。

撤銷和淘汰授予都會刪除授予。但是，由授予中指定的主體完成淘汰動作。撤銷通常是由金鑰管理員完成。如需詳細資訊，請參閱[淘汰和撤銷授予](#)。

最終一致性 (用於授予)

AWS KMS API 遵循[最終一致性](#)模型。當您建立、淘汰或撤銷授權時，在該變更適用於整個 AWS KMS 之前，可能會有短暫延遲。變更傳播到整個系統通常需要不到幾秒鐘的時間，但在某些情況下可能需要幾分鐘。

如果遇到非預期的錯誤，您可能會發現這個短暫的延遲。例如，如果您在完全知道授予之前嘗試管理新的授予或使用新授予中的許可 AWS KMS，您可能會收到存取遭拒錯誤。如果您淘汰或撤銷授予，承授者主體可能仍然可以在短期內使用其許可，直到授予完全刪除為止。典型的策略是重試請求，而 AWS SDKs 包含自動退避和重試邏輯。

AWS KMS 具有可減輕此短暫延遲的功能。

- 若要立即使用新授予中的許可，請使用[授予字符](#)。您可以使用授予字符來引用任何[授予操作](#)中的授予。如需說明，請參閱[使用授予字符](#)。
- [CreateGrant](#) 操作具有 `Name` 參數，可防止重試操作建立重複授予。

Note

授予許可會取代授予的有效性，直到服務中的所有端點都更新為新的授予狀態為止。在大多數情況下，最終一致性將在五分鐘內達成。

如需詳細資訊，請參閱 [《AWS KMS 最終一致性》](#)。

AWS KMS 授予的最佳實務

AWS KMS 在建立、使用和管理授予時，建議下列最佳實務。

- 將授予中的許可限制為承授者主體需要的許可。使用[最低權限存取](#)的原則。
- 使用特定承授者主體 (例如 IAM 角色)，並讓承授者主體僅使用其所需 API 操作的許可。

- 使用加密內容[授予限制條件](#)，以確保呼叫者使用 KMS 金鑰來達到預期目的。如需如何在請求中使用加密內容來保護資料的詳細資訊，請參閱 AWS 安全部落格中的[如何使用 AWS Key Management Service 和 EncryptionContext 來保護加密資料的完整性](#)。

i Tip

盡可能使用 [EncryptionContextEqual](#) 授予限制條件。[EncryptionContextSubset](#) 授予限制條件更難以正確使用。如果您需要使用，請仔細閱讀文件並測試授予限制條件，以確保其按預期工作。

- 刪除重複的授予。重複授予具有相同的金鑰 ARN、API 動作、承受者主體、加密內容和名稱。如果您淘汰或撤銷原始授予但保留重複項目，則剩餘的重複授予會構成非預期的權限升級。為了避免在重試 CreateGrant 請求時重複授予，請使用 [Name 參數](#)。若要偵測重複授予，請使用 [ListGrants](#) 操作。如果意外建立了重複授予，請儘快淘汰或撤銷該授予。

i Note

[AWS 受管金鑰](#) 授予可能看起來像重複項，但具有不同的承受者主體。

GranteePrincipal 回應中的 ListGrants 欄位通常包含授與的承受者主體。不過，當授予中的承受者委託人是 AWS 服務時，GranteePrincipal 欄位會包含[服務委託人](#)，這可能代表數個不同的承受者委託人。

- 請記住，授予不會自動過期。不再需要許可時，即刻[淘汰或撤銷授予](#)。未刪除的授予可能會對加密資源造成安全風險。

控制對授予的存取

您可以控制對在金鑰政策、IAM 政策和授予中建立和管理授予之操作的存取。從授予取得 CreateGrant 許可的主體具有[更有限的授予許可](#)。

API 操作	金鑰政策或 IAM 政策	授權
CreateGrant	✓	✓
ListGrants	✓	-
ListRetirableGrants	✓	-

API 操作	金鑰政策或 IAM 政策	授權
淘汰授予	(有限。請參閱 淘汰和撤銷授予)	✓
RevokeGrant	✓	-

當您使用金鑰政策或 IAM 政策來控制對建立和管理授予之操作的存取時，您可以使用下列一或多個政策條件來限制許可。AWS KMS 支援下列所有授予相關條件金鑰。如需詳細資訊和範例，請參閱 [AWS KMS 條件索引鍵](#)。

[kms:GrantConstraintType](#)

僅當授予包含指定的[授予限制條件](#)時，才允許主體建立授予。

[kms:GrantIsForAWSResource](#)

RevokeGrant 只有在與 [整合 AWS 的服務 AWS KMS](#) 代表委託人傳送請求時 CreateGrant，允許委託人呼叫 ListGrants、或。

[kms:GrantOperations](#)

允許主體建立授予，但將授予限制為指定的操作。

[kms:GranteePrincipal](#)

允許主體僅針對指定的[承授者主體](#)建立授予。

[kms:RetiringPrincipal](#)

僅當授予指定特定的[淘汰主體](#)時，才允許主體建立授予。

建立授予

建立授予之前，請先了解自訂授予的選項。您可以使用授予限制條件以限制授予中的許可。此外，了解授予 CreateGrant 許可。從授予取得建立授予之許可的主體，在其可以建立的授予中受到限制。

主題

- [建立授與](#)
- [授與 CreateGrant 許可](#)

建立授與

若要建立授予，請呼叫 [CreateGrant](#) 操作。指定 KMS 金鑰、[承授者主體](#)和允許的[授予操作](#)清單。您也可以指定選用的[淘汰主體](#)。若要自訂授與，請使用選用 Constraints 參數來定義[授與限制](#)。

當您建立、淘汰或撤銷授予時，可能會有短暫的延遲 (通常不到五分鐘)，然後才會在整個 AWS KMS 中可用。如需詳細資訊，請參閱[最終一致性模式 \(授權\)](#)。

例如，下列 CreateGrant 命令會建立授予，以允許被授權擔任 keyUserRole 角色的使用者對指定的[對稱 KMS 金鑰](#)呼叫 [Decrypt](#) 操作。授權會使用 RetiringPrincipal 參數指定可淘汰授權的主體。同時也包含授權限制條件，僅當請求中的[加密內容](#)包含 "Department": "IT" 時，才允許許可。

```
$ aws kms create-grant \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --grantee-principal arn:aws:iam::111122223333:role/keyUserRole \  
  --operations Decrypt \  
  --retiring-principal arn:aws:iam::111122223333:role/adminRole \  
  --constraints EncryptionContextSubset={Department=IT}
```

如果您的程式碼重試 CreateGrant 操作，或使用[自動重試請求的AWS 開發套件](#)，則請使用選用[名稱](#)參數，以防止建立重複授予。如果 AWS KMS 取得與現有授予具有相同屬性的授予CreateGrant請求，包括名稱，它會將請求識別為重試，並且不會建立新的授予。您無法使用 Name 值以識別任何 AWS KMS 操作中的授予。

Important

請勿在授權名稱包含機密或敏感資訊。其可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

```
$ aws kms create-grant \  
  --name IT-1234abcd-keyUserRole-decrypt \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --grantee-principal arn:aws:iam::111122223333:role/keyUserRole \  
  --operations Decrypt \  
  --retiring-principal arn:aws:iam::111122223333:role/adminRole \  
  --constraints EncryptionContextSubset={Department=IT}
```

如需示範如何以多種程式設計語言建立授予的程式碼範例，請參閱 [CreateGrant 搭配 AWS SDK 或 CLI 使用](#)。

使用授予限制條件

[授予限制條件](#)會在為承授者主體提供授予的許可上設定條件。授予限制條件會取代[金鑰政策](#)或 [IAM 政策](#)中的[條件索引鍵](#)。每個授予限制條件值最多可以包含 8 個加密內容對。每個授予限制條件中的加密內容值不能超過 384 個字元。

Important

請勿在此欄位包含機密或敏感資訊。此欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

AWS KMS 支援兩個授予限制，EncryptionContextEquals 和 EncryptionContextSubset，這兩者在密碼編譯操作的請求中建立[加密內容](#)的要求。

加密內容授予限制條件是設計用於與擁有加密內容參數的[授予操作](#)搭配使用。

- 加密內容限制條件僅在對稱加密 KMS 金鑰的授予中有效。其他 KMS 金鑰的密碼編譯操作不支援加密內容。
- 加密內容限制條件會忽略 DescribeKey 和 RetireGrant 操作。DescribeKey 和 RetireGrant 沒有加密內容參數，但您可以將這些操作包含在具有加密內容限制條件的授予中。
- 您可以在 CreateGrant 操作的授予中使用加密內容限制條件。加密內容限制條件要求使用 CreateGrant 許可建立的任何授予具有同樣嚴格或更嚴格的加密內容限制條件。

AWS KMS 支援下列加密內容授予限制。

EncryptionContextEquals

使用 EncryptionContextEquals，以指定允許請求的確切加密內容。

EncryptionContextEquals 需要請求中的加密內容對完全符合 (包括大小寫) 授予限制條件中的加密內容。此對組可以任何順序顯示，但每個對組中的金鑰和值不能改變。

例如，如果 EncryptionContextEquals 授予限制條件需要 "Department": "IT" 加密內容對，則僅當請求中的加密內容正好是 "Department": "IT" 時，授予才會允許指定類型的請求。

EncryptionContextSubset

使用 EncryptionContextSubset 來要求請求包含特定的加密內容對。

EncryptionContextSubset 需要請求包含授予限制條件中的所有加密內容對 (完全符合，包括大小寫)，但請求也可以有其他加密內容對。此對組可以任何順序顯示，但每個對組中的金鑰和值不能改變。

例如，如果 EncryptionContextSubset 授予限制條件需要 Department=IT 加密內容對，則當請求中的加密內容為 "Department": "IT" 時，或包含 "Department": "IT" 與其他加密內容對 (例如 "Department": "IT", "Purpose": "Test") 時，授予才會允許指定類型的請求。

若要指定對稱加密 KMS 金鑰授予中的加密內容限制條件，請使用 [CreateGrant](#) 操作中的 Constraints 參數。此命令建立的授予會給予被授權擔任 keyUserRole 角色的使用者呼叫 [Decrypt](#) 操作的許可。但是，僅當 Decrypt 請求中的加密內容為 "Department": "IT" 加密內容對時，該許可才會生效。

```
$ aws kms create-grant \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --grantee-principal arn:aws:iam::111122223333:role/keyUserRole \
  --operations Decrypt \
  --retiring-principal arn:aws:iam::111122223333:role/adminRole \
  --constraints EncryptionContextEquals={Department=IT}
```

產生的授與看起來如下。請注意，授予給 keyUserRole 角色的許可只有在 Decrypt 請求使用授予限制條件中指定的相同加密內容對時才有效。如要尋找 KMS 金鑰的授予，請使用 [ListGrants](#) 操作。

```
$ aws kms list-grants --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
  "Grants": [
    {
      "Name": "",
      "IssuingAccount": "arn:aws:iam::111122223333:root",
      "GrantId":
"abcde1237f76e4ba7987489ac329fbfba6ad343d6f7075dbd1ef191f0120514a",
      "Operations": [
        "Decrypt"
      ],
      "GranteePrincipal": "arn:aws:iam::111122223333:role/keyUserRole",
      "Constraints": {
        "EncryptionContextEquals": {
          "Department": "IT"
        }
      }
    }
  ]
}
```

```

    },
    "CreationDate": 1568565290.0,
    "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "RetiringPrincipal": "arn:aws:iam::111122223333:role/adminRole"
  }
]
}

```

為了滿足 `EncryptionContextEquals` 授予限制條件，`Decrypt` 操作請求中的加密內容必須是 `"Department": "IT"` 對。來自承授者主體的類似下列請求應當滿足 `EncryptionContextEquals` 授予限制條件。

```

$ aws kms decrypt \
  --key-id arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab \
  --ciphertext-blob fileb://encrypted_msg \
  --encryption-context Department=IT

```

當授予限制條件為 `EncryptionContextSubset`，請求中的加密內容對必須包含授予限制條件中的加密內容對，但請求也可以包含其他加密內容對。下列授予限制條件需要請求中的其中一個加密內容對為 `"Department": "IT"`。

```

"Constraints": {
  "EncryptionContextSubset": {
    "Department": "IT"
  }
}

```

來自承授者主體的類似下列請求應當同時滿足此範例中的 `EncryptionContextEqual` 和 `EncryptionContextSubset` 授予限制條件。

```

$ aws kms decrypt \
  --key-id arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab \
  --ciphertext-blob fileb://encrypted_msg \
  --encryption-context Department=IT

```

然而，來自承授者主體的類似下列請求應當滿足 `EncryptionContextSubset` 授予限制條件，但它無法滿足 `EncryptionContextEquals` 授予限制條件。

```
$ aws kms decrypt \
  --key-id arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab \
  --ciphertext-blob fileb://encrypted_msg \
  --encryption-context Department=IT,Purpose=Test
```

AWS 服務通常會在授予中使用加密內容限制，授予他們在您的 中使用 KMS 金鑰的許可 AWS 帳戶。例如，Amazon DynamoDB 使用類似下列授予來取得在您帳戶中為 DynamoDB 使用 [AWS 受管金鑰](#) 的許可。此 EncryptionContextSubset 授與中的授與限制只有在請求中的加密內容包含 "subscriberID": "111122223333" 和 "tableName": "Services" 對時，才會讓授與中的許可生效。此授予限制條件表示授予允許 DynamoDB 僅針對您 AWS 帳戶中的特定資料表使用指定的 KMS 金鑰。

若要取得此輸出，請在您帳戶中 AWS 受管金鑰 適用於 DynamoDB 的 上執行 [ListGrants](#) 操作。

```
$ aws kms list-grants --key-id 0987dcba-09fe-87dc-65ba-ab0987654321

{
  "Grants": [
    {
      "Operations": [
        "Decrypt",
        "Encrypt",
        "GenerateDataKey",
        "ReEncryptFrom",
        "ReEncryptTo",
        "RetireGrant",
        "DescribeKey"
      ],
      "IssuingAccount": "arn:aws:iam::111122223333:root",
      "Constraints": {
        "EncryptionContextSubset": {
          "aws:dynamodb:tableName": "Services",
          "aws:dynamodb:subscriberId": "111122223333"
        }
      },
      "CreationDate": 1518567315.0,
      "KeyId": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321",
      "GranteePrincipal": "dynamodb.us-west-2.amazonaws.com",
      "RetiringPrincipal": "dynamodb.us-west-2.amazonaws.com",
      "Name": "8276b9a6-6cf0-46f1-b2f0-7993a7f8c89a",
```

```

      "GrantId":
        "1667b97d27cf748cf05b487217dd4179526c949d14fb3903858e25193253fe59"
    }
  ]
}

```

授與 CreateGrant 許可

授予可以包含呼叫 CreateGrant 操作的許可。但是，當[承授者主體](#)從授予 (而非政策) 取得呼叫 CreateGrant 的許可時，該許可會受到限制。

- 承授者主體只能建立允許父項授予中部分或全部操作的授予。
- 其所建立授予中的[授予限制條件](#)必須至少與父項授予一樣嚴格。

這些限制不適用於從政策取得 CreateGrant 許可的主體，雖然其許可可以受到[政策條件](#)限制。

例如，假設有一個授與允許承授者主體呼叫 GenerateDataKey、Decrypt 和 CreateGrant 操作。我們稱之為授予，允許 CreateGrant 許可父項授予。

```

# The original grant in a ListGrants response.
{
  "Grants": [
    {
      "KeyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "CreationDate": 1572216195.0,
      "GrantId": "abcde1237f76e4ba7987489ac329fbfba6ad343d6f7075dbd1ef191f0120514a",
      "Operations": [
        "GenerateDataKey",
        "Decrypt",
        "CreateGrant"
      ],
      "RetiringPrincipal": "arn:aws:iam::111122223333:role/adminRole",
      "Name": "",
      "IssuingAccount": "arn:aws:iam::111122223333:root",
      "GranteePrincipal": "arn:aws:iam::111122223333:role/keyUserRole",
      "Constraints": {
        "EncryptionContextSubset": {
          "Department": "IT"
        }
      }
    },
  ],
}

```

```
    }
  ]
}
```

承授者主體 `exampleUser` 可以使用此許可來建立授予，其中包含原始授予中指定的任何操作子集，例如 `CreateGrant` 和 `Decrypt`。子項授予不能包含其他操作，例如 `ScheduleKeyDeletion` 或 `ReEncrypt`。

此外，子項授予中的[授予限制條件](#)必須與父項授予中的限制程度相等嚴格或比之更嚴格。例如，子授與可以新增對組到父授與的 `EncryptionContextSubset` 限制，但不能將其移除。子授與可將 `EncryptionContextSubset` 限制變更為 `EncryptionContextEquals` 限制，但不能反向。

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 中的安全性最佳實務](#)。

例如，承授者主體可以使用其從父項授予中取得的 `CreateGrant` 許可，建立以下子項授予。子項授予的操作是父項授予操作的子集，授予限制條件更嚴格。

```
# The child grant in a ListGrants response.
{
  "Grants": [
    {
      "KeyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "CreationDate": 1572249600.0,
      "GrantId": "fedcba9999c1e2e9876abcde6e9d6c9b6a1987650000abcee009abcdef40183f",
      "Operations": [
        "CreateGrant"
        "Decrypt"
      ]
      "RetiringPrincipal": "arn:aws:iam::111122223333:user/exampleUser",
      "Name": "",
      "IssuingAccount": "arn:aws:iam::111122223333:root",
      "GranteePrincipal": "arn:aws:iam::111122223333:user/anotherUser",
      "Constraints": {
        "EncryptionContextEquals": {
          "Department": "IT"
        }
      },
    }
  ]
}
```

```
}
```

子項授予的承授者主體 `anotherUser`，可以使用其 `CreateGrant` 許可建立授予。然而，`anotherUser` 建立的授予必須在其父項授予或子集中包含操作，並且授予限制條件必須相同或更嚴格。

檢視授予

若要檢視授權，請使用 [ListGrants](#) 操作。您必須指定授予套用的 KMS 金鑰。您也可以透過授予 ID 或承授者主體篩選授予清單。如需更多範例，請參閱[ListGrants 搭配 AWS SDK 或 CLI 使用](#)。

若要檢視具有特定[淘汰委託](#)人之 AWS 帳戶 和 區域中的所有授予，請使用 [ListRetirableGrants](#)。回應包括每項授予的詳細資訊。

Note

`GranteePrincipal` 回應中的 `ListGrants` 欄位通常包含授與的承授者主體。不過，當授予中的承授者委託人是 AWS 服務時，`GranteePrincipal` 欄位會包含[服務委託](#)人，這可能代表數個不同的承授者委託人。

例如，下列命令會列出 KMS 金鑰的所有授予。

```
$ aws kms list-grants --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
  "Grants": [
    {
      "KeyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "CreationDate": 1572216195.0,
      "GrantId": "abcde1237f76e4ba7987489ac329fbfba6ad343d6f7075dbd1ef191f0120514a",
      "Constraints": {
        "EncryptionContextSubset": {
          "Department": "IT"
        }
      },
      "RetiringPrincipal": "arn:aws:iam::111122223333:role/adminRole",
      "Name": "",
      "IssuingAccount": "arn:aws:iam::111122223333:root",
      "GranteePrincipal": "arn:aws:iam::111122223333:user/exampleUser",
```



```

        "Operations": [
            "Decrypt"
        ]
    }
}

```

使用授予字符

AWS KMS API 遵循[最終一致性](#)模型。當建立授予時，授予可能無法立即生效。在該變更適用於整個 AWS KMS 之前，可能會有短暫延遲。變更傳播到整個系統通常需要不到幾秒鐘的時間，但在某些情況下可能需要幾分鐘。一旦授權完全人傳播至整個系統，承授者主體即可使用授權的許可，而不需指定授權字符或授權的任何證據。不過，如果授予是如此新，但尚未讓所有 知道 AWS KMS，則請求可能會失敗並顯示 `AccessDeniedException` 錯誤。

若要立即使用新授予中的許可，請使用授予的[授予字符](#)。儲存 [CreateGrant](#) 操作傳回的授予字符。然後在 AWS KMS 操作的請求中提交授予字符。您可以提交授予字符到任何 AWS KMS [授予操作](#)，也可以在相同的請求中提交多個授予字符。

以下範例使用 `CreateGrant` 操作來建立允許 [GenerateDataKey](#) 和 [Decrypt](#) 操作的授予。它會儲存 `CreateGrant` 在 `token` 變數中傳回的授予字符。然後，在呼叫 `GenerateDataKey` 操作時，它會使用 `token` 變數中的授予字符。

```

# Create a grant; save the grant token
$ token=$(aws kms create-grant \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::111122223333:user/appUser \
    --retiring-principal arn:aws:iam::111122223333:user/acctAdmin \
    --operations GenerateDataKey Decrypt \
    --query GrantToken \
    --output text)

# Use the grant token in a request
$ aws kms generate-data-key \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --key-spec AES_256 \
    --grant-tokens $token

```

具有許可的委託人也可以使用授予權杖淘汰新的授予，即使整個授予都可用 AWS KMS。
(`RevokeGrant` 操作不接受授予字符。) 如需詳細資訊，請參閱[淘汰和撤銷授予](#)。

```
# Retire the grant
$ aws kms retire-grant --grant-token $token
```

淘汰和撤銷授予

若要刪除授予，請淘汰或撤銷授予。

[RetireGrant](#) 和 [RevokeGrant](#) 操作相互之間非常相似。這兩項操作都會刪除授予，這會消除授予允許的許可。這些操作之間的主要區別在於它們是如何取得授權的。

RevokeGrant

與大多數 AWS KMS 操作一樣，RevokeGrant 操作的存取是透過 [金鑰政策](#) 和 [IAM 政策](#) 來控制。[RevokeGrant](#) API 可以由任何具有 kms:RevokeGrant 許可的主體呼叫。此許可會納入提供給金鑰管理員的標準許可中。一般而言，管理員會撤銷授予，以拒絕授予允許的許可。

RetireGrant

授予決定誰可以將其淘汰。此設計可讓您控制授予的生命週期，而不需變更金鑰政策或 IAM 政策。一般而言，當您使用其許可完成時，就會淘汰授予。

可以透過授予中指定的選用 [淘汰主體](#) 淘汰授予。[承授者主體](#) 亦可他淘汰授予，但前提是其也是淘汰主體或包含 RetireGrant 操作的授予。作為備份，建立授予 AWS 帳戶 的 可以淘汰授予。

有可以在 IAM 政策中使用的 kms:RetireGrant 許可，但其公用程式有限。在授予中指定的主體可以淘汰授予，無需 kms:RetireGrant 許可。單獨的 kms:RetireGrant 許可不允許主體淘汰授予。kms:RetireGrant 許可可在 [金鑰政策](#) 或 [資源控制政策](#) 中無效。

- 若要拒絕淘汰授予的許可，您可以在 IAM 政策中使用具有 kms:RetireGrant 許可 Deny 的動作。
- AWS 帳戶 擁有 KMS 金鑰的 可以將 kms:RetireGrant 許可委派給帳戶中的 IAM 主體。
- 如果淘汰委託人不同 AWS 帳戶，其他帳戶中的管理員可以使用 kms:RetireGrant 將淘汰授予的許可委派給該帳戶中的 IAM 委託人。

AWS KMS API 遵循 [最終一致性](#) 模型。當您建立、淘汰或撤銷授權時，在該變更適用於整個 AWS KMS 之前，可能會有短暫延遲。變更傳播到整個系統通常需要不到幾秒鐘的時間，但在某些情況下可能需要幾分鐘。如果您需要立即刪除新的授予，在全部可用之前 AWS KMS，[請使用授予字符](#) 來淘汰授予。您無法使用授予字符來撤銷授予。

的條件索引鍵 AWS KMS

您可以在控制 AWS KMS 資源存取的[金鑰政策和 IAM 政策](#)中指定條件。政策陳述式只有在符合下列條件時才有效。例如，您可能希望在特定日期之後才套用政策陳述式。或者，您可能會希望政策陳述式只在 API 請求中出現特定的值時才控制存取。

若要指定條件，請在具有 [IAM 條件運算子](#)的政策陳述式 [Condition 元素](#)中使用條件鍵。有些條件金鑰通常適用於 AWS；其他則專屬於 AWS KMS。

條件索引鍵值必須遵循 AWS KMS 索引鍵政策和 IAM 政策的字元和編碼規則。如需有關金鑰政策文件規則的詳細資訊，請參閱[金鑰政策格式](#)。如需有關 IAM 政策文件規則的詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 名稱需求](#)。

主題

- [AWS 全域條件索引鍵](#)
- [AWS KMS 條件索引鍵](#)
- [AWS KMS 已驗證平台的條件索引鍵](#)

AWS 全域條件索引鍵

AWS 定義[全域條件索引鍵](#)，這是一組政策條件索引鍵，適用於使用 IAM 進行存取控制的所有 AWS 服務。AWS KMS 支援所有全域條件索引鍵。您可以在 AWS KMS 金鑰政策和 IAM 政策中使用它們。

例如，您可以使用 [aws:PrincipalArn](#) 條件索引鍵，僅在請求中之主體的表示方式是條件索引鍵值中的 Amazon Resource Name (ARN) 時，允許存取 AWS KMS key (KMS 金鑰)。若要在中支援[屬性型存取控制](#) (ABAC) AWS KMS，您可以在 IAM 政策中使用 [aws : ResourceTag/tag-key](#) 全域條件金鑰，以允許存取具有特定標籤的 KMS 金鑰。

為了協助防止服務在委託人為 AWS 服務[AWS 委託人](#)的政策中用作混淆代理人，您可以使用 [aws:SourceArn](#)或 [aws:SourceAccount](#)全域條件金鑰。如需詳細資訊，請參閱[使用 aws:SourceArn 或 aws:SourceAccount 條件金鑰](#)。

如需 AWS 全域條件金鑰的相關資訊，包括可用的請求類型，請參閱《IAM 使用者指南》中的[AWS 全域條件內容金鑰](#)。如需 IAM 政策中使用全域條件索引鍵的範例，請參閱《IAM 使用者指南》中的[控制對請求的存取](#)和[控制標籤索引鍵](#)。

下列主題提供的特殊指導，讓您能根據 IP 位址和 VPC 端點來使用條件金鑰。

主題

- [在具有 AWS KMS 許可的政策中使用 IP 地址條件](#)
- [在具有 AWS KMS 許可的政策中使用 VPC 端點條件](#)
- [在 IAM 和 AWS KMS 金鑰政策中使用 IPv6 地址](#)

在具有 AWS KMS 許可的政策中使用 IP 地址條件

您可以使用 AWS KMS 來保護[整合 AWS 服務](#)中的資料。但是，在允許或拒絕存取的相同政策陳述式中指定 [IP 地址條件運算子](#)或`aws:SourceIp`條件索引鍵時，請小心 AWS KMS。例如，中的政策[AWS：AWS 根據來源 IP 拒絕存取](#)將 AWS 動作限制為來自指定 IP 範圍的請求。

考慮以下情形：

1. 您可以連接類似 的政策[AWS：AWS 根據來源 IP 拒絕存取](#)到 IAM 身分。您將 `aws:SourceIp` 條件金鑰的值設定為使用者公司的 IP 地址範圍。這個 IAM 身分還有其他連接的政策，允許他使用 Amazon EBS、Amazon EC2 和 AWS KMS。
2. 該身分嘗試將加密的 EBS 磁碟區連接到 EC2 執行個體。即使使用者有權使用所有相關的服務，這個動作仍會失敗，發生授權錯誤。

步驟 2 失敗，因為 AWS KMS 對 的解密磁碟區的加密資料金鑰請求來自與 Amazon EC2 基礎設施相關聯的 IP 地址。若要成功，請求必須來自原始使用者的 IP 地址。由於步驟 1 中的政策明確拒絕來自指定 IP 地址以外的所有請求，因此 Amazon EC2 被拒絕解密 EBS 磁碟區之加密資料金鑰的許可。

此外，當請求來自 [Amazon VPC 端點](#)時，`aws:SourceIP` 條件索引鍵無效。若要將請求限制為 VPC 端點 (包含 [AWS KMS VPC 端點](#))，請使用 `aws:SourceVpce` 或 `aws:SourceVpc` 條件金鑰。如需詳細資訊，請參閱 [Amazon VPC 使用者指南](#)中的 VPC 端點 - 控制端點的使用。

在具有 AWS KMS 許可的政策中使用 VPC 端點條件

[AWS KMS 支援採用 PrivateLink 技術的 Amazon Virtual Private Cloud \(Amazon VPC\) 端點](#)。 [AWS PrivateLink](#) 您可以在金鑰政策和 IAM 政策中使用下列[全域條件金鑰](#)，在請求來自 VPC 或使用 VPC 端點時控制對 AWS KMS 資源的存取。如需詳細資訊，請參閱[使用 VPC 端點控制對 AWS KMS 資源的存取](#)。

- `aws:SourceVpc` 會限制對指定 VPC 所發出之請求的存取權。
- `aws:SourceVpce` 會限制對指定 VPC 端點所發出之請求的存取權。

如果您使用這些條件金鑰來控制對 KMS 金鑰的存取，您可能會不小心拒絕 AWS KMS 代表您使用 AWS 的服務存取。

請注意避免 [IP 地址條件金鑰](#) 範例這樣的狀況。如果您將 KMS 金鑰的請求限制為 VPC 或 VPC 端點，則 AWS KMS 來自 Amazon S3 或 Amazon EBS 等整合服務的呼叫可能會失敗。即使來源請求最終源自 VPC 或來自 VPC 端點，也會發生這種情況。

在 IAM 和 AWS KMS 金鑰政策中使用 IPv6 地址

嘗試 AWS KMS 透過 IPv6 存取之前，請確保更新任何包含 IP 地址限制的金鑰和 IAM 政策，以包含 IPv6 地址範圍。未更新以處理 IPv6 地址的 IP 型政策可能會導致用戶端在開始使用 IPv6 時錯誤遺失或取得存取權。如需 KMS 存取控制的一般指引，請參閱 [KMS 金鑰存取和許可](#)。若要了解 KMS 和雙堆疊支援，請參閱 [雙堆疊端點支援](#)。

Important

這些陳述式不允許任何動作。將這些陳述式與允許特定動作的其他陳述式結合使用。

下列陳述式明確拒絕存取來自 IPv4 地址 192.0.2.* 範圍之請求的所有 KMS 許可。超出此範圍的任何 IP 地址都不會明確拒絕 KMS 許可。由於所有 IPv6 地址都在拒絕範圍之外，此陳述式不會明確拒絕任何 IPv6 地址的 KMS 許可。

```
{
  "Sid": "DenyKMSPermissions",
  "Effect": "Deny",
  "Action": [
    "kms:*"
  ],
  "Resource": "*",
  "Condition": {
    "NotIpAddress": {
      "aws:SourceIp": [
        "192.0.2.0/24"
      ]
    }
  }
}
```

您可以修改 Condition 元素以拒絕 IPv4 (192.0.2.0/24) 和 IPv6 (2001:db8:1234::/32) 地址範圍，如下列範例所示。

```
{
  "Sid": "DenyKMSPermissions",
  "Effect": "Deny",
  "Action": [
    "kms:*"
  ],
  "Resource": "*",
  "Condition": {
    "NotIpAddress": {
      "aws:SourceIp": [
        "192.0.2.0/24",
        "2001:db8:1234::/32"
      ]
    }
  }
}
```

AWS KMS 條件索引鍵

AWS KMS 提供一組條件金鑰，您可以在金鑰政策和 IAM 政策中使用。這些條件索引鍵是特定的 AWS KMS。例如，在控制對稱加密 KMS 金鑰的存取權時，您可以使用 `kms:EncryptionContext:context-key` 條件金鑰來要求特定的[加密內容](#)。

API 操作請求的條件

許多 AWS KMS 條件金鑰會根據 AWS KMS 操作請求中的參數值來控制對 KMS 金鑰的存取。例如，您可使用 IAM 政策中的 [kms:KeySpec](#) 條件索引鍵，在僅當 `CreateKey` 請求中的 `KeySpec` 參數值為 `RSA_4096` 時，才允許使用 [CreateKey](#) 操作。

即使請求中未出現參數 (例如使用參數的預設值時)，這種類型的條件仍會發生作用。例如，您可使用 [kms:KeySpec](#) 條件索引鍵，在僅當 `KeySpec` 參數值為 `SYMMETRIC_DEFAULT` 時 (預設值)，才允許使用者使用 `CreateKey` 操作。此條件允許 `KeySpec` 參數值為 `SYMMETRIC_DEFAULT` 的請求，以及沒有 `KeySpec` 參數的請求。

在 API 操作中使用 KMS 金鑰需滿足的條件

有些 AWS KMS 條件金鑰可以根據操作中使用的 KMS 金鑰屬性來控制對操作的存取。例如，您可使用 [kms:KeyOrigin](#) 條件，在僅當 KMS 金鑰的 `Origin` 為 `AWS_KMS` 時，才允許主體呼叫 KMS 金鑰的 [GenerateDataKey](#)。若要確定可否以這種方式使用條件金鑰，請參閱條件金鑰的描述。

操作必須是 KMS 金鑰資源操作，也就是授權特定 KMS 金鑰的操作。若要識別 KMS 金鑰資源操作，請在[動作與資源表](#)中尋找操作之 `Resources` 資料欄的 KMS `key` 值。如果對特定 KMS 金鑰資源使用

此類型的條件索引鍵搭配未授權的操作，例如 [ListKeys](#)，因為條件永遠無法滿足，所以許可無效。授權 `ListKeys` 操作中沒有相關的 KMS 金鑰資源，也沒有 `KeySpec` 屬性。

下列主題說明每個 AWS KMS 條件索引鍵，並包含示範政策語法的政策陳述式範例。

使用帶有條件索引鍵的集合運算子

當政策條件比較兩組值時，例如請求中的一組標籤和政策中的一組標籤時，您需要知道 AWS 如何比較組。IAM 為此定義了兩個集合的運算子，`ForAnyValue` 和 `ForAllValues`。使用僅具有所需多重值條件索引鍵的集合運算子。請勿將集合運算子與單值條件索引鍵搭配使用。如往常一樣，在生產環境中使用之前，完整測試您的政策陳述式。

條件索引鍵是單值或多重值。若要判斷 AWS KMS 條件索引鍵是單值還是多值，請參閱條件索引鍵描述中的值類型欄。

- 單值條件索引鍵在授權內容 (請求或資源) 中最多有一個值。例如，因為每個 API 呼叫只能源自於一個 AWS 帳戶，所以 [kms:CallerAccount](#) 是單值條件索引鍵。請勿使用具有單值條件索引鍵的集合運算子。
- 單值條件索引鍵在授權內容 (請求或資源) 中有多個值。例如，因為每個 KMS 金鑰都可以有多個別名，所以 [kms:ResourceAliases](#) 可以擁有多個值。多重值條件索引鍵需要集合運算子。

請注意，單值和多重值條件索引鍵之間的差異取決於授權內容中值的數目；而非政策條件中值的數目。

Warning

將集合運算子搭配單值條件索引鍵使用，可以建立過度寬鬆 (或過度限制) 的政策陳述式。只能將集合運算子與多值條件索引鍵搭配使用。

如果您建立或更新的政策包含具有 `kms:EncryptionContext : context-key` 或 `aws:RequestTag/tag-key` 條件索引鍵的 `ForAllValues` 集合運算子，會 AWS KMS 傳回下列錯誤訊息：

```
OverlyPermissiveCondition: Using the ForAllValues set operator with a single-valued condition key matches requests without the specified [encryption context or tag] or with an unspecified [encryption context or tag]. To fix, remove ForAllValues.
```


如需 ForAnyValue 和 ForAllValues 集合運算子的詳細資訊，請參閱《IAM 使用者指南》中的[使用多個金鑰和值](#)。如需使用具有單一值條件的 ForAllValues 集合運算子的風險資訊，請參閱《IAM 使用者指南》中的[安全性警告 – 具有單一值索引鍵的 ForAllValues](#)。

主題

- [kms:BypassPolicyLockoutSafetyCheck](#)
- [kms:CallerAccount](#)
- [kms:CustomerMasterKeySpec](#) (已被取代)
- [kms: CustomerMasterKeyUsage](#) (已被取代)
- [kms:DataKeyPairSpec](#)
- [kms:EncryptionAlgorithm](#)
- [kms:EncryptionContext:context-key](#)
- [kms:EncryptionContextKeys](#)
- [kms:ExpirationModel](#)
- [kms:GrantConstraintType](#)
- [kms:GrantIsForAWSResource](#)
- [kms:GrantOperations](#)
- [kms:GranteePrincipal](#)
- [kms:KeyAgreementAlgorithm](#)
- [kms:KeyOrigin](#)
- [kms:KeySpec](#)
- [kms:KeyUsage](#)
- [kms:MacAlgorithm](#)
- [kms:MessageType](#)
- [kms:MultiRegion](#)
- [kms:MultiRegionKeyType](#)
- [kms:PrimaryRegion](#)
- [kms:ReEncryptOnSameKey](#)
- [kms:RequestAlias](#)
- [kms:ResourceAliases](#)

- [kms:ReplicaRegion](#)
- [kms:RetiringPrincipal](#)
- [kms:RotationPeriodInDays](#)
- [kms:ScheduleKeyDeletionPendingWindowInDays](#)
- [kms:SigningAlgorithm](#)
- [kms:ValidTo](#)
- [kms:ViaService](#)
- [kms:WrappingAlgorithm](#)
- [kms:WrappingKeySpec](#)

kms:BypassPolicyLockoutSafetyCheck

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:BypassPolicyLockoutSafetyCheck	Boolean	單一值	CreateKey PutKeyPolicy	僅限 IAM 政策 金鑰政策和 IAM 政策

根據請求中的 BypassPolicyLockoutSafetyCheck 參數值，kms:BypassPolicyLockoutSafetyCheck 條件索引鍵會控制對 [CreateKey](#) 和 [PutKeyPolicy](#) 操作的存取。

以下範例 IAM 政策陳述式在 CreateKey 請求中的 BypassPolicyLockoutSafetyCheck 參數值是 true。時，拒絕使用者建立 KMS 金鑰的許可，來防止使用者繞過政策鎖定安全檢查。

```
{
  "Effect": "Deny",
  "Action": [
    "kms:CreateKey",
    "kms:PutKeyPolicy"
  ],
  "Resource": "*",
  "Condition": {
```

```

    "Bool": {
      "kms:BypassPolicyLockoutSafetyCheck": true
    }
  }
}

```

您也可以在 IAM 政策或金鑰政策中使用 `kms:BypassPolicyLockoutSafetyCheck` 條件索引鍵，以控制對 `PutKeyPolicy` 操作的存取。來自金鑰政策的以下範例政策陳述式防止使用者在變更 KMS 金鑰的政策時繞過政策鎖定安全檢查。

除了使用明確 `Deny`，此政策陳述式使用 `Allow` 搭配 [Null 條件運算子](#)，只在請求不包含 `BypassPolicyLockoutSafetyCheck` 參數時才允許存取。不使用參數時，預設值為 `false`。在極少數需要繞過的情況下，您可以覆寫這個比較弱的政策陳述式。

```

{
  "Effect": "Allow",
  "Action": "kms:PutKeyPolicy",
  "Resource": "*",
  "Condition": {
    "Null": {
      "kms:BypassPolicyLockoutSafetyCheck": true
    }
  }
}

```

另請參閱

- [kms:KeySpec](#)
- [kms:KeyOrigin](#)
- [kms:KeyUsage](#)

kms:CallerAccount

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
<code>kms:CallerAccount</code>	String	單一值	KMS 金鑰資源操作	金鑰政策和 IAM 政策

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
			自訂金鑰存放操作	

您可以使用此條件索引鍵，來允許或拒絕 AWS 帳戶中所有身分 (使用者和角色) 的存取。在金鑰政策中，您可以使用 Principal 元素來指定政策陳述式套用的身分。Principal 元素的語法不提供用來指定 AWS 帳戶中所有身分的方式。但是，您可以將此條件索引鍵與指定所有 AWS 身分的 Principal 元素結合，以達到此效果。

您可以使用它來控制對任何 KMS 金鑰資源操作的存取，也就是使用特定 KMS 金鑰的任何 AWS KMS 操作。若要識別 KMS 金鑰資源操作，請在[動作與資源表](#)中尋找操作之 Resources 資料欄的 KMS key 值。其也適用於管理[自訂金鑰存放](#)的操作。

例如，以下金鑰政策陳述式示範了如何使用 kms:CallerAccount 條件索引鍵。此政策陳述式位於 AWS 受管金鑰 適用於 Amazon EBS 的金鑰政策中。它結合了指定所有 AWS 身分的 Principal 元素與 kms:CallerAccount 條件索引鍵，以有效地允許存取 AWS 帳戶 111122223333 中的所有身分。它包含額外的 AWS KMS 條件金鑰 (kms:ViaService)，只允許透過 Amazon EBS 發出的請求，以進一步限制許可。如需詳細資訊，請參閱[kms:ViaService](#)。

```
{
  "Sid": "Allow access through EBS for all principals in the account that are
authorized to use EBS",
  "Effect": "Allow",
  "Principal": {"AWS": "*"},
  "Condition": {
    "StringEquals": {
      "kms:CallerAccount": "111122223333",
      "kms:ViaService": "ec2.us-west-2.amazonaws.com"
    }
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:CreateGrant",
    "kms:DescribeKey"
  ],
}
```

```
"Resource": "*"
}
```

kms:CustomerMasterKeySpec (已被取代)

kms:CustomerMasterKeySpec 條件索引鍵已被取代。請改用 [kms:KeySpec](#) 條件索引鍵。

kms:CustomerMasterKeySpec 和 kms:KeySpec 條件索引鍵的運作方式相同。只有名稱不同。建議您使用 kms:KeySpec。不過，為了避免中斷變更，AWS KMS 支援兩個條件索引鍵。

kms:CustomerMasterKeyUsage (已被取代)

kms:CustomerMasterKeyUsage 條件索引鍵已被取代。請改用 [kms:KeyUsage](#) 條件索引鍵。

kms:CustomerMasterKeyUsage 和 kms:KeyUsage 條件索引鍵的運作方式相同。只有名稱不同。建議您使用 kms:KeyUsage。不過，為了避免中斷變更，AWS KMS 支援兩個條件索引鍵。

kms:DataKeyPairSpec

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:DataKeyPairSpec	String	單一值	GeneratedDataKeyPair GeneratedDataKeyPairWithoutPlaintext	金鑰政策和 IAM 政策

根據請求中的 KeyPairSpec 參數值，您可以使用此條件索引鍵控制對 [GenerateDataKeyPair](#) 和 [GenerateDataKeyPairWithoutPlaintext](#) 操作的存取。例如，您可以允許使用者產生僅限特定類型的資料金鑰對。

下列範例金鑰政策陳述式使用 kms:DataKeyPairSpec 條件索引鍵，允許使用者使用 KMS 金鑰只產生 RSA 資料金鑰對。

```
{
```

```
"Effect": "Allow",
"Principal": {
  "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
},
"Action": [
  "kms:GenerateDataKeyPair",
  "kms:GenerateDataKeyPairWithoutPlaintext"
],
"Resource": "*",
"Condition": {
  "StringLike": {
    "kms:DataKeyPairSpec": "RSA*"
  }
}
```

另請參閱

- [kms:KeySpec](#)
- [the section called “kms:EncryptionAlgorithm”](#)
- [the section called “kms:EncryptionContext:context-key”](#)
- [the section called “kms:EncryptionContextKeys”](#)

kms:EncryptionAlgorithm

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:EncryptionAlgorithm	String	單一值	Decrypt	金鑰政策和 IAM 政策
			Encrypt	
			GenerateDataKey	
			GenerateDataKeyPair	
			GenerateDataKeyPair	

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
			WithoutPlaintext	
			GeneratedDataKeyWithoutPlaintext	
			ReEncrypt	

您可以使用 `kms:EncryptionAlgorithm` 條件金鑰，根據操作中使用的加密演算法控制對密碼編譯操作的存取。對於 [Encrypt](#)、[Decrypt](#) 和 [ReEncrypt](#) 操作，則根據請求中的 [EncryptionAlgorithm](#) 參數值控制存取。對於產生資料金鑰和資料金鑰對的操作，根據加密資料金鑰所用之加密演算法控制存取。

此條件金鑰不會影響在外部執行的操作 AWS KMS，例如在外部非對稱 KMS 金鑰對中使用公有金鑰加密 AWS KMS。

請求中的 EncryptionAlgorithm 參數

若要允許使用者只使用特定加密演算法搭配 KMS 金鑰，請使用具有 Deny 效果的政策陳述式和 `StringNotEquals` 條件運算子。例如，下列範例金鑰政策陳述式禁止擔任 `ExampleRole` 角色的主體在指定的密碼編譯操作中使用此 KMS 金鑰，除非請求中的加密演算法是 `RSAES_OAEP_SHA_256` (即與 RSA KMS 金鑰搭配使用的非對稱加密演算法)。

與允許使用者使用特定加密演算法的政策陳述式不同，具有像這樣的雙負數的政策陳述式可防止此 KMS 金鑰的其他政策和授權讓此角色使用其他加密演算法。此金鑰政策陳述式中的 Deny 優先於任何金鑰政策或具有 Allow 效果的 IAM 政策，也優先於此 KMS 金鑰及其主體的所有授權。

```
{
  "Sid": "Allow only one encryption algorithm with this asymmetric KMS key",
  "Effect": "Deny",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
```



```

    "kms:ReEncrypt*"
  ],
  "Resource": "*",
  "Condition": {
    "StringNotEquals": {
      "kms:EncryptionAlgorithm": "RSAES_OAEP_SHA_256"
    }
  }
}

```

用於操作的加密演算法

您也可以使用 `kms:EncryptionAlgorithm` 條件索引鍵，即使在請求中未指定演算法，也可根據操作中使用的加密演算法來控制對操作的存取。這可讓您需要或禁止 `SYMMETRIC_DEFAULT` 演算法，這可能不會在請求中指定，因為其是預設值。

此功能能讓您使用 `kms:EncryptionAlgorithm` 條件索引鍵控制對產生資料金鑰和資料金鑰對之操作的存取。這些操作只使用對稱加密 KMS 金鑰和 `SYMMETRIC_DEFAULT` 演算法。

例如，此 IAM 政策會將其主體限制在對稱加密。除非請求中指定或操作中使用的加密演算法是 `SYMMETRIC_DEFAULT`，否則拒絕存取範例帳戶中密碼編譯操作的任何 KMS 金鑰。包含 `GenerateDataKey*` 新增 [GenerateDataKey](#)、[GenerateDataKeyWithoutPlaintext](#)、[GenerateDataKeyPair](#) 和 [GenerateDataKeyPairWithoutPlaintext](#) 至許可。條件對這些操作沒有影響，因為其一律使用對稱式加密演算法。

```

{
  "Sid": "AllowOnlySymmetricAlgorithm",
  "Effect": "Deny",
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:key/*",
  "Condition": {
    "StringNotEquals": {
      "kms:EncryptionAlgorithm": "SYMMETRIC_DEFAULT"
    }
  }
}

```

另請參閱

- [the section called “kms:MacAlgorithm”](#)
- [kms:SigningAlgorithm](#)

kms:EncryptionContext:context-key

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:EncryptionContext:context-key	String	單一值	CreateGrant Encrypt Decrypt GenerateDataKey GenerateDataKeyPair GenerateDataKeyPairWithoutPlaintext GenerateDataKeyWithoutPlaintext ReEncrypt RetireGrant	金鑰政策和 IAM 政策

您可以使用 kms:EncryptionContext:context-key 條件金鑰，依據[密碼編譯操作](#)請求中的[加密內容](#)，控制對[對稱加密 KMS 金鑰](#)的存取。使用此條件索引鍵可同時評估加密內容對中的金鑰和值。若

只評估加密內容索引鍵，或需要加密內容，而不考慮金鑰或值，請使用 [kms:EncryptionContextKeys](#) 條件索引鍵。

 Note

條件金鑰值必須符合金鑰政策和 IAM 政策的字元規則。某些在加密內容中有效的字元在政策中無效。您可能無法使用此條件金鑰來表示所有的有效加密內容值。如需有關金鑰政策文件規則的詳細資訊，請參閱[金鑰政策格式](#)。如需有關 IAM 政策文件規則的詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 名稱需求](#)。

您不能使用[非對稱 KMS 金鑰](#)或 [HMAC KMS 金鑰](#)在密碼編譯操作中指定加密內容。非對稱演算法和 MAC 演算法不支援加密內容。

若要使用 `kms:EncryptionContext:context-key` 條件索引鍵，請使用加密內容索引鍵取代 *context-key* 預留位置。將 *context-value* 預留位置取代為加密內容值。

```
"kms:EncryptionContext:context-key": "context-value"
```

例如，以下條件索引鍵指定加密內容，其中金鑰是 `AppName`，值是 `ExampleApp` (`AppName = ExampleApp`)。

```
"kms:EncryptionContext:AppName": "ExampleApp"
```

這是[單一值條件索引鍵](#)。條件索引鍵中的金鑰會指定特定的加密內容索引鍵 (`context-key`)。雖然您可以在每個 API 請求中包含多個加密內容對，但是與指定 `context-key` 對的加密內容只能有一個值。例如，`kms:EncryptionContext:Department` 條件索引鍵僅適用於與 `Department` 金鑰對的加密內容，以及任何與 `Department` 金鑰對的指定加密內容只能有一個值。

請勿將集合運算子與 `kms:EncryptionContext:context-key` 條件索引鍵搭配使用。如果您建立的政策陳述式包含 `Allow` 動作、`kms:EncryptionContext:context-key` 條件索引鍵和 `ForAllValues` 集合運算子，則條件會允許沒有加密內容的請求，以及未在政策條件中指定的加密內容對的請求。

 Warning

請勿將 `ForAnyValue` 或 `ForAllValues` 集合運算子與此單一值條件索引鍵搭配使用。這些集合運算子可以建立不需要您想要要求之值的政策條件，並允許您想要禁止的值。

如果您建立或更新的政策包含具有 `kms:EncryptionContext : context-key` 的 `ForAllValues` 集合運算子，會 AWS KMS 傳回下列錯誤訊息：

`OverlyPermissiveCondition:EncryptionContext: Using the ForAllValues set operator with a single-valued condition key matches requests without the specified encryption context or with an unspecified encryption context. To fix, remove ForAllValues.`

若要要求特定的加密內容對，請將 `kms:EncryptionContext:context-key` 條件索引鍵與 `StringEquals` 運算子搭配使用。

以下範例金鑰政策陳述式，僅當請求中的加密內容包含 `AppName:ExampleApp` 對時，才允許可以擔任角色的主體使用 `GenerateDataKey` 請求中的 KMS 金鑰。允許其他加密內容對。

金鑰名稱不會區分大小寫。值是否區分大小寫，取決於條件運算子，例如 `StringEquals`。如需詳細資訊，請參閱[加密內容條件需區分大小寫](#)。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:AppName": "ExampleApp"
    }
  }
}
```

若要要求加密內容對並禁止所有其他加密內容對，請同時使用政策陳述式中的 `kms:EncryptionContext:context-key` 和 [kms:EncryptionContextKeys](#)。以下範例政策陳述式使用 `kms:EncryptionContext:AppName` 條件索引鍵，要求請求中有 `AppName=ExampleApp` 加密內容對。它還會將 `kms:EncryptionContextKeys` 條件索引鍵與 `ForAllValues` 集合運算子搭配使用，以僅允許 `AppName` 加密內容索引鍵。

`ForAllValues` 集合運算子將請求中的加密內容索引鍵限制為 `AppName`。如果具有 `ForAllValues` 集合運算子的 `kms:EncryptionContextKeys` 條件在政策陳述式中單獨使用，則此集合運算子會允許沒有加密內容的請求。不過，如果請求沒有加密內容，則 `kms:EncryptionContext:AppName` 條

件將會失敗。如需 `ForAllValues` 集合運算子的詳細資訊，請參閱《IAM 使用者指南》中的[使用多個索引鍵和值](#)。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/KeyUsers"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:AppName": "ExampleApp"
    },
    "ForAllValues:StringEquals": {
      "kms:EncryptionContextKeys": [
        "AppName"
      ]
    }
  }
}
```

您也可以使用此條件索引鍵來拒絕存取特定操作的 KMS 金鑰。以下範例金鑰政策陳述式使用 `Deny` 效果，在請求中加密內容包含 `Stage=Restricted` 加密內容對的情形下，禁止主體使用 KMS 金鑰。此條件允許使用其他加密內容對的請求，包括具有 `Stage` 金鑰和其他值的加密內容對，例如 `Stage=Test`。

```
{
  "Effect": "Deny",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:Stage": "Restricted"
    }
  }
}
```

使用多個加密內容對

您可以要求或禁止多個加密內容對。您也可以要求其中一個加密內容對。如需用來解譯這些條件之邏輯的詳細資訊，請參閱《IAM 使用者指南》中的[建立具有多個索引鍵或值的條件](#)。

Note

本主題的舊版會顯示使用具有 `kms:EncryptionContext:context-key` 條件索引鍵之 `ForAnyValue` 和 `ForAllValues` 集合運算子的政策陳述式。使用具有[單一值條件索引鍵](#)的集合運算子可能會導致政策允許沒有加密內容和未指定加密內容對的請求。

例如，政策條件具有 `Allow` 效果、`ForAllValues` 集合運算子和

`"kms:EncryptionContext:Department": "IT"` 條件索引鍵不會將加密內容限制為 `"Department=IT"` 對。它允許沒有加密內容的請求和未指定加密內容對的請求，例如 `Stage=Restricted`。

請檢閱您的政策，並從任何具有 `kms:EncryptionContext:context-key` 的條件中排除集合運算子。嘗試使用此格式建立或更新政策會失敗，`OverlyPermissiveCondition` 為例外狀況。若要解決錯誤，請刪除集合運算子。

若要要求多個加密內容對，請以相同的條件列出對。以下範例金鑰政策陳述式需要兩個加密內容對，`Department=IT` 和 `Project=Alpha`。因為條件有不同的索引鍵 (`kms:EncryptionContext:Department` 和 `kms:EncryptionContext:Project`)，所以它們由 `AND` 運算子隱式連線。允許其他加密內容對，但不是必需的。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:Decrypt",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:Department": "IT",
      "kms:EncryptionContext:Project": "Alpha"
    }
  }
}
```

若需要一個加密內容對 OR 另一個對，請將每個條件索引鍵放在個別的政策陳述式中。以下範例金鑰政策需要 Department=IT 或 Project=Alpha 對，或兩者。允許其他加密內容對，但不是必需的。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:Department": "IT"
    }
  }
},
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:Project": "Alpha"
    }
  }
}
}
```

若需要特定的加密對並排除所有其他加密內容對，請同時使用政策陳述式中的 kms:EncryptionContext:context-key 和 [kms:EncryptionContextKeys](#)。以下金鑰政策陳述式使用 kms:EncryptionContext:context-key 條件來要求同時使用 Department=IT 和 Project=Alpha 對的加密內容。它會將 kms:EncryptionContextKeys 條件索引鍵與 ForAllValues 集合運算子搭配使用，以僅允許 Department 和 Project 加密內容索引鍵。

ForAllValues 集合運算子將請求中的加密內容索引鍵限制為 Department 和 Project。如果它是單獨使用的條件，則此集合運算子將允許沒有加密內容的請求，但在此組態中，kms:EncryptionContext:context-key 在這種情況下將失敗。

```
{
  "Effect": "Allow",
```



```

"Principal": {
  "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
},
"Action": "kms:GenerateDataKey",
"Resource": "*",
"Condition": {
  "StringEquals": {
    "kms:EncryptionContext:Department": "IT",
    "kms:EncryptionContext:Project": "Alpha"
  },
  "ForAllValues:StringEquals": {
    "kms:EncryptionContextKeys": [
      "Department",
      "Project"
    ]
  }
}
}

```

您也可以禁止多個加密內容對。以下範例金鑰政策陳述式使用 Deny 效果，在請求中的加密內容包含 Stage=Restricted 和 Stage=Production 對的情形下，禁止主體使用 KMS 金鑰。

相同索引鍵 (kms:EncryptionContext:Stage) 的多個值 (Restricted 和 Production) 由 OR 隱式連線。如需詳細資訊，請參閱《IAM 使用者指南》中的[具有多個索引鍵或值之條件的評估邏輯](#)。

```

{
  "Effect": "Deny",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:Stage": [
        "Restricted",
        "Production"
      ]
    }
  }
}

```

加密內容條件需區分大小寫

解密操作中指定的加密內容和加密操作中指定的加密內容必須完全一樣，且大小寫相符。只有具有多對之加密內容中的配對順序可以改變。

不過，在政策條件中，條件金鑰不區分大小寫。條件值是否區分大小寫，取決於您使用的[政策條件運算子](#)，例如 `StringEquals` 或 `StringEqualsIgnoreCase`。

因此，由 `kms:EncryptionContext:` 字首和 *context-key* 替換組成的條件金鑰不區分大小寫。使用此條件的政策不會檢查條件金鑰任一元素的大小寫。值 (亦即 *context-value* 替換) 是否區分大小寫，取決於政策條件運算子。

例如，以下政策陳述式允許操作的加密內容包含 `Appname` 金鑰，無論其大小寫。`StringEquals` 條件要求 `ExampleApp` 符合指定的大寫形式。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:Decrypt",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:Appname": "ExampleApp"
    }
  }
}
```

若需要區分大小寫的加密內容金鑰，請使用 [kms:EncryptionContextKeys](#) 政策條件並搭配區分大小寫的條件運算子，例如 `StringEquals`。在這個政策條件中，因為加密內容索引鍵是此政策條件的值，所以其是否區分大小寫取決於條件運算子。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
```

```
"ForAnyValue:StringEquals": {  
  "kms:EncryptionContextKeys": "AppName"  
}  
}  
}
```

若需要同時評估加密內容索引鍵和值是否區分大小寫，請在相同的政策陳述式中一起使用 `kms:EncryptionContextKeys` 和 `kms:EncryptionContext:context-key` 政策條件。區分大小寫的條件運算子 (例如 `StringEquals`) 一律適用於條件的值。加密內容索引鍵 (例如 `AppName`) 是 `kms:EncryptionContextKeys` 條件的值。加密內容值 (例如 `ExampleApp`) 是 `kms:EncryptionContext:context-key` 條件的值。

例如，在以下範例政策陳述式中，因為 `StringEquals` 運算子區分大小寫，所以加密內容索引鍵和加密內容值會區分大小寫。

```
{  
  "Effect": "Allow",  
  "Principal": {  
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"  
  },  
  "Action": "kms:GenerateDataKey",  
  "Resource": "*",  
  "Condition": {  
    "ForAnyValue:StringEquals": {  
      "kms:EncryptionContextKeys": "AppName"  
    },  
    "StringEquals": {  
      "kms:EncryptionContext:AppName": "ExampleApp"  
    }  
  }  
}
```

在加密內容條件中使用變數

加密內容對中的金鑰和值必須是簡單的常值字串。它們不能是整數或物件，或任何未完全解析的類型。如果您使用不同的類型，例如整數或浮點數，會將其 AWS KMS 解譯為文字字串。

```
"encryptionContext": {  
  "department": "10103.0"  
}
```

不過，`kms:EncryptionContext:context-key` 條件索引鍵的值可以是 [IAM 政策變數](#)。這些政策變數會根據要求中的值在執行時間中解析。例如，`aws:CurrentTime` 解析為請求的時間，而 `aws:username` 解析為易記的發起人名稱。

您可以使用這些政策變數來建立政策陳述式，其條件加密內容中需要非常特定的資訊，例如發起人的使用者名稱。由於它包含變數，因此您可以對可擔任角色的所有使用者使用相同的政策陳述式。您不需要為每個使用者個別編寫政策陳述式。

請考慮以下情況，您想要所有可擔任角色的使用者使用相同的 KMS 金鑰來加密和解密其資料。不過，您只想要允許他們解密由他們加密的資料。首先，要求每個請求都 AWS KMS 包含加密內容，其中金鑰為 `user`，而值為發起人的 AWS 使用者名稱，例如下列使用者名稱。

```
"encryptionContext": {
  "user": "bob"
}
```

然後，若要強制執行此需求，您可以使用類似以下範例中的政策陳述式。此政策陳述式提供 `TestTeam` 角色使用 KMS 金鑰加密和解密資料的許可。不過，許可只有在請求中的加密內容包含 `"user": "<username>"` 對時才有效。為了代表使用者名稱，條件使用 [aws:username](#) 政策變數。

評估請求時，發起人的使用者名稱會取代條件中的變數。因此，條件需要 `"user": "bob"` 的「bob」和 `"user": "alice"` 的「alice」加密內容。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/TestTeam"
  },
  "Action": [
    "kms:Decrypt",
    "kms:Encrypt"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:user": "${aws:username}"
    }
  }
}
```

您只能在 `kms:EncryptionContext:context-key` 條件索引鍵的值中使用 IAM 政策變數。您不能在金鑰中使用變數。

您也可以使用變數中的[供應商特定的內容金鑰](#)。這些內容金鑰可唯一識別 AWS 使用 Web 聯合身分登入的使用者。

如同所有變數，這些變數只能用於 `kms:EncryptionContext:context-key` 政策條件，不能用於實際的加密內容。而且只能用於條件的值，不能用於金鑰。

例如，下列金鑰政策陳述式與前一個陳述式類似。不過，條件需要加密內容，其金鑰是 `sub` 而值是可唯一識別登入 Amazon Cognito 使用者集區的使用者。如需有關在 Amazon Cognito 中識別使用者和角色的詳細資訊，請參閱《[Amazon Cognito 開發人員指南](#)》中的 [IAM 角色](#)。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/TestTeam"
  },
  "Action": [
    "kms:Decrypt",
    "kms:Encrypt"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:sub": "${cognito-identity.amazonaws.com:sub}"
    }
  }
}
```

另請參閱

- [the section called “kms:EncryptionContextKeys”](#)
- [the section called “kms:GrantConstraintType”](#)

kms:EncryptionContextKeys

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:EncryptionContextKeys	字串 (清單)	多重值	CreateGrant Decrypt Encrypt GeneratedDataKey GeneratedDataKeyPair GeneratedDataKeyPairWithoutPlaintext GeneratedDataKeyWithoutPlaintext ReEncrypt RetireGrant	金鑰政策和 IAM 政策

您可以使用 `kms:EncryptionContextKeys` 條件金鑰，依據密碼編譯操作請求中的[加密內容](#)，控制對[對稱加密 KMS 金鑰](#)的存取。使用此條件索引鍵，只評估每個加密內容對中的金鑰。若要同時評估加密內容中的金鑰和值，請使用 `kms:EncryptionContext:context-key` 條件索引鍵。

您不能使用[非對稱 KMS 金鑰](#)或 [HMAC KMS 金鑰](#)在密碼編譯操作中指定加密內容。非對稱演算法和 MAC 演算法不支援加密內容。

Note

條件金鑰值，包括加密內容金鑰，必須符合 AWS KMS 金鑰政策的字元和編碼規則。您可能無法使用此條件金鑰來表示所有的有效加密內容金鑰。如需有關金鑰政策文件規則的詳細資訊，請參閱[金鑰政策格式](#)。如需有關 IAM 政策文件規則的詳細資訊，請參閱《IAM 使用者指南》中的[IAM 名稱需求](#)。

這是[多重值條件索引鍵](#)。您可以在每個 API 請求中指定多個加密內容對。kms:EncryptionContextKeys 會將請求中的加密內容索引鍵與政策中的加密內容索引鍵集合進行比較。若要判斷這些集合的比較方式，您必須提供政策條件中的 ForAnyValue 或 ForAllValues 集合運算子。如需集合運算子的詳細資訊，請參閱《IAM 使用者指南》中的[使用多個索引鍵和值](#)。

- ForAnyValue：請求中至少有一個加密內容索引鍵必須符合政策條件中的加密內容索引鍵。允許使用其他加密內容索引鍵。如果請求沒有加密內容，則不符合條件。
- ForAllValues：請求中每個加密內容索引鍵必須符合政策條件中的加密內容索引鍵。此集合運算子會將加密內容索引鍵限制為政策條件中的索引鍵。它不需要任何加密內容索引鍵，但它禁止未指定的加密內容索引鍵。

以下範例金鑰政策陳述式將 kms:EncryptionContextKeys 條件索引鍵與 ForAnyValue 集合運算子搭配使用。此政策陳述式僅在請求中的至少一個加密內容對包含 AppName 金鑰時 (無論其值為何)，才允許使用指定操作的 KMS 金鑰。

例如，此金鑰政策陳述式允許具有兩個加密內容對、AppName=Helper 和 Project=Alpha 的 GenerateDataKey 請求，因為第一個加密內容對符合條件。僅具有 Project=Alpha 或沒有加密內容的請求將失敗。

由於 [StringEquals](#) 條件操作會區分大小寫，因此此政策陳述式需要加密內容索引鍵的拼字和大小寫。但是，您也可以使用忽略金鑰大小寫的條件運算子，例如 StringEqualsIgnoreCase。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": [
    "kms:Encrypt",
```

```
    "kms:GenerateDataKey*"
  ],
  "Resource": "*",
  "Condition": {
    "ForAnyValue:StringEquals": {
      "kms:EncryptionContextKeys": "AppName"
    }
  }
}
```

您也可以使用 `kms:EncryptionContextKeys` 條件索引鍵來要求使用 KMS 金鑰之密碼編譯操作中的加密內容 (任何加密內容)。

以下範例金鑰政策陳述式使用 `kms:EncryptionContextKeys` 條件索引鍵搭配 [Null 條件運算子](#)，僅在 API 請求中存在加密內容 (不是 null) 時，才允許存取 KMS 金鑰。此情況不會檢查加密內容的索引鍵或值。它只會驗證加密內容是否存在。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": [
    "kms:Encrypt",
    "kms:GenerateDataKey*"
  ],
  "Resource": "*",
  "Condition": {
    "Null": {
      "kms:EncryptionContextKeys": false
    }
  }
}
```

另請參閱

- [kms:EncryptionContext:context-key](#)
- [kms:GrantConstraintType](#)

kms:ExpirationModel

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:ExpirationModel	String	單一值	ImportKeyMaterial	金鑰政策和 IAM 政策

kms:ExpirationModel 條件索引鍵會根據請求中的 [ExpirationModel](#) 參數值，控制對 [ImportKeyMaterial](#) 操作的存取。

ExpirationModel 是選擇性參數，用來決定匯入的金鑰材料是否過期。有效值為 KEY_MATERIAL_EXPIRES 和 KEY_MATERIAL_DOES_NOT_EXPIRE。KEY_MATERIAL_EXPIRES 為預設值。

過期日期和時間取決於 [ValidTo](#) 參數的值。除非 ExpirationModel 參數的值是 KEY_MATERIAL_DOES_NOT_EXPIRE，否則 ValidTo 參數為必要。您也可以使用 [kms:ValidTo](#) 條件金鑰來要求特定過期日期做為存取條件。

以下範例政策陳述式使用 kms:ExpirationModel 條件索引鍵，只在請求包含 ExpirationModel 參數且其值為 KEY_MATERIAL_DOES_NOT_EXPIRE 時，才允許使用者將金鑰材料匯入 KMS 金鑰。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:ImportKeyMaterial",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ExpirationModel": "KEY_MATERIAL_DOES_NOT_EXPIRE"
    }
  }
}
```

您也可以使用 kms:ExpirationModel 條件索引鍵，只在金鑰材料過期時才允許使用者匯入金鑰材料。以下範例金鑰政策陳述式使用 kms:ExpirationModel 條件索引鍵搭配 [Null 條件運算子](#)，僅

當請求沒有 `ExpirationModel` 參數時，才允許使用者匯入金鑰材料。`ExpirationModel` 的預設值為 `KEY_MATERIAL_EXPIRES`。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:ImportKeyMaterial",
  "Resource": "*",
  "Condition": {
    "Null": {
      "kms:ExpirationModel": true
    }
  }
}
```

另請參閱

- [kms:ValidTo](#)
- [kms:WrappingAlgorithm](#)
- [kms:WrappingKeySpec](#)

kms:GrantConstraintType

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:GrantConstraintType	String	單一值	CreateGrant RetireGrant	金鑰政策和 IAM 政策

您可以使用此條件索引鍵，根據請求中的[授予限制條件](#)來控制對 [CreateGrant](#) 操作的存取。

建立授與時，您可以選擇指定授與限制以允許操作，僅在特定[加密內容](#)存在時才授與許可。授與限制可以是兩種類型其中之一：`EncryptionContextEquals` 或 `EncryptionContextSubset`。您可以使用此條件金鑰來檢查請求是否包含其中一個類型。

⚠ Important

請勿在此欄位包含機密或敏感資訊。此欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

以下範例金鑰政策陳述式使用 `kms:GrantConstraintType` 條件索引鍵，只在請求包含 `EncryptionContextEquals` 授予限制條件時才允許使用者建立授予。此範例顯示金鑰政策中的政策陳述式。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
  },
  "Action": "kms:CreateGrant",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:GrantConstraintType": "EncryptionContextEquals"
    }
  }
}
```

另請參閱

- [kms:EncryptionContext:context-key](#)
- [kms:EncryptionContextKeys](#)
- [kms:GrantIsForAWSResource](#)
- [kms:GrantOperations](#)
- [kms:GranteePrincipal](#)
- [kms:RetiringPrincipal](#)

kms:GrantIsForAWSResource

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:GrantIsForAWSResource	Boolean	單一值	CreateGrant ListGrants RevokeGrant	金鑰政策和 IAM 政策

只有在[AWS 與 整合的服務 AWS KMS](#)代表使用者呼叫操作時，才允許或拒絕 [CreateGrant](#)、[ListGrants](#) 或 [RevokeGrant](#) 操作的許可。此政策條件不允許使用者直接呼叫這些授予操作。

以下範例金鑰政策陳述式會使用 kms:GrantIsForAWSResource 條件索引鍵。它允許與 整合 AWS 的服務 AWS KMS，例如 Amazon EBS，代表指定的委託人為此 KMS 金鑰建立授予。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:CreateGrant",
  "Resource": "*",
  "Condition": {
    "Bool": {
      "kms:GrantIsForAWSResource": true
    }
  }
}
```

另請參閱

- [kms:GrantConstraintType](#)
- [kms:GrantOperations](#)
- [kms:GranteePrincipal](#)
- [kms:RetiringPrincipal](#)

kms:GrantOperations

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:GrantOperations	String	多重值	CreateGrant	金鑰政策和 IAM 政策

您可以使用此條件索引鍵，根據請求中的[授予操作](#)來控制對 [CreateGrant](#) 操作的存取。例如，您可以允許使用者建立授予，以委派加密 (但不是解密) 的許可。如需授予的詳細資訊，請參閱[使用授予](#)。

這是[多重值條件索引鍵](#)。kms:GrantOperations 會將 CreateGrant 請求中的一組授予操作與政策中的一組授予操作進行比較。若要判斷這些集合的比較方式，您必須提供政策條件中的 ForAnyValue 或 ForAllValues 集合運算子。如需集合運算子的詳細資訊，請參閱《IAM 使用者指南》中的[使用多個索引鍵和值](#)。

- ForAnyValue：請求中至少有一個授予操作必須符合政策條件中的其中一個授予操作。允許其他授予操作。
- ForAllValues：請求中每個授予操作必須符合政策條件中的授予操作。此集合運算子會將授予操作限制為政策條件中指定的操作。它不需要任何授予操作，但它禁止未指定的授予操作。

當請求中沒有授予操作時，ForAllValues 也會傳回 true，但 CreateGrant 不允許。如果 Operations 參數遺失或具有空值時，則 CreateGrant 請求會失敗。

以下範例政策陳述式使用 kms:GrantOperations 條件索引鍵，僅當授予操作為 Encrypt、ReEncryptTo 或兩者時，才允許建立授予。如果授予包含任何其他操作，則 CreateGrant 請求會失敗。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:CreateGrant",
  "Resource": "*",
  "Condition": {
    "ForAllValues:StringEquals": {
      "kms:GrantOperations": [
```

```

        "Encrypt",
        "ReEncryptTo"
    ]
}
}
}

```

如果您將政策條件中的集合運算子變更為 `ForAnyValue`，則政策陳述式會要求授予中至少有一個授予操作為 `Encrypt` 或 `ReEncryptTo`，但它會允許其他授予操作，例如 `Decrypt` 或 `ReEncryptFrom`。

另請參閱

- [kms:GrantConstraintType](#)
- [kms:GrantIsForAWSResource](#)
- [kms:GranteePrincipal](#)
- [kms:RetiringPrincipal](#)

kms:GranteePrincipal

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
<code>kms:GranteePrincipal</code>	String	單一值	<code>CreateGrant</code>	IAM 和金鑰政策

您可以使用此條件索引鍵，根據請求中的 [GranteePrincipal](#) 參數值，控制對 [CreateGrant](#) 操作的存取。例如，您可以只在 `CreateGrant` 請求中的承授者主體符合條件陳述式中指定的主體時，才可建立授予來使用 KMS 金鑰。

若要指定承授者委託人，請使用委託人的 AWS Amazon Resource Name (ARN)。有效主體包括 AWS 帳戶 IAM 使用者、IAM 角色、聯合身分使用者和擔任的角色使用者。如需主體 ARN 語法的說明，請參閱《[IAM 使用者指南](#)》中的 [IAM ARNs](#)。

以下範例政策陳述式使用 `kms:GranteePrincipal` 條件索引鍵，僅當授予中的承授者主體是 `LimitedAdminRole` 時，才可建立 KMS 金鑰的授予。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:CreateGrant",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:GranteePrincipal": "arn:aws:iam::111122223333:role/LimitedAdminRole"
    }
  }
}
```

另請參閱

- [kms:GrantConstraintType](#)
- [kms:GrantIsForAWSResource](#)
- [kms:GrantOperations](#)
- [kms:RetiringPrincipal](#)

kms:KeyAgreementAlgorithm

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:KeyAgreementAlgorithm	String	單一值	DeriveSharedSecret	金鑰政策和 IAM 政策

您可以使用 kms:KeyAgreementAlgorithm 條件金鑰，根據請求中的 KeyAgreementAlgorithm 參數值來控制對 [DeriveSharedSecret](#) 操作的存取。的唯一有效值 KeyAgreementAlgorithm 是 ECDH。

例如，下列金鑰政策陳述式使用 kms:KeyAgreementAlgorithm 條件金鑰拒絕對 DeriveSharedSecret 的所有存取，除非 KeyAgreementAlgorithm 是 ECDH。

```
{
  "Effect": "Deny",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:DeriveSharedSecret",
  "Resource": "*",
  "Condition": {
    "StringNotEquals": {
      "kms:KeyAgreementAlgorithm": "ECDH"
    }
  }
}
```

另請參閱

- [the section called “kms:KeyUsage”](#)

kms:KeyOrigin

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:KeyOrigin	String	單一值	CreateKey	IAM 政策
			KMS 金鑰資源操作	金鑰政策和 IAM 政策

kms:KeyOrigin 條件索引鍵會根據此操作建立或使用之 KMS 金鑰的 Origin 屬性值，控制對操作的存取。以資源條件或請求條件運作。

您可以使用此條件索引鍵，根據請求中的 [Origin](#) 參數值，控制對 [CreateKey](#) 操作的存取。的有效值Origin為 AWS_KMS、EXTERNAL_KEY_STORE、AWS_CLOUDHSM和 EXTERNAL。

例如，只有在 AWS KMS (AWS_KMS) 中產生金鑰材料時、只有在與 [CloudHSM 自訂金鑰存放區](#) 相關聯的 AWS CloudHSM 叢集中產生金鑰材料時 (AWS_CLOUDHSM)、只有在[外部金鑰存放區中產生金鑰材料時](#) (EXTERNAL_KEY_STORE)，或只有在從外部來源[匯入金鑰材料時](#) ()，您才能建立 KMS 金鑰EXTERNAL。

下列範例金鑰政策陳述式使用 `kms:KeyOrigin` 條件金鑰，僅在 建立金鑰材料時 AWS KMS 建立 KMS 金鑰。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
      },
      "Action": "kms:CreateKey",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:KeyOrigin": "AWS_KMS"
        }
      }
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:GenerateDataKey",
        "kms:GenerateDataKeyWithoutPlaintext",
        "kms:GenerateDataKeyPair",
        "kms:GenerateDataKeyPairWithoutPlaintext",
        "kms:ReEncrypt*"
      ],
      "Resource": "arn:aws:kms:us-west-2:111122223333:key/*",
      "Condition": {
        "StringEquals": {
          "kms:KeyOrigin": "AWS_CLOUDHSM"
        }
      }
    }
  ]
}
```

```
}
```

您也可以使用 `kms:KeyOrigin` 條件索引鍵，根據用於操作之 KMS 金鑰的 `Origin` 屬性控制對使用或管理 KMS 金鑰之操作的存取。操作必須是 KMS 金鑰資源操作，也就是授權特定 KMS 金鑰的操作。若要識別 KMS 金鑰資源操作，請在[動作與資源表](#)中尋找操作之 Resources 資料欄的 KMS key 值。

例如，下列 IAM 政策允許主體執行指定的 KMS 金鑰資源操作，但僅能使用帳戶中於自訂金鑰存放區中建立的 KMS 金鑰。

```
{
  "Effect": "Allow",
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:GenerateDataKey",
    "kms:GenerateDataKeyWithoutPlaintext",
    "kms:GenerateDataKeyPair",
    "kms:GenerateDataKeyPairWithoutPlaintext",
    "kms:ReEncrypt*"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:key/*",
  "Condition": {
    "StringEquals": {
      "kms:KeyOrigin": "AWS_CLOUDHSM"
    }
  }
}
```

另請參閱

- [kms:BypassPolicyLockoutSafetyCheck](#)
- [kms:KeySpec](#)
- [kms:KeyUsage](#)

kms:KeySpec

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:KeySpec	String	單一值	CreateKey	IAM 政策
			KMS 金鑰資源操作	金鑰政策和 IAM 政策

kms:KeySpec 條件索引鍵會根據此操作建立或使用之 KMS 金鑰的 KeySpec 屬性值，控制對操作的存取。

您可以使用 IAM 政策中的此條件索引鍵，根據 CreateKey 請求中的 [KeySpec](#) 參數值，控制對 [CreateKey](#) 操作的存取。例如，您可以使用此條件，允許使用者僅建立對稱加密 KMS 金鑰，或僅建立 HMAC KMS 金鑰。

下列範例 IAM 政策陳述式會使用 kms:KeySpec 條件金鑰，允許主體僅建立 RSA 非對稱 KMS 金鑰。僅在請求中的 KeySpec 以 RSA_ 開頭時，許可才有效。

```
{
  "Effect": "Allow",
  "Action": "kms:CreateKey",
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "kms:KeySpec": "RSA_*"
    }
  }
}
```

您也可以使用 kms:KeySpec 條件索引鍵，根據用於操作之 KMS 金鑰的 KeySpec 屬性控制對使用或管理 KMS 金鑰之操作的存取。操作必須是 KMS 金鑰資源操作，也就是授權特定 KMS 金鑰的操作。若要識別 KMS 金鑰資源操作，請在[動作與資源表](#)中尋找操作之 Resources 資料欄的 KMS key 值。

例如，下列 IAM 政策允許主體執行指定的 KMS 金鑰資源操作，但僅能使用帳戶中的對稱加密 KMS 金鑰。

```
{
  "Effect": "Allow",
```

```

"Action": [
  "kms:Encrypt",
  "kms:Decrypt",
  "kms:ReEncrypt*",
  "kms:DescribeKey"
],
"Resource": "arn:aws:kms:us-west-2:111122223333:key/*",
"Condition": {
  "StringEquals": {
    "kms:KeySpec": "SYMMETRIC_DEFAULT"
  }
}
}

```

另請參閱

- [kms:BypassPolicyLockoutSafetyCheck](#)
- [kms:CustomerMasterKeySpec](#) (已被取代)
- [kms:DataKeyPairSpec](#)
- [kms:KeyOrigin](#)
- [kms:KeyUsage](#)

kms:KeyUsage

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:KeyUsage	String	單一值	CreateKey	IAM 政策
			KMS 金鑰資源操作	金鑰政策和 IAM 政策

kms:KeyUsage 條件索引鍵會根據此操作建立或使用之 KMS 金鑰的 KeyUsage 屬性值，控制對操作的存取。

您可以使用此條件索引鍵，根據請求中的 [KeyUsage](#) 參數值，控制對 [CreateKey](#) 操作的存取。的有效值KeyUsage為 ENCRYPT_DECRYPT、GENERATE_VERIFY_MAC、SIGN_VERIFY和 KEY_AGREEMENT。

例如，您只能在 KeyUsage 為 ENCRYPT_DECRYPT 時建立 KMS 金鑰，或在 KeyUsage 為 SIGN_VERIFY 時拒絕使用者許可。

下列範例 IAM 政策陳述式會使用 kms:KeyUsage 條件索引鍵，僅在 KeyUsage 為 ENCRYPT_DECRYPT 時才建立 KMS 金鑰。

```
{
  "Effect": "Allow",
  "Action": "kms:CreateKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:KeyUsage": "ENCRYPT_DECRYPT"
    }
  }
}
```

您也可以使用 kms:KeyUsage 條件索引鍵，根據操作中 KMS 金鑰的 KeyUsage 屬性控制對使用或管理 KMS 金鑰之操作的存取。操作必須是 KMS 金鑰資源操作，也就是授權特定 KMS 金鑰的操作。若要識別 KMS 金鑰資源操作，請在[動作與資源表](#)中尋找操作之 Resources 資料欄的 KMS key 值。

例如，下列 IAM 政策允許主體執行指定的 KMS 金鑰資源操作，但僅能使用帳戶中用於簽署和驗證的 KMS 金鑰。

```
{
  "Effect": "Allow",
  "Action": [
    "kms:CreateGrant",
    "kms:DescribeKey",
    "kms:GetPublicKey",
    "kms:ScheduleKeyDeletion"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:key/*",
  "Condition": {
    "StringEquals": {
      "kms:KeyUsage": "SIGN_VERIFY"
    }
  }
}
```

另請參閱

- [kms:BypassPolicyLockoutSafetyCheck](#)
- [kms:CustomerMasterKeyUsage](#) (已被取代)
- [kms:KeyOrigin](#)
- [kms:KeySpec](#)

kms:MacAlgorithm

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:MacAlgorithm	String	單一值	GenerateMac VerifyMac	金鑰政策和 IAM 政策

您可以使用 kms:MacAlgorithm 條件金鑰，依據請求中的 MacAlgorithm 參數值，控制對 [GenerateMac](#) 和 [VerifyMac](#) 操作的存取。

以下範例金鑰政策允許可擔任 testers 角色的使用者，僅可在請求中的 MAC 演算法為 HMAC_SHA_384 或 HMAC_SHA_512 時，使用 HMAC KMS 金鑰來產生及驗證 HMAC 標籤。此政策會使用兩個不同的政策陳述式，每個陳述式都有各自的條件。若您在單一條件陳述式中指定多個 MAC 演算法，則該條件會同時需要兩個演算法，而非僅任一個演算法。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/testers"
      },
      "Action": [
        "kms:GenerateMac",
        "kms:VerifyMac"
      ],
      "Resource": "*"
    }
  ]
}
```

```

    "Condition": {
      "StringEquals": {
        "kms:MacAlgorithm": "HMAC_SHA_384"
      }
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/testers"
      },
      "Action": [
        "kms:GenerateMac",
        "kms:VerifyMac"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:MacAlgorithm": "HMAC_SHA_512"
        }
      }
    }
  ]
}

```

另請參閱

- [the section called “kms:EncryptionAlgorithm”](#)
- [kms:SigningAlgorithm](#)

kms:MessageType

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:MessageType	String	單一值	Sign Verify	金鑰政策和 IAM 政策

根據請求中的 `MessageType` 參數值，`kms:MessageType` 條件索引鍵會控制對 [Sign](#) 和 [Verify](#) 操作的存取。`MessageType` 的有效值為 `RAW` 和 `DIGEST`。

例如，下列金鑰政策陳述式使用 `kms:MessageType` 條件索引鍵，以使用非對稱 KMS 金鑰簽署訊息，而非訊息摘要。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:Sign",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:MessageType": "RAW"
    }
  }
}
```

另請參閱

- [the section called “kms:SigningAlgorithm”](#)

kms:MultiRegion

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:MultiRegion	Boolean	單一值	CreateKey KMS 金鑰資源操作	金鑰政策和 IAM 政策

您可以使用此條件索引鍵，僅允許對單一區域金鑰的操作，或僅允許對[多區域金鑰](#)的操作。`kms:MultiRegion` 條件金鑰會根據 KMS 金鑰的 `MultiRegion` 屬性值，控制對 KMS 金鑰上 AWS KMS 操作和 [CreateKey](#) 操作的存取。有效值為 `true` (多區域) 和 `false` (單一區域)。所有 KMS 金鑰都有 `MultiRegion` 屬性。

例如，下列 IAM 政策陳述式會使用 `kms:MultiRegion` 條件索引鍵，允許主體僅建立單一區域金鑰。

```
{
  "Effect": "Allow",
  "Action": "kms:CreateKey",
  "Resource": "*",
  "Condition": {
    "Bool": {
      "kms:MultiRegion": false
    }
  }
}
```

kms:MultiRegionKeyType

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:MultiRegionKeyType	String	單一值	CreateKey KMS 金鑰資源操作	金鑰政策和 IAM 政策

您可以使用此條件索引鍵，僅允許對[多區域主要金鑰](#)的操作，或僅允許對[多區域複本金鑰](#)的操作。`kms:MultiRegionKeyType` 條件金鑰會根據 KMS 金鑰的 `MultiRegionKeyType` 屬性，控制對 KMS 金鑰 AWS KMS 和 [CreateKey](#) 操作的存取。有效值為 `PRIMARY` 和 `REPLICA`。只有多區域金鑰具有 `MultiRegionKeyType` 屬性。

一般而言，您需要使用 IAM 政策中的 `kms:MultiRegionKeyType` 條件索引鍵，以控制對多個 KMS 金鑰的存取。不過，由於指定的多區域金鑰可以變更為主要或複本金鑰，因此您可能想要在金鑰政策中使用此條件，僅當特定的多區域金鑰是主要金鑰或複本金鑰時才允許操作。

例如，以下 IAM 政策陳述式使用 `kms:MultiRegionKeyType` 條件索引鍵，以允許主體僅對指定 AWS 帳戶中的多區域複本金鑰排程和取消金鑰刪除。

```
{
  "Effect": "Allow",
  "Action": [
    "kms:ScheduleKeyDeletion",
```

```

    "kms:CancelKeyDeletion"
  ],
  "Resource": "arn:aws:kms:*:111122223333:key/*",
  "Condition": {
    "StringEquals": {
      "kms:MultiRegionKeyType": "REPLICA"
    }
  }
}

```

若要允許或拒絕存取所有多區域金鑰，您可以將兩個值或 Null 值與 `kms:MultiRegionKeyType` 搭配使用。不過，會建議 [kms:MultiRegion](#) 條件索引鍵用於此目的。

kms:PrimaryRegion

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
<code>kms:PrimaryRegion</code>	字串 (清單)	單一值	<code>UpdatePrimaryRegion</code>	金鑰政策和 IAM 政策

您可以使用此條件索引鍵來限制 [UpdatePrimaryRegion](#) 操作中的目的地區域。這些 AWS 區域 可以託管您的多區域主索引鍵。

`kms:PrimaryRegion` 條件索引鍵會根據 `PrimaryRegion` 參數值控制對 [UpdatePrimaryRegion](#) 操作的存取。`PrimaryRegion` 參數會指定要提升為主要 AWS 區域 節點的 [多區域複本金鑰](#) 的。條件的值是一或多個 AWS 區域 名稱，例如 `us-east-1` 或 `ap-southeast-2`，或區域名稱模式，例如 `eu-*`。

例如，以下金鑰政策陳述式使用 `kms:PrimaryRegion` 條件索引鍵，以允許主體將多重區域金鑰的主要區域更新為四個指定區域之一。

```

{
  "Effect": "Allow",
  "Action": "kms:UpdatePrimaryRegion",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/Developer"
  },
  "Resource": "*",

```

```
"Condition": {
  "StringEquals": {
    "kms:PrimaryRegion": [
      "us-east-1",
      "us-west-2",
      "eu-west-3",
      "ap-southeast-2"
    ]
  }
}
```

kms:ReEncryptOnSameKey

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:ReEncryptOnSameKey	Boolean	單一值	ReEncrypt	金鑰政策和 IAM 政策

您可以使用此條件索引鍵，根據請求指定的目的地 CMK 是否與原始加密所用的 KMS 金鑰相同，來控制對 [ReEncrypt](#) 操作的存取。

例如，下列金鑰政策陳述式使用 kms:ReEncryptOnSameKey 條件索引鍵，只在目的地 CMK 與原始加密所用的 KMS 金鑰相同時，才可重新加密。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:ReEncrypt*",
  "Resource": "*",
  "Condition": {
    "Bool": {
      "kms:ReEncryptOnSameKey": true
    }
  }
}
```

kms:RequestAlias

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:RequestAlias	字串 (清單)	單一值	密碼編譯操作 DescribeKey GetPublicKey	金鑰政策和 IAM 政策

您可以使用此條件索引鍵，僅當請求使用特定別名來識別 KMS 金鑰時，才允許操作。kms:RequestAlias 條件索引鍵可根據在請求中識別該 KMS 金鑰的[別名](#)，控制對密碼編譯操作、GetPublicKey 或 DescribeKey 中所使用 KMS 金鑰的存取。(此政策條件不會影響[GenerateRandom](#) 操作，因為操作不會使用 KMS 金鑰或別名)。

此條件支援 [中的屬性型存取控制](#) (ABAC) AWS KMS，可讓您根據 KMS 金鑰的標籤和別名來控制對 KMS 金鑰的存取。您可以使用標籤和別名，來允許或拒絕對 KMS 金鑰的存取，而無需變更政策或授予。如需詳細資訊，請參閱[適用於的 ABAC AWS KMS](#)。

若要在此政策條件中指定別名，請使用[別名名稱](#)，例如 alias/project-alpha，或別名名稱模式，例如 alias/*test*。您不能指定此條件索引鍵值中的[別名 ARN](#)。

若要滿足此條件，請求中的 KeyId 參數值必須為相符的別名名稱或別名 ARN。如果請求使用不同的[金鑰識別符](#)，則其不滿足條件，即使識別為相同的 KMS 金鑰。

例如，下列金鑰政策陳述式允許主體呼叫對 KMS 金鑰的 [GenerateDataKey](#) 操作。但是，僅當請求中的 KeyId 參數值為 alias/finance-key 或具有該別名名稱的別名 ARN，例如 arn:aws:kms:us-west-2:111122223333:alias/finance-key。

```
{
  "Sid": "Key policy using a request alias condition",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/developer"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*",
  "Condition": {
```

```

    "StringEquals": {
      "kms:RequestAlias": "alias/finance-key"
    }
  }
}

```

您無法使用此條件索引鍵來控制對別名操作的存取，例如 [CreateAlias](#) 或 [DeleteAlias](#)。如需控制對別名操作之存取的詳細資訊，請參閱 [控制對別名的存取](#)。

kms:ResourceAliases

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:ResourceAliases	字串 (清單)	多重值	KMS 金鑰資源操作	僅限 IAM 政策

使用此條件索引鍵，根據與 KMS 金鑰相關聯的**別名**，控制對 KMS 金鑰的存取。操作必須是 KMS 金鑰資源操作，也就是授權特定 KMS 金鑰的操作。若要識別 KMS 金鑰資源操作，請在[動作與資源表](#)中尋找操作之 Resources 資料欄的 KMS key 值。

此條件支援 AWS KMS 中的屬性型存取控制 (ABAC)。透過 ABAC，您可以根據指派給 KMS 金鑰的標籤和與 KMS 金鑰相關聯的別名，來控制對 KMS 金鑰的存取。您可以使用標籤和別名，來允許或拒絕對 KMS 金鑰的存取，而無需變更政策或授予。如需詳細資訊，請參閱[適用於的 ABAC AWS KMS](#)。

別名在 AWS 帳戶和區域中必須是唯一的，但此條件可讓您控制對相同區域中多個 KMS 金鑰（使用 StringLike 比較運算子）的存取，或對不同帳戶中多個 KMS 金鑰 AWS 區域的存取。

Note

僅當 KMS 金鑰符合[每個 KMS 金鑰的別名配額](#)時，[kms:ResourceAliases](#) 條件才會有效。如果 KMS 金鑰超過此配額，則會拒絕透過 kms:ResourceAliases 條件授權使用 KMS 金鑰的主體存取 KMS 金鑰。

若要在此政策條件中指定別名，請使用**別名名稱**，例如 alias/project-alpha，或別名名稱模式，例如 alias/*test*。您不能指定此條件索引鍵值中的**別名 ARN**。若要滿足條件，操作中使用的 KMS 金鑰必須具有指定的別名。在操作要求中是否識別或如何識別 KMS 金鑰並不重要。

這是多重值條件索引鍵，會將與 KMS 金鑰相關聯的一組別名與政策中的一組別名進行比較。若要判斷這些集合的比較方式，您必須提供政策條件中的 `ForAnyValue` 或 `ForAllValues` 集合運算子。如需集合運算子的詳細資訊，請參閱《IAM 使用者指南》中的[使用多個索引鍵和值](#)。

- `ForAnyValue`：至少有一個與 KMS 金鑰相關聯的別名必須與政策條件中的別名相符。允許使用其他別名。如果 KMS 金鑰沒有別名，則不符合條件。
- `ForAllValues`：與 KMS 金鑰關聯的每個別名都必須與政策中的別名相符。此集合運算子會將與 KMS 金鑰相關聯的別名限制為政策條件中的別名。它不需要任何別名，但會禁止未指定的別名。

例如，下列 IAM 政策陳述式允許主體在與別名 AWS 帳戶 相關聯的指定 中的任何 KMS 金鑰上呼叫 [GenerateDataKey](#) `finance-key` 操作。(受影響的 KMS 金鑰的金鑰政策也必須允許主體的帳戶使用這些金鑰來進行此操作。) 若要在可能與 KMS 金鑰相關聯的許多別名之一為 `alias/finance-key` 時滿足該條件，條件會使用 `ForAnyValue` 集合運算子。

由於 `kms:ResourceAliases` 條件是根據資源，而非請求，對於任何與 `finance-key` 別名相關聯的 KMS 金鑰的 `GenerateDataKey` 呼叫會成功，即使請求使用[金鑰 ID](#) 或[金鑰 ARN](#) 來識別 KMS 金鑰。

```
{
  "Sid": "AliasBasedIAMPolicy",
  "Effect": "Allow",
  "Action": "kms:GenerateDataKey",
  "Resource": [
    "arn:aws:kms:*:111122223333:key/*",
    "arn:aws:kms:*:444455556666:key/*"
  ],
  "Condition": {
    "ForAnyValue:StringEquals": {
      "kms:ResourceAliases": "alias/finance-key"
    }
  }
}
```

下列範例 IAM 政策陳述式允許主體啟用和停用 KMS 金鑰，但僅當 KMS 金鑰的所有別名都包含 `Test` 時。此政策陳述式使用兩個條件。具有 `ForAllValues` 集合運算子的條件需要所有與 KMS 金鑰相關聯的別名都包含「測試」。具有 `ForAnyValue` 集合運算子的條件需要 KMS 金鑰至少有一個具有「測試」的別名。沒有 `ForAnyValue` 條件時，此政策陳述式會允許主體使用沒有別名的 KMS 金鑰。

```
{
  "Sid": "AliasBasedIAMPolicy",
  "Effect": "Allow",
  "Action": [
    "kms:EnableKey",
    "kms:DisableKey"
  ],
  "Resource": "arn:aws:kms:*:111122223333:key/*",
  "Condition": {
    "ForAllValues:StringLike": {
      "kms:ResourceAliases": [
        "alias/*Test*"
      ]
    },
    "ForAnyValue:StringLike": {
      "kms:ResourceAliases": [
        "alias/*Test*"
      ]
    }
  }
}
```

kms:ReplicaRegion

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:ReplicaRegion	字串 (清單)	單一值	Replicate Key	金鑰政策和 IAM 政策

您可以使用此條件金鑰來限制委託人可以複寫 [多區域金鑰](#) AWS 區域 的。根據請求中的 [ReplicaRegion](#) 參數值，kms:ReplicaRegion 條件索引鍵可控制對 [ReplicateKey](#) 操作的存取。此參數指定為新 [複本金鑰](#) 指定 AWS 區域。

條件的值是一或多個 AWS 區域 名稱，例如 us-east-1 或 ap-southeast-2，或名稱模式，例如 eu-*。如需 AWS 區域 AWS KMS 支援的名稱清單，請參閱 中的 [AWS Key Management Service 端點和配額](#) AWS 一般參考。

例如，以下金鑰政策陳述式使用 kms:ReplicaRegion 條件索引鍵，僅在 ReplicaRegion 參數值為指定區域之一時，才允許主體呼叫 [ReplicateKey](#) 操作。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/Administrator"
  },
  "Action": "kms:ReplicateKey"
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ReplicaRegion": [
        "us-east-1",
        "eu-west-3",
        "ap-southeast-2"
      ]
    }
  }
}
```

此條件索引鍵僅控制對 [ReplicateKey](#) 操作的存取。若要控制對 [UpdatePrimaryRegion](#) 操作的存取，請使用 [kms:PrimaryRegion](#) 條件索引鍵。

kms:RetiringPrincipal

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:RetiringPrincipal	字串 (清單)	單一值	CreateGrant	金鑰政策和 IAM 政策

您可以使用此條件索引鍵，根據請求中的 [RetiringPrincipal](#) 參數值，控制對 [CreateGrant](#) 操作的存取。例如，您可以只在 [CreateGrant](#) 請求中的 [RetiringPrincipal](#) 符合條件陳述式中的 [RetiringPrincipal](#) 時，才可建立授予來使用 KMS 金鑰。

若要指定淘汰委託人，請使用委託人的 AWS Amazon Resource Name (ARN)。有效的委託人包括 AWS 帳戶 IAM 使用者、IAM 角色、聯合身分使用者和擔任的角色使用者。如需主體 ARN 語法的說明，請參閱《[IAM 使用者指南](#)》中的 [IAM ARNs](#)。

下列範例金鑰政策陳述式允許使用者為 KMS 金鑰建立授予。kms:RetiringPrincipal 條件索引鍵會將許可限制為 [CreateGrant](#) 請求，其中授予中的淘汰委託人是 [LimitedAdminRole](#)。


```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:CreateGrant",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:RetiringPrincipal": "arn:aws:iam::111122223333:role/LimitedAdminRole"
    }
  }
}
```

另請參閱

- [kms:GrantConstraintType](#)
- [kms:GrantIsForAWSResource](#)
- [kms:GrantOperations](#)
- [kms:GranteePrincipal](#)

kms:RotationPeriodInDays

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:RotationPeriodInDays	數值	單一值	EnableKeyRotation	金鑰政策和 IAM 政策

您可以使用此條件索引鍵來限制主體可以在 [EnableKeyRotation](#) 請求的 RotationPeriodInDays 參數中指定的值。

RotationPeriodInDays 指定每個自動金鑰輪換日期之間的天數。AWS KMS 可讓您指定介於 90 到 2560 天的輪換期間，但您可以使用 kms:RotationPeriodInDays 條件金鑰進一步限制輪換期間，例如強制執行有效範圍內的最短輪換期間。

例如，下列金鑰政策陳述式使用 `kms:RotationPeriodInDays` 條件金鑰，以防止主體在輪換期間小於或等於 180 天時啟用金鑰輪換。

```
{
  "Effect": "Deny",
  "Action": "kms:EnableKeyRotation",
  "Principal": "*",
  "Resource": "*",
  "Condition": {
    "NumericLessThanEquals": {
      "kms:RotationPeriodInDays": "180"
    }
  }
}
```

`kms:ScheduleKeyDeletionPendingWindowInDays`

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
<code>kms:ScheduleKeyDeletionPendingWindowInDays</code>	數值	單一值	<code>ScheduleKeyDeletion</code>	金鑰政策和 IAM 政策

您可利用此條件金鑰，針對主體對 [ScheduleKeyDeletion](#) 請求指定的 `PendingWindowInDays` 參數值加以限制。

`PendingWindowInDays` 指定刪除金鑰之前 AWS KMS 要等待的天數。AWS KMS 可讓您指定 7 到 30 天的等待期間，但您可以使用 `kms:ScheduleKeyDeletionPendingWindowInDays` 條件金鑰進一步限制等待期間，例如強制執行有效範圍內的最短等待期間。

例如，下列金鑰政策陳述式採用 `kms:ScheduleKeyDeletionPendingWindowInDays` 條件金鑰，以便防止主體在等待期間小於或等於 21 天時，排程刪除金鑰。

```
{
  "Effect": "Deny",
```

```

"Action": "kms:ScheduleKeyDeletion",
"Principal": "*",
"Resource": "*",
"Condition" : {
    "NumericLessThanEquals" : {
        "kms:ScheduleKeyDeletionPendingWindowInDays" : "21"
    }
}
}

```

kms:SigningAlgorithm

AWS KMS 條件 索引鍵	條件類型	值類型	API 操作	Policy type (政策 類型)
kms:Signi ngAlgorit hm	String	單一值	Sign Verify	金鑰政策和 IAM 政策

根據請求中的 [SigningAlgorithm](#) 參數值，您可以使用 kms:SigningAlgorithm 條件金鑰控制對 [Sign](#) 和 [Verify](#) 操作的存取。此條件金鑰不會影響在 外部執行的操作 AWS KMS，例如在 外部的非對稱 KMS 金鑰對中使用公有金鑰驗證簽章 AWS KMS。

下列範例金鑰政策允許可以擔任 testers 角色的使用者只有在請求所用的簽署演算法是 RSASSA_PSS 演算法 (例如 RSASSA_PSS_SHA512) 時，才能使用 KMS 金鑰簽署訊息。

```

{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/testers"
  },
  "Action": "kms:Sign",
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "kms:SigningAlgorithm": "RSASSA_PSS*"
    }
  }
}

```

另請參閱

- [kms:EncryptionAlgorithm](#)
- [the section called “kms:MacAlgorithm”](#)
- [the section called “kms:MessageType”](#)

kms:ValidTo

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:ValidTo	時間戳記	單一值	ImportKeyMaterial	金鑰政策和 IAM 政策

kms:ValidTo 條件索引鍵會根據請求中的 [ValidTo](#) 參數值 (其決定了匯入金鑰材料的過期時間)，控制對 [ImportKeyMaterial](#) 操作的存取。這個值是以 [Unix 時間](#) 表示。

在預設情況下，ImportKeyMaterial 請求需要 ValidTo 參數。不過，如果 [ExpirationModel](#) 參數的值是 KEY_MATERIAL_DOES_NOT_EXPIRE，則 ValidTo 參數無效。您也可以使用 [kms:ExpirationModel](#) 條件金鑰來要求 ExpirationModel 參數或特定的參數值。

以下範例政策陳述式允許使用者將金鑰材料匯入 KMS 金鑰。kms:ValidTo 條件金鑰限制 ImportKeyMaterial 請求的許可，其中 ValidTo 值小於或等於 1546257599.0 (2018 年 12 月 31 日下午 11:59:59)。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:ImportKeyMaterial",
  "Resource": "*",
  "Condition": {
    "NumericLessThanEquals": {
      "kms:ValidTo": "1546257599.0"
    }
  }
}
```

另請參閱

- [kms:ExpirationModel](#)
- [kms:WrappingAlgorithm](#)
- [kms:WrappingKeySpec](#)

kms:ViaService

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:ViaService	String	單一值	KMS 金鑰資源操作	金鑰政策和 IAM 政策

kms:ViaService 條件金鑰會將 KMS 金鑰的使用限制為來自指定的請求 AWS 服務。此條件金鑰僅適用於[轉送存取工作階段](#)。您可以在每個 kms:ViaService 條件金鑰中指定一或多個服務。操作必須是 KMS 金鑰資源操作，也就是授權特定 KMS 金鑰的操作。若要識別 KMS 金鑰資源操作，請在[動作與資源表](#)中尋找操作之 Resources 資料欄的 KMS key 值。

例如，以下金鑰政策陳述式使用 kms:ViaService 條件索引鍵，僅當代表 ExampleRole 的請求來自美國西部 (奧勒岡) 區域的 Amazon EC2 或 Amazon RDS 時，才會允許[客戶受管金鑰](#)用於指定動作。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:CreateGrant",
    "kms:ListGrants",
    "kms:DescribeKey"
  ],
  "Resource": "*",
  "Condition": {
```

```

    "StringEquals": {
      "kms:ViaService": [
        "ec2.us-west-2.amazonaws.com",
        "rds.us-west-2.amazonaws.com"
      ]
    }
  }
}

```

您也可以使用 `kms:ViaService` 條件索引鍵，在請求來自特定的服務時，拒絕使用 KMS 金鑰的許可。例如，以下來自金鑰政策的陳述式使用 `kms:ViaService` 條件索引鍵，當代表 `ExampleRole` 的請求來自 AWS Lambda 時，防止將客戶受管金鑰用於 `Encrypt` 操作。

```

{
  "Effect": "Deny",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": [
    "kms:Encrypt"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ViaService": [
        "lambda.us-west-2.amazonaws.com"
      ]
    }
  }
}

```

Important

當您使用 `kms:ViaService` 條件索引鍵時，該服務會代表 AWS 帳戶中的主體提出請求。這些主體必須擁有以下許可：

- 使用 KMS 金鑰的許可。主體需要授予這些許可給整合服務，以便該服務可以代表主體使用客戶受管金鑰。如需詳細資訊，請參閱[搭配 AWS 服務使用 AWS KMS 加密](#)。
- 使用整合服務的許可。如需讓使用者存取與整合之 AWS 服務的詳細資訊 AWS KMS，請參閱整合服務的文件。

所有 [AWS 受管金鑰](#) 在其金鑰政策文件中使用 `kms:ViaService` 條件索引鍵。只在請求是來自建立 KMS 金鑰的服務時，這個條件才僅允許將 KMS 金鑰用於這些請求。若要查看的金鑰政策 AWS 受管金鑰，請使用 [GetKeyPolicy](#) 操作。

`kms:ViaService` 條件金鑰在 IAM 和金鑰政策陳述式中有效。您指定的服務必須與 [AWS KMS 整合](#)，並支援 `kms:ViaService` 條件金鑰。

支援 **kms:ViaService** 條件金鑰的服務

下表列出與整合 AWS 的服務，AWS KMS 並支援在客戶受管金鑰中使用 `kms:ViaService` 條件金鑰。此表格中的服務可能無法在所有區域中使用。在所有 AWS 分割區中使用 AWS KMS ViaService .amazonaws.com 名稱的尾碼。

Note

您可能需要水平或垂直捲動，才能查看此資料表中的所有資料。

服務名稱	AWS KMS ViaService 名稱
Amazon AI 操作	aiops. <i>AWS_region</i> .amazonaws.com
AWS App Runner	apprunner. <i>AWS_region</i> .amazonaws.com
AWS AppFabric	appfabric. <i>AWS_region</i> .amazonaws.com
Amazon AppFlow	appflow. <i>AWS_region</i> .amazonaws.com
AWS Application Migration Service	mgn. <i>AWS_region</i> .amazonaws.com
Amazon Athena	athena. <i>AWS_region</i> .amazonaws.com
AWS Audit Manager	auditmanager. <i>AWS_region</i> .amazonaws.com
Amazon Aurora	rds. <i>AWS_region</i> .amazonaws.com
AWS Backup	backup. <i>AWS_region</i> .amazonaws.com

服務名稱	AWS KMS ViaService 名稱
AWS Backup 閘道	backup-gateway. <i>AWS_region</i> .amazonaws.com
Amazon Bedrock 模型複製	bedrock. <i>AWS_region</i> .amazonaws.com
Amazon Chime SDK	chimevoiceconnector. <i>AWS_region</i> .amazonaws.com
AWS Clean Rooms ML	cleanrooms-ml. <i>AWS_region</i> .amazonaws.com
AWS CodeArtifact	codeartifact. <i>AWS_region</i> .amazonaws.com
Amazon CodeGuru Reviewer	codeguru-reviewer. <i>AWS_region</i> .amazonaws.com
Amazon Comprehend	comprehend. <i>AWS_region</i> .amazonaws.com
Amazon Connect	connect. <i>AWS_region</i> .amazonaws.com
Amazon Connect Customer Profiles	profile. <i>AWS_region</i> .amazonaws.com
Amazon Q in Connect	wisdom. <i>AWS_region</i> .amazonaws.com
AWS Database Migration Service (AWS DMS)	dms. <i>AWS_region</i> .amazonaws.com
AWS DeepRacer	deepracer. <i>AWS_region</i> .amazonaws.com
AWS Directory Service	directoryservice. <i>AWS_region</i> .amazonaws.com
Amazon DocumentDB	docdb-elastic. <i>AWS_region</i> .amazonaws.com

服務名稱	AWS KMS ViaService 名稱
Amazon DynamoDB	dynamodb. <i>AWS_region</i> .amazonaws.com
Amazon EC2 Systems Manager (SSM)	ssm. <i>AWS_region</i> .amazonaws.com
Amazon Elastic Block Store (Amazon EBS)	ec2. <i>AWS_region</i> .amazonaws.com (僅 EBS)
Amazon Elastic Container Registry (Amazon ECR)	ecr. <i>AWS_region</i> .amazonaws.com
Amazon Elastic File System (Amazon EFS)	elasticfilesystem. <i>AWS_region</i> .amazonaws.com
Amazon ElastiCache	在條件索引鍵值中均包含 ViaService 名稱： • elasticache. <i>AWS_region</i> .amazonaws.com • dax.<i>AWS_region</i> .amazonaws.com
AWS Elemental MediaTailor	mediatailor. <i>AWS_region</i> .amazonaws.com
AWS 實體解析	entityresolution. <i>AWS_region</i> .amazonaws.com
Amazon EventBridge	events. <i>AWS_region</i> .amazonaws.com
Amazon FinSpace	finspace. <i>AWS_region</i> .amazonaws.com
Amazon Forecast	forecast. <i>AWS_region</i> .amazonaws.com
Amazon FSx	fsx. <i>AWS_region</i> .amazonaws.com
AWS Glue	glue. <i>AWS_region</i> .amazonaws.com

服務名稱	AWS KMS ViaService 名稱
AWS Ground Station	groundstation. <i>AWS_region</i> .amazonaws.com
Amazon GuardDuty	malware-protection. <i>AWS_region</i> .amazonaws.com
AWS HealthLake	healthlake. <i>AWS_region</i> .amazonaws.com
AWS IoT SiteWise	iotsitewise. <i>AWS_region</i> .amazonaws.com
Amazon Kendra	kendra. <i>AWS_region</i> .amazonaws.com
Amazon Keyspaces (適用於 Apache Cassandra)	cassandra. <i>AWS_region</i> .amazonaws.com
Amazon Kinesis	kinesis. <i>AWS_region</i> .amazonaws.com
Amazon Data Firehose	firehose. <i>AWS_region</i> .amazonaws.com
Amazon Kinesis Video Streams	kinesisvideo. <i>AWS_region</i> .amazonaws.com
AWS Lambda	lambda. <i>AWS_region</i> .amazonaws.com
Amazon Lex	lex. <i>AWS_region</i> .amazonaws.com
AWS License Manager	license-manager. <i>AWS_region</i> .amazonaws.com
Amazon Location Service	geo. <i>AWS_region</i> .amazonaws.com
Amazon Lookout for Equipment	lookoutequipment. <i>AWS_region</i> .amazonaws.com

服務名稱	AWS KMS ViaService 名稱
Amazon Lookout for Metrics	lookoutmetrics. <i>AWS_region</i> .amazonaws.com
Amazon Lookout for Vision	lookoutvision. <i>AWS_region</i> .amazonaws.com
Amazon Macie	macie. <i>AWS_region</i> .amazonaws.com
AWS Mainframe Modernization	m2. <i>AWS_region</i> .amazonaws.com
AWS Mainframe Modernization 應用程式測試	apptest. <i>AWS_region</i> .amazonaws.com
Amazon Managed Blockchain	managedblockchain. <i>AWS_region</i> .amazonaws.com
Amazon Managed Streaming for Apache Kafka (Amazon MSK)	kafka. <i>AWS_region</i> .amazonaws.com
Amazon Managed Workflows for Apache Airflow (MWAA)	airflow. <i>AWS_region</i> .amazonaws.com
Amazon MemoryDB	memorydb. <i>AWS_region</i> .amazonaws.com
Amazon Monitron	monitron. <i>AWS_region</i> .amazonaws.com
Amazon MQ	mq. <i>AWS_region</i> .amazonaws.com
Amazon Neptune	rds. <i>AWS_region</i> .amazonaws.com
Amazon Nimble Studio	nimble. <i>AWS_region</i> .amazonaws.com
AWS HealthOmics	omics. <i>AWS_region</i> .amazonaws.com
Amazon OpenSearch Service	es. <i>AWS_region</i> .amazonaws.com , aoss. <i>AWS_region</i> .amazonaws.com

服務名稱	AWS KMS ViaService 名稱
Amazon OpenSearch 自訂套件	custom-packages. <i>AWS_region</i> .amazonaws.com
AWS Proton	proton. <i>AWS_region</i> .amazonaws.com
Amazon Quantum Ledger Database (Amazon QLDB)	qldb. <i>AWS_region</i> .amazonaws.com
Amazon RDS Performance Insights	rds. <i>AWS_region</i> .amazonaws.com
Amazon Redshift	redshift. <i>AWS_region</i> .amazonaws.com
Amazon Redshift 查詢編輯器第 2 版	sqlworkbench. <i>AWS_region</i> .amazonaws.com
Amazon Redshift Serverless	redshift-serverless. <i>AWS_region</i> .amazonaws.com
Amazon Rekognition	rekognition. <i>AWS_region</i> .amazonaws.com
Amazon Relational Database Service (Amazon RDS)	rds. <i>AWS_region</i> .amazonaws.com
Amazon Replicated Data Store	ards. <i>AWS_region</i> .amazonaws.com
Amazon SageMaker AI	sagemaker. <i>AWS_region</i> .amazonaws.com
AWS Secrets Manager	secretsmanager. <i>AWS_region</i> .amazonaws.com
Amazon Security Lake	securitylake. <i>AWS_region</i> .amazonaws.com
Amazon Simple Email Service (Amazon SES)	ses. <i>AWS_region</i> .amazonaws.com

服務名稱	AWS KMS ViaService 名稱
Amazon Simple Notification Service (Amazon SNS)	sns. <i>AWS_region</i> .amazonaws.com
Amazon Simple Queue Service (Amazon SQS)	sqs. <i>AWS_region</i> .amazonaws.com
Amazon Simple Storage Service (Amazon S3)	s3. <i>AWS_region</i> .amazonaws.com
Amazon S3 Tables	s3tables. <i>AWS_region</i> .amazonaws.com
AWS Snowball Edge	importexport. <i>AWS_region</i> .amazonaws.com
AWS Step Functions	states. <i>AWS_region</i> .amazonaws.com
AWS Storage Gateway	storagegateway. <i>AWS_region</i> .amazonaws.com
AWS Systems Manager Incident Manager	ssm-incidents. <i>AWS_region</i> .amazonaws.com
AWS Systems Manager Incident Manager 聯絡人	ssm-contacts. <i>AWS_region</i> .amazonaws.com
Amazon Timestream	timestream. <i>AWS_region</i> .amazonaws.com
Amazon Translate	translate. <i>AWS_region</i> .amazonaws.com
AWS Verified Access	verified-access. <i>AWS_region</i> .amazonaws.com
Amazon WorkMail	workmail. <i>AWS_region</i> .amazonaws.com
Amazon WorkSpaces	workspaces. <i>AWS_region</i> .amazonaws.com

服務名稱	AWS KMS ViaService 名稱
Amazon WorkSpaces 精簡型客戶端	thinclient. <i>AWS_region</i> .amazonaws.com
Amazon WorkSpaces Web	workspaces-web. <i>AWS_region</i> .amazonaws.com
AWS X-Ray	xray. <i>AWS_region</i> .amazonaws.com

kms:WrappingAlgorithm

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:WrappingAlgorithm	String	單一值	GetParametersForImport	金鑰政策和 IAM 政策

根據請求中的 [WrappingAlgorithm](#) 參數值，此條件索引鍵會控制對 [GetParametersForImport](#) 操作的存取。您可以使用此條件，要求主體在匯入過程中使用特定的演算法來加密金鑰材料。在指定不同的包裝演算法時，要求所需的公有金鑰和匯入符記會失敗。

下列範例金鑰政策陳述式會使用 kms:WrappingAlgorithm 條件索引鍵來提供範例使用者呼叫 GetParametersForImport 操作的許可，但會防止他們使用 RSAES_OAEP_SHA_1 包裝演算法。GetParametersForImport 請求中的 WrappingAlgorithm 是 RSAES_OAEP_SHA_1 時，操作會失敗。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:GetParametersForImport",
  "Resource": "*",
  "Condition": {
    "StringNotEquals": {
      "kms:WrappingAlgorithm": "RSAES_OAEP_SHA_1"
    }
  }
}
```

```

    }
  }
}
```

另請參閱

- [kms:ExpirationModel](#)
- [kms:ValidTo](#)
- [kms:WrappingKeySpec](#)

kms:WrappingKeySpec

AWS KMS 條件索引鍵	條件類型	值類型	API 操作	Policy type (政策類型)
kms:WrappingKeySpec	String	單一值	GetParametersForImport	金鑰政策和 IAM 政策

根據請求中的 [WrappingKeySpec](#) 參數值，此條件索引鍵會控制對 [GetParametersForImport](#) 操作的存取。您可以使用此條件，要求主體在匯入過程中使用特定類型的公開金鑰。如果請求指定不同的金鑰類型，它會失敗。

由於 WrappingKeySpec 參數值的唯一有效值是 RSA_2048，防止使用者使用此值可以有效避免他們使用 GetParametersForImport 操作。

以下範例政策陳述式使用 kms:WrappingAlgorithm 條件金鑰，要求請求中的 WrappingKeySpec 必須是 RSA_4096。

```

{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:GetParametersForImport",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
```

```
    "kms:WrappingKeySpec": "RSA_4096"
  }
}
```

另請參閱

- [kms:ExpirationModel](#)
- [kms:ValidTo](#)
- [kms:WrappingAlgorithm](#)

AWS KMS 已驗證平台的條件索引鍵

AWS KMS 提供條件金鑰以支援 [AWS Nitro Enclaves](#) 和 NitroTPM 的密碼編譯認證。AWS Nitro Enclaves 是一種 Amazon EC2 功能，可讓您建立稱為 [enclaves](#) 的隔離運算環境，以保護和處理高度敏感的資料。NitroTPM 將類似的認證功能擴展到 EC2 執行個體。

當您使用已簽署的證明文件呼叫

[Decrypt](#)、[DeriveSharedSecret](#)、[GenerateDataKey](#)、[GenerateDataKeyPair](#) 或 [GenerateRandom](#) API 操作時，這些 APIs 會在證明文件的公有金鑰下加密回應中的純文字，並傳回加密文字而非純文字。僅當使用 enclave 的私有金鑰時才能解密此密文。如需詳細資訊，請參閱[中的密碼編譯認證支援 AWS KMS](#)。

Note

如果您在建立金鑰時未提供 AWS KMS 金鑰政策，會為您 AWS 建立一個金鑰政策。此[預設金鑰政策](#)會授予 AWS 帳戶擁有 KMS 金鑰完整存取金鑰的，並允許帳戶使用 IAM 政策來允許存取金鑰。此政策允許 [Decrypt](#) 等所有動作。AWS 建議[最低權限許可](#)將的主體套用至您的 KMS 金鑰政策。您也可以將的 [KMS 金鑰政策動作修改](#)`kms:*`為，以限制存取[NotAction:kms:Decrypt](#)。

下列條件鍵可讓您根據已簽署的證明文件內容，限制這些操作的許可。允許 操作之前，會將證明文件與這些 AWS KMS 條件索引鍵中的值 AWS KMS 進行比較。

Nitro Enclaves 的條件索引鍵

下列條件索引鍵專屬於 Nitro Enclaves 認證：

kms:RecipientAttestation:ImageSha384

AWS KMS 條件金鑰	條件類型	值類型	API 操作	政策類型
kms:RecipientAttestation:ImageSha384	String	單一值	Decrypt DeriveSharedSecret GenerateDataKey GenerateDataKeyPair GenerateRandom	金鑰政策和 IAM 政策

當請求中已簽署證明文件的映像摘要符合條件金鑰中的值時，kms:RecipientAttestation:ImageSha384 條件金鑰 GenerateRandom 會使用 KMS 金鑰控制對 Decrypt DeriveSharedSecret GenerateDataKeyPair、GenerateDataKey、和的存取。ImageSha384 值對應證明文件的 PCR0。只有在請求中的 Recipient 參數指定 AWS Nitro enclave 的已簽署證明文件時，此條件金鑰才有效。

此值也包含在對 AWS KMS for Nitro enclaves 的請求的 [CloudTrail 事件](#) 中。

例如，下列金鑰政策陳述式允許 data-processing 角色將 KMS 金鑰用於 [Decrypt](#)、[DeriveSharedSecret](#)、[GenerateDataKey](#)、[GenerateDataKeyPair](#) 和 [GenerateRandom](#) 操作。僅當請求中證明文件的影像摘要值 (PCR0) 符合條件的影像摘要值時，kms:RecipientAttestation:ImageSha384 條件金鑰才會允許操作。只有在請求中的 Recipient 參數指定 AWS Nitro enclave 的已簽署證明文件時，此條件金鑰才有效。

如果請求不包含來自 AWS Nitro enclave 的有效證明文件，則會拒絕許可，因為不符合此條件。

```
{
  "Sid" : "Enable enclave data processing",
  "Effect" : "Allow",
  "Principal" : {
```

```
    "AWS" : "arn:aws:iam::111122223333:role/data-processing"
  },
  "Action": [
    "kms:Decrypt",
    "kms:DeriveSharedSecret",
    "kms:GenerateDataKey",
    "kms:GenerateDataKeyPair",
    "kms:GenerateRandom"
  ],
  "Resource" : "*",
  "Condition": {
    "StringEqualsIgnoreCase": {
      "kms:RecipientAttestation:ImageSha384":
"9fedcba8abcdef7abcdef6abcdef5abcdef4abcdef3abcdef2abcdef1abcdef1abcdef0abcdef1abcdef2abcdef3a
    }
  }
}
```

kms:RecipientAttestation:PCR<PCR_ID>

AWS KMS 條件 金鑰	條件類型	值類型	API 操作	政策類型
kms:Recip ientAttes tation:PC R<PCR_ID>	String	單一值	Decrypt DeriveSha redSecret Generated ataKey Generated ataKeyPair GenerateR andom	金鑰政策和 IAM 政策

kms:RecipientAttestation:PCR<PCR_ID> 只有在請求中簽署的證明文
件的平台組態註冊 PCR) 符合條件金鑰中的 PCR) 時，條件金鑰才會控制對
DecryptDeriveSharedSecretGenerateDataKeyGenerateDataKeyPair、、、和

GenerateRandom 的存取。只有在請求中的 Recipient 參數指定來自 AWS Nitro enclave 的已簽署證明文件時，此條件索引鍵才有效。

此值也包含在 [CloudTrail 事件中](#)，[這些事件](#) 代表對 AWS KMS 的 Nitro enclaves 請求。

若要指定 PCR 值，請使用以下格式。將 PCR ID 串連到條件索引鍵名稱。您可以指定識別 [六個 enclave 測量](#) 之一的 PCR ID，或是您為特定使用案例定義的自訂 PCR ID。PCR 值必須是最多 96 個位元組的小寫十六進位字串。

```
"kms:RecipientAttestation:PCRPCR_ID": "PCR_value"
```

例如，下列條件金鑰指定 PCR1 的特定值，對應於用於 enclave 與啟動程序處理的核心雜湊。

```
kms:RecipientAttestation:PCR1:
"abc1abcdef2abcdef3abcdef4abcdef5abcdef6abcdef7abcdef8abcdef9abcdef8abcdef7abcdef6abcdef5abcde
```

下列範例金鑰政策陳述式允許 data-processing 角色利用 KMS 金鑰進行 [Decrypt](#) 操作。

僅當請求中已簽署證明文件中的 PCR1 值符合條件中的 kms:RecipientAttestation:PCR1 值時，本陳述式中的 kms:RecipientAttestation:PCR 條件索引鍵才會允許操作。使用 StringEqualsIgnoreCase 政策運算子要求 PCR 值不區分大小寫的比較。

如請求不含證明文件，則會因未滿足此條件而拒絕許可。

```
{
  "Sid" : "Enable enclave data processing",
  "Effect" : "Allow",
  "Principal" : {
    "AWS" : "arn:aws:iam::111122223333:role/data-processing"
  },
  "Action": "kms:Decrypt",
  "Resource" : "*",
  "Condition": {
    "StringEqualsIgnoreCase": {
      "kms:RecipientAttestation:PCR1":
      "abc1de4f2dcf774f6e3b679f62e5f120065b2e408dcea327bd1c9dddaea6664e7af7935581474844767453082c6f1"
    }
  }
}
```

NitroTPM 的條件索引鍵

下列條件金鑰專屬於 NitroTPM 認證：

kms:RecipientAttestation : NitroTPMPCR<PCR_ID>

AWS KMS 條件金鑰	條件類型	值類型	API 操作	政策類型
kms:RecipientAttestation:NitroTPMPCR<PCR_ID>	String	單一值	Decrypt DeriveSharedSecret GenerateDataKey GenerateDataKeyPair GenerateRandom	金鑰政策和 IAM 政策

kms:RecipientAttestation:NitroTPMPCR<PCR_ID> 只有在請求中簽署的證明文件的平台組態註冊 PCR(s) 符合條件金鑰中的 PCR(s) 時，條件金鑰才會控制對 GenerateDataKeyPair、DecryptDeriveSharedSecretGenerateDataKey、和 GenerateRandom 的存取。只有在請求中的 Recipient 參數指定來自 NitroTPM 的已簽署證明文件時，此條件金鑰才有效。

此值也包含在代表對 AWS KMS for NitroTPM 請求的 [CloudTrail 事件](#) 中。

若要指定 PCR 值，請使用以下格式。將 PCR ID 串連到條件索引鍵名稱。PCR 值必須是最多 96 個位元組的小寫十六進位字串。

```
"kms:RecipientAttestation:NitroTPMPCRPCR_ID": "PCR_IDvalue"
```

例如，下列條件索引鍵會指定 PCR4 的特定值：

```
kms:RecipientAttestation:NitroTPMPCR4:
```

```
"abc1abcdef2abcdef3abcdef4abcdef5abcdef6abcdef7abcdef8abcdef9abcdef8abcdef7abcdef6abcdef5abcde
```

下列範例金鑰政策陳述式允許 data-processing 角色利用 KMS 金鑰進行 [Decrypt](#) 操作。

此陳述式中的 kms:RecipientAttestation:NitroTPMPCR 條件索引鍵僅在請求中已簽署證明文件中的 PCR4 值符合條件中的 kms:RecipientAttestation:NitroTPMPCR4 值時，才允許操作。使用 StringEqualsIgnoreCase 政策運算子要求 PCR 值不區分大小寫的比較。

如請求不含證明文件，則會因未滿足此條件而拒絕許可。

```
{
  "Sid" : "Enable NitroTPM data processing",
  "Effect" : "Allow",
  "Principal" : {
    "AWS" : "arn:aws:iam::111122223333:role/data-processing"
  },
  "Action": "kms:Decrypt",
  "Resource" : "*",
  "Condition": {
    "StringEqualsIgnoreCase": {
      "kms:RecipientAttestation:NitroTPMPCR4":
      "abc1de4f2dcf774f6e3b679f62e5f120065b2e408dcea327bd1c9dddaea6664e7af7935581474844767453082c6f1
    }
  }
}
```

最低權限許可

由於 KMS 金鑰會保護敏感資訊，因此建議您遵循最低權限存取原則。當您定義金鑰政策時，委派執行任務所需的最低許可。只有在您計劃使用其他 IAM 政策進一步限制許可時，才允許 KMS 金鑰政策上的所有動作 (kms:*)。如果您計劃使用 IAM 政策管理許可，請限制誰能夠建立 IAM 政策並將其連接到 IAM 主體，並[監控政策變更](#)。

如果您同時允許金鑰政策和 IAM 政策中的所有動作 (kms:*)，委託人會同時擁有 KMS 金鑰的管理和使用許可。作為安全最佳實務，我們建議僅將這些許可委派給特定委託人。您可以透過在金鑰政策中明確命名主體，或限制 IAM 政策連接的主體來執行此操作。您也可以使用[條件金鑰](#)來限制許可。例如，如果發出 API 呼叫的委託人具有條件規則中指定的標籤，您可以使用 [aws:PrincipalTag](#) 來允許所有動作。

如需了解政策陳述式如何評估的說明 AWS，請參閱《IAM 使用者指南》中的[政策評估邏輯](#)。我們建議您在撰寫政策之前檢閱此主題，以減少政策產生意外效果的機會，例如提供存取權給不應存取的主體。

Tip

在非生產環境中測試應用程式時，請使用 [IAM Access Analyzer](#) 協助您將最低權限套用至 IAM 政策。

如果您使用 IAM 使用者而非 IAM 角色，強烈建議您啟用 AWS [多重要素驗證](#) (MFA)，以減輕長期憑證的漏洞。您可以使用 MFA 執行下列操作：

- 要求使用者在執行特殊權限動作之前，使用 MFA 驗證其登入資料，例如排程金鑰刪除。
- 管理員帳戶密碼和 MFA 裝置在個人之間分割擁有權，以實作分割授權。

進一步了解

- [AWS 任務函數的 受管政策](#)
- [用於寫入最低權限 IAM 政策的技巧](#)

實作最低權限的許可

當您授予 AWS 服務使用 KMS 金鑰的許可時，請確定該許可僅適用於服務必須代表您存取的資源。此最低權限策略有助於防止 AWS 在服務之間傳遞請求時，未經授權使用 KMS 金鑰。

若要實作最低權限策略，建議您使用 AWS KMS 加密內容條件金鑰和全域來源 ARN 或來源帳戶條件金鑰。

使用加密內容條件金鑰

在使用 AWS KMS 資源時實作最低權限許可的最有效方法是在政策中包含 [kms:EncryptionContext:context-key](#) 或 [kms:EncryptionContextKeys](#) 條件金鑰，允許主體呼叫 AWS KMS 密碼編譯操作。這些條件金鑰特別有效，因為它們會將許可與在資源加密時繫結至加密文字的[加密內容](#)建立關聯。

只有在政策陳述式中的動作為 [CreateGrant](#) AWS KMS 或採用 `EncryptionContext` 參數的對稱密碼編譯操作，例如 [GenerateDataKey](#) 或 [Decrypt](#) 等操作時，才能使用加密內容條件金鑰。(如需受支援操

作的清單，請參閱 [kms:EncryptionContext:context-key](#) 或 [kms:EncryptionContextKeys](#)。) 如果您使用這些條件金鑰來允許其他操作，例如 [DescribeKey](#)，則許可會遭到拒絕。

將值設定為服務在加密資源時使用的加密內容。此資訊通常可在服務文件的「安全」一章中取得。例如，[AWS Proton 的加密內容](#)會識別 AWS Proton 資源及其相關聯的範本。[AWS Secrets Manager 加密內容](#)會識別秘密及其版本。[Amazon Location 的加密內容](#)會識別追蹤器或收集。

下列範例金鑰政策陳述式允許 Amazon Location Service 代表授權使用者建立授予。此原則陳述式會藉由使用 [kms:ViaService](#)、[kms:CallerAccount](#) 和 `kms:EncryptionContext:context-key` 條件金鑰，將許可綁定至特定追蹤器資源來限制許可。

```
{
  "Sid": "Allow Amazon Location to create grants on behalf of authorized users",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/LocationTeam"
  },
  "Action": "kms:CreateGrant",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ViaService": "geo.us-west-2.amazonaws.com",
      "kms:CallerAccount": "111122223333",
      "kms:EncryptionContext:aws:geo:arn": "arn:aws:geo:us-west-2:111122223333:tracker/SAMPLE-Tracker"
    }
  }
}
```

使用 `aws:SourceArn` 或 `aws:SourceAccount` 條件金鑰

當主要政策陳述式中的主體是 [AWS 服務主體時](#)，強烈建議您除了 `kms:EncryptionContext:context-key` 條件金鑰之外，再使用 [aws:SourceArn](#) 或 [aws:SourceAccount](#) 全域條件金鑰。只有在 AWS KMS 請求來自其他服務時，授權內容才會包含 ARN 和帳戶值 AWS。這個條件組合會實作最低權限許可，並避免潛在的[混淆代理人案例](#)。服務主體通常不會用作金鑰政策中的主體，但某些 AWS 服務需要它 AWS CloudTrail，例如。

若要使用 `aws:SourceArn` 或 `aws:SourceAccount` 全域條件金鑰，請將值設定為要加密之資源的 Amazon Resource Name (ARN) 或帳戶。例如，在給予 AWS CloudTrail 許可可以加密追蹤的金鑰政策陳述式中，將 `aws:SourceArn` 的值設定為追蹤的 ARN。盡可能使用 `aws:SourceArn`，這

更為具體。將值設定為 ARN 或具有萬用字元的 ARN 模式。如果您不知道資源的 ARN，請改為使用 `aws:SourceAccount`。

Note

如果資源 ARN 包含 AWS KMS 金鑰政策中不允許的字元，則無法在 `aws:SourceArn` 條件金鑰的值中使用該資源 ARN。請改用 `aws:SourceAccount` 條件金鑰。如需有關金鑰政策文件規則的詳細資訊，請參閱[金鑰政策格式](#)。

在下列範例金鑰政策中，取得許可的主體是 AWS CloudTrail 服務主體 `cloudtrail.amazonaws.com`。為了實作最低權限，此政策會使用 `aws:SourceArn` 和 `kms:EncryptionContext:context-key` 條件金鑰。政策陳述式允許 CloudTrail 使用 KMS 金鑰[產生資料金鑰](#)，它可用來加密追蹤。獨立評估 `aws:SourceArn` 和 `kms:EncryptionContext:context-key` 條件。針對指定操作使用 KMS 金鑰的任何請求都必須滿足這兩個條件。

若要將服務的許可限制為範例帳戶 (111122223333) 和 `us-west-2` 區域中的 `finance` 追蹤，此政策陳述式會將 `aws:SourceArn` 條件金鑰設定為特定追蹤的 ARN。條件陳述式使用 [ArnEquals](#) 運算子來確保在比對時獨立評估 ARN 中的每個元素。此範例也會使用 `kms:EncryptionContext:context-key` 條件金鑰來限制特定帳戶和區域中追蹤的許可。

在使用此金鑰政策之前，請將範例帳戶 ID、區域和追蹤名稱取代為您帳戶的有效值。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCloudTrailToEncryptLogs",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudtrail.amazonaws.com"
      },
      "Action": "kms:GenerateDataKey",
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
          "aws:SourceArn": [
```



```

        "arn:aws:cloudtrail:us-west-2:111122223333:trail/finance"
    ],
    },
    "StringLike": {
        "kms:EncryptionContext:aws:cloudtrail:arn": [
            "arn:aws:cloudtrail:*:111122223333:trail/*"
        ]
    }
}
}
]
}
}

```

適用於 的 ABAC AWS KMS

屬性型存取控制 (ABAC) 是一種授權策略，可根據屬性定義許可。AWS KMS 支援 ABAC，可讓您根據與 KMS 金鑰相關聯的標籤和別名來控制對客戶受管金鑰的存取。在 中啟用 ABAC 的標籤和別名條件金鑰 AWS KMS 提供強大且靈活的方式，以授權主體使用 KMS 金鑰，而無需編輯政策或管理授予。但是，您應該謹慎使用這些功能，避免委託人意外被允許或拒絕存取。

如果您使用 ABAC，請注意管理標籤和別名的許可現在是存取控制許可。部署依存於標籤或別名的政策之前，請務必先知曉所有 KMS 金鑰上的現有標籤和別名。新增、刪除和更新別名，以及標記和取消標記金鑰時，請採取合理的預防措施。只將管理標籤和別名的許可授予需要的委託人，並限制他們可以管理的標籤和別名。

備註

使用 ABAC for 時 AWS KMS，請謹慎授予主體管理標籤和別名的許可。變更標籤或別名可能會允許或拒絕 KMS 金鑰的許可。沒有變更主要政策或建立授權之許可的重要管理員可以控制 KMS 金鑰的存取，如果他們擁有管理標籤或別名的許可。

可能最多需要五分鐘才能將標籤和別名變更體現在 KMS 金鑰授權上。最近的變更可能會在 API 操作中可見，然後才會影響授權。

若要根據 KMS 金鑰的別名來控制對 KMS 金鑰的存取，您必須使用條件索引鍵。您不能使用別名來代表政策陳述式 Resource 元素中的 KMS 金鑰。當別名出現在 Resource 元素時，政策陳述式會套用至別名，而不是相關聯的 KMS 金鑰。

進一步了解

- 如需 ABAC AWS KMS 支援的詳細資訊，包括範例，請參閱 [使用別名來控制對 KMS 金鑰的存取](#) 和 [使用標籤來控制對 KMS 金鑰的存取](#)。
- 如需使用標籤控制資源 AWS 存取的一般資訊，請參閱《IAM 使用者指南》中的 [ABAC 是什麼 AWS ?](#) 以及 [使用資源標籤控制 AWS 資源存取](#)。

的 ABAC 條件金鑰 AWS KMS

若要根據 KMS 金鑰的標籤和別名授權存取，請在金鑰政策或 IAM 政策中使用下列條件索引鍵。

ABAC 條件索引鍵	Description	Policy type (政策類型)	AWS KMS 操作
aws:ResourceTag	KMS 金鑰上的標籤 (鍵和值) 與政策中的標籤 (鍵和值) 或標籤模式相符	僅限 IAM 政策	KMS 金鑰資源操作 ²
aws:RequestTag/tag-key	請求中的標籤 (鍵和值) 與政策中的標籤 (鍵和值) 或標籤模式相符	金鑰政策和 IAM 政策 ¹	TagResource 、 UntagResource
aws:TagKeys	請求中的標籤索引鍵與政策中的標籤索引鍵相符	金鑰政策和 IAM 政策 ¹	TagResource 、 UntagResource
kms:ResourceAliases	與 KMS 金鑰相關聯的別名符合政策中的別名或別名模式	僅限 IAM 政策	KMS 金鑰資源操作 ²
kms:RequestAlias	代表請求中 KMS 金鑰的別名與政策中的別名或別名模式相符。	金鑰政策和 IAM 政策 ¹	Cryptographic operations 、 DescribeKey 、 GetPublicKey

¹ 可在金鑰政策中使用的任何條件索引鍵也可以在 IAM 政策中使用，但只能在[金鑰政策允許](#)的情形下。

²KMS 金鑰資源操作是針對特定 KMS 金鑰授權的操作。若要識別 KMS 金鑰資源操作，請在 [AWS KMS 許可表](#) 中尋找 Resources 欄 KMS 金鑰的值，以執行操作。

例如，您可以使用這些條件索引鍵來建立下列政策。

- 具有 `kms:ResourceAliases` 的 IAM 政策，允許使用具有特定別名或別名模式的 KMS 金鑰的許可。這與依賴標籤的政策略有不同：雖然您可以在政策中使用別名模式，但每個別名在 AWS 帳戶和區域中必須是唯一的。這可讓您將政策套用至一組精選 KMS 金鑰，而不會在政策陳述式中列出 KMS 金鑰的金鑰 ARN。若要為集合新增 KMS 金鑰或從中移除 KMS 金鑰，請變更 KMS 金鑰的別名。
- 具有 `kms:RequestAlias` 的金鑰政策允許委託人在 `Encrypt` 操作中使用 KMS 金鑰，但僅在 `Encrypt` 請求會使用該別名來識別 KMS 金鑰的情形下。
- 具有 `aws:ResourceTag/tag-key` 的 IAM 政策拒絕將 KMS 金鑰與特定標籤索引鍵和標籤值使用的許可。這可讓您將政策套用至一組精選 KMS 金鑰，而不會在政策陳述式中列出 KMS 金鑰的金鑰 ARN。若要為集合新增 KMS 金鑰或從中移除 KMS 金鑰，請標記或取消標記 KMS 金鑰。
- 具有 `aws:RequestTag/tag-key` 的 IAM 政策允許委託人只刪除 `"Purpose"="Test"` KMS 金鑰標籤。
- 具有 `aws:TagKeys` 的 IAM 政策拒絕使用 `Restricted` 標籤索引鍵標記或取消標記 KMS 金鑰的許可。

ABAC 讓存取管理具有靈活性和可擴展性。例如，您可以使用 `aws:ResourceTag/tag-key` 條件索引鍵來建立 IAM 政策，此政策只允許委託人在 KMS 金鑰具有 `Purpose=Test` 標籤時針對特定操作使用 KMS 金鑰。此政策適用於 AWS 帳戶所有區域中的所有 KMS 金鑰。

當連接至使用者或角色時，下列 IAM 政策允許委託人將所有現有的 KMS 金鑰與 `Purpose=Test` 標籤搭配用於指定的操作。若要將此存取權提供給新的或現有的 KMS 金鑰，則您不需要變更政策。只需將 `Purpose=Test` 標籤連接至 KMS 金鑰。同樣地，若要從具有 `Purpose=Test` 標籤的 KMS 金鑰中移除此存取權，請編輯或刪除標籤。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AliasBasedIAMPolicy",
      "Effect": "Allow",
```

```
"Action": [
  "kms:Decrypt",
  "kms:Encrypt",
  "kms:GenerateDataKey*",
  "kms:DescribeKey"
],
"Resource": "arn:aws:kms:*:111122223333:key/*",
"Condition": {
  "StringEquals": {
    "aws:ResourceTag/Purpose": "Test"
  }
}
}
```

但是，如果您使用此功能，則請謹慎管理標籤和別名。新增、變更或刪除標籤或別名可能會意外允許或拒絕對 KMS 金鑰的存取。如果他們有管理標籤和別名的許可，則沒有變更主要政策或建立授予之許可的金鑰管理員可以控制對 KMS 金鑰的存取。為了減輕這種風險，請考慮[限制管理標籤的許可](#)和[別名](#)。例如，您可能只允許選取的委託人管理 Purpose=Test 標籤。如需詳細資訊，請參閱[使用別名來控制對 KMS 金鑰的存取](#)和[使用標籤來控制對 KMS 金鑰的存取](#)。

標籤或別名？

AWS KMS 支援具有標籤和別名的 ABAC。這兩種選項都提供靈活、可擴展的存取控制策略，但彼此略有不同。

您可以根據您的特定使用模式，決定 AWS 使用標籤或使用別名。例如，如果您已將標記許可指定給大多數管理員，則根據別名來控制授權策略可能會比較容易。或者，如果接近[每個 KMS 金鑰的別名配額](#)，則您可能會偏好基於標籤的授權策略。

以下是一般利益的優勢。

標籤型存取控制的優勢

- 適用於不同資源類型的相同授權機制 AWS。

您可以使用相同的標籤或標籤金鑰來控制對多種資源類型的存取，例如 Amazon Relational Database Service (Amazon RDS) 叢集、Amazon Elastic Block Store (Amazon EBS) 磁碟區和 KMS 金鑰。此功能可啟用數種不同的授權模式，這些模式比傳統的角色型存取控制更靈活。

- 授權一組 KMS 金鑰的存取權。

您可以使用標籤來管理對相同 AWS 帳戶 和區域中一組 KMS 金鑰的存取。將相同的標籤或標籤金鑰指派給您選擇的 KMS 金鑰。然後建立以標籤或標籤索引鍵為基礎的簡單、易於維護的政策陳述式。若要為授權群組新增 KMS 金鑰或從中移除金鑰，請新增或移除標籤；您無需編輯政策。

別名型存取控制的優勢

- 根據別名授權存取密碼編譯操作。

大部分屬性的請求型政策條件，包括 [aws:RequestTag/tag-key](#)，僅影響新增、編輯或刪除屬性的操作。但 [kms:RequestAlias](#) 條件索引鍵根據用於識別請求中 KMS 金鑰的別名控制對密碼編譯操作的存取。例如，您可以授予委託人在 Encrypt 操作中使用 KMS 金鑰的許可，但僅在 KeyId 參數值為 alias/restricted-key-1 的情形下。若要滿足此條件，需要下列所有項目：

- KMS 金鑰必須與該別名相關聯。
- 請求必須使用別名來識別 KMS 金鑰。
- 委託人必須擁有許可才能使用受限於 kms:RequestAlias 條件的 KMS 金鑰。

如果您的應用程式通常使用別名名稱或別名 ARN 來引用 KMS 金鑰，則這會特別有用。

- 提供非常有限的許可。

別名在 AWS 帳戶 和 區域中必須是唯一的。因此，授予委託人根據別名存取 KMS 金鑰的許可，可能比為委託人授予標籤型存取權更嚴格。與別名不同的是，標籤可指派給相同帳戶和區域中的多個 KMS 金鑰。如果選擇，則您可以使用別名模式，例如 alias/test*，讓委託人可以存取相同帳戶和區域中的一組 KMS 金鑰。不過，允許或拒絕存取特定別名，實現了對 KMS 金鑰非常嚴格的控制。

對適用於 的 ABAC 進行故障診斷 AWS KMS

根據 KMS 金鑰的標籤和別名來控制對 KMS 金鑰的存取非常方便且功能強大。但是，很容易出現一些您想要防止的可預測錯誤。

存取因標籤變更而變更

如果刪除標籤或變更其值，則只能根據該標籤存取 KMS 金鑰的委託人將被拒絕存取 KMS 金鑰。當拒絕政策陳述式中包含的標籤新增至 KMS 金鑰時，也可能會發生這種情況。將政策相關標籤新增至 KMS 金鑰可以允許存取應被拒絕存取 KMS 金鑰的委託人。

例如，假設委託人可以根據 Project=Alpha 標籤存取 KMS 金鑰，例如下列範例 IAM 政策陳述式所提供的許可。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPolicyWithResourceTag",
      "Effect": "Allow",
      "Action": [
        "kms:GenerateDataKeyWithoutPlaintext",
        "kms:Decrypt"
      ],
      "Resource": "arn:aws:kms:ap-southeast-1:111122223333:key/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Project": "Alpha"
        }
      }
    }
  ]
}
```

如果從該 KMS 金鑰刪除標籤或標籤值變更，則委託人就不再具有使用 KMS 金鑰進行指定操作的許可。當委託人嘗試在使用客戶受管金鑰的 AWS 服務中讀取或寫入資料時，這可能會變得很明顯。若要追蹤標籤變更，請檢閱 [TagResource](#) 或 [UntagResource](#) 項目的 CloudTrail 日誌。

若要在不更新政策的情況下還原存取權，請變更 KMS 金鑰上的標籤。這個動作的影響最小，除了很短的一段時間，該動作會在整個 AWS KMS 中有效。為了防止類似錯誤，請僅將標記和取消標記許可提供給需要的委託人，並將其標記許可限制為他們需要管理的標籤。變更標籤之前，搜尋政策可偵測依存於標籤的存取權，並在具有該標籤的所有區域中取得 KMS 金鑰。當特定標籤變更時，您可能會考慮建立 Amazon CloudWatch 警示。

存取因別名變更而變更

如果別名遭到刪除或與不同的 KMS 金鑰相關聯，則只能以該別名為基礎存取 KMS 金鑰的委託人將會被拒絕存取 KMS 金鑰。當與 KMS 金鑰相關聯的別名包含在拒絕政策陳述式中時，也可能會發生這種情況。將政策相關別名新增至 KMS 金鑰也可以允許存取應被拒絕存取 KMS 金鑰的委託人。

例如，下列 IAM 政策陳述式使用 [kms:ResourceAliases](#) 條件索引鍵，以允許使用任何指定的別名存取帳戶中不同區域的 KMS 金鑰。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AliasBasedIAMPolicy",
      "Effect": "Allow",
      "Action": [
        "kms:List*",
        "kms:Describe*",
        "kms:Decrypt"
      ],
      "Resource": "arn:aws:kms:*:111122223333:key/*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "kms:ResourceAliases": [
            "alias/ProjectAlpha",
            "alias/ProjectAlpha_Test",
            "alias/ProjectAlpha_Dev"
          ]
        }
      }
    }
  ]
}
```

若要追蹤別名變更，請檢閱 CloudTrail 日誌，以取得 [CreateAlias](#)、[UpdateAlias](#) 和 [DeleteAlias](#) 項目。

若要在不更新政策的情況下還原存取權，請變更與 KMS 金鑰相關聯的別名。由於每個別名只能與帳戶和區域中的一個 KMS 金鑰相關聯，因此管理別名會比管理標籤困難。還原某個 KMS 金鑰上的某些委託人的存取，可以拒絕相同或其他委託人存取不同的 KMS 金鑰。

若要避免發生此錯誤，請僅將別名管理許可提供給需要的委託人，並將[其別名管理許可](#)限制為需要管理的別名。在更新或刪除別名之前，搜尋政策以偵測取決於別名的存取權，並在與別名相關聯的所有區域中尋找 KMS 金鑰。

因別名配額而拒絕存取

如果 KMS 金鑰超過該帳戶和區域中[每個 KMS 金鑰別名](#)的預設配額，則已授權透過 [kms:ResourceAliases](#) 條件使用 KMS 金鑰的使用者會遇到 AccessDenied 例外狀況。

若要還原存取權，請刪除與 KMS 金鑰相關聯的別名，使其符合配額。或者使用替代機制，讓使用者存取 KMS 金鑰。

延遲的授權變更

您對標籤和別名所做的變更可能需要最長五分鐘才會體現在 KMS 金鑰授權上。因此，標籤或別名變更可能會反映在 API 操作的回應中，然後才會影響授權。此延遲可能比影響大多數 AWS KMS 操作的短暫最終一致性延遲更長。

例如，您可能擁有 IAM 政策，允許特定委託人將任何 KMS 金鑰與 "Purpose"="Test" 標籤搭配使用。然後，您將 "Purpose"="Test" 標籤新增至 KMS 金鑰。雖然 [TagResource](#) 操作已完成，[ListResourceTags](#) 回應會確認標籤已指派給 KMS 金鑰，但委託人最多可能在五分鐘內無法存取 KMS 金鑰。

若要防止錯誤，請將此預期延遲建置到您的程式碼中。

因別名更新而失敗的請求

當更新別名時，您會將現有的別名關聯至不同的 KMS 金鑰。

指定[別名名稱](#)或[別名 ARN](#) [Decrypt](#) 和 [ReEncrypt](#) 請求可能會失敗，因為別名現在與未加密文字的 KMS 金鑰相關聯。這種情況通常會傳回 IncorrectKeyException 或 NotFoundException。或者，如果請求沒有 KeyId 或 DestinationKeyId 參數，則操作可能會失敗，並顯示 AccessDenied 例外狀況，因為呼叫者不再具有加密文字之 KMS 金鑰的存取權。

您可以透過查看 CloudTrail 日誌來取得 [CreateAlias](#)、[UpdateAlias](#) 和 [DeleteAlias](#) 項目，從而追蹤變更。您也可以使用 [ListAliases](#) 回應中 LastUpdatedDate 欄位的值來偵測變更。

例如，下列 [ListAliases](#) 範例回應顯示了 kms:ResourceAliases 條件中 ProjectAlpha_Test 別名已更新。因此，具有別名型存取權的委託人會失去對先前關聯 KMS 金鑰的存取權。相反地，他們可以存取新關聯的 KMS 金鑰。

```
$ aws kms list-aliases --query 'Aliases[?starts_with(AliasName, `alias/ProjectAlpha`)]'
{
```



```

"Aliases": [
  {
    "AliasName": "alias/ProjectAlpha_Test",
    "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/ProjectAlpha_Test",
    "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
    "CreationDate": 1566518783.394,
    "LastUpdatedDate": 1605308931.903
  },
  {
    "AliasName": "alias/ProjectAlpha_Restricted",
    "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/ProjectAlpha_Restricted",
    "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "CreationDate": 1553410800.010,
    "LastUpdatedDate": 1553410800.010
  }
]
}

```

這項變更的補救措施並不簡單。您可以再次更新別名，將別名與原始 KMS 金鑰相關聯。不過，在採取行動之前，您需要考慮該變更對目前相關聯 KMS 金鑰的影響。如果委託人在密碼編譯操作中使用後一個 KMS 金鑰，則他們可能需要繼續存取該金鑰。在此情形下，您可能想要更新政策，以確保委託人擁有使用這兩個 KMS 金鑰的許可。

您可以避免這樣的錯誤：在更新別名之前，搜尋政策以偵測取決於別名的存取權。然後在與別名相關聯的所有區域中取得 KMS 金鑰。請僅將別名管理許可提供給需要的委託人，並將[其別名管理許可](#)限制為需要管理的別名。

的 RBAC AWS KMS

角色型存取控制 (RBAC) 是一種授權策略，僅提供使用者執行其任務所需的許可，而無其他功能。AWS KMS 支援 RBAC，可讓您透過在金鑰[政策中指定金鑰用量的精細許可來控制對金鑰](#)的存取。金鑰政策會指定資源、動作、效果、主體和選用條件，以授予對金鑰的存取權。

若要在 中實作 RBAC AWS KMS，我們建議區隔金鑰使用者和金鑰管理員的許可。

Key users

下列金鑰政策範例允許 ExampleUserRole IAM 角色使用 KMS 金鑰。

```
{
```

```

    "Sid": "Allow use of the key",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws: iam::111122223333:role/ExampleUserRole"
    },
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
    "Resource": "*"
}

```

您的金鑰使用者可能需要比本範例中的使用者更少的許可。僅指派使用者所需的許可。使用下列問題來協助您進一步精簡許可。

- 哪些 IAM 主體（角色或使用者）需要存取金鑰？
- 每個委託人需要使用 金鑰執行哪些動作？例如，委託人是否只需要加密和簽署許可？
- 使用者是人類還是 AWS 服務？如果是 AWS 服務，您可以使用 [條件金鑰](#) 將金鑰用量限制為特定 AWS 服務。

Key administrators

下列金鑰政策範例允許 IAM ExampleAdminRole 角色管理 KMS 金鑰。

```

{
    "Sid": "Allow access for Key Administrators",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws: iam::111122223333:role/ExampleAdminRole"
    },
    "Action": [
        "kms:Create*",
        "kms:Describe*",
        "kms:Enable*",
        "kms:List*",
        "kms:Put*",
        "kms:Update*",
        "kms:Revoke*",

```

```
        "kms:Disable*",
        "kms:Get*",
        "kms:Delete*",
        "kms:TagResource",
        "kms:UntagResource",
        "kms:ScheduleKeyDeletion",
        "kms:CancelKeyDeletion"
    ],
    "Resource": "*"
}
```

在此範例中，您的金鑰管理員可能需要比管理員更少的許可。僅指派金鑰管理員所需的許可。

僅提供使用者履行其角色所需的許可。使用者的許可可能會因在測試或生產環境中使用金鑰而有所不同。如果您在特定非生產環境中使用較不嚴格的許可，請實作程序來測試政策，然後再發佈至生產環境。

進一步了解

- [IAM 身分（使用者、使用者群組和角色）](#)
- [存取控制的類型](#)

允許其他帳戶中的使用者使用 KMS 金鑰

您可以允許不同 中的使用者或角色在您的帳戶中 AWS 帳戶 使用 KMS 金鑰。跨帳戶存取需要 KMS 金鑰的金鑰政策和外部使用者帳戶之 IAM 政策中的許可。

跨帳戶許可僅適用於下列操作：

- [密碼編譯操作](#)
- [CreateGrant](#)
- [DescribeKey](#)
- [GetKeyRotationStatus](#)
- [GetPublicKey](#)
- [ListGrants](#)
- [RetireGrant](#)
- [RevokeGrant](#)

如果您提供許可給不同帳戶中的使用者，讓他們能夠進行其他操作，則這些許可沒有任何作用。例如，如果您為不同帳戶中的委託人提供 IAM 政策中的 [kms:ListKeys](#) 許可，或金鑰政策中 KMS 金鑰的 [kms:ScheduleKeyDeletion](#) 許可，則使用者對資源呼叫這些操作的嘗試仍會失敗。

如需在不同帳戶中使用 KMS 金鑰進行 AWS KMS 操作的詳細資訊，請參閱 [AWS KMS 許可](#) 和 中的跨帳戶使用欄 [在其他帳戶中使用 KMS 金鑰](#)。還有 [AWS Key Management Service API 參考](#) 中每個 API 描述的跨帳戶使用章節。

Warning

請謹慎為委託人提供使用 KMS 金鑰的許可。只要有可能，請遵循最低權限原則。讓使用者只能存取其所需的 KMS 金鑰，以便只能存取所需的操作。

此外，請謹慎使用任何不熟悉的 KMS 金鑰，尤其是不同帳戶中的 KMS 金鑰。惡意使用者可能會授予您使用其 KMS 金鑰的許可，以取得關於您或您帳戶的資訊。

如需使用政策來保護帳戶中資源的詳細資訊，請參閱 [IAM 政策的最佳實務](#)。

若要提供 KMS 金鑰的使用許可給另一個帳戶中的使用者和角色，則需使用兩種不同類型的政策：

- KMS 金鑰的金鑰政策必須提供外部帳戶 (或外部帳戶中的使用者和角色) 使用 KMS 金鑰的許可。金鑰政策位在擁有 KMS 金鑰的帳戶中。
- 外部帳戶的 IAM 政策必須委派金鑰政策許可給其使用者和角色。這些政策會在外部帳戶中設定，並提供許可給該帳戶中的使用者和角色。

金鑰政策決定誰可以存取 KMS 金鑰。IAM 政策決定誰可以存取 KMS 金鑰。單獨的金鑰政策或 IAM 政策都不夠，您必須同時變更兩者。

若要編輯金鑰政策，您可以在 AWS 管理主控台 中使用 [政策檢視](#)，或使用 [CreateKey](#) 或 [PutKeyPolicy](#) 操作。

如需編輯 IAM 政策的說明，請參閱 [搭配使用 IAM 政策 AWS KMS](#)。

如需說明金鑰政策和 IAM 政策如何搭配運作，以允許在不同帳戶中使用 KMS 金鑰的範例，請參閱 [範例 2：使用者擔任的角色具有在不同 中使用 KMS 金鑰的許可 AWS 帳戶](#)。

您可以在 [AWS CloudTrail 日誌](#) 中的 KMS 金鑰上檢視產生的跨帳戶 AWS KMS 操作。對其他帳戶中使用 KMS 金鑰的操作會同時記入呼叫者帳戶和 KMS 金鑰擁有者帳戶。

主題

- [步驟 1：在本機帳戶中新增金鑰政策陳述式](#)
- [步驟 2：在外部帳戶中新增 IAM 政策](#)
- [允許使用外部 KMS 金鑰搭配 AWS 服務](#)
- [在其他帳戶中使用 KMS 金鑰](#)

Note

本主題中的範例旨在說明如何結合使用金鑰政策和 IAM 政策，來提供和限制存取 KMS 金鑰的權限。這些一般範例不適用於表示任何特定對 KMS 金鑰 AWS 服務所需的許可。如需 AWS 服務所需許可的相關資訊，請參閱服務文件中的加密主題。

步驟 1：在本機帳戶中新增金鑰政策陳述式

KMS 金鑰的金鑰政策是決定誰可以存取 KMS 金鑰及其可執行操作的主要因素。金鑰政策一律位在擁有 KMS 金鑰的帳戶中。與 IAM 政策不同，金鑰政策並不會指定資源。資源即是與該金鑰政策相關聯的 KMS 金鑰。當提供跨帳戶許可時，KMS 金鑰的金鑰政策必須提供外部帳戶 (或外部帳戶中的使用者和角色) 使用 KMS 金鑰的許可。

若要提供外部帳戶使用 KMS 金鑰的許可，請新增陳述式至指定外部帳戶的金鑰政策。在金鑰政策的 Principal 元素中，輸入外部帳戶的 Amazon Resource Name (ARN)。

當您在金鑰政策中指定外部帳戶時，該外部帳戶中的 IAM 管理員可以使用 IAM 政策來將這些許可委派給外部帳戶中的任何使用者和角色。他們也能決定使用者和角色可以執行金鑰政策中所指定的哪些動作。

只有在託管 KMS 金鑰及其金鑰政策的區域中啟用外部帳戶時，授予給外部帳戶及其委託人的許可才有效。如需相關資訊了解哪些區域預設為未啟用 (「選擇加入區域」)，請參閱《AWS 一般參考》[的管理 AWS 區域](#)。

例如，假設您想允許帳戶 444455556666 使用帳戶 111122223333 中的對稱加密 KMS 金鑰。若要執行此操作，請將政策陳述式新增至帳戶 111122223333 中 KMS 金鑰的金鑰政策，如以下範例所示。此政策陳述式會授予外部帳戶 444455556666 許可，在對稱加密 KMS 金鑰的密碼編譯操作中使用 KMS 金鑰。

 Note

下列範例顯示與其他帳戶共用 KMS 金鑰的金鑰政策範例。將範例 Sid、Principal 和 Action 值取代為 KMS 金鑰之預定用途的有效值。

```
{
  "Sid": "Allow an external account to use this KMS key",
  "Effect": "Allow",
  "Principal": {
    "AWS": [
      "arn:aws:iam::444455556666:root"
    ]
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:DescribeKey"
  ],
  "Resource": "*"
}
```

您可以在金鑰政策中指定特定的外部使用者和角色，而不用提供許可給外部帳戶。不過，除非外部帳戶中的 IAM 管理員將適當的 IAM 政策連接至各自身分，否則這些使用者和角色無法使用 KMS 金鑰。IAM 政策可以將許可提供給金鑰政策中指定的所有或部分外部使用者和角色。而且，其可允許金鑰政策中指定的全部動作或部分動作。

在金鑰政策中指定身分可以限制外部帳戶 IAM 管理員能提供的許可。然而，這會讓兩個帳戶的政策管理作業更加複雜。舉例來說，假設您需要新增使用者或角色。您必須將該身分新增至擁有 KMS 金鑰之帳戶中的金鑰政策，並在該身分的帳戶中建立 IAM 政策。

若要在金鑰政策中指定特定的外部使用者或角色，則請在 Principal 元素中輸入外部帳戶使用者或角色的 Amazon Resource Name (ARN)。

例如，以下金鑰政策陳述式範例允許帳戶 444455556666 中的 ExampleRole 使用帳戶 111122223333 中的 KMS 金鑰。此金鑰政策陳述式會授予外部帳戶 444455556666 許可，在對稱加密 KMS 金鑰的密碼編譯操作中使用 KMS 金鑰。

 Note

下列範例顯示與其他帳戶共用 KMS 金鑰的金鑰政策範例。將範例 Sid、Principal 和 Action 值取代為 KMS 金鑰之預定用途的有效值。

```
{
  "Sid": "Allow an external account to use this KMS key",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::444455556666:role/ExampleRole"
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:DescribeKey"
  ],
  "Resource": "*"
}
```

 Note

除非採用[條件](#)來限制金鑰政策，否則請勿在任何允許許可的金鑰政策陳述式將主體設為星號 (*)。除非另一個政策陳述式明確拒絕，否則星號會為每個使用 KMS 金鑰的 AWS 帳戶 許可提供每個身分。其他 中的使用者只要在自己的帳戶中擁有對應的許可，AWS 帳戶 就可以使用您的 KMS 金鑰。

您也需要決定要提供哪些許可給外部帳戶。例如，您可能想提供使用者能解密但無法加密的許可，或是能檢視但無法使用 KMS 金鑰的許可。如需 KMS 金鑰上的許可清單，請參閱 [AWS KMS 許可](#)。

建立 KMS 金鑰時設定金鑰政策

當您使用 [CreateKey](#) 操作建立 KMS 金鑰時，您可以使用其 `Policy` 參數來指定金鑰政策，以授予外部帳戶或外部使用者和角色使用 KMS 金鑰的許可。

當您在 中建立 KMS 金鑰時 AWS 管理主控台，您也可以建立其金鑰政策。當您在 Key Administrators (金鑰管理員) 和 Key Users (金鑰使用者) 區段中選取身分時，AWS KMS 會將這些

身分的政策陳述式新增至 KMS 金鑰的金鑰政策。Key Users (金鑰使用者) 區段也可讓您將外部帳戶做為金鑰使用者予以新增。

當您輸入外部帳戶的帳戶 ID 時，會將兩個陳述式 AWS KMS 新增至金鑰政策。此動作只會影響金鑰政策。在您連接 IAM 政策以提供部分或所有許可前，外部帳戶中的使用者和角色都無法使用 KMS 金鑰。

第一個金鑰政策陳述式會提供外部帳戶在密碼編譯操作中使用 KMS 金鑰的許可。第二個金鑰政策陳述式允許外部帳戶建立、檢視和撤銷 KMS 金鑰的授予，但僅限於請求來自[AWS 與整合的服務 AWS KMS](#)時。這些許可允許加密使用者資料的其他 AWS 服務使用 KMS 金鑰。這些許可專為加密服務中 AWS 使用者資料的 KMS 金鑰而設計。

步驟 2：在外部帳戶中新增 IAM 政策

擁有 KMS 金鑰之帳戶中的金鑰政策能夠設定許可的有效範圍。但是，在您連接能夠委派這些許可或使用授予來管理 KMS 金鑰存取權限的 IAM 政策前，外部帳戶中的使用者和角色都無法使用 KMS 金鑰。IAM 政策是在外部帳戶中設定。

如果金鑰政策是將許可提供給外部帳戶，您就能將 IAM 政策連接至該帳戶中的任何角色或使用者。但若金鑰政策是將許可提供給指定的使用者或角色，則 IAM 政策僅可提供這些許可給所有或部分指定的使用者和角色。如果 IAM 政策將 KMS 金鑰存取權限提供給其他外部使用者或角色，並不會起任何作用。

金鑰政策也能限制 IAM 政策中的動作。IAM 政策可以委派金鑰政策中指定的全部動作或部分動作。如果 IAM 政策列出金鑰政策中未指定的動作，則這些許可不會生效。

以下 IAM 政策範例允許委託人使用帳戶 111122223333 中的 KMS 金鑰來進行密碼編譯操作。若要提供此許可給帳戶 444455556666 中的使用者和角色，請[連接政策](#)至帳戶 444455556666 中的使用者或角色。

Note

下列範例顯示與其他帳戶共用 KMS 金鑰的 IAM 政策範例。將範例 Sid、Resource 和 Action 值取代為 KMS 金鑰之預定用途的有效值。

```
{
  "Sid": "AllowUseOfKeyInAccount111122223333",
  "Effect": "Allow",
```



```
"Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:DescribeKey"
],
"Resource": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
}
```

請注意有關此政策的下列詳細資訊：

- 與金鑰政策不同，IAM 政策陳述式不包含 Principal 元素。在 IAM 政策中，委託人即為該政策欲連接的身分。
- IAM 政策中的 Resource 元素會識別該委託人可以使用的 KMS 金鑰。若要指定 KMS 金鑰，請將其[金鑰 ARN](#)新增至 Resource 元素。
- 您可以在 Resource 元素中指定多個 KMS 金鑰。不過，如果您沒有在 Resource 元素中指定特定的 KMS 金鑰，則提供的 KMS 金鑰存取權限可能會不小心超過預期。
- 若要允許外部使用者搭配[與 AWS KMS 整合的 AWS 服務](#)使用 KMS 金鑰，您可能需要新增許可至金鑰政策或 IAM 政策。如需詳細資訊，請參閱[允許使用外部 KMS 金鑰搭配 AWS 服務](#)。

如需使用 IAM 政策的詳細資訊，請參閱 [IAM 政策](#)。

允許使用外部 KMS 金鑰搭配 AWS 服務

您可以提供許可給不同帳戶中的使用者，讓他們能夠搭配與 AWS KMS 整合的服務使用 KMS 金鑰。例如，外部帳戶中的使用者可以使用 KMS 金鑰來加密 Amazon S3 儲存貯體中的物件，或加密其存放的秘密 AWS Secrets Manager。

金鑰政策必須提供 KMS 金鑰使用許可給外部使用者或外部使用者的帳戶。此外，您還需要將 IAM 政策連接至能將 AWS 服務使用許可提供給使用者的身分。服務也可能會要求使用者在金鑰政策或 IAM 政策中擁有額外許可。如需 AWS 服務 需要的客戶受管金鑰許可清單，請參閱《使用者指南》的安全章節或 服務的開發人員指南》中的資料保護主題。

在其他帳戶中使用 KMS 金鑰

如果您有在不同 中使用 KMS 金鑰的許可 AWS 帳戶，則可以在 AWS CLI、AWS 管理主控台 AWS SDKs 和 中使用 KMS 金鑰 AWS Tools for PowerShell。

若要在 shell 命令或 API 請求中識別不同帳戶中的 KMS 金鑰，請使用下列[金鑰識別符](#)。

- 對於[密碼編譯操作](#)、[DescribeKey](#) 和 [GetPublicKey](#)，請使用 KMS 金鑰的[金鑰 ARN](#) 或[別名 ARN](#)。
- 對於 [CreateGrant](#)、[GetKeyRotationStatus](#)、[ListGrants](#) 和 [RevokeGrant](#)，請使用 KMS 金鑰的金鑰 ARN。

如果您只輸入金鑰 ID 或別名名稱，會 AWS 假設 KMS 金鑰位於您的帳戶中。

即使您具有使用 KMS 金鑰的許可，AWS KMS 主控台也不會在其他帳戶中顯示 KMS 金鑰。此外，在其他 AWS 服務的主控台中顯示的 KMS 金鑰清單不會在其他帳戶中包含 KMS 金鑰。

若要在 AWS 服務主控台的不同帳戶中指定 KMS 金鑰，您必須輸入 KMS 金鑰的金鑰 ARN 或別名 ARN。必要的金鑰識別符會隨服務而有所不同，服務主控台及其 API 操作可能會有所不同。如需詳細資訊，請參閱服務文件。

控制對多區域金鑰的存取

您可以在合規、災難復原和備份案例中使用多區域金鑰，而單一區域金鑰會比較複雜。不過，由於多區域金鑰的安全屬性與單一區域金鑰的安全屬性有很大的不同，所以建議在授權建立、管理和使用多區域金鑰時謹慎使用。

Note

在 Resource 欄位包含萬用字元的現有 IAM 政策陳述式，現在會同時套用至單一區域和多區域金鑰。若要將其限制為單一區域 KMS 金鑰或多區域金鑰，請使用 [kms:MultiRegion](#) 條件索引鍵。

使用您的授權工具來防止在任何單一區域足夠的情況下建立和使用多區域金鑰。允許主體僅將多區域金鑰複寫至需要它們 AWS 區域的。只將多區域金鑰的許可授予需要這些金鑰的委託人，並且僅為需要這些金鑰的任務授予。

您可以使用金鑰政策、IAM 政策和授權，允許 IAM 主體管理和使用您 中的多區域金鑰 AWS 帳戶。每個多區域金鑰都是具有唯一金鑰 ARN 和金鑰政策的獨立資源。您需要為每個金鑰建立和維護金鑰政策，並確保新的和現有的 IAM 政策會實作您的授權策略。

若要支援多區域金鑰，AWS KMS 會使用 IAM 服務連結角色。這個角色為 AWS KMS 提供其所需的許可，以同步[共用屬性](#)。如需詳細資訊，請參閱[授權 AWS KMS 同步多區域金鑰](#)。

主題

- [多區域金鑰的授權基礎知識](#)
- [授權多區域主要管理員和使用者](#)

多區域金鑰的授權基礎知識

針對多區域金鑰設計金鑰政策和 IAM 政策時，請考慮下列政策。

- 金鑰政策 – 每個多區域金鑰都是獨立的 KMS 金鑰資源，具有其自己的[金鑰政策](#)。您可以將相同或不同的金鑰政策套用至相關多區域金鑰集合中的每個金鑰。金鑰政策不是多區域金鑰的[共用屬性](#)。AWS KMS 不會在相關的多區域金鑰之間複製或同步金鑰政策。

當您在 AWS KMS 主控台中建立複本金鑰時，主控台會顯示主金鑰的目前金鑰政策，以方便使用。您可以使用此金鑰政策，加以編輯，或刪除並取代。但即使您接受主金鑰政策不變，AWS KMS 也不會同步政策。例如，如果您變更主要金鑰的金鑰政策，複本金鑰的金鑰政策會保持不變。

- 預設金鑰政策 – 當您使用 [CreateKey](#) 和 [ReplicateKey](#) 操作建立多區域金鑰時，會套用[預設金鑰政策](#)，除非您在請求中指定金鑰政策。這與套用至單一區域金鑰的預設金鑰政策相同。
- IAM 政策 – 與所有 KMS 金鑰一樣，您可以使用 IAM 政策來控制多區域金鑰的存取，僅當[金鑰政策允許](#)時。根據 AWS 區域預設，[IAM 政策](#)適用於所有。不過，您可以使用條件索引鍵，例如 [aws:RequestedRegion](#)，以限制特定區域的許可。

若要建立主要金鑰和複本金鑰，委託人必須在 IAM 政策中具有 `kms:CreateKey` 許可，該許可會套用到建立金鑰的區域。

- 授予 - AWS KMS [授予](#)是區域性的。每個授予都允許一個 KMS 金鑰的許可。您可以使用授予來允許多區域主要金鑰或複本金鑰的許可。但是，您無法使用單一授予來允許多個 KMS 金鑰的許可，即使這些金鑰是相關的多區域金鑰。
- 金鑰 ARN – 每個多區域金鑰都有[唯一金鑰 ARN](#)。相關多區域金鑰的金鑰 ARN 具有相同的分割區、帳戶和金鑰 ID，但區域不同。

若要將 IAM 政策陳述式套用至特定的多區域金鑰，請使用其金鑰 ARN 或包含區域的金鑰 ARN 模式。若要將 IAM 政策陳述式套用到所有相關的多區域金鑰，請在 ARN 的區域元素中使用萬用字元 (*)，如下列範例所示。

```
{
  "Effect": "Allow",
  "Action": [
    "kms:Describe*",
```

```
"kms:List*"
],
"Resource": {
  "arn:aws:kms:*::111122223333:key/mrk-1234abcd12ab34cd56ef1234567890ab"
}
}
```

若要將政策陳述式套用至 中的所有多區域金鑰 AWS 帳戶，您可以使用 [kms:MultiRegion](#) 政策條件或包含獨特mrk-字首的金鑰 ID 模式。

- 服務連結角色 – 建立多區域主要金鑰的委託人必須具有 [iam:CreateServiceLinkedRole](#) 許可。

若要同步相關多區域金鑰的共用屬性，AWS KMS 會擔任 IAM [服務連結角色](#)。每當您 AWS KMS 建立多區域主金鑰時，會在 AWS 帳戶中建立服務連結角色。(如果角色存在，AWS KMS 會重新建立角色，這沒有任何有害的影響。) 角色在所有區域都有效。若要允許 AWS KMS 建立（或重新建立）服務連結角色，建立多區域主金鑰的主體必須具有 [iam : CreateServiceLinkedRole](#) 許可。

授權多區域主要管理員和使用者

建立和管理多區域金鑰的委託人需要在主要區域和複本區域中的下列許可：

- kms:CreateKey
- kms:ReplicateKey
- kms:UpdatePrimaryRegion
- iam:CreateServiceLinkedRole

建立主要金鑰

若要[建立多區域主要金鑰](#)，委託人需要 IAM 政策中的 [kms:CreateKey](#) 和 [iam:CreateServiceLinkedRole](#) 許可，IAM 政策在主要金鑰的區域中有效。擁有這些許可的委託人可以建立單一區域和多區域金鑰，除非您限制其許可。

[iam:CreateServiceLinkedRole](#) 許可允許 AWS KMS 建立 [AWSServiceRoleForKeyManagementServiceMultiRegionKeys](#) 角色，以同步相關多區域金鑰的[共用屬性](#)。

例如，此 IAM 政策允許主體建立多區域金鑰、連接這些金鑰的政策，以及多區域金鑰的服務連結角色。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Action": [
      "kms:CreateKey",
      "iam:CreateServiceLinkedRole"
    ],
    "Effect": "Allow",
    "Resource": "*"
  }
}
```

若要允許或拒絕建立多區域主要金鑰的許可，請使用 [kms:MultiRegion](#) 條件索引鍵。有效值為 true (多區域金鑰) 或 false (單一區域金鑰)。例如，下列 IAM 政策陳述式會使用具有 kms:MultiRegion 條件索引鍵的 Deny 動作，以防止委託人建立多區域金鑰。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Action": "kms:CreateKey",
    "Effect": "Deny",
    "Resource": "*",
    "Condition": {
      "Bool": {
        "kms:MultiRegion": true
      }
    }
  }
}
```

複寫金鑰

若要[建立多區域複本金鑰](#)，則委託人需要下列許可：

- 主要金鑰之金鑰政策中的 [kms:ReplicateKey](#) 許可。

- 在複本金鑰區域中有效之 IAM 政策中的 [kms:CreateKey](#) 許可。

允許這些許可時請小心謹慎。其允許委託人建立 KMS 金鑰，以及授權其使用的金鑰政策。kms:ReplicateKey 許可還授權在 AWS KMS 範圍內跨區域邊界傳輸金鑰材料。

若要限制可複寫多區域金鑰 AWS 區域的，請使用 [kms:ReplicaRegion](#) 條件金鑰。它只限制 kms:ReplicateKey 許可。否則，它沒有影響。例如，下列金鑰政策允許委託人複寫該主要金鑰，但僅限於指定區域中。

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/Administrator"
  },
  "Action": "kms:ReplicateKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ReplicaRegion": [
        "us-east-1",
        "eu-west-3",
        "ap-southeast-2"
      ]
    }
  }
}
```

更新主要區域

授權的委託人可以將複本金鑰轉換為主要金鑰，這會將先前的主要金鑰變更為複本。這個動作稱為[更新主要區域](#)。要更新主要區域，委託人需要兩個區域的 [kms:UpdatePrimaryRegion](#) 許可。您可以在金鑰政策或 IAM 政策中提供這些許可。

- 主要金鑰上的 kms:UpdatePrimaryRegion。此許可必須在主要金鑰區域有效。
- 複本金鑰上的 kms:UpdatePrimaryRegion。此許可必須在複本金鑰區域中生效。

例如，下列金鑰政策會為可以擔任管理員角色的使用者提供許可，以更新 KMS 金鑰的主要區域。此 KMS 金鑰可以是此操作中的主要金鑰或複本金鑰。

```
{
```

```

"Effect": "Allow",
"Resource": "*",
"Principal": {
  "AWS": "arn:aws:iam::111122223333:role/Administrator"
},
"Action": "kms:UpdatePrimaryRegion"
}

```

若要限制可託管主索引鍵 AWS 區域的，請使用 [kms:PrimaryRegion](#) 條件索引鍵。例如，下列 IAM 政策陳述式允許主體更新 中多區域金鑰的主要區域 AWS 帳戶，但只有在新的主要區域是其中一個指定的區域時。

```

{
  "Effect": "Allow",
  "Action": "kms:UpdatePrimaryRegion",
  "Resource": {
    "arn:aws:kms*:111122223333:key/*"
  },
  "Condition": {
    "StringEquals": {
      "kms:PrimaryRegion": [
        "us-west-2",
        "sa-east-1",
        "ap-southeast-1"
      ]
    }
  }
}

```

使用和管理多區域金鑰

根據預設，在 AWS 帳戶 和區域中擁有使用和管理 KMS 金鑰之許可的委託人也具有使用和管理多區域金鑰的許可。不過，您可以使用 [kms:MultiRegion](#) 條件索引鍵，只允許單一區域索引鍵或僅允許多區域索引鍵。或者使用 [kms:MultiRegionKeyType](#) 條件索引鍵，只允許多區域主要金鑰或只允許複本金鑰。兩個條件索引鍵都會控制對 [CreateKey](#) 操作和使用現有 KMS 金鑰之任何操作 (例如，[Encrypt](#) 或 [EnableKey](#)) 的存取。

下列範例 IAM 政策陳述式使用 `kms:MultiRegion` 條件索引鍵，以防止委託人使用或管理任何多區域金鑰。

```

{
  "Effect": "Deny",

```

```
"Action": "kms:*",
"Resource": "*",
"Condition": {
  "Bool": "kms:MultiRegion": true
}
}
```

此範例 IAM 政策陳述式使用 `kms:MultiRegionKeyType` 條件，以允許委託人排程和取消金鑰刪除，但僅限於多區域複本金鑰。

```
{
  "Effect": "Allow",
  "Action": [
    "kms:ScheduleKeyDeletion",
    "kms:CancelKeyDeletion"
  ],
  "Resource": {
    "arn:aws:kms:us-west-2:111122223333:key/*"
  },
  "Condition": {
    "StringEquals": "kms:MultiRegionKeyType": "REPLICA"
  }
}
```

判斷對 的存取 AWS KMS keys

若要判斷目前可存取的人員或內容的完整範圍 AWS KMS key，您必須檢查 KMS 金鑰的金鑰政策、適用於 KMS 金鑰的所有[授予](#)，以及可能的所有 AWS Identity and Access Management (IAM) 政策。您可以這樣做來判斷 KMS 金鑰的潛在使用範圍，可協助您符合規範或稽核要求。您可以善用下列主題，藉此產生目前有權存取 KMS 金鑰的 AWS 主體 (身分) 完整清單。

主題

- [檢查金鑰政策](#)
- [檢查 IAM 政策](#)
- [檢查授與](#)

檢查金鑰政策

[金鑰政策](#)是控制對 KMS 金鑰之存取的主要方式。每個 KMS 金鑰只有一個金鑰政策。

當金鑰政策包含[預設金鑰政策](#)時，金鑰政策可讓帳戶中的 IAM 管理員使用 IAM 政策控制 KMS 金鑰的存取。此外，如果金鑰政策授權[另一個 AWS 帳戶](#)使用 KMS 金鑰，則外部帳戶的 IAM 管理員可以使用 IAM 政策委派這些許可。請[檢查 IAM 政策](#)，以判斷可存取 KMS 金鑰之主體的完整清單。

若要檢視 AWS KMS [客戶受管金鑰](#)或[AWS 受管金鑰](#)帳戶中的金鑰政策，請使用 AWS KMS API 中的 AWS 管理主控台 或 [GetKeyPolicy](#) 操作。若要檢視金鑰政策，您必須具有 KMS 金鑰的 `kms:GetKeyPolicy` 許可。如需檢視 KMS 金鑰之金鑰政策的說明，請參閱 [the section called “檢視金鑰政策”](#)。

檢查金鑰政策文件並記下在各政策陳述式 Principal 元素中指定的所有主體。在具有 Allow 效果的政策陳述 AWS 帳戶 式中，IAM 使用者、IAM 角色和 Principal 元素中的 可以存取此 KMS 金鑰。

Note

除非採用[條件](#)來限制金鑰政策，否則請勿在任何允許許可的金鑰政策陳述式將主體設為星號 (*)。除非另一個政策陳述式明確拒絕，否則星號會提供每個 AWS 帳戶 許可中的每個身分使用 KMS 金鑰。其他 中的使用者只要在自己的帳戶中擁有對應的許可，AWS 帳戶 就可以使用您的 KMS 金鑰。

以下範例使用[預設金鑰政策](#)中的政策陳述式來示範如何執行此操作。

Example 政策陳述式 1

```
{
  "Sid": "Enable IAM User Permissions",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
  "Action": "kms:*",
  "Resource": "*"
}
```

在政策陳述式 1 中，`arn:aws:iam::111122223333:root` 是參考 AWS 帳戶 111122223333 [AWS 的帳戶委託人](#)。(其不是帳戶根使用者。) 根據預設，當您使用 建立新的 KMS 金鑰，或以程式設計方式建立新的 KMS 金鑰，但不提供金鑰政策時 AWS 管理主控台，金鑰政策文件中會包含像這樣的政策陳述式。

金鑰政策文件具有允許存取的陳述式，AWS 帳戶 可讓[帳戶中的 IAM 政策允許存取 KMS 金鑰](#)。這表示帳戶中的使用者和角色可能可以存取 KMS 金鑰，即使他們未在金鑰政策文件中被明確列為主體。請注意檢查 AWS 帳戶 所有列為主體的 中的所有 [IAM 政策](#)，以判斷它們是否允許存取此 KMS 金鑰。

Example 政策陳述式 2

```
{
  "Sid": "Allow access for Key Administrators",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/KMSKeyAdmins"},
  "Action": [
    "kms:Describe*",
    "kms:Put*",
    "kms:Create*",
    "kms:Update*",
    "kms:Enable*",
    "kms:Revoke*",
    "kms:List*",
    "kms:Disable*",
    "kms:Get*",
    "kms>Delete*",
    "kms:ScheduleKeyDeletion",
    "kms:CancelKeyDeletion"
  ],
  "Resource": "*"
}
```

在政策陳述式 2 中，arn:aws:iam::111122223333:role/KMSKeyAdmins 是指名為 KMSKeyAdmins 在 AWS 帳戶 111122223333 的 IAM 角色。被授權擔任此角色的使用者可以執行政策陳述式中所列的動作，這些是用於管理 KMS 金鑰的管理動作。

Example 政策陳述式 3

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/EncryptionApp"},
  "Action": [
    "kms:DescribeKey",
    "kms:GenerateDataKey*",
    "kms:Encrypt",
    "kms:ReEncrypt*",
    "kms:Decrypt"
  ],
  "Resource": "*"
}
```

在政策陳述式 3 中，`arn:aws:iam::111122223333:role/EncryptionApp`是指名為 EncryptionApp in AWS 帳戶 111122223333 的 IAM 角色。被授權擔任此角色的主體可以執行政策陳述式中所列的動作，包括對稱加密 KMS 金鑰的[密碼編譯操作](#)。

Example 政策陳述式 4

```
{
  "Sid": "Allow attachment of persistent resources",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/EncryptionApp"},
  "Action": [
    "kms:ListGrants",
    "kms:CreateGrant",
    "kms:RevokeGrant"
  ],
  "Resource": "*",
  "Condition": {"Bool": {"kms:GrantIsForAWSResource": true}}
}
```

在政策陳述式 4 中，`arn:aws:iam::111122223333:role/EncryptionApp`是指名為 EncryptionApp in AWS 帳戶 111122223333 的 IAM 角色。被授權擔任此角色的主體能夠執行政策陳述式中所列的動作。當這些動作結合範例政策陳述式 3 中允許的動作時，是委派 KMS 金鑰的使用給大部分與 [AWS KMS 整合的 AWS 服務](#) (尤其是使用[授予](#)的服務) 的必要動作。Condition 元素中的 [kms:GrantIsForAWSResource](#) 值可確保僅當委派代表是與整合 AWS KMS 並使用授權的 AWS 服務時，才允許委派。

若要了解在金鑰政策文件中指定主體的所有不同方法，請參閱《[IAM 使用者指南](#)》中的 Specifying a Principal (指定主體)。

若要進一步了解 AWS KMS 金鑰政策，請參閱 [中的金鑰政策 AWS KMS](#)。

檢查 IAM 政策

除了金鑰政策和授予之外，您也可以使用 [IAM 政策](#)，以允許存取 KMS 金鑰。如需金鑰政策與 IAM 政策如何同時運作的相關資訊，請參閱 [故障診斷 AWS KMS 許可](#)。

若要判斷哪些主體目前可透過 IAM 政策存取 KMS 金鑰，您可以使用以瀏覽器為基礎的 [IAM 政策模擬器](#) 工具，或者可以提出請求至 IAM API。

檢查 IAM 政策的方式

- [使用 IAM 政策模擬器來檢查 IAM 政策](#)

- [使用 IAM API 檢查 IAM 政策](#)

使用 IAM 政策模擬器來檢查 IAM 政策

IAM 政策模擬器可協助您了解哪些原則有權限透過 IAM 政策來存取 KMS 金鑰。

若要使用 IAM 政策模擬器來判斷是否可存取 KMS 金鑰

1. 登入 [AWS 管理主控台](#) 然後在 開啟 IAM 政策模擬器<https://policysim.aws.amazon.com/>。
2. 在使用者、群組和角色窗格，選擇您想要模擬所屬政策的使用者、群組或角色。
3. (選用) 清除任何您想要從模擬中刪除的政策旁的核取方塊。若要模擬所有政策，請保持勾選所有政策。
4. 在 Policy Simulator (政策模擬器) 窗格中，執行下列動作：
 - a. 在 Select service (選取服務) 中，選擇 Key Management Service (金鑰管理服務)。
 - b. 若要模擬特定 AWS KMS 動作，請在選取動作中選擇要模擬的動作。若要模擬所有 AWS KMS 動作，請選擇全選。
5. (選用) 政策模擬器根據預設將模擬所有對 KMS 金鑰的存取。若要模擬存取特定 KMS 金鑰，請選擇 Simulation Settings (模擬設定)，接著輸入要模擬之 KMS 金鑰的 Amazon Resource Name (ARN)。
6. 選擇 Run Simulation (執行模擬)。

您可在 Results (結果) 部分查看模擬結果。對於 AWS 帳戶中的每個 IAM 使用者、群組和角色，請重複步驟 2 到步驟 6。

使用 IAM API 檢查 IAM 政策

您可以使用 IAM API 來以程式設計方式檢查 IAM 政策。以下步驟提供執行此操作方法的一般概述：

1. 對於金鑰政策中 AWS 帳戶 列為主體的每個 (即此格式指定的每個[AWS 帳戶主體](#)："Principal": {"AWS": "arn:aws:iam::111122223333:root"}), 請使用 IAM API 中的 [ListUsers](#) 和 [ListRoles](#) 操作來取得帳戶中的所有使用者和角色。
2. 對於清單中的每個使用者和角色，請使用 IAM API 中的 [SimulatePrincipalPolicy](#) 操作，以下列參數傳遞：
 - 針對 PolicySourceArn，請指定清單中使用者或角色的 Amazon Resource Name (ARN)。您只能為每個 SimulatePrincipalPolicy 請求指定一個 PolicySourceArn，因此您必須多次呼叫此操作，針對清單中的每個使用者與角色各呼叫一次。

- 針對 ActionNames 清單，指定要模擬的每個 AWS KMS API 動作。若要模擬所有 AWS KMS API 動作，請使用 kms:*。若要測試個別 AWS KMS API 動作，請在每個 API 動作前面加上 "kms:"，例如 "kms:ListKeys"。如需 AWS KMS API 動作的完整清單，請參閱 AWS Key Management Service API 參考中的 [動作](#)。
- (選用) 若要判斷使用者或角色是否可以存取特定 KMS 金鑰，請使用 ResourceArns 參數來指定 KMS 金鑰的 Amazon Resource Name (ARN) 清單。若要判斷使用者或角色是否可以存取任何 KMS 金鑰，請忽略 ResourceArns 參數。

IAM 將以下列評估決定來回應每個 SimulatePrincipalPolicy 請求：allowed、explicitDeny 或 implicitDeny。對於包含評估決策的每個回應 allowed，回應會包含允許的特定 AWS KMS API 操作的名稱。還會包含評估時所使用的 KMS 金鑰的 ARN (如果有)。

檢查授與

授予是進階機制，用於指定您或與整合的 AWS 服務 AWS KMS 可以使用的許可，以指定 KMS 金鑰的使用方式和時間。授予會連接到 KMS 金鑰，而且每個授予包含獲得使用 KMS 金鑰之許可的主體以及允許的操作清單。授與是金鑰政策的替代選項，適用於特定的使用案例。如需詳細資訊，請參閱 [在中授予 AWS KMS](#)。

若要取得 KMS 金鑰的授予清單，請使用 AWS KMS [ListGrants](#) 操作。您可以檢查 KMS 金鑰的授予，以判斷何人或何物目前具有透過這些授予使用 KMS 金鑰的存取權。例如，以下是從 AWS CLI 中 [list-grants](#) 命令取得的 JSON 授予表示。

```
{"Grants": [{
  "Operations": ["Decrypt"],
  "KeyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "Name": "0d8aa621-43ef-4657-b29c-3752c41dc132",
  "RetiringPrincipal": "arn:aws:iam::123456789012:root",
  "GranteePrincipal": "arn:aws:sts::111122223333:assumed-role/aws:ec2-infrastructure/i-5d476fab",
  "GrantId": "dc716f53c93acacf291b1540de3e5a232b76256c83b2ecb22cdefa26576a2d3e",
  "IssuingAccount": "arn:aws:iam::111122223333:root",
  "CreationDate": 1.444151834E9,
  "Constraints": {"EncryptionContextSubset": {"aws:ebs:id": "vol-5cccfb4e"}}
}]}
```

若要了解何人或何物具有使用 KMS 金鑰的存取權，請查看 "GranteePrincipal" 元素。在上述範例中，承授者主體是與 EC2 執行個體 i-5d476fab 相關聯的擔任角色使用者。EC2 基礎設施使用此角

色，將加密的 EBS 磁碟區 vol-5cccfb4e 連接到執行個體。在這種情況下，EC2 基礎設施角色有許可使用 KMS 金鑰，因為您之前建立受此 KMS 金鑰保護的加密 EBS 磁碟區。然後，您將此磁碟區連接到 EC2 執行個體。

以下是從 AWS CLI 中 [list-grants](#) 命令取得的另一個範例 JSON 授予表示。在下列範例中，承授者委託人是另一個 AWS 帳戶。

```
{"Grants": [{
  "Operations": ["Encrypt"],
  "KeyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "Name": "",
  "GranteePrincipal": "arn:aws:iam::444455556666:root",
  "GrantId": "f271e8328717f8bde5d03f4981f06a6b3fc18bcae2da12ac38bd9186e7925d11",
  "IssuingAccount": "arn:aws:iam::111122223333:root",
  "CreationDate": 1.444151269E9
}]}
```

加密內容

Note

您不能使用 [非對稱 KMS 金鑰](#) 或 [HMAC KMS 金鑰](#) 在密碼編譯操作中指定加密內容。非對稱演算法和 MAC 演算法不支援加密內容。

具有 [對稱加密 KMS 金鑰](#) 的全 AWS KMS [加密操作](#) 接受加密內容，這是一組選用的非私密金鑰/值對，可包含有關資料的其他內容資訊。您可以在 `Encrypt` 操作中插入加密內容 AWS KMS，以增強 AWS KMS API 解密呼叫的授權和可稽核性。AWS KMS 會使用加密內容做為額外的驗證資料 (AAD)，以支援驗證加密。加密內容以密碼編譯方式繫結至加密文字，因此需要相同的加密內容才能解密資料。

加密內容不是秘密，也不被加密。它會以純文字出現在 [AWS CloudTrail Logs](#) 中，因此您可以使用它來識別和分類密碼編譯操作。您的加密內容不應包含敏感資訊。建議在您的加密內容中描述要加密或解密的資料。例如，當您加密檔案時，您可以使用檔案路徑的一部分作為加密內容。

```
"encryptionContext": {
  "department": "10103.0"
}
```

例如，加密使用 [Amazon Elastic Block Store](#) (Amazon EBS) [CreateSnapshot](#) 操作建立的磁碟區和快照時，Amazon EBS 會使用磁碟區 ID 作為加密內容值。

```
"encryptionContext": {  
  "aws:eks:id": "vol-abcde12345abc1234"  
}
```

您也可以使用加密內容來精簡或限制您帳戶中對 AWS KMS keys 的存取。您可以使用加密內容[做為授與中的限制](#)，以及做為[政策陳述式中的條件](#)。加密內容金鑰及其值可以是使用的任意字串aws。這些值會對比[AWS 產生的標籤](#)，例如 [aws : cloudformation : stack-name](#)。如需詳細資訊，請參閱[kms:EncryptionContext:context-key](#)

若要了解如何使用加密內容來保護加密資料的完整性，請參閱 AWS 安全部落格中的文章[如何使用 AWS Key Management Service 和 EncryptionContext 來保護加密資料的完整性](#)。

加密內容規則

AWS KMS 針對加密內容金鑰和值強制執行下列規則。

- 加密內容對中的金鑰和值必須是簡單的常值字串。如果您使用不同的類型，例如整數或浮點數，AWS KMS 會將它解譯為字串。
- 加密內容中的金鑰和值可以包含 Unicode 字元。如果加密內容包含金鑰政策或 IAM 政策中不允許的字元，則您將無法在政策條件金鑰中指定加密內容，例如 [kms:EncryptionContext:context-key](#) 和 [kms:EncryptionContextKeys](#)。如需有關金鑰政策文件規則的詳細資訊，請參閱[金鑰政策格式](#)。如需有關 IAM 政策文件規則的詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 名稱需求](#)。

政策中的加密內容

加密內容主要用於驗證完整性和真確性。但是，您也可以在金鑰政策和 IAM 政策中，使用加密內容來控制對對稱加密 AWS KMS keys 的存取。

[kms:EncryptionContext](#) 和 [kms:EncryptionContextKeys](#) 條件索引鍵只有在請求包含特定加密內容索引鍵或鍵值對時，才會允許 (或拒絕) 許可。

例如，下列金鑰政策陳述式允許 RoleForExampleApp 角色在 Decrypt 操作中使用 KMS 金鑰。它使用 [kms:EncryptionContext:context-key](#) 條件金鑰，只在請求中的加密內容包含 AppName:ExampleApp 加密內容對時，才允許此許可。

```
{
```



```
"Effect": "Allow",
"Principal": {
  "AWS": "arn:aws:iam::111122223333:role/RoleForExampleApp"
},
"Action": "kms:Decrypt",
"Resource": "*",
"Condition": {
  "StringEquals": {
    "kms:EncryptionContext:AppName": "ExampleApp"
  }
}
```

如需這些加密內容條件金鑰的詳細資訊，請參閱 [的條件索引鍵 AWS KMS](#)。

授與中的加密內容

當您[建立授予](#)時，您可以包含授予[限制](#)條件，以建立授予許可的條件。AWS KMS 支援兩個授予限制，`EncryptionContextEquals`以及 `EncryptionContextSubset`，這兩者都涉及密碼編譯操作請求中的[加密內容](#)。當您使用這些授予限制條件時，只有在密碼編譯操作請求中的加密內容滿足授予限制條件的需求時，授予中的許可才會生效。

例如，您可以將 `EncryptionContextEquals` 授予限制條件新增至允許 [GenerateDataKey](#) 操作的授予。藉由此限制條件，只有在請求中的加密內容與授予限制條件中的加密內容大小寫全部相符時，授予才會允許操作。

```
$ aws kms create-grant \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --grantee-principal arn:aws:iam::111122223333:user/exampleUser \
  --retiring-principal arn:aws:iam::111122223333:role/adminRole \
  --operations GenerateDataKey \
  --constraints EncryptionContextEquals={Purpose=Test}
```

來自承授者委託人的類似下列請求應當滿足 `EncryptionContextEquals` 限制條件。

```
$ aws kms generate-data-key \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --key-spec AES_256 \
  --encryption-context Purpose=Test
```


如需授予限制條件的詳細資訊，請參閱 [使用授予限制條件](#)。如需授予的詳細資訊，請參閱 [the section called “授權”](#)。

記錄加密內容

AWS KMS 使用 AWS CloudTrail 記錄加密內容，讓您可以判斷已存取哪些 KMS 金鑰和資料。日誌項目會準確顯示哪個 KMS 金鑰用來加密或解密由日誌項目中加密內容參考的特定資料。

Important

由於加密內容會被記錄，因此不能包含敏感資訊。

儲存加密內容

如要簡化在您呼叫 [Decrypt](#) 或 [ReEncrypt](#) 操作時的任何加密內容使用，您可以將加密內容與加密的資料存放在一起。我們建議您只存放足夠的加密內容，以協助您在需要用它來加密或解密時建立完整的加密內容。

例如，如果加密內容是完整路徑檔案，只需將部分路徑與加密的檔案內容一起存放。接著，當您需要完整的加密內容，請從存放的片段中重新建構。如果有人擅自竄改檔案，例如重新命名或移到不同的位置，加密內容值會變更，解密請求會失敗。

測試您的許可

若要使用 AWS KMS，您必須擁有 AWS 可用於驗證 API 請求的登入資料。憑證必須包含存取 KMS 金鑰與別名的許可。許可由金鑰政策、IAM 政策、授權以及跨帳戶存取控制決定。除控制 KMS 金鑰的存取權之外，您還可控制 CloudHSM 與自訂金鑰存放區的存取權。

您可指定 DryRun API 參數，驗證您擁有運用 AWS KMS 金鑰的必要許可。您也可以使用 DryRun 來驗證 AWS KMS API 呼叫中的請求參數是否已正確指定。

主題

- [什麼是 DryRun 參數？](#)
- [利用 API 指定 DryRun](#)

什麼是 DryRun 參數？

DryRun 是選用的 API 參數，您可加以指定以便驗證 AWS KMS API 呼叫是否成功。在實際呼叫 AWS KMS 之前，利用 DryRun 來測試 API 呼叫。您可以驗證下列各項。

- 您擁有必要的許可，可運用 AWS KMS 金鑰。
- 您已正確指定呼叫的參數。

AWS KMS 支援在特定 API 動作中使用 DryRun 參數：

- [CreateGrant](#)
- [解密](#)
- [DeriveSharedSecret](#)
- [加密](#)
- [GenerateDataKey](#)
- [GenerateDataKeyPair](#)
- [GenerateDataKeyPairWithoutPlaintext](#)
- [GenerateDataKeyWithoutPlaintext](#)
- [GenerateMac](#)
- [ReEncrypt](#)
- [RetireGrant](#)
- [RevokeGrant](#)
- [符號](#)
- [確認](#)
- [VerifyMac](#)

採用 DryRun 參數將產生費用，並作為標準 API 請求計費。如需 AWS KMS 定價的詳細資訊，請參閱 [AWS Key Management Service 定價](#)。

採用 DryRun 參數的所有 API 請求都會套用至 API 的請求配額，如您超出 API 請求配額，則可能導致限流例外狀況。例如，無論採用 DryRun 還是不採用 DryRun 來呼叫 [Decrypt](#)，都會根據相同的密碼編譯操作配額進行計數。如需進一步了解，請參閱《[調節 AWS KMS 請求](#)》。

每個對 AWS KMS API 操作的呼叫都會擷取為事件，並記錄在 AWS CloudTrail 日誌中。任何指定 DryRun 參數的操作輸出均會顯示在 CloudTrail 日誌。如需詳細資訊，請參閱[使用 記錄 AWS KMS API 呼叫 AWS CloudTrail](#)。

利用 API 指定 DryRun

若要使用 DryRun，請在支援 `--dry-run` 參數的命令和 AWS KMS API 呼叫中 AWS CLI 指定參數。當您這麼做時，AWS KMS 會驗證您的呼叫是否成功。使用的 AWS KMS 呼叫 DryRun 一律會失敗，並傳回訊息，其中包含呼叫失敗原因的相關資訊。訊息可能包含下列例外狀況：

- `DryRunOperationException` - 如未指定 DryRun，請求就會成功。
- `ValidationException` - 請求因指定不正確 API 參數而失敗。
- `AccessDeniedException` - 您不具權限，無法對 KMS 資源執行指定 API 動作。

例如，下列命令採用 [CreateGrant](#) 操作並建立授權，以便允許獲授權的使用者擔任 `keyUserRole` 角色，並對指定的[對稱 KMS 金鑰](#)呼叫 [Decrypt](#) 操作。指定 DryRun 參數。

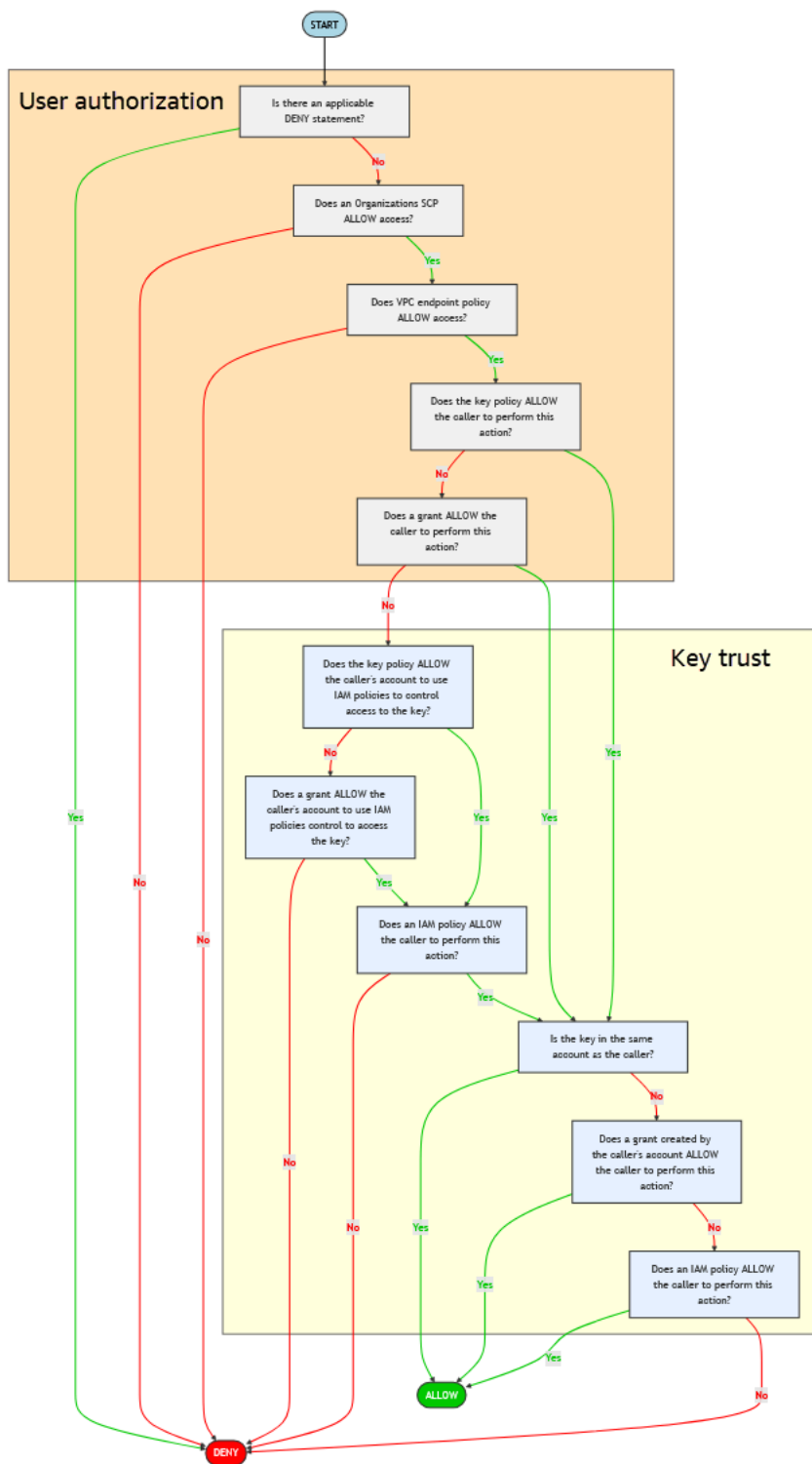
```
$ aws kms create-grant \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --grantee-principal arn:aws:iam::111122223333:role/keyUserRole \  
  --operations Decrypt \  
  --dry-run
```

故障診斷 AWS KMS 許可

授權存取 KMS 金鑰時，會 AWS KMS 評估下列項目：

- 連接到 KMS 金鑰的[金鑰政策](#)。金鑰政策一律在擁有 KMS 金鑰的 AWS 帳戶和區域中定義。
- 連接到提出請求的使用者或角色的所有 [IAM 政策](#)。管理主體使用 KMS 金鑰的 IAM 政策一律在主體的 AWS 帳戶中定義。
- 適用於 KMS 金鑰的所有[授予](#)。
- 其他可能套用至使用 KMS 金鑰之請求的政策類型，例如 [AWS Organizations 服務控制政策](#)和 [VPC 端點政策](#)。這些政策是選用的，依預設允許所有動作，但您可以使用其來限制許可，否則會將許可授予主體。

AWS KMS 會一起評估這些政策機制，以判斷是否允許或拒絕存取 KMS 金鑰。若要這樣做，AWS KMS 會使用類似下列流程圖中描述的程序。以下流程圖提供政策評估程序的視覺化呈現。



此流程圖分為兩個部分。這兩個部分應有其先後順序，但評估通常會同時進行。

- 使用授權可根據其金鑰政策、IAM 政策、授予和其他適用政策決定是否允許您使用 KMS 金鑰。

- 金鑰信任會決定您是否應信任您被允許使用的 KMS 金鑰。一般而言，您會信任 中的資源 AWS 帳戶。但是，AWS 帳戶 如果您帳戶中的授予或 IAM 政策允許您使用 KMS 金鑰，您也可以放心在不同的 中使用 KMS 金鑰。

您可以使用此流程圖來探索呼叫者被允許或拒絕使用 KMS 金鑰許可的原因。您也可以使用它來評估政策和授予。例如，流程圖會顯示呼叫者的存取被拒絕可能是由於明確 DENY 陳述式或金鑰政策、IAM 政策或授予中缺少明確 ALLOW 陳述式。

流程圖可以說明一些常見許可案例。

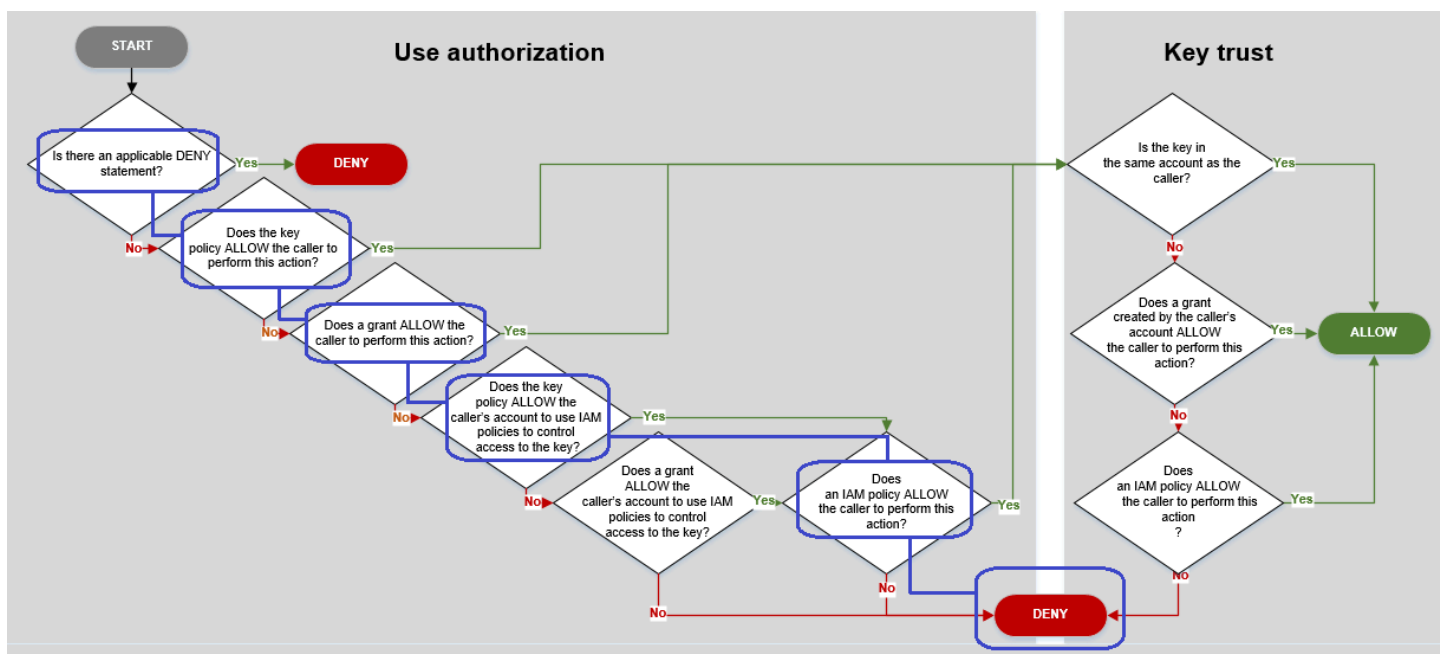
許可範例

- [範例 1：拒絕使用者存取其 中的 KMS 金鑰 AWS 帳戶](#)
- [範例 2：使用者擔任的角色具有在不同 中使用 KMS 金鑰的許可 AWS 帳戶](#)

範例 1：拒絕使用者存取其 中的 KMS 金鑰 AWS 帳戶

Alice 是 111122223333 AWS 帳戶中的 IAM 使用者。她被拒絕存取同一個 AWS 帳戶中的 KMS 金鑰。為什麼 Alice 無法使用 KMS 金鑰？

在此案例中，Alice 對 KMS 金鑰的存取遭拒，因為沒有賦予她必要許可的金鑰政策、IAM 政策或授予。KMS 金鑰的金鑰政策允許 AWS 帳戶 使用 IAM 政策來控制對 KMS 金鑰的存取，但沒有 IAM 政策會授予 Alice 使用 KMS 金鑰的許可。



請考量此範例的相關政策。

- Alice 想要使用的 KMS 金鑰具有[預設金鑰政策](#)。本政策[允許擁有 KMS 金鑰的 AWS 帳戶](#)，以使用 IAM 政策控制對 KMS 金鑰的存取。此金鑰政策符合流程圖中的金鑰政策是否允許呼叫者帳戶使用 IAM 政策來控制對金鑰的存取？條件。

JSON

```
{
  "Version": "2012-10-17",
  "Id": "key-test-1",
  "Statement": [ {
    "Sid": "DelegateToIAMPolicies",
    "Effect": "Allow",
    "Principal": {
      "AWS": "arn:aws:iam::111122223333:root"
    },
    "Action": "kms:*",
    "Resource": "*"
  } ]
}
```

- 不過，沒有可賦予 Alice 使用 KMS 金鑰之許可的金鑰政策、IAM 政策或授予。因此，Alice 使用 KMS 金鑰的許可遭拒。

範例 2：使用者擔任的角色具有在不同 中使用 KMS 金鑰的許可 AWS 帳戶

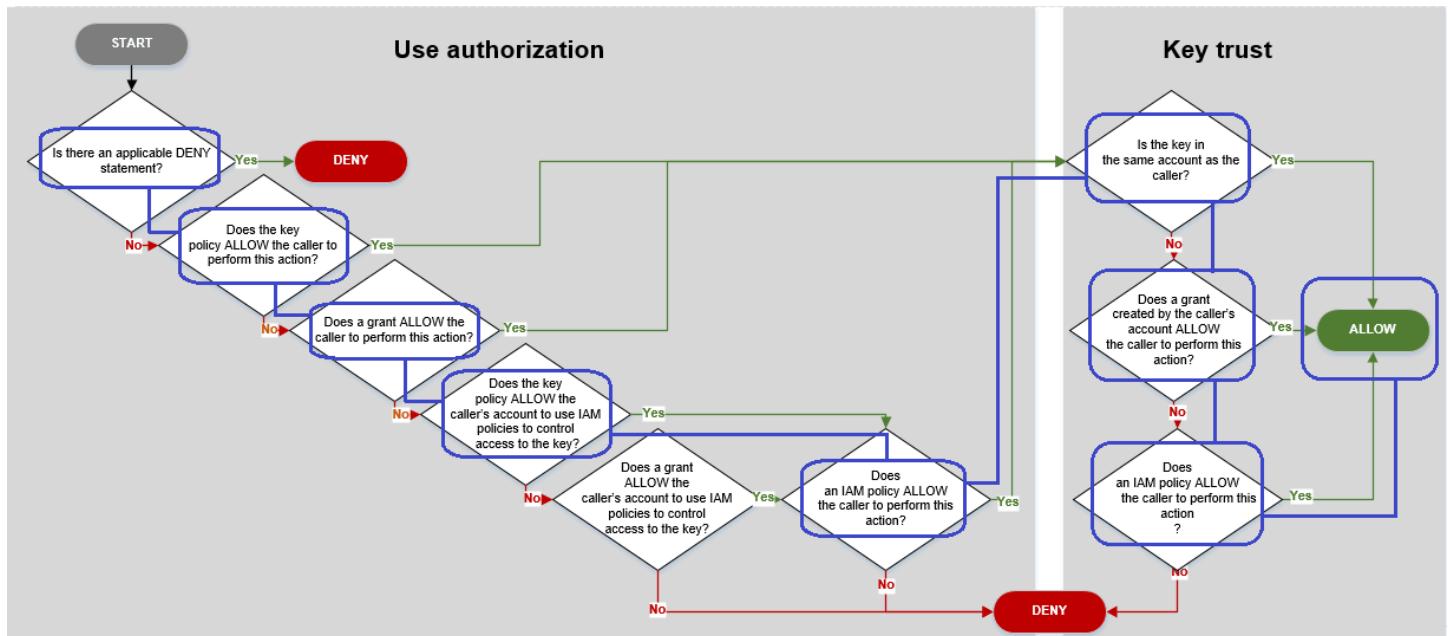
Bob 是帳戶 1 (111122223333) 的使用者。他獲准在[密碼編譯操作](#)中使用帳戶 2 (444455556666) 中的 KMS 金鑰。如何實現？

Tip

評估跨帳戶許可時，請記住，金鑰政策已於 KMS 金鑰的帳戶中指定。在呼叫者帳戶中指定 IAM 政策，即使呼叫者位於不同帳戶中。如需提供跨帳戶存取 KMS 金鑰的詳細資訊，請參閱[允許其他帳戶中的使用者使用 KMS 金鑰](#)。

- 帳戶 2 中 KMS 金鑰的金鑰政策允許帳戶 2 使用 IAM 政策來控制對 KMS 金鑰的存取。
- 帳戶 2 中 KMS 金鑰的金鑰政策允許帳戶 1 在密碼編譯操作中使用 KMS 金鑰。不過，帳戶 1 必須使用 IAM 政策賦予其主體存取 KMS 金鑰的權限。

- 帳戶 1 中的 IAM 政策允許 Engineering 角色使用帳戶 2 的 KMS 金鑰進行密碼編譯操作。
- Bob (帳戶 1 的使用者) 具有採用 Engineering 角色的許可。
- Bob 可以信任此 KMS 金鑰，因為即使它不在他的帳戶中，他帳戶中的 IAM 政策可為他提供此 KMS 金鑰的明確使用許可。



讓我們看看可讓 Bob (帳戶 1 的使用者) 使用帳戶 2 中 KMS 金鑰的政策。

- KMS 金鑰的金鑰政策允許帳戶 2 (444455556666，擁有 KMS 金鑰的帳戶) 使用 IAM 政策來控制對 KMS 金鑰的存取。此金鑰政策也允許帳戶 1 (111122223333) 在密碼編譯操作中使用 KMS 金鑰 (在政策陳述式的 Action 元素中指定)。不過，在帳戶 1 定義可賦予主體存取 KMS 金鑰的 IAM 政策前，帳戶 1 中沒有人可以使用帳戶 2 的 KMS 金鑰。

在流程圖中，帳戶 2 的此金鑰政策符合金鑰政策是否允許呼叫者帳戶使用 IAM 政策來控制對金鑰的存取？條件。

JSON

```
{
  "Id": "key-policy-acct-2",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PermissionUseIAMpolicies",
      "Effect": "Allow",
```

```

    "Principal": {
      "AWS": "arn:aws:iam::444455556666:root"
    },
    "Action": "kms:*",
    "Resource": "*"
  },
  {
    "Sid": "AllowAccount1UseThisKMSKeys",
    "Effect": "Allow",
    "Principal": {
      "AWS": "arn:aws:iam::111122223333:root"
    },
    "Action": [
      "kms:Encrypt",
      "kms:Decrypt",
      "kms:ReEncryptFrom",
      "kms:ReEncryptTo",
      "kms:GenerateDataKey",
      "kms:GenerateDataKeyWithoutPlaintext",
      "kms:DescribeKey"
    ],
    "Resource": "*"
  }
]
}

```

- 發起人 AWS 帳戶（帳戶 1，111122223333）中的 IAM 政策授予委託人使用帳戶 2（444455556666）中的 KMS 金鑰執行密碼編譯操作的許可。Action 元素為主體指派了與帳戶 2 中金鑰政策提供給帳戶 1 相同的許可。若要將這些許可提供給帳戶 1 中的 Engineering 角色，[此內嵌政策會內嵌於 Engineering 角色](#)。

只有在帳戶 2 中 KMS 金鑰的金鑰政策賦予帳戶 1 使用 KMS 金鑰的許可時，這類的跨帳戶 IAM 政策才有效。此外，帳戶 1 賦予其主體執行動作的許可僅限金鑰政策賦予該帳戶的許可。

在流程圖中，這可符合 IAM 政策是否允許呼叫者執行此動作？條件。

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",

```



```

        "Action": [
            "kms:Encrypt",
            "kms:Decrypt",
            "kms:ReEncryptFrom",
            "kms:ReEncryptTo",
            "kms:GenerateDataKey",
            "kms:GenerateDataKeyWithoutPlaintext",
            "kms:DescribeKey"
        ],
        "Resource": [
            "arn:aws:kms:us-
west-2:444455556666:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        ]
    }
}

```

- 最後一個必要元素是帳戶 1 中 Engineering 角色的定義。該角色的 AssumeRolePolicyDocument 可讓 Bob 採用 Engineering 角色。

```

{
  "Role": {
    "Arn": "arn:aws:iam::111122223333:role/Engineering",
    "CreateDate": "2019-05-16T00:09:25Z",
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": {
        "Principal": {
          "AWS": "arn:aws:iam::111122223333:user/bob"
        },
        "Effect": "Allow",
        "Action": "sts:AssumeRole"
      }
    },
    "Path": "/",
    "RoleName": "Engineering",
    "RoleId": "AR0A4KJY2TU23Y7NK62MV"
  }
}

```

AWS KMS 存取控制詞彙表

下列主題說明 AWS KMS 存取控制中的重要術語和概念。

身分驗證

身分驗證是驗證您身分的過程。若要傳送請求至 AWS KMS，您必須 AWS 使用您的 登入資料 AWS 登入。

Authorization

授權提供許可，以傳送建立、管理或使用 AWS KMS 資源的請求。例如，您必須取得授權，才可以在密碼編譯操作中使用 KMS 金鑰。

若要控制對 AWS KMS 資源的存取，請使用[金鑰政策](#)、[IAM 政策和授權](#)。每個 KMS 金鑰都必須有一個金鑰政策。如果金鑰政策允許，您也可以使用 IAM 政策和授予，以允許主體存取 KMS 金鑰。若要優化您的授權，您可以使用[條件金鑰](#)，只有在請求或資源符合您指定的條件時，條件金鑰才允許或拒絕存取。您也可以允許存取[其他 AWS 帳戶](#) 中您信任的主體。

使用身分驗證

身分驗證是您 AWS 使用身分憑證登入 的方式。您必須驗證為 AWS 帳戶根使用者、IAM 使用者或擔任 IAM 角色。

您可以使用身分來源的登入資料，例如 AWS IAM Identity Center (IAM Identity Center)、單一登入身分驗證或 Google/Facebook 登入資料，以聯合身分的形式登入。如需有關登入的詳細資訊，請參閱《AWS 登入 使用者指南》中的[如何登入您的 AWS 帳戶](#)。

對於程式設計存取，AWS 提供 SDK 和 CLI 以密碼編譯方式簽署請求。如需詳細資訊，請參閱《IAM 使用者指南》中的[API 請求的AWS 第 4 版簽署程序](#)。

AWS 帳戶 根使用者

當您建立 時 AWS 帳戶，您會從一個名為 AWS 帳戶 theroot 使用者的登入身分開始，該身分具有對所有 AWS 服務 和 資源的完整存取權。強烈建議不要使用根使用者來執行日常任務。有關需要根使用者憑證的任務，請參閱《IAM 使用者指南》中的[需要根使用者憑證的任務](#)。

聯合身分

最佳實務是要求人類使用者使用聯合身分提供者，以 AWS 服務 使用臨時憑證存取。

聯合身分是您企業目錄、Web 身分提供者的使用者，或使用來自身分來源的 AWS 服務憑證存取 Directory Service。聯合身分會擔任角色，而該角色會提供臨時憑證。

若需集中化管理存取權限，建議使用 AWS IAM Identity Center。如需詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的 [什麼是 IAM Identity Center？](#)。

IAM 使用者和群組

IAM 使用者 https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users.html 是一種身分具備單人或應用程式的特定許可權。建議以臨時憑證取代具備長期憑證的 IAM 使用者。如需詳細資訊，請參閱《IAM 使用者指南》中的 [要求人類使用者使用聯合身分提供者來 AWS 使用臨時憑證存取](#)。

[IAM 群組](#) 會指定 IAM 使用者集合，使管理大量使用者的許可權更加輕鬆。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 使用者的使用案例](#)。

IAM 角色

IAM 角色 https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles.html 的身分具有特定許可權，其可以提供臨時憑證。您可以透過 [從使用者切換到 IAM 角色（主控台）](#) 或呼叫 AWS CLI 或 AWS API 操作來擔任角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [擔任角色的方法](#)。

IAM 角色適用於聯合身分使用者存取、臨時 IAM 使用者許可、跨帳戶存取權與跨服務存取，以及在 Amazon EC2 執行的應用程式。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的快帳戶資源存取](#)。

使用政策管理存取權

您可以透過建立政策並將其連接到身分或資源 AWS 來控制 AWS 中的存取。政策定義與身分或資源相關聯的許可。當委託人提出請求時 AWS，會評估這些政策。大多數政策會以 JSON 文件 AWS 形式存放在中。如需進一步了解 JSON 政策文件，請參閱《IAM 使用者指南》中的 [JSON 政策概觀](#)。

管理員會使用政策，透過定義哪些主體可在哪些條件下對哪些資源執行動作，以指定可存取的範圍。

預設情況下，使用者和角色沒有許可。IAM 管理員會建立 IAM 政策並將其新增至角色，供使用者後續擔任。IAM 政策定義動作的許可，無論採用何種方式執行。

身分型政策

身分型政策是附加至身分 (使用者、使用者群組或角色) 的 JSON 許可政策文件。這類政策控制身分可對哪些資源執行哪些動作，以及適用的條件。如需了解如何建立身分型政策，請參閱《IAM 使用者指南》中的 [透過客戶管理政策定義自訂 IAM 許可](#)。

身分型政策可分為內嵌政策 (直接內嵌於單一身分) 與受管政策 (可附加至多個身分的獨立政策)。如需了解如何在受管政策及內嵌政策之間做選擇，請參閱《IAM 使用者指南》中的[在受管政策與內嵌政策之間選擇](#)。

資源型政策

AWS KMS [Ankey 政策](#)是以資源為基礎的政策，可控制對 KMS 金鑰的存取。每個 KMS 金鑰都必須有一個金鑰政策。您可以使用其他授權機制來允許存取 KMS 金鑰，但前提是金鑰政策允許。(您可以使用 IAM 政策來拒絕對 KMS 金鑰的存取，即使金鑰政策未明確允許。)

資源型政策是連接到資源 (如 KMS 金鑰) 的 JSON 政策文件，以控制對特定資源的存取。資源型政策可定義指定的主體在何種條件下對該資源執行哪些動作。您不要在資源型政策中指定資源，但必須指定委託人，例如帳戶、使用者、角色、聯合身分使用者或 AWS 服務。資源型政策是管理該資源的服務中的內嵌政策。您無法在資源型政策中使用來自 IAM 的 AWS 受管政策，例如 [AWSKeyManagementServicePowerUser](#) 受管政策。

其他政策類型

AWS 支援其他政策類型，可設定更多常見政策類型授予的最大許可：

- 許可界限 — 設定身分型政策可授與 IAM 實體的最大許可。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 實體許可界限](#)。
- 服務控制政策 (SCP) — 為 AWS Organizations 中的組織或組織單位指定最大許可。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的 [服務控制政策](#)。
- 資源控制政策 (RCP) — 設定您帳戶中資源可用許可的上限。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的 [資源控制政策 \(RCP\)](#)。
- 工作階段政策 — 在以程式設計方式為角色或聯合身分使用者建立臨時工作階段時，以參數形式傳遞的進階政策。如需詳細資訊，請參閱《IAM 使用者指南》中的 [工作階段政策](#)。

多種政策類型

當多種類型的政策適用於請求時，產生的許可會更複雜而無法理解。若要了解如何 AWS 在涉及多種政策類型時決定是否允許請求，請參閱《IAM 使用者指南》中的 [政策評估邏輯](#)。

AWS KMS 資源

在中 AWS KMS，主要資源是 AWS KMS key。AWS KMS 也支援[別名](#)，這是為 KMS 金鑰提供易記名稱的獨立資源。有些 AWS KMS 操作可讓您使用別名來識別 KMS 金鑰。

KMS 金鑰或別名的每個執行個體都具有唯一的 [Amazon Resource Name](#) (ARN)，具有標準格式。在 AWS KMS 資源中，AWS 服務名稱為 kms。

- AWS KMS key

ARN 格式：

`arn:AWS partition name:AWS service name:AWS #:AWS ## ID:key/key ID`

範例 ARN：

`arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab`

- Alias (別名)

ARN 格式：

`arn:AWS partition name:AWS service name:AWS #:AWS ## ID:alias/alias name`

範例 ARN：

`arn:aws:kms:us-west-2:111122223333:alias/example-alias`

AWS KMS 提供一組 API 操作，以使用您的 AWS KMS 資源。如需在 AWS 管理主控台 和 AWS KMS API 操作中識別 KMS 金鑰的詳細資訊，請參閱 [金鑰識別符 \(KeyID\)](#)。如需 AWS KMS 操作清單，請參閱 [AWS Key Management Service API 參考](#)。

建立 KMS 金鑰

您可以在 AWS KMS keys 中 AWS 管理主控台或使用 [CreateKey](#) 操作或 [AWS::KMS::Key AWS CloudFormation 資源](#) 來建立。在此過程中，您可以設定 KMS 金鑰的金鑰政策，您可以隨時變更。您也可以選取下列值來定義您建立的 KMS 金鑰類型。建立 KMS 金鑰之後即無法變更這些屬性。

KMS 金鑰類型

金鑰類型是一種屬性，可決定建立的密碼編譯金鑰類型。AWS KMS 提供三種金鑰類型來保護資料：

- 進階加密標準 (AES) 對稱金鑰

256 位元金鑰，用於 AES 的 Galois 計數器模式 (GCM) 模式，以提供大小小於 4KB 的資料經過驗證的加密/解密。這是最常見的金鑰類型，用於保護應用程式中使用的其他資料加密金鑰 AWS 服務，以及代表您加密資料的方式。

- RSA、橢圓曲線或 SM2（僅限中國區域）非對稱金鑰

這些金鑰提供各種大小，並支援許多演算法。它們可用於加密和解密、簽署和驗證，或根據演算法選擇衍生共用秘密操作。

- 用於執行雜湊型訊息驗證碼 (HMAC) 操作的對稱金鑰

這些金鑰是用於簽署和驗證操作的 256 位元金鑰。

KMS 金鑰無法以純文字從服務匯出。它們是由 產生，並且只能在 服務使用的硬體安全模組 (HSMs) 內使用。這是 的基本安全屬性 AWS KMS，以確保金鑰不會洩露。

金鑰用量

金鑰用途是決定該金鑰支援之密碼編譯操作的屬性。KMS 金鑰的金鑰使用量可以是 ENCRYPT_DECRYPT、GENERATE_VERIFY_MAC、SIGN_VERIFY 或 KEY_AGREEMENT。每個 KMS 金鑰都僅有一種金鑰用途。這遵循美國 [國家標準技術研究所 \(NIST\) 特別出版物 800-57 的金鑰管理建議](#) 第 5.2 節金鑰用量的金鑰使用最佳實務。使用 KMS 金鑰執行多種類型的操作，會使得兩種操作的產品更容易受到攻擊。

金鑰規格

金鑰規格是一種代表金鑰密碼編譯組態的屬性。金鑰規格的含義與金鑰類型不同。

對於 KMS 金鑰，金鑰規格會決定 KMS 金鑰是對稱或非對稱。其也會決定其金鑰材料的類型，及其支援的演算法。

預設金鑰規格 [SYMMETRIC_DEFAULT](#) 表示 256 位元對稱加密金鑰。如需所有支援金鑰規格的詳細說明，請參閱 [金鑰規格參考](#)。

金鑰材料來源

金鑰材料來源 是 KMS 金鑰屬性，用於識別 KMS 金鑰中金鑰材料的來源。您可以在建立 KMS 金鑰時選擇金鑰材料來源，之後便無法進行變更。金鑰材料的來源會影響 KMS 金鑰的安全性、持久性、可用性、延遲和輸送量特性。

每個 KMS 金鑰在中繼資料中都包含對其金鑰資料的引用。對稱加密 KMS 金鑰的金鑰資料來源可能會有所不同。您可以使用 AWS KMS 產生的金鑰材料、在 [自訂金鑰存放](#) 區中產生的金鑰材料，或 [匯入您自己的金鑰材料](#)。

根據預設，每個 KMS 金鑰都有唯一的金鑰材料。不過，您可以建立一組具有相同金鑰材料的 [多區域金鑰](#)。

KMS 金鑰可以具有下列其中一個金鑰材料來源值：AWS_KMS、EXTERNAL([匯入的金鑰材料](#))、AWS_CLOUDHSM ([AWS CloudHSM 金鑰存放區中的 KMS 金鑰](#)) 或 EXTERNAL_KEY_STORE (外部金鑰存放區中的 KMS 金鑰)。???

主題

- [建立 KMS 金鑰的許可](#)
- [選擇要建立的 KMS 金鑰類型](#)
- [建立對稱加密 KMS 金鑰](#)
- [建立非對稱 KMS 金鑰](#)
- [建立 HMAC KMS 金鑰](#)
- [建立多區域主索引鍵](#)
- [建立多區域複本金鑰](#)
- [使用匯入的金鑰材料建立 KMS 金鑰](#)
- [在金鑰存放區中建立 KMS AWS CloudHSM 金鑰](#)
- [在外部金鑰存放區中建立 KMS 金鑰](#)

建立 KMS 金鑰的許可

若要在主控台或使用 API 建立 KMS 金鑰，您必須在 IAM 政策中具有下列許可。盡可能使用[條件金鑰](#)以限制許可。例如，您可使用 IAM 政策中的 [kms:KeySpec](#) 條件金鑰，以僅允許主體建立對稱加密 KMS 金鑰。

如需建立金鑰之主體的 IAM 政策範例，請參閱 [允許使用者建立 KMS 金鑰](#)。

Note

授予主體管理標籤和別名的許可時，請務必謹慎。變更標記或別名可允許或拒絕客戶受管金鑰的許可。如需詳細資訊，請參閱[適用於的 ABAC AWS KMS](#)。

- [kms:CreateKey](#) 是必要項目。
- [kms:CreateKey](#) 是在主控台中建立 KMS 金鑰的必要項目，其中每個新 KMS 金鑰都需要別名。
- [kms:TagResource](#) 是建立 KMS 金鑰時新增標籤的必要項目。
- [iam:CreateServiceLinkedRole](#) 是建立多區域主要金鑰的必要項目。如需詳細資訊，請參閱[控制對多區域金鑰的存取](#)。

建立 KMS 金鑰時，不需要 [kms:PutKeyPolicy](#) 許可。kms:CreateKey 許可包含設定初始金鑰政策的許可。但是，您必須在建立 KMS 金鑰時將此許可新增至金鑰政策，以確保您可以控制對 KMS 金鑰的存取。另一種方法是使用 [BypassLockoutSafetyCheck](#) 參數，但不建議使用。

KMS 金鑰屬於建立金鑰 AWS 的帳戶。建立 KMS 金鑰的 IAM 使用者不會被視為金鑰擁有者，而且他們不會自動擁有使用或管理他們建立之 KMS 金鑰的許可。與任何其他主體一樣，金鑰建立者必須透過金鑰政策、IAM 政策或授予取得許可。不過，擁有 kms:CreateKey 許可的主體可以設定初始金鑰政策，並授予自己使用或管理金鑰的許可。

選擇要建立的 KMS 金鑰類型

您建立的 KMS 金鑰類型，主要取決於您計劃如何使用 KMS 金鑰、您的安全需求，以及您的授權需求。KMS 金鑰的金鑰類型和金鑰用量決定金鑰可執行哪些密碼編譯操作。每個 KMS 金鑰僅有一個金鑰用途。使用 KMS 金鑰執行多種類型的操作，會使得所有操作的產品更容易受到攻擊。

若要允許主體僅為特定金鑰用途建立 KMS 金鑰，請使用 [kms:KeyUsage](#) 條件金鑰。您也可以使用 kms:KeyUsage 條件索引鍵來允許委託人根據金鑰使用情形呼叫 KMS 金鑰的 API 操作。例如，您可以允許只有在金鑰使用情形為 SIGN_VERIFY 時，才能停用 KMS 金鑰的許可。

根據您的使用案例，使用下列指導判斷您需要的 KMS 金鑰類型。

加密和解密資料

請使用[對稱 KMS 金鑰](#)處理對於大多數需要加密和解密資料的使用案例。AWS KMS 使用的對稱加密演算法快速又有效率，並可確保資料的機密性和真實性。其支援經定義為[加密內容](#)之額外的驗證資料 (AAD) 所驗證的加密。這種類型的 KMS 金鑰需要加密資料的寄件者和收件人擁有有效的 AWS 登入資料才能呼叫 AWS KMS。

如果您的使用案例需要在 外部 AWS 由無法呼叫 的使用者加密 AWS KMS，[非對稱 KMS 金鑰](#)是不錯的選擇。您可以分配非對稱 KMS 金鑰的公有金鑰，讓這些使用者加密資料。而需要解密該資料的應用程式，則可以使用 AWS KMS 內的非對稱 KMS 金鑰私有金鑰。

簽署訊息及驗證簽章

若要簽署訊息及驗證簽章，您必須使用[非對稱 KMS 金鑰](#)。您可以使用 KMS 金鑰搭配代表 RSA 金鑰對、橢圓曲線 (ECC) 金鑰對、ML-DSA 金鑰對或 SM2 金鑰對的[金鑰規格](#)（僅限中國區域）。您選擇的金鑰規格取決於您要使用的簽署演算法。ECC 金鑰對支援的 ECDSA 簽章演算法優於 RSA 簽章演算法。從 RSA 或 ECC 金鑰遷移至後量子金鑰時，請使用 ML-DSA 金鑰對。但是，您可能需要使用特定的金鑰規格和簽章演算法，以支援在 AWS 以外驗證簽章的使用者。

使用非對稱金鑰對加密

若要使用非對稱金鑰對加密資料，您必須使用非對稱 KMS 金鑰搭配[RSA 金鑰規格](#)或[SM2 金鑰規格](#)（僅限中國區域）。若要使用 KMS 金鑰的公有金鑰加密 AWS KMS 的資料，請使用[Encrypt](#) 操作。您也可以[下載公有金鑰](#)，並與需要在 外部加密資料的各方共用 AWS KMS。

當您下載非對稱 KMS 金鑰的公有金鑰時，您可以在 AWS KMS 之外使用。但是，它不再受保護 KMS 金鑰的安全控制約束 AWS KMS。例如，您無法使用 AWS KMS 金鑰政策或授權來控制公有金鑰的使用。您也無法控制金鑰是否僅用於使用 AWS KMS 支援的加密演算法進行加密和解密。如需詳細資訊，請參閱[下載公有金鑰的特殊考量](#)。

若要解密在 外部使用公有金鑰加密的資料 AWS KMS，請呼叫[Decrypt](#) 操作。如果資料是以公有金鑰從 KMS 金鑰加密，且金鑰使用量為 `SIGN_VERIFY`，Decrypt 則操作會失敗。如果使用 AWS KMS 不支援您所選金鑰規格的演算法加密，則也會失敗。如需金鑰規格和支援的演算法的詳細資訊，請參閱[金鑰規格參考](#)。

為了避免這些錯誤，在 外部使用公有金鑰的任何人 AWS KMS 都必須存放金鑰組態。AWS KMS 主控台和[GetPublicKey](#) 回應提供共用公有金鑰時必須包含的資訊。

衍生共用秘密

若要衍生共用秘密，請使用 KMS 金鑰搭配 [NIST 標準橢圓曲線](#) 或 [SM2](#)（僅限中國區域）金鑰材料。AWS KMS 會使用 [橢圓曲線密碼編譯 Cofactor Diffie-Hellman 基本](#) (ECDH)，透過從橢圓曲線公有私有金鑰對衍生共用秘密，在兩個對之間建立金鑰協議。您可以使用 [DeriveSharedSecret](#) 操作傳回的原始共用秘密來衍生對稱金鑰，以加密和解密在兩方之間傳送的資料，或產生和驗證 HMACs。AWS KMS 當您使用原始共用秘密衍生對稱金鑰時，建議您遵循 [金鑰衍生的 NIST 建議](#)。

產生和驗證 HMAC 代碼

若要產生和驗證雜湊訊息驗證碼，請使用 HMAC KMS 金鑰。當您在 中建立 HMAC 金鑰時 AWS KMS，會 AWS KMS 建立並保護您的金鑰材料，並確保您為金鑰使用正確的 MAC 演算法。HMAC 代碼也可以用來作為虛擬亂數，並在特定情況下用於對稱簽署和字符化。

HMAC KMS 金鑰為對稱金鑰。在 AWS KMS 主控台中建立 HMAC KMS 金鑰時，請選擇 Symmetric 金鑰類型。

搭配 AWS 服務使用

若要建立 KMS 金鑰以與 [AWS 整合 AWS KMS](#) 的服務搭配使用，請參閱 `service. AWS services` 的文件。加密資料的需要 [對稱加密 KMS 金鑰](#)。

除了這些考量外，具有不同金鑰規格的 KMS 金鑰的密碼編譯操作也有不同的定價和不同的請求配額。如需 AWS KMS 定價的資訊，請參閱 [AWS Key Management Service 定價](#)。如需請求配額的詳細資訊，請參閱 [請求配額](#)。

建立對稱加密 KMS 金鑰

本主題說明如何建立基本 KMS 金鑰，這是具有金鑰材料之單一區域的 [對稱加密 KMS](#) 金鑰 AWS KMS。您可使用此 KMS 金鑰來保護 AWS 服務中的資料。

您可以在 AWS KMS 主控台中使用 [CreateKey](#) API 或使用 [AWS::KMS::Key CloudFormation](#)。

預設金鑰規格 [SYMMETRIC_DEFAULT](#) 是對稱加密 KMS 金鑰的金鑰規格。當您在 AWS KMS 主控台中選取對稱金鑰類型和加密和解密金鑰用量時，它會選取 SYMMETRIC_DEFAULT 金鑰規格。在 [CreateKey](#) 操作中，如果您不指定 KeySpec 值，即會選取 SYMMETRIC_DEFAULT。如果沒有使用不同金鑰規格的理由，SYMMETRIC_DEFAULT 是很好的選擇。

如需 KMS 金鑰配額的詳細資訊，請參閱 [配額](#)。

使用 AWS KMS 主控台

您可以使用 AWS 管理主控台 建立 AWS KMS keys (KMS 金鑰)。

Important

請勿在別名、說明或標籤包含機密或敏感資訊。這些欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
4. 選擇建立金鑰。
5. 若要建立對稱加密 KMS 金鑰，在 Key type (金鑰類型) 欄位中，選擇 Symmetric (對稱)。
6. 在 Key usage (金鑰用途) 欄位中，系統會自動選取 Encrypt and decrypt (加密和解密) 選項。
7. 選擇下一步。
8. 輸入 KMS 金鑰的別名。別名名稱的開頭不可以是 **aws/**。字 **aws/** 首由 Amazon Web Services 保留，以在您的帳戶 AWS 受管金鑰 中代表。

Note

新增、刪除或更新別名可允許或拒絕 KMS 金鑰的許可。如需詳細資訊，請參閱 [適用於的 ABAC AWS KMS](#) 和 [使用別名來控制對 KMS 金鑰的存取](#)。

別名是您可用來識別 KMS 金鑰的顯示名稱。我們建議您選擇別名來表示您計劃保護的資料類型，或您計劃搭配 KMS 金鑰一起使用的應用程式。

在 AWS 管理主控台中建立 KMS 金鑰時需要別名。但在使用 [CreateKey](#) 操作時是選用的。

9. (選用) 輸入 KMS 金鑰的描述。

您可以立即新增描述或在任意時間更新，除非 [金鑰狀態](#) 為 Pending Deletion 或 Pending Replica Deletion。若要新增、變更或刪除現有客戶受管金鑰的描述，請在 [中編輯 KMS 金鑰詳細資訊](#) 頁面上的描述，AWS 管理主控台 或使用 [UpdateKeyDescription](#) 操作。

10. (選用) 輸入標籤索引鍵和選用標籤值。若要將其他標籤新增至 KMS 金鑰，請選擇 Add tag (新增標籤)。

 Note

標記或取消標記 KMS 金鑰可以允許或拒絕 KMS 金鑰的許可。如需詳細資訊，請參閱 [適用於的 ABAC AWS KMS](#) 和 [使用標籤來控制對 KMS 金鑰的存取](#)。

當您將標籤新增至 AWS 資源時，AWS 會產生成本分配報告，其中包含依標籤彙總的用量和成本。標籤也可以用來控制 KMS 金鑰的存取。如需標記 KMS 金鑰的詳細資訊，請參閱 [中的標籤 AWS KMS](#) 和 [適用於的 ABAC AWS KMS](#)。

11. 選擇下一步。
12. 選取可管理 KMS 金鑰的 IAM 使用者和角色。

 備註

此金鑰政策提供此 KMS 金鑰的完整 AWS 帳戶 控制權。它允許帳戶管理員使用 IAM 政策授予其他主體管理 KMS 金鑰的許可。如需詳細資訊，請參閱 [the section called “預設金鑰政策”](#)。

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的安全性最佳實務](#)。

AWS KMS 主控台會將金鑰管理員新增至陳述式識別碼下的金鑰政策 "Allow access for Key Administrators"。修改此陳述式識別符可能會影響主控台顯示您對陳述式所做的更新。

13. (選用) 為了防止選取的 IAM 使用者和角色刪除此 KMS 金鑰，請在頁面底部的 Key deletion (金鑰刪除) 區段中，清除 Allow key administrators to delete this key (允許金鑰管理員刪除此金鑰) 核取方塊。
14. 選擇下一步。
15. 選取可將金鑰用於 [密碼編譯操作](#) 的 IAM 使用者和角色

 備註

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的安全性最佳實務](#)。

AWS KMS 主控台會將金鑰使用者新增至陳述式識別碼 "Allow use of the key" 和下的金鑰政策 "Allow attachment of persistent resources"。修改這些陳述式識別符可能會影響主控台顯示您對陳述式所做的更新的方式。

16. (選用) 您可以允許其他 AWS 帳戶使用此 KMS 金鑰進行密碼編譯操作。若要這樣做，請在頁面底部的其他 AWS 帳戶區段中，選擇新增另一個 AWS 帳戶，然後輸入外部帳戶的 AWS 帳戶識別號碼。若要新增多個外部帳戶，請重複此步驟。

Note

若要允許外部帳戶中的主體使用 KMS 金鑰，外部帳戶的管理員必須建立 IAM 政策來提供這些許可。如需詳細資訊，請參閱 [允許其他帳戶中的使用者使用 KMS 金鑰](#)。

17. 選擇下一步。
18. 檢閱金鑰的金鑰政策陳述式。若要變更金鑰政策，請選取編輯。
19. 選擇下一步。
20. 檢閱您選擇的金鑰設定。您仍然可以返回並變更所有設定。
21. 選擇 Finish (完成) 來建立 KMS 金鑰。

使用 AWS KMS API

您可以使用 [CreateKey](#) AWS KMS keys 操作來建立所有類型的。這些範例使用 [AWS Command Line Interface \(AWS CLI\)](#)。如需多種程式設計語言的範例，請參閱 [CreateKey 搭配 AWS SDK 或 CLI 使用](#)。

Important

請勿在 Description 或 Tags 欄位包含機密或敏感資訊。這些欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

下列操作會在由產生的金鑰材料支援的單一區域中建立對稱加密金鑰 AWS KMS。此操作沒有必要參數。但您可能還想要使用 Policy 參數指定金鑰政策。您隨時可以變更金鑰政策 ([PutKeyPolicy](#)) 並新增選用元素，例如[描述](#)和[標籤](#)。您還可以建立[非對稱金鑰](#)、[多區域金鑰](#)、[匯入金鑰資料](#)中的金鑰，以及[自訂金鑰存放區](#)中的金鑰。若要建立用於用戶端加密的資料金鑰，請使用 [GenerateDataKey](#) 操作。

CreateKey 操作不允許您指定別名，但您可以使用 [CreateAlias](#) 操作來建立新 KMS 金鑰的別名。

以下範例呼叫 `CreateKey` 操作，不帶參數。這個命令會使用所有的預設值。其會使用 AWS KMS 產生的金鑰資料來建立對稱加密 KMS 金鑰。

```
$ aws kms create-key
{
  "KeyMetadata": {
    "Origin": "AWS_KMS",
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "Description": "",
    "KeyManager": "CUSTOMER",
    "Enabled": true,
    "KeySpec": "SYMMETRIC_DEFAULT",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "KeyState": "Enabled",
    "CreationDate": 1502910355.475,
    "Arn": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "AWSAccountId": "111122223333",
    "MultiRegion": false
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ],
  }
}
```

如果您不為您的新 KMS 金鑰指定金鑰政策，則在用其建立新 KMS 金鑰時，`CreateKey` 套用的[預設金鑰政策](#)會不同於主控台套用的預設金鑰政策。

例如，呼叫 [GetKeyPolicy](#) 操作會傳回 `CreateKey` 套用的金鑰政策。它提供 KMS 金鑰的 AWS 帳戶存取權，並允許它為 KMS 金鑰建立 AWS Identity and Access Management (IAM) 政策。如需有關 KMS 金鑰之 IAM 政策和金鑰政策的詳細資訊，請參閱 [KMS 金鑰存取和許可](#)。

```
$ aws kms get-key-policy --key-id 1234abcd-12ab-34cd-56ef-1234567890ab --policy-name
default --output text
```

JSON

```
{
  "Version": "2012-10-17",
  "Id" : "key-default-1",
```



```
"Statement" : [ {  
  "Sid" : "EnableIAMUserPermissions",  
  "Effect" : "Allow",  
  "Principal" : {  
    "AWS" : "arn:aws:iam::111122223333:root"  
  },  
  "Action" : "kms:*",  
  "Resource" : "*"   
} ]  
}
```

建立非對稱 KMS 金鑰

您可以在 AWS KMS 主控台使用 [CreateKey](#) API 或使用 `AWS::KMS::Key` 範本來建立 [非對稱 KMS 金鑰](#)。[AWS::KMS::Key CloudFormation](#) 非對稱 KMS 金鑰代表可用於加密、簽署或衍生共用秘密的公有和私有金鑰對。私有金鑰會保留在內 AWS KMS。若要下載要在外部使用的公有金鑰 AWS KMS，請參閱 [下載公有金鑰](#)。

建立非對稱 KMS 金鑰時，您必須選取金鑰規格。通常，您選取的金鑰規格取決於法規、安全或業務需求。但也可能受需要加密或簽署之訊息的大小所左右。一般而言，較長的加密金鑰更能抵禦暴力破解攻擊。如需所有支援金鑰規格的詳細說明，請參閱 [金鑰規格參考](#)。

AWS 與整合的服務 AWS KMS 不支援非對稱 KMS 金鑰。如果您想要建立 KMS 金鑰來加密您在 AWS 服務中存放或管理的資料，[請建立對稱加密 KMS 金鑰](#)。

如需建立 KMS 金鑰所需之許可的詳細資訊，請參閱 [建立 KMS 金鑰的許可](#)。

使用 AWS KMS 主控台

您可以使用 AWS 管理主控台 來建立非對稱 AWS KMS keys (KMS 金鑰)。每個非對稱 KMS 金鑰皆代表一對公有和私有金鑰對。

Important

請勿在別名、說明或標籤包含機密或敏感資訊。這些欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。

2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
4. 選擇建立金鑰。
5. 若要建立非對稱 KMS 金鑰，請在金鑰類型中選擇 Asymmetric (非對稱)。
6. 若要建立用於公有金鑰加密的非對稱 KMS 金鑰，請在金鑰使用情形中選擇 Encrypt and decrypt (加密和解密)。

若要建立用於簽署訊息和驗證簽章的非對稱 KMS 金鑰，請在金鑰用量中選擇簽署和驗證。

若要建立用於衍生共用秘密的非對稱 KMS 金鑰，請在金鑰用量中選擇金鑰協議。

如需選擇金鑰用途值的說明，請參閱[選擇要建立的 KMS 金鑰類型](#)。

7. 選取非對稱 KMS 金鑰的規格 (Key spec (金鑰規格))。
8. 選擇下一步。
9. 輸入 KMS 金鑰的**別名**。別名名稱的開頭不可以是 **aws/**。**aws/** 字首由 Amazon Web Services 保留，以代表您帳戶中的 AWS 受管金鑰。

別名是易記的名稱，可用來識別主控台和部分 AWS KMS APIs 中的 KMS 金鑰。我們建議您選擇別名來表示您計劃保護的資料類型，或您計劃搭配 KMS 金鑰一起使用的應用程式。

在 AWS 管理主控台中建立 KMS 金鑰時需要別名。使用 [CreateKey](#) 操作時，您不能指定別名，但您可以使用主控台或 [CreateAlias](#) 操作來建立現有 KMS 金鑰的別名。如需詳細資訊，請參閱[中的別名 AWS KMS](#)。

10. (選用) 輸入 KMS 金鑰的描述。

輸入描述來說明您計劃保護的資料類型，或您計劃搭配 KMS 金鑰一起使用的應用程式。

您可以立即新增描述或在任意時間更新，除非**金鑰狀態**為 Pending Deletion 或 Pending Replica Deletion。若要新增、變更或刪除現有客戶受管金鑰的描述，請在 中編輯 KMS 金鑰詳細資訊頁面上的描述，AWS 管理主控台 或使用 [UpdateKeyDescription](#) 操作。

11. (選用) 輸入標籤索引鍵和選用標籤值。若要將其他標籤新增至 KMS 金鑰，請選擇 Add tag (新增標籤)。

當您將標籤新增至 AWS 資源時，AWS 會產生成本分配報告，其中包含依標籤彙總的用量和成本。標籤也可以用來控制 KMS 金鑰的存取。如需標記 KMS 金鑰的詳細資訊，請參閱 [中的標籤 AWS KMS](#) 和 [適用於的 ABAC AWS KMS](#)。

12. 選擇下一步。

13. 選取可管理 KMS 金鑰的 IAM 使用者和角色。

備註

此金鑰政策提供此 KMS 金鑰的完整 AWS 帳戶 控制權。它允許帳戶管理員使用 IAM 政策授予其他主體管理 KMS 金鑰的許可。如需詳細資訊，請參閱[the section called “預設金鑰政策”](#)。

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的安全性最佳實務](#)。

AWS KMS 主控台會將金鑰管理員新增至陳述式識別碼 下的金鑰政策 "Allow access for Key Administrators"。修改此陳述式識別符可能會影響主控台顯示您對陳述式所做的更新的方式。

14. (選用) 為了防止選取的 IAM 使用者和角色刪除此 KMS 金鑰，請在頁面底部的 Key deletion (金鑰刪除) 區段中，清除 Allow key administrators to delete this key (允許金鑰管理員刪除此金鑰) 核取方塊。

15. 選擇下一步。

16. 選取可將 KMS 金鑰用於[密碼編譯操作](#)的 IAM 使用者和角色。

備註

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的安全性最佳實務](#)。

AWS KMS 主控台會將金鑰使用者新增至陳述式識別碼 "Allow use of the key" 和 下的金鑰政策 "Allow attachment of persistent resources"。修改這些陳述式識別符可能會影響主控台顯示您對陳述式所做的更新。

17. (選用) 您可以允許其他 AWS 帳戶 使用此 KMS 金鑰進行密碼編譯操作。若要這樣做，請在頁面底部的其他 AWS 帳戶 區段中，選擇新增另一個 AWS 帳戶，然後輸入外部帳戶的 AWS 帳戶 識別號碼。若要新增多個外部帳戶，請重複此步驟。

Note

若要允許外部帳戶中的主體使用 KMS 金鑰，外部帳戶的管理員必須建立 IAM 政策來提供這些許可。如需詳細資訊，請參閱[允許其他帳戶中的使用者使用 KMS 金鑰](#)。

18. 選擇下一步。

19. 檢閱金鑰的金鑰政策陳述式。若要變更金鑰政策，請選取編輯。
20. 選擇下一步。
21. 檢閱您選擇的金鑰設定。您仍然可以返回並變更所有設定。
22. 選擇 Finish (完成) 來建立 KMS 金鑰。

使用 AWS KMS API

您可以使用 [CreateKey](#) 操作來建立非對稱 AWS KMS key。以下範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

當您建立非對稱 KMS 金鑰時，必須指定 KeySpec 參數，決定您建立的金鑰類型。此外，您必須指定 ENCRYPT_DECRYPT、SIGN_VERIFY 或 KEY_AGREEMENT KeyUsage 的值。建立 KMS 金鑰之後即無法變更這些屬性。

CreateKey 操作不允許您指定別名，但您可以使用 [CreateAlias](#) 操作來建立新 KMS 金鑰的別名。

Important

請勿在 Description 或 Tags 欄位包含機密或敏感資訊。這些欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

建立用於公有加密的非對稱 KMS 金鑰對

下列範例會使用 CreateKey 操作建立專為公有金鑰加密設計之 4096 位元 RSA 金鑰的非對稱 KMS 金鑰。

```
$ aws kms create-key --key-spec RSA_4096 --key-usage ENCRYPT_DECRYPT
{
  "KeyMetadata": {
    "KeyState": "Enabled",
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",
    "Description": "",
    "Arn": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "CreationDate": 1569973196.214,
    "MultiRegion": false,
    "KeySpec": "RSA_4096",
    "CustomerMasterKeySpec": "RSA_4096",
```

```

    "KeyUsage": "ENCRYPT_DECRYPT",
    "EncryptionAlgorithms": [
        "RSAES_OAEP_SHA_1",
        "RSAES_OAEP_SHA_256"
    ],
    "AWSAccountId": "111122223333",
    "Origin": "AWS_KMS",
    "Enabled": true
}
}

```

建立用於簽署和驗證的非對稱 KMS 金鑰對

下列範例命令會建立非對稱 KMS 金鑰，代表用於簽署和驗證的一組 ECC 金鑰。您無法建立用於加密和解密的橢圓曲線金鑰對。

```

$ aws kms create-key --key-spec ECC_NIST_P521 --key-usage SIGN_VERIFY
{
  "KeyMetadata": {
    "KeyState": "Enabled",
    "KeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
    "CreationDate": 1570824817.837,
    "Origin": "AWS_KMS",
    "SigningAlgorithms": [
      "ECDSA_SHA_512"
    ],
    "Arn": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321",
    "AWSAccountId": "111122223333",
    "KeySpec": "ECC_NIST_P521",
    "CustomerMasterKeySpec": "ECC_NIST_P521",
    "KeyManager": "CUSTOMER",
    "Description": "",
    "Enabled": true,
    "MultiRegion": false,
    "KeyUsage": "SIGN_VERIFY"
  }
}

```

建立非對稱 KMS 金鑰對以衍生共用秘密

下列範例命令會建立非對稱 KMS 金鑰，代表用於衍生共用秘密的 ECDH 金鑰對。您無法建立用於加密和解密的橢圓曲線金鑰對。

```
$ aws kms create-key --key-spec ECC_NIST_P256 --key-usage KEY_AGREEMENT
{
  "KeyMetadata": {
    "AWSAccountId": "111122223333",
    "KeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
    "Arn": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321",
    "CreationDate": "2023-12-27T19:10:15.063000+00:00",
    "Enabled": true,
    "Description": "",
    "KeyUsage": "KEY_AGREEMENT",
    "KeyState": "Enabled",
    "Origin": "AWS_KMS",
    "KeyManager": "CUSTOMER",
    "CustomerMasterKeySpec": "ECC_NIST_P256",
    "KeySpec": "ECC_NIST_P256",
    "KeyAgreementAlgorithms": [
      "ECDH"
    ],
    "MultiRegion": false
  }
}
```

建立 HMAC KMS 金鑰

您可以在 AWS KMS 主控台中使用 [CreateKey](#) API 或使用 [AWS::KMS::Key CloudFormation](#)。

建立 HMAC KMS 金鑰時，您必須選取金鑰規格。AWS KMS 支援 [HMAC KMS 金鑰的多個金鑰規格](#)。金鑰規格可根據您的法規、安全或業務需求進行選取。一般而言，較長的金鑰更能抵禦暴力破解攻擊。

如需建立 KMS 金鑰所需之許可的詳細資訊，請參閱 [建立 KMS 金鑰的許可](#)。

使用 AWS KMS 主控台

您可以使用 AWS 管理主控台 來建立 HMAC KMS 金鑰。HMAC KMS 金鑰是對稱金鑰，其金鑰用途為 Generate and verify MAC (產生和驗證 MAC)。您也可以建立多區域 HMAC 金鑰。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。

3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
4. 選擇建立金鑰。
5. 針對 Key type (金鑰類型)，請選擇 Symmetric (對稱)。

HMAC KMS 金鑰為對稱金鑰。您可以使用相同的金鑰產生和驗證 HMAC 標籤。

6. 在 Key usage (金鑰用途) 欄位中，請選擇 Generate and verify MAC (產生和驗證 MAC)。

產生和驗證 MAC 是 HMAC KMS 金鑰的唯一有效金鑰用途。

 Note

僅當所選區域支援 HMAC KMS 金鑰時，才會顯示對稱金鑰的 Key usage (金鑰用途)。

7. 選取 HMAC KMS 金鑰的規格 (Key spec (金鑰規格))。

金鑰規格可根據您的法規、安全或業務需求進行選取。通常，較長的金鑰更安全。

8. 若要建立 [多區域](#) 主要 HMAC 金鑰，請在 Advanced options (進階選項) 中選擇 Multi-Region key (多區域金鑰)。您為此 KMS 金鑰定義的 [共用屬性](#) (如金鑰類型和金鑰用途) 將與其複本金鑰共享。

您無法使用此程序來建立複本金鑰。若要建立多區域複本 HMAC 金鑰，請遵循 [建立複本金鑰的指示](#)。

9. 選擇下一步。
10. 輸入 KMS 金鑰的 [別名](#)。別名名稱的開頭不可以是 **aws/**。**aws/** 字首由 Amazon Web Services 保留，以代表您帳戶中的 AWS 受管金鑰。

建議您使用能將 KMS 金鑰識別為 HMAC 金鑰的別名，例如 HMAC/test-key。這可讓您更輕鬆地在 AWS KMS 主控台中識別 HMAC 金鑰，您可以在其中依標籤和別名排序和篩選金鑰，但不能依金鑰規格或金鑰用量。

在 AWS 管理主控台中建立 KMS 金鑰時需要別名。使用 [CreateKey](#) 操作時，您不能指定別名，但您可以使用主控台或 [CreateAlias](#) 操作來建立現有 KMS 金鑰的別名。如需詳細資訊，請參閱 [中的別名 AWS KMS](#)。

11. (選用) 輸入 KMS 金鑰的描述。

輸入描述來說明您計劃保護的資料類型，或您計劃搭配 KMS 金鑰一起使用的應用程式。

您可以立即新增描述或在任意時間更新，除非 [金鑰狀態](#) 為 Pending Deletion 或 Pending Replica Deletion。若要新增、變更或刪除現有客戶受管金鑰的描述，請在 [中編輯中 KMS](#)

AWS 管理主控台 金鑰詳細資訊頁面上的描述，AWS 管理主控台 或使用 [UpdateKeyDescription](#) 操作。

12. (選用) 輸入標籤索引鍵和選用標籤值。若要將其他標籤新增至 KMS 金鑰，請選擇 Add tag (新增標籤)。

考慮新增能將金鑰識別為 HMAC 金鑰的標籤，例如 Type=HMAC。這可讓您更輕鬆地在 AWS KMS 主控台中識別 HMAC 金鑰，您可以在其中依標籤和別名排序和篩選金鑰，但不能依金鑰規格或金鑰用量。

當您將標籤新增至 AWS 資源時，AWS 會產生成本分配報告，其中包含依標籤彙總的用量和成本。標籤也可以用來控制 KMS 金鑰的存取。如需標記 KMS 金鑰的詳細資訊，請參閱 [中的標籤 AWS KMS](#) 和 [適用於的 ABAC AWS KMS](#)。

13. 選擇下一步。
14. 選取可管理 KMS 金鑰的 IAM 使用者和角色。

備註

此金鑰政策提供此 KMS 金鑰的完整 AWS 帳戶 控制權。它允許帳戶管理員使用 IAM 政策 授予其他主體管理 KMS 金鑰的許可。如需詳細資訊，請參閱 [the section called “預設金鑰政策”](#)。

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的安全性最佳實務](#)。

AWS KMS 主控台會將金鑰管理員新增至陳述式識別碼下的金鑰政策 "Allow access for Key Administrators"。修改此陳述式識別符可能會影響主控台顯示您對陳述式所做的更新。

15. (選用) 為了防止選取的 IAM 使用者和角色刪除此 KMS 金鑰，請在頁面底部的 Key deletion (金鑰刪除) 區段中，清除 Allow key administrators to delete this key (允許金鑰管理員刪除此金鑰) 核取方塊。
16. 選擇下一步。
17. 選取可將 KMS 金鑰用於 [密碼編譯操作](#) 的 IAM 使用者和角色。

備註

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的安全性最佳實務](#)。

AWS KMS 主控台會將金鑰使用者新增至陳述式識別符 "Allow use of the key" 和下的金鑰政策 "Allow attachment of persistent resources"。修改這些陳述式識別符可能會影響主控台顯示您對陳述式所做的更新的方式。

18. (選用) 您可以允許其他 AWS 帳戶使用此 KMS 金鑰進行密碼編譯操作。若要這樣做，請在頁面底部的其他 AWS 帳戶 區段中，選擇新增另一個 AWS 帳戶，然後輸入外部帳戶的 AWS 帳戶識別號碼。若要新增多個外部帳戶，請重複此步驟。

 Note

若要允許外部帳戶中的主體使用 KMS 金鑰，外部帳戶的管理員必須建立 IAM 政策來提供這些許可。如需詳細資訊，請參閱 [允許其他帳戶中的使用者使用 KMS 金鑰](#)。

19. 選擇下一步。
20. 檢閱金鑰的金鑰政策陳述式。若要變更金鑰政策，請選取編輯。
21. 選擇下一步。
22. 檢閱您選擇的金鑰設定。您仍然可以返回並變更所有設定。
23. 選擇 Finish (完成) 來建立 HMAC KMS 金鑰。

使用 AWS KMS API

您可以使用 [CreateKey](#) 操作來建立 HMAC KMS 金鑰。以下範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

建立 HMAC KMS 金鑰時，必須指定 KeySpec 參數，以決定 KMS 金鑰的類型。此外，您必須指定 GENERATE_VERIFY_MAC 的 KeyUsage 值，即使它是 HMAC 金鑰的唯一有效金鑰用途值。若要建立 [多區域](#) HMAC KMS 金鑰，請新增值為 true 的 MultiRegion 參數。建立 KMS 金鑰之後即無法變更這些屬性。

CreateKey 操作不允許您指定別名，但您可以使用 [CreateAlias](#) 操作來建立新 KMS 金鑰的別名。建議您使用能將 KMS 金鑰識別為 HMAC 金鑰的別名，例如 HMAC/test-key。這可讓您更輕鬆地在 AWS KMS 主控台中識別 HMAC 金鑰，您可以在其中依別名排序和篩選金鑰，但不能依金鑰規格或金鑰使用量。

如果您嘗試在不支援 HMAC 金鑰 AWS 區域 的中建立 HMAC KMS 金鑰，CreateKey 操作會傳回 UnsupportedOperationException

下列範例會使用 CreateKey 操作建立 512 位元的 HMAC KMS 金鑰。


```
$ aws kms create-key --key-spec HMAC_512 --key-usage GENERATE_VERIFY_MAC
{
  "KeyMetadata": {
    "KeyState": "Enabled",
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",
    "Description": "",
    "Arn": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "CreationDate": 1669973196.214,
    "MultiRegion": false,
    "KeySpec": "HMAC_512",
    "CustomerMasterKeySpec": "HMAC_512",
    "KeyUsage": "GENERATE_VERIFY_MAC",
    "MacAlgorithms": [
      "HMAC_SHA_512"
    ],
    "AWSAccountId": "111122223333",
    "Origin": "AWS_KMS",
    "Enabled": true
  }
}
```

建立多區域主索引鍵

您可以在 AWS KMS 主控台或使用 AWS KMS API 建立 [多區域主索引鍵](#)。您可以在 AWS KMS 支援多區域金鑰的任何 AWS 區域 中建立主金鑰。

若要建立多區域主要金鑰，委託人需要其建立任何 KMS 金鑰所需的相同 [許可](#)，包括 IAM 政策中的 [kms:CreateKey](#) 許可。委託人還需要 [iam:CreateServiceLinkedRole](#) 許可。您可以使用 [kms:MultiRegionKeyType](#) 條件索引鍵，以允許或拒絕建立多區域主要金鑰的許可。

Note

建立多區域主金鑰時，請仔細考慮您選取要管理和使用金鑰的 IAM 使用者和角色。IAM 政策可授權其他 IAM 使用者和角色來管理 KMS 金鑰。

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的安全性最佳實務](#)。

使用 AWS KMS 主控台

若要在 AWS KMS 主控台中建立多區域主金鑰，請使用您用來建立任何 KMS 金鑰的相同程序。您在 Advanced options (進階選項) 中選取多區域金鑰。如需完整說明，請參閱 [建立 KMS 金鑰](#)。

Important

請勿在別名、說明或標籤包含機密或敏感資訊。這些欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
4. 選擇建立金鑰。
5. 選取 [對稱或非對稱](#) 金鑰類型。對稱金鑰為預設值。

您可以建立多區域對稱和非對稱金鑰，包括多區域 HMAC KMS 金鑰，這些都是對稱金鑰。

6. 選擇您的金鑰用途。Encrypt and decrypt (加密和解密) 為預設值。

如需相關說明，請參閱 [建立 KMS 金鑰](#)、[the section called “建立非對稱 KMS 金鑰”](#) 或 [the section called “建立 HMAC KMS 金鑰”](#)。

7. 展開 Advanced options (進階選項)。
8. 在金鑰材料來源下，若要讓 AWS KMS 產生主要金鑰和複本金鑰將共用的金鑰材料，請選擇 KMS。若您要將 [金鑰材料匯入](#) 主要金鑰和複本金鑰中，請選擇 External (Import key material) (外部 (匯入金鑰材料))。
9. 在區域性下，選擇多區域金鑰。

您無法在建立 KMS 金鑰之後變更此設定。

10. 輸入主要金鑰的 [別名](#)。

別名不是多區域金鑰的共用屬性。您可以為多區域主要金鑰及其複本提供相同的別名或不同的別名。AWS KMS 不會同步多區域金鑰的別名。

Note

新增、刪除或更新別名可允許或拒絕 KMS 金鑰的許可。如需詳細資訊，請參閱 [適用於的 ABAC AWS KMS](#) 和 [使用別名來控制對 KMS 金鑰的存取](#)。

11. (選用) 輸入主要金鑰的描述。

描述不是多區域金鑰的共用屬性。您可以為多區域主要金鑰及其複本提供相同的描述或不同的描述。AWS KMS 不會同步多區域金鑰的金鑰描述。

12. (選用) 輸入標籤索引鍵和選用標籤值。若要為主要金鑰指派超過一個標籤，請選擇 Add tag (新增標籤)。

標籤不是多區域金鑰的共用屬性。您可以為多區域主要金鑰及其複本提供相同的標籤或不同的標籤。AWS KMS 不會同步多區域金鑰的標籤。您可以隨時變更 KMS 金鑰上的標籤。

Note

標記或取消標記 KMS 金鑰可以允許或拒絕 KMS 金鑰的許可。如需詳細資訊，請參閱 [適用於的 ABAC AWS KMS](#) 和 [使用標籤來控制對 KMS 金鑰的存取](#)。

13. 選取可管理主要金鑰的 IAM 使用者和角色。**備註**

- 這個步驟會開始為主要金鑰建立 [金鑰政策](#) 的程序。金鑰政策不是多區域金鑰的共用屬性。您可以為多區域主金鑰及其複本提供相同的金鑰政策或不同的金鑰政策。AWS KMS 不會同步多區域金鑰的金鑰政策。您可以隨時變更 KMS 金鑰的金鑰政策。
- 建立多區域主金鑰時，請考慮使用主控台產生的 [預設金鑰政策](#)。如果您修改此政策，主控台不會在建立複本金鑰時提供選取金鑰管理員和使用者的步驟，也不會新增對應的政策陳述式。因此，您需要手動新增這些項目。
- AWS KMS 主控台會將金鑰管理員新增至陳述式識別碼下的金鑰政策 "Allow access for Key Administrators"。修改此陳述式識別符可能會影響主控台顯示您對陳述式所做的更新的方式。

14. (選用) 為了防止選取的 IAM 使用者和角色刪除此 KMS 金鑰，請在頁面底部的 Key deletion (金鑰刪除) 區段中，清除 Allow key administrators to delete this key (允許金鑰管理員刪除此金鑰) 核取方塊。
15. 選擇下一步。
16. 選取可將 KMS 金鑰用於[密碼編譯操作](#)的 IAM 使用者和角色。

 備註

AWS KMS 主控台會將金鑰使用者新增至陳述式識別碼 "Allow use of the key" 和下的金鑰政策 "Allow attachment of persistent resources"。修改這些陳述式識別符可能會影響主控台顯示您對陳述式所做的更新。

17. (選用) 您可以允許其他 AWS 帳戶使用此 KMS 金鑰進行密碼編譯操作。若要這樣做，請在頁面底部的其他 AWS 帳戶 區段中，選擇新增另一個 AWS 帳戶，然後輸入外部帳戶的 AWS 帳戶識別號碼。若要新增多個外部帳戶，請重複此步驟。

 Note

若要允許外部帳戶中的主體使用 KMS 金鑰，外部帳戶的管理員必須建立 IAM 政策來提供這些許可。如需詳細資訊，請參閱[允許其他帳戶中的使用者使用 KMS 金鑰](#)。

18. 選擇下一步。
19. 檢閱金鑰的金鑰政策陳述式。若要變更金鑰政策，請選取編輯。
20. 選擇下一步。
21. 檢閱您選擇的金鑰設定。您仍然可以返回並變更所有設定。
22. 選擇完成以建立多區域主索引鍵。

使用 AWS KMS API

若要建立多區域主要金鑰，請使用 [CreateKey](#) 操作。使用值為 True 的 MultiRegion 參數。

例如，下列命令會在呼叫者的 AWS 區域 (us-east-1) 中建立多區域主索引鍵。它會接受所有其他屬性 (包括金鑰政策) 的預設值。多區域主要金鑰的預設值與所有其他 KMS 金鑰的預設值相同，包括[預設金鑰政策](#)。此程序會建立一個對稱加密金鑰，即預設 KMS 金鑰。

回應包含具有一般子元素的 MultiRegion 元素和 MultiRegionConfiguration 元素，以及無複本金鑰的多區域主要金鑰值。多區域金鑰的[金鑰 ID](#) 始終以 mrk- 開頭。

⚠ Important

請勿在 Description 或 Tags 欄位包含機密或敏感資訊。這些欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

```
$ aws kms create-key --multi-region
{
  "KeyMetadata": {
    "Origin": "AWS_KMS",
    "KeyId": "mrk-1234abcd12ab34cd56ef1234567890ab",
    "Description": "",
    "KeyManager": "CUSTOMER",
    "Enabled": true,
    "KeySpec": "SYMMETRIC_DEFAULT",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "KeyState": "Enabled",
    "CreationDate": 1606329032.475,
    "Arn": "arn:aws:kms:us-east-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
    "AWSAccountId": "111122223333",
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ],
    "MultiRegion": true,
    "MultiRegionConfiguration": {
      "MultiRegionKeyType": "PRIMARY",
      "PrimaryKey": {
        "Arn": "arn:aws:kms:us-east-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
        "Region": "us-east-1"
      },
      "ReplicaKeys": [ ]
    }
  }
}
```

建立多區域複本金鑰

您可以在 AWS KMS 主控台中使用 [ReplicateKey](#) 操作或使用 [AWS::KMS::ReplicaKey CloudFormation 範本](#) 來建立 [多區域複本金鑰](#)。您無法使用 [CreateKey](#) 操作來建立複本金鑰。

您可以使用這些程序複製任何多區域主要金鑰，包括[對稱加密 KMS 金鑰](#)、[非對稱 KMS 金鑰](#)或 [HMAC KMS 金鑰](#)。

當這項操作完成時，新的複本金鑰會有短暫的 [Creating 金鑰狀態](#)。當建立新複本金鑰的程序完成數秒後，此金鑰狀態會變更為 Enabled (或者 PendingImport 如果您建立具有 [匯入金鑰材料](#) 的多區域金鑰)。雖然金鑰狀態為 Creating，您可以管理金鑰，但還無法在密碼編譯操作中使用金鑰。如果您要以程式設計方式建立並使用複本金鑰，請在 `KMSInvalidStateException` 上重試或呼叫 [DescribeKey](#) 以檢查其 `KeyState` 值，然後再使用。

如果錯誤地刪除了複本金鑰，則可使用此程序重新建立。如果您在同一個區域中複寫相同的主要金鑰，則您建立的新複本金鑰將具有相同的 [共用屬性](#) 作為原始複本金鑰。

Important

請勿在別名、說明或標籤包含機密或敏感資訊。這些欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

若要使用 AWS CloudFormation 範本建立複本金鑰，請參閱 AWS CloudFormation 《使用者指南》中的 [AWS::KMS::ReplicaKey](#)。

步驟 1：選擇複本區域

您通常會 AWS 區域 根據您的商業模型和法規要求，選擇將多區域金鑰複寫到。例如，您可能會將金鑰複寫至您保留資源的區域。或者，為了符合災難復原要求，您可以將金鑰複寫至地理位置偏遠的區域。

以下是複本區域 AWS KMS 的需求。如果您選擇的區域不符合這些要求，則嘗試複寫金鑰會失敗。

- 每個區域一個相關的多區域金鑰 – 您無法在與其主要金鑰相同的區域中建立複本金鑰，或在與主要金鑰之另一個複本相同的區域中建立複本金鑰。

如果嘗試在已有該主要金鑰的區域中複寫主要金鑰，則嘗試會失敗。如果區域中目前的複本金鑰處於 [PendingDeletion 金鑰狀態](#)，您可以 [取消複本金鑰刪除](#)，或等到複本金鑰刪除完成。

- 同一區域中多個不相關的多個區域金鑰 – 您可以在同一區域中擁有多個不相關的多區域金鑰。例如，您可以在 us-east-1 區域中有兩個多區域主要金鑰。每個主要金鑰都可以在 us-west-2 區域中有複本金鑰。
- 相同分割區中的區域 – 複本金鑰區域必須位於與主要金鑰區域相同的 [AWS 分割區](#) 中。
- 必須啟用區域 – 如果區域[預設為停用](#)，則您無法在該區域中建立任何資源，直到為您的 AWS 帳戶啟用。

步驟 2：建立複本金鑰

Note

建立複本金鑰時，請仔細考慮您選取要管理和使用複本金鑰的 IAM 使用者和角色。IAM 政策可授權其他 IAM 使用者和角色來管理 KMS 金鑰。

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的安全性最佳實務](#)。

使用 AWS KMS 主控台

在 AWS KMS 主控台中，您可以在相同的操作中建立多區域主索引鍵的一或多個複本。

此程序類似於在主控台中建立標準的單一區域 KMS 金鑰。不過，因為複本金鑰是以主要金鑰為基礎，所以您不會選取[共用屬性](#)的值，例如金鑰規格 (對稱或非對稱)、金鑰使用情形或金鑰來源。

您可以指定不共用的屬性，包括別名、標籤、描述和金鑰政策。為了方便起見，主控台會顯示主要金鑰的目前屬性值，但您可以對其進行變更。即使您保持主索引鍵值，AWS KMS 也不會同步這些值。

Important

請勿在別名、說明或標籤包含機密或敏感資訊。這些欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。

4. 選取[多區域主要金鑰](#)的金鑰 ID 或別名。如此會開啟 KMS 金鑰的金鑰詳細資訊頁面。

若要識別多區域主要金鑰，請使用右上角的工具圖示新增 Regionality (區域性) 資料欄至資料表。

5. 選擇 Regionality (區域性) 索引標籤。
6. 在 Related multi-Region keys (相關的多區域金鑰) 區段中，選擇 Create new replica keys (建立新的複本金鑰)。

Related multi-Region keys (相關的多區域金鑰) 區段會顯示主要金鑰及其複本金鑰的區域。您可以使用此顯示來協助您為新複本金鑰選擇區域。

7. 選擇一或多個 AWS 區域。此操作程序會在您選取的每個區域中建立複本金鑰。

功能表僅包含與主索引鍵位於相同 AWS 分割區中的區域。已有相關多區域金鑰的區域會顯示出來，但無法選取。您可能沒有許可將金鑰複寫至選單上的所有區域。

完成選擇「區域」後，請關閉選單。隨即會顯示您選擇的區域。若要取消複寫至區域，請選擇區域名稱旁邊的 X。

8. 輸入複本金鑰的[別名](#)。

主控台會顯示主要金鑰目前的其中一個別名，但您可以對其進行變更。您可以為多區域主要金鑰及其複本提供相同的別名或不同的別名。別名不是多區域金鑰的[共用屬性](#)。AWS KMS 不會同步多區域金鑰的別名。

新增、刪除或更新別名可允許或拒絕 KMS 金鑰的許可。如需詳細資訊，請參閱 [適用於的 ABAC AWS KMS](#) 和 [使用別名來控制對 KMS 金鑰的存取](#)。

9. (選用) 輸入複本金鑰的描述。

主控台會顯示主要金鑰的當前描述，但您可以對其進行變更。描述不是多區域金鑰的共用屬性。您可以為多區域主要金鑰及其複本提供相同的描述或不同的描述。AWS KMS 不會同步多區域金鑰的金鑰描述。

10. (選用) 輸入標籤索引鍵和選用標籤值。若要為複本金鑰指派超過一個標籤，請選擇 Add tag (新增標籤)。

主控台會顯示目前連接至主要金鑰的標籤，但您可以對其進行變更。標籤不是多區域金鑰的共用屬性。您可以為多區域主金鑰及其複本提供相同的標籤或不同的標籤。AWS KMS 不會同步多區域金鑰的標籤。

標記或取消標記 KMS 金鑰可以允許或拒絕 KMS 金鑰的許可。如需詳細資訊，請參閱 [適用於的 ABAC AWS KMS](#) 和 [使用標籤來控制對 KMS 金鑰的存取](#)。

11. 選取可管理複本金鑰的 IAM 使用者和角色。

備註

- 如果您在建立多區域主要金鑰時修改預設金鑰政策，主控台不會在建立複本金鑰期間提示您選取金鑰管理員或金鑰使用者（步驟 11-15）。在此情況下，您需要手動將金鑰管理員和使用者的必要許可新增至金鑰政策，方法是在編輯金鑰政策步驟（步驟 17）中選取編輯。
- 這個步驟會開始為複本金鑰建立[金鑰政策](#)的程序。主控台會顯示主要金鑰的當前金鑰政策，但您可以對其進行變更。金鑰政策不是多區域金鑰的共用屬性。您可以為多區域主要金鑰及其複本提供相同的金鑰政策或不同的金鑰政策。AWS KMS 不會同步金鑰政策。您可以隨時變更任何 KMS 金鑰的金鑰政策。
- AWS KMS 主控台會將金鑰管理員新增至陳述式識別碼下的金鑰政策"Allow access for Key Administrators"。修改此陳述式識別符可能會影響主控台顯示您對陳述式所做的更新。

12. (選用) 為了防止選取的 IAM 使用者和角色刪除此 KMS 金鑰，請在頁面底部的 Key deletion (金鑰刪除) 區段中，清除 Allow key administrators to delete this key (允許金鑰管理員刪除此金鑰) 核取方塊。

13. 選擇下一步。

14. 選取可將 KMS 金鑰用於[密碼編譯操作](#)的 IAM 使用者和角色。

注意

AWS KMS 主控台會將金鑰使用者新增至陳述式識別符 "Allow use of the key" 和下的金鑰政策 "Allow attachment of persistent resources"。修改這些陳述式識別符可能會影響主控台顯示您對陳述式所做的更新的方式。

15. (選用) 您可以允許其他 AWS 帳戶使用此 KMS 金鑰進行密碼編譯操作。若要這樣做，請在頁面底部的其他 AWS 帳戶 區段中，選擇新增另一個 AWS 帳戶，然後輸入外部帳戶的 AWS 帳戶識別號碼。若要新增多個外部帳戶，請重複此步驟。

Note

若要允許外部帳戶中的主體使用 KMS 金鑰，外部帳戶的管理員必須建立 IAM 政策來提供這些許可。如需詳細資訊，請參閱[允許其他帳戶中的使用者使用 KMS 金鑰](#)。

16. 選擇下一步。
17. 檢閱金鑰的金鑰政策陳述式。若要變更金鑰政策，請選取編輯。
18. 選擇下一步。
19. 檢閱您選擇的金鑰設定。您仍然可以返回並變更所有設定。
20. 選擇完成以建立多區域複本金鑰。

使用 AWS KMS API

若要建立多區域複本金鑰，請使用 [ReplicateKey](#) 操作。您無法使用 [CreateKey](#) 操作來建立複本金鑰。此操作一次會建立一個複本金鑰。您指定的區域必須符合複本金鑰的[區域要求](#)。

當使用 [ReplicateKey](#) 操作時，您不需要為多區域金鑰的任何[共用屬性](#)指定值。共用屬性值會從主要金鑰複製並保持同步。然而，您可以為不共用的屬性指定值。否則，會 AWS KMS 套用 KMS 金鑰的標準預設值，而非主金鑰的值。

Note

如果您未指定 `Description`、`KeyPolicy` 或 `Tags` 參數的值，會 AWS KMS 建立具有空字串描述、[預設金鑰政策且沒有標籤的複本金鑰](#)。

請勿在 `Description` 或 `Tags` 欄位包含機密或敏感資訊。這些欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

例如，以下命令會在亞太區域 (雪梨) 區域 (ap-southeast-2) 建立多區域複本金鑰。此複本金鑰是基於美國東部 (維吉尼亞北部) 區域 (us-east-1) 的主要金鑰進行建模，由 `KeyId` 參數值識別。此範例會接受所有其他屬性 (包括金鑰政策) 的預設值。

回應會描述新的複本金鑰。它包含共用屬性的欄位，例如 `KeyId`、`KeySpec`、`KeyUsage`，以及金鑰材料來源 (`Origin`)。它也包含獨立於主要金鑰的屬性，例如 `Description`、金鑰政策 (`ReplicaKeyPolicy`) 和標籤 (`ReplicaTags`)。

回應還包括金鑰 ARN 和主要金鑰的區域及其所有複本金鑰，包括剛剛在 ap-southeast-2 區域中建立的金鑰。在此範例中，`ReplicaKey` 元素顯示此主要金鑰已在歐洲 (愛爾蘭) 區域 (eu-west-1) 中複寫。

```
$ aws kms replicate-key \
  --key-id arn:aws:kms:us-east-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab \
  --replica-region ap-southeast-2
{
```

```

    "ReplicaKeyMetadata": {
      "MultiRegion": true,
      "MultiRegionConfiguration": {
        "MultiRegionKeyType": "REPLICA",
        "PrimaryKey": {
          "Arn": "arn:aws:kms:us-east-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
          "Region": "us-east-1"
        },
        "ReplicaKeys": [
          {
            "Arn": "arn:aws:kms:ap-southeast-2:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
            "Region": "ap-southeast-2"
          },
          {
            "Arn": "arn:aws:kms:eu-west-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
            "Region": "eu-west-1"
          }
        ]
      },
      "AWSAccountId": "111122223333",
      "Arn": "arn:aws:kms:ap-southeast-2:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
      "CreationDate": 1607472987.918,
      "Description": "",
      "Enabled": true,
      "KeyId": "mrk-1234abcd12ab34cd56ef1234567890ab",
      "KeyManager": "CUSTOMER",
      "KeySpec": "SYMMETRIC_DEFAULT",
      "KeyState": "Enabled",
      "KeyUsage": "ENCRYPT_DECRYPT",
      "Origin": "AWS_KMS",
      "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
      "EncryptionAlgorithms": [
        "SYMMETRIC_DEFAULT"
      ]
    },
    "ReplicaKeyPolicy": "{\n  \"Version\" : \"2012-10-17\",\n  \"Id\" : \"key-
default-1\",...,
    \"ReplicaTags\": []
  }

```

使用匯入的金鑰材料建立 KMS 金鑰

匯入的金鑰材料可讓您使用產生的密碼編譯金鑰來保護 AWS 資源。以下概觀說明如何將金鑰材料匯入 AWS KMS。如需程序中每個步驟的詳細資訊，請參閱對應的主題。

1. [建立不含金鑰資料的 KMS 金鑰](#) — 來源必須是 EXTERNAL。金鑰來源 EXTERNAL 表示金鑰專為匯入的金鑰材料而設計，並 AWS KMS 防止產生 KMS 金鑰的金鑰材料。在後續步驟中，您會將您自己的金鑰材料匯入到這個 KMS 金鑰中。

您匯入的金鑰材料必須與相關聯金鑰的 AWS KMS 金鑰規格相容。如需相容性的詳細資訊，請參閱 [the section called “匯入金鑰資料需求”](#)。

2. [下載包裝公有金鑰及匯入字符](#) — 在完成步驟 1 之後，請下載包裝公有金鑰及匯入字符。這些項目會在匯入至時保護您的金鑰材料 AWS KMS。

您可在此步驟選擇 RSA 包裝金鑰的類型（「金鑰規格」），以及用來加密資料的包裝演算法，以便將其傳輸至 AWS KMS。每次匯入或重新匯入相同金鑰資料時，您都可選擇不同包裝金鑰規格與包裝金鑰演算法。

3. [加密金鑰資料](#) — 利用您在步驟 2 下載的包裝公有金鑰來加密您在自己系統建立的金鑰資料。
4. [匯入金鑰材料](#) — 上傳您在步驟 3 建立的加密金鑰材料，以及您在步驟 2 下載的匯入字符。

在此階段，您可以[設定選用的到期時間](#)。當匯入的金鑰材料過期時，會將其 AWS KMS 刪除，KMS 金鑰會變成無法使用。若要繼續運用 KMS 金鑰，您必須重新匯入相同的金鑰資料。

當匯入操作成功完成後，KMS 金鑰的金鑰狀態會從 PendingImport 變成 Enabled。您現在可以在密碼編譯操作中使用 KMS 金鑰。

AWS KMS 當您[建立 KMS 金鑰](#)、[下載包裝公有金鑰和匯入字符](#)，以及匯入金鑰材料時，會在 AWS CloudTrail 日誌中記錄項目。當您刪除匯入的金鑰材料或 AWS KMS [刪除過期的金鑰材料](#)時，AWS KMS 也會記錄項目。 [???](#)

匯入金鑰材料的許可

若要使用匯入的金鑰材料建立和管理 KMS 金鑰，使用者需要此程序中的操作許可。您可以在建立 KMS 金鑰時提供金鑰政策中的 kms:GetParametersForImport、kms:ImportKeyMaterial 和 kms>DeleteImportedKeyMaterial 許可。在 AWS KMS 主控台中，當您使用外部金鑰材料來源建立金鑰時，系統會自動為金鑰管理員新增這些許可。

若要建立具有匯入金鑰材料的 KMS 金鑰，主體需要下列許可。

- [kms:CreateKey](#) (IAM 政策)
 - 若要將此許可限制至具有匯入金鑰材料的 KMS 金鑰，請使用具有 EXTERNAL 值的 [kms:KeyOrigin](#) 政策條件。

```
{
  "Sid": "CreateKMSKeysWithoutKeyMaterial",
  "Effect": "Allow",
  "Resource": "*",
  "Action": "kms:CreateKey",
  "Condition": {
    "StringEquals": {
      "kms:KeyOrigin": "EXTERNAL"
    }
  }
}
```

- [kms:GetParametersForImport](#) (金鑰政策或 IAM 政策)
 - 若要將此許可限制至使用特定包裝演算法和包裝金鑰規格的請求，請使用 [kms:WrappingAlgorithm](#) 和 [kms:WrappingKeySpec](#) 政策條件。
- [kms:ImportKeyMaterial](#) (金鑰政策或 IAM 政策)
 - 若要允許或禁止過期的金鑰材料並控制過期日期，請使用 [kms:ExpirationModel](#) 和 [kms:ValidTo](#) 政策條件。

若要重新匯入匯入的金鑰材料，主體需要 [kms:GetParametersForImport](#) 和 [kms:ImportKeyMaterial](#) 許可。

若要刪除匯入的金鑰材料，主體需要 [kms:DeleteImportedKeyMaterial](#) 許可。

例如，若要准許範例 KMSAdminRole 管理具有匯入金鑰資料的 KMS 金鑰的所有方面，請在 KMS 金鑰的金鑰政策中包含如下所示的金鑰政策陳述式。

```
{
  "Sid": "Manage KMS keys with imported key material",
  "Effect": "Allow",
  "Resource": "*",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/KMSAdminRole"
  },
  "Action": [
    "kms:GetParametersForImport",
```

```
"kms:ImportKeyMaterial",
"kms>DeleteImportedKeyMaterial"
]
}
```

匯入金鑰資料需求

您匯入的金鑰資料必須與關聯 KMS 金鑰的[金鑰規格](#)相容。對於非對稱金鑰對，請僅匯入配對的私有金鑰。會從私有金鑰 AWS KMS 衍生公有金鑰。

AWS KMS 支援具有匯入金鑰材料的 KMS 金鑰的下列金鑰規格。

- 對稱加密金鑰
 - 金鑰規格：
 - SYMMETRIC_DEFAULT。
 - 使用要求：
 - 256 位元 (32 位元組) 的二進位資料。
 - 在中國區域，必須是 128 位元 (16 位元組) 的二進位資料。
- HMAC 金鑰
 - 金鑰規格：
 - HMAC_224
 - HMAC_256
 - HMAC_384
 - HMAC_512
 - 使用要求：
 - HMAC 金鑰資料必須符合 [RFC 2104](#)。
 - 金鑰長度必須至少與金鑰規格指定的長度相同。金鑰長度上限為 1024 位元。
 - 如果您的金鑰材料超過 1024 位元，您可以雜湊金鑰材料並匯入雜湊輸出。雜湊演算法必須符合您正在建立之 HMAC KMS 金鑰的金鑰規格。
 - 範例：
 - 若要將 2048 位元的金鑰材料匯入 HMAC_256 金鑰，請先計算 2048 位元金鑰材料的 SHA-256 雜湊，然後將產生的 256 位元雜湊輸出匯入 KMS 金鑰。
 - 有效的金鑰長度：
 - HMAC_224：224–1024 位元

- HMAC_256 : 256–1024 位元
- HMAC_384 : 384–1024 位元
- HMAC_512 : 512–1024 位元
- RSA 非對稱私有金鑰
 - 金鑰規格：
 - RSA_2048
 - RSA_3072
 - RSA_4096
 - 使用要求：
 - 您匯入的 RSA 非對稱私有金鑰必須是符合 [RFC 3447](#) 金鑰對的一部分。
 - 模數：2048 位元、3072 位元或 4096 位元
 - 素數：2 (不支持多素 RSA 金鑰)
 - 非對稱金鑰材料必須使用符合 [RFC 5208](#) 的公有金鑰密碼編譯標準 (PKCS) #8 格式的 BER 編碼或 DER 編碼。
- 橢圓曲線非對稱私有金鑰
 - 金鑰規格：
 - ECC_NIST_P256 (secp256r1)
 - ECC_NIST_P384 (secp384r1)
 - ECC_NIST_P521 (secp521r1)
 - ECC_SECG_P256K1 (secp256k1)
 - ECC_NIST_EDWARDS25519 (ed25519)
 - 使用要求：
 - 您匯入的 ECC 非對稱私有金鑰必須是符合 [RFC 5915](#) 的一種金鑰對。
 - 曲線：NIST P-256、NIST P-384、NIST P-521, Secp256k1、NIST Ed25519。
 - 參數：僅限具名曲線 (具有明確參數的 ECC 金鑰會遭到拒絕) 。
 - 公有點座標：可壓縮、未壓縮或投影。
 - 非對稱金鑰材料必須以符合 [RFC 5208](#) 的公有金鑰密碼編譯標準 (PKCS) #8 格式進行 BER 編碼或 DER 編碼。
- ML-DSA 金鑰
 - 金鑰規格：
 - ML_DSA_44

- ML_DSA_65
- ML_DSA_87

Important

不支援匯入 ML-DSA 金鑰。

- SM2 非對稱私有金鑰（僅限中國區域）
- 使用要求：
 - 您匯入的 SM2 非對稱私有金鑰必須是符合 GM/T 0003 之金鑰對的一部分。
 - 曲線：SM2。
 - 參數：僅限具名曲線（具有明確參數的 SM2 金鑰會遭到拒絕）。
 - 公有點座標：可壓縮、未壓縮或投影。
 - 非對稱金鑰材料必須使用符合 [RFC 5208](#) 的公有金鑰密碼編譯標準 (PKCS) #8 格式的 BER 編碼或 DER 編碼。

步驟 1：建立 AWS KMS key 不含金鑰材料的

根據預設，會在您 AWS KMS 建立 KMS 金鑰時為您建立金鑰材料。若是要匯入您自己的金鑰材料，請從建立一個不含金鑰材料的 KMS 金鑰開始。然後再匯入金鑰材料。若要建立不含金鑰材料的 KMS 金鑰，請使用 AWS KMS 主控台或 [CreateKey](#) 操作。

若要建立不含金鑰資料的金鑰，請指定 EXTERNAL 的 [來源](#)。KMS 金鑰的來源屬性不可變。建立 KMS 金鑰之後，您就無法將專為匯入金鑰材料設計的 KMS 金鑰轉換為 KMS 金鑰，其中包含來自 AWS KMS 或任何其他來源的金鑰材料。

具有 EXTERNAL 來源且沒有金鑰材料的 KMS 金鑰的 [金鑰狀態](#) 為 PendingImport。KMS 金鑰可以無限期保持在 PendingImport 狀態。然而，您無法在密碼編譯操作採用 PendingImport 狀態的 KMS 金鑰。在匯入金鑰資料之後，KMS 金鑰的金鑰狀態會變為 Enabled，且您可在密碼編譯操作使用該 KMS 金鑰。

AWS KMS 當您 [建立 KMS 金鑰](#)、[下載公有金鑰和匯入字符](#)，以及 [匯入金鑰材料](#) 時，會在 AWS CloudTrail 日誌中記錄事件。AWS KMS 當您 [刪除匯入的金鑰材料](#) 或 AWS KMS [刪除過期的金鑰材料](#) 時，也會記錄 CloudTrail 事件。

主題

- [建立不含金鑰材料的 KMS 金鑰 \(主控台\)](#)

- [建立不含金鑰材料 \(AWS KMS API\) 的 KMS 金鑰](#)

建立不含金鑰材料的 KMS 金鑰 (主控台)

您只需為匯入的金鑰資料建立一次 KMS 金鑰。您可視需要無限次匯入及重新匯入相同金鑰資料至現有 KMS 金鑰，但您不能匯入不同金鑰資料至一個 KMS 金鑰。如需詳細資訊，請參閱[步驟 2：下載包裝公有金鑰及匯入字符](#)。

若要在 Customer managed keys (客戶受管金鑰) 資料表中尋找具有匯入金鑰材料的現有 KMS 金鑰，請使用右上角的齒輪圖示在 KMS 金鑰清單中顯示 Origin (來源) 資料欄。匯入金鑰的來源值為外部 (匯入金鑰資料)。

若要使用匯入的金鑰材料建立 KMS 金鑰，請先遵循[建立您慣用金鑰類型的 KMS 金鑰的指示](#)，但有下列例外。

在選擇金鑰用途之後，請執行下列步驟：

1. 展開 Advanced options (進階選項)。
2. 對於 Key material origin (金鑰材料來源)，選擇 External (Import key material) (外部 (匯入金鑰材料))。
3. 然後選擇我了解採用已匯入金鑰的安全性及持久性隱憂旁邊的核取方塊，表示您了解使用匯入金鑰資料的隱憂。如要閱讀這些隱憂的相關資訊，請參閱 [保護匯入的金鑰資料](#)。
4. 選用：若要使用匯入的[金鑰材料建立多區域 KMS](#) 金鑰，請在區域性下選取多區域金鑰。
5. 返回基本說明。對於該類型的所有 KMS 金鑰，基本程序的其餘步驟均相同。

當您選擇完成時，您已建立不含金鑰資料的 KMS 金鑰，且狀態 ([金鑰狀態](#)) 為待匯入。

然而，主控台不會返回客戶自管金鑰表，而是顯示頁面讓您可在其中下載匯入金鑰資料所需的公有金鑰與匯入字符。您可立即繼續執行下載步驟，或選擇取消並在此時停止。您可隨時返回此下載步驟。

下一步：[步驟 2：下載包裝公有金鑰及匯入字符](#)。

建立不含金鑰材料 (AWS KMS API) 的 KMS 金鑰

若要使用 [AWS KMS API](#) 建立不含金鑰資料的對稱加密 KMS 金鑰，請傳送 [CreateKey](#) 請求並將 Origin 參數設為 EXTERNAL。以下範例顯示如何使用 [AWS Command Line Interface \(AWS CLI\)](#) 執行此作業。


```
$ aws kms create-key --origin EXTERNAL
```

如果命令成功執行，您會看到類似如下的輸出。AWS KMS 金鑰的 Origin 是 EXTERNAL，其 KeyState 是 PendingImport。

 Tip

如果命令未成功，您可能看到 `KMSInvalidStateException` 或 `NotFoundException`。您可重試請求。

```
{
  "KeyMetadata": {
    "Origin": "EXTERNAL",
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "Description": "",
    "Enabled": false,
    "MultiRegion": false,
    "KeyUsage": "ENCRYPT_DECRYPT",
    "KeyState": "PendingImport",
    "CreationDate": 1568289600.0,
    "Arn": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "AWSAccountId": "111122223333",
    "KeyManager": "CUSTOMER",
    "KeySpec": "SYMMETRIC_DEFAULT",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ]
  }
}
```

從命令輸出複製 KeyId 值以供後續步驟使用，然後繼續進行 [步驟 2：下載包裝公有金鑰及匯入字符](#)。

 Note

此命令會利用 SYMMETRIC_DEFAULT 的 KeySpec 與 ENCRYPT_DECRYPT 的 KeyUsage 來建立對稱加密 KMS 金鑰。您可利用選用參數 `--key-spec` 與 `--key-usage` 來建立非對稱 HMAC KMS 金鑰。如需詳細資訊，請參閱 [CreateKey](#) 操作。

步驟 2：下載包裝公有金鑰及匯入字符

[建立 AWS KMS key 不含金鑰資料的](#) 後，請使用 AWS KMS 主控台或 [GetParametersForImport](#) API，下載該 KMS 金鑰的包裝公有金鑰和匯入字符。包裝公有金鑰與匯入字符是不可分割的集合，必須一起使用。

您將利用包裝公有金鑰來[加密您的金鑰資料](#)，以便進行傳輸。在下載 RSA 包裝金鑰對之前，您可以選取 RSA 包裝金鑰對的長度（金鑰規格），以及用於在[步驟 3](#) 中加密匯入金鑰材料以進行傳輸的包裝演算法。AWS KMS 也支援 SM2 包裝金鑰規格（僅限中國區域）。

每個包裝公有金鑰與匯入字符集合的有效期為 24 小時。如您未在下載後 24 小時內利用它們來匯入金鑰資料，則必須下載新的集合。您可隨時下載新的包裝公有金鑰與匯入字符集合。這可讓您更改 RSA 包裝金鑰長度（「金鑰規格」）或替取代遺失的集合。

您也可下載包裝公有金鑰與匯入字符集合，以便[重新匯入相同金鑰資料](#)至 KMS 金鑰。這樣做可設定或變更金鑰資料的過期時間，或者還原已過期或已刪除的金鑰資料。每次匯入金鑰材料時，都必須下載並重新加密。AWS KMS

運用包裝公有金鑰

下載包含專屬於您的公有金鑰 AWS 帳戶，也稱為包裝公有金鑰。

匯入金鑰材料之前，您可以使用公有包裝金鑰加密金鑰材料，然後將加密的金鑰材料上傳至其中 AWS KMS。當 AWS KMS 收到您的加密金鑰材料時，它會使用對應的私有金鑰解密金鑰材料，然後在 AES 對稱金鑰下重新加密金鑰材料，全部都在 AWS KMS 硬體安全模組 (HSM) 內。

使用匯入符記

該下載包括匯入字符，字符所帶的中繼資料可確保金鑰材料正確匯入。當您上傳加密的金鑰資料到時 AWS KMS，您必須上傳您在此步驟中下載的相同匯入字符。

選取包裝公有金鑰規格

為了在匯入期間保護您的金鑰材料，您可以使用您下載的包裝公有金鑰 AWS KMS 和支援的[包裝演算法](#)來加密它。在下載包裝公有金鑰與匯入字符之前，請先選取金鑰規格。所有包裝金鑰對都會在 AWS KMS 硬體安全模組 (HSMs) 中產生。您的私有金鑰永遠不會以純文字形式離開 HSM。

RSA 包裝金鑰規格

包裝公有金鑰的金鑰規格決定 RSA 金鑰對的金鑰長度，該金鑰對在傳輸至 AWS KMS 的過程會保護您的金鑰資料。一般情況而言，建議採用符合實用的最長公有金鑰。我們提供數種包裝公有金鑰規格，以便支援各種 HSM 與金鑰管理器。

AWS KMS 支援用於匯入所有類型金鑰材料的 RSA 包裝金鑰的下列金鑰規格，除非另有說明。

- RSA_4096 (建議)
- RSA_3072
- RSA_2048

Note

不支援下列組合：ECC_NIST_P521 金鑰資料、RSA_2048 公有包裝金鑰規格以及 RSAES_OAE_SHA_* 包裝演算法。

您無法使用 RSA_2048 公有包裝金鑰直接包裝 ECC_NIST_P521 金鑰材質。使用較大的包裝金鑰或 RSA_AES_KEY_WRAP_SHA_* 包裝演算法。

SM2 包裝金鑰規格 (僅限中國區域)

AWS KMS 支援用於匯入非對稱金鑰材料的 SM2 包裝金鑰的下列金鑰規格。

- SM2

選取包裝演算法

為在匯入期間保護金鑰資料，請利用下載的包裝公有金鑰及支援的包裝演算法來加密金鑰資料。

AWS KMS 支援數個標準 RSA 包裝演算法和兩個步驟的混合包裝演算法。一般而言，建議採用相容匯入金鑰資料及[包裝金鑰規格](#)的最安全包裝演算法。您通常是選擇硬體安全模組 (HSM) 或保護金鑰資料的金鑰管理系統所支援的演算法。

下表顯示每種金鑰資料與 KMS 金鑰類型所支援的包裝演算法。演算法以偏好順序列出。

金鑰資料	支援的包裝演算法與規格
對稱加密金鑰	包裝演算法：
256 位元 AES 金鑰	RSAES_OAEP_SHA_256

金鑰資料	支援的包裝演算法與規格
128 位元 SM4 金鑰 (僅限中國區域)	RSAES_OAEP_SHA_1 已取代的包裝演算法： RSAES_PKCS1_V1  Note 截至 2023 年 10 月 10 日，AWS KMS 不支援 RSAES_PKCS1_V1_5 包裝演算法。 包裝金鑰規格： RSA_2048 RSA_3072 RSA_4096
非對稱 RSA 私有金鑰	包裝演算法： RSA_AES_KEY_WRAP_SHA_256 RSA_AES_KEY_WRAP_SHA_1 SM2PKE (僅限中國區域) 包裝金鑰規格： RSA_2048 RSA_3072 RSA_4096 SM2 (僅限中國區域)

金鑰資料	支援的包裝演算法與規格
非對稱橢圓曲線 (ECC) 私有金鑰 您不能搭配採用 RSAES_OAEP_SHA_* 包裝演算法與 RSA_2048 包裝金鑰規格來包裝 ECC_NIST_P521 金鑰資料。	包裝演算法： RSA_AES_KEY_WRAP_SHA_256 RSA_AES_KEY_WRAP_SHA_1 RSAES_OAEP_SHA_256 RSAES_OAEP_SHA_1 SM2PKE (僅限中國區域) 包裝金鑰規格： RSA_2048 RSA_3072 RSA_4096 SM2 (僅限中國區域)
非對稱 SM2 私有金鑰 (僅限中國區域)	包裝演算法： RSAES_OAEP_SHA_256 RSAES_OAEP_SHA_1 SM2PKE (僅限中國區域) 包裝金鑰規格： RSA_2048 RSA_3072 RSA_4096 SM2 (僅限中國區域)

金鑰資料	支援的包裝演算法與規格
HMAC 金鑰	包裝演算法： RSAES_OAEP_SHA_256 RSAES_OAEP_SHA_1 包裝金鑰規格： RSA_2048 RSA_3072 RSA_4096

Note

中國區域不支援 RSA_AES_KEY_WRAP_SHA_256 和 RSA_AES_KEY_WRAP_SHA_1 包裝演算法。

- RSA_AES_KEY_WRAP_SHA_256 - 此為兩步驟混合包裝演算法，結合加密金鑰資料以及您產生的 AES 對稱金鑰，然後利用下載的 RSA 公有包裝金鑰與 RSAES_OAEP_SHA_256 包裝演算法來加密 AES 對稱金鑰。

除了您必須使用 RSA_AES_KEY_WRAP_SHA_* 包裝演算法的中國區域以外，包裝 RSA 私有金鑰材料需要 SM2PKE 包裝演算法。

- RSA_AES_KEY_WRAP_SHA_1 - 此為兩步驟混合包裝演算法，結合加密金鑰資料以及您產生的 AES 對稱金鑰，然後利用下載的 RSA 包裝公有金鑰與 RSAES_OAEP_SHA_1 包裝演算法來加密 AES 對稱金鑰。

除了您必須使用 RSA_AES_KEY_WRAP_SHA_* 包裝演算法的中國區域以外，包裝 RSA 私有金鑰材料需要 SM2PKE 包裝演算法。

- RSAES_OAEP_SHA_256 – RSA 加密演算法搭配最佳非對稱加密填補 (OAEP) 以及 SHA-256 雜湊函數。
- RSAES_OAEP_SHA_1 – RSA 加密演算法搭配最佳非對稱加密填補 (OAEP) 以及 SHA-1 雜湊函數。

- RSAES_PKCS1_V1_5 (已棄用；截至 2023 年 10 月 10 日，AWS KMS 不支援 RSAES_PKCS1_V1_5 包裝演算法) – RSA 加密演算法搭配 PKCS #1 版本 1.5 中定義的填補格式。
- SM2PKE (僅限中國區域) – OSCCA 在 GM/T 0003.4-2012 中定義的橢圓曲線型加密演算法。

主題

- [下載包裝公有金鑰與匯入字符 \(主控台\)](#)
- [下載包裝公有金鑰和匯入字符 \(AWS KMS API\)](#)

下載包裝公有金鑰與匯入字符 (主控台)

您可以使用 AWS KMS 主控台下載包裝公有金鑰和匯入字符。

1. 如果您剛完成[建立不含金鑰材料之 KMS 金鑰](#)的步驟，且目前是在 Download wrapping key and import token (下載包裝金鑰和匯入字符) 頁面，請跳到 [Step 10](#)。
2. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
3. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
4. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。

Tip

您僅能匯入金鑰資料至來源是外部 (匯入金鑰資料) 的 KMS 金鑰。這表示建立的 KMS 金鑰不含金鑰資料。如要將 Origin (來源) 資料行新增至您的資料表，請在頁面的右上角選擇設定圖示



開啟 Origin (來源)，然後選擇 Confirm (確認)。

5. 選擇待匯入的 KMS 金鑰的別名或金鑰 ID。
6. 選擇 Cryptographic configuration (密碼編譯組態) 索引標籤並檢視其值。索引標籤位於 General Configuration (一般組態) 區段下。

您僅能匯入金鑰資料至來源是外部 (匯入金鑰資料) 的 KMS 金鑰。如需有關建立具有匯入金鑰材料之 KMS 金鑰的資訊，請參閱 [匯入金鑰的 AWS KMS 金鑰材料](#)。

7. 根據您的金鑰類型選擇適當的標籤。

- 對於非對稱和 HMAC 金鑰，請選擇金鑰材料索引標籤。
- 針對對稱加密金鑰，選擇金鑰材料和輪換標籤。

8. 選擇匯入動作。

- 針對非對稱和 HMAC 金鑰，選擇匯入金鑰材料。
- 針對對稱加密金鑰，選擇下列其中一項：
 - 匯入初始金鑰材料（如果尚未匯入金鑰材料）
 - 匯入新的金鑰材料（新增要輪換的新材料）
 - 重新匯入金鑰材料（可從金鑰材料資料表中的動作功能表取得）

 Note

對於多區域金鑰，您必須先將新的金鑰材料匯入主要區域金鑰。然後，將相同的金鑰材料匯入每個複本區域金鑰。

對於主要多區域金鑰，金鑰資料表包含複本匯入狀態欄，顯示所有複本區域的匯入狀態（例如，匯入 3 的 0）。選擇複本匯入狀態值，以開啟顯示每個複本區域的匯入狀態的模態。模態為尚未匯入新金鑰材料的複本區域提供匯入金鑰材料連結。

9. 針對選取包裝金鑰規格，請選擇 KMS 金鑰的組態。在建立此金鑰後，您便無法變更金鑰規格。
10. 對於 Select wrapping algorithm (選取包裝演算法)，請選擇您將用來加密金鑰材料的選項。如需選項的詳細資訊，請參閱[選取包裝演算法](#)。
11. 選擇下載包裝公有金鑰與匯入字符，然後儲存檔案。

如果您有 Next (下一步) 選項，現在請繼續程序，選擇 Next (下一步)。若要稍後再繼續，請選擇 Cancel (取消)。

12. 解壓縮您在上一個步驟中儲存的 .zip 檔案
(Import_Parameters_<key_id>_<timestamp>)。

資料夾內含下列檔案：

- 在名為 的檔案中包裝公有金鑰WrappingPublicKey.bin。
- 匯入字符位於名為 ImportToken.bin 的檔案。
- 名為 README.txt 的文本檔案。此檔案包含包裝公有金鑰的相關資訊、用來加密金鑰資料的包裝演算法，以及包裝公有金鑰與匯入字符過期的日期及時間。

13. 若要繼續程序，請參閱[加密金鑰材料](#)。

下載包裝公有金鑰和匯入字符 (AWS KMS API)

若要下載公有金鑰和匯入字符，請使用 [GetParametersForImport](#) API。指定將與匯入金鑰材料相關聯的 KMS 金鑰。此 KMS 金鑰的 [Origin](#) (來源) 值必須為 EXTERNAL。

Note

您無法匯入 ML-DSA KMS 金鑰的金鑰材料。

此範例指定 RSA_AES_KEY_WRAP_SHA_256 包裝演算法、RSA_3072 包裝公有金鑰規格，以及金鑰 ID 範例。將這些範例值取代為下載的有效值。對於金鑰 ID，您可採用 [金鑰 ID](#) 或 [金鑰 ARN](#)，但您不能在此操作採用 [別名名稱](#) 或 [別名 ARN](#)。

```
$ aws kms get-parameters-for-import \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --wrapping-algorithm RSA_AES_KEY_WRAP_SHA_256 \
  --wrapping-key-spec RSA_3072
```

如果命令成功執行，您會看到類似如下的輸出：

```
{
  "ParametersValidTo": 1568290320.0,
  "PublicKey": "public key (base64 encoded)",
  "KeyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "ImportToken": "import token (base64 encoded)"
}
```

為了準備下一步要使用的資料，將對公有金鑰和匯入字符進行 base64 解碼，然後將解碼的值保存在檔案中。

若要對公有金鑰和匯入字符進行 base64 解碼：

1. 複製 base64 編碼公有金鑰 (在範例輸出中表示為 `#### (base64 ##)`)，將其貼入新檔案，然後儲存檔案。以描述性名稱命名檔案，例如 PublicKey.b64。
2. 使用 [OpenSSL](#) 來以 base64 解碼檔案的內容，並將解碼資料儲存到新的檔案。以下範例會解碼您在上一個步驟中所儲存檔案的資料 (PublicKey.b64)，並將輸出儲存到新的檔案，檔名為 WrappingPublicKey.bin。

```
$ openssl enc -d -base64 -A -in PublicKey.b64 -out WrappingPublicKey.bin
```

- 複製 base64 編碼匯入字符 (在範例輸出中表示為#### (*base64 ##*))、將其貼入新檔案，然後儲存檔案。提供檔案描述性的名稱，例如 importtoken.b64。
- 使用 [OpenSSL](#) 來以 base64 解碼檔案的內容，並將解碼資料儲存到新的檔案。以下範例會解碼您在上一個步驟中所儲存檔案的資料 (ImportToken.b64)，並將輸出儲存到新的檔案，檔名為 ImportToken.bin。

```
$ openssl enc -d -base64 -A -in importtoken.b64 -out ImportToken.bin
```

繼續執行「[步驟 3：加密金鑰材料](#)」。

步驟 3：加密金鑰材料

在您[下載公有金鑰和匯入字符](#)之後，請使用您下載的公有金鑰與您指定的包裝演算法來加密金鑰資料。如果您需要替換公有金鑰或匯入字符，或要變更包裝算法，則必須下載新的公有金鑰和匯入字符。如需 AWS KMS 支援的公有金鑰和包裝演算法的相關資訊，請參閱 [選取包裝公有金鑰規格](#) 和 [選取包裝演算法](#)。

金鑰材料必須是二進位格式。如需詳細資訊，請參閱 [匯入金鑰資料需求](#)。

Note

對於非對稱金鑰對，請僅加密和匯入私有金鑰。會從私有金鑰 AWS KMS 衍生公有金鑰。

不支援下列組合：ECC_NIST_P521 金鑰資料、RSA_2048 公有包裝金鑰規格以及 RSAES_OAE_SHA_* 包裝演算法。

您無法使用 RSA_2048 公有包裝金鑰直接包裝 ECC_NIST_P521 金鑰材質。使用較大的包裝金鑰或 RSA_AES_KEY_WRAP_SHA_* 包裝演算法。

中國區域不支援 RSA_AES_KEY_WRAP_SHA_256 和 RSA_AES_KEY_WRAP_SHA_1 包裝演算法。

一般而言，當您從硬體安全模組 (HSM) 或金鑰管理系統匯出金鑰材料，您會將其加密。如需如何以二進位格式匯出金鑰材料的詳細資訊，請參閱您的 HSM 或金鑰管理系統的文件。您也可以參閱以下章節，其中提供使用 OpenSSL 的概念驗證示範。

當加密金鑰資料時，請使用您在[下載包裝公有金鑰與匯入字符](#)時所指定的包裝演算法。若要尋找您指定的包裝演算法，請參閱 CloudTrail 日誌事件，以瞭解關聯的 [GetParametersForImport](#) 要求。

產生用於測試的金鑰資料

下列 OpenSSL 指令會產生每種受支援類型的金鑰資料以供測試。這些範例僅供測試與概念驗證示範之用。對於生產系統，請使用更安全方法 (例如商業 HSM 或金鑰管理系統) 來產生並存放您的金鑰資料。

若要將非對稱金鑰配對的私有金鑰轉換為 DER 編碼格式，請將金鑰資料產生命令傳輸至下列 `openssl pkcs8` 命令。此 `topk8` 參數會指示 OpenSSL 取得私有金鑰做為匯入，並傳回 PKCS #8 格式化的金鑰。(預設行為是相反。)

```
openssl pkcs8 -topk8 -outform der -nocrypt
```

下列命令會為每個受支援的金鑰類型產生測試金鑰資料。

- 對稱加密金鑰 (32 位元組)

此命令會產生 256 位元對稱金鑰 (32 位元組隨機字串)，並將其儲存在 `PlaintextKeyMaterial.bin` 檔案。您不需要對此金鑰資料進行編碼。

```
openssl rand -out PlaintextKeyMaterial.bin 32
```

僅在中國區域，您必須產生 128 位元對稱金鑰 (16 位元組隨機字串)。

```
openssl rand -out PlaintextKeyMaterial.bin 16
```

- HMAC 金鑰

此命令產生指定大小的隨機位元組字串。您不需要對此金鑰資料進行編碼。

HMAC 金鑰的長度必須符合 KMS 金鑰規格所定義的長度。例如，如果 KMS 金鑰是 HMAC_384，您必須匯入 384 位元 (48 位元組) 金鑰。

```
openssl rand -out HMAC_224_PlaintextKey.bin 28
openssl rand -out HMAC_256_PlaintextKey.bin 32
openssl rand -out HMAC_384_PlaintextKey.bin 48
openssl rand -out HMAC_512_PlaintextKey.bin 64
```

- RSA 私有金鑰

```
openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:2048 | openssl pkcs8 -topk8 -outform der -nocrypt > RSA_2048_PrivateKey.der
```

```
openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:3072 | openssl pkcs8 -topk8 -outform der -nocrypt > RSA_3072_PrivateKey.der
```

```
openssl genpkey -algorithm rsa -pkeyopt rsa_keygen_bits:4096 | openssl pkcs8 -topk8 -outform der -nocrypt > RSA_4096_PrivateKey.der
```

- ECC 私有金鑰

```
openssl genpkey -algorithm ec -pkeyopt ec_paramgen_curve:P-256 | openssl pkcs8 -topk8 -outform der -nocrypt > ECC_NIST_P256_PrivateKey.der
```

```
openssl genpkey -algorithm ec -pkeyopt ec_paramgen_curve:P-384 | openssl pkcs8 -topk8 -outform der -nocrypt > ECC_NIST_P384_PrivateKey.der
```

```
openssl genpkey -algorithm ec -pkeyopt ec_paramgen_curve:P-521 | openssl pkcs8 -topk8 -outform der -nocrypt > ECC_NIST_P521_PrivateKey.der
```

```
openssl genpkey -algorithm ec -pkeyopt ec_paramgen_curve:secp256k1 | openssl pkcs8 -topk8 -outform der -nocrypt > ECC_SECG_P256K1_PrivateKey.der
```

- SM2 私有金鑰 (僅限中國區域)

```
openssl genpkey -algorithm ec -pkeyopt ec_paramgen_curve:sm2 | openssl pkcs8 -topk8 -outform der -nocrypt > SM2_PrivateKey.der
```

範例：使用 OpenSSL 加密金鑰資料

下列範例說明如何使用 [OpenSSL](#) 以下載的公有金鑰來加密金鑰內容。若要使用 SM2 公有金鑰加密金鑰材料 (僅限中國區域)，請使用 [SM2OfflineOperationHelper](#) 類別。如需每個包裝演算法支援的金鑰材料類型的詳細資訊，請參閱 [the section called “選取包裝演算法”](#)。

Important

此範例僅為概念驗證示範。對於生產系統，請使用更安全方法 (例如商業 HSM 或金鑰管理系統) 來產生和存放您的金鑰材料。

不支援下列組合：ECC_NIST_P521 金鑰資料、RSA_2048 公有包裝金鑰規格以及 RSAES_OAE_SHA_* 包裝演算法。

您無法使用 RSA_2048 公有包裝金鑰直接包裝 ECC_NIST_P521 金鑰材質。使用較大的包裝金鑰或 RSA_AES_KEY_WRAP_SHA_* 包裝演算法。

RSAES_OAEP_SHA_1

AWS KMS 支援對稱加密金鑰 (SYMMETRIC_DEFAULT)、橢圓曲線 (ECC) 私有金鑰、SM2 私有金鑰和 HMAC 金鑰的 RSAES_OAEP_SHA_1。

RSAES_OAEP_SHA_1 不支援 RSA 私有金鑰。此外，您不能將 RSA_2048 公有包裝金鑰與任何 RSAES_OAEP_SHA_* 包裝演算法搭配使用，來包裝 ECC_NIST_P521 (secp521r1) 私有金鑰。您必須使用較大的公有包裝金鑰或 RSA_AES_KEY_WRAP 包裝演算法。

下列範例會使用[您下載的公有金鑰](#)與 RSAES_OAEP_SHA_1 包裝演算法來加密金鑰資料，並將其儲存在 EncryptedKeyMaterial.bin 檔案。

在此範例中：

- *WrappingPublicKey.bin* 是包含下載的包裝公有金鑰的檔案。
- *PlaintextKeyMaterial.bin* 是包含您正在加密的金鑰資料的檔案，例如 PlaintextKeyMaterial.bin、HMAC_384_PlaintextKey.bin 或 ECC_NIST_P521_PrivateKey.der。

```
$ openssl pkeyutl \  
  -encrypt \  
  -in PlaintextKeyMaterial.bin \  
  -out EncryptedKeyMaterial.bin \  
  -inkey WrappingPublicKey.bin \  
  -keyform DER \  
  -pubin \  
  -pkeyopt rsa_padding_mode:oaep \  
  -pkeyopt rsa_oaep_md:sha1
```

RSAES_OAEP_SHA_256

AWS KMS 支援對稱加密金鑰 (SYMMETRIC_DEFAULT)、橢圓曲線 (ECC) 私有金鑰、SM2 私有金鑰和 HMAC 金鑰的 RSAES_OAEP_SHA_256。

RSAES_OAEP_SHA_256 不支援 RSA 私有金鑰。此外，您不能將 RSA_2048 公有包裝金鑰與任何 RSAES_OAEP_SHA_* 包裝演算法搭配使用，來包裝 ECC_NIST_P521 (secp521r1) 私有金鑰。您必須使用較大的公有金鑰或 RSA_AES_KEY_WRAP 包裝演算法。

下列範例會使用[您下載的公有金鑰](#)與 RSAES_OAEP_SHA_256 包裝演算法來加密金鑰資料，並將其儲存在 EncryptedKeyMaterial.bin 檔案。

在此範例中：

- *WrappingPublicKey.bin* 是包含下載的公有包裝金鑰的檔案。如果從主控台下載公有金鑰，這個檔案名稱為 wrappingKey_*KMS key_key_ID_timestamp* (例如 wrappingKey_f44c4e20-f83c-48f4-adc6-a1ef38829760_0809092909)。
- *PlaintextKeyMaterial.bin* 是包含您正在加密金鑰資料的檔案，例如 PlaintextKeyMaterial.bin、HMAC_384_PlaintextKey.bin 或 ECC_NIST_P521_PrivateKey.der。

```
$ openssl pkeyutl \  
-encrypt \  
-in PlaintextKeyMaterial.bin \  
-out EncryptedKeyMaterial.bin \  
-inkey WrappingPublicKey.bin \  
-keyform DER \  
-pubin \  
-pkeyopt rsa_padding_mode:oaep \  
-pkeyopt rsa_oaep_md:sha256 \  
-pkeyopt rsa_mgf1_md:sha256
```

RSA_AES_KEY_WRAP_SHA_1

此 RSA_AES_KEY_WRAP_SHA_1 RSA 包裝演算法涉及兩個加密操作。

1. 使用您產生的 AES 對稱金鑰與 AES 對稱加密演算法來加密金鑰資料。
2. 將您所下載的公有金鑰與 RSAES_OAEP_SHA_1 包裝演算法搭配使用的 AES 對稱金鑰加密。

The RSA_AES_KEY_WRAP_SHA_1 包裝演算法需要 OpenSSL 版本 3. x 或更新版本。

1. 產生 256 位元 AES 對稱加密金鑰

此命令會產生由 256 個隨機位元組成的 AES 對稱加密金鑰，並將其儲存在 `aes-key.bin` 檔案

```
# Generate a 32-byte AES symmetric encryption key
$ openssl rand -out aes-key.bin 32
```

2. 使用 AES 對稱加密金鑰加密您的金鑰資料

此命令會使用 AES 對稱加密金鑰加密金鑰，並將加密的金鑰資料儲存在 `key-material-wrapped.bin` 檔案。

以下是範例命令：

- *PlaintextKeyMaterial.bin* 是包含您要匯入金鑰資料的檔案，例如 `PlaintextKeyMaterial.bin`、`HMAC_384_PlaintextKey.bin`、`RSA_3072_PrivateKey.der` 或 `ECC_NIST_P521_PrivateKey.der`。
- *aes-key.bin* 是包含您在上一個命令中產生的 256 位元 AES 對稱加密金鑰的檔案。

```
# Encrypt your key material with the AES symmetric encryption key
$ openssl enc -id-aes256-wrap-pad \
    -K "$(xxd -p < aes-key.bin | tr -d '\n')" \
    -iv A65959A6 \
    -in PlaintextKeyMaterial.bin \
    -out key-material-wrapped.bin
```

3. 使用公有金鑰加密 AES 對稱加密金鑰

此命令使用您下載的公有金鑰與 `RSAES_OAEP_SHA_1` 包裝演算法對您的 AES 對稱加密金鑰進行加密，並將其保存在 `aes-key-wrapped.bin` 檔案。

以下是範例命令：

- *WrappingPublicKey.bin* 是包含下載的公有包裝金鑰的檔案。如果從主控台下載公有金鑰，此檔案名稱為 `wrappingKey_KMS key_key_ID_timestamp` (例如 `wrappingKey_f44c4e20-f83c-48f4-adc6-a1ef38829760_0809092909`)。
- *aes-key.bin* 是包含您在此範例序列的第一個命令所產生的 256 位元 AES 對稱加密金鑰的檔案。

```
# Encrypt your AES symmetric encryption key with the downloaded public key
$ openssl pkeyutl \
  -encrypt \
  -in aes-key.bin \
  -out aes-key-wrapped.bin \
  -inkey WrappingPublicKey.bin \
  -keyform DER \
  -pubin \
  -pkeyopt rsa_padding_mode:oaep \
  -pkeyopt rsa_oaep_md:sha1 \
  -pkeyopt rsa_mgf1_md:sha1
```

4. 產生要匯入的檔案

將檔案與加密的金鑰資料及檔案與加密的 AES 金鑰連接起來。將它們儲存在 EncryptedKeyMaterial.bin 檔案，這是您將匯入在 [步驟 4：匯入金鑰材料](#) 檔案。

以下是範例命令：

- *key-material-wrapped.bin* 是包含加密金鑰資料的檔案。
- *aes-key-wrapped.bin* 是包含加密 AES 加密金鑰的檔案。

```
# Combine the encrypted AES key and encrypted key material in a file
$ cat aes-key-wrapped.bin key-material-wrapped.bin > EncryptedKeyMaterial.bin
```

RSA_AES_KEY_WRAP_SHA_256

此 RSA_AES_KEY_WRAP_SHA_256 RSA 包裝演算法涉及兩個加密操作。

1. 使用您產生的 AES 對稱金鑰與 AES 對稱加密演算法來加密金鑰資料。
2. 將您所下載的公有金鑰與 RSAES_OAEP_SHA_256 包裝演算法搭配使用的 AES 對稱金鑰加密。

The RSA_AES_KEY_WRAP_SHA_256 包裝演算法需要 OpenSSL 版本 3.0 或更新版本。

1. 產生 256 位元 AES 對稱加密金鑰

此命令會產生由 256 個隨機位元組成的 AES 對稱加密金鑰，並將其儲存在 `aes-key.bin` 檔案

```
# Generate a 32-byte AES symmetric encryption key
$ openssl rand -out aes-key.bin 32
```

2. 使用 AES 對稱加密金鑰加密您的金鑰資料

此命令會使用 AES 對稱加密金鑰加密金鑰，並將加密的金鑰資料儲存在 `key-material-wrapped.bin` 檔案。

以下是範例命令：

- *PlaintextKeyMaterial.bin* 是包含您要匯入金鑰資料的檔案，例如 `PlaintextKeyMaterial.bin`、`HMAC_384_PlaintextKey.bin`、`RSA_3072_PrivateKey.der` 或 `ECC_NIST_P521_PrivateKey.der`。
- *aes-key.bin* 是包含您在上一個命令中產生的 256 位元 AES 對稱加密金鑰的檔案。

```
# Encrypt your key material with the AES symmetric encryption key
$ openssl enc -id-aes256-wrap-pad \
    -K "$(xxd -p < aes-key.bin | tr -d '\n')" \
    -iv A65959A6 \
    -in PlaintextKeyMaterial.bin \
    -out key-material-wrapped.bin
```

3. 使用公有金鑰加密 AES 對稱加密金鑰

此命令使用您下載的公有金鑰與 `RSAES_OAEP_SHA_256` 包裝演算法對您的 AES 對稱加密金鑰進行加密，並將其保存在 `aes-key-wrapped.bin` 檔案。

以下是範例命令：

- *WrappingPublicKey.bin* 是包含下載的公有包裝金鑰的檔案。如果從主控台下載公有金鑰，此檔案名稱為 `wrappingKey_KMS key_key_ID_timestamp` (例如 `wrappingKey_f44c4e20-f83c-48f4-adc6-a1ef38829760_0809092909`)。
- *aes-key.bin* 是包含您在此範例序列的第一個命令所產生的 256 位元 AES 對稱加密金鑰的檔案。

```
# Encrypt your AES symmetric encryption key with the downloaded public key
$ openssl pkeyutl \
  -encrypt \
  -in aes-key.bin \
  -out aes-key-wrapped.bin \
  -inkey WrappingPublicKey.bin \
  -keyform DER \
  -pubin \
  -pkeyopt rsa_padding_mode:oaep \
  -pkeyopt rsa_oaep_md:sha256 \
  -pkeyopt rsa_mgf1_md:sha256
```

4. 產生要匯入的檔案

將檔案與加密的金鑰資料及檔案與加密的 AES 金鑰連接起來。將它們儲存在 EncryptedKeyMaterial.bin 檔案，這是您將匯入在 [步驟 4：匯入金鑰材料](#) 檔案。

以下是範例命令：

- *key-material-wrapped.bin* 是包含加密金鑰資料的檔案。
- *aes-key-wrapped.bin* 是包含加密 AES 加密金鑰的檔案。

```
# Combine the encrypted AES key and encrypted key material in a file
$ cat aes-key-wrapped.bin key-material-wrapped.bin > EncryptedKeyMaterial.bin
```

繼續執行「[步驟 4：匯入金鑰材料](#)」。

步驟 4：匯入金鑰材料

[加密您的金鑰材料之後](#)，您可以匯入金鑰材料以便搭配使用 AWS KMS key。若要匯入金鑰材料，請上傳在 [步驟 3：加密金鑰材料](#) 加密的金鑰材料，以及在 [步驟 2：下載包裝公有金鑰及匯入字符](#) 下載的匯入符記。您必須將金鑰材料匯入至您在 [下載公有金鑰和匯入字符](#) 時指定的同一個 KMS 金鑰。在成功匯入金鑰資料之後，KMS 金鑰的 [金鑰狀態](#) 會變為 Enabled，這表示您可在密碼編譯操作使用 KMS 金鑰。

在匯入金鑰材料時，您可以為金鑰材料 [設定可選的過期時間](#)。當金鑰材料過期時，AWS KMS 會刪除金鑰材料，讓 KMS 金鑰變成不可用。匯入金鑰材料後，您無法設定、變更或取消目前匯入材料的過期日期。若要變更這些值，您必須 [重新匯入](#) 相同的金鑰材料。

對於具有EXTERNAL原始伺服器的所有 KMS 金鑰，匯入其中的第一個金鑰材料會成為最新版本，並永久與其相關聯。具有EXTERNAL原始伺服器的對稱加密金鑰支援隨需輪換。您可以將多個金鑰材料與支援隨需輪換的匯入金鑰建立關聯。匯入新金鑰材料的程序會因單一區域和多區域金鑰而有所不同，如[匯入新金鑰材料](#)一節所述。您必須使用 [ImportKeyMaterial](#) 動作NEW_KEY_MATERIAL將 importType 參數設定為 `NEW_KEY_MATERIAL`，以將新的金鑰材料與 KMS 金鑰建立關聯。選用ImportType參數的預設值為 `EXISTING_KEY_MATERIAL`。當您省略 ImportType 參數或將其指定為 `EXISTING_KEY_MATERIAL`，您必須匯入先前與 KMS 金鑰相關聯的金鑰材料。

對於非對稱或具有EXTERNAL原始伺服器的 HMAC KMS 金鑰，只能有一個金鑰材料與金鑰相關聯。AWS KMS 將使用 ImportType 參數拒絕 [ImportKeyMaterial](#) API 請求。

匯入與 KMS 金鑰永久相關聯的所有金鑰材料時，KMS 金鑰可用於密碼編譯操作。如果這些金鑰材料中的任何一種遭到刪除或允許過期，KMS 金鑰狀態會變更為 `PendingImport`且金鑰不適用於密碼編譯操作。

若要匯入金鑰材料，您可以使用 [AWS KMS 主控台](#) 或 [ImportKeyMaterial](#) API。您可以透過提出 HTTP 請求，或透過使用 [AWS 開發套件](#)、[AWS Command Line Interface](#) 或 [AWS Tools for PowerShell](#)，直接使用 API。

當您匯入金鑰材料時，會將 [ImportKeyMaterial](#) 項目新增至您的 AWS CloudTrail 日誌，以記錄ImportKeyMaterial操作。無論您使用 AWS KMS 主控台或 AWS KMS API，CloudTrail 項目都相同。

設定到期時間 (選用)

匯入 KMS 金鑰的金鑰材料時，可以為金鑰材料設定選用的到期日期和時間，從匯入日期算起最多 365 天。當匯入的金鑰材料過期時，會將其 AWS KMS 刪除。此動作會將 KMS 金鑰的[金鑰狀態](#)變更為 `PendingImport`，這會防止在任何密碼編譯操作中對其進行使用。若要使用 KMS 金鑰，您必須[重新匯入原始金鑰材料的複本](#)。

確保匯入的金鑰材料頻繁地到期，可協助您滿足法規要求，但這會對使用 KMS 金鑰加密的資料帶來額外的風險。在您重新匯入原始金鑰材料的複本之前，包含過期金鑰材料的 KMS 金鑰將無法使用，而且使用該 KMS 金鑰進行加密的任何資料都無法存取。如果您因任何原因而無法重新匯入金鑰材料，包括遺失原始金鑰材料的複本，則該 KMS 金鑰將永久無法使用，而且使用該 KMS 金鑰進行加密的資料將無法復原。

若要降低此風險，請確定您可以存取匯入金鑰材料的副本，並設計系統在金鑰材料過期和中斷 AWS 工作負載之前刪除和重新匯入金鑰材料。建議您為已匯入的金鑰材料到期[設定一個警示](#)，讓您有足夠的時間在金鑰材料到期前重新匯入。您也可以使用 CloudTrail 日誌來稽核[匯入 \(含重新匯入\) 金鑰材料](#)和[刪除已匯入的金鑰材料](#)的操作，以及[刪除過期金鑰材料](#)的 AWS KMS 操作。

AWS KMS 無法還原、復原或重現已刪除的金鑰材料。您可以透過程式設計方式定期[刪除](#)和[重新匯入](#)已匯入的金鑰材料，而不是設定到期時間，但是保留原始金鑰材料複本的要求是相同的。

您可以在匯入金鑰材料時確定金鑰材料是否有到期時間以及何時到期。不過，您可以開啟和關閉過期，或透過重新匯入金鑰材料來設定新的過期時間。使用 [ImportKeyMaterial](#) 的 `ExpirationModel` 參數來開啟 (`KEY_MATERIAL_EXPIRES`) 和關閉 (`KEY_MATERIAL_DOES_NOT_EXPIRE`) 到期時間，以及使用 `ValidTo` 參數來設定何時到期。最長時間為匯入資料後 365 天；沒有最低限度，但時間必須為將來的時間。

設定金鑰材料描述

具有EXTERNAL原始伺服器的對稱加密金鑰可以有多個與其相關聯的金鑰材料。您可以在將金鑰材料匯入此類金鑰時指定選用的金鑰材料描述。描述可用來追蹤在外部長期維護對應金鑰材料的位置 AWS KMS。

對於多區域金鑰，您只能在主要區域金鑰上設定或變更金鑰材料描述。AWS KMS 會自動將金鑰材料描述傳播至複本區域金鑰。

匯入新的金鑰資料

若要在具有匯入金鑰材料的對稱加密 KMS 金鑰上執行隨需輪換，您必須先匯入新的金鑰材料，先前不會與金鑰相關聯。

- 單一區域金鑰
 - 使用 [ImportKeyMaterial](#) 操作，並將 `ImportType` 參數設定為 `NEW_KEY_MATERIAL` 來完成此任務。在您執行 [RotateKeyOnDemand](#) 操作或在 中輪換金鑰之前，此金鑰材料不會永久與金鑰相關聯 AWS 管理主控台。在此之前，此金鑰材料處於 `PENDING_ROTATION` 狀態。KMS 金鑰隨時最多可有一個處於 `PENDING_ROTATION` 狀態的金鑰材料。可以刪除處於 `PENDING_ROTATION` 狀態的金鑰材料，而不會影響金鑰在密碼編譯操作中的可用性。
- 多區域金鑰
 - 若要將金鑰材料匯入多區域金鑰，您必須先將新的金鑰材料匯入主要區域金鑰。您無法將新的金鑰材料直接匯入複本區域金鑰。將新的金鑰材料匯入主要區域金鑰後，您可以將相同的金鑰材料匯入複本區域金鑰。
 - 針對主要區域金鑰使用 `ImportType` 參數設定為 `NEW_KEY_MATERIAL` [ImportKeyMaterial](#) 的操作來完成此任務。對於複本區域金鑰，請使用的 `EXISTING_KEY_MATERIAL` 參數 `ImportType` 進行 [ImportKeyMaterial](#) 操作。
 - 在金鑰材料狀態變更為 `PENDING_ROTATION` 狀態之前，對稱加密多區域金鑰的金鑰材料必須匯入所有複本區域金鑰和主要區域金鑰。在此之前，新金鑰材料的狀態為

PENDING_MULTI_REGION_IMPORT_AND_ROTATION。KMS 金鑰隨時最多可有一個處於 PENDING_ROTATION 或 PENDING_MULTI_REGION_IMPORT_AND_ROTATION 狀態的金鑰材料（請參閱 [RotationsListEntry](#) 中的 KeyMaterialState 描述）。處於 PENDING_MULTI_REGION_IMPORT_AND_ROTATION 或 PENDING_ROTATION 狀態的金鑰材料不會永久與金鑰相關聯，而且可以在不影響金鑰在密碼編譯操作中的可用性的情況下刪除。

重新匯入金鑰材料

如果您管理的 KMS 金鑰含有已匯入的金鑰資料，您可能需要重新匯入金鑰資料。您可能重新匯入金鑰材料來取代即將過期或已刪除的金鑰材料，或變更金鑰材料的過期模型或過期日期。

您可以根據自己的安全性需求來排程或隨時重新匯入金鑰材料。不必等到金鑰材料到期或接近到期時間。

重新匯入金鑰材料的程序與您第一次用來匯入金鑰材料的程序相同，但有下列例外。

- 使用現有的 KMS 金鑰，而不是建立新的 KMS 金鑰。您可以略過匯入程序的 [步驟 1](#)。
- 重新匯入金鑰資料時，您可以變更過期模型和過期日期。對於對稱加密金鑰，您也可以變更金鑰材料描述。

對於多區域金鑰，您只能在主要區域金鑰上設定或變更金鑰材料描述。AWS KMS 會自動將金鑰材料描述傳播至複本區域金鑰。

每次將金鑰材料匯入 KMS 金鑰時，您需要為 KMS 金鑰 [下載並使用新的包裝金鑰和匯入字符](#)。包裝程序不會影響金鑰資料內容，因此您可採用不同包裝公有金鑰和不同包裝演算法來匯入相同金鑰資料。

匯入金鑰材料 (主控台)

您可以使用 AWS 管理主控台 匯入金鑰材料。

1. 如您在 Upload your wrapped key material (上傳包裝的金鑰資料) 頁面，請跳至 [Step 10](#)。
2. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
3. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
4. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
5. 針對您已下載公開金鑰和匯入字符的 KMS 金鑰，選擇其金鑰 ID 或別名。
6. 選擇 Cryptographic configuration (密碼編譯組態) 索引標籤並檢視其值。這些索引標籤位於 General configuration (一般組態) 區段下 KMS 金鑰的詳細資訊頁面上。

您僅能將金鑰資料匯入 Origin (來源) 是 EXTERNAL (外部 (匯入金鑰材料)) 的 KMS 金鑰。如需有關建立具有匯入金鑰材料之 KMS 金鑰的資訊，請參閱 [匯入金鑰的 AWS KMS 金鑰材料](#)。

7. 根據您的金鑰類型選擇適當的標籤。

- 對於非對稱和 HMAC 金鑰，請選擇金鑰材料索引標籤。
- 針對對稱加密金鑰，選擇金鑰材料和輪換索引標籤。

8. 選擇匯入動作。

- 針對非對稱和 HMAC 金鑰，選擇匯入金鑰材料。
- 針對對稱加密金鑰，選擇下列其中一項：
 - 匯入初始金鑰材料 (如果尚未匯入金鑰材料)
 - 匯入新的金鑰材料 (新增用於輪換的新材料)
 - 重新匯入金鑰材料 (可從金鑰材料資料表中的動作功能表取得)

 Note

對於多區域金鑰，您必須先將新的金鑰材料匯入主要區域金鑰。然後，將相同的金鑰材料匯入每個複本區域金鑰。

對於主要多區域金鑰，金鑰資料資料表包含複本匯入狀態欄，顯示所有複本區域的匯入狀態 (例如，匯入 3 個中的 0 個)。選擇複本匯入狀態值，以開啟顯示每個複本區域的匯入狀態的模態。模態為尚未匯入新金鑰材料的複本區域提供匯入金鑰材料連結。

9. 如您已下載金鑰資料、匯入字符並加密金鑰資料，請選擇 Next (下一步)。

 Note

對於多區域金鑰，您必須先將新的金鑰材料匯入主要區域金鑰。然後，您可以將相同的金鑰材料匯入複本區域金鑰。

10. 在 Encrypted key material and import token (加密金鑰資料與匯入字符) 區段，執行下列動作。

- a. 在 Wrapped key material (包裝的金鑰資料)，選擇 Choose file (選擇檔案)。然後，上傳內含包裝 (加密) 金鑰資料的檔案。
- b. 在 Import token (匯入字符)，選擇 Choose file (上傳檔案)。上傳內含您[下載](#)之匯入符記的檔案。

11. 在 Expiration option (過期選項) 區段中，您可以決定金鑰材料是否會過期。若要設定過期日期和時間，請選擇 Key material expires (金鑰資料過期)，然後使用行事曆選取日期和時間。您可以指定自目前日期和時間起的日期，但最長不超過 365 天。
12. 對於對稱加密金鑰，您可以選擇指定要匯入之金鑰資料的描述。
13. 選擇匯入金鑰材料。

匯入金鑰材料 (AWS KMS API)

若要匯入金鑰材料，請使用 [ImportKeyMaterial](#) 操作。以下範例使用 [AWS CLI](#)，但您可以使用任何支援的程式設計語言。

若要使用此範例：

1. 將 `1234abcd-12ab-34cd-56ef-1234567890ab` 取代為您在下載公有金鑰和匯入字符時指定的 KMS 金鑰的金鑰 ID。若要識別 KMS 金鑰，請使用它的 [金鑰 ID](#) 或 [金鑰 ARN](#)。不能對此操作使用 [別名](#) 或 [別名 ARN](#)。
2. 將 `EncryptedKeyMaterial.bin` 取代為包含加密金鑰材料的檔案名稱。
3. 將 `ImportToken.bin` 取代為包含匯入符記的檔案名稱。
4. 如果希望匯入的金鑰材料過期，請將 `expiration-model` 參數設定為其預設值 `KEY_MATERIAL_EXPIRES`，或省略 `expiration-model` 參數。然後，將 `valid-to` 參數的值取代為您希望金鑰材料到期的日期和時間。日期和時間最多可以是自提出請求之日起的 365 天內。

```
$ aws kms import-key-material --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --encrypted-key-material fileb://EncryptedKeyMaterial.bin \
  --import-token fileb://ImportToken.bin \
  --expiration-model KEY_MATERIAL_EXPIRES \
  --valid-to 2023-06-17T12:00:00-08:00
```

如果不希望匯入的金鑰材料過期，請將 `expiration-model` 參數的值設定為 `KEY_MATERIAL_DOES_NOT_EXPIRE`，並省略命令中的 `valid-to` 參數。

```
$ aws kms import-key-material --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --encrypted-key-material fileb://EncryptedKeyMaterial.bin \
  --import-token fileb://ImportToken.bin \
  --expiration-model KEY_MATERIAL_DOES_NOT_EXPIRE
```

5. 如果您想要匯入新的金鑰材料，但先前未與 KMS 金鑰相關聯，請將 `ImportType` 參數設定為 `NEW_KEY_MATERIAL`。此選項只能與對稱加密金鑰搭配使用。對於這些金鑰，您也可以在下列命令列範例中，使用選用 `KeyMaterialDescription` 參數來設定匯入金鑰資料的描述：

```
$ aws kms import-key-material --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --encrypted-key-material fileb://EncryptedKeyMaterial.bin \
  --import-token fileb://ImportToken.bin \
  --expiration-model KEY_MATERIAL_EXPIRES \
  --valid-to 2023-06-17T12:00:00-08:00 \
  --import-type NEW_KEY_MATERIAL \
  --key-material-description "Q2 2025 Rotation"
```

6. 對於多區域金鑰，您只能在主要區域金鑰上設定或變更金鑰材料描述。AWS KMS 會自動將金鑰材料描述傳播至複本區域金鑰。

Tip

如果命令未成功，您可能看到 `KMSInvalidStateException` 或 `NotFoundException`。您可重試請求。

在金鑰存放區中建立 KMS AWS CloudHSM 金鑰

建立 AWS CloudHSM 金鑰存放區之後，您可以在金鑰存放區 AWS KMS keys 中建立。它們必須是具有 AWS KMS 產生的金鑰材料的[對稱加密 KMS 金鑰](#)。您無法在自訂金鑰存放區中建立[非對稱 KMS 金鑰](#)、[HMAC KMS 金鑰](#)，或是含有[匯入金鑰資料](#)的 KMS 金鑰。此外，您無法在自訂金鑰存放區中使用對稱加密 KMS 金鑰來產生非對稱資料金鑰對。KMS 無法透過 IPv6 與 AWS CloudHSM 金鑰存放區通訊。

若要在金鑰存放區中建立 KMS AWS CloudHSM 金鑰，AWS CloudHSM 金鑰存放區必須[連線至相關聯的 AWS CloudHSM 叢集](#)，而且叢集必須在不同的可用區域中包含至少兩個作用中的 HSMs。若要尋找 HSM 的連接狀態和數目，請檢視 AWS 管理主控台內的 [AWS CloudHSM 金鑰存放區頁面](#)。使用 API 操作時，請使用 [DescribeCustomKeyStores](#) 操作來驗證 AWS CloudHSM 金鑰存放區是否已連線。若要驗證叢集中的作用中 HSMs 數量及其可用區域，請使用 AWS CloudHSM [DescribeClusters](#) 操作。

當您在 AWS CloudHSM 金鑰存放區中建立 KMS 金鑰時，會在其中 AWS KMS 建立 KMS 金鑰 AWS KMS。但是，它會為相關聯 AWS CloudHSM 叢集中的 KMS 金鑰建立金鑰材料。具體而言，會將

AWS KMS 登入叢集，做為 [kmsuser](#) 您建立的 CU。然後，它會在叢集建立持久性、不可擷取的 256 位元進階加密標準 (AES) 對稱金鑰。AWS KMS 會將只在叢集內可見的 [金鑰標籤屬性](#) 的值，設定為 KMS 金鑰的 Amazon Resource Name (ARN)。

當命令成功時，新 KMS 金鑰的 [金鑰狀態](#) 是 Enabled，而其來源是 AWS_CLOUDHSM。建立任何 KMS 金鑰之後就無法變更其來源。當您在 AWS KMS 主控台的 AWS CloudHSM 金鑰存放區中或使用 [DescribeKey](#) 操作檢視 KMS 金鑰時，您可以看到典型屬性，例如其金鑰 ID、金鑰狀態和建立日期。但是，您也可以查看自訂金鑰存放區 ID 和 (選擇性) AWS CloudHSM 叢集 ID。

如果您嘗試在金鑰存放區中建立 KMS AWS CloudHSM 金鑰失敗，請使用錯誤訊息來協助您判斷原因。這可能表示 AWS CloudHSM 金鑰存放區未連線 (CustomKeyStoreInvalidStateException)，或關聯的 AWS CloudHSM 叢集沒有此操作所需的兩個作用中 HSMs (CloudHsmClusterInvalidConfigurationException)。如需協助，請參閱 [對自訂金鑰存放區進行故障診斷](#)。

如需在金鑰存放區中建立 KMS AWS CloudHSM 金鑰之操作的 AWS CloudTrail 日誌範例，請參閱 [CreateKey](#)。

在 CloudHSM 金鑰存放區中建立新的 KMS 金鑰

您可以在主控台的 AWS CloudHSM AWS KMS 金鑰存放區中或使用 [CreateKey](#) 操作來建立對稱加密 KMS 金鑰。

使用 AWS KMS 主控台

使用下列程序在金鑰存放區中建立對稱加密 KMS AWS CloudHSM 金鑰。

Note

請勿在別名、說明或標籤包含機密或敏感資訊。這些欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
4. 選擇建立金鑰。

5. 選擇 Symmetric (對稱)。
6. 在 Key usage (金鑰用途) 欄位中，系統會自動選取 Encrypt and decrypt (加密和解密) 選項。請勿變更該欄位。
7. 選擇 Advanced options (進階選項)。
8. 對於金鑰資料來源，選擇 AWS CloudHSM 金鑰存放區。

您無法在 金鑰存放區中建立多區域 AWS CloudHSM 金鑰。

9. 選擇下一步。
10. 為您的新 KMS AWS CloudHSM 金鑰選取金鑰存放區。若要建立新的 AWS CloudHSM 金鑰存放區，請選擇建立自訂金鑰存放區。

您選擇的 AWS CloudHSM 金鑰存放區必須具有連線狀態。其關聯的 AWS CloudHSM 叢集必須處於作用中狀態，並在不同的可用區域中包含至少兩個作用中的 HSMs。

如需連接 AWS CloudHSM 金鑰存放區的協助，請參閱 [中斷連接 AWS CloudHSM 金鑰存放區](#)。
如需新增 HSM 的說明，請參閱《AWS CloudHSM 使用者指南》中的 [新增 HSM](#)。

11. 選擇下一步。
12. 輸入 KMS 金鑰的別名和選用描述。
13. (選用)。在 Add Tags (新增標籤) 頁面，新增標籤來識別或分類 KMS 金鑰。

當您將標籤新增至 AWS 資源時，AWS 會產生成本分配報告，其中包含依標籤彙總的用量和成本。標籤也可以用來控制 KMS 金鑰的存取。如需標記 KMS 金鑰的詳細資訊，請參閱 [中的標籤 AWS KMS](#) 和 [適用於的 ABAC AWS KMS](#)。

14. 選擇下一步。
15. 在 Key Administrators (金鑰管理員) 區段中，選取可管理 KMS 金鑰的 IAM 使用者和角色。如需詳細資訊，請參閱 [允許金鑰管理員來管理 KMS 金鑰](#)。

備註

IAM 政策可授權其他 IAM 使用者和角色來使用 KMS 金鑰。

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的安全性最佳實務](#)。

AWS KMS 主控台會將金鑰管理員新增至陳述式識別碼下的金鑰政策 "Allow access for Key Administrators"。修改此陳述式識別符可能會影響主控台顯示您對陳述式所做的更新。

16. (選用) 若要防止這些金鑰管理員刪除此 KMS 金鑰，請清除頁面底部的 Allow key administrators to delete this key (允許金鑰管理員刪除此金鑰) 方塊。
17. 選擇下一步。
18. 在此帳戶區段中，選取可在[密碼編譯操作](#)中使用 KMS 金鑰的 IAM AWS 帳戶使用者和角色。如需詳細資訊，請參閱[允許金鑰使用者使用 KMS 金鑰](#)。

 備註

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的安全性最佳實務](#)。

AWS KMS 主控台會將金鑰使用者新增至陳述式識別碼 "Allow use of the key" 和下的金鑰政策 "Allow attachment of persistent resources"。修改這些陳述式識別符可能會影響主控台顯示您對陳述式所做的更新的方式。

19. (選用) 您可以允許其他 AWS 帳戶使用此 KMS 金鑰進行密碼編譯操作。若要這樣做，請在頁面底部的其他 AWS 帳戶區段中，選擇新增另一個 AWS 帳戶並輸入外部帳戶的 AWS 帳戶 ID。若要新增多個外部帳戶，請重複此步驟。

 Note

其他的管理員 AWS 帳戶也必須透過為其使用者建立 IAM 政策來允許存取 KMS 金鑰。如需詳細資訊，請參閱[允許其他帳戶中的使用者使用 KMS 金鑰](#)。

20. 選擇下一步。
21. 檢閱金鑰的金鑰政策陳述式。若要變更金鑰政策，請選取編輯。
22. 選擇下一步。
23. 檢閱您選擇的金鑰設定。您仍然可以返回並變更所有設定。
24. 完成時，請選擇 Finish (完成) 以建立金鑰。

當程序成功時，顯示器會在您選擇的金鑰存放區中顯示新的 KMS AWS CloudHSM 金鑰。當您選擇新 KMS 金鑰的名稱或別名時，其詳細資訊頁面上的密碼編譯組態索引標籤會顯示 KMS 金鑰的來源 (AWS CloudHSM)、自訂金鑰存放區的名稱、ID 和類型，以及 AWS CloudHSM 叢集的 ID。如果程序失敗，則會出現錯誤訊息來描述失敗。

i Tip

為了更輕鬆識別自訂金鑰存放區中的 KMS 金鑰，請在 Customer managed keys (客戶受管金鑰) 頁面上，新增要顯示的 Custom key store ID (自訂金鑰存放區 ID) 資料欄。按一下右上方的齒輪圖示，然後選取 Custom key store ID (自訂金鑰存放區 ID)。如需詳細資訊，請參閱[自訂您的主控台檢視](#)。

使用 AWS KMS API

若要在 AWS CloudHSM 金鑰存放區中建立新的 AWS KMS key (KMS 金鑰)，請使用 [CreateKey](#) 操作。使用 CustomKeyId 參數來識別您的自訂金鑰存放區，並指定 Origin 值為 AWS_CLOUDHSM。

您可能還想要使用 Policy 參數來指定金鑰政策。您隨時可以變更金鑰政策 ([PutKeyPolicy](#)) 並新增選用元素，例如[描述](#)和[標籤](#)。

本節中的範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

下列範例從呼叫 [DescribeCustomKeyStores](#) 操作開始，以驗證 AWS CloudHSM 金鑰存放區是否已連接到其相關聯的 AWS CloudHSM 叢集。在預設情況下，此操作會傳回您帳戶和區域中的所有自訂金鑰存放區。若要僅描述特定 AWS CloudHSM 金鑰存放區，請使用其 CustomKeyId 或 CustomKeyName 參數（但不能同時使用兩者）。

執行此命令之前，請將範例自訂金鑰存放區 ID 換成有效的 ID。

i Note

請勿在 Description 或 Tags 欄位包含機密或敏感資訊。這些欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

```
$ aws kms describe-custom-key-stores --custom-key-store-id cks-1234567890abcdef0
{
  "CustomKeyStores": [
    {
      "CustomKeyId": "cks-1234567890abcdef0",
      "CustomKeyName": "ExampleKeyStore",
      "CustomKeyType": "AWS CloudHSM key store",
```

```

    "CloudHsmClusterId": "cluster-1a23b4cdefg",
    "TrustAnchorCertificate": "<certificate string appears here>",
    "CreationDate": "1.499288695918E9",
    "ConnectionState": "CONNECTED"
  ],
}

```

下一個範例命令使用 [DescribeClusters](#) 操作來驗證與 ExampleKeyStore(cluster-1a23b4cdefg) 相關聯的 AWS CloudHSM 叢集至少有兩個作用中 HSMs。如果叢集的 HSM 少於兩個，CreateKey 操作會失敗。

```

$ aws cloudhsmv2 describe-clusters
{
  "Clusters": [
    {
      "SubnetMapping": {
        ...
      },
      "CreateTimestamp": 1507133412.351,
      "ClusterId": "cluster-1a23b4cdefg",
      "SecurityGroup": "sg-865af2fb",
      "HsmType": "hsm1.medium",
      "VpcId": "vpc-1a2b3c4d",
      "BackupPolicy": "DEFAULT",
      "Certificates": {
        "ClusterCertificate": "-----BEGIN CERTIFICATE-----\...\n-----END
CERTIFICATE-----\n"
      },
      "Hsms": [
        {
          "AvailabilityZone": "us-west-2a",
          "EniIp": "10.0.1.11",
          "ClusterId": "cluster-1a23b4cdefg",
          "EniId": "eni-ea8647e1",
          "StateMessage": "HSM created.",
          "SubnetId": "subnet-a6b10bd1",
          "HsmId": "hsm-abcdefghijkl",
          "State": "ACTIVE"
        },
        {
          "AvailabilityZone": "us-west-2b",
          "EniIp": "10.0.0.2",
          "ClusterId": "cluster-1a23b4cdefg",

```

```

        "EniId": "eni-ea8647e1",
        "StateMessage": "HSM created.",
        "SubnetId": "subnet-b6b10bd2",
        "HsmId": "hsm-zyxwvutsrq",
        "State": "ACTIVE"
    },
    ],
    "State": "ACTIVE"
}
]
}

```

此範例命令使用 [CreateKey](#) 操作在 金鑰存放區中建立 KMS AWS CloudHSM 金鑰。若要在金鑰存放區中建立 KMS AWS CloudHSM 金鑰，您必須提供金鑰存放區的自訂 AWS CloudHSM 金鑰存放區 ID，並指定 `Origin` 的值 `AWS_CLOUDHSM`。

回應包含自訂金鑰存放區和 AWS CloudHSM 叢集的 IDs。

執行此命令之前，請將範例自訂金鑰存放區 ID 換成有效的 ID。

```

$ aws kms create-key --origin AWS_CLOUDHSM --custom-key-store-id cks-1234567890abcdef0
{
  "KeyMetadata": {
    "AWSAccountId": "111122223333",
    "Arn": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "CreationDate": 1.499288695918E9,
    "Description": "Example key",
    "Enabled": true,
    "MultiRegion": false,
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",
    "KeyState": "Enabled",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "Origin": "AWS_CLOUDHSM",
    "CloudHsmClusterId": "cluster-1a23b4cdefg",
    "CustomKeyStoreId": "cks-1234567890abcdef0",
    "KeySpec": "SYMMETRIC_DEFAULT",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ]
  }
}

```

```
}
```

在外部金鑰存放區中建立 KMS 金鑰

[建立](#)並[連接](#)外部金鑰存放區之後，您可以在金鑰存放區 AWS KMS keys 中建立。其必須是具有 External key store (外部金鑰存放區) (EXTERNAL_KEY_STORE) 原始值的[對稱加密 KMS 金鑰](#)。您無法在自訂金鑰存放區中建立[非對稱 KMS 金鑰](#)、[HMAC KMS 金鑰](#)，或是含有[匯入金鑰資料](#)的 KMS 金鑰。此外，您無法在自訂金鑰存放區中使用對稱加密 KMS 金鑰來產生非對稱資料金鑰對。

與標準 KMS 金鑰相比，外部金鑰存放區中的 KMS 金鑰可能具有較差的延遲、耐久性和可用性，因為其依賴於 AWS 之外的元件。在外部金鑰存放區中建立或使用 KMS 金鑰之前，請確認您需要具有外部金鑰存放區屬性的金鑰。

Note

有些外部金鑰管理器會提供更簡單的方法來建立外部金鑰存放區的 KMS 金鑰。如需詳細資訊，請參閱外部金鑰管理器文件。

若要在外部金鑰存放區建立 KMS 金鑰，您必須指定下列項目：

- 外部金鑰存放區的 ID。
- 外部金鑰存放區 (EXTERNAL_KEY_STORE) 的[金鑰材料來源](#)。
- 與外部金鑰存放區相關聯的[外部金鑰管理器](#)中的現有[外部金鑰](#) ID。此外部金鑰作為 KMS 金鑰的金鑰材料。您無法在建立 KMS 金鑰之後變更外部金鑰 ID。

AWS KMS 在加密和解密操作的請求中提供外部金鑰存放區代理的外部金鑰 ID。AWS KMS 無法直接存取您的外部金鑰管理器或其任何密碼編譯金鑰。

除了外部金鑰之外，外部金鑰存放區中的 KMS 金鑰也具有 AWS KMS 金鑰材料。在 KMS 金鑰下加密的所有資料會先 AWS KMS 使用金鑰的 AWS KMS 金鑰材料在 中加密，然後使用外部金鑰由外部金鑰管理器加密。此[雙重加密](#)程序可確保受外部金鑰存放區中 KMS 金鑰保護的密文至少與僅受 AWS KMS 保護的密文一樣強。如需詳細資訊，請參閱[外部金鑰存放區的運作方式](#)。

當 CreateKey 操作成功時，新 KMS 金鑰的[金鑰狀態](#)為 Enabled。當您[在外部金鑰存放區檢視 KMS 金鑰](#)時，您會看到一般屬性，例如其金鑰 ID、[金鑰規格](#)、[金鑰用途](#)、[金鑰狀態](#)和建立日期。但是您也可以看到外部金鑰存放區的 ID 和[連接狀態](#)以及外部金鑰的 ID。

如果您嘗試在外部金鑰存放區建立 KMS 金鑰失敗，請使用錯誤訊息來判斷原因。這可能表示外部金鑰存放區未連接 (`CustomKeyStoreInvalidStateException`)、外部金鑰存放區代理找不到具有指定外部金鑰 ID 的外部金鑰 (`XksKeyNotFoundException`)，或外部金鑰已與相同外部金鑰存放區中的 KMS 金鑰相關聯 `XksKeyAlreadyInUseException`。

如需在外部金鑰存放區中建立 KMS 金鑰之操作的 AWS CloudTrail 日誌範例，請參閱 [CreateKey](#)。

主題

- [外部金鑰存放區中 KMS 金鑰的要求](#)
- [在外部金鑰存放區中建立新的 KMS 金鑰](#)

外部金鑰存放區中 KMS 金鑰的要求

若要在外部金鑰存放區中建立 KMS 金鑰，外部金鑰存放區、KMS 金鑰以及作為 KMS 金鑰之外部密碼編譯金鑰材料的外部金鑰需要下列屬性。

外部金鑰存放區要求

- 必須連接到其外部金鑰存放區代理。

若要檢視外部金鑰存放區的[連接狀態](#)，請參閱 [檢視外部金鑰存放區](#)。若要連接外部金鑰存放區，請參閱 [連接和中斷連接外部金鑰存放區](#)。

KMS 金鑰要求

您無法在建立 KMS 金鑰之後變更這些屬性。

- 金鑰規格：SYMMETRIC_DEFAULT
- 金鑰用途：ENCRYPT_DECRYPT
- 金鑰材料來源：EXTERNAL_KEY_STORE
- 多區域：FALSE

外部金鑰要求

- 256 位元 AES 密碼編譯金鑰 (256 個隨機位元)。外部金鑰的 `KeySpec` 必須為 `AES_256`。
- 已啟用並可供使用。外部金鑰的 `Status` 必須為 `ENABLED`。
- 設定用於加密和解密。外部金鑰的 `KeyUsage` 必須包含 `ENCRYPT` 和 `DECRYPT`。

- 僅與此 KMS 金鑰搭配使用。外部金鑰存放區中的每個 KMS key 都必須與不同的外部金鑰關聯。

AWS KMS 也建議將外部金鑰專用於外部金鑰存放區。此限制可讓您更輕鬆地識別並解決金鑰的問題。

- 可供外部金鑰存放區的[外部金鑰存放區代理](#)存取。

如果外部金鑰存放區代理找不到使用指定外部金鑰 ID 的金鑰，CreateKey 操作就會失敗。

- 可以處理您使用 AWS 服務產生的預期流量。AWS KMS 建議外部金鑰準備處理每秒最多 1800 個請求。

在外部金鑰存放區中建立新的 KMS 金鑰

您可以在 AWS KMS 主控台的外部金鑰存放區中或使用 [CreateKey](#) 操作建立新的 KMS 金鑰。

使用 AWS KMS 主控台

有兩種方法可以在外部金鑰存放區中建立 KMS 金鑰。

- 方法 1 (建議選擇)：選擇外部金鑰存放區，然後在該外部金鑰存放區建立 KMS 金鑰。
- 方法 2：建立 KMS 金鑰，然後指出金鑰位於外部金鑰存放區。

如果您使用方法 1，在建立金鑰之前選擇外部金鑰存放區，會為您 AWS KMS 選擇所有必要的 KMS 金鑰屬性，並填入外部金鑰存放區的 ID。此方法可避免您在建立 KMS 金鑰時可能發生的錯誤。

Note

請勿在別名、說明或標籤包含機密或敏感資訊。這些欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

方法 1 (建議選擇)：在外部金鑰存放區開始

若要使用此方法，請選擇外部金鑰存放區，然後建立 KMS 金鑰。AWS KMS 主控台會為您選擇所有必要的屬性，並填入外部金鑰存放區的 ID。此方法可避免您在建立 KMS 金鑰時可能發生的許多錯誤。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。

3. 在導覽窗格中，依次選擇 Custom key stores (自訂金鑰存放區)、External key stores (外部金鑰存放區)。
4. 選擇外部金鑰存放區的名稱。
5. 在右上角，選擇 Create a KMS key in this key store (在此金鑰存放區建立 KMS 金鑰)。

如果未連接外部金鑰存放區，系統會提示您進行連接。如果連接嘗試失敗，則您需要先解決問題並連接外部金鑰存放區，才能在其中建立新的 KMS 金鑰。

如果外部金鑰存放區已連接，則系統會將您重新導向至 Customer managed keys (客戶受管金鑰) 頁面以建立金鑰。已為您選擇所需的 Key configuration (金鑰組態) 值。此外，會填寫外部金鑰存放區的自訂金鑰存放區 ID，不過您可以變更它。

6. 在[外部金鑰管理器](#)中輸入[外部金鑰](#)的金鑰 ID。此外部金鑰必須[滿足要求](#)，才能與 KMS 金鑰搭配使用。建立金鑰之後，便無法變更此值。

如果外部金鑰有多個 ID，請輸入外部金鑰存放區代理用來識別外部金鑰的金鑰 ID。

7. 確認您要在指定的外部金鑰存放區中建立 KMS 金鑰。
8. 選擇下一步。

此程序的其餘部分與[建立標準 KMS 金鑰](#)相同。

9. 輸入 KMS 金鑰的別名 (必需) 和描述 (選用)。
10. (選用)。在 Add Tags (新增標籤) 頁面，新增標籤來識別或分類 KMS 金鑰。

當您將標籤新增至 AWS 資源時，AWS 會產生成本分配報告，其中包含依標籤彙總的用量和成本。標籤也可以用來控制 KMS 金鑰的存取。如需標記 KMS 金鑰的詳細資訊，請參閱[中的標籤 AWS KMS](#) 和 [適用於的 ABAC AWS KMS](#)。

11. 選擇下一步。
12. 在 Key Administrators (金鑰管理員) 區段中，選取可管理 KMS 金鑰的 IAM 使用者和角色。如需詳細資訊，請參閱[允許金鑰管理員來管理 KMS 金鑰](#)。

Note

IAM 政策可授權其他 IAM 使用者和角色來使用 KMS 金鑰。

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 中的安全性最佳實務](#)。

13. (選用) 若要防止這些金鑰管理員刪除此 KMS 金鑰，請清除 Allow key administrators to delete this key (允許金鑰管理員刪除此金鑰) 核取方塊。

刪除 KMS 金鑰是一種破壞性和不可復原的操作，可可能導致密文無法復原。即使您擁有外部金鑰材料，也無法在外部金鑰存放區中重新建立對稱 KMS 金鑰。不過，刪除 KMS 金鑰不會影響其相關聯的外部金鑰。如需有關從外部金鑰存放區刪除 KMS 金鑰的資訊，請參閱[刪除金鑰的特殊考量](#)。

14. 選擇下一步。

15. 在此帳戶區段中，選取可在[密碼編譯操作](#)中使用 KMS 金鑰的 IAM AWS 帳戶使用者和角色。如需詳細資訊，請參閱[允許金鑰使用者使用 KMS 金鑰](#)。

 Note

IAM 政策可授權其他 IAM 使用者和角色來使用 KMS 金鑰。

IAM 最佳實務不建議使用具有長期憑證的 IAM 使用者。盡可能使用提供臨時憑證的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 中的安全性最佳實務](#)。

16. (選用) 您可以允許其他 AWS 帳戶使用此 KMS 金鑰進行密碼編譯操作。若要這樣做，請在頁面底部的其他 AWS 帳戶區段中，選擇新增另一個 AWS 帳戶並輸入外部帳戶的 AWS 帳戶 ID。若要新增多個外部帳戶，請重複此步驟。

 Note

其他的管理員 AWS 帳戶也必須透過為其使用者建立 IAM 政策來允許存取 KMS 金鑰。如需詳細資訊，請參閱[允許其他帳戶中的使用者使用 KMS 金鑰](#)。

17. 選擇下一步。

18. 檢閱您選擇的金鑰設定。您仍然可以返回並變更所有設定。

19. 完成時，請選擇 Finish (完成) 以建立金鑰。

方法 2：在客戶受管金鑰中開始

此程序與使用金鑰材料建立對稱加密 AWS KMS 金鑰的程序相同。但是，在此程序中，您可指定外部金鑰存放區的自訂金鑰存放區 ID 和外部金鑰的金鑰 ID。您也必須為外部金鑰存放區中的 KMS 金鑰指定[必要的屬性值](#)，例如金鑰規格和金鑰用途。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。

3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
4. 選擇建立金鑰。
5. 選擇 Symmetric (對稱)。
6. 在 Key usage (金鑰用途) 欄位中，系統會自動選取 Encrypt and decrypt (加密和解密) 選項。請勿變更該欄位。
7. 選擇 Advanced options (進階選項)。
8. 對於 Key material origin (金鑰材料來源)，選擇 External key store (外部金鑰存放區)。
9. 確認您要在指定的外部金鑰存放區中建立 KMS 金鑰。
10. 選擇下一步。
11. 選擇代表新 KMS 金鑰的外部金鑰存放區的資料列。

您無法選擇已中斷連接的外部金鑰存放區。若要連接已中斷連接的金鑰存放區，請選擇金鑰存放區名稱，然後在 Key store actions (金鑰存放區動作) 中選擇 Connect (連接)。如需詳細資訊，請參閱[使用 AWS KMS 主控台](#)。

12. 在[外部金鑰管理器](#)中輸入[外部金鑰](#)的金鑰 ID。此外部金鑰必須[滿足要求](#)，才能與 KMS 金鑰搭配使用。建立金鑰之後，便無法變更此值。

如果外部金鑰有多個 ID，請輸入外部金鑰存放區代理用來識別外部金鑰的金鑰 ID。

13. 選擇下一步。

此程序的其餘部分與[建立標準 KMS 金鑰](#)相同。

14. 輸入 KMS 金鑰的別名和選用描述。
15. (選用)。在 Add Tags (新增標籤) 頁面，新增標籤來識別或分類 KMS 金鑰。

當您將標籤新增至 AWS 資源時，AWS 會產生成本分配報告，其中包含依標籤彙總的用量和成本。標籤也可以用來控制 KMS 金鑰的存取。如需標記 KMS 金鑰的詳細資訊，請參閱[中的標籤](#)
[AWS KMS](#) 和 [適用於的 ABAC AWS KMS](#)。

16. 選擇下一步。
17. 在 Key Administrators (金鑰管理員) 區段中，選取可管理 KMS 金鑰的 IAM 使用者和角色。如需詳細資訊，請參閱[允許金鑰管理員來管理 KMS 金鑰](#)。

 Note

IAM 政策可授權其他 IAM 使用者和角色來使用 KMS 金鑰。

18. (選用) 若要防止這些金鑰管理員刪除此 KMS 金鑰，請清除 Allow key administrators to delete this key (允許金鑰管理員刪除此金鑰) 核取方塊。

刪除 KMS 金鑰是一種破壞性和不可復原的操作，可可能導致密文無法復原。即使您擁有外部金鑰材料，也無法在外部金鑰存放區中重新建立對稱 KMS 金鑰。不過，刪除 KMS 金鑰不會影響其相關聯的外部金鑰。如需從外部金鑰存放區刪除 KMS 金鑰的相關資訊，請參閱 [刪除 AWS KMS key](#)。

19. 選擇下一步。
20. 在此帳戶區段中，選取可在[密碼編譯操作](#)中使用 KMS 金鑰的 IAM AWS 帳戶使用者和角色。如需詳細資訊，請參閱[允許金鑰使用者使用 KMS 金鑰](#)。

 Note

IAM 政策可授權其他 IAM 使用者和角色來使用 KMS 金鑰。

21. (選用) 您可以允許其他 AWS 帳戶使用此 KMS 金鑰進行密碼編譯操作。若要這樣做，請在頁面底部的其他 AWS 帳戶區段中，選擇新增另一個 AWS 帳戶，然後輸入外部帳戶的 AWS 帳戶 ID。若要新增多個外部帳戶，請重複此步驟。

 Note

其他的管理員 AWS 帳戶也必須透過為其使用者建立 IAM 政策來允許存取 KMS 金鑰。如需詳細資訊，請參閱[允許其他帳戶中的使用者使用 KMS 金鑰](#)。

22. 選擇下一步。
23. 檢閱您選擇的金鑰設定。您仍然可以返回並變更所有設定。
24. 完成時，請選擇 Finish (完成) 以建立金鑰。

當處理程序成功時，畫面會在您選擇的外部金鑰存放區中顯示新的 KMS 金鑰。當您選擇新 KMS 金鑰的名稱或別名時，其詳細資訊頁面上的 Cryptographic configuration (密碼編譯組態) 標籤會顯示 KMS 金鑰的來源 (External key store (外部金鑰存放區))、名稱、ID、自訂金鑰存放區類型，以及外部金鑰的 ID、金鑰用途和狀態。如果程序失敗，則會出現錯誤訊息來描述失敗。若為，請參閱 [外部金鑰存放區故障診斷](#)。

i Tip

為了更輕鬆識別自訂金鑰存放區中的 KMS 金鑰，請在 Customer managed keys (客戶受管金鑰) 頁面上，新增要顯示的 Origin (來源) 和 Custom key store ID (自訂金鑰存放區 ID) 資料欄。若要變更資料表欄位，請選擇頁面右上角的齒輪圖示。如需詳細資訊，請參閱[自訂您的主控台檢視](#)。

使用 AWS KMS API

若要在外部金鑰存放區建立新的 KMS 金鑰，請使用 [CreateKey](#) 操作。下列是必要參數：

- Origin 值必須為 EXTERNAL_KEY_STORE。
- CustomKeyStoreId 參數可識別您的外部金鑰存放區。指定的外部金鑰存放區的 [ConnectionState](#) 必須為 CONNECTED。若要尋找 CustomKeyStoreId 和 ConnectionState，請使用 DescribeCustomKeyStores 操作。
- XksKeyId 參數可識別外部金鑰。此外部金鑰必須[滿足要求](#)，才能與 KMS 金鑰相關聯。

您也可以使用 CreateKey 操作的任何選用參數，例如使用 Policy 或 [Tags](#) (標籤) 參數。

i Note

請勿在 Description 或 Tags 欄位包含機密或敏感資訊。這些欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

本節中的範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

此範例命令使用 [CreateKey](#) 操作以在外部金鑰存放區建立 KMS 金鑰。回應包括 KMS 金鑰的屬性、外部金鑰存放區 ID 以及外部金鑰的 ID、用途和狀態。

執行此命令之前，請將範例自訂金鑰存放區 ID 換成有效的 ID。

```
$ aws kms create-key --origin EXTERNAL_KEY_STORE --custom-key-store-id cks-1234567890abcdef0 --xks-key-id bb8562717f809024 {
  "KeyMetadata": {
```

```
    "Arn": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "AWSAccountId": "111122223333",
    "CreationDate": "2022-12-02T07:48:55-07:00",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "CustomKeyStoreId": "cks-1234567890abcdef0",
    "Description": "",
    "Enabled": true,
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ],
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",
    "KeySpec": "SYMMETRIC_DEFAULT",
    "KeyState": "Enabled",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "MultiRegion": false,
    "Origin": "EXTERNAL_KEY_STORE",
    "XksKeyConfiguration": {
      "Id": "bb8562717f809024"
    }
  }
}
```


識別和檢視金鑰

您可以使用 [AWS 管理主控台](#) 或 [AWS Key Management Service \(AWS KMS\) API](#) 在每個 AWS KMS keys 帳戶和區域中檢視，包括您管理的 KMS 金鑰和由 管理的 KMS 金鑰 AWS。

主題

- [尋找金鑰 ID 和金鑰 ARN](#)
- [存取和列出 KMS 金鑰詳細資訊](#)
- [識別不同的金鑰類型](#)
- [自訂您的主控台檢視](#)
- [在 金鑰存放區中尋找 KMS 金鑰和 AWS CloudHSM 金鑰材料](#)

尋找金鑰 ID 和金鑰 ARN

若要識別 AWS KMS key，您可以使用 [金鑰 ID](#) 或 Amazon Resource Name ([金鑰 ARN](#))。在 [密碼編譯操作](#) 中，您也可以使用 [別名名稱](#) 或 [別名 ARN](#)。

您可以使用 [AWS KMS 主控台](#) 或 [ListKeys](#) 操作來識別帳戶和區域中每個 KMS 金鑰的金鑰 ID 和金鑰 ARN。

如需 支援之 KMS 金鑰識別符的詳細資訊 AWS KMS，請參閱 [金鑰識別符 \(KeyID\)](#)。如需尋找別名名稱和別名 ARN 的說明，請參閱 [尋找 KMS 金鑰的別名名稱和別名 ARN](#)。

使用 AWS KMS 主控台

1. 在 <https://console.aws.amazon.com/kms> 開啟 AWS KMS 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 若要檢視您所建立及管理帳戶中的金鑰，請在導覽窗格中選擇 Customer managed keys (客戶受管金鑰)。若要檢視帳戶中為您 AWS 建立和管理的金鑰，請在導覽窗格中選擇 AWS 受管金鑰。
4. 若要尋找 KMS 金鑰的 [金鑰 ID](#)，請查看以 KMS 金鑰別名開頭的資料列。

根據預設，Key ID (金鑰 ID) 資料行會出現在資料表中。如果 Key ID (金鑰 ID) 資料行並未出現在您的資料表內，請使用 [the section called “自訂您的主控台檢視”](#) 中說明的程序來還原。您也可以 在 KMS 金鑰的詳細資訊頁面上檢視 KMS 金鑰的金鑰 ID。

Customer managed keys				
			Key actions ▼	Create key
< 1 > ⚙				
☐	Aliases ▲	Key ID ▼	Status	Creation date
☐	key-test	1234abcd-12ab-34cd-56ef-1234567890ab	Enabled	Oct 19, 2018 12:43 PDT

5. 若要尋找 KMS 金鑰的 Amazon Resource Name (ARN)，請選擇金鑰 ID 或別名。General Configuration (一般組態) 區段中會顯示的[金鑰 ARN](#)。

General configuration		
Aliases key-test	Status Enabled	ARN arn:aws:kms:us-east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab
Description -	Creation date Nov 06, 2018 15:11 PST	

使用 AWS KMS API

若要尋找的[金鑰 ID](#)和[金鑰 ARN](#) AWS KMS key，請使用 [ListKeys](#) 操作。

ListKeys 操作會傳回發起人帳戶和區域中所有 KMS 金鑰的金鑰 ID 和 Amazon Resource Name (ARN)。

例如，呼叫 ListKeys 操作會傳回這個虛構帳戶中每個 KMS 金鑰的 ID 和 ARN。如需多種程式設計語言的範例，請參閱[ListKeys 搭配 AWS SDK 或 CLI 使用](#)。

```
$ aws kms list-keys
{
  "Keys": [
    {
      "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
      "KeyArn": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    {
      "KeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
```

```
"KeyArn": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321"
    }
  ]
}
```

存取和列出 KMS 金鑰詳細資訊

您可以使用 [AWS KMS 主控台](#) 或 [DescribeKey](#) 操作來存取和列出帳戶和區域中 KMS 金鑰的詳細資訊。

下列程序示範如何存取 KMS 金鑰詳細資訊，例如金鑰 ID、金鑰規格、金鑰用量等。

使用 AWS KMS 主控台

每個 KMS 金鑰的詳細資訊頁面會顯示 KMS 金鑰的屬性。這會根據不同 KMS 金鑰類型而有些微不同。

若要顯示 KMS 金鑰的詳細資訊，請在 AWS 受管金鑰 或客戶受管金鑰頁面上，選擇 KMS 金鑰的別名或金鑰 ID。

KMS 金鑰的詳細資訊頁面包含 General Configuration (一般組態) 區段，顯示 KMS 金鑰的基本屬性。它還包含您可以檢視和編輯 KMS 金鑰屬性的標籤，例如金鑰政策、密碼編譯組態、標籤、金鑰材料和輪換（適用於支援自動或隨需輪換的 KMS 金鑰）、地區性（適用於多區域金鑰）和公有金鑰（適用於非對稱 KMS 金鑰）。

Note

AWS KMS 主控台會顯示您在帳戶和區域中有權[檢視](#)的 KMS 金鑰。其他 中的 KMS 金鑰 AWS 帳戶 不會出現在主控台中，即使您具有檢視、管理和使用這些金鑰的許可。若要檢視其他帳戶中的 KMS 金鑰，請使用 [DescribeKey](#) 操作。

瀏覽 KMS 金鑰的金鑰詳細資訊頁面。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 若要檢視您所建立及管理帳戶中的金鑰，請在導覽窗格中選擇 Customer managed keys (客戶受管金鑰)。若要檢視帳戶中為您 AWS 建立和管理的金鑰，請在導覽窗格中選擇 AWS 受管金鑰。

4. 若要開啟金鑰詳細資訊頁面，請在金鑰資料表中選擇 KMS 金鑰的金鑰 ID 或別名。

如果 KMS 金鑰具有多個別名，則別名摘要 (+n more (+n 等)) 會顯示在其中一個別名的名稱旁邊。選擇別名摘要會直接帶您前往金鑰詳細資訊頁面上的 Aliases (別名) 索引標籤。

☰ [KMS](#) > [Customer managed keys](#) > Key ID: 6f84178a-007d-4501-8229-abc6af9e609c ⓘ ⓘ

6f84178a-007d-4501-8229-abc6af9e609c Key actions ▾ Edit

General configuration

Alias kms-test	Status Enabled	Creation date May 26, 2025 01:49 PDT
ARN  arn:aws:kms:us-east-1:849743133991:key/6f84178a-007d-4501-8229-abc6af9e609c	Description -	Regionality Single Region
Current key material ID f2e38520f5b8424d0ccd2dc64530a25e830926738224b0cb97d8e62302549f81		

Key policy | **Cryptographic configuration** | **Tags** | **Key material and rotations** | **Aliases**

Cryptographic configuration

Key Type Symmetric	Origin AWS KMS	Key Spec ⓘ SYMMETRIC_DEFAULT	Key Usage Encrypt and decrypt

以下清單描述了詳細顯示中的欄位，包括索引標籤中的欄位。其中有些欄位也可以在資料表顯示中做為資料行使用。

別名

其中：別名索引標籤

KMS 金鑰的易記名稱。您可以使用別名來識別主控台和部分 AWS KMS APIs 中的 KMS 金鑰。如需詳細資訊，請參閱 [中的別名 AWS KMS](#)。

別名索引標籤會顯示與 AWS 帳戶 和 區域中 KMS 金鑰相關聯的所有別名。

ARN

其中：一般組態區段

KMS 金鑰的 Amazon Resource Name (ARN)。該值會專屬識別 KMS 金鑰。您可以使用該值在 AWS KMS API 操作中識別 KMS 金鑰。

連線狀態

指示 [自訂金鑰存放區](#) 是否已連接至其備份金鑰存放區。當 KMS 金鑰已在自訂金鑰存放區中建立時，才會顯示此欄位。

如需此欄位中值的相關資訊，請參閱《AWS KMS API 參考》中的 [ConnectionState](#)。

Creation (建立) 日期

其中：一般組態區段

建立 KMS 金鑰的日期和時間。這個值會以裝置的本地時間顯示。時區不會依存區域。

與 Expiration (過期) 不同，建立指的只是 KMS 金鑰，而非其金鑰材料。

CloudHSM 叢集 ID

其中：密碼編譯組態索引標籤

叢集的 AWS CloudHSM 叢集 ID，其中包含 KMS 金鑰的金鑰材料。當 KMS 金鑰已在 [自訂金鑰存放區](#) 中建立時，才會顯示此欄位。

如果您選擇 CloudHSM 叢集 ID，它會在 AWS CloudHSM 主控台中開啟叢集頁面。

目前的金鑰材料

其中：一般組態區段

具有 AWS_KMS 原始伺服器的對稱加密金鑰支援自動和隨需輪換。具有 EXTERNAL 原始伺服器的對稱加密金鑰支援隨需輪換。這些金鑰可以有多個與金鑰相關聯的金鑰材料。最近輪換的金鑰材料可用於加密和解密。此金鑰材料會識別為目前的金鑰材料。其他金鑰資料只能用於解密。KMS 金鑰的自動或隨需金鑰輪換會變更其目前的金鑰材料。

自訂金鑰存放區 ID

其中：密碼編譯組態索引標籤

包含 KMS 金鑰的 [自訂金鑰存放區](#) ID。當 KMS 金鑰已在自訂金鑰存放區中建立時，才會顯示此欄位。

如果您選擇自訂金鑰存放區 ID，它會在 AWS KMS 主控台中開啟自訂金鑰存放區頁面。

自訂金鑰存放區名稱

其中：密碼編譯組態索引標籤

包含 KMS 金鑰的 [自訂金鑰存放區](#) 名稱。當 KMS 金鑰已在自訂金鑰存放區中建立時，才會顯示此欄位。

自訂金鑰存放區類型

其中：密碼編譯組態索引標籤

指示自訂金鑰存放區是 [AWS CloudHSM 金鑰存放區](#) 還是 [外部金鑰存放區](#)。當 KMS 金鑰已在 [自訂金鑰存放區](#) 中建立時，才會顯示此欄位。

Description

其中：一般組態區段

KMS 金鑰的簡短、選用描述，您可以撰寫和編輯。如要新增或更新客戶受管金鑰的描述，請在 General Configuration (一般組態) 上方，選擇 Edit (編輯)。

加密演算法

其中：密碼編譯組態索引標籤

列出可搭配 AWS KMS 中 KMS 金鑰使用的加密演算法。此欄位只有在 Key type (金鑰類型) 是 Asymmetric (非對稱) 且 Key usage (金鑰使用方式) 是 Encrypt and decrypt (加密及解密) 時才會出現。如需 AWS KMS 支援的加密演算法相關資訊，請參閱 [SYMMETRIC_DEFAULT 金鑰規格](#) 和 [用於加密和解密的 RSA 金鑰規格](#)。

過期日期

其中：金鑰材料索引標籤

KMS 金鑰之金鑰材料過期的日期和時間。此欄位只會針對具備 [匯入金鑰材料](#) 的 KMS 金鑰顯示，即 Origin (來源) 是 External (外部) 且 KMS 金鑰具有會過期的金鑰材料時。對稱加密金鑰可以有多個與其相關聯的金鑰材料。對於此類金鑰，此欄位會指出其中一個相關聯金鑰材料過期的最早日期和時間。

外部金鑰 ID

其中：密碼編譯組態索引標籤

與 [外部金鑰存放區](#) 中的 KMS 金鑰相關聯的 [外部金鑰](#) ID。此欄位只會針對外部金鑰存放區中的 KMS 金鑰顯示。

外部金鑰狀態

其中：密碼編譯組態索引標籤

[外部金鑰存放區代理](#)針對與 KMS 金鑰相關聯的[外部金鑰](#)所報告的最新狀態。此欄位只會針對外部金鑰存放區中的 KMS 金鑰顯示。

外部金鑰使用情況

其中：密碼編譯組態索引標籤

在與 KMS 金鑰相關聯的[外部金鑰](#)上啟用的密碼編譯操作。此欄位只會針對外部金鑰存放區中的 KMS 金鑰顯示。

金鑰政策

其中：金鑰政策索引標籤

與 [IAM 政策](#)和[授予](#)一同控制對 KMS 金鑰的存取。每個 KMS 金鑰都有一個金鑰政策。這是唯一的強制授權元素。如要變更客戶受管金鑰的金鑰政策，請在 Key policy (金鑰政策) 標籤上，選擇 Edit (編輯)。如需詳細資訊，請參閱[the section called “金鑰政策”](#)。

金鑰材料和輪換

其中：金鑰材料和輪換索引標籤

此標籤僅針對具有AWS_KMS原始伺服器的對稱加密金鑰（支援自動和隨需輪換）以及具有EXTERNAL原始伺服器的單一區域對稱加密金鑰（支援隨需輪換）顯示。

標籤有三個面板：

自動輪換：啟用和停用[客戶受管 KMS](#) 金鑰中金鑰材料的[自動輪換](#)。若要變更[客戶受管](#)金鑰的金鑰輪換狀態，請使用核取方塊。您不能啟用或停用 [AWS 受管金鑰](#) 中金鑰資料的輪換。AWS 受管金鑰 每年會自動輪換一次。

隨需輪換：啟動[客戶受管](#)金鑰中金鑰材料的[隨需輪換](#)。對於匯入的金鑰，必須已有PENDING_ROTATION狀態為的匯入金鑰材料，才能使用立即輪換選項。

金鑰材料：列出與 KMS 金鑰相關聯的所有金鑰材料。每個金鑰材料都有唯一的識別符，其資料列會顯示金鑰材料的其他資訊，例如金鑰材料可在 KMS 中使用的輪換日期。對於匯入的金鑰，每一列也具有動作功能表，可用於刪除特定金鑰材料或將其重新匯入 KMS 金鑰。

金鑰規格

其中：密碼編譯組態索引標籤

KMS 金鑰中的金鑰材料類型。AWS KMS 支援對稱加密 KMS 金鑰 (SYMMETRIC_DEFAULT)、不同長度的 HMAC KMS 金鑰、不同長度 RSA 金鑰的 KMS 金鑰，以及具有不同曲線的橢圓曲線金鑰。如需詳細資訊，請參閱[Key spec](#)。

Key type

其中：密碼編譯組態索引標籤

指出 KMS 金鑰是 Symmetric (對稱) 還是 Asymmetric (非對稱)。

金鑰用途

其中：密碼編譯組態索引標籤

指出 KMS 金鑰是用於 Encrypt and decrypt (加密及解密)、Sign and verify (簽署及驗證) 還是 Generate and verify MAC (產生及驗證 MAC)。如需詳細資訊，請參閱[Key usage](#)。

Origin

其中：密碼編譯組態索引標籤

KMS 金鑰之金鑰材料的來源。有效的值如下：

- AWS KMS 適用於 AWS KMS 產生的金鑰材料
- AWS CloudHSM 適用於 [AWS CloudHSM 金鑰存放區](#) 中的 KMS 金鑰
- External (外部) 適用於 [匯入金鑰材料](#) (BYOK)
- External key store (外部金鑰存放區) 適用於 [外部金鑰存放區](#) 中的 KMS 金鑰

MAC 演算法

其中：密碼編譯組態索引標籤

列出可搭配 AWS KMS 中 HMAC KMS 金鑰使用的 MAC 演算法。僅在 Key spec (金鑰規格) 為 HMAC 金鑰規格 (HMAC_*) 時，才會顯示此欄位。如需有關 AWS KMS 支援的 MAC 演算法的詳細資訊，請參閱 [HMAC KMS 金鑰的金鑰規格](#)。

主索引鍵

其中：區域性索引標籤

表示此 KMS 金鑰是 [多區域主要金鑰](#)。授權使用者可以使用此區段 [將主要金鑰變更](#) 為不同的相關多區域金鑰。僅在 KMS 金鑰為多區域主要金鑰時，才會顯示此欄位。

公有金鑰

其中：公有金鑰索引標籤

顯示非對稱 KMS 金鑰的公有金鑰。獲得授權的使用者可以使用此標籤來 [複製及下載公有金鑰](#)。

區域性

其中：「一般組態」區段和「區域性」索引標籤

指出 KMS 金鑰是否為單一區域金鑰、[多區域主要金鑰](#)，或[多區域複本金鑰](#)。僅在 KMS 金鑰為多區域金鑰時，才會顯示此欄位。

相關的多區域金鑰

其中：區域性索引標籤

顯示所有相關[多區域主要金鑰和複本金鑰](#)，但目前的 KMS 金鑰除外。僅在 KMS 金鑰為多區域金鑰時，才會顯示此欄位。

在主要金鑰的相關多區域金鑰區段，授權使用者可以[建立新的複本金鑰](#)。

複本金鑰

其中：區域性索引標籤

表示此 KMS 金鑰是[多區域複本金鑰](#)。僅在 KMS 金鑰為多區域複本金鑰時，才會顯示此欄位。

簽署演算法

其中：密碼編譯組態索引標籤

列出可搭配 AWS KMS 中 KMS 金鑰使用的簽署演算法。此欄位只有在 Key type (金鑰類型) 是 Asymmetric (非對稱) 且 Key usage (金鑰使用方式) 是 Sign and verify (簽署及驗證) 時才會出現。如需 AWS KMS 支援的簽署演算法的相關資訊，請參閱 [用於簽署和驗證的 RSA 金鑰規格](#) 和 [橢圓曲線金鑰規格](#)。

狀態

其中：一般組態區段

KMS 金鑰的金鑰狀態。只有在狀態為 Enabled (啟用) 時，您才能在[密碼編譯操作](#)中使用 KMS 金鑰。如需每個 KMS 金鑰狀態的詳細說明及其對您可以在 KMS 金鑰上執行操作所帶來的影響，請參閱 [金鑰的 AWS KMS 金鑰狀態](#)。

Tags (標籤)

其中：標籤索引標籤

描述 KMS 金鑰的選用鍵值對。若要新增或變更 KMS 金鑰的標籤，請在 Tags (標籤) 索引標籤上，選擇 Edit (編輯)。

當您將標籤新增至 AWS 資源時，AWS 會產生成本分配報告，其中包含依標籤彙總的用量和成本。標籤也可以用來控制 KMS 金鑰的存取。如需標記 KMS 金鑰的詳細資訊，請參閱 [中的標籤 AWS KMS](#) 和 [適用於的 ABAC AWS KMS](#)。

使用 AWS KMS API

[DescribeKey](#) 操作會傳回指定之 KMS 金鑰的詳細資訊。若要識別 KMS 金鑰，請使用[金鑰 ID](#)、[金鑰 ARN](#)、[別名名稱](#)或[別名 ARN](#)。

與 [ListKeys](#) 操作不同，該操作僅顯示呼叫者帳戶和區域中的 KMS 金鑰，授權用戶可以使用 [DescribeKey](#) 操作以獲取有關其他帳戶中 KMS 金鑰的詳細資訊。

Note

[DescribeKey](#) 回應包括具有相同值的 `KeySpec` 和 `CustomerMasterKeySpec` 成員。已取代 `CustomerMasterKeySpec` 成員。

例如，呼叫 [DescribeKey](#) 會傳回對稱加密 KMS 金鑰的相關資訊。回應中的欄位會隨 [AWS KMS key 規格](#)、[金鑰狀態](#)和[金鑰材料來源](#)而有所不同。如需多種程式設計語言的範例，請參閱[DescribeKey 搭配 AWS SDK 或 CLI 使用](#)。

```
$ aws kms describe-key --key-id 1234abcd-12ab-34cd-56ef-1234567890ab

{
  "KeyMetadata": {
    "Origin": "AWS_KMS",
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "Description": "",
    "KeyManager": "CUSTOMER",
    "Enabled": true,
    "KeySpec": "SYMMETRIC_DEFAULT",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "KeyState": "Enabled",
    "CreationDate": 1499988169.234,
    "MultiRegion": false,
    "Arn": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "AWSAccountId": "111122223333",
    "EncryptionAlgorithms": [
```

```

        "SYMMETRIC_DEFAULT"
    ],
    "CurrentKeyMaterialId":
    "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
  }
}

```

這個範例會在用於簽署和驗證的非對稱 KMS 金鑰上呼叫 `DescribeKey` 操作。回應包含 AWS KMS 針對此 KMS 金鑰支援的簽署演算法。

```

$ aws kms describe-key --key-id 0987dcba-09fe-87dc-65ba-ab0987654321

{
  "KeyMetadata": {
    "KeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
    "Origin": "AWS_KMS",
    "Arn": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321",
    "KeyState": "Enabled",
    "KeyUsage": "SIGN_VERIFY",
    "CreationDate": 1569973196.214,
    "Description": "",
    "KeySpec": "ECC_NIST_P521",
    "CustomerMasterKeySpec": "ECC_NIST_P521",
    "AWSAccountId": "111122223333",
    "Enabled": true,
    "MultiRegion": false,
    "KeyManager": "CUSTOMER",
    "SigningAlgorithms": [
      "ECDSA_SHA_512"
    ]
  }
}

```

識別不同的金鑰類型

下列主題說明如何在 AWS KMS 主控台和 [DescribeKey](#) 回應中識別不同的金鑰類型。

如需導覽至 KMS 金鑰詳細資訊頁面上密碼編譯組態索引標籤的說明，請參閱 [the section called “存取和列出 KMS 金鑰詳細資訊”](#)。

主題

- [識別非對稱 KMS 金鑰](#)
- [識別 HMAC KMS 金鑰](#)
- [識別多區域 KMS 金鑰](#)
- [使用匯入的金鑰材料識別 KMS 金鑰](#)
- [識別金鑰存放區中的 KMS AWS CloudHSM 金鑰](#)
- [識別外部金鑰存放區中的 KMS 金鑰](#)

識別非對稱 KMS 金鑰

在 AWS KMS 主控台中

客戶受管金鑰資料表的金鑰類型欄會顯示每個 KMS 金鑰是對稱或非對稱。您可以依金鑰類型值篩選資料表，以僅顯示非對稱 KMS 金鑰。如需更多資訊，請參閱[the section called “排序和篩選您的 KMS 金鑰”](#)。

KMS 金鑰詳細資訊頁面上的密碼編譯組態索引標籤會顯示金鑰類型，指出金鑰是對稱或非對稱。它也會顯示金鑰用量，指出您的非對稱 KMS 金鑰是否用於加密和解密、簽署和驗證，還是衍生共用秘密。

在 DescribeKey 回應中

當您在非對稱 KMS 金鑰上呼叫 DescribeKey 操作時，回應會包含 KeySpec 和 KeyUsage 值，可用於判斷 KMS 金鑰是對稱或非對稱。

如果 KeySpec 值為 SYMMETRIC_DEFAULT，則金鑰為對稱加密 KMS 金鑰。如需非對稱金鑰規格的詳細資訊，請參閱 [金鑰規格參考](#)。

如果 KeyUsage 值為 SIGN_VERIFY 或 KEY_AGREEMENT，則金鑰為非對稱 KMS 金鑰。

DescribeKey 操作也會傳回非對稱 KMS 金鑰的下列詳細資訊。

- 對於 KeyUsage 值為 的非對稱 KMS 金鑰 ENCRYPT_DECRYPT，操作會傳回 EncryptionAlgorithms，列出金鑰的有效加密演算法。
- 對於 KeyUsage 值為 的非對稱 KMS 金鑰 SIGN_VERIFY，操作會傳回 SigningAlgorithms，列出金鑰的有效簽署演算法。
- 對於 KeyUsage 值為 的非對稱 KMS 金鑰 KEY_AGREEMENT，操作會傳回 KeyAgreementAlgorithms，其中列出金鑰的有效金鑰協議演算法。

如需非對稱 KMS 金鑰的詳細資訊，請參閱 [the section called “非對稱金鑰。”](#)。

識別 HMAC KMS 金鑰

在 AWS KMS 主控台中

HMAC KMS 金鑰包含在客戶受管金鑰資料表中，但您無法依識別 HMAC 金鑰的金鑰規格或金鑰用量值來排序或篩選此資料表。若要更容易地找到您的 HMAC 金鑰，請為其指派一個獨特的別名或標籤。然後，您便可以依別名或標籤進行排序或篩選。

KMS 金鑰詳細資訊頁面上的密碼編譯組態索引標籤會顯示金鑰類型，指出金鑰是對稱或非對稱。HMAC KMS 金鑰為對稱金鑰。密碼編譯組態索引標籤也會顯示金鑰用量。對於 HMAC KMS 金鑰，金鑰用量值一律為產生和驗證 MAC。

在 DescribeKey 回應中

當您在 HMAC KMS 金鑰上呼叫 DescribeKey 操作時，回應會包含 KeySpec 和 KeyUsage 值。對於 HMAC KMS 金鑰，金鑰用量值一律為 GENERATE_VERIFY_MAC，而金鑰規格值一律以開頭 HMAC_。

如需 HMAC KMS 金鑰的詳細資訊，請參閱 [the section called “HMAC 金鑰”](#)。

識別多區域 KMS 金鑰

在 AWS KMS 主控台中

客戶受管金鑰資料表只會顯示所選區域中的 KMS 金鑰。您可以檢視所選區域中的多區域主要金鑰和複本金鑰。若要變更 AWS 區域，請使用主控台右上角的區域選擇器。

若要更輕鬆地識別客戶受管金鑰資料表中的多區域金鑰，請將區域性資料欄新增至資料表。如需協助，請參閱 [the section called “自訂您的 KMS 金鑰資料表”](#)。

多區域 KMS 金鑰的詳細資訊頁面包含區域性索引標籤。Regionality (區域性) 索引標籤包括「變更主要區域」和「建立新的複本金鑰」按鈕。(複本金鑰的「區域性」索引標籤沒有按鈕)。Related multi-Region keys (相關的多區域金鑰) 區段會列出所有與目前區域相關的多區域金鑰。如果目前金鑰是複本金鑰，則此清單會包含主要金鑰。

如果您從相關多區域金鑰資料表中選擇相關的多區域金鑰，AWS KMS 主控台會變更為所選金鑰的區域，並開啟金鑰的詳細資訊頁面。例如，如果您從下面的相關多區域金鑰範例區段中選擇 sa-east-1 區域中的複本金鑰，則 AWS KMS 主控台會變更為 sa-east-1 區域，以顯示該複本金鑰的詳細資訊頁面。您可以執行這項操作來檢視複本金鑰的別名或金鑰政策。若要再次變更區域，請使用頁面右上角的區域選取器。

在 DescribeKey 回應中

根據預設，AWS KMS API 操作為區域性，只會傳回目前或指定區域中的資源。但是，當您在多區域 KMS 金鑰上呼叫 DescribeKey 操作時，回應會在 MultiRegionConfiguration 元素的其他 AWS 區域中包含所有相關多區域金鑰。

如需多區域 KMS 金鑰的詳細資訊，請參閱 [the section called “多區域金鑰”](#)。

使用匯入的金鑰材料識別 KMS 金鑰

在 AWS KMS 主控台中

若要更輕鬆地在客戶受管金鑰資料表中識別具有匯入金鑰資料的 KMS 金鑰，請將原始伺服器欄新增至資料表。來源欄可讓您輕鬆識別具外部 (匯入金鑰資料) 來源屬性值的 KMS 金鑰。如需協助，請參閱 [the section called “自訂您的 KMS 金鑰資料表”](#)。

KMS 金鑰詳細資訊頁面上的密碼編譯組態索引標籤會顯示 Origin，以識別 KMS 金鑰的金鑰材料來源。對於具有匯入金鑰材料的 KMS 金鑰，原始伺服器值一律為外部 (匯入金鑰材料)。詳細資訊頁面也包含金鑰材料索引標籤，可提供匯入金鑰材料的詳細資訊。具有 EXTERNAL 原始伺服器的對稱加密金鑰支援隨需輪換，並且可以具有與其相關聯的多個金鑰材料。對於此類金鑰，標籤會標記為金鑰材料和輪換。

在 DescribeKey 回應中

當您使用匯入的金鑰材料呼叫 KMS 金鑰上的 DescribeKey 操作時，回應會包含 Origin、ExpirationModel 和 ValidTo 值。對於具有匯入金鑰材料的 KMS 金鑰，原始伺服器值一律為 EXTERNAL。ExpirationModel 值表示金鑰材料是否設定為過期，ValidTo 值表示金鑰材料何時過期。當多個金鑰材料與金鑰相關聯時，ValidTo 值會指出所有金鑰材料的最早到期時間 (除了一個待定輪換) ExpirationModel，並且 DOES_NOT_EXPIRE 只有在這些金鑰材料都未設定為過期時，才會設定為。如需詳細資訊，請參閱 [設定到期時間 \(選用\)](#)。

如需匯入金鑰材料之 KMS 金鑰的詳細資訊，請參閱 [the section called “匯入的金鑰資料”](#)。

識別金鑰存放區中的 KMS AWS CloudHSM 金鑰

在 AWS KMS 主控台中

若要更輕鬆地識別客戶受管金鑰資料表中 AWS CloudHSM 金鑰存放區中的 KMS 金鑰，請將 Origin 欄新增至資料表。Origin (來源) 欄可讓您輕鬆識別具有 AWS CloudHSM 來源屬性值的 KMS 金鑰。如需協助，請參閱 [the section called “自訂您的 KMS 金鑰資料表”](#)。

KMS 金鑰詳細資訊頁面上的密碼編譯組態索引標籤會顯示 Origin，以識別 KMS 金鑰的金鑰材料來源。對於金鑰存放區中的 KMS AWS CloudHSM 金鑰，原始伺服器值一律為 AWS CloudHSM。

對於 AWS CloudHSM 金鑰存放區中的 KMS 金鑰，密碼編譯組態索引標籤包含額外的區段，自訂金鑰存放區，提供與 KMS 金鑰相關聯的 AWS CloudHSM 金鑰存放區和 AWS CloudHSM 叢集的相關資訊。

在 DescribeKey 回應中

當您在金鑰存放區中的 KMS AWS CloudHSM 金鑰上呼叫 DescribeKey 操作時，回應會包含 Origin，可識別金鑰材料的來源。對於金鑰存放區中的 KMS AWS CloudHSM 金鑰，原始伺服器值一律為 AWS_CLOUDHSM。操作也會傳回金鑰 AWS CloudHSM 存放區中 KMS 金鑰的下列特殊欄位：

- CloudHsmClusterId
- CustomKeyStoreId

如需 AWS CloudHSM 金鑰存放區的詳細資訊，請參閱 [the section called “AWS CloudHSM 金鑰存放區”](#)。

識別外部金鑰存放區中的 KMS 金鑰

在 AWS KMS 主控台中

若要更輕鬆地識別客戶受管金鑰資料表中外部的金鑰存放區中的 KMS 金鑰，請將原始伺服器欄新增至您的資料表。Origin 欄可讓您輕鬆識別具有外部金鑰存放區原始屬性值的 KMS 金鑰。如需協助，請參閱 [the section called “自訂您的 KMS 金鑰資料表”](#)。

KMS 金鑰詳細資訊頁面上的密碼編譯組態索引標籤會顯示 Origin，以識別 KMS 金鑰的金鑰材料來源。對於外部金鑰存放區中的 KMS 金鑰，原始伺服器值一律為外部金鑰存放區。

對於外部金鑰存放區中的 KMS 金鑰，密碼編譯組態索引標籤包含兩個額外區段：自訂金鑰存放區和外部金鑰。自訂金鑰存放區資料表提供與 KMS 金鑰相關聯之外部金鑰存放區的相關資訊。外部金鑰資料表只會針對 AWS KMS 外部金鑰存放區中的 KMS 金鑰顯示在主控台中。它提供與 KMS 金鑰相關聯的外部金鑰的資訊。[外部金鑰](#)是外部的密碼編譯金鑰 AWS，可做為外部金鑰存放區中 KMS 金鑰的金鑰材料。當您使用 KMS 金鑰進行加密或解密時，[外部金鑰管理器](#)將使用指定的外部金鑰執行操作。

下列值會顯示在 External key (外部金鑰) 區段中。

外部金鑰 ID

外部金鑰在其外部金鑰管理器中的識別符。這是外部金鑰存放區代理用來識別外部金鑰的值。您可以在建立 KMS 金鑰時指定外部金鑰 ID，並且無法變更它。如果用來建立 KMS 金鑰的外部金鑰 ID 值變更或變得無效，您必須[排程要刪除的 KMS 金鑰](#)，並使用正確的外部金鑰 ID 值[建立新的 KMS 金鑰](#)。

在 DescribeKey 回應中

當您在外部金鑰存放區中的 KMS 金鑰上呼叫 DescribeKey 操作時，回應會包含 Origin，可識別金鑰材料的來源。對於金鑰存放區中的 KMS AWS CloudHSM 金鑰，原始伺服器值一律為 EXTERNAL_KEY_STORE。操作也會傳回 CustomKeyId 元素，識別與 KMS 金鑰相關聯的外部金鑰存放區。

如需外部金鑰存放區的詳細資訊，請參閱 [the section called “外部金鑰存放區”](#)。

自訂您的主控台檢視

您可以自訂 AWS KMS 主控台的檢視，讓您更輕鬆地尋找 KMS 金鑰。自訂顯示在 AWS 受管金鑰和客戶受管金鑰頁面上的資料表，以顯示您最需要的資訊，或排序和篩選資料表中傳回的 KMS 金鑰。

主題

- [排序和篩選您的 KMS 金鑰](#)
- [自訂您的 KMS 金鑰資料表](#)

排序和篩選您的 KMS 金鑰

若要更容易地在主控台中找到您的 KMS 金鑰，則您可以排序和篩選金鑰資料表。

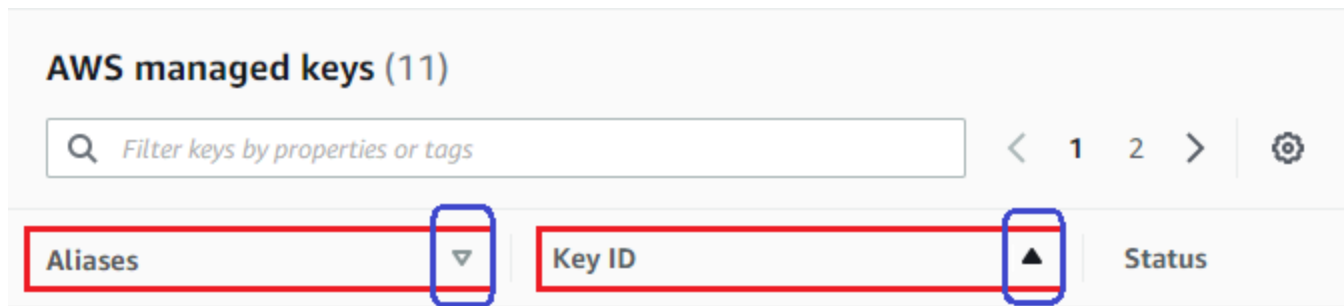
Sort

您可以根據資料欄的值，以遞增或遞減順序排序 KMS 金鑰。這項功能會排序資料表中的所有 KMS 金鑰，即使 KMS 金鑰沒有在目前的資料表頁面上顯示也一樣。

可排序的資料行在其資料行名稱的旁邊會有一個箭頭。在 AWS 受管金鑰 頁面上，您可以依 Aliases (別名) 或 Key ID (金鑰 ID) 排序。在 Customer managed keys (客戶受管金鑰) 頁面上，您可以根據 Aliases (別名)、Key ID (金鑰 ID) 或 Key type (金鑰類型) 排序。

如要依照遞增順序排序，請選擇資料行的標頭，直到箭頭指向上方。如要依照遞減順序排序，請選擇資料行的標頭，直到箭頭指向下方。您一次只能依據一欄來排序。

例如，您可以根據金鑰 ID，以遞增順序排序 KMS 金鑰，而非使用預設的別名。



當您在 Customer managed keys (客戶受管金鑰) 頁面上依 Key type (金鑰類型) 按遞增順序排列 KMS 金鑰時，所有非對稱金鑰會顯示在所有對稱金鑰之前。

篩選條件

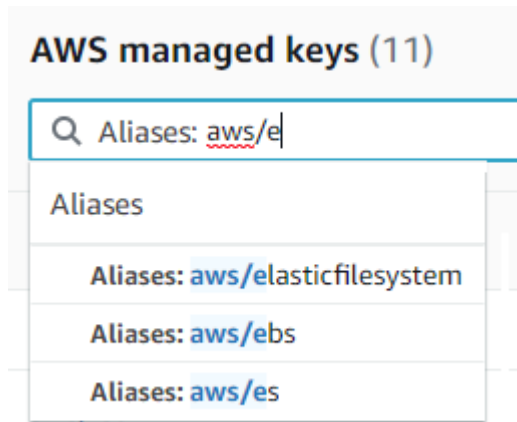
您可以依據 KMS 金鑰的屬性值或標籤來篩選 KMS 金鑰。篩選條件適用於所有資料表中的 KMS 金鑰，即使 CMK 沒有出現在目前的資料表頁面上也一樣。篩選條件不區分大小寫。

可篩選的屬性會列在篩選條件方塊中。在 AWS 受管金鑰 頁面上，您可以依別名和金鑰 ID 進行篩選。在 Customer managed keys (客戶受管金鑰) 頁面上，您可以根據別名、金鑰 ID 和金鑰類型以及根據標籤進行篩選。

- 在 AWS 受管金鑰 頁面上，您可以依別名和金鑰 ID 進行篩選。
- 在 Customer managed keys (客戶受管金鑰) 頁面上，您可以根據標籤，或根據別名、金鑰 ID 和金鑰類型或區域性屬性進行篩選。

若要根據屬性值進行篩選，請選擇篩選條件、選擇屬性名稱，然後從實際屬性值的清單中選擇。若要依標籤進行篩選，請選擇標籤索引鍵，然後從實際標籤值清單中選擇。在選擇屬性或標籤索引鍵後，您也可以輸入全部或部分的屬性值或標籤值。在您做出選擇之前，您將會看到結果的預覽。

例如，如要顯示別名名稱包含 aws/e 的 KMS 金鑰，請選擇篩選條件方塊、選擇 Alias (別名)、輸入 aws/e，然後按 Enter 或 Return 來新增篩選條件。



建議的 KMS 金鑰資料表篩選條件

篩選非對稱 KMS 金鑰

若要在 Customer managed keys (客戶受管金鑰) 頁面上僅顯示非對稱 KMS 金鑰，請按一下篩選條件方塊，選擇 Key type (金鑰類型)，然後選擇 Key type: Asymmetric (金鑰類型：不對稱)。只有當資料表中有非對稱 KMS 金鑰時，才會顯示 Asymmetric (非對稱) 選項。

篩選多區域金鑰

若只要顯示多區域金鑰，請在 Customer managed keys (客戶受管金鑰) 頁面上，選擇篩選條件方塊、選擇 Regionality (區域性)，然後選擇 Regionality: Multi-Region (區域性：多區域)。Multi-Region (多區域) 選項僅當資料表中有多區域金鑰時才會顯示。

篩選標籤

若只要顯示具有特定標籤的 KMS 金鑰，請選擇篩選方塊、選擇標籤索引鍵，然後從實際標籤值中選擇。您也可以輸入全部或部分的標籤值。

產生的資料表會顯示具有所選標籤的所有 KMS 金鑰。但不會顯示標籤。若要查看標籤，請選擇 KMS 金鑰的金鑰 ID 或別名，然後在其詳細資訊頁面上選擇 Tags (標籤) 索引標籤。索引標籤會顯示在 General Configuration (一般組態) 區段下。

此篩選條件需要標籤索引鍵和標籤值。只輸入標籤索引鍵或只輸入其值，就無法找到 KMS 金鑰。若要依標籤索引鍵或值的全部或部分篩選標籤，請使用 [ListResourceTags](#) 操作來取得標記的 KMS 金鑰，然後使用程式設計語言的篩選功能。

依文字篩選

若要搜尋文字，請在篩選條件方塊中輸入全部或部分的別名、金鑰 ID、金鑰類型或標籤索引鍵。(選取標籤索引鍵後，您可以搜尋標籤值)。在您做出選擇之前，您將會看到結果的預覽。

例如，若要顯示在其標籤索引鍵中具有 `test` 的 KMS 金鑰或具有可篩選屬性的 KMS 金鑰，請輸入在篩選條件方塊中輸入 `test`。預覽會顯示篩選條件會選取的 KMS 金鑰。在此案例中，`test` 只會出現在 Alias (別名) 屬性中。

自訂您的 KMS 金鑰資料表

您可以自訂出現在 和 中 AWS 受管金鑰客戶受管金鑰頁面上的資料表 AWS 管理主控台，以符合您的需求。您可以選擇資料表欄、每個頁面上 AWS KMS keys 的數目 (頁面大小)，以及文字包裝。您選擇的組態會在您確認時儲存，並會在您開啟頁面時重新套用。

若要自訂您的 KMS 金鑰資料表

1. 在 AWS 受管金鑰 或 Customer managed keys (客戶受管金鑰) 頁面上，選擇位於頁面右上角的設定圖示



)。

2. 在 Preferences (偏好設定) 頁面上，選擇您偏好的設定，然後選擇 Confirm (確認)。

請考慮使用 Page size (頁面大小) 設定來增加每個頁面上顯示的 KMS 金鑰數量，特別是如果您通常使用容易捲動的裝置的話。

您顯示的資料資料行可能會因資料表、您的任務角色和帳戶及區域中的 KMS 金鑰類型而有所不同。下表提供了一些建議的組態。如需資料行的說明，請參閱 [使用 AWS KMS 主控台](#)。

建議的 KMS 金鑰資料表組態

您可以自訂 KMS 金鑰資料表中顯示的欄位，以顯示您需要的 KMS 金鑰相關資訊。

AWS 受管金鑰

根據預設，AWS 受管金鑰 資料表會顯示 Aliases (別名)、Key ID (金鑰 ID) 和 Status (狀態) 資料欄。這些資料行適合大多數的使用案例。

對稱加密 KMS 金鑰

如果只搭配 AWS KMS 產生的金鑰資料使用對稱加密 KMS 金鑰，Aliases (別名)、Key ID (金鑰 ID)、Status (狀態) 和 Creation date (建立日期) 資料欄可能最為有用。

非對稱 KMS 金鑰

如果使用非對稱 KMS 金鑰，則除了 Aliases (別名)、Key ID (金鑰 ID) 和 Status (狀態) 資料欄外，請考慮新增 Key type (金鑰類型)、Key spec (金鑰規格) 和 Key usage (金鑰使用情形) 資料欄。這些資料行會顯示 KMS 金鑰是對稱還是非對稱、金鑰材料的類型，以及 KMS 金鑰可用於加密還是簽署。

HMAC KMS 金鑰

如果使用 HMAC KMS 金鑰，則除了 Aliases (別名)、Key ID (金鑰 ID) 和 Status (狀態) 資料欄外，還可以考慮新增 Key spec (金鑰規格) 和 Key usage (金鑰用途) 資料欄。這些資料欄會顯示 KMS 金鑰是否為 HMAC 金鑰。由於您無法依金鑰規格或金鑰用量排序 KMS 金鑰，請使用別名和標籤來識別 HMAC 金鑰，然後使用 AWS KMS 主控台的[篩選功能](#)來依別名或標籤進行篩選。

匯入的金鑰資料

如果您的 KMS 金鑰具有[匯入的金鑰材料](#)，請考慮新增 Origin (來源) 和 Expiration date (過期日期) 資料欄。這些資料欄會顯示 KMS 金鑰中的金鑰材料是否由 匯入或產生，AWS KMS 以及金鑰材料何時過期，如果有的話。Creation date (建立日期) 欄位會顯示建立 KMS 金鑰的日期 (沒有金鑰材料時)。其不會反映任何金鑰材料的特性。

自訂金鑰存放區中的金鑰

如果您在[自訂金鑰存放區](#)中擁有 KMS 金鑰，則請考慮新增 Origin (來源) 和 Custom key store ID (自訂金鑰存放區 ID) 資料欄。這些資料欄會顯示 KMS 金鑰位於自訂金鑰存放區中、顯示自訂金鑰存放區類型，並識別自訂金鑰存放區。

多區域金鑰

如果您擁有[多區域金鑰](#)，請考慮新增 Regionality (區域性) 資料欄。這會顯示 KMS 金鑰是否為單一區域金鑰、[多區域主要金鑰](#)，或[多區域複本金鑰](#)。

在金鑰存放區中尋找 KMS 金鑰和 AWS CloudHSM 金鑰材料

如果您管理 AWS CloudHSM 金鑰存放區，您可能需要識別每個 AWS CloudHSM 金鑰存放區中的 KMS 金鑰。例如，您可能需要執行下列一些任務。

- 在 AWS CloudTrail 日誌中追蹤 AWS CloudHSM 金鑰存放區中的 KMS 金鑰。
- 預測中斷連接金鑰存放區對 KMS AWS CloudHSM 金鑰的影響。
- 先排程刪除 KMS 金鑰，再刪除 AWS CloudHSM 金鑰存放區。

此外，您可能想要識別 AWS CloudHSM 叢集中做為 KMS 金鑰金鑰材料的金鑰。雖然會 AWS KMS 管理 KMS 金鑰和金鑰材料，但您仍保有管理 AWS CloudHSM 叢集的控制權和責任，以及 HSMs 中的 HSMs 和備份和金鑰。您可能需要識別金鑰，以稽核金鑰材料、避免意外刪除它，或在刪除 KMS 金鑰後從 HSM 和叢集備份中刪除它。

金鑰存放區中 KMS 金鑰的所有 AWS CloudHSM 金鑰材料都由 [kmsuser 加密使用者](#) (CU) 擁有。會將金鑰標籤屬性 AWS KMS 設定為 KMS 金鑰的 Amazon Resource Name (ARN) AWS CloudHSM 中只能檢視。

若要尋找 KMS 金鑰和金鑰材料，請使用下列任何技巧。

- [在 金鑰存放區中尋找 KMS AWS CloudHSM 金鑰](#) — 如何識別一個或所有金鑰存放區中的 KMS AWS CloudHSM 金鑰。
- [尋找 金鑰存放區的所有 AWS CloudHSM 金鑰](#) — 如何在叢集內尋找金鑰，這些金鑰全部作為 AWS CloudHSM 金鑰存放區中的 KMS 金鑰的金鑰材料。
- [尋找 KMS AWS CloudHSM 金鑰的金鑰](#) — 如何在叢集中尋找做為金鑰存放區中特定 KMS 金鑰之金鑰材料的 AWS CloudHSM 金鑰。
- [尋找 金鑰的 KMS AWS CloudHSM 金鑰](#) — 如何尋找叢集中特定金鑰的 KMS 金鑰。

在 金鑰存放區中尋找 KMS AWS CloudHSM 金鑰

如果您管理 AWS CloudHSM 金鑰存放區，您可能需要識別每個 AWS CloudHSM 金鑰存放區中的 KMS 金鑰。您可以使用此資訊來追蹤 AWS CloudTrail 日誌中的 KMS 金鑰操作、預測在 KMS 金鑰上中斷連接自訂金鑰存放區的效果，或在刪除金鑰存放區之前排程刪除 KMS AWS CloudHSM 金鑰。

在 金鑰存放區中尋找 KMS AWS CloudHSM 金鑰（主控台）

若要尋找特定金鑰存放區中的 KMS AWS CloudHSM 金鑰，請在客戶受管金鑰頁面上，檢視自訂金鑰存放區名稱或自訂金鑰存放區 ID 欄位中的值。若要識別任何金鑰存放區中的 KMS AWS CloudHSM 金鑰，請尋找原始伺服器值為 `aws-kms` 的 KMS 金鑰 AWS CloudHSM。若要將選用欄新增到畫面上，請選擇頁面右上角的齒輪圖示。

在 金鑰存放區 (API) 中尋找 KMS AWS CloudHSM 金鑰

若要在金鑰存放區中尋找 KMS AWS CloudHSM 金鑰，請使用 [ListKeys](#) 和 [DescribeKey](#) 操作，然後依 `CustomKeyStoreId` 值篩選。執行下列範例之前，請將虛構的自訂金鑰存放區 ID 值取代為有效的值。

Bash

若要在特定金鑰存放區中尋找 KMS AWS CloudHSM 金鑰，請取得帳戶和區域中的所有 KMS 金鑰。然後按自訂金鑰存放區的 ID 進行篩選。

```
for key in $(aws kms list-keys --query 'Keys[*].KeyId' --output text) ;
do aws kms describe-key --key-id $key |
grep '"CustomKeyStoreId": "cks-1234567890abcdef0"' --context 100; done
```

若要在帳戶和區域的任何 AWS CloudHSM 金鑰存放區中取得 KMS 金鑰，請搜尋值 CustomKeyStoreType 為 AWS_CloudHSM。

```
for key in $(aws kms list-keys --query 'Keys[*].KeyId' --output text) ;
do aws kms describe-key --key-id $key |
grep '"CustomKeyStoreType": "AWS_CloudHSM"' --context 100; done
```

PowerShell

若要在特定金鑰存放區中尋找 KMS AWS CloudHSM 金鑰，請使用 [Get-KmsKeyList](#) 和 [Get-KmsKey](#) cmdlet 取得帳戶和區域中的所有 KMS 金鑰。然後按自訂金鑰存放區的 ID 進行篩選。

```
PS C:\> Get-KMSKeyList | Get-KMSKey | where CustomKeyStoreId -eq
'cks-1234567890abcdef0'
```

若要在帳戶和區域的任何 AWS CloudHSM 金鑰存放區中取得 KMS 金鑰，請篩選 CustomKeyStoreType 值 AWS_CLOUDHSM。

```
PS C:\> Get-KMSKeyList | Get-KMSKey | where CustomKeyStoreType -eq 'AWS_CLOUDHSM'
```

尋找 金鑰存放區的所有 AWS CloudHSM 金鑰

您可以識別 AWS CloudHSM 叢集中做為金鑰存放區金鑰材料的 AWS CloudHSM 金鑰。若要這樣做，請使用 CloudHSM CLI 中的 [金鑰清單](#) 命令。

您也可以使用金鑰清單命令來尋找 AWS KMS 金鑰 AWS CloudHSM 的。當在 AWS CloudHSM 叢集中 AWS KMS 建立 KMS 金鑰的金鑰材料時，它會在金鑰標籤中寫入 KMS 金鑰的 Amazon Resource Name (ARN)。金鑰清單命令會傳回 key-reference 和 label。

i 備註

下列程序使用 AWS CloudHSM 用戶端 SDK 5 命令列工具 [CloudHSM CLI](#)。CloudHSM CLI `key-handle` 會取代為 `key-reference`。

2025 年 1 月 1 日，AWS CloudHSM 將結束對用戶端 SDK 3 命令列工具、CloudHSM 管理公用程式 (CMU) 和金鑰管理公用程式 (KMU) 的支援。如需用戶端 SDK 3 命令列工具和用戶端 SDK 5 命令列工具之間差異的詳細資訊，請參閱 AWS CloudHSM 《使用者指南》中的 [從用戶端 SDK 3 CMU 和 KMU 遷移至用戶端 SDK 5 CloudHSM CLI](#)。

若要執行此程序，您需要暫時中斷連接 AWS CloudHSM 金鑰存放區，以便以 `CU kmsuser` 身分登入。

1. 如果 AWS CloudHSM 金鑰存放區尚未中斷連線，請以身分登入 `kmsuser`，如中所述 [如何中斷連線和登入](#)。

i Note

當自訂金鑰存放區中斷連接時，所有在自訂金鑰存放區中建立 KMS 金鑰的嘗試，或在密碼編譯操作中使用現有 KMS 金鑰的嘗試，均會失敗。此動作可防止使用者存放和存取敏感資料。

2. 使用 CloudHSM CLI 中的 [金鑰清單](#) 命令，尋找 AWS CloudHSM 叢集中目前使用者的所有金鑰。

根據預設，僅顯示目前登入使用者的 10 個金鑰，且只顯示 `key-reference` 和 `label` 作為輸出。如需更多選項，請參閱 AWS CloudHSM 《使用者指南》中的 [金鑰清單](#)。

```
aws-cloudhsm > key list
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000000000123",
        "attributes": {
          "label": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
      },
      {
```

```
    "key-reference": "0x00000000000000456",
    "attributes": {
      "label": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321"
    }
  },
  ...8 keys later...
],
"total_key_count": 56,
"returned_key_count": 10,
"next_token": "10"
}
```

3. 如中所述登出並重新連接 AWS CloudHSM 金鑰存放區[如何登出和重新連線](#)。

尋找 金鑰的 KMS AWS CloudHSM 金鑰

如果您知道 叢集中 kmsuser 擁有之金鑰的 AWS CloudHSM 金鑰參考或 ID，您可以使用該值來識別金鑰存放區中相關聯的 KMS 金鑰。

當在 AWS CloudHSM 叢集中 AWS KMS 建立 KMS 金鑰的金鑰材料時，它會在金鑰標籤中寫入 KMS 金鑰的 Amazon Resource Name (ARN)。除非您已變更標籤值，否則您可以使用 CloudHSM CLI 中的[金鑰清單](#)命令來識別與金鑰相關聯的 KMS AWS CloudHSM 金鑰。

備註

下列程序使用 AWS CloudHSM 用戶端 SDK 5 命令列工具 [CloudHSM CLI](#)。CloudHSM CLI `key-handle` 會取代為 `key-reference`。

2025 年 1 月 1 日，AWS CloudHSM 將結束對用戶端 SDK 3 命令列工具、CloudHSM 管理公用程式 (CMU) 和金鑰管理公用程式 (KMU) 的支援。如需用戶端 SDK 3 命令列工具和用戶端 SDK 5 命令列工具之間差異的詳細資訊，請參閱 AWS CloudHSM 《使用者指南》中的[從用戶端 SDK 3 CMU 和 KMU 遷移至用戶端 SDK 5 CloudHSM CLI](#)。

若要執行這些程序，您需要暫時中斷連接 AWS CloudHSM 金鑰存放區，以便以 CU kmsuser 身分登入。

Note

當自訂金鑰存放區中斷連接時，所有在自訂金鑰存放區中建立 KMS 金鑰的嘗試，或在密碼編譯操作中使用現有 KMS 金鑰的嘗試，均會失敗。此動作可防止使用者存放和存取敏感資料。

主題

- [識別與金鑰參考相關聯的 KMS 金鑰](#)
- [識別與備份金鑰 ID 相關聯的 KMS 金鑰](#)

識別與金鑰參考相關聯的 KMS 金鑰

下列程序示範如何在 CloudHSM CLI 中使用 [金鑰清單](#) 命令搭配 `key-reference` 屬性篩選條件，以尋找叢集中做為金鑰存放區中特定 KMS 金鑰之金鑰材料的 AWS CloudHSM 金鑰。

1. 如果 AWS CloudHSM 金鑰存放區尚未中斷連線，請以身分登入 `kmsuser`，如中所述 [如何中斷連線和登入](#)。
2. 使用 CloudHSM CLI 中的 [金鑰清單](#) 命令，依 `key-reference` 屬性篩選。指定引數 `verbose` 以包含相符金鑰的所有屬性和金鑰資訊。如果您未指定 `verbose` 引數，金鑰清單操作只會傳回相符金鑰的 `key-reference` 和 `label` 屬性。

執行此命令之前，請將範例取代 `key-reference` 為您的帳戶中有效的範例。

```
aws-cloudhsm > key list --filter attr.key-reference="0x0000000000120034" --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x0000000000120034",
        "key-info": {
          "key-owners": [
            {
              "username": "kmsuser",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "cluster-coverage": "full"
        }
      }
    ]
  }
}
```



```

    },
    "attributes": {
      "key-type": "aes",
      "label": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "id": "0xbacking-key-id",
      "check-value": "0x29bbd1",
      "class": "my_test_key",
      "encrypt": true,
      "decrypt": true,
      "token": true,
      "always-sensitive": true,
      "derive": false,
      "destroyable": true,
      "extractable": false,
      "local": true,
      "modifiable": true,
      "never-extractable": false,
      "private": true,
      "sensitive": true,
      "sign": false,
      "trusted": false,
      "unwrap": true,
      "verify": false,
      "wrap": true,
      "wrap-with-trusted": false,
      "key-length-bytes": 32
    }
  ],
  "total_key_count": 1,
  "returned_key_count": 1
}

```

3. 如中所述登出並重新連接 AWS CloudHSM 金鑰存放區[如何登出和重新連線](#)。

識別與備份金鑰 ID 相關聯的 KMS 金鑰

金鑰 AWS CloudHSM 存放區中具有 KMS 金鑰之密碼編譯操作的所有 CloudTrail 日誌項目都包含具有 customKeyStoreId 和 additionalEventData 的欄位 backingKeyId。欄位傳回的值與

CloudHSM 金鑰id屬性backingKeyId相關。您可以依 id 屬性篩選[金鑰清單](#)操作，以識別與特定 相關聯的 KMS 金鑰backingKeyId。

1. 如果 AWS CloudHSM 金鑰存放區尚未中斷連線，請以 身分登入kmsuser，如 中所述[如何中斷連線和登入](#)。
2. 使用 CloudHSM CLI 中的[金鑰清單](#)命令搭配屬性篩選條件，尋找叢集中做為金鑰存放區中特定 KMS 金鑰之金鑰材料的 AWS CloudHSM 金鑰。

下列範例示範如何依 id 屬性篩選。會將id值 AWS CloudHSM 辨識為十六進位值。若要依 id 屬性篩選金鑰清單操作，您必須先將 CloudTrail 日誌項目中識別backingKeyId的值轉換為可 AWS CloudHSM 辨識的格式。

- a. 使用下列 Linux 命令將 backingKeyId 轉換為十六進位表示法。

```
echo backingKeyId | tr -d '\n' | xxd -p
```

下列範例示範如何將backingKeyId位元組陣列轉換為十六進位表示法。

```
echo 5890723622dc15f699aa9ab2387a9f744b2b884c18b2186ee8ada4f556a2eb9d | tr -d '\n' | xxd -p
353839303732333632326463313566363939616139616232333837613966373434623262383834633138623
```

- b. 在 的十六進位表示法前面backingKeyId加上 0x。

```
0x353839303732333632326463313566363939616139616232333837613966373434623262383834633138623
```

- c. 使用轉換backingKeyId的值依 id 屬性篩選。指定引verbose數以包含相符金鑰的所有屬性和金鑰資訊。如果您未指定verbose引數，金鑰清單操作只會傳回相符金鑰的 key-reference 和 label 屬性。

```
aws-cloudhsm > key list --filter
attr.id="0x353839303732333632326463313566363939616139616232333837613966373434623262383834633138623"
--verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000000120034",
        "key-info": {
```

```

    "key-owners": [
      {
        "username": "kmsuser",
        "key-coverage": "full"
      }
    ],
    "shared-users": [],
    "cluster-coverage": "full"
  },
  "attributes": {
    "key-type": "aes",
    "label": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "id":
"0x35383930373233363232646331356636393961613961623233383761396637343462326238383463313
    "check-value": "0x29bbd1",
    "class": "my_test_key",
    "encrypt": true,
    "decrypt": true,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": false,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": false,
    "trusted": false,
    "unwrap": true,
    "verify": false,
    "wrap": true,
    "wrap-with-trusted": false,
    "key-length-bytes": 32
  }
}
],
"total_key_count": 1,
"returned_key_count": 1
}
}

```

3. 如中所述登出並重新連接 AWS CloudHSM 金鑰存放區[如何登出和重新連線](#)。

尋找 KMS AWS CloudHSM 金鑰的金鑰

您可以使用 金鑰存放區中 KMS 金鑰的 KMS AWS CloudHSM 金鑰 ID，來識別 AWS CloudHSM 叢集中做為其金鑰材料的金鑰。

當在 AWS CloudHSM 叢集中 AWS KMS 建立 KMS 金鑰的金鑰材料時，它會在金鑰標籤中寫入 KMS 金鑰的 Amazon Resource Name (ARN)。除非您已變更標籤值，否則您可以使用 CloudHSM CLI 中的[金鑰清單](#)命令來尋找 KMS 金鑰之金鑰材料的金鑰資源和 ID。

金鑰 AWS CloudHSM 存放區中具有 KMS 金鑰之密碼編譯操作的所有 CloudTrail 日誌項目都包含具有 customKeyId 和 additionalEventData 的欄位 backingKeyId。backingKeyId 欄位中傳回的值是 id AWS CloudHSM 金鑰屬性。您可以依 KMS 金鑰 ARN 篩選金鑰清單 AWS CloudHSM CLI 操作，以識別與特定 KMS 金鑰相關聯的 CloudHSM 金鑰 id 屬性。

若要執行此程序，您需要暫時中斷連接 AWS CloudHSM 金鑰存放區，以便以 CU kmsuser 身分登入。

備註

下列程序使用 AWS CloudHSM 用戶端 SDK 5 命令列工具 [CloudHSM CLI](#)。CloudHSM CLI key-handle 會取代為 key-reference。

2025 年 1 月 1 日，AWS CloudHSM 將結束對用戶端 SDK 3 命令列工具、CloudHSM 管理公用程式 (CMU) 和金鑰管理公用程式 (KMU) 的支援。如需用戶端 SDK 3 命令列工具和用戶端 SDK 5 命令列工具之間差異的詳細資訊，請參閱 AWS CloudHSM 《使用者指南》中的[從用戶端 SDK 3 CMU 和 KMU 遷移至用戶端 SDK 5 CloudHSM CLI](#)。

1. 如果 AWS CloudHSM 金鑰存放區尚未中斷連線，請以 身分登入 kmsuser，如中所述[如何中斷連線和登入](#)。

Note

當自訂金鑰存放區中斷連接時，所有在自訂金鑰存放區中建立 KMS 金鑰的嘗試，或在密碼編譯操作中使用現有 KMS 金鑰的嘗試，均會失敗。此動作可防止使用者存放和存取敏感資料。

2. 使用 CloudHSM CLI 中的[金鑰清單](#)命令，並依篩選label，以尋找 AWS CloudHSM 叢集中特定金鑰的 KMS 金鑰。指定引verbose數，以包含相符金鑰的所有屬性和金鑰資訊。如果您未指定verbose引數，金鑰清單操作只會傳回相符金鑰的金鑰參考和標籤屬性。

下列範例示範如何依存放 KMS 金鑰 ARN 的 label 屬性進行篩選。執行此命令之前，請將範例 KMS 金鑰 ARN 取代為您的帳戶中的有效金鑰 ARN。

```
aws-cloudhsm > key list --filter attr.label="arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab" --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000000120034",
        "key-info": {
          "key-owners": [
            {
              "username": "kmsuser",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "aes",
          "label": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
          "id": "0xbacking-key-id",
          "check-value": "0x29bbd1",
          "class": "my_test_key",
          "encrypt": true,
          "decrypt": true,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": false,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
```

```
        "sensitive": true,  
        "sign": false,  
        "trusted": false,  
        "unwrap": true,  
        "verify": false,  
        "wrap": true,  
        "wrap-with-trusted": false,  
        "key-length-bytes": 32  
    }  
}  
],  
"total_key_count": 1,  
"returned_key_count": 1  
}  
}
```

3. 如中所述登出並重新連接 AWS CloudHSM 金鑰存放區[如何登出和重新連線](#)。

啟用和停用金鑰

您可以停用和重新啟用客戶受管金鑰。當您建立 KMS 金鑰時，它預設為啟用。如果您停用 KMS 金鑰，則在您重新啟用前，其無法在任何[密碼編譯操作](#)中使用。

因其為暫時性並且容易撤消，因此停用 KMS 金鑰是刪除 KMS 金鑰的安全替代方法，刪除是一項具破壞性且不可逆轉的動作。如果您正在考慮刪除 KMS 金鑰，請先將其停用並設定 [CloudWatch 警示](#) 或類似機制，以確保您不再會需要用到該金鑰來解密已加密的資料。

當您停用 KMS 金鑰時，該 KMS 金鑰會立即變為無法使用 (視最終一致性而定)。不過，使用受 KMS 金鑰保護之[資料金鑰](#)所加密的資源不會受影響，直到再次使用 KMS 金鑰為止 (例如解密資料金鑰)。此問題會影響 AWS 服務，其中許多 會使用資料金鑰來保護您的 資源。如需詳細資訊，請參閱[無法使用的 KMS 金鑰如何影響資料金鑰](#)。

您無法啟用或停用 [AWS 受管金鑰](#) 或 [AWS 擁有的金鑰](#)。AWS 受管金鑰 永久啟用以供[使用的服務 AWS KMS](#)使用。AWS 擁有的金鑰 僅由擁有它們的服務管理。

Note

AWS KMS 當客戶受管金鑰停用時，不會輪換金鑰材料。如需詳細資訊，請參閱[金鑰輪換的運作方式](#)。

使用 AWS KMS 主控台

您可以使用 AWS KMS 主控台來啟用和停用[客戶受管金鑰](#)。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
4. 選擇您要啟用或停用的 KMS 金鑰的核取方塊。
5. 若要啟用 KMS 金鑰，請選擇 Key actions (金鑰動作)、Enable (啟用)。若要停用 KMS 金鑰，請選擇 Key actions (金鑰動作)、Disable (停用)。

使用 AWS KMS API

[EnableKey](#) 操作會啟用停用 AWS KMS key。以下範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。key-id 參數是必要參數。

此操作不會傳回輸出。若要查看金鑰狀態，請使用 [DescribeKey](#) 操作。

```
$ aws kms enable-key --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

[DisableKey](#) 操作會停用已啟用的 KMS 金鑰。key-id 參數是必要參數。

```
$ aws kms disable-key --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

此操作不會傳回輸出。若要查看金鑰狀態，請使用 [DescribeKey](#) 操作，並查看 Enabled 欄位。

```
$ aws kms describe-key --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
  "KeyMetadata": {
    "Origin": "AWS_KMS",
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "Description": "",
    "KeyManager": "CUSTOMER",
    "MultiRegion": false,
    "Enabled": false,
    "KeyState": "Disabled",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "CreationDate": 1502910355.475,
    "Arn": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "AWSAccountId": "111122223333",
    "KeySpec": "SYMMETRIC_DEFAULT",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ]
  }
}
```


輪換 AWS KMS keys

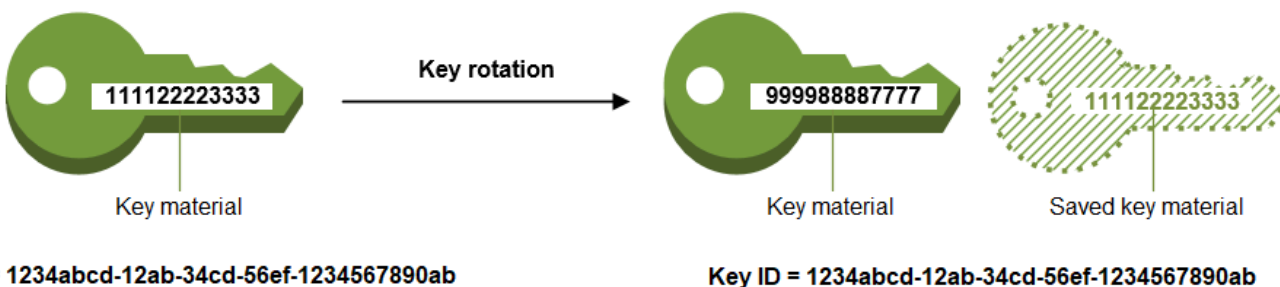
若要為您的[客戶受管金鑰](#)建立新的加密材料，您可以建立新的 KMS 金鑰，然後變更您的應用程式或別名來使用新的 KMS 金鑰。或者，您可以透過啟用自動金鑰輪換或執行隨需輪換，來輪換與現有 KMS 金鑰相關聯的金鑰材料。

根據預設，當您為 KMS 金鑰啟用自動金鑰輪換時，會每年為 KMS 金鑰 AWS KMS 產生新的密碼編譯資料。您也可以指定自訂[rotation-period](#)來定義啟用自動金鑰輪換後 AWS KMS 將輪換金鑰材料的天數，以及之後每次自動輪換之間的天數。如果您需要立即啟動金鑰材料輪換，無論是否啟用自動金鑰輪換，您都可以執行隨需輪換。隨需輪換不會變更現有的自動輪換排程。

您可以在 Amazon CloudWatch AWS CloudTrail和 AWS Key Management Service 主控台中[追蹤 KMS 金鑰的金鑰材料輪換](#)。您也可以使用 [GetKeyRotationStatus](#) 操作來驗證 KMS 金鑰是否已啟用自動輪換，並識別任何進行中的隨需輪換。您可以使用 [ListKeyRotations](#) 操作來檢視已完成輪換的詳細資訊。

金鑰輪換只會變更目前的金鑰材料，這是用於加密操作的密碼編譯秘密。當您使用輪換的 KMS 金鑰來解密加密文字時，AWS KMS 會使用用來加密的金鑰材料。您無法選取特定金鑰材料進行解密操作，AWS KMS 會自動選擇正確的金鑰材料。由於使用適當的金鑰材料 AWS KMS 透明地解密，因此您可以在應用程式中安全地使用輪換的 KMS 金鑰，AWS 服務無需變更程式碼。

無論其金鑰材料變更多少次，KMS 金鑰都是相同的邏輯資源。KMS 金鑰的屬性不會變更，如下圖所示。



您可以決定建立新的 KMS 金鑰，並使用它來代替原始 KMS 金鑰。這與輪換現有 KMS 金鑰的金鑰材料有相同效果，所以通常被視為[手動輪換金鑰](#)。當您想要輪換不符合自動或隨需金鑰輪換資格的 KMS 金鑰時，包括[非對稱 KMS 金鑰](#)、[HMAC KMS 金鑰](#)和[自訂金鑰存放](#)區中的 KMS 金鑰，手動輪換是不錯的選擇。

 Note

金鑰輪換不會影響 KMS 金鑰保護的資料。它不會輪換 KMS 金鑰產生的資料金鑰，或重新加密受 KMS 金鑰保護的任何資料。金鑰輪換不會降低資料金鑰遭到入侵的影響。

金鑰輪換和定價

AWS KMS 會針對為 KMS 金鑰維護之金鑰材料的第一次和第二次輪換收取每月費用。此價格提高上限為第二次輪換，任何後續輪換都不會計費。如需詳細資訊，請參閱 [AWS Key Management Service 定價](#)。

 Note

您可利用 [AWS Cost Explorer Service](#) 來檢視金鑰儲存費用明細。例如，您可指定 \$REGION-KMS-Keys 作為用量類型，然後按 API 操作將資料分組，以便篩選檢視並查看作為目前及輪換 KMS 金鑰計費的金鑰總費用。

您可能仍會看到歷史日期的舊版 Unknown API 操作執行個體。

金鑰輪換和配額

在計算金鑰資源配額時，每個 KMS 金鑰都被視為一個金鑰，而不考慮輪換的金鑰資料版本編號。

如需金鑰資料和輪換的詳細資訊，請參閱 [AWS Key Management Service 密碼編譯詳細資訊](#)。

主題

- [為什麼要輪換 KMS 金鑰？](#)
- [金鑰輪換的運作方式](#)
- [啟用自動金鑰輪換](#)
- [停用自動金鑰輪換](#)
- [執行隨需金鑰輪換](#)
- [列出輪換和金鑰資料](#)
- [手動輪換金鑰](#)
- [變更一組多區域金鑰中的主金鑰](#)

為什麼要輪換 KMS 金鑰？

密碼編譯最佳實務不鼓勵直接加密資料的金鑰廣泛重複使用，例如 AWS KMS 產生的[資料金鑰](#)。當 256 位元資料金鑰加密數百萬則訊息時，這些金鑰可能會用盡，並開始產生帶有詭祕模式的密文，狡猾分子可利用這些模式來發現金鑰中的位元。最好使用資料金鑰一次或幾次，以緩解此金鑰耗盡。

然而，KMS 金鑰最常用作包裝金鑰，也稱為金鑰加密的金鑰。包裝金鑰不會加密資料，其所加密的是用於加密資料的資料金鑰。因此，其使用頻率遠低於資料金鑰，且重複使用頻率之低，鮮少具金鑰額度用盡風險。

儘管額度用盡風險非常低，但由於業務或合約規則或政府法規，您仍可能需要輪換 KMS 金鑰。當您被迫輪換 KMS 金鑰時，建議您在支援的地方使用自動金鑰輪換、在不支援自動輪換時使用隨需輪換，以及在不支援自動或隨需金鑰輪換時使用手動金鑰輪換。

您可以考慮執行隨需輪換，以示範關鍵材料輪換功能或驗證自動化指令碼。建議針對意外輪換使用隨需輪換，並盡可能使用自動金鑰輪換搭配自訂[輪換期間](#)。

金鑰輪換的運作方式

AWS KMS 金鑰輪換旨在透明且易於使用。僅 AWS KMS 支援[客戶受管](#)金鑰的選用自動和隨需金鑰輪換。

自動金鑰輪換

AWS KMS 會在輪換期間定義的下一個輪換日期自動輪換 KMS 金鑰。您不必記住或排程更新。

只有具有 AWS KMS 產生 (AWS_KMS 原始伺服器) 金鑰材料的對稱加密 KMS 金鑰才支援自動金鑰輪換。

對於客戶受管 KMS 金鑰，自動輪換是選用的。AWS KMS 一律每年輪換 AWS 受管 KMS 金鑰的金鑰材料。AWS 擁有的 KMS 金鑰輪換由 AWS 服務擁有金鑰的管理。

隨需輪換

立即啟動與 KMS 金鑰相關聯的金鑰材料輪換，無論是否啟用自動金鑰輪換。

具有 AWS KMS 產生 (AWS_KMS 原始伺服器) 金鑰材料的對稱加密 KMS 金鑰和具有匯入金鑰材料 (EXTERNAL 原始伺服器) 的對稱加密 KMS 金鑰支援隨需金鑰輪換。

手動輪換

下列類型的 KMS 金鑰不支援自動或隨需金鑰輪換，但您可以[手動輪換這些 KMS 金鑰](#)。

- [非對稱 KMS 金鑰](#)
- [HMAC KMS 金鑰](#)
- [自訂金鑰存放區](#)中的 KMS 金鑰

管理金鑰資料

AWS KMS 即使金鑰輪換已停用，仍會保留具有AWS_KMS原始伺服器的 KMS 金鑰的所有金鑰材料。只有在您刪除 KMS 金鑰時，才會 AWS KMS 刪除金鑰材料。

您可以使用EXTERNAL原始伺服器管理對稱加密金鑰的金鑰材料。您可以使用 [DeleteImportedKeyMaterial](#) 操作刪除任何金鑰材料，或在匯入材料時設定過期時間。一旦其任何資料過期或刪除，KMS 金鑰就會變得無法使用。

使用金鑰資料

當您使用輪換的 KMS 金鑰來加密資料時，AWS KMS 會使用目前的金鑰材料。當您使用輪換的 KMS 金鑰來對加密文字進行解密時，AWS KMS 會使用原先用來加密的相同版本的金鑰資料。您無法為解密操作選取特定版本的金鑰材料，AWS KMS 而會自動選擇正確的版本。

輪換期間

輪換期間會定義您啟用自動金鑰輪換後 AWS KMS 將輪換金鑰材料的天數，以及之後每次自動金鑰輪換之間的天數。如果您在啟用自動金鑰輪換RotationPeriodInDays時未指定的值，預設值為 365 天。

您可以使用 [kms:RotationPeriodInDays](#) 條件索引鍵，進一步限制主體可以在 RotationPeriodInDays 參數中指定的值。

輪換日期

輪換日期反映 KMS 金鑰的目前金鑰材料因為自動（排程）輪換或隨需金鑰輪換而更新的日期。

輪換日期

AWS KMS 會在輪換期間定義的輪換日期自動輪換 KMS 金鑰。預設輪換期間為 365 天。

客戶自管金鑰

由於自動金鑰輪換在[客戶自管金鑰](#)是選擇性，且可隨時啟用及停用，因此輪換日期取決於最近啟用輪換的日期。如果您修改先前啟用自動金鑰輪換之金鑰的輪換期間，日期可能會變更。在金鑰的生命週期內，輪換日期可能會變更多次。

例如，如果您在 2022 年 1 月 1 日建立客戶受管金鑰，並在 2022 年 3 月 15 日啟用預設輪換期間為 365 天的自動金鑰輪換，則會在 2023 年 3 月 15 日、2024 年 3 月 15 日及之後每 365 天 AWS KMS 輪換金鑰材料。

下列範例假設已啟用自動金鑰輪換，預設輪換期間為 365 天。這些範例示範可能會影響金鑰輪換期間的特殊案例。

- 停用金鑰輪換 — 如果您在任何時候[停用自動金鑰輪換](#)，KMS 金鑰會繼續使用停用輪換時所用的金鑰資料版本。如果您再次啟用自動金鑰輪換，會根據新的啟用輪換日期輪換 AWS KMS 金鑰材料。
- 停用 KMS 金鑰 — 當 KMS 金鑰停用時，AWS KMS 不會輪換它。不過，金鑰輪換狀態不會變更，而且當 KMS 金鑰停用時您無法變更它。當 KMS 金鑰重新啟用時，如果金鑰材料超過其上次排定的輪換日期，會立即 AWS KMS 輪換它。如果金鑰材料尚未錯過其上次排定的輪換日期，會 AWS KMS 繼續原始金鑰輪換排程。
- KMS 金鑰待刪除 — 當 KMS 金鑰待刪除時，AWS KMS 不會輪換它。金鑰輪換狀態會設定為 `false`，且等待刪除時您無法變更它。如果刪除已取消，會恢復先前的金鑰輪換狀態。如果金鑰材料超過上次排定的輪換日期，會立即 AWS KMS 輪換。如果金鑰材料尚未錯過其上次排定的輪換日期，會 AWS KMS 繼續原始金鑰輪換排程。

AWS 受管金鑰

AWS KMS AWS 受管金鑰 每年自動輪換（約 365 天）。您無法啟用或停用 [AWS 受管金鑰](#) 的金鑰輪換。

的金鑰材料 AWS 受管金鑰 會在建立日期的一年後首次輪換，之後每年（大約從上次輪換起 365 天）輪換。

Note

在 2022 年 5 月，的輪換排程 AWS 受管金鑰 從每三年（約 1,095 天）AWS KMS 變更為每年（約 365 天）。

AWS 擁有的金鑰

您無法啟用或停用 AWS 擁有的金鑰的金鑰輪換。的[金鑰輪換](#)策略 AWS 擁有的金鑰 取決於建立和管理金鑰 AWS 的服務。如需詳細資訊，請參閱該服務使用者指南或開發人員指南中的靜態加密主題。

旋轉多區域金鑰

輪換行為會因金鑰材料是由 AWS KMS (AWS_KMS原始伺服器) 產生或匯入 (EXTERNAL原始伺服器) 而有所不同。

具有AWS_KMS原始伺服器的多區域金鑰

您可以啟用和停用自動輪換，並在具有AWS_KMS原始伺服器的對稱加密[多區域](#)金鑰中執行金鑰資料的隨需輪換。金鑰輪換是多區域金鑰的[共用屬性](#)。

您只能在主要金鑰上啟用和停用自動金鑰輪換。您只能在主索引鍵上啟動隨需輪換。

- 當 AWS KMS 同步多區域金鑰時，它會將金鑰輪換屬性設定從主金鑰複製到其所有相關複本金鑰。
- 當 AWS KMS 輪換金鑰材料時，它會為主要金鑰建立新的金鑰材料，然後將新的金鑰材料跨區域邊界複製到所有相關複本金鑰。金鑰材料永遠不會讓 AWS KMS 處於未加密狀態。這個步驟經過精心控制，以確保在密碼編譯操作中使用任何金鑰之前，金鑰材料已完全同步。
- AWS KMS 不會使用新的金鑰材料加密任何資料，直到該金鑰材料可在主金鑰及其每個複本金鑰中使用。
- 當您複寫已輪換的主要金鑰時，新的複本金鑰會具有目前的金鑰材料，以及其相關多區域金鑰的所有舊版金鑰材料。

這種模式可確保相關的多區域金鑰完全可互通操作。任何多區域金鑰都可以解密由相關多區域金鑰加密的任何加密文字，即使加密文字在建立金鑰之前已加密。

具有EXTERNAL原始伺服器的多區域金鑰

您可以在具有EXTERNAL原始伺服器的對稱加密[多區域](#)金鑰中，執行金鑰資料的隨需輪換。金鑰輪換是多區域金鑰的[共用屬性](#)。

只有在將新金鑰材料匯入主金鑰和每個複本金鑰之後，您才會在主金鑰上啟動隨需輪換。

- 當您將新的金鑰材料匯入主金鑰時，會將金鑰材料識別符和金鑰材料描述從主金鑰 AWS KMS 複製到所有相關複本金鑰。它不會複製金鑰材料。
- 您必須個別將相同的金鑰材料匯入每個複本金鑰。金鑰材料匯入所有相關多區域金鑰後，您可以在主金鑰上啟動隨需輪換。這可確保 AWS KMS 不會使用新的金鑰材料加密任何資料，直到該金鑰材料可在主金鑰及其每個複本金鑰中使用。
- 主索引鍵或任何複本索引鍵中的每個索引鍵資料，可以與相同索引鍵或任何其他相關多區域索引鍵中的其他索引鍵資料分開過期或刪除。

AWS 服務

您可以在用於 AWS 服務中伺服器端加密的[客戶受管金鑰](#)上啟用自動金鑰輪換。年度輪換是透明的，並且與 AWS 服務相容。

監控金鑰輪換

當 AWS KMS 輪換 [AWS 受管金鑰](#) 或 [客戶受管金鑰](#) 的金鑰材料時，它會將 KMS CMK Rotation 事件寫入 Amazon EventBridge，並將 [RotateKey 事件](#) 寫入您的 AWS CloudTrail 日誌。您可以使用這些記錄來驗證 KMS 金鑰是否已輪換。

您可以使用 AWS Key Management Service 主控台來檢視剩餘隨需輪換的數量，以及 KMS 金鑰所有已完成金鑰材料輪換的清單。

您可以使用 [ListKeyRotations](#) 操作來檢視已完成輪換的詳細資訊。

最終一致性

金鑰輪換會受到與其他 AWS KMS 管理操作相同的最終一致性影響。在新的金鑰材料可用於整個 AWS KMS 之前，可能會稍微延遲。然而，輪換金鑰材料不會造成密碼編譯操作中的任何中斷或延遲。在新的金鑰材料可用於整個 AWS KMS 之前，目前的金鑰材料會用於密碼編譯操作。當多區域金鑰的金鑰材料自動輪換時，AWS KMS 會使用目前的金鑰材料，直到新的金鑰材料可用於具有相關多區域金鑰的所有區域。

啟用自動金鑰輪換

根據預設，當您啟用 KMS 金鑰的自動金鑰輪換時，每年都會產生 KMS 金鑰 AWS KMS 的新密碼編譯資料。您也可以指定自訂 [rotation-period](#) 來定義啟用自動金鑰輪換後 AWS KMS 將輪換金鑰材料的天數，以及之後每次自動輪換之間的天數。

自動金鑰輪換有下列好處：

- KMS 金鑰的屬性，包括其 [金鑰 ID](#)、[金鑰 ARN](#)、區域、政策和許可，在金鑰輪換時不會變更。
- 您不需要變更參考 KMS 金鑰之金鑰 ID 或金鑰 ARN 的應用程式或別名。
- 輪換金鑰資料不會影響 KMS 金鑰在任何 AWS 服務之中的使用。
- 啟用金鑰輪換之後，AWS KMS 會在輪換期間定義的下一個輪換日期自動輪換 KMS 金鑰。您不必記住或排程更新。

您可以在 AWS KMS 主控台或使用 [EnableKeyRotation](#) 操作來啟用自動金鑰輪換。若要啟用自動金鑰輪換，您需要 kms:EnableKeyRotation 許可。如需 AWS KMS 許可的詳細資訊，請參閱 [許可參考](#)。

使用 AWS KMS 主控台

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。(您不能啟用或停用 AWS 受管金鑰的輪換。它們會每年自動輪換一次。)
4. 選擇 KMS 金鑰的別名或金鑰 ID。
5. 選擇 Key rotation (金鑰輪換) 標籤。

僅對稱加密 KMS 金鑰 (其含有 AWS KMS 產生的金鑰資料 (Origin (來源) 是 AWS_KMS)) (包括 [多區域](#) 對稱加密 KMS 金鑰) 的詳細資訊頁面會顯示 Key rotation (金鑰輪換) 索引標籤。

您無法自動輪換非對稱 KMS 金鑰、HMAC KMS 金鑰、含有 [匯入金鑰資料](#) 的 KMS 金鑰，或是位於 [自訂金鑰存放區](#) 中的 KMS 金鑰。但是，您可以 [手動進行輪換](#)。

6. 在自動金鑰輪換區段中，選擇編輯。
7. 針對金鑰輪換，選取啟用。

Note

如果 KMS 金鑰已停用或待刪除，AWS KMS 不會輪換金鑰材料，而且您無法更新自動金鑰輪換狀態或輪換期間。啟用 KMS 金鑰或取消刪除以更新自動金鑰輪換組態。如需詳細資訊，請參閱 [金鑰輪換的運作方式](#) 和 [金鑰的 AWS KMS 金鑰狀態](#)。

8. (選用) 輸入介於 90 到 2560 天的輪換期間。預設值為 365 天。如果您未指定自訂輪換期間，AWS KMS 會每年輪換金鑰材料。

您可以使用 [kms:RotationPeriodInDays](#) 條件索引鍵來限制主體在輪換期間可以指定的值。

9. 選擇儲存。

使用 AWS KMS API

您可以使用 [AWS Key Management Service \(AWS KMS\) API](#) 來啟用自動金鑰輪換，並檢視任何客戶受管金鑰的目前輪換狀態。以下範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

[EnableKeyRotation](#) 操作會啟用指定 KMS 金鑰的自動金鑰輪換。若要識別此操作中的 KMS 金鑰，請使用其[金鑰 ID](#) 或[金鑰 ARN](#)。依預設，客戶受管金鑰會停用金鑰輪換。

您可以使用 [kms:RotationPeriodInDays](#) 條件索引鍵來限制委託人可為EnableKeyRotation請求的RotationPeriodInDays 參數指定的值。

下列範例會在指定的對稱加密 KMS 金鑰上啟用輪換期間為 180 天的金鑰輪換，並使用[GetKeyRotationStatus](#) 操作來查看結果。

```
$ aws kms enable-key-rotation \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --rotation-period-in-days 180

$ aws kms get-key-rotation-status --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
  "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
  "KeyRotationEnabled": true,
  "RotationPeriodInDays": 180,
  "NextRotationDate": "2024-02-14T18:14:33.587000+00:00"
}
```

停用自動金鑰輪換

在客戶受管金鑰上啟用自動金鑰輪換後，您可以選擇隨時將其停用。

如果您停用自動金鑰輪換，KMS 金鑰會繼續使用輪換停用時所使用的金鑰材料版本。如果您再次啟用自動金鑰輪換，會根據新的啟用輪換日期 AWS KMS 輪換金鑰材料。

停用自動輪換不會影響您[執行隨需輪換](#)的能力，也不會取消任何進行中的隨需輪換。

您可以在 AWS KMS 主控台或使用 [DisableKeyRotation](#) 操作來停用自動金鑰輪換。若要停用自動金鑰輪換，您需要 kms:DisableKeyRotation 許可。如需 AWS KMS 許可的詳細資訊，請參閱 [許可參考](#)。

使用 AWS KMS 主控台

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。

3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。(您不能啟用或停用 AWS 受管金鑰的輪換。它們會每年自動輪換一次。)
4. 選擇 KMS 金鑰的別名或金鑰 ID。
5. 選擇 Key rotation (金鑰輪換) 標籤。

僅對稱加密 KMS 金鑰 (其含有 AWS KMS 產生的金鑰資料 (Origin (來源) 是 AWS_KMS)) (包括[多區域](#)對稱加密 KMS 金鑰) 的詳細資訊頁面會顯示 Key rotation (金鑰輪換) 索引標籤。

您無法自動輪換非對稱 KMS 金鑰、HMAC KMS 金鑰、含有[匯入金鑰資料](#)的 KMS 金鑰，或是位於[自訂金鑰存放區](#)中的 KMS 金鑰。但是，您可以[手動進行輪換](#)。

6. 在自動金鑰輪換區段中，選擇編輯。
7. 針對金鑰輪換，選取停用。

Note

如果 KMS 金鑰已停用或待刪除，AWS KMS 不會輪換金鑰材料，而且您無法更新自動金鑰輪換狀態或輪換期間。啟用 KMS 金鑰或取消刪除以更新自動金鑰輪換組態。如需詳細資訊，請參閱[金鑰輪換的運作方式](#)和[金鑰的 AWS KMS 金鑰狀態](#)。

8. 選擇儲存。

使用 AWS KMS API

您可以使用 [AWS Key Management Service \(AWS KMS\) API](#) 來停用自動金鑰輪換，並檢視任何客戶受管金鑰的目前輪換狀態。此範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

[DisableKeyRotation](#) 操作會停用自動金鑰輪換。若要識別此操作中的 KMS 金鑰，請使用其[金鑰 ID](#)或[金鑰 ARN](#)。依預設，客戶受管金鑰會停用金鑰輪換。

下列範例會在指定的對稱加密 KMS 金鑰上停用自動金鑰輪換，並使用 [GetKeyRotationStatus](#) 操作來查看結果。

```
$ aws kms disable-key-rotation --key-id 1234abcd-12ab-34cd-56ef-1234567890ab

$ aws kms get-key-rotation-status --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
  "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
```

```
"KeyRotationEnabled": false
}
```

執行隨需金鑰輪換

無論是否啟用自動金鑰輪換，您都可以在客戶受管 KMS 金鑰中執行金鑰資料的隨需輪換。停用自動輪換 ([DisableKeyRotation](#)) 不會影響您執行隨需輪換的能力，也不會取消任何進行中的隨需輪換。隨需輪換不會變更現有的自動輪換排程。例如，假設 KMS 金鑰已啟用自動金鑰輪換，輪換期間為 730 天。如果金鑰排定在 2024 年 4 月 14 日自動輪換，且您在 2024 年 4 月 10 日執行隨需輪換，則金鑰將在 2024 年 4 月 14 日及其後每 730 天自動輪換。

您可以執行隨需金鑰輪換，每個 KMS 金鑰最多 25 次。您可以使用 AWS KMS 主控台來檢視 KMS 金鑰可用的剩餘隨需輪換次數。

僅對稱加密 KMS 金鑰支援隨需金鑰輪換。您無法在 [自訂金鑰存放](#) 區中執行 [非對稱 KMS 金鑰](#)、[HMAC KMS 金鑰](#) 或 KMS 金鑰的隨需輪換。若要執行一組相關 [多區域金鑰](#) 的隨需輪換，請在主金鑰上叫用隨需輪換。

具有 `kms:RotateKeyOnDemand` 和 `kms:GetKeyRotationStatus` 許可的授權使用者可以使用 AWS KMS 主控台和 AWS KMS API 啟動隨需金鑰輪換，並檢視金鑰輪換狀態。使用 [ListKeyRotations](#) 來檢視 KMS 金鑰已完成的輪換。

主題

- [啟動隨需金鑰輪換（主控台）](#)
- [啟動隨需金鑰輪換 \(AWS KMS API\)](#)

啟動隨需金鑰輪換（主控台）

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。(您無法執行的隨需輪換 AWS 受管金鑰。它們每年會自動輪換。)
4. 選擇 KMS 金鑰的別名或金鑰 ID。
5. 選擇金鑰材料和輪換索引標籤。

金鑰材料和輪換索引標籤只會顯示在支援自動或隨需輪換的對稱加密 KMS 金鑰的詳細資訊頁面上。這包括 KMS 金鑰與產生的金鑰材料 AWS KMS (AWS_KMS 原始伺服器)，以及 KMS 金鑰與匯入的金鑰材料 (EXTERNAL 原始伺服器)

您無法在[自訂金鑰存放](#)區中執行非對稱 KMS 金鑰、HMAC KMS 金鑰或 KMS 金鑰的隨需輪換。但是，您可以[手動進行輪換](#)。

6. 選擇立即輪換。對於具有匯入金鑰材料的對稱加密金鑰，只有在您先前已[匯入新的金鑰材料](#)且處於待輪換狀態時，才能使用立即輪換選項。

Note

對於多區域金鑰，只能輪換主要區域金鑰。

7. 閱讀並考慮警告和有關金鑰剩餘隨需輪換次數的資訊。您也會看到輪換後將成為最新版本之金鑰材料的 ID、描述和過期時間等資訊。如果您決定不想繼續隨需輪換，請選擇取消。
8. 選擇輪換鍵以確認隨需輪換。

Note

隨需輪換會受到與其他 AWS KMS 管理操作相同的最終一致性影響。在新的金鑰材料可用於整個 AWS KMS 之前，可能會稍微延遲。主控台頂端的橫幅會在隨需輪換完成時通知您。

啟動隨需金鑰輪換 (AWS KMS API)

您可以使用 [AWS Key Management Service \(AWS KMS\) API](#) 來啟動隨需金鑰輪換，並檢視任何客戶受管金鑰的目前輪換狀態。此範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

[RotateKeyOnDemand](#) 操作會立即啟動指定 KMS 金鑰的隨需金鑰輪換。若要在這些操作中識別 KMS 金鑰，請使用其[金鑰 ID](#) 或[金鑰 ARN](#)。

下列範例會在指定的對稱加密 KMS 金鑰上啟動隨需金鑰輪換，並使用 [GetKeyRotationStatus](#) 操作來驗證隨需輪換正在進行中。kms:GetKeyRotationStatus 回應OnDemandRotationStartDate中的 會識別啟動進行中隨需輪換的日期和時間。在此範例中，KMS 金鑰也已啟用自動輪換，期間為 365 天。

```
$ aws kms rotate-key-on-demand --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
  "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
}

$ aws kms get-key-rotation-status --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
  "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
  "KeyRotationEnabled": true,
  "NextRotationDate": "2024-03-14T18:14:33.587000+00:00",
  "OnDemandRotationStartDate": "2024-02-24T18:44:48.587000+00:00"
  "RotationPeriodInDays": 365
}
```

如果 KMS 金鑰不支援自動輪換或未啟用自動輪換，`kms:GetKeyRotationStatus` 回應會有較少的欄位，如下列範例所示：

```
$ aws kms rotate-key-on-demand --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
  "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
}

$ aws kms get-key-rotation-status --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
  "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
  "KeyRotationEnabled": false,
  "OnDemandRotationStartDate": "2024-02-24T18:44:48.587000+00:00"
}
```

列出輪換和金鑰資料

支援自動或隨需金鑰輪換的 KMS 金鑰可以有多個與其相關聯的金鑰材料。這些金鑰具有初始金鑰材料，以及每個自動或隨需輪換的額外金鑰材料。

具有 `kms:ListKeyRotations` 許可的授權使用者可以使用 AWS KMS 主控台和 [ListKeyRotations API](#) 來列出與 KMS 金鑰相關聯的所有金鑰材料，包括已完成自動輪換和隨需輪換的金鑰材料。

主題

- [列出輪換和金鑰材料（主控台）](#)
- [列出輪換和金鑰材料 \(AWS KMS API\)](#)

列出輪換和金鑰材料 (主控台)

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
 2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
 3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
 4. 選擇 KMS 金鑰的別名或金鑰 ID。
 5. 選擇金鑰材料和輪換索引標籤。
- 金鑰材料和輪換索引標籤只會顯示在支援自動或隨需輪換的對稱加密 KMS 金鑰的詳細資訊頁面上。這包括 KMS 金鑰與 AWS KMS 產生的金鑰材料 (AWS_KMS 原始伺服器)，以及 KMS 金鑰與匯入的金鑰材料 (EXTERNAL 原始伺服器)。
 - 金鑰材料和輪換索引標籤中的金鑰材料資料表會列出與 KMS 金鑰相關聯的所有金鑰材料。對於每個金鑰材料，對應的項目會顯示由指派的唯一識別符、AWS KMS 輪換日期和金鑰材料狀態。輪換日期會識別金鑰材料在自動或隨需金鑰輪換之後何時變成最新版本。第一個或 Pending rotation 金鑰材料沒有相關聯的輪換日期。金鑰材料狀態決定如何使用 AWS KMS 金鑰材料。目前的金鑰材料用於加密和解密。非目前金鑰材料僅用於解密。金鑰材料狀態 Pending rotation 表示金鑰材料已分階段進行輪換。在隨需金鑰輪換使其成為目前的金鑰材料之前，此金鑰材料不會用於任何密碼編譯操作。金鑰材料顯示的其他資訊取決於 KMS 金鑰的類型。
 - 對於具有 AWS_KMS 原始伺服器的對稱加密 KMS 金鑰，每一列也會顯示輪換類型 - On-demand 或 Automatic。
 - 具有匯入金鑰材料 (EXTERNAL 原始伺服器) 的對稱加密 KMS 金鑰僅支援 On-demand 輪換，因此沒有輪換類型資料欄。反之，每一列都會顯示匯入狀態、使用者指定的描述、過期資訊和動作功能表。匯入狀態為匯入，表示金鑰材料可在內部使用 AWS KMS，或待匯入，表示金鑰材料無法在內部使用 AWS KMS。動作功能表可用來刪除匯入的金鑰材料或重新匯入金鑰材料。如果金鑰資料的匯入狀態為待匯入，則會停用刪除金鑰材料動作。Reimport 金鑰材料動作一律可用。您不需要等待金鑰材料過期或刪除，即可重新匯入。

列出輪換和金鑰材料 (AWS KMS API)

您可以使用 [AWS Key Management Service \(AWS KMS\) API](#) 啟動隨需金鑰輪換，並檢視任何客戶受管金鑰的目前輪換狀態。此範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

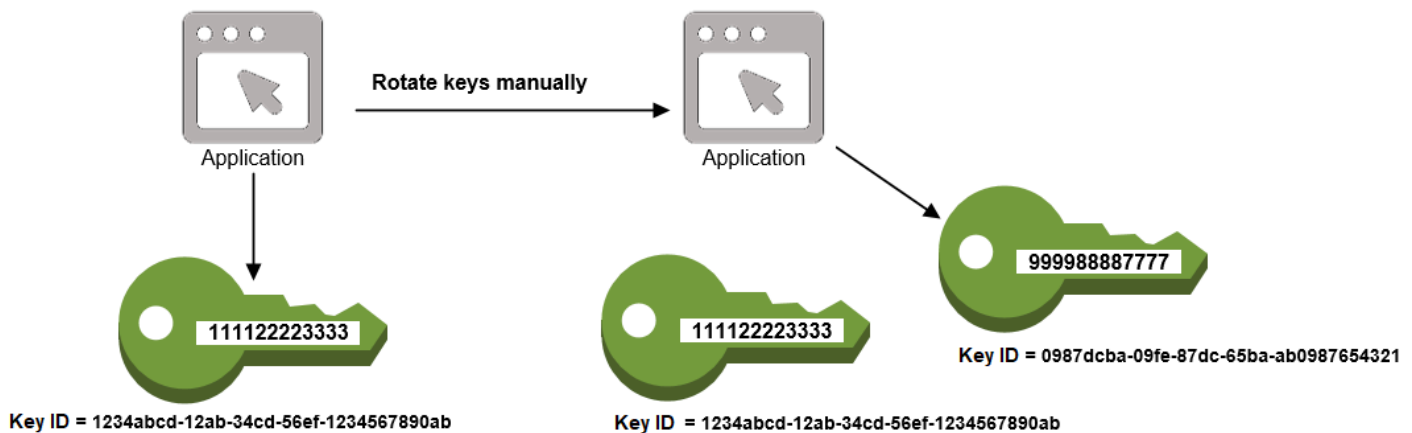
[ListKeyRotations](#) 操作會列出指定 KMS 金鑰的所有輪換和金鑰材料。若要在這些操作中識別 KMS 金鑰，請使用其[金鑰 ID](#) 或[金鑰 ARN](#)。

此操作支援選用 `IncludeKeyMaterial` 參數。此參數的預設值為 `ROTATIONS_ONLY`。如果您省略此參數，會 AWS KMS 傳回自動或隨需金鑰輪換所建立之金鑰資料的相關資訊。當您指定的值為 `ALL_KEY_MATERIAL`，AWS KMS 會將第一個金鑰材料和任何匯入的金鑰材料新增至回應，等待輪換。此參數只能與支援自動或隨需金鑰輪換的 KMS 金鑰搭配使用。

```
$ aws kms list-key-rotations --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --include-key-material ALL_KEY_MATERIAL
{
  "Rotations": [
    {
      "KeyId": 1234abcd-12ab-34cd-56ef-1234567890ab,
      "KeyMaterialId":
123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0,
      "KeyMaterialDescription": "KeyMaterialA",
      "ImportState": "PENDING_IMPORT",
      "KeyMaterialState": "NON_CURRENT"
    },
    {
      "KeyId": 1234abcd-12ab-34cd-56ef-1234567890ab,
      "KeyMaterialId":
96083e4fb6dbc41d77578a213a6b6669c044dd4c143e96755396d2bf11fd6068,
      "ImportState": "IMPORTED",
      "KeyMaterialState": "CURRENT",
      "ExpirationModel": "KEY_MATERIAL_DOES_NOT_EXPIRE",
      "RotationDate": "2025-05-01T15:50:51.045000-07:00",
      "RotationType": "ON_DEMAND"
    }
  ],
  "Truncated": false
}
```

手動輪換金鑰

您可能想要建立新的 KMS 金鑰，並使用它來取代目前的 KMS 金鑰，而不是使用自動或隨需金鑰輪換。新的 KMS 金鑰擁有與目前 KMS 金鑰不同的密碼編譯資料，使用新的 KMS 金鑰與變更現有 KMS 金鑰中的金鑰材料具有相同的效果。將一個 KMS 金鑰更換為另一個 KMS 金鑰的程序稱為手動金鑰輪換。



當您想要輪換不符合自動或隨需金鑰輪換資格的 KMS 金鑰時，例如非對稱 KMS 金鑰、HMAC KMS 金鑰和 [自訂金鑰存放](#) 區中的 KMS 金鑰，手動輪換是不錯的選擇。

Note

當您開始使用新的 KMS 金鑰時，請務必保持啟用原始 KMS 金鑰，以便 AWS KMS 可以解密原始 KMS 金鑰加密的資料。

在手動輪換 KMS 金鑰時，您還需要更新應用程式中對 KMS 金鑰 ID 或金鑰 ARN 的參考。[別名](#) 會將易記名稱關聯至 KMS 金鑰，讓此程序更容易進行。您可在應用程式中使用別名來參考 KMS 金鑰。而後，在您想要變更應用程式使用的 KMS 金鑰時，變更別名的目標 KMS 金鑰即可，而無需編輯應用程式代碼。如需詳細資訊，請參閱 [了解如何在應用程式中使用別名](#)。

Note

指向最新版本手動輪換 KMS 金鑰的別名是 [DescribeKey](#)、[GetPublicKey](#) 和密碼編譯操作的良好解決方案，例如

[DeriveSharedSecret](#)、[Encrypt](#)、[GenerateDataKey](#)、[GenerateDataKeyPair](#)、[GenerateMac](#)、[VerifyMac](#) and [Verify](#)。不允許在管理 KMS 金鑰的操作 (例如 [DisableKey](#) 或 [ScheduleKeyDeletion](#)) 中使用別名。

在手動輪換對稱加密 KMS 金鑰上呼叫 [Decrypt](#) 操作時，請省略 command 中的 KeyId 參數。AWS KMS 自動使用加密加密文字的 KMS 金鑰。

在使用非對稱 KMS 金鑰呼叫 [Decrypt](#) 或 [Verify](#) 時，或是使用 HMAC KMS 金鑰呼叫 [VerifyMac](#) 時，KeyId 參數是必要項目。當 KeyId 參數的值是不再指向執行密碼編譯操作的

KMS 金鑰的別名時 (例如手動輪換金鑰時)，這些請求將失敗。若要避免此錯誤，您必須追蹤每個操作並為其指定正確的 KMS 金鑰。

若要變更別名的目標 KMS 金鑰，請使用 AWS KMS API 中的 [UpdateAlias](#) 操作。例如，此命令會更新 alias/TestKey 別名以指向新的 KMS 金鑰。由於操作不會傳回任何輸出，範例使用 [ListAliases](#) 操作來顯示別名現在已與不同的 KMS 金鑰關聯，且 LastUpdatedDate 欄位已更新。ListAliases 命令使用中的 [query 參數](#) AWS CLI，只取得 alias/TestKey 別名。

```
$ aws kms list-aliases --query 'Aliases[?AliasName==`alias/TestKey`]'
{
  "Aliases": [
    {
      "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/TestKey",
      "AliasName": "alias/TestKey",
      "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
      "CreationDate": 1521097200.123,
      "LastUpdatedDate": 1521097200.123
    },
  ]
}

$ aws kms update-alias --alias-name alias/TestKey --target-key-id
0987dcba-09fe-87dc-65ba-ab0987654321

$ aws kms list-aliases --query 'Aliases[?AliasName==`alias/TestKey`]'
{
  "Aliases": [
    {
      "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/TestKey",
      "AliasName": "alias/TestKey",
      "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
      "CreationDate": 1521097200.123,
      "LastUpdatedDate": 1604958290.722
    },
  ]
}
```

變更一組多區域金鑰中的主金鑰

每組相關的多區域金鑰都必須有主要金鑰。但您可以變更主要金鑰。這個動作，稱為更新主要區域，會將目前的主要金鑰轉換為複本金鑰，並將其中一個相關的複本金鑰轉換為主要金鑰。如果您需要在維護複本金鑰的同時刪除目前的主要金鑰，或是要在與金鑰管理員相同的區域中尋找主要金鑰，則可以執行此動作。

您可以選取任何相關的複本金鑰做為新的主要金鑰。主要金鑰和複本金鑰都必須在操作開始時處於 Enabled [金鑰狀態](#)。

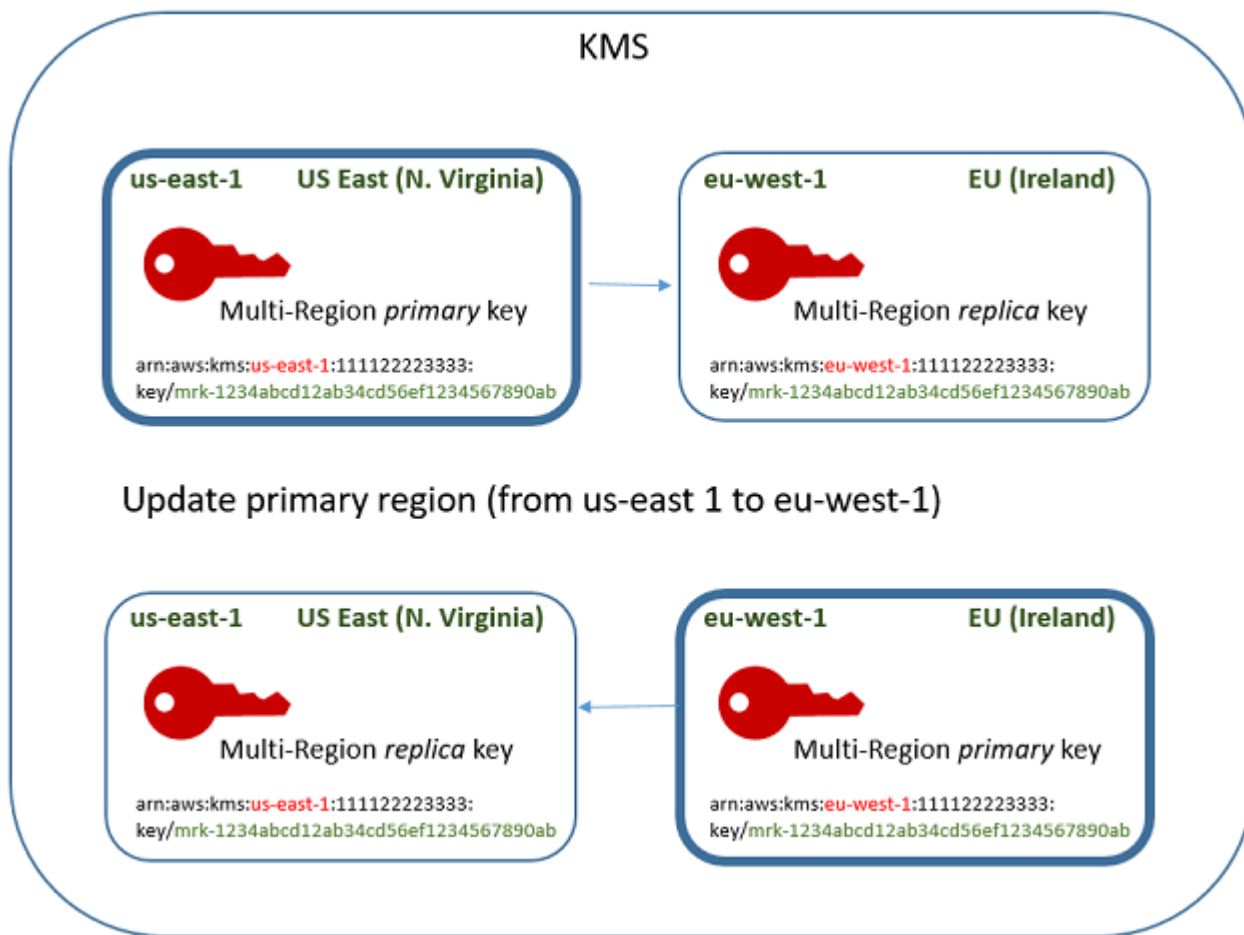
Updating 金鑰狀態

即使 UpdatePrimaryRegion 操作完成後，更新主要區域的程序仍可能再進行幾秒鐘。在此期間，舊的和新的主要金鑰具有 [更新](#) 的暫時性金鑰狀態。雖然金鑰狀態為 Updating，您可以在密碼編譯操作中使用金鑰，但無法複寫新的主要金鑰或執行特定的管理操作，例如啟用或停用這些金鑰。[DescribeKey](#) 等操作可能會同時將舊主要金鑰和新主要金鑰顯示為複本。更新完成時，Enabled 金鑰狀態會還原。

如需有關 Updating 金鑰狀態之影響的資訊，請參閱 [金鑰的 AWS KMS 金鑰狀態](#)。

運作方式

假設您在美國東部 (維吉尼亞北部) (us-east-1) 中有主要金鑰和在歐洲 (愛爾蘭) (eu-west-1) 中有複本金鑰。您可以使用更新功能將美國東部 (維吉尼亞北部) (us-east-1) 的主要金鑰變更為複本金鑰，並將歐洲 (愛爾蘭) (eu-west-1) 的複本金鑰變更為主要金鑰。



當更新程序完成時，歐洲 (愛爾蘭) (eu-west-1) 區域的多區域金鑰是多區域主要金鑰，而美國東部 (維吉尼亞北部) (us-east-1) 區域的多區域主要金鑰是其複本金鑰。如果有其他相關的複本金鑰，它們會成為新主要金鑰的複本。下次 AWS KMS 同步多區域金鑰的共用屬性時，它會從新的主金鑰取得**共用屬性**，並將其複製到其複本金鑰，包括先前的主金鑰。

更新操作並不會影響任何多區域金鑰的**金鑰 ARN**。它也不會影響共用屬性 (例如金鑰材料) 或獨立屬性 (例如金鑰政策)。不過，您可能想要**更新新主要金鑰的金鑰政策**。例如，您可能想要為可信任委託人將 [kms:ReplicateKey](#) 許可新增至新的主要金鑰，並將其從新的複本金鑰中移除。

更新主要區域

您可以將複本金鑰轉換為主金鑰，這會將先前的主金鑰變更為複本。若要更新主要區域，您需要兩個區域中的 [kms:UpdatePrimaryRegion](#) 許可。

您可以在 AWS KMS 主控台中或使用 [UpdatePrimaryRegion](#) 操作來更新主要區域。

使用 AWS KMS 主控台

您可以在 AWS KMS 主控台中更新主索引鍵。從目前主要金鑰的金鑰詳細資訊頁面開始。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
4. 選取 [多區域主要金鑰](#) 的金鑰 ID 或別名。如此會開啟主要金鑰的金鑰詳細資訊頁面。

若要識別多區域主要金鑰，請使用右上角的工具圖示新增 Regionality (區域性) 資料欄至資料表。

5. 選擇 Regionality (區域性) 索引標籤。
6. 在 Primary key (主要金鑰) 區段中，選擇 Change primary Region (變更主要區域)。
7. 選擇新主要金鑰的區域。您只可以從選單中選擇一個區域。

Change primary Regions (變更主要區域) 選單只包含具有相關多區域金鑰的區域。您可能沒有[更新選單上所有區域中主要區域的許可](#)。

8. 選擇 Change primary Region (變更主要區域)。

使用 AWS KMS API

若要變更一組相關多區域金鑰中的主要金鑰，請使用 [UpdatePrimaryRegion](#) 操作。

使用 KeyId 參數來識別目前的主要金鑰。使用 PrimaryRegion 參數來指出新主索引鍵 AWS 區域的。如果主要金鑰尚未在新的主要區域中有複本，則操作會失敗。

下列範例會將 us-west-2 區域中多區域金鑰的主要金鑰變更為其在 eu-west-1 區域的複本。KeyId 參數會識別 us-west-2 區域中的目前主要金鑰。PrimaryRegion 參數會指定新主索引鍵 AWS 區域的 eu-west-1。

```
$ aws kms update-primary-region \
    --key-id arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab \
    --primary-region eu-west-1
```

成功時，此操作不會返回任何輸出；只是 HTTP 狀態碼。若要查看效果，請針對任一多區域金鑰呼叫 [DescribeKey](#) 操作。您可能需要等到金鑰狀態返回到 Enabled。金鑰狀態為[正在更新](#)時，金鑰的值可能仍在變化中。

例如，下列 DescribeKey 呼叫會取得有關 eu-west-1 區域中多區域金鑰的詳細資訊。輸出顯示 eu-west-1 區域中的多區域金鑰現在已成為主要金鑰。us-west-2 區域中相關的多區域金鑰 (相同金鑰 ID) 現在是複本金鑰。

```
$ aws kms describe-key \
    --key-id arn:aws:kms:eu-west-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab \

{
  "KeyMetadata": {
    "AWSAccountId": "111122223333",
    "KeyId": "mrk-1234abcd12ab34cd56ef1234567890ab",
    "Arn": "arn:aws:kms:eu-west-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
    "CreationDate": 1609193147.831,
    "Enabled": true,
    "Description": "multi-region-key",
    "KeySpec": "SYMMETRIC_DEFAULT",
    "KeyState": "Enabled",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "Origin": "AWS_KMS",
    "KeyManager": "CUSTOMER",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ],
    "MultiRegion": true,
    "MultiRegionConfiguration": {
      "MultiRegionKeyType": "PRIMARY",
      "PrimaryKey": {
        "Arn": "arn:aws:kms:eu-west-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
        "Region": "eu-west-1"
      },
      "ReplicaKeys": [
        {
          "Arn": "arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
          "Region": "us-west-2"
        }
      ]
    }
  }
}
```

```
}
```

刪除 AWS KMS key

刪除 AWS KMS key 具有破壞性和潛在危險性。這樣會刪除金鑰資料，還有與 KMS 金鑰相關聯的所有中繼資料，而且無法復原。刪除 KMS 金鑰之後，您就再也無法解密以該 KMS 金鑰加密的資料，這表示該資料已無法復原。(唯一的例外是[多區域複本金鑰](#)，以及具有匯入金鑰資料的非對稱金鑰與 HMAC KMS 金鑰。)對於[用於加密的非對稱 KMS 金鑰](#)而言，此風險很重要，其中使用者可以在沒有警告或錯誤的情況下，繼續產生具有公有金鑰的加密文字，該公有金鑰在刪除私有金鑰之後無法解密 AWS KMS。

只有當您確定不再需要使用 KMS 金鑰時，才應刪除 KMS 金鑰。如果您不確定，請考慮[停用 KMS 金鑰](#)，而不是刪除。您可以重新啟用已停用的 KMS 金鑰並[取消排程刪除](#) KMS 金鑰，但您無法復原已刪除的 KMS 金鑰。

您只能排程刪除客戶受管金鑰。您無法刪除 AWS 受管金鑰 或 AWS 擁有的金鑰。

在刪除 KMS 金鑰之前，您可能想知道在該 KMS 金鑰下有多少加密文字。AWS KMS 不會儲存此資訊，也不會儲存任何加密文字。若要取得此資訊，您必須判斷 KMS 金鑰的過去使用情形。如需協助，請前往[判斷 KMS 金鑰的過去用量](#)。

AWS KMS 永遠不會刪除您的 KMS 金鑰，除非您明確排定刪除它們，且強制等待期間過期。

然而，您可能會因以下一或多個原因而選擇刪除 KMS 金鑰：

- 為了完成不再需要的 KMS 金鑰生命週期
- 為了避免管理開銷以及維護未使用 KMS 金鑰的相關[成本](#)
- 為了降低針對[KMS 金鑰資源配額](#)列入計算的 KMS 金鑰數量

Note

如果您[關閉 AWS 帳戶](#)，您的 KMS 金鑰將無法存取，而且您不再需要支付這些金鑰的費用。

AWS KMS 當您[排程刪除](#) KMS 金鑰以及[實際刪除 KMS 金鑰](#)時，會在 AWS CloudTrail 日誌中記錄項目。

關於等待期

由於刪除 KMS 金鑰具有破壞性和潛在危險性，AWS KMS 因此需要您設定 7 – 30 天的等待期間。預設等待期間為 30 天。

不過，實際等待期可能比您排定的等待期長最多 24 小時。若要取得刪除 KMS 金鑰的實際日期和時間，請使用 [DescribeKey](#) 操作。或者在 AWS KMS 主控台，KMS 金鑰的[詳細資訊頁面](#)，在 General configuration (一般組態) 區段中，請參閱 Scheduled deletion date (排定的刪除日期)。請務必注意時區。

在等待期間，KMS 金鑰狀態和金鑰狀態為 Pending deletion (待刪除)。

- 正在等待刪除的 KMS 金鑰不能用於任何[密碼編譯操作](#)。
- AWS KMS 不會[輪換待刪除 KMS 金鑰的金鑰材料](#)。

等待期間結束後，會 AWS KMS 刪除 KMS 金鑰、其別名和所有相關 AWS KMS 中繼資料。

排程刪除 KMS 金鑰可能不會立即影響 KMS 金鑰所加密的資料金鑰。如需詳細資訊，請參閱[無法使用的 KMS 金鑰如何影響資料金鑰](#)。

使用等待期間可確保您現在或未來不需要 KMS 金鑰。您可以[設定 Amazon CloudWatch 警示](#)來警告您有人或應用程式在等待期間嘗試使用 KMS 金鑰。若要復原 KMS 金鑰，您可以在等待期間結束前取消金鑰刪除。在等待期間結束後，您無法取消刪除金鑰，並 AWS KMS 刪除 KMS 金鑰。

特殊考量

排定要刪除的金鑰之前，請檢閱下列刪除特殊用途 KMS 金鑰的特殊考量。

刪除非對稱 KMS 金鑰

[獲得授權](#)的使用者可以刪除對稱或非對稱 KMS 金鑰。兩種類型金鑰排程刪除這些 KMS 金鑰的程序都是一樣的。不過，由於[非對稱 KMS 金鑰的公有金鑰可以在外部下載](#)和使用 AWS KMS，因此操作會帶來重大的額外風險，特別是用於加密的非對稱 KMS 金鑰（金鑰使用量為 ENCRYPT_DECRYPT）。

- 當您排程刪除 KMS 金鑰時，KMS 金鑰的金鑰狀態會變更為 Pending deletion (待刪除)，而 KMS 金鑰不能用於[密碼編譯操作](#)。不過，排程刪除不會影響外部的公有金鑰 AWS KMS。擁有公有金鑰的使用者可以繼續使用這些金鑰加密訊息。他們不會收到金鑰狀態變更的任何通知。除非取消刪除，否則無法解密使用公有金鑰建立的加密文字。

- 可偵測到嘗試使用待刪除 KMS 金鑰的警示、日誌和其他策略，偵測不到在 AWS KMS 外部對公有金鑰的使用。
- 刪除 KMS 金鑰時，所有涉及該 KMS 金鑰 AWS KMS 的動作都會失敗。不過，擁有公有金鑰的使用者可以繼續使用這些金鑰加密訊息。無法解密這些加密文字。

如果您必須刪除金鑰用途為 ENCRYPT_DECRYPT 的非對稱 KMS 金鑰，請使用 CloudTrail Log 項目判斷是否已下載並共用公有金鑰。如果是，請驗證公有金鑰未用在 AWS KMS 之外。然後，考慮[停用 KMS 金鑰](#)而不是刪除 KMS 金鑰。

對於含匯入金鑰資料的非對稱 KMS 金鑰來說，可減輕刪除非對稱 KMS 金鑰所造成的風險。如需詳細資訊，請參閱[Deleting KMS keys with imported key material](#)。

刪除多區域金鑰

若要刪除主要金鑰，您必須排程刪除其所有複本金鑰，然後等待刪除複本金鑰。刪除主要金鑰所需的等待期間在刪除其最後一個複本金鑰時開始。如果您必須從特定區域刪除主要金鑰而不刪除其複本金鑰，請透過[更新主要區域](#)將主要金鑰變更為複本金鑰。

您可以隨時刪除複本金鑰。它不取決於任何其他 KMS 金鑰的金鑰狀態。如果錯誤地刪除了複本密鑰，則可透過在同一區域中複寫相同的主要金鑰來重新建立。您建立的新複本金鑰將具有相同的[共用屬性](#)作為原始複本金鑰。

使用匯入的金鑰材料刪除 KMS 金鑰

刪除包含匯入金鑰資料的 KMS 金鑰資料為暫時性且可復原。若要還原金鑰，請重新匯入其金鑰資料。

相對地，刪除 KMS 金鑰則無法復原。如果您[排定金鑰刪除](#)，且所需的等待期間過期，會 AWS KMS 永久且無法復原地刪除 KMS 金鑰、其金鑰材料，以及與 KMS 金鑰相關聯的所有中繼資料。

然而，刪除包含匯入金鑰資料的 KMS 金鑰所產生的風險與後果取決於 KMS 金鑰的類型（「金鑰規格」）。

- 對稱加密金鑰 — 如您刪除對稱加密 KMS 金鑰，則該金鑰加密的所有剩餘密文均無法復原。您無法建立新的對稱加密 KMS 金鑰來針對已刪除的對稱加密 KMS 金鑰解密其密文，即使您擁有相同金鑰資料也無法做到。每個 KMS 金鑰的唯一中繼資料會以密碼編譯方式繫結至每個對稱密文。此安全性功能可保證僅經加密對稱密文的 KMS 金鑰才能將其解密，但這會讓您無法重新建立對等 KMS 金鑰。
- 非對稱和 HMAC 金鑰 — 如果您有原始金鑰材料，您可以建立新的 KMS 金鑰，其密碼編譯屬性與已刪除的非對稱或 HMAC KMS 金鑰相同。AWS KMS 會產生標準 RSA 加密文字和簽

章、ECC 簽章和 HMAC 標籤，其中不包含任何唯一的安全功能。此外，您也可在 AWS 外部利用 HMAC 金鑰或非對稱金鑰對的私有金鑰。

您利用相同非對稱或 HMAC 金鑰資料建立的新 KMS 金鑰將具有不同金鑰識別碼。您必須建立新的金鑰政策、重新建立任何別名，以及更新現有 IAM 政策與授權，以便參考新金鑰。

從金鑰存放區刪除 KMS AWS CloudHSM 金鑰

當您排定從金鑰存放區刪除 KMS AWS CloudHSM 金鑰時，其[金鑰狀態](#)會變更為待刪除。KMS 金鑰會在整個等待期保持在 Pending deletion (等待刪除) 狀態，即使 KMS 金鑰變為無法使用，因為您已[中斷連接自訂金鑰存放區](#)。這可讓您在等待期間隨時取消刪除 KMS 金鑰。

當等待期間到期時，會從 AWS KMS 中刪除 KMS 金鑰 AWS KMS。然後 AWS KMS，會盡最大努力從相關聯的 AWS CloudHSM 叢集中刪除金鑰材料。如果 AWS KMS 無法刪除金鑰資料 (例如，當金鑰存放區與 AWS KMS 中斷連接時)，您可能需要手動從叢集[刪除遺棄的金鑰資料](#)。

AWS KMS 不會從叢集備份中刪除金鑰材料。即使您從刪除 KMS 金鑰 AWS KMS 並從 AWS CloudHSM 叢集刪除其金鑰材料，從備份建立的叢集可能包含已刪除的金鑰材料。若要永久刪除金鑰材料，請使用 [DescribeKey](#) 操作來識別 KMS 金鑰的建立日期。然後，[刪除所有叢集備份](#)，其中可能包含金鑰資料。

當您排程從金鑰存放區刪除 KMS AWS CloudHSM 金鑰時，KMS 金鑰會立即變成無法使用 (取決於最終一致性)。不過，使用受 KMS 金鑰保護之[資料金鑰](#)所加密的資源不會受影響，除非再次使用 KMS 金鑰 (例如解密資料金鑰)。此問題會影響 AWS 服務，其中許多會使用資料金鑰來保護您的資源。如需詳細資訊，請參閱[無法使用的 KMS 金鑰如何影響資料金鑰](#)。

從外部金鑰存放區刪除 KMS 金鑰

從外部金鑰存放區刪除 KMS 金鑰不會影響作為其金鑰材料的[外部金鑰](#)。

排程從外部金鑰存放區刪除 KMS 金鑰時，其[金鑰狀態](#)會變更為 Pending deletion (等待刪除)。KMS 金鑰會在整個等待期保持在 Pending deletion (等待刪除) 狀態，即使 KMS 金鑰變為無法使用，因為您已[中斷連接外部金鑰存放區](#)。這可讓您在等待期間隨時取消刪除 KMS 金鑰。當等待期間過期時，會從 AWS KMS 中刪除 KMS 金鑰 AWS KMS。

當您排程從外部金鑰存放區刪除 KMS 金鑰時，KMS 金鑰會立即變為無法使用 (視最終一致性而定)。不過，使用受 KMS 金鑰保護之[資料金鑰](#)所加密的資源不會受影響，除非再次使用 KMS 金鑰 (例如解密資料金鑰)。此問題會影響 AWS 服務，其中許多服務會使用資料金鑰來保護您的資源。如需詳細資訊，請參閱[無法使用的 KMS 金鑰如何影響資料金鑰](#)。

控制金鑰刪除的存取權

如果您使用 IAM 政策來允許 AWS KMS 許可，則已允許具有 AWS 管理員存取權 ("Action": "*") 或 AWS KMS 完整存取權 ("Action": "kms:*") 的 IAM 身分來排程和取消金鑰刪除 KMS 金鑰。若要允許金鑰管理員排程和取消金鑰政策中的金鑰刪除，請使用 AWS KMS 主控台或 AWS KMS API。

通常，只有金鑰管理員才有權排程或取消金鑰刪除。不過，您可以將 `kms:ScheduleKeyDeletion` 和 `kms:CancelKeyDeletion` 許可新增至金鑰政策或 IAM 政策，從而將這些許可授予給其他 IAM 身分。您也可利用此 [kms:ScheduleKeyDeletionPendingWindowInDays](#) 條件金鑰，針對主體對 [ScheduleKeyDeletion](#) 請求指定的 `PendingWindowInDays` 參數值加以進一步限制。

允許金鑰管理員排程和取消金鑰刪除

使用 AWS KMS 主控台

將排程和取消金鑰刪除的許可授予給金鑰管理員。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
4. 針對您要變更許可的 KMS 金鑰，選擇其別名或金鑰 ID。
5. 選擇 Key policy (金鑰政策) 標籤。
6. 金鑰政策的 default view (預設檢視) 和 policy view (政策檢視) 的下一步有所不同。只有當您使用預設的主控台金鑰政策時，才能使用預設檢視。否則，只能使用政策檢視。

當預設檢視可用時，Switch to policy view (切換為政策檢視) 或 Switch to default view (切換為預設檢視) 按鈕顯示在 Key policy (金鑰政策) 索引標籤中。

- 在預設檢視中：
 - 在 Key deletion (金鑰刪除) 下方，選擇 Allow key administrators to delete this key (允許金鑰管理員刪除此金鑰)。
- 在政策檢視中：
 - a. 選擇編輯。
 - b. 在金鑰管理員的政策陳述式中，將 `kms:ScheduleKeyDeletion` 和 `kms:CancelKeyDeletion` 許可新增至 Action 元素。

```
{
  "Sid": "Allow access for Key Administrators",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:user/KMSKeyAdmin"},
  "Action": [
    "kms:Create*",
    "kms:Describe*",
    "kms:Enable*",
    "kms:List*",
    "kms:Put*",
    "kms:Update*",
    "kms:Revoke*",
    "kms:Disable*",
    "kms:Get*",
    "kms>Delete*",
    "kms:ScheduleKeyDeletion",
    "kms:CancelKeyDeletion"
  ],
  "Resource": "*"
}
```

c. 選擇儲存變更。

使用 AWS KMS API

您可以使用 AWS Command Line Interface 新增排程和取消刪除金鑰的許可。

若要新增排程和取消金鑰刪除的許可

1. 使用 [aws kms get-key-policy](#) 命令擷取現有的金鑰政策，然後將政策文件儲存至檔案。
2. 在您偏好的文字編輯器中開啟政策文件。在金鑰管理員的政策陳述式中，新增 `kms:ScheduleKeyDeletion` 和 `kms:CancelKeyDeletion` 許可。以下範例顯示具有這兩個許可的政策陳述式：

```
{
  "Sid": "Allow access for Key Administrators",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:user/KMSKeyAdmin"},
  "Action": [
    "kms:Create*",
    "kms:Describe*",
```

```
"kms:Enable*",
"kms:List*",
"kms:Put*",
"kms:Update*",
"kms:Revoke*",
"kms:Disable*",
"kms:Get*",
"kms:Delete*",
"kms:ScheduleKeyDeletion",
"kms:CancelKeyDeletion"
],
"Resource": "*"
}
```

3. 使用 [aws kms put-key-policy](#) 命令將金鑰政策套用到 KMS 金鑰。

排程金鑰刪除

下列程序說明如何 AWS KMS 使用 和 AWS KMS API，在 中排程金鑰刪除 AWS 管理主控台 和取消金鑰刪除 AWS KMS keys (KMS 金鑰)。

Warning

刪除 KMS 金鑰很具有破壞性，可能有潛在危險。只有當您確定不再需要使用 KMS 金鑰，將來也不需要 KMS 金鑰時，才應繼續進行。如果不確定，您應[停用 KMS 金鑰](#)，而不是刪除。

在可以刪除 KMS 金鑰之前，您必須擁有執行此操作的許可。如需有關將這些許可授予給金鑰管理員的資訊，請參閱 [控制金鑰刪除的存取權](#)。您也可利用 [kms:ScheduleKeyDeletionPendingWindowInDays](#) 條件金鑰來進一步限制等待期間，例如強制執行最短等待期。

AWS KMS 當您[排定刪除](#) KMS 金鑰和[實際刪除 KMS 金鑰](#)時，會在 AWS CloudTrail 日誌中記錄項目。

使用 AWS KMS 主控台

在 中 AWS 管理主控台，您可以一次排程和取消刪除多個 KMS 金鑰。

排程金鑰刪除

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。

您無法排程刪除 [AWS 受管金鑰](#) 或 [AWS 擁有的金鑰](#)。

4. 選擇您要刪除之 KMS 金鑰旁的核取方塊。
5. 選擇 Key actions (金鑰動作)、Schedule key deletion (排程金鑰刪除)。
6. 閱讀並考量於等待期間取消刪除的警告及資訊。如果決定取消刪除，請選擇頁面底部的 Cancel (取消)。
7. 對於 Waiting period (in days) (等候期間 (以天為單位))，輸入介於 7 和 30 之間的數字。
8. 檢閱您要刪除的 KMS 金鑰。
9. 選擇 Confirm you want to schedule this key for deletion in **<number of days>** days. (確認您想要排程在 <number of days> 天內刪除此金鑰。) 旁的核取方塊。
10. 選擇 Schedule deletion (排定刪除)。

KMS 金鑰狀態會變更為 Pending deletion (待刪除)。

使用 AWS KMS API

使用 [aws kms schedule-key-deletion](#) 命令排程 [客戶受管金鑰](#) 的金鑰刪除，如以下範例所示。

您無法排程刪除 AWS 受管金鑰 或 AWS 擁有的金鑰。

```
$ aws kms schedule-key-deletion --key-id 1234abcd-12ab-34cd-56ef-1234567890ab --
pending-window-in-days 10
```

成功使用時，會 AWS CLI 傳回輸出，如下列範例所示：

```
{
  "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "DeletionDate": 1598304792.0,
  "KeyState": "PendingDeletion",
```

```
"PendingWindowInDays": 10
}
```

取消刪除金鑰

排程刪除 KMS 金鑰之後，您可以在金鑰仍處於待定刪除狀態時取消金鑰刪除。您可以在 AWS KMS 主控台或使用 [CancelKeyDeletion](#) 操作取消金鑰刪除。取消 KMS 金鑰的待定刪除後，KMS 金鑰的金鑰狀態為 Disabled。如需啟用 KMS 金鑰的詳細資訊，請參閱 [啟用和停用金鑰](#)。

使用 AWS KMS 主控台

若要取消金鑰刪除

1. 在 <https://console.aws.amazon.com/kms> 開啟 AWS KMS 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
4. 選擇您要復原的 KMS 金鑰旁邊的核取方塊。
5. 選擇 Key actions (金鑰動作)、Cancel key deletion (取消金鑰刪除)。

KMS 金鑰狀態會從 Pending deletion (待刪除) 變更為 Disabled (已停用)。若要使用 KMS 金鑰，您必須先[將其啟用](#)。

使用 AWS KMS API

使用 [aws kms cancel-key-deletion](#) 命令從 取消金鑰刪除 AWS CLI，如下列範例所示。

```
$ aws kms cancel-key-deletion --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

成功使用時，會 AWS CLI 傳回輸出，如下列範例所示：

```
{
  "KeyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
}
```

KMS 金鑰狀態會從 Pending Deletion (待刪除) 變更為 Disabled (已停用)。若要使用 KMS 金鑰，您必須先[將其啟用](#)。

建立偵測 KMS 金鑰待刪除使用情況的警示

您可以結合 Amazon CloudWatch Logs AWS CloudTrail 和 Amazon Simple Notification Service (Amazon SNS) 的功能來建立 Amazon CloudWatch 警示，以便在您帳戶中有人嘗試使用待刪除的 KMS 金鑰時通知您。如果您收到此通知，您可能想要取消刪除 KMS 金鑰並重新考慮您的刪除決定。

下列程序會建立一個警示，在 "**Key ARN** is pending deletion" 錯誤訊息寫入 CloudTrail 日誌檔案時通知您。此錯誤訊息指示有人或應用程式嘗試在[密碼編譯操作](#)中使用 KMS 金鑰。由於通知是連結到錯誤訊息，因此當您使用在待刪除的 KMS 金鑰上獲允許的 API 操作時，例如 ListKeys、CancelKeyDeletion 和 PutKeyPolicy，不會觸發通知。若要查看傳回此錯誤訊息的 AWS KMS API 操作清單，請參閱 [金鑰的 AWS KMS 金鑰狀態](#)。

您收到的通知電子郵件不會列出 KMS 金鑰或密碼編譯操作。您可以在 [CloudTrail 日誌](#) 中找到該資訊。反之，電子郵件會報告警示狀態從正常變更為警示。如需有關 CloudWatch 警示和狀態變更的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的[使用 Amazon CloudWatch 警示](#)。

Warning

此 Amazon CloudWatch 警示偵測不到在 AWS KMS 外部對非對稱 KMS 金鑰之公有金鑰的使用。如需刪除用於公有金鑰密碼編譯之非對稱 KMS 金鑰特殊風險的詳細資訊，包括建立無法解密的加密文字，請參閱 [Deleting asymmetric KMS keys](#)。

在此程序中，您會建立 CloudWatch 日誌群組指標篩選條件，以尋找待刪除例外狀況的執行個體。然後，您可以根據日誌群組指標建立 CloudWatch 警示。如需有關日誌群組指標篩選條件的資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的[使用篩選條件從日誌事件建立指標](#)。

1. 建立可剖析 CloudTrail 日誌的 CloudWatch 指標篩選條件。

遵循[建立日誌群組的指標篩選條件](#)中的指示，使用以下所需值。對於其他欄位，請接受預設值，並按要求提供名稱。

欄位	Value
篩選條件模式	{ \$.eventSource = kms* && \$.errorMessage = "* is pending deletion." }
指標值	1

2. 根據您在步驟 1 中建立的指標篩選條件建立 CloudWatch 警示。

使用下列必要值，[遵循根據日誌群組指標篩選條件建立 CloudWatch 警示](#)中的指示。對於其他欄位，請接受預設值，並按要求提供名稱。

欄位	Value
指標篩選條件	您在步驟 1 中建立的指標篩選條件的名稱。
閾值類型	靜態
條件	每當 <i>metric-name</i> 大於/等於 1
警示的資料點	1 個 (共 1 個)
遺失資料處理	將遺失資料視為良好 (不違反閾值)

完成此程序後，新 CloudWatch 警示每次進入 ALARM 狀態時，您都會收到通知。如果您收到此警示的通知，可能表示仍需要排程刪除的 KMS 金鑰來加密或解密資料。在這種情況下，請[取消刪除 KMS 金鑰](#)並重新考慮您的刪除決定。

判斷 KMS 金鑰的過去用量

在刪除 KMS 金鑰之前，您可能想知道在該金鑰下有多少加密文字。AWS KMS 不會儲存此資訊，也不會儲存任何加密文字。了解 KMS 金鑰的過去使用情形，可能有助您決定將來是否需要它。本主題會建議幾項策略，協助您判斷 KMS 金鑰的過去使用情形。

Warning

這些用於確定過去和實際用量的策略僅適用於 AWS 使用者和 AWS KMS 操作。他們偵測不到在 AWS KMS 外部對非對稱 KMS 金鑰之公有金鑰的使用。如需刪除用於公有金鑰密碼編譯之非對稱 KMS 金鑰特殊風險的詳細資訊，包括建立無法解密的加密文字，請參閱 [Deleting asymmetric KMS keys](#)。

主題

- [檢查 KMS 金鑰許可，以判斷潛在用量的範圍](#)
- [檢查 AWS CloudTrail 日誌以判斷實際用量](#)

檢查 KMS 金鑰許可，以判斷潛在用量的範圍

判斷何人或何物目前可存取 KMS 金鑰，可能有助您判斷 KMS 金鑰的使用範圍以及是否仍然需要它。若要了解如何判斷何人或何物目前可以存取 KMS 金鑰，請前往 [判斷對的存取 AWS KMS keys](#)。

檢查 AWS CloudTrail 日誌以判斷實際用量

您可以使用 KMS 金鑰使用歷史記錄來協助您判斷是否擁有在特定 KMS 金鑰下加密的加密文字。

所有 AWS KMS API 活動都會記錄在 AWS CloudTrail 日誌檔案中。如果您已在 KMS 金鑰所在的區域中[建立 CloudTrail 追蹤](#)，您可以檢查 CloudTrail 日誌檔案，以檢視特定 KMS 金鑰的所有 AWS KMS API 活動歷史記錄。即使沒有追蹤，您依然可以在 [CloudTrail 事件歷史記錄](#)中檢視最新事件。如需如何使用 AWS KMS CloudTrail 的詳細資訊，請參閱 [使用記錄 AWS KMS API 呼叫 AWS CloudTrail](#)。

以下範例會顯示 KMS 金鑰用來保護存放在 Amazon Simple Storage Service (Amazon S3) 中的物件時，所產生的 CloudTrail 日誌項目。在此範例中，物件會使用[透過含有 KMS 金鑰的伺服器端加密 \(SSE-KMS\) 來保護資料](#)上傳至 Simple Storage Service (Amazon S3)。當您使用 SSE-KMS 將物件上傳到 Amazon S3 時，請指定用於保護物件的 KMS 金鑰。Amazon S3 使用 AWS KMS [GenerateDataKey](#)操作來請求物件的唯一資料金鑰，而此請求事件會以類似如下的項目記錄在 CloudTrail 中：

```
{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROACKCEVSQ6C2EXAMPLE:user",
    "arn": "arn:aws:sts::111122223333:assumed-role/Admins/example-user",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2015-09-10T23:12:48Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:role/Admins",
        "accountId": "111122223333",
        "userName": "Admins"
      }
    }
  },
}
```

```

    "invokedBy": "internal.amazonaws.com"
  },
  "eventTime": "2015-09-10T23:58:18Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GenerateDataKey",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "internal.amazonaws.com",
  "userAgent": "internal.amazonaws.com",
  "requestParameters": {
    "encryptionContext": {"aws:s3:arn": "arn:aws:s3:::example_bucket/example_object"},
    "keySpec": "AES_256",
    "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "responseElements": null,
  "requestID": "cea04450-5817-11e5-85aa-97ce46071236",
  "eventID": "80721262-21a5-49b9-8b63-28740e7ce9c9",
  "readOnly": true,
  "resources": [{
    "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "accountId": "111122223333"
  }],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}

```

當您稍後從 Amazon S3 下載此物件時，Amazon S3 會傳送 Decrypt 請求至 AWS KMS，以使用指定的 KMS 金鑰解密物件的資料金鑰。當您執行此操作，您的 CloudTrail 日誌檔案會包含類似以下內容的項目：

```

{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROACKCEVSQ6C2EXAMPLE:example-user",
    "arn": "arn:aws:sts::111122223333:assumed-role/Admins/example-user",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2015-09-10T23:12:48Z"
      }
    }
  }
}

```

```

    },
    "sessionIssuer": {
      "type": "Role",
      "principalId": "AROACKCEVSQ6C2EXAMPLE",
      "arn": "arn:aws:iam::111122223333:role/Admins",
      "accountId": "111122223333",
      "userName": "Admins"
    }
  },
  "invokedBy": "internal.amazonaws.com"
},
"eventTime": "2015-09-10T23:58:39Z",
"eventSource": "kms.amazonaws.com",
"eventName": "Decrypt",
"awsRegion": "us-west-2",
"sourceIPAddress": "internal.amazonaws.com",
"userAgent": "internal.amazonaws.com",
"requestParameters": {
  "encryptionContext": {"aws:s3:arn": "arn:aws:s3:::example_bucket/example_object"}},
"responseElements": null,
"requestID": "db750745-5817-11e5-93a6-5b87e27d91a0",
"eventID": "ae551b19-8a09-4cfc-a249-205ddba330e3",
"readOnly": true,
"resources": [{
  "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "accountId": "111122223333"
}],
"eventType": "AwsApiCall",
"recipientAccountId": "111122223333"
}

```

CloudTrail 會記錄所有 AWS KMS API 活動。透過評估這些日誌項目，您或許可以確定特定 KMS 金鑰的過去使用情形，這可能有助於您確定是否要刪除它。

若要查看 AWS KMS API 活動如何出現在 CloudTrail 日誌檔案中的更多範例，請前往 [使用 記錄 AWS KMS API 呼叫 AWS CloudTrail](#)。如需 CloudTrail 的詳細資訊，請參閱《[AWS CloudTrail 使用者指南](#)》。

刪除匯入的金鑰材料

您可以隨時刪除 KMS 金鑰中匯入的金鑰材料。此外，當具有過期日期的匯入金鑰材料過期時，會 AWS KMS 刪除金鑰材料。在任一情況下，刪除金鑰材料時，KMS 金鑰的[金鑰狀態](#)會變更為待匯入，且 KMS 金鑰無法用於任何密碼編譯操作。

對稱加密金鑰可以有多個與其相關聯的金鑰材料。對於這些金鑰，KMS 會為每個金鑰材料指派唯一的識別符。您可以使用 [ListKeyRotations](#) API 來檢視這些金鑰材料識別符和對應的金鑰材料狀態（請參閱 [RotationsListEntry](#)）。金鑰材料狀態為 PENDING_ROTATION 或 PENDING_MULTI_REGION_IMPORT_AND_ROTATION 表示金鑰材料未永久與 KMS 金鑰相關聯。刪除或過期任何永久關聯的金鑰材料會將金鑰狀態變更為待匯入。您可以使用 [DeleteImportedKeyMaterial](#) API 中的 key-material-id 參數指定特定金鑰材料的識別符。

多區域金鑰的考量事項

- 當您刪除處於 PENDING_ROTATION 或 PENDING_MULTI_REGION_IMPORT_AND_ROTATION 狀態之主要區域金鑰的金鑰材料時，您也會刪除複本區域金鑰的金鑰材料。
- 如果您刪除主要或複本區域金鑰中的金鑰材料，則只會影響該特定金鑰，而其他相關的多區域金鑰保持不變。具有其所有永久關聯金鑰材料的任何主要或複本區域金鑰都會繼續用於密碼編譯操作。

Warning

key-material-id 參數是選用的，AWS KMS 如果您未指定，會刪除目前的金鑰材料。

除停用 KMS 金鑰及撤回權限外，刪除金鑰資料也可作為快速但暫時停止使用 KMS 金鑰的策略。反之，利用匯入的金鑰資料來排程刪除 KMS 金鑰也可快速停止運用 KMS 金鑰。不過，如果在等待期間未取消刪除，KMS 金鑰、相關聯的金鑰材料和所有金鑰中繼資料都會永久刪除。如需詳細資訊，請參閱[Deleting KMS keys with imported key material](#)。

若要刪除金鑰材料，您可以使用 AWS KMS 主控台或 [DeleteImportedKeyMaterial](#) API 操作。當您[刪除匯入的金鑰材料](#)，以及[AWS KMS 刪除過期的金鑰材料](#)時，會在 AWS CloudTrail 日誌中 AWS KMS 記錄項目。

刪除金鑰材料如何影響 AWS 服務

當您刪除任何金鑰材料時，KMS 金鑰會立即變成無法使用（取決於最終一致性）。不過，使用受 KMS 金鑰保護之[資料金鑰](#)所加密的資源不會受影響，除非再次使用 KMS 金鑰（例如解密資料金

鑰)。此問題會影響 AWS 服務，其中許多會使用資料金鑰來保護您的資源。如需詳細資訊，請參閱[無法使用的 KMS 金鑰如何影響資料金鑰](#)。

使用 AWS KMS 主控台

您可以使用 AWS KMS 主控台來刪除金鑰材料。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
4. 執行以下任意一項：
 - 選取含有匯入金鑰資料的 KMS 金鑰的核取方塊。選擇 Key actions (金鑰動作)、Delete key material (刪除金鑰材料)。對於具有多個金鑰材料關聯的對稱加密金鑰，這會刪除目前的金鑰材料。
 - 對於具有匯入金鑰材料的對稱加密 KMS 金鑰，請選擇 KMS 金鑰的別名或金鑰 ID。選擇金鑰材料和輪換索引標籤。金鑰材料資料表會列出與金鑰相關聯的所有金鑰材料。在對應至您要刪除之金鑰材料的資料列中，從動作選單中選擇刪除金鑰材料。
5. 確認您要刪除金鑰材料，然後選擇 Delete key material (刪除金鑰材料)。KMS 金鑰的狀態 (對應其[金鑰狀態](#)) 會變更為 Pending import (待匯入)。如果刪除的金鑰材料處於 PENDING_ROTATION 狀態，KMS 金鑰的狀態不會變更。

使用 AWS KMS API

若要使用 [AWS KMS API](#) 來刪除金鑰材料，請傳送 [DeleteImportedKeyMaterial](#) 請求。以下範例顯示如何使用 [AWS CLI](#) 執行此作業。

將 `1234abcd-12ab-34cd-56ef-1234567890ab` 替換為您要刪除其金鑰材料之 KMS 金鑰的金鑰 ID。您可以使用 KMS 金鑰的金鑰 ID 或 ARN，但不能對此操作使用別名。下列命令會刪除目前的金鑰材料，這可能是與金鑰相關聯的唯一金鑰材料。

```
$ aws kms delete-imported-key-material --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

若要刪除特定金鑰材料，請使用 key-material-id 參數指定識別的金鑰材料。`123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0` 將取代為您要刪除之金鑰資料的識別符。

```
$ aws kms delete-imported-key-material --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
--key-material-id 123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0
```

產生資料金鑰

資料金鑰是對稱金鑰，可用於加密資料，包括大量資料和其他資料加密金鑰。與無法下載的對稱 KMS 金鑰不同，資料金鑰會傳回給您，以便在外部使用 AWS KMS。

當 AWS KMS 產生資料金鑰時，它會傳回純文字資料金鑰以供立即使用（選用），以及您可以安全地與資料一起存放的資料金鑰加密副本。當您準備好解密資料時，請先要求 AWS KMS 解密加密的資料金鑰。

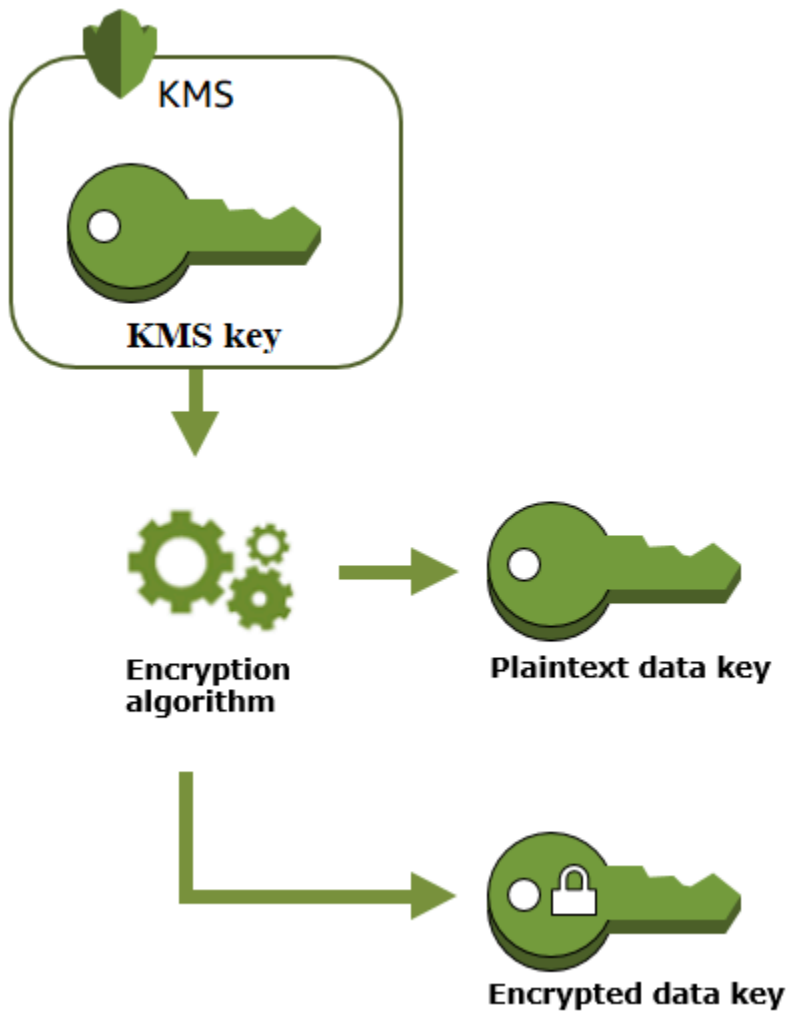
AWS KMS 會產生、加密和解密資料金鑰。不過，AWS KMS 不會存放、管理或追蹤您的資料金鑰，也不會使用資料金鑰執行密碼編譯操作。您必須使用和管理外部的資料金鑰 AWS KMS。如需安全使用資料金鑰的說明，請參閱 [AWS Encryption SDK](#)。

主題

- [建立資料金鑰](#)
- [使用資料金鑰進行密碼編譯操作的運作方式](#)
- [無法使用的 KMS 金鑰如何影響資料金鑰](#)

建立資料金鑰

若要建立資料金鑰，請呼叫 [GenerateDataKey](#) 操作。AWS KMS 產生資料金鑰。然後它會加密在您指定的[對稱加密 KMS 金鑰](#)下的資料金鑰複本。操作會傳回資料金鑰的純文字複本和以 KMS 金鑰加密的資料金鑰複本。下圖顯示此操作。



AWS KMS 也支援 [GenerateDataKeyWithoutPlaintext](#) 操作，只會傳回加密的資料金鑰。當您需要使用資料金鑰時，請 AWS KMS 將其[解密](#)。

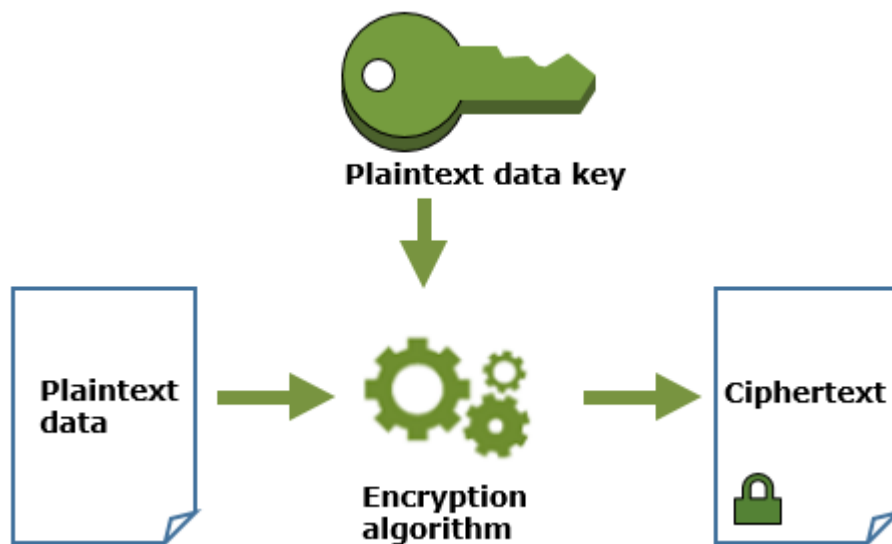
使用資料金鑰進行密碼編譯操作的運作方式

下列主題說明 [GenerateDataKey](#) 或 [GenerateDataKeyWithoutPlaintext](#) 操作所產生的資料金鑰如何運作。

使用資料金鑰來加密資料

AWS KMS 無法使用資料金鑰來加密資料。但是，您可以在 外部使用資料金鑰 AWS KMS，例如使用 OpenSSL 或密碼編譯程式庫，例如 [AWS Encryption SDK](#)。

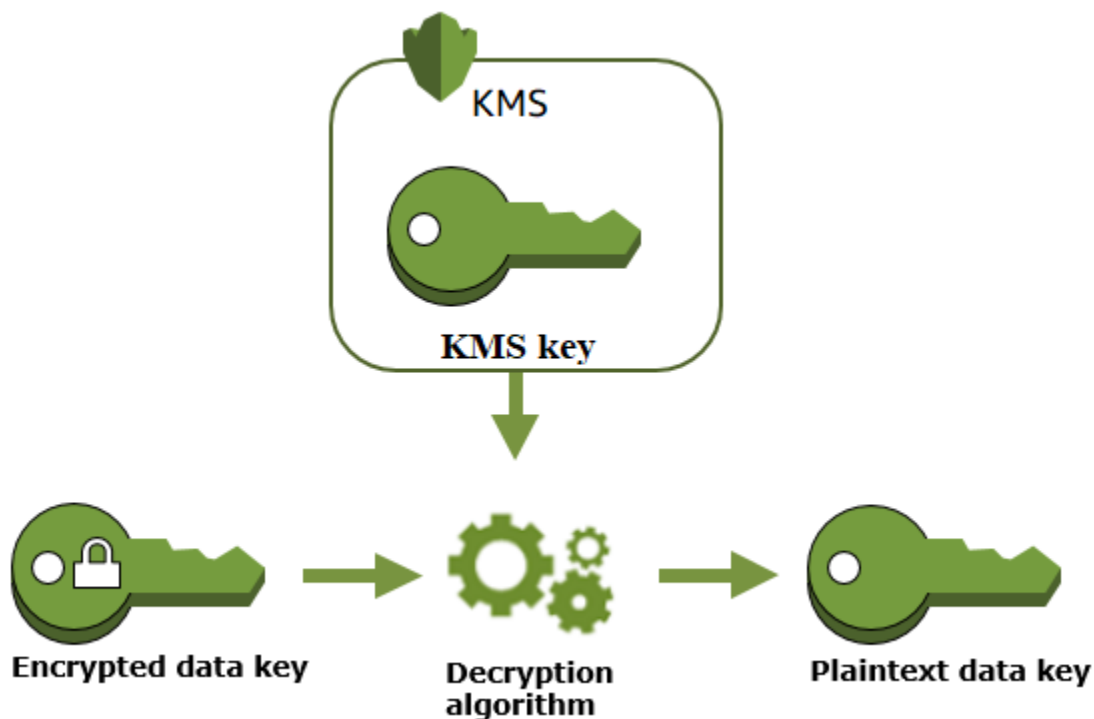
使用純文字資料金鑰加密資料後，請盡快從記憶體中移除該金鑰。您可將加密的資料金鑰與加密的資料安全地存放在一起，以使用來隨需解密資料。



使用資料金鑰來解密資料

若要解密您的資料，請將加密的資料金鑰傳遞至 [Decrypt](#) 操作。AWS KMS 會使用 KMS 金鑰來解密資料金鑰，然後傳回純文字資料金鑰。使用純文字資料金鑰來解密您的資料，然後盡快從記憶體中移除純文字資料金鑰。

下圖說明如何使用 Decrypt 操作來解密加密的資料金鑰。



無法使用的 KMS 金鑰如何影響資料金鑰

當 KMS 金鑰變得無法使用時，效果幾乎是即時的 (視最終一致性而定)。KMS 金鑰的[金鑰狀態](#)變更反映了其最新狀況，所有在[密碼編譯操作](#)中使用 KMS 金鑰的請求都將失敗。

但是，對由 KMS 金鑰加密的資料金鑰的影響以及對由資料金鑰加密的資料的影響會延遲，除非再次使用 KMS 金鑰，例如解密資料金鑰。

KMS 金鑰變得無法使用的原因有很多，包括您可能執行的下列動作。

- [停用 KMS 金鑰](#)
- [排程要刪除的 KMS 金鑰](#)
- 從具有匯入金鑰材料的 KMS 金鑰中[刪除金鑰材料](#)，或允許匯入的金鑰材料過期。如果具有EXTERNAL原始伺服器的 KMS 金鑰有多個相關聯的金鑰材料，則任何金鑰材料的刪除或過期都會導致金鑰無法使用。
- [中斷連接託管 KMS 金鑰的 AWS CloudHSM 金鑰存放區](#)，或從做為 KMS [金鑰金鑰材料的 AWS CloudHSM 叢集刪除金鑰](#)。
- [中斷連接託管 KMS 金鑰的外部金鑰存放區](#)，或任何其他干擾外部金鑰存放區代理的加密和解密請求的動作，包括從其外部金鑰管理器刪除外部金鑰。

對於許多使用資料金鑰來保護服務管理的資源 AWS 服務 的人來說，此效果特別重要。下列範例使用 Amazon Elastic Block Store (Amazon EBS) 和 Amazon Elastic Compute Cloud (Amazon EC2)。不同的 會以不同的方式 AWS 服務 使用資料金鑰。如需詳細資訊，請參閱 AWS 服務的「安全性」一章的「資料保護」一節。

例如，考量以下情境：

1. 您可以[建立已加密的 EBS 磁碟區](#)，並指定 KMS 金鑰來進行保護。Amazon EBS 會要求 AWS KMS 使用您的 KMS 金鑰來為磁碟區[產生加密資料金鑰](#)。Amazon EBS 會隨著磁碟區的中繼資料存放加密資料金鑰。
2. 當您將 EBS 磁碟區連接到 EC2 執行個體時，Amazon EC2 會使用您的 KMS 金鑰來解密 EBS 磁碟區的加密資料金鑰。Amazon EC2 使用 Nitro 硬體中的資料金鑰，負責將所有磁碟 I/O 加密至 EBS 磁碟區。只要 EBS 磁碟區連接 EC2 執行個體，資料金鑰就會存在 Nitro 硬體。
3. 您執行的動作使 KMS 金鑰無法使用。這不會立即影響 EC2 執行個體或 EBS 磁碟區。當磁碟區連接執行個體時，Amazon EC2 會採用資料金鑰 (而非 KMS 金鑰) 來加密所有磁碟 I/O。
4. 然而，當加密的 EBS 磁碟區從 EC2 執行個體中斷連接時，Amazon EBS 就會從 Nitro 硬體移除資料金鑰。下次再將加密的 EBS 磁碟區連接到 EC2 執行個體，連接會失敗，因為 Amazon EBS 無法

使用 KMS 金鑰來解密磁碟區的加密資料金鑰。若要再次使用 EBS 磁碟區，您必須讓 KMS 金鑰變得再次可用。

產生資料金鑰對

非對稱 KMS 金鑰代表資料金鑰對。資料金鑰對是非對稱的資料金鑰，由數學上相關的公有金鑰和私有金鑰組成。它們旨在用於用戶端加密和解密、在 外部簽署和驗證 AWS KMS，或在兩個對等之間建立共用秘密。

與 OpenSSL 等工具產生的資料金鑰對不同，會在 AWS KMS 您指定的對稱加密 KMS 金鑰下 AWS KMS 保護每個資料金鑰對中的私有金鑰。不過，AWS KMS 不會儲存、管理或追蹤您的資料金鑰對，或使用資料金鑰對執行密碼編譯操作。您必須在 AWS KMS 外部使用和管理資料金鑰對。

主題

- [建立資料金鑰對](#)
- [使用資料金鑰對進行密碼編譯操作的運作方式](#)

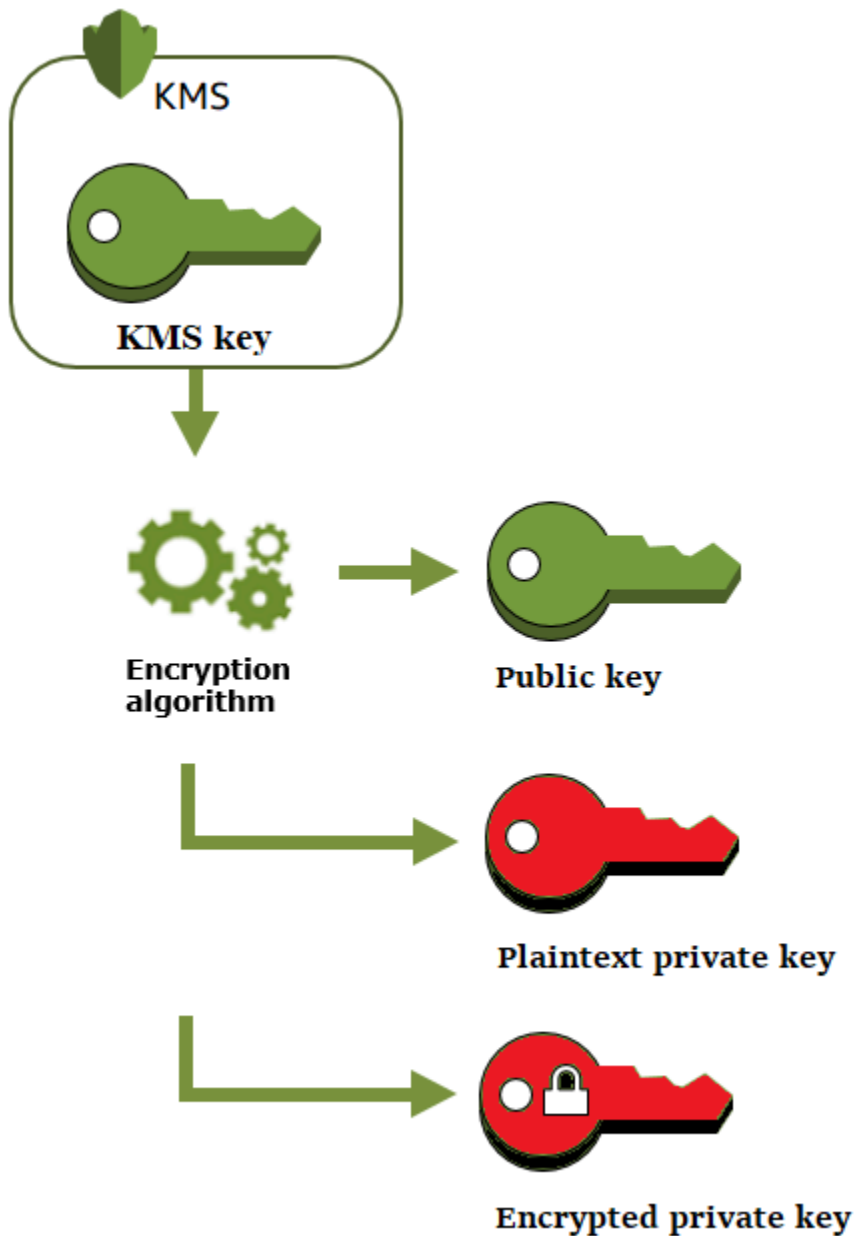
建立資料金鑰對

如要建立資料金鑰對，請呼叫 [GenerateDataKeyPair](#) 或 [GenerateDataKeyPairWithoutPlaintext](#) 操作。指定您希望用來加密私有金鑰的[對稱加密 KMS 金鑰](#)。

`GenerateDataKeyPair` 會傳回純文字的公有金鑰、純文字的私有金鑰和加密後的私有金鑰。請在立即需要純文字私有金鑰的時候使用此操作 (例如產生數位簽章)。

`GenerateDataKeyPairWithoutPlaintext` 會傳回純文字的公有金鑰和加密後的私有金鑰，但其中不包括純文字的私有金鑰。請在您不立即需要純文字私有金鑰的時候使用此操作 (例如使用公有金鑰進行加密時)。稍後，當您需要純文字的私有金鑰來解密資料時，您可以呼叫 [Decrypt](#) 操作。

下圖顯示 `GenerateDataKeyPair` 操作。`GenerateDataKeyPairWithoutPlaintext` 操作會省略純文字私有金鑰。



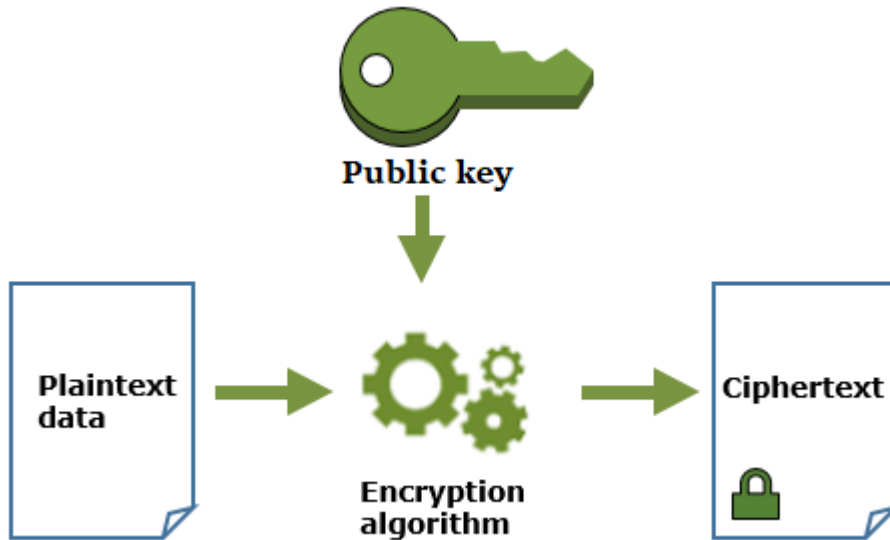
使用資料金鑰對進行密碼編譯操作的運作方式

下列主題說明您可以使用 [GenerateDataKeyPair](#) 或 [GenerateDataKeyPairWithoutPlaintext](#) 操作產生的資料金鑰對執行哪些密碼編譯操作，以及它們的運作方式。

使用資料金鑰對加密資料

當您使用資料金鑰對進行加密時，您可以使用金鑰隊的公有金鑰來加密資料，並使用相同金鑰對的私有金鑰來解密資料。通常，在多方需要加密僅持有私有金鑰的一方能夠解密的資料時，您會使用資料金鑰對。

持有公有金鑰的一方會使用該金鑰來加密資料，如下圖所示。

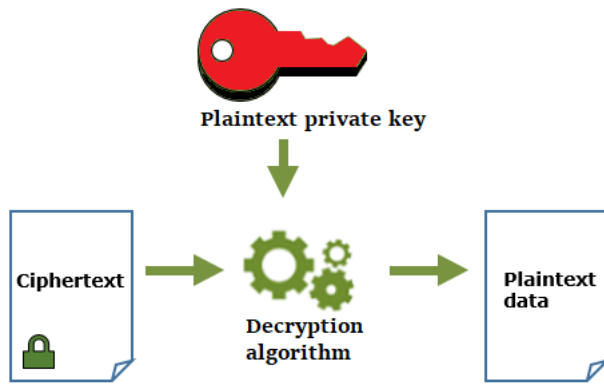


使用資料金鑰對解密資料

如要解密您的資料，請使用資料金鑰對中的私有金鑰。如要讓操作成功，公有和私有金鑰必須來自相同的資料金鑰對，且您必須使用相同的加密演算法。

如要解密經過加密的私有金鑰，請將該金鑰傳遞至 [Decrypt](#) 操作。使用純文字私有金鑰來解密資料。然後盡快從記憶體中移除純文字的私有金鑰。

下圖顯示如何使用資料金鑰對中的私有金鑰來解密加密文字。



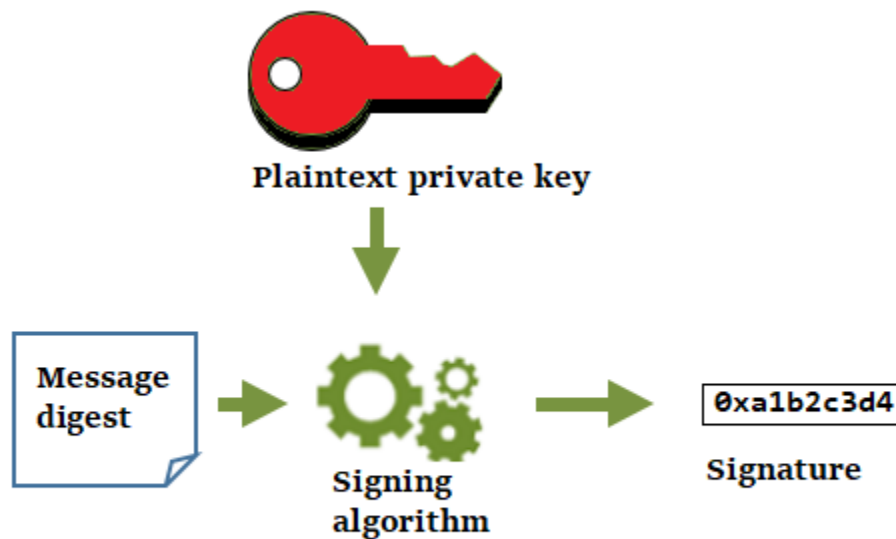
使用資料金鑰對簽署訊息

如要產生訊息的密碼編譯簽章，請使用資料金鑰對中的私有金鑰。任何具備公有金鑰的人員都能夠使用該金鑰來驗證訊息是使用您的私有金鑰進行簽署的，且在簽署之後並未發生任何變更。

如果您加密私有金鑰，請將加密的私有金鑰傳遞至 [Decrypt](#) 操作。AWS KMS 會使用 KMS 金鑰來解密資料金鑰，然後傳回純文字私有金鑰。使用純文字私有金鑰來產生簽章。然後盡快從記憶體中移除純文字的私有金鑰。

如要簽署訊息，請使用密碼編譯雜湊函數來建立訊息摘要，例如 OpenSSL 中的 [dgst](#) 命令。然後，將您的純文字私有金鑰傳遞至簽署演算法。結果便是代表訊息內容的簽章。(您可以在不先建立摘要的情況下簽署較短的訊息。訊息大小上限會因您使用的簽署工具而有所不同。)

下圖顯示如何使用資料金鑰對中的私有金鑰來簽署訊息。

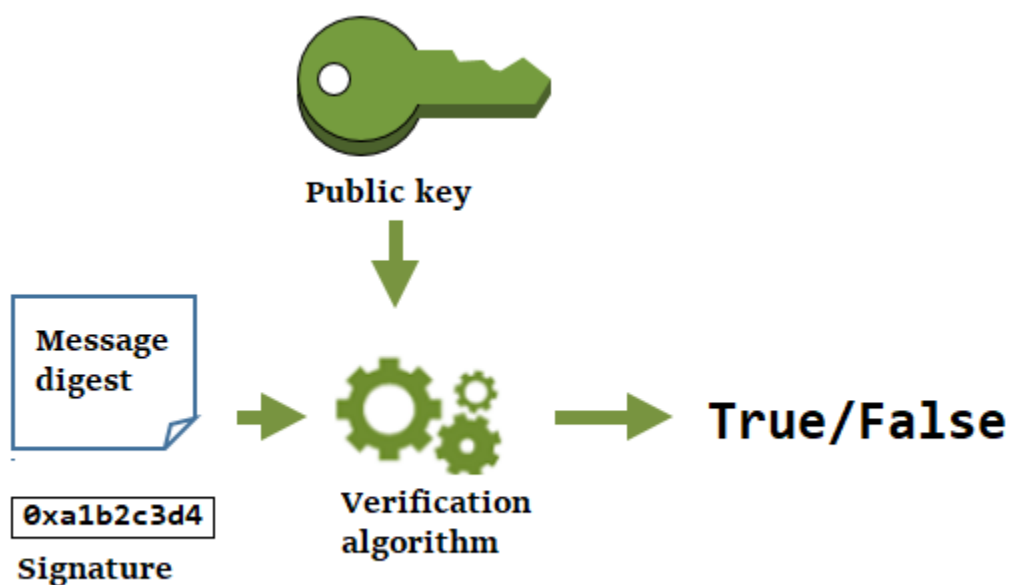


使用資料金鑰對驗證簽章

任何持有您資料金鑰對中公有金鑰的人員，都能使用該金鑰來驗證您使用私有金鑰產生的簽章。驗證會確認經過授權的使用者使用了指定的私有金鑰和簽署演算法簽署了訊息，且訊息自簽署以來並未產生變更。

如要成功，驗證簽章的一方必須產生相同的摘要類型、使用相同演算法，然後使用對應至用來簽署訊息私有金鑰的公有金鑰。

下圖顯示如何使用資料金鑰對中的公有金鑰來驗證訊息簽章。



衍生具有資料金鑰對的共用秘密

金鑰協議可讓兩個對等，每個對等都有橢圓曲線公有私有金鑰對，透過不安全的頻道建立共用秘密。若要[衍生共用秘密](#)，兩個對等端必須透過不安全的通訊管道（例如網際網路）交換其公有金鑰。然後，各方會使用其私有金鑰及其對等的公有金鑰，使用金鑰協議演算法來計算相同的共用秘密。您可以使用共用秘密值衍生對稱金鑰，以加密和解密兩個對等之間傳送的資料，或產生和驗證 HMACs。

Note

AWS KMS 強烈建議您在使用公有金鑰衍生共用秘密之前，先驗證您收到的公有金鑰是否來自預期的一方。

使用公有金鑰執行離線操作

在非對稱 KMS 金鑰中，私有金鑰是在 中建立 AWS KMS ，絕不會保持 AWS KMS 未加密狀態。若要使用私有金鑰，您必須呼叫 AWS KMS。您可以透過呼叫 AWS KMS API 操作 AWS KMS ，在 中使用公有金鑰。或者，您可以[下載公有金鑰](#)並共用以供 外部使用 AWS KMS。

您可以共用公有金鑰，讓其他人加密 以外的資料 AWS KMS ，您只能使用私有金鑰解密。或者，您可以允許其他人員在 AWS KMS 的外部驗證您使用私有金鑰產生的數位簽章。或者，與對等共用公有金鑰以衍生共用秘密。

當您在其中的非對稱 KMS 金鑰中使用公有金鑰時 AWS KMS ，您會受益於每個 AWS KMS 操作一部分的身分驗證、授權和記錄。您也可以降低加密無法解密資料的風險。這些功能在 之外無效 AWS KMS。如需詳細資訊，請參閱[下載公開金鑰的特殊考量](#)。

Tip

正在尋找資料金鑰或 SSH 金鑰？本主題介紹如何在不可匯出私有金鑰的 AWS Key Management Service 中管理非對稱金鑰。如需有關可匯出資料金鑰對 (其中私有金鑰受對稱加密 KMS 金鑰的保護) 的資訊，請參閱 [GenerateDataKeyPair](#)。如需下載與 Amazon EC2 執行個體相關聯的公有金鑰的說明，請參閱《[Amazon EC2 使用者指南](#)》和 [Amazon EC2 使用者指南](#)》中的擷取公有金鑰。

主題

- [下載公開金鑰的特殊考量](#)
- [下載公有金鑰](#)
- [離線操作範例](#)

下載公開金鑰的特殊考量

為了保護您的 KMS 金鑰，AWS KMS 提供存取控制、已驗證加密和每個操作的詳細日誌。AWS KMS 也可讓您暫時或永久避免使用 KMS 金鑰。最後，AWS KMS 操作旨在將加密無法解密資料的風險降至最低。當您在 外部使用下載的公有金鑰時，無法使用這些功能 AWS KMS。

Authorization

控制在 內存取 KMS 金鑰的[金鑰政策和 IAM 政策](#) AWS KMS 不會影響在 外部執行的操作 AWS。任何可以取得公有金鑰的使用者都可以在 外部使用它，AWS KMS 即使他們沒有使用 KMS 金鑰加密資料或驗證簽章的許可。

金鑰使用方式限制

金鑰使用限制在 之外無效 AWS KMS。如果您使用的 KMS 金鑰呼叫 [Encrypt](#) KeyUsage 操作SIGN_VERIFY，AWS KMS 操作會失敗。但是，如果您 AWS KMS 使用 KeyUsageSIGN_VERIFY或的 KMS 金鑰中的公有金鑰來加密 外部的資料KEY_AGREEMENT，則無法解密資料。

演算法限制

對 AWS KMS 支援的加密和簽署演算法的限制在 之外無效 AWS KMS。如果您使用外部 KMS 金鑰的公有金鑰來加密資料 AWS KMS，並使用 AWS KMS 不支援的加密演算法，則無法解密資料。

停用和刪除 KMS 金鑰

您可以採取防止在 密碼編譯操作中使用 KMS 金鑰的動作 AWS KMS，不會阻止任何人在 外部使用公有金鑰 AWS KMS。例如，停用 KMS 金鑰、排程刪除 KMS 金鑰、刪除 KMS 金鑰，或是從 KMS 金鑰刪除金鑰材料等，對 AWS KMS外部的公有金鑰沒有任何影響。如果您刪除非對稱 KMS 金鑰，或刪除或遺失其金鑰材料，則您在 外部使用公有金鑰加密的資料將無法 AWS KMS 復原。

日誌

AWS CloudTrail 記錄每個 AWS KMS 操作的日誌，包括請求、回應、日期、時間和授權使用者，不會記錄 外部公有金鑰的使用 AWS KMS。

使用 SM2 金鑰對進行離線驗證 (僅限中國區域)

若要 AWS KMS 使用 SM2 公有金鑰驗證 外部的簽章，您必須指定辨別 ID。根據預設，AWS KMS 會使用 1234567812345678做為辨別 ID。如需更多資訊，請參閱[使用 SM2 金鑰對進行離線驗證 \(僅限中國區域\)](#)。

下載公有金鑰

您可以從 AWS KMS 主控台的非對稱 KMS 金鑰對或使用 [GetPublicKey](#) 操作下載公有金鑰。若要下載公有金鑰，您必須擁有非對稱 KMS 金鑰的kms:GetPublicKey許可。

AWS KMS 傳回的公有金鑰是 DER 編碼的 X.509 公有金鑰，也稱為 SubjectPublicKeyInfo (SPKI)，如 [RFC 5280](#) 所定義。當您使用 HTTP API 或 時 AWS CLI，值為 Base64-encoded。否則，它不是 Base64-encoded。

若要從非對稱 KMS 金鑰對下載公有金鑰，您需要 `kms:GetPublicKey` 許可。如需 AWS KMS 許可的詳細資訊，請參閱 [許可參考](#)。

使用 AWS KMS 主控台

您可以使用 從 中的非對稱 KMS 金鑰 AWS 管理主控台 檢視、複製和下載公有金鑰 AWS 帳戶。若要從不同的非對稱 KMS 金鑰下載公有金鑰 AWS 帳戶，請使用 AWS KMS API。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
4. 選擇非對稱 KMS 金鑰的別名或金鑰 ID。
5. 選擇 Cryptographic configuration (密碼編譯組態) 索引標籤。記錄 Key spec (金鑰規格)、Key usage (金鑰使用情形) 和 Encryption algorithms (加密演算法) 或 Signing Algorithms (簽署演算法) 欄位的值。您需要使用這些值，才能在 外部使用公有金鑰 AWS KMS。請務必在您共享公有金鑰時共享此資訊。
6. 選擇 Public key (公有金鑰) 標籤。
7. 如要將公有金鑰複製到您的剪貼簿，請使用 Copy (複製)。如要將公有金鑰下載至檔案，請選擇 Download (下載)。

使用 AWS KMS API

[GetPublicKey](#) 操作會傳回非對稱 KMS 金鑰中的公有金鑰。它也會傳回您在 外部正確使用公有金鑰所需的重要資訊 AWS KMS，包括金鑰用量和加密演算法。請務必儲存這些值，並在您共享公有金鑰時也共享這些資訊。

本節中的範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

若要指定 KMS 金鑰，請使用它的 [金鑰 ID](#)、[金鑰 ARN](#)、[別名名稱](#) 或 [別名 ARN](#)。使用別名時，請加上 alias/ 字首。若要在不同的 中指定 KMS 金鑰 AWS 帳戶，您必須使用其金鑰 ARN 或別名 ARN。

執行此命令前，請將範例別名替換成 KMS 金鑰的有效識別符。如要執行此命令，您必須擁有 KMS 金鑰的 `kms:GetPublicKey` 許可。

```
$ aws kms get-public-key --key-id alias/example_RSA_3072

{
  "KeySpec": "RSA_3072",
  "KeyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "KeyUsage": "ENCRYPT_DECRYPT",
  "EncryptionAlgorithms": [
    "RSAES_OAEP_SHA_1",
    "RSAES_OAEP_SHA_256"
  ],
  "PublicKey": "MIIBojANBgkqhkiG..."
}
```

離線操作範例

[下載非對稱 KMS 金鑰對的公有金鑰](#)之後，您可以與他人共用，並使用它來執行離線操作。

AWS CloudTrail 記錄每個 AWS KMS 操作的日誌，包括請求、回應、日期、時間和授權使用者，不會記錄 外部公有金鑰的使用 AWS KMS。

本主題提供範例離線操作，以及工具 AWS KMS 提供的詳細資訊，讓您更輕鬆地進行離線操作。

主題

- [離線衍生共用秘密](#)
- [使用 ML-DSA 金鑰對進行離線驗證](#)
- [使用 SM2 金鑰對進行離線驗證 \(僅限中國區域\)](#)

離線衍生共用秘密

您可以[下載 ECC 金鑰對的公有金鑰](#)，用於離線操作，也就是 外部的操作 AWS KMS。

下列 [OpenSSL](#) 演練示範一種在外部使用 ECC KMS 金鑰對的 AWS KMS 公有金鑰和使用 OpenSSL 建立的私有金鑰衍生共用秘密的方法。

1. 在 OpenSSL 中建立 ECC 金鑰對，並準備與 搭配使用 AWS KMS。

```
// Create an ECC key pair in OpenSSL and save the private key in
openssl_ecc_key_priv.pem
export OPENSSL_CURVE_NAME="P-256"
export KMS_CURVE_NAME="ECC_NIST_P256"

export OPENSSL_KEY1_PRIV_PEM="openssl_ecc_key1_priv.pem"
openssl ecparam -name ${OPENSSL_CURVE_NAME} -genkey -out ${OPENSSL_KEY1_PRIV_PEM}

// Derive the public key from the private key
export OPENSSL_KEY1_PUB_PEM="openssl_ecc_key1_pub.pem"
openssl ec -in ${OPENSSL_KEY1_PRIV_PEM} -pubout -outform pem \
    -out ${OPENSSL_KEY1_PUB_PEM}

// View the PEM file containing the public key and extract the public key as a
// Base64 encoded string into OPENSSL_KEY1_PUB_BASE64 for use with AWS KMS
export OPENSSL_KEY1_PUB_BASE64=`cat ${OPENSSL_KEY1_PUB_PEM} | \
    tee /dev/stderr | grep -v "PUBLIC KEY" | tr -d "\n"`
```

2. 在中建立 ECC 金鑰協議金鑰對，AWS KMS 並準備與 OpenSSL 搭配使用。

```
// Create a KMS key on the same curve as the key pair from step 1
// with a key usage of KEY_AGREEMENT
// Save its ARN in KMS_KEY1_ARN.
export KMS_KEY1_ARN=`aws kms create-key --key-spec ${KMS_CURVE_NAME} \
    --key-usage KEY_AGREEMENT | tee /dev/stderr | jq -r .KeyMetadata.Arn`

// Download the public key and save the Base64-encoded version in KMS_KEY1_PUB_BASE64

export KMS_KEY1_PUB_BASE64=`aws kms get-public-key --key-id ${KMS_KEY1_ARN} | \
    tee /dev/stderr | jq -r .PublicKey`

// Create a PEM file for the public KMS key for use with OpenSSL
export KMS_KEY1_PUB_PEM="aws_kms_ecdh_key1_pub.pem"
echo "-----BEGIN PUBLIC KEY-----" > ${KMS_KEY1_PUB_PEM}
echo ${KMS_KEY1_PUB_BASE64} | fold -w 64 >> ${KMS_KEY1_PUB_PEM}
echo "-----END PUBLIC KEY-----" >> ${KMS_KEY1_PUB_PEM}
```

3. 使用 OpenSSL 中的私有金鑰和公有 KMS 金鑰，衍生 OpenSSL 中的共用秘密。

```
export OPENSSL_SHARED_SECRET1_BIN="openssl_shared_secret1.bin"
openssl pkeyutl -derive -inkey ${OPENSSL_KEY1_PRIV_PEM} \
```

```
-peerkey ${KMS_KEY1_PUB_PEM} -out ${OPENSSL_SHARED_SECRET1_BIN}
```

使用 ML-DSA 金鑰對進行離線驗證

AWS KMS 支援 ML-DSA 簽署的對沖變體，如[聯邦資訊處理標準 \(FIPS\) 204 標準](#)第 3.4 節所述，適用於高達 4 KB 位元組的訊息。

若要簽署大於 4 KB 的訊息，請在 外部執行訊息預先處理步驟 AWS KMS。此雜湊步驟會建立 64 位元組訊息代表 μ ，如 NIST FIPS 204 第 6.2 節所定義。

AWS KMS 對於大於 EXTERNAL_MU 4 KB 的訊息，有一個名為 的訊息類型。當您使用此功能而非 RAW 訊息類型時，請執行下列動作 AWS KMS：

- 假設您已執行雜湊步驟
- 略過其內部雜湊程序
- 適用於任何大小的訊息

驗證訊息時，您使用的方法取決於外部系統或程式庫的大小限制，以及是否支援 64 位元組訊息代表 μ ：

- 如果訊息小於大小限制，請使用 RAW 訊息類型。
- 如果訊息大於大小限制，請在外部系統中使用 代表 μ 。

下列各節示範如何使用 簽署訊息 AWS KMS，並使用 OpenSSL 驗證訊息。我們提供低於和超過 4 KB 訊息大小限制的訊息範例 AWS KMS。OpenSSL 不會對訊息大小施加限制以進行驗證。

對於這兩個範例，請先從 取得公有金鑰 AWS KMS。使用下列 AWS CLI 命令：

```
aws kms get-public-key \
  --key-id _<1234abcd-12ab-34cd-56ef-1234567890ab>_ \
  --output text \
  --query PublicKey | base64 --decode > public_key.der
```

訊息大小小於 4KB

對於小於 4 KB 的訊息，請使用 RAW 訊息類型 AWS KMS。雖然您可以使用 EXTERNAL_MU，但大小限制內的訊息並不需要。

使用下列 AWS CLI 命令來簽署訊息：

```
aws kms sign \  
  --key-id _<1234abcd-12ab-34cd-56ef-1234567890ab>_ \  
  --message 'your message' \  
  --message-type RAW \  
  --signing-algorithm ML_DSA_SHAKE_256 \  
  --output text \  
  --query Signature | base64 --decode > ExampleSignature.bin
```

若要使用 OpenSSL 驗證此訊息，請使用下列命令：

```
echo -n 'your message' | ./openssl dgst -verify public_key.der -signature  
ExampleSignature.bin
```

訊息大小超過 4KB

若要簽署大於 4KB 的訊息，請使用 EXTERNAL_MU 訊息類型。使用時EXTERNAL_MU，您會預先將訊息置於 NIST FIPS 204 第 6.2 節所定義的 64 位元組代表 μ 外部，並將其傳遞給簽署或驗證操作。請注意，這與 NIST FIPS 204 第 5.4 節中定義的「雜湊前 MLDSA」或 HashML-DSA 不同。

1. 首先，建構訊息字首。字首包含網域分隔符號、任何內容的長度，以及內容。網域分隔符號和內容長度的預設值為零。
2. 在訊息前面加上訊息字首。
3. 使用 SHAKE256 雜湊公有金鑰，並在步驟 2 的結果前面加上。
4. 最後，雜湊步驟 3 的結果以產生 64 位元組的 EXTERNAL_MU。

下列範例使用 OpenSSL 3.5 來建構 EXTERNAL_MU：

```
{  
  openssl asn1parse -inform DER -in public_key.der -strparse 17 -noout -out - 2>/dev/  
null |  
  openssl dgst -provider default -shake256 -xoflen 64 -binary;  
  printf '\x00\x00';  
  echo -n "your message"  
} | openssl dgst -provider default -shake256 -xoflen 64 -binary > mu.bin
```

建立mu.bin檔案之後，請使用下列命令呼叫 AWS KMS API 來簽署訊息：

```
aws kms sign \  
  --key-id _<1234abcd-12ab-34cd-56ef-1234567890ab>_ \  
  --message fileb://mu.bin \  
  --message-type EXTERNAL_MU \  
  --signing-algorithm ML_DSA_SHAKE_256 \  
  --output text \  
  --query Signature | base64 --decode > ExampleSignature.bin
```

產生的簽章與原始訊息上的RAW簽章相同。您可以使用相同的 OpenSSL 3.5 命令來驗證訊息：

```
echo -n 'your message' | ./openssl dgst -verify public_key.der -signature  
ExampleSignature.bin
```

使用 SM2 金鑰對進行離線驗證 (僅限中國區域)

若要 AWS KMS 使用 SM2 公有金鑰驗證 外部的簽章，您必須指定辨別 ID。當您將原始訊息傳遞 [MessageType:RAW](#) 至 [Sign](#) API 時，AWS KMS 會使用 OSCCA 在 GM/T 0009-2012 中 1234567812345678 定義的預設辨別 ID。您無法在 AWS KMS 中指定自己的辨別 ID。

不過，如果您要在 外部產生訊息摘要 AWS，您可以指定自己的辨別 ID，然後將訊息摘要 傳遞 [MessageType:DIGEST](#) 給 AWS KMS 以簽署。若要執行此操作，請變更 `SM2OfflineOperationHelper` 類別中的 `DEFAULT_DISTINGUISHING_ID` 值。您指定的辨別 ID 可以是最長 8,192 個字元的任何字串。AWS KMS 簽署訊息摘要後，您需要訊息摘要或訊息，以及用來計算摘要以離線驗證摘要的辨別 ID。

Important

`SM2OfflineOperationHelper` 參考程式碼設計為與 [Bouncy Castle](#) 1.68 版相容。如需其他版本的幫助，請聯絡 [bouncycastle.org](#)。

SM2OfflineOperationHelper 類別

為了協助您使用 SM2 金鑰進行離線操作，適用於 Java 的 `SM2OfflineOperationHelper` 類別具有可為您執行任務的方法。您可以使用此輔助程式類別作為針對其他密碼提供者的模型。

在其中 AWS KMS，會自動進行原始加密文字轉換和 SM2DSA 訊息摘要計算。並非所有加密提供者都以相同的方式實施 SM2。有些程式庫，例如 [OpenSSL](#) 1.1.1 版和更新版本，會自動執行這些動作。在

測試 OpenSSL 3.0 版時 AWS KMS 確認此行為。使用以下 `SM2OfflineOperationHelper` 類別和像是 [Bouncy Castle](#) 之類的程式庫，則需要您手動執行這些轉換和計算。

所以 `SM2OfflineOperationHelper` 類別提供了進行以下離線操作的方法：

- 訊息摘要計算

若要產生離線訊息摘要，供您用於離線驗證，或您可以傳遞給 AWS KMS 進行簽署，請使用 `calculateSM2Digest` 方法。`calculateSM2Digest` 會使用 SM3 雜湊演算法產生訊息摘要。[GetPublicKey](#) API 會傳回二進位格式的公有金鑰。您必須將二進位金鑰剖析為 `Java PublicKey`。提供剖析好的公有金鑰以及訊息。該方法會自動將您的訊息與預設的辨別 ID (1234567812345678) 相結合，但您可以藉由變更 `DEFAULT_DISTINGUISHING_ID` 值來設置自己的辨別 ID。

- 確認

若要離線驗證簽署，請使用 `offlineSM2DSAVerify` 方法。`offlineSM2DSAVerify` 方法使用從指定辨別 ID 計算出的訊息摘要，以及您提供的原始訊息來驗證數位簽署。[GetPublicKey](#) API 會傳回二進位格式的公有金鑰。您必須將二進位金鑰剖析為 `Java PublicKey`。提供已剖析的公開金鑰，以及您要驗證的原始訊息和簽章。如需詳細資訊，請參閱[使用 SM2 金鑰對進行離線驗證](#)。

- 加密

若要離線加密明文，請使用 `offlineSM2PKEEncrypt` 方法。此方法可確保加密文字的格式 AWS KMS 可以解密。`offlineSM2PKEEncrypt` 方法加密明文，然後將由 SM2PKE 產生的原始密文轉換為 ASN.1 格式。[GetPublicKey](#) API 會傳回二進位格式的公有金鑰。您必須將二進位金鑰剖析為 `Java PublicKey`。提供剖析好的公有金鑰以及以您要加密的明文。

如果您不確定是否需要執行轉換，請使用以下 OpenSSL 操作來測試您的密文格式。如果操作失敗，則需要將密文轉換為 ASN.1 格式。

```
openssl asn1parse -inform DER -in ciphertext.der
```

根據預設，當產生用於 SM2DSA 操作的訊息摘要時，`SM2OfflineOperationHelper` 類別使用預設的辨別 ID，即 1234567812345678。

```
package com.amazon.kms.utils;

import javax.crypto.BadPaddingException;
```

```
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.NoSuchPaddingException;
import java.io.IOException;
import java.math.BigInteger;
import java.nio.ByteBuffer;
import java.nio.charset.StandardCharsets;
import java.security.InvalidKeyException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.security.NoSuchProviderException;
import java.security.PrivateKey;
import java.security.PublicKey;

import org.bouncycastle.crypto.CryptoException;
import org.bouncycastle.jce.interfaces.ECPublicKey;

import java.util.Arrays;

import org.bouncycastle.asn1.ASN1EncodableVector;
import org.bouncycastle.asn1.ASN1Integer;
import org.bouncycastle.asn1.DEROctetString;
import org.bouncycastle.asn1.DERSequence;
import org.bouncycastle.asn1.gm.GMNamedCurves;
import org.bouncycastle.asn1.x9.X9ECParameters;
import org.bouncycastle.crypto.CipherParameters;
import org.bouncycastle.crypto.params.ParametersWithID;
import org.bouncycastle.crypto.params.ParametersWithRandom;
import org.bouncycastle.crypto.signers.SM2Signer;
import org.bouncycastle.jcajce.provider.asymmetric.util.ECUtil;

public class SM2OfflineOperationHelper {
    // You can change the DEFAULT_DISTINGUISHING_ID value to set your own
    // distinguishing ID,
    // the DEFAULT_DISTINGUISHING_ID can be any string up to 8,192 characters long.
    private static final byte[] DEFAULT_DISTINGUISHING_ID =
"1234567812345678".getBytes(StandardCharsets.UTF_8);
    private static final X9ECParameters SM2_X9EC_PARAMETERS =
GMNamedCurves.getByName("sm2p256v1");

    // ***calculateSM2Digest***
    // Calculate message digest
    public static byte[] calculateSM2Digest(final PublicKey publicKey, final byte[]
message) throws
```

```

        NoSuchProviderException, NoSuchAlgorithmException {
        final ECPublicKey ecPublicKey = (ECPublicKey) publicKey;

        // Generate SM3 hash of default distinguishing ID, 1234567812345678
        final int entlenA = DEFAULT_DISTINGUISHING_ID.length * 8;
        final byte [] entla = new byte[] { (byte) (entlenA & 0xFF00), (byte) (entlenA &
0x00FF) };
        final byte [] a = SM2_X9EC_PARAMETERS.getCurve().getA().getEncoded();
        final byte [] b = SM2_X9EC_PARAMETERS.getCurve().getB().getEncoded();
        final byte [] xg = SM2_X9EC_PARAMETERS.getG().getXCoord().getEncoded();
        final byte [] yg = SM2_X9EC_PARAMETERS.getG().getYCoord().getEncoded();
        final byte[] xa = ecPublicKey.getQ().getXCoord().getEncoded();
        final byte[] ya = ecPublicKey.getQ().getYCoord().getEncoded();
        final byte[] za = MessageDigest.getInstance("SM3", "BC")
            .digest(ByteBuffer.allocate(entla.length +
DEFAULT_DISTINGUISHING_ID.length + a.length + b.length + xg.length + yg.length +
            xa.length +
ya.length).put(entla).put(DEFAULT_DISTINGUISHING_ID).put(a).put(b).put(xg).put(yg).put(xa).put
            .array());

        // Combine hashed distinguishing ID with original message to generate final
digest
        return MessageDigest.getInstance("SM3", "BC")
            .digest(ByteBuffer.allocate(za.length +
message.length).put(za).put(message)
            .array());
    }

    // ***offlineSM2DSAVerify***
    // Verify digital signature with SM2 public key
    public static boolean offlineSM2DSAVerify(final PublicKey publicKey, final byte []
message,
        final byte [] signature) throws InvalidKeyException {
        final SM2Signer signer = new SM2Signer();
        CipherParameters cipherParameters =
ECUtil.generatePublicKeyParameter(publicKey);
        cipherParameters = new ParametersWithID(cipherParameters,
DEFAULT_DISTINGUISHING_ID);
        signer.init(false, cipherParameters);
        signer.update(message, 0, message.length);
        return signer.verifySignature(signature);
    }

    // ***offlineSM2PKEEncrypt***

```

```

// Encrypt data with SM2 public key
public static byte[] offlineSM2PKEEncrypt(final PublicKey publicKey, final byte []
plaintext) throws
    NoSuchPaddingException, NoSuchAlgorithmException, NoSuchProviderException,
    InvalidKeyException,
    BadPaddingException, IllegalBlockSizeException, IOException {
    final Cipher sm2Cipher = Cipher.getInstance("SM2", "BC");
    sm2Cipher.init(Cipher.ENCRYPT_MODE, publicKey);

    // By default, Bouncy Castle returns raw ciphertext in the c1c2c3 format
    final byte [] cipherText = sm2Cipher.doFinal(plaintext);

    // Convert the raw ciphertext to the ASN.1 format before passing it to AWS KMS
    final ASN1EncodableVector asn1EncodableVector = new ASN1EncodableVector();
    final int coordinateLength = (SM2_X9EC_PARAMETERS.getCurve().getFieldSize() +
7) / 8 * 2 + 1;
    final int sm3HashLength = 32;
    final int xCoordinateInCipherText = 33;
    final int yCoordinateInCipherText = 65;
    byte[] coords = new byte[coordinateLength];
    byte[] sm3Hash = new byte[sm3HashLength];
    byte[] remainingCipherText = new byte[cipherText.length - coordinateLength -
sm3HashLength];

    // Split components out of the ciphertext
    System.arraycopy(cipherText, 0, coords, 0, coordinateLength);
    System.arraycopy(cipherText, cipherText.length - sm3HashLength, sm3Hash, 0,
sm3HashLength);
    System.arraycopy(cipherText, coordinateLength, remainingCipherText,
0, cipherText.length - coordinateLength - sm3HashLength);

    // Build standard SM2PKE ASN.1 ciphertext vector
    asn1EncodableVector.add(new ASN1Integer(new BigInteger(1,
Arrays.copyOfRange(coords, 1, xCoordinateInCipherText))));
    asn1EncodableVector.add(new ASN1Integer(new BigInteger(1,
Arrays.copyOfRange(coords, xCoordinateInCipherText, yCoordinateInCipherText))));
    asn1EncodableVector.add(new DEROctetString(sm3Hash));
    asn1EncodableVector.add(new DEROctetString(remainingCipherText));

    return new DERSequence(asn1EncodableVector).getEncoded("DER");
}
}

```

監控 AWS KMS keys

監控是了解 AWS KMS keys 中的可用性、狀態和用量 AWS KMS，以及維護 AWS 解決方案的可靠性、可用性和效能的重要部分。收集 AWS 解決方案全面的監控資料，可協助在出現多點故障時進行偵錯。不過，在您開始監控 KMS 金鑰之前，應先建立監控計畫，為下列問題提供解答：

- 監控目標是什麼？
- 要監控哪些資源？
- 監控這些資源的頻率為何？
- 要使用哪些[監控工具](#)？
- 誰將執行監控任務？
- 發生問題時應該通知誰？

下一個步驟是隨時間監控您的 KMS 金鑰，以建立您環境中 AWS KMS 正常使用和期望的基準。當您監控 KMS 金鑰時，請存放歷史記錄監控資料，如此才能與目前的資料做比較、辨識正常模式和異常狀況、規劃問題處理方式。

例如，您可以監控會影響 KMS 金鑰的 AWS KMS API 活動和事件。當資料高於或低於既定常規，您可能需要調查或採取修正動作。

若要建立正常模式的基準，請監控下列項目：

- AWS KMS 資料平面操作的 API 活動。這些是使用 KMS 金鑰的[密碼編譯操作](#)，例如 [Decrypt](#)、[Encrypt](#)、[ReEncrypt](#) 和 [GenerateDataKey](#)。
- AWS KMS 控制平面操作的 API 活動對您來說很重要。這些操作管理 KMS 金鑰，而且您可能希望監控會變更 KMS 金鑰可用性 (例如 [ScheduleKeyDeletion](#)、[CancelKeyDeletion](#)、[DisableKey](#)、[EnableKey](#)、[ImportKeyMaterial](#) 和 [DeleteImportedKeyMaterial](#)) 或變更 KMS 金鑰存取控制 (例如 [PutKeyPolicy](#) 和 [RevokeGrant](#)) 的項目。
- 其他 AWS KMS 指標 (例如在您[匯入金鑰材料](#)過期之前剩餘的時間) 和事件 (例如匯入金鑰材料過期或 KMS 金鑰的刪除或金鑰輪換)。

監控工具

AWS 提供各種工具，可讓您用來監控 KMS 金鑰。您可以設定其中一些工具來進行監控，但有些工具需要手動介入。建議您盡可能自動化監控任務。

自動化監控工具

您可以使用下列自動化監控工具來監看 KMS 金鑰，並在發生變更時進行回報。

- AWS CloudTrail 日誌監控 – 在帳戶之間共用日誌檔案、透過將日誌檔案傳送到 CloudTrail CloudWatch Logs 來即時監控 CloudTrail 日誌檔案、使用 [CloudTrail Processing Library](#) 寫入日誌處理應用程式，以及驗證您的日誌檔案在 CloudTrail 交付後並未變更。如需詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的[使用 CloudTrail 日誌檔案](#)。
- Amazon CloudWatch 警示：監看指定時段內的單一指標，並根據與多個時段內給定之閾值相對的指標值來執行一或多個動作。此動作是傳送到 Amazon Simple Notification Service (Amazon SNS) 主題或 Amazon EC2 Auto Scaling 政策的通知。CloudWatch 警示不會只因處於特定狀態就調用動作，狀態必須已變更並已維持一段指定的時間。如需詳細資訊，請參閱[使用 Amazon CloudWatch 監控 KMS 金鑰](#)。
- Amazon EventBridge – 匹配事件並將事件路由至一或多個目標函數或串流，以擷取狀態資訊，如果需要的話，也能進行變更或採取修正動作。如需詳細資訊，請參閱[使用 Amazon EventBridge 監控 KMS 金鑰](#) 和《[Amazon EventBridge 使用者指南](#)》。
- Amazon CloudWatch Logs – 從 AWS CloudTrail 或其他來源監控、存放和存取您的日誌檔案。如需詳細資訊，請參閱[Amazon CloudWatch Logs 使用者指南](#)。

手動監控工具

監控 KMS 金鑰的另一個重要部分是手動監控 CloudWatch 警示和事件未涵蓋的項目。AWS KMS、CloudWatch AWS Trusted Advisor和其他 AWS 儀表板可讓您at-a-glance檢視環境 AWS 的狀態。

您可以[自訂 AWS KMS 主控台](#)的 AWS 受管金鑰 和客戶受管金鑰頁面，以顯示有關每個 KMS 金鑰的下列資訊：

- 金鑰 ID
- 狀態
- Creation (建立) 日期

- 過期日期 (適用於具有[匯入金鑰材料](#)的 KMS 金鑰)
- Origin
- 自訂金鑰存放區 ID (適用於[自訂金鑰存放區](#)中的 KMS 金鑰)

[CloudWatch 主控台儀表板](#)會顯示下列項目：

- 目前警示與狀態
- 警示與資源的圖表
- 服務運作狀態

此外，您可以使用 CloudWatch 執行下列動作：

- 建立[自定儀表板](#)來監控您注重的服務
- 用於疑難排解問題以及探索趨勢的圖形指標資料。
- 搜尋和瀏覽您的所有 AWS 資源指標
- 建立與編輯要通知發生問題的警示

AWS Trusted Advisor 可協助您監控 AWS 資源，以提高效能、可靠性、安全性和成本效益。所有使用者皆可使用四個 Trusted Advisor 檢查；商業或企業支援計劃的使用者可使用超過 50 個檢查。如需詳細資訊，請參閱[AWS Trusted Advisor](#)。

使用 記錄 AWS KMS API 呼叫 AWS CloudTrail

AWS KMS 已與 整合[AWS CloudTrail](#)，此服務會記錄 AWS KMS 使用者、角色和其他 AWS 服務對的所有呼叫。CloudTrail 會將對的所有 API 呼叫擷取 AWS KMS 為事件，包括來自 AWS KMS 主控台、AWS KMS APIs、CloudFormation 範本、AWS Command Line Interface (AWS CLI) 和 的呼叫 AWS Tools for PowerShell。

CloudTrail 會記錄所有 AWS KMS 操作，包括唯讀操作，例如 [ListAliases](#) 和 [GetKeyRotationStatus](#)、管理 KMS 金鑰的操作，例如 [CreateKey](#) 和 [PutKeyPolicy](#)，以及[密碼編譯操作](#)，例如 [GenerateDataKey](#) 和 [Decrypt](#)。它也會記錄 為您 AWS KMS 呼叫的內部操作，例如 [DeleteExpiredKeyMaterial](#)、[DeleteKey](#)、[SynchronizeMultiRegionKey](#) 和 [RotateKey](#)。

CloudTrail 會記錄所有成功的操作，在某些情況下會嘗試失敗的呼叫，例如當呼叫者無法存取資源時。[對 KMS 金鑰的跨帳戶操作](#)會同時記入呼叫者帳戶和 KMS 金鑰擁有者帳戶。不過，因為存取遭拒而遭到拒絕的跨帳戶 AWS KMS 請求只會記錄在發起人的帳戶中。

基於安全考量，AWS KMS 日誌項目會省略某些欄位，例如 [Encrypt](#) 請求的 Plaintext 參數，以及對 [GetKeyPolicy](#) 或任何密碼編譯操作的回應。為了更輕鬆地搜尋特定 KMS 金鑰的 CloudTrail 日誌項目，會將受影響 KMS 金鑰的 [金鑰 ARN](#) 新增至某些 AWS KMS 金鑰管理操作日誌項目中的 responseElements 欄位，即使 API 操作未傳回金鑰 ARN。

雖然預設會將所有 AWS KMS 動作記錄為 CloudTrail 事件，但您可以從 CloudTrail 追蹤中排除 AWS KMS 動作。如需詳細資訊，請參閱 [從線索排除 AWS KMS 事件](#)。

進一步了解：

- 如需已驗證平台 AWS KMS 操作的 CloudTrail 日誌範例，請參閱 [監控已驗證的請求](#)。

主題

- [在 CloudTrail 中尋找 AWS KMS 日誌項目](#)
- [從線索排除 AWS KMS 事件](#)
- [AWS KMS 日誌項目的範例](#)

在 CloudTrail 中尋找 AWS KMS 日誌項目

若要搜尋 CloudTrail 日誌項目，請使用 [CloudTrail 主控台](#) 或 [CloudTrail LookupEvents](#) 操作。CloudTrail 支援多種 [屬性值](#) 來篩選搜尋，包括事件名稱、使用者名稱和事件來源。

為了協助您在 CloudTrail 中搜尋 AWS KMS 日誌項目，會 AWS KMS 填入下列 CloudTrail 日誌項目欄位。

Note

從 2022 年 12 月開始，會在變更特定 KMS 金鑰的所有管理操作中 AWS KMS 填入資源類型和資源名稱屬性。對於下列操作，較舊 CloudTrail 項目中的這些屬性值可能為空值：[CreateAlias](#)、[CreateGrant](#)、[DeleteAlias](#)、[DeleteImportedKeyMaterial](#)、[ImportKeyMaterial](#)、[Replicate](#) 和 [UpdatePrimaryRegion](#)。

屬性	Value	日誌項目
事件來源 (EventSource)	kms.amazonaws.com	所有操作。

屬性	Value	日誌項目
資源類型 (ResourceType)	AWS::KMS::Key	可變更特定 KMS 金鑰的管理操作，例如 CreateKey 和 EnableKey ，但不包括 ListKeys。
資源名稱 (ResourceName)	金鑰 ARN (或金鑰 ID 和金鑰 ARN)	可變更特定 KMS 金鑰的管理操作，例如 CreateKey 和 EnableKey ，但不包括 ListKeys。

為了協助您在特定 KMS 金鑰上尋找管理操作的日誌項目，AWS KMS 會在日誌項目的 `responseElements.keyId` 元素中記錄受影響 KMS 金鑰的金鑰 ARN，即使 AWS KMS API 操作未傳回金鑰 ARN。

例如，對 [DisableKey](#) 操作的成功呼叫不會在回應中傳回任何值，而是傳回空值，[DisableKey 日誌項目](#) 中的 `responseElements.keyId` 值包含停用的 KMS 金鑰的金鑰 ARN。

此功能於 2022 年 12 月新增，會影響下列 CloudTrail 日誌項

目：[CreateAlias](#)、[CreateGrant](#)、[DeleteAlias](#)、[DeleteKey](#)、[DisableKey](#)、[EnableKey](#)、[EnableKeyRotation](#)、[UpdatePrimaryRegion](#)。

從線索排除 AWS KMS 事件

為了提供 AWS KMS 資源使用和管理的記錄，大多數 AWS KMS 使用者依賴 CloudTrail 追蹤中的事件。追蹤可以是稽核重要事件的寶貴資料來源，例如建立、停用和刪除 AWS KMS keys、變更金鑰政策，以及由 AWS 服務代表您使用 KMS 金鑰。在某些情況下，CloudTrail 日誌項目中的中繼資料，例如加密操作中的[加密內容](#)，可協助您避免或解決錯誤。

不過，由於 AWS KMS 可以產生大量事件，因此 AWS CloudTrail 會讓您從追蹤中排除 AWS KMS 事件。此每軌設定會排除所有 AWS KMS 事件；您無法排除特定 AWS KMS 事件。

Warning

從 CloudTrail Log 排除 AWS KMS 事件可能會隱藏使用您的 KMS 金鑰的動作。授予委託人執行此操作所需的 `cloudtrail:PutEventSelectors` 許可時時，請務必小心。

若要從追蹤中排除 AWS KMS 事件：

- 在 CloudTrail 主控台中，當您[建立追蹤](#)或[更新追蹤](#)時，請使用日誌金鑰管理服務事件設定。如需說明，請參閱 AWS CloudTrail 《使用者指南》中的[使用 記錄管理事件 AWS 管理主控台](#)。
- 在 CloudTrail API 中，使用 [PutEventSelectors](#) 操作。將 ExcludeManagementEventSources 屬性新增加到您的事件選擇器，其值為 kms.amazonaws.com。如需範例，請參閱 AWS CloudTrail 《使用者指南》中的[範例：不記錄 AWS Key Management Service 事件的線索](#)。

您可以變更主控台設定或線索的事件選擇器，以隨時停用此排除。然後，線索會開始記錄 AWS KMS 事件。不過，它無法復原排除生效時發生的 AWS KMS 事件。

當您使用主控台或 API 排除 AWS KMS 事件時，產生的 CloudTrail PutEventSelectors API 操作也會記錄在 CloudTrail 日誌中。如果 AWS KMS 事件未出現在 CloudTrail Logs 中，請尋找 ExcludeManagementEventSources 屬性設為 kms.amazonaws.com 的 PutEventSelectors 事件。

AWS KMS 日誌項目的範例

AWS KMS 當您呼叫 AWS KMS 操作時，以及當服務代您呼叫 操作時，AWS 會將項目寫入您的 CloudTrail 日誌。AWS KMS 也會在為您呼叫 操作時寫入項目。例如，當[刪除您排定刪除的 KMS 金鑰](#)時，它會寫入一條項目。

下列主題顯示 AWS KMS 操作的 CloudTrail 日誌項目範例。

如需 AWS KMS 來自已驗證平台之 請求的 CloudTrail 日誌項目範例，請參閱 [監控已驗證的請求](#)。

主題

- [CancelKeyDeletion](#)
- [ConnectCustomKeyStore](#)
- [CreateAlias](#)
- [CreateCustomKeyStore](#)
- [CreateGrant](#)
- [CreateKey](#)
- [解密](#)
- [DeleteAlias](#)
- [DeleteCustomKeyStore](#)
- [DeleteExpiredKeyMaterial](#)

- [DeleteImportedKeyMaterial](#)
- [DeleteKey](#)
- [DescribeCustomKeyStores](#)
- [DescribeKey](#)
- [DisableKey](#)
- [DisableKeyRotation](#)
- [DisconnectCustomKeyStore](#)
- [EnableKey](#)
- [EnableKeyRotation](#)
- [加密](#)
- [GenerateDataKey](#)
- [GenerateDataKeyPair](#)
- [GenerateDataKeyPairWithoutPlaintext](#)
- [GenerateDataKeyWithoutPlaintext](#)
- [GenerateMac](#)
- [GenerateRandom](#)
- [GetKeyPolicy](#)
- [GetKeyRotationStatus](#)
- [GetParametersForImport](#)
- [ImportKeyMaterial](#)
- [ListAliases](#)
- [ListGrants](#)
- [ListKeyRotations](#)
- [PutKeyPolicy](#)
- [ReEncrypt](#)
- [ReplicateKey](#)
- [RetireGrant](#)
- [RevokeGrant](#)
- [RotateKey](#)
- [RotateKeyOnDemand](#)

- [ScheduleKeyDeletion](#)
- [符號](#)
- [SynchronizeMultiRegionKey](#)
- [TagResource](#)
- [UntagResource](#)
- [UpdateAlias](#)
- [UpdateCustomKeyStore](#)
- [UpdateKeyDescription](#)
- [UpdatePrimaryRegion](#)
- [VerifyMac](#)
- [確認](#)
- [Amazon EC2 範例 1](#)
- [Amazon EC2 範例 2](#)

CancelKeyDeletion

下列範例顯示呼叫 [CancelKeyDeletion](#) 操作所產生的 AWS CloudTrail 日誌項目。如需刪除的資訊 AWS KMS keys，請參閱 [刪除 AWS KMS key](#)。

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2020-07-27T21:53:17Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "CancelKeyDeletion",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
  }
}
```

```

    },
    "responseElements": {
      "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "e3452e68-d4b0-4ec7-a768-7ae96c23764f",
    "eventID": "d818bf03-6655-48e9-8b26-f279a07075fd",
    "readOnly": false,
    "resources": [
      {
        "accountId": "111122223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
  }
}

```

ConnectCustomKeyStore

下列範例顯示呼叫 [ConnectCustomKeyStore](#) 操作所產生的 AWS CloudTrail 日誌項目。如需連線自訂金鑰存放區的詳細資訊，請參閱 [中斷連接 AWS CloudHSM 金鑰存放區](#)。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2021-10-21T20:17:32Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "ConnectCustomKeyStore",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "customKeyStoreId": "cks-1234567890abcdef0"
  }
}

```

```

    },
    "responseElements": null,
    "additionalEventData": {
        "customKeyStoreName": "ExampleKeyStore",
        "clusterId": "cluster-1a23b4cdefg"
    },
    "requestID": "abcde9e1-f1a3-4460-a423-577fb6e695c9",
    "eventID": "114b61b9-0ea6-47f5-a9d2-4f2bdd0017d5",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
}

```

CreateAlias

下列範例顯示 [CreateAlias](#) 操作的 AWS CloudTrail 日誌項目。resources 元素包含別名和 KMS 金鑰資源的欄位。如需在 中建立別名的詳細資訊 AWS KMS，請參閱 [建立別名](#)。

在 2022 年 12 月或之後記錄的此操作的 CloudTrail 日誌項目會在 responseElements.keyId 值中包含受影響 KMS 金鑰的金鑰 ARN，即使此操作不會傳回金鑰 ARN。

```

{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2022-08-14T23:08:31Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateAlias",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "aliasName": "alias/ExampleAlias",
        "targetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": {

```



```

    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "requestID": "caec1e0c-ce03-419e-bdab-6ab1f7c57c01",
  "eventID": "2dd6e784-8286-46a6-befd-d64e5a02fb28",
  "readOnly": false,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:alias/ExampleAlias"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management"
}

```

CreateCustomKeyStore

下列範例顯示呼叫 AWS CloudHSM 金鑰存放區上的 [CreateCustomKeyStore](#) 操作所產生的 AWS CloudTrail 日誌項目。如需建立自訂金鑰存放區的詳細資訊，請參閱 [建立 AWS CloudHSM 金鑰存放區](#)。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2021-10-21T20:17:32Z",
  "eventSource": "kms.amazonaws.com",

```

```

    "eventName": "CreateCustomKeyStore",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "customKeyStoreName": "ExampleKeyStore",
        "clusterId": "cluster-1a23b4cdefg"
    },
    "responseElements": {
        "customKeyStoreId": "cks-1234567890abcdef0"
    },
    "requestID": "abcde9e1-f1a3-4460-a423-577fb6e695c9",
    "eventID": "114b61b9-0ea6-47f5-a9d2-4f2bdd0017d5",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
}

```

CreateGrant

下列範例顯示 [CreateGrant](#) 操作的 AWS CloudTrail 日誌項目。如需在 中建立授與的詳細資訊 AWS KMS，請參閱 [在 中授予 AWS KMS](#)。

在 2022 年 12 月或之後記錄的此操作的 CloudTrail 日誌項目會在 `responseElements.keyId` 值中包含受影響 KMS 金鑰的金鑰 ARN，即使此操作不會傳回金鑰 ARN。

```

{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2014-11-04T00:53:12Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "CreateGrant",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",

```

```

"requestParameters": {
  "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "constraints": {
    "encryptionContextSubset": {
      "ContextKey1": "Value1"
    }
  },
  "operations": ["Encrypt",
  "RetireGrant"],
  "granteePrincipal": "EX_PRINCIPAL_ID"
},
"responseElements": {
  "grantId": "f020fe75197b93991dc8491d6f19dd3cebb24ee62277a05914386724f3d48758",
  "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
},
"requestID": "f3c08808-63bc-11e4-bc2b-4198b6150d5c",
"eventID": "5d529779-2d27-42b5-92da-91aaea1fc4b5",
"readOnly": false,
"resources": [{
  "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "accountId": "111122223333"
}],
"eventType": "AwsApiCall",
"recipientAccountId": "111122223333"
}

```

CreateKey

這些範例顯示 [CreateKey](#) 操作的 AWS CloudTrail 日誌項目。

CreateKey 日誌項目可能起因於 CreateKey 請求或 [ReplicateKey](#) 請求的 CreateKey 操作。

以下範例會顯示 [CreateKey](#) 操作的 CloudTrail 日誌項目，該操作會建立 [對稱加密 KMS 金鑰](#)。如需建立 KMS 金鑰的相關資訊，請參閱 [建立 KMS 金鑰](#)。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",

```

```

    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2022-08-10T22:38:27Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "CreateKey",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "description": "",
    "origin": "EXTERNAL",
    "bypassPolicyLockoutSafetyCheck": false,
    "customerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "keySpec": "SYMMETRIC_DEFAULT",
    "keyUsage": "ENCRYPT_DECRYPT"
  },
  "responseElements": {
    "keyMetadata": {
      "AWSAccountId": "111122223333",
      "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
      "arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "creationDate": "Aug 10, 2022, 10:38:27 PM",
      "enabled": false,
      "description": "",
      "keyUsage": "ENCRYPT_DECRYPT",
      "keyState": "PendingImport",
      "origin": "EXTERNAL",
      "keyManager": "CUSTOMER",
      "customerMasterKeySpec": "SYMMETRIC_DEFAULT",
      "keySpec": "SYMMETRIC_DEFAULT",
      "encryptionAlgorithms": [
        "SYMMETRIC_DEFAULT"
      ],
      "multiRegion": false
    }
  },
  "requestID": "1aef6713-0223-4ff7-9a6d-781360521930",
  "eventID": "36327b37-f4f6-40a9-92ab-48064ec905a2",
  "readOnly": false,
  "resources": [

```

```

    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management"
}

```

以下範例顯示會在 [AWS CloudHSM 金鑰存放區](#) 中建立對稱加密 KMS 金鑰之 CreateKey 操作的 CloudTrail 日誌。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2021-10-14T17:39:50Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "CreateKey",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyUsage": "ENCRYPT_DECRYPT",
    "bypassPolicyLockoutSafetyCheck": false,
    "origin": "AWS_CLOUDHSM",
    "keySpec": "SYMMETRIC_DEFAULT",
    "customerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "customKeyStoreId": "cks-1234567890abcdef0",
    "description": ""
  },
  "responseElements": {
    "keyMetadata": {

```

```

        "awsAccountId": "111122223333",
        "keyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
        "arn": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321",
        "creationDate": "Oct 14, 2021, 5:39:50 PM",
        "enabled": true,
        "description": "",
        "keyUsage": "ENCRYPT_DECRYPT",
        "keyState": "Enabled",
        "origin": "AWS_CLOUDHSM",
        "customKeyStoreId": "cks-1234567890abcdef0",
        "cloudHsmClusterId": "cluster-1a23b4cdefg",
        "keyManager": "CUSTOMER",
        "customerMasterKeySpec": "SYMMETRIC_DEFAULT",
        "keySpec": "SYMMETRIC_DEFAULT",
        "encryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ],
        "multiRegion": false
    },
    {
        "additionalEventData": {
            "backingKey": "{\"backingKeyId\": \"backing-key-id\"}"
        },
        "requestID": "4f0b185c-588c-4767-9e90-c618f7e13cad",
        "eventID": "c73964b8-703d-49e4-bd9e-f773d0ee1e65",
        "readOnly": false,
        "resources": [
            {
                "accountId": "111122223333",
                "type": "AWS::KMS::Key",
                "ARN": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321"
            }
        ],
        "eventType": "AwsApiCall",
        "managementEvent": true,
        "recipientAccountId": "111122223333",
        "eventCategory": "Management"
    }
}

```

以下範例顯示會在[外部金鑰存放區](#)中建立對稱加密 KMS 金鑰之 CreateKey 操作的 CloudTrail 日誌。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2022-09-07T22:37:45Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "CreateKey",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "tags": [],
    "keyUsage": "ENCRYPT_DECRYPT",
    "description": "",
    "origin": "EXTERNAL_KEY_STORE",
    "multiRegion": false,
    "keySpec": "SYMMETRIC_DEFAULT",
    "customerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "bypassPolicyLockoutSafetyCheck": false,
    "customKeyStoreId": "cks-1234567890abcdef0",
    "xksKeyId": "bb8562717f809024"
  },
  "responseElements": {
    "keyMetadata": {
      "awsAccountId": "111122223333",
      "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
      "arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "creationDate": "Dec 7, 2022, 10:37:45 PM",
      "enabled": true,
      "description": "",
      "keyUsage": "ENCRYPT_DECRYPT",
      "keyState": "Enabled",
      "origin": "EXTERNAL_KEY_STORE",
      "customKeyStoreId": "cks-1234567890abcdef0",
      "keyManager": "CUSTOMER",
      "customerMasterKeySpec": "SYMMETRIC_DEFAULT",

```

```

        "keySpec": "SYMMETRIC_DEFAULT",
        "encryptionAlgorithms": [
            "SYMMETRIC_DEFAULT"
        ],
        "multiRegion": false,
        "xksKeyConfiguration": {
            "id": "bb8562717f809024"
        }
    },
    "requestID": "ba197c82-3ac7-487a-8ff4-7736bbeb1316",
    "eventID": "838ad5f4-5fdd-4044-afd7-4dbd88c6af56",
    "readOnly": false,
    "resources": [
        {
            "accountId": "227179770375",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-east-1:227179770375:key/39c5eb22-f37c-4956-92ca-89e8f8b57ab2"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}

```

解密

這些範例顯示 [Decrypt](#) 操作的 AWS CloudTrail 日誌項目。

Decrypt 操作的 CloudTrail 日誌項目一律包含 requestParameters 中的 encryptionAlgorithm，即使未在請求中指定加密演算法。省略請求中的加密文字和回應中的純文字。

主題

- [使用標準對稱加密金鑰進行解密](#)
- [使用標準對稱加密金鑰進行解密失敗](#)
- [使用金鑰存放區中的 KMS AWS CloudHSM 金鑰進行解密](#)
- [使用外部金鑰存放區中的 KMS 金鑰進行解密](#)
- [使用外部金鑰存放區中的 KMS 金鑰進行解密失敗](#)

- [透過後量子 TLS 連線使用標準對稱加密金鑰進行解密](#)

使用標準對稱加密金鑰進行解密

以下是使用標準對稱加密金鑰之 Decrypt 操作的 CloudTrail 日誌項目範例。

```
{
  "eventVersion": "1.11",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2025-05-20T20:45:00Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "Decrypt",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "encryptionContext": {
      "Department": "Engineering",
      "Project": "Alpha"
    }
  },
  "responseElements": null,
  "additionalEventData": {
    "keyMaterialId": "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
  },
  "requestID": "12345126-30d5-4b28-98b9-9153da559963",
  "eventID": "abcde202-ba1a-467c-b4ba-f729d45ae521",
  "readOnly": true,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",

```

```

      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES256-GCM-SHA384",
    "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
  }
}

```

使用標準對稱加密金鑰進行解密失敗

以下 CloudTrail 日誌項目範例記錄了使用標準對稱加密 KMS 金鑰的失敗 Decrypt 操作。包括例外狀況 (errorCode) 和錯誤消息 (errorMessage)，可幫助您解決錯誤。

在這種情況下，在 Decrypt 請求中指定的對稱加密 KMS 金鑰不是用來加密資料的對稱加密 KMS 金鑰。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2022-11-24T18:57:43Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "Decrypt",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "errorCode": "IncorrectKeyException"
  "errorMessage": "The key ID in the request does not identify a CMK that can perform this operation.",
  "requestParameters": {

```

```

    "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
    "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "encryptionContext": {
        "Department": "Engineering",
        "Project": "Alpha"
    }
},
"responseElements": null,
"requestID": "22345126-30d5-4b28-98b9-9153da559963",
"eventID": "abcde202-ba1a-467c-b4ba-f729d45ae521",
"readOnly": true,
"resources": [
    {
        "accountId": "111122223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
],
"eventType": "AwsApiCall",
"recipientAccountId": "111122223333"
}

```

使用金鑰存放區中的 KMS AWS CloudHSM 金鑰進行解密

以下範例 CloudTrail 日誌項目會記錄 Decrypt 操作，它具有 [AWS CloudHSM 金鑰存放區](#) 的 KMS 金鑰。在自訂金鑰存放區中使用 KMS 金鑰進行密碼編譯操作的所有日誌項目都包含具有 customKeyId 和 additionalEventData 的欄位 backingKeyId。backingKeyId 欄位中傳回的值是 CloudHSM 金鑰 id 屬性。請求中未指定 additionalEventData。

```

{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2021-10-26T23:41:27Z",
    "eventSource": "kms.amazonaws.com",

```

```

    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "requestParameters": {
      "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
      "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "encryptionContext": {
        "Department": "Development",
        "Purpose": "Test"
      }
    },
    "responseElements": null,
    "additionalEventData": {
      "customKeyStoreId": "cks-1234567890abcdef0"
    },
    "requestID": "e1b881f8-2048-41f8-b6cc-382b7857ec61",
    "eventID": "a79603d5-4cde-46fc-819c-a7cf547b9df4",
    "readOnly": true,
    "resources": [
      {
        "accountId": "111122223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
  }
}

```

使用外部金鑰存放區中的 KMS 金鑰進行解密

以下範例 CloudTrail 日誌項目會記錄 Decrypt 操作，它具有[外部金鑰存放區](#)的 KMS 金鑰。除了 customKeyStoreId 之外，additionalEventData 欄位還包括[外部金鑰 ID](#) (XksKeyId)。請求中未指定 additionalEventData。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",

```

```

    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2022-11-24T00:26:58Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "Decrypt",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "AWS Internal",
  "requestParameters": {
    "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321",
    "encryptionContext": {
      "Department": "Engineering",
      "Purpose": "Test"
    }
  },
  "responseElements": null,
  "additionalEventData": {
    "customKeyStoreId": "cks-9876543210fedcba9",
    "xksKeyId": "abc01234567890fe"
  },
  "requestID": "f1b881f8-2048-41f8-b6cc-382b7857ec61",
  "eventID": "b79603d5-4cde-46fc-819c-a7cf547b9df4",
  "readOnly": true,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management"
}

```

使用外部金鑰存放區中的 KMS 金鑰進行解密失敗

以下範例 CloudTrail 日誌項目會記錄 Decrypt 操作的失敗請求，它具有[外部金鑰存放區](#)的 KMS 金鑰。除了成功的請求外，CloudWatch 還會記錄失敗的請求。當記錄失敗時，CloudTrail 日誌項目會包含例外狀況 (errorCode) 和對應的錯誤訊息 (errorMessage)。

如果失敗的請求到達您的外部金鑰存放區代理 (如本範例所示)，則您可以使用 requestId 值將失敗的請求與外部金鑰存放區代理所記錄的對應請求建立關聯 (如果代理提供的話)。

如需外部金鑰存放區中 Decrypt 請求的說明，請參閱 [解密錯誤](#)。

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2022-11-24T00:26:58Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "Decrypt",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "AWS Internal",
  "userAgent": "AWS Internal",
  "errorCode": "KMSInvalidStateException",
  "errorMessage": "The external key store proxy rejected the request because the specified ciphertext or additional authenticated data is corrupted, missing, or otherwise invalid.",
  "requestParameters": {
    "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321",
    "encryptionContext": {
      "Department": "Engineering",
      "Purpose": "Test"
    }
  },
  "responseElements": null,
  "additionalEventData": {
    "customKeyStoreId": "cks-9876543210fedcba9",
  }
```

```

    "xksKeyId": "abc01234567890fe"
  },
  "requestID": "f1b881f8-2048-41f8-b6cc-382b7857ec61",
  "eventID": "b79603d5-4cde-46fc-819c-a7cf547b9df4",
  "readOnly": true,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management"
}

```

透過後量子 TLS 連線使用標準對稱加密金鑰進行解密

以下是透過後量子 TLS 連線搭配標準對稱加密金鑰之Decrypt操作的 CloudTrail 日誌項目範例。tlsDetails 區段中的 keyExchange 欄位提及 X25519MLKEM768。這是混合式演算法，結合了 [Elliptic Curve Diffie-Hellman](#) (ECDH) 與 [Curve 25519](#)，這是 TLS 中目前使用的傳統金鑰交換演算法，搭配 [Module-Lattice-Based金鑰封裝機制](#) (ML-KEM)、公有金鑰加密和金鑰建立演算法，美國國家標準技術研究所 (NIST) [已指定為第一個標準](#)後量子金鑰協議演算法。

```

{
  "eventVersion": "1.11",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2025-11-12T15:16:26Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "Decrypt",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "aws-sdk-java/2.30.22 md/io#async md/http#AwsCommonRuntime ...",

```

```

    "requestParameters": {
      "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
      "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "encryptionContext": {
        "Department": "Engineering",
        "Project": "Alpha"
      }
    },
    "responseElements": null,
    "additionalEventData": {
      "keyMaterialId":
"123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0",
    },
    "requestID": "12345126-30d5-4b28-98b9-9153da559963",
    "eventID": "abcde202-ba1a-467c-b4ba-f729d45ae521",
    "readOnly": true,
    "resources": [
      {
        "accountId": "111122223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
      "tlsVersion": "TLSv1.3",
      "cipherSuite": "TLS_AES_256_GCM_SHA384",
      "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com",
      "keyExchange": "X25519MLKEM768"
    }
  }
}

```

DeleteAlias

下列範例顯示 [DeleteAlias](#) 操作的 AWS CloudTrail 日誌項目。如需刪除別名的詳細資訊，請參閱 [刪除別名](#)。

在 2022 年 12 月或之後記錄的此操作的 CloudTrail 日誌項目會在 `responseElements.keyId` 值中包含受影響 KMS 金鑰的金鑰 ARN，即使此操作不會傳回金鑰 ARN。

```
{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2014-11-04T00:52:27Z"
      }
    }
  },
  "eventTime": "2014-11-04T00:52:27Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "DeleteAlias",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "aliasName": "alias/my_alias"
  },
  "responseElements": {
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "requestID": "d9542792-63bc-11e4-bc2b-4198b6150d5c",
  "eventID": "12f48554-bb04-4991-9cfc-e7e85f68eda0",
  "readOnly": false,
  "resources": [{
    "ARN": "arn:aws:kms:us-east-1:111122223333:alias/my_alias",
    "accountId": "111122223333"
  },
  {
    "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "accountId": "111122223333"
  }
]
```

```
  }],  
  "eventType": "AwsApiCall",  
  "recipientAccountId": "111122223333"  
}
```

DeleteCustomKeyStore

下列範例顯示呼叫 [DeleteCustomKeyStore](#) 操作所產生的 AWS CloudTrail 日誌項目。如需建立自訂金鑰存放區的詳細資訊，請參閱 [刪除 AWS CloudHSM 金鑰存放區](#)。

```
{  
  "eventVersion": "1.08",  
  "userIdentity": {  
    "type": "IAMUser",  
    "principalId": "EX_PRINCIPAL_ID",  
    "arn": "arn:aws:iam::111122223333:user/Alice",  
    "accountId": "111122223333",  
    "accessKeyId": "EXAMPLE_KEY_ID",  
    "userName": "Alice"  
  },  
  "eventTime": "2021-10-21T20:17:32Z",  
  "eventSource": "kms.amazonaws.com",  
  "eventName": "DeleteCustomKeyStore",  
  "awsRegion": "us-east-1",  
  "sourceIPAddress": "192.0.2.0",  
  "userAgent": "AWS Internal",  
  "requestParameters": {  
    "customKeyStoreId": "cks-1234567890abcdef0"  
  },  
  "responseElements": null,  
  "additionalEventData": {  
    "customKeyStoreName": "ExampleKeyStore",  
    "clusterId": "cluster-1a23b4cdefg"  
  },  
  "requestID": "abcde9e1-f1a3-4460-a423-577fb6e695c9",  
  "eventID": "114b61b9-0ea6-47f5-a9d2-4f2bdd0017d5",  
  "readOnly": false,  
  "eventType": "AwsApiCall",  
  "managementEvent": true,  
  "recipientAccountId": "111122223333"  
}
```

DeleteExpiredKeyMaterial

當您將金鑰材料匯入 AWS KMS key (KMS 金鑰) 時，您可以設定該金鑰材料的過期日期和時間。當您匯入金鑰材料 (使用過期設定) 以及 刪除過期金鑰材料時，會在 CloudTrail 日誌中 AWS KMS 記錄項目。AWS KMS 如需有關建立具有匯入金鑰材料之 KMS 金鑰的資訊，請參閱 匯入金鑰的 AWS KMS 金鑰材料。

下列範例顯示當 AWS KMS 刪除過期金鑰材料時產生的 AWS CloudTrail 日誌項目。

```
{
  "eventVersion": "1.11",
  "userIdentity": {
    "accountId": "111122223333",
    "invokedBy": "AWS Internal"
  },
  "eventTime": "2025-05-22T19:55:11Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "DeleteExpiredKeyMaterial",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "AWS Internal",
  "userAgent": "AWS Internal",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "cfa932fd-0d3a-4a76-a8b8-616863a2b547",
  "readOnly": false,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ],
  "eventType": "AwsServiceEvent",
  "recipientAccountId": "111122223333",
  "serviceEventDetails": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "keyMaterialId": "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
  },
  "eventCategory": "Management"
}
```

DeleteImportedKeyMaterial

如果您將金鑰材料匯入 KMS 金鑰，則可以隨時使用 [DeleteImportedKeyMaterial](#) 操作來刪除匯入金鑰材料。從 KMS 金鑰中刪除匯入金鑰材料後，KMS 金鑰的金鑰狀態變更為 PendingImport，並且不能在任何密碼編譯操作中使用 KMS 金鑰。如需詳細資訊，請參閱[刪除匯入的金鑰材料](#)。

下列範例顯示為 DeleteImportedKeyMaterial 操作產生的 AWS CloudTrail 日誌項目。

```
{
  "eventVersion": "1.11",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2025-05-20T20:45:08Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "DeleteImportedKeyMaterial",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "keyMaterialId":
"123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
  },
  "responseElements": {
    "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "keyMaterialId":
"123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
  },
  "requestID": "dcf0e82f-dad0-4622-a378-a5b964ad42c1",
  "eventID": "2afbb991-c668-4641-8a00-67d62e1fecbd",
  "readOnly": false,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
```

```

      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
      "tlsVersion": "TLSv1.2",
      "cipherSuite": "ECDHE-RSA-AES256-GCM-SHA384",
      "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    }
  }
}

```

DeleteKey

這些範例顯示刪除 KMS 金鑰時產生的 AWS CloudTrail 日誌項目。若要刪除 KMS 金鑰，您可以使用 [ScheduleKeyDeletion](#) 操作。在指定的等待期過期後，AWS KMS 會刪除 KMS 金鑰，並在 CloudTrail 日誌中記錄類似以下內容的項目，以記錄該事件。

在 2022 年 12 月或之後記錄的此操作的 CloudTrail 日誌項目會在 `responseElements.keyId` 值中包含受影響 KMS 金鑰的金鑰 ARN，即使此操作不會傳回金鑰 ARN。

如需 `ScheduleKeyDeletion` 操作的 CloudTrail 日誌項目範例，請參閱 [ScheduleKeyDeletion](#)。如需刪除 KMS 金鑰的相關資訊，請參閱 [刪除 AWS KMS key](#)。

以下範例 CloudTrail 日誌項目會記錄具有 AWS KMS 中金鑰材料的 KMS 金鑰的 `DeleteKey` 操作。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "accountId": "111122223333",
    "invokedBy": "AWS Internal"
  },
  "eventTime": "2020-07-31T00:07:00Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "DeleteKey",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "AWS Internal",
  "userAgent": "AWS Internal",
  "requestParameters": null,
  "responseElements": null,

```

```

    "eventID": "b25f9cda-74e1-4458-847b-4972a0bf9668",
    "readOnly": false,
    "resources": [
      {
        "accountId": "111122223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      }
    ],
    "eventType": "AwsServiceEvent",
    "recipientAccountId": "111122223333",
    "managementEvent": true,
    "eventCategory": "Management"
  }
}

```

下列 CloudTrail 日誌項目會記錄 AWS CloudHSM [自訂金鑰存放區中 KMS 金鑰](#) DeleteKey 的操作。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "accountId": "111122223333",
    "invokedBy": "AWS Internal"
  },
  "eventTime": "2021-10-26T23:41:27Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "DeleteKey",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "AWS Internal",
  "userAgent": "AWS Internal",
  "requestParameters": null,
  "responseElements": {
    "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "additionalEventData": {
    "customKeyStoreId": "cks-1234567890abcdef0",
    "clusterId": "cluster-1a23b4cdefg",
    "backingKeys": "[{\"backingKeyId\": \"backing-key-id\"}]",
    "backingKeysDeletionStatus": "[{\"backingKeyId\": \"backing-key-id\",
\"deletionStatus\": \"SUCCESS\"}]"]
  },
  "eventID": "1234585c-4b0c-4340-ab11-662414b79239",

```

```

    "readOnly": false,
    "resources": [
      {
        "accountId": "111122223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      }
    ],
    "eventType": "AwsServiceEvent",
    "recipientAccountId": "111122223333",
    "managementEvent": true,
    "eventCategory": "Management"
  }
}

```

DescribeCustomKeyStores

下列範例顯示呼叫 [DescribeCustomKeyStores](#) 操作所產生的 AWS CloudTrail 日誌項目。如需檢視自訂金鑰存放區的詳細資訊，請參閱 [檢視 AWS CloudHSM 金鑰存放區](#)。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2021-10-21T20:17:32Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "DescribeCustomKeyStores",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "customKeyStoreId": "cks-1234567890abcdef0"
  },
  "responseElements": null,
  "requestID": "abcde9e1-f1a3-4460-a423-577fb6e695c9",
  "eventID": "2ea1735f-628d-43e3-b2ee-486d02913a78",
  "readOnly": true,
}

```

```
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "111122223333"
}
```

DescribeKey

下列範例顯示 [DescribeKey](#) 操作的 AWS CloudTrail 日誌項目。當您在 AWS KMS 主控台中呼叫 [DescribeKey](#) 操作或 [檢視 KMS 金鑰](#) 時，會 AWS KMS 記錄如下的項目。此呼叫是在 AWS KMS 管理主控台中檢視金鑰的結果。

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2022-09-26T18:01:36Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "DescribeKey",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "responseElements": null,
  "requestID": "12345126-30d5-4b28-98b9-9153da559963",
  "eventID": "abcde202-ba1a-467c-b4ba-f729d45ae521",
  "readOnly": true,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ],
  "eventType": "AwsApiCall",
```



```
"recipientAccountId": "111122223333"
}
```

DisableKey

下列範例顯示 [DisableKey](#) 操作的 AWS CloudTrail 日誌項目。如需在 AWS KMS keys 中啟用和停用的資訊 AWS KMS，請參閱 [啟用和停用金鑰](#)。

在 2022 年 12 月或之後記錄的此操作的 CloudTrail 日誌項目會在 `responseElements.keyId` 值中包含受影響 KMS 金鑰的金鑰 ARN，即使此操作不會傳回金鑰 ARN。

```
{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2014-11-04T00:52:43Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "DisableKey",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "responseElements": {
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "requestID": "12345126-30d5-4b28-98b9-9153da559963",
  "eventID": "abcde202-ba1a-467c-b4ba-f729d45ae521",
  "readOnly": false,
  "resources": [{
    "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "accountId": "111122223333"
  }],
  "eventType": "AwsApiCall",
}
```

```
"recipientAccountId": "111122223333"
}
```

DisableKeyRotation

下列範例顯示呼叫 [DisableKeyRotation](#) 操作所產生的 AWS CloudTrail 日誌項目。如需有關自動金鑰輪換的詳細資訊，請參閱 [輪換 AWS KMS keys](#)。

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2022-09-01T19:31:39Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "DisableKeyRotation",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "responseElements": null,
  "requestID": "d6a9351a-ed6e-4581-88d1-2a9a8a538497",
  "eventID": "6313164c-83aa-4cc3-9e1a-b7c426f7a5b1",
  "readOnly": false,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
```

```
"eventCategory": "Management"
}
```

DisconnectCustomKeyStore

下列範例顯示呼叫 [DisconnectCustomKeyStore](#) 操作所產生的 AWS CloudTrail 日誌項目。如需中斷連線自訂金鑰存放區的詳細資訊，請參閱 [中斷連接 AWS CloudHSM 金鑰存放區](#)。

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2021-10-21T20:17:32Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "DisconnectCustomKeyStore",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "customKeyStoreId": "cks-1234567890abcdef0"
  },
  "responseElements": null,
  "additionalEventData": {
    "customKeyStoreName": "ExampleKeyStore",
    "clusterId": "cluster-1a23b4cdefg"
  },
  "requestID": "abcde9e1-f1a3-4460-a423-577fb6e695c9",
  "eventID": "114b61b9-0ea6-47f5-a9d2-4f2bdd0017d5",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333"
}
```

EnableKey

下列範例顯示 [EnableKey](#) 操作的 AWS CloudTrail 日誌項目。如需在 AWS KMS keys 中啟用和停用的資訊 AWS KMS，請參閱 [啟用和停用金鑰](#)。

在 2022 年 12 月或之後記錄的此操作的 CloudTrail 日誌項目會在 `responseElements.keyId` 值中包含受影響 KMS 金鑰的金鑰 ARN，即使此操作不會傳回金鑰 ARN。

```
{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2014-11-04T00:52:20Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "EnableKey",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "responseElements": {
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "requestID": "d528a6fb-63bc-11e4-bc2b-4198b6150d5c",
  "eventID": "be393928-3629-4370-9634-567f9274d52e",
  "readOnly": false,
  "resources": [{
    "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "accountId": "111122223333"
  }],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

EnableKeyRotation

下列範例顯示呼叫 [EnableKeyRotation](#) 操作的 AWS CloudTrail 日誌項目。如需輪換金鑰時所寫入的 CloudTrail 日誌項目範例，請參閱 [RotateKey](#)。如需關於輪換 AWS KMS keys 的資訊，請參閱 [輪換 AWS KMS keys](#)。

Note

[rotation-period](#) 是選用的請求參數。如果您在啟用自動金鑰輪換時未指定輪換期間，預設值為 365 天。

在 2022 年 12 月或之後記錄的此操作的 CloudTrail 日誌項目會在 `responseElements.keyId` 值中包含受影響 KMS 金鑰的金鑰 ARN，即使此操作不會傳回金鑰 ARN。

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2020-07-25T23:41:56Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "EnableKeyRotation",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "rotationPeriodInDays": 180
  },
  "responseElements": {
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "requestID": "81f5b794-452b-4d6a-932b-68c188165273",
  "eventID": "fefc43a7-8e06-419f-bcab-b3bf18d6a401",
  "readOnly": false,
```

```
"resources": [
  {
    "accountId": "111122223333",
    "type": "AWS::KMS::Key",
    "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  }
],
"eventType": "AwsApiCall",
"recipientAccountId": "111122223333"
}
```

加密

下列範例顯示 [Encrypt](#) 操作的 AWS CloudTrail 日誌項目。

```
{
  "eventVersion": "1.11",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2025-05-20T20:46:16Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "Encrypt",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "encryptionContext": {
      "Department": "Engineering"
    },
    "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
  },
  "responseElements": null,
  "additionalEventData": {
```

```

    "keyMaterialId":
      "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
    },
    "requestID": "f3423043-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "91235988-eb87-476a-ac2c-0cdc244e6dca",
    "readOnly": true,
    "resources": [{
      "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "accountId": "111122223333"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
    "eventCategory": "Management",
    "tlsDetails": {
      "tlsVersion": "TLSv1.3",
      "cipherSuite": "TLS_AES_256_GCM_SHA384",
      "clientProvidedHostHeader": "kms.us-east-1.amazonaws.com"
    }
  }
}

```

GenerateDataKey

下列範例顯示 [GenerateDataKey](#) 操作的 AWS CloudTrail 日誌項目。

```

{
  "eventVersion": "1.11",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2025-05-20T20:46:16Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GenerateDataKey",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {

```

```

    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "keySpec": "AES_256",
    "encryptionContext": {
      "Department": "Engineering",
      "Project": "Alpha"
    }
  },
  "responseElements": null,
  "additionalEventData": {
    "keyMaterialId":
"123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
  },
  "requestID": "e0eb83e3-63bc-11e4-bc2b-4198b6150d5c",
  "eventID": "a9dea4f9-8395-46c0-942c-f509c02c2b71",
  "readOnly": true,
  "resources": [{
    "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "accountId": "111122223333"
  }],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management",
  "tlsDetails": {
    "tlsVersion": "TLSv1.3",
    "cipherSuite": "TLS_AES_256_GCM_SHA384",
    "clientProvidedHostHeader": "kms.us-east-1.amazonaws.com"
  }
}

```

GenerateDataKeyPair

下列範例顯示 [GenerateDataKeyPair](#) 操作的 AWS CloudTrail 日誌項目。此範例會記錄產生 RSA 金鑰對的操作，該 RSA 金鑰對採用對稱加密 AWS KMS key 進行加密。

```

{
  "eventVersion": "1.11",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",

```



```

        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2025-05-20T20:46:16Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyPair",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyPairSpec": "RSA_3072",
        "encryptionContext": {
            "Project": "Alpha"
        },
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": null,
    "additionalEventData": {
        "keyMaterialId":
"123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
    },
    "requestID": "52fb127b-0fe5-42bb-8e5e-f560febde6b0",
    "eventID": "9b6bd6d2-529d-4890-a949-593b13800ad7",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    }
}

```

GenerateDataKeyPairWithoutPlaintext

下列範例顯示 [GenerateDataKeyPairWithoutPlaintext](#) 操作的 AWS CloudTrail 日誌項目。此範例會記錄產生 RSA 金鑰對的操作，該 RSA 金鑰對採用對稱加密 AWS KMS key 進行加密。

```
{
  "eventVersion": "1.11",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2025-05-20T20:46:16Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GenerateDataKeyPairWithoutPlaintext",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyPairSpec": "RSA_4096",
    "encryptionContext": {
      "Index": "5"
    }
  },
  "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
},
"responseElements": null,
"additionalEventData": {
  "keyMaterialId":
  "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
},
"requestID": "52fb127b-0fe5-42bb-8e5e-f560febde6b0",
"eventID": "9b6bd6d2-529d-4890-a949-593b13800ad7",
"readOnly": true,
"resources": [
  {
    "accountId": "111122223333",
    "type": "AWS::KMS::Key",
    "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  }
]
```

```
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
    "eventCategory": "Management",
    "tlsDetails": {
      "tlsVersion": "TLSv1.3",
      "cipherSuite": "TLS_AES_256_GCM_SHA384",
      "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    }
  }
}
```

GenerateDataKeyWithoutPlaintext

下列範例顯示 [GenerateDataKeyWithoutPlaintext](#) 操作的 AWS CloudTrail 日誌項目。

```
{
  "eventVersion": "1.11",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2025-05-20T20:46:16Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GenerateDataKeyWithoutPlaintext",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "keySpec": "AES_256",
    "encryptionContext": {
      "Project": "Alpha"
    }
  },
  "responseElements": null,
  "additionalEventData": {
    "keyMaterialId":
      "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
  }
}
```

```

    },
    "requestID": "d6b8e411-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "f7734272-9ec5-4c80-9f36-528ebbe35e4a",
    "readOnly": true,
    "resources": [{
        "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "accountId": "111122223333"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
  },
  "eventCategory": "Management",
  "tlsDetails": {
    "tlsVersion": "TLSv1.3",
    "cipherSuite": "TLS_AES_256_GCM_SHA384",
    "clientProvidedHostHeader": "kms.us-east-1.amazonaws.com"
  }
}

```

GenerateMac

下列範例顯示 [GenerateMac](#) 操作的 AWS CloudTrail 日誌項目。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2022-12-23T19:26:54Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GenerateMac",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "macAlgorithm": "HMAC_SHA_512",
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
  }
}

```

```

    },
    "responseElements": null,
    "requestID": "e0eb83e3-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "a9dea4f9-8395-46c0-942c-f509c02c2b71",
    "readOnly": true,
    "resources": [
      {
        "accountId": "111122223333",
        "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
  }
}

```

GenerateRandom

下列範例顯示 [GenerateRandom](#) 操作的 AWS CloudTrail 日誌項目。由於此操作不使用 AWS KMS key，因此 resources 欄位為空白。

```

{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2014-11-04T00:52:37Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GenerateRandom",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": null,
  "responseElements": null,
  "requestID": "df1e3de6-63bc-11e4-bc2b-4198b6150d5c",
  "eventID": "239cb9f7-ae05-4c94-9221-6ea30eef0442",
  "readOnly": true,

```

```
"resources": [],
"eventType": "AwsApiCall",
"recipientAccountId": "111122223333"
}
```

GetKeyPolicy

下列範例顯示 [GetKeyPolicy](#) 操作的 AWS CloudTrail 日誌項目。如需檢視 KMS 金鑰之金鑰政策的詳細資訊，請參閱 [檢視金鑰政策](#)。

```
{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2014-11-04T00:50:30Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GetKeyPolicy",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "policyName": "default"
  },
  "responseElements": null,
  "requestID": "93746dd6-63bc-11e4-bc2b-4198b6150d5c",
  "eventID": "4aa7e4d5-d047-452a-a5a6-2cce282a7e82",
  "readOnly": true,
  "resources": [{
    "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "accountId": "111122223333"
  }],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

GetKeyRotationStatus

下列範例顯示 [GetKeyRotationStatus](#) 操作的 AWS CloudTrail 日誌項目。如需 KMS 金鑰金鑰金鑰材料自動輪換和隨需輪換的相關資訊，請參閱 [輪換 AWS KMS keys](#)。

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2024-02-20T19:16:45Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GetKeyRotationStatus",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "responseElements": null,
  "requestID": "12f9b7e8-49b9-4c1c-a7e3-34ac0cdf0467",
  "eventID": "3d082126-9e7d-4167-8372-a6cfcbed4be6",
  "readOnly": true,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES256-GCM-SHA384",
```

```

    "clientProvidedHostHeader": "kms.us-east-1.amazonaws.com"
  }
}

```

GetParametersForImport

下列範例顯示使用 [GetParametersForImport](#) 操作時產生的 AWS CloudTrail 日誌項目。此操作會傳回您在將金鑰材料匯入 KMS 金鑰時使用的公有金鑰和匯入字符。當您使用 [GetParametersForImport](#) 操作或使用 AWS KMS 主控台 [下載公有金鑰和匯入字符](#) 時，會記錄相同的 CloudTrail 項目。

```

{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2020-07-25T23:58:23Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GetParametersForImport",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "wrappingAlgorithm": "RSAES_OAEP_SHA_256",
    "wrappingKeySpec": "RSA_2048"
  },
  "responseElements": null,
  "requestID": "b5786406-e3c7-43d6-8d3c-6d5ef96e2278",
  "eventID": "4023e622-0c3e-4324-bdef-7f58193bba87",
  "readOnly": true,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ]
}

```



```
],  
  "eventType": "AwsApiCall",  
  "recipientAccountId": "111122223333"  
}
```

ImportKeyMaterial

下列範例顯示使用 [ImportKeyMaterial](#) 操作時產生的 AWS CloudTrail 日誌項目。當您使用 ImportKeyMaterial 操作或使用 AWS KMS 主控台將 [金鑰材料匯入](#) 時，會記錄相同的 CloudTrail 項目 AWS KMS key。

在 2022 年 12 月或之後記錄的此操作的 CloudTrail 日誌項目會在 responseElements.keyId 值中包含受影響 KMS 金鑰的金鑰 ARN，即使此操作不會傳回金鑰 ARN。

```
{  
  "eventVersion": "1.11",  
  "userIdentity": {  
    "type": "IAMUser",  
    "principalId": "EX_PRINCIPAL_ID",  
    "arn": "arn:aws:iam::111122223333:user/Alice",  
    "accountId": "111122223333",  
    "accessKeyId": "EXAMPLE_KEY_ID",  
    "userName": "Alice"  
  },  
  "eventTime": "2025-05-21T05:42:31Z",  
  "eventSource": "kms.amazonaws.com",  
  "eventName": "ImportKeyMaterial",  
  "awsRegion": "us-west-2",  
  "sourceIPAddress": "192.0.2.0",  
  "userAgent": "AWS Internal",  
  "requestParameters": {  
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",  
    "validTo": "May 21, 2025, 5:47:45 AM",  
    "expirationModel": "KEY_MATERIAL_EXPIRES",  
    "importType": "NEW_KEY_MATERIAL",  
    "keyMaterialDescription": "ExampleKeyMaterialA"  
  },  
  "responseElements": {  
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",  
    "keyMaterialId":  
      "123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"  
  },  
}
```

```

    "requestID": "89e10ee7-a612-414d-95a2-a128346969fd",
    "eventID": "c7abd205-a5a2-4430-bbfa-fc10f3e2d79f",
    "readOnly": false,
    "resources": [
      {
        "accountId": "111122223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333"
  },
  "eventCategory": "Management",
  "tlsDetails": {
    "tlsVersion": "TLSv1.3",
    "cipherSuite": "TLS_AES_256_GCM_SHA384",
    "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
  }
}

```

ListAliases

下列範例顯示 [ListAliases](#) 操作的 AWS CloudTrail 日誌項目。由於此操作不會使用任何特定別名或 AWS KMS key，因此 resources 欄位為空白。如需在 中檢視別名的詳細資訊 AWS KMS，請參閱 [尋找 KMS 金鑰的別名名稱和別名 ARN](#)。

```

{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2014-11-04T00:51:45Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "ListAliases",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",

```

[illegible]

ListGrants

下列範例顯示 [ListGrant](#) 操作的 AWS CloudTrail 日誌項目。如需 中授予的相關資訊 AWS KMS，請參閱 [在中授予 AWS KMS](#)。

```
{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2014-11-04T00:52:49Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "ListGrants",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "marker":
      "eyJncmFudElkJoiMwY4M2U2ZmM0YTlYNDgxYjQ2Yzc4MTdhM2Y4YmQwMDFkZDNiYmQ1MGVlYTMxY2RmOWFiNWY1Nzc1NA\u003d\u003d",
    "limit": 10
  },
}
```

```

    "responseElements": null,
    "requestID": "e5c23960-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "d24380f5-1b20-4253-8e92-dd0492b3bd3d",
    "readOnly": true,
    "resources": [{
        "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "accountId": "111122223333"
    }],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}

```

ListKeyRotations

下列範例顯示 [ListKeyRotations](#) 操作的 AWS CloudTrail 日誌項目。如需 KMS 金鑰金鑰金鑰材料自動輪換和隨需輪換的相關資訊，請參閱 [輪換 AWS KMS keys](#)。

```

{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2025-05-21T05:42:35Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "ListKeyRotations",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "includeKeyMaterial": "ALL_KEY_MATERIAL"
    },
    "responseElements": null,
    "requestID": "99c88d32-f2db-455e-8a9a-23855258a452",
    "eventID": "8ce0e74b-b9c7-45a2-96ef-83136d38068e",
    "readOnly": true,
    "resources": [

```

```

    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
      "tlsVersion": "TLSv1.2",
      "cipherSuite": "ECDHE-RSA-AES256-GCM-SHA384",
      "clientProvidedHostHeader": "kms.us-east-1.amazonaws.com"
    }
  }
}

```

PutKeyPolicy

下列範例顯示呼叫 [PutKeyPolicy](#) 操作所產生的 AWS CloudTrail 日誌項目。如需有關更新金鑰政策的資訊，請參閱 [變更金鑰政策](#)。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2022-09-01T20:06:16Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "PutKeyPolicy",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "policyName": "default",

```

```

    "policy": "{\n  \"Version\" : \"2012-10-17\",\n  \"Id\" : \"key-default-1\",\n  \"Statement\" : [ {\n    \"Sid\" : \"Enable IAM User Permissions\",\n    \"Effect\" : \"Allow\",\n    \"Principal\" : {\n      \"AWS\" : \"arn:aws:iam::111122223333:root\"\n    },\n    \"Action\" : \"kms:*\",\n    \"Resource\" : \"*\"\n  } ]\n}",
    "bypassPolicyLockoutSafetyCheck": false
  },
  "responseElements": null,
  "requestID": "7bb906fa-dc21-4350-b65c-808ff0f72f55",
  "eventID": "c217db1f-903f-4a2f-8f88-9580182d6313",
  "readOnly": false,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management"
}

```

ReEncrypt

下列範例顯示 [ReEncrypt](#) 操作的 AWS CloudTrail 日誌項目。此日誌項目中的 `resources` 欄位會依該順序指定兩個 AWS KMS keys 來源 KMS 金鑰和目的地 KMS 金鑰。

```

{
  "eventVersion": "1.11",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2025-05-22T19:34:55Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "ReEncrypt",
  "awsRegion": "us-west-2",

```

```

"sourceIpAddress": "192.0.2.0",
"userAgent": "AWS Internal",
"requestParameters": {
  "sourceEncryptionAlgorithm": "SYMMETRIC_DEFAULT",
  "sourceEncryptionContext": {
    "Project": "Alpha",
    "Department": "Engineering"
  },
  "destinationKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
  "destinationEncryptionAlgorithm": "SYMMETRIC_DEFAULT",
  "destinationEncryptionContext": {
    "Level": "3A"
  }
},
"responseElements": null,
"additionalEventData": {
  "destinationKeyMaterialId":
"96083e4fb6dbc41d77578a213a6b6669c044dd4c143e96755396d2bf11fd6068",
  "sourceKeyMaterialId":
"123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
},
"requestID": "03769fd4-acf9-4b33-adf3-2ab8ca73aadf",
"eventID": "542d9e04-0e8d-4e05-bf4b-4bdeb032e6ec",
"readOnly": true,
"resources": [
  {
    "accountId": "111122223333",
    "type": "AWS::KMS::Key",
    "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  {
    "accountId": "111122223333",
    "type": "AWS::KMS::Key",
    "ARN": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321"
  }
],
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "111122223333"
"eventCategory": "Management",
"tlsDetails": {
  "tlsVersion": "TLSv1.3",

```

```

    "cipherSuite": "TLS_AES_256_GCM_SHA384",
    "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
  }
}

```

ReplicateKey

下列範例顯示呼叫 [ReplicateKey](#) 操作所產生的 AWS CloudTrail 日誌項目。ReplicateKey 請求會產生 ReplicateKey 操作和 [CreateKey](#) 操作。

如需複寫多區域金鑰的相關資訊，請參閱 [建立多區域複本金鑰](#)。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2020-11-18T01:29:18Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "ReplicateKey",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "replicaRegion": "us-west-2",
    "bypassPolicyLockoutSafetyCheck": false,
    "description": ""
  },
  "responseElements": {
    "replicaKeyMetadata": {
      "awsAccountId": "111122223333",
      "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
      "arn": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "creationDate": "Nov 18, 2020, 1:29:18 AM",
      "enabled": false,
      "description": "",

```



```

    "keyUsage": "ENCRYPT_DECRYPT",
    "keyState": "Creating",
    "origin": "AWS_KMS",
    "keyManager": "CUSTOMER",
    "keySpec": "SYMMETRIC_DEFAULT",
    "customerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "encryptionAlgorithms": [
        "SYMMETRIC_DEFAULT"
    ],
    "multiRegion": true,
    "multiRegionConfiguration": {
        "multiRegionKeyType": "REPLICA",
        "primaryKey": {
            "arn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "region": "us-east-1"
        },
        "replicaKeys": [
            {
                "arn": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
                "region": "us-west-2"
            }
        ]
    },
    "replicaPolicy": "{\n  \"Version\": \"2012-10-17\", \n  \"Statement\": [\n    {\n      \"Effect\": \"Allow\", \n      \"Principal\": {\n        \"AWS\": \"arn:aws:iam::123456789012:user/Alice\"\n      }, \n      \"Action\": \"kms:*\", \n      \"Resource\": \"*\"\n    }, {\n      \"Effect\": \"Allow\", \n      \"Principal\": {\n        \"AWS\": \"arn:aws:iam::012345678901:user/Bob\"\n      }, \n      \"Action\": \"kms:CreateGrant\", \n      \"Resource\": \"*\"\n    }, {\n      \"Effect\": \"Allow\", \n      \"Principal\": {\n        \"AWS\": \"arn:aws:iam::012345678901:user/Charlie\"\n      }, \n      \"Action\": \"kms:Encrypt\", \n      \"Resource\": \"*\"\n    }]\n  }",
    "requestID": "abcdef68-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "fedcba44-6773-4f96-8763-1993aec9ae6a",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ]
}

```

```
],
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "111122223333",
"eventCategory": "Management"
}
```

RetireGrant

下列範例顯示呼叫 [RetireGrant](#) 操作所產生的 AWS CloudTrail 日誌項目。如需有關淘汰授予的詳細資訊，請參閱 [淘汰和撤銷授予](#)。

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2022-09-01T19:39:33Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "RetireGrant",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": null,
  "responseElements": null,
  "additionalEventData": {
    "grantId": "abcde1237f76e4ba7987489ac329fbfba6ad343d6f7075dbd1ef191f0120514a"
  },
  "requestID": "1d274d57-5697-462c-a004-f25fcc29fa26",
  "eventID": "0771bcfb-3e24-4332-9ac8-e1c06563eecf",
  "readOnly": false,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ]
}
```

```

],
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "111122223333",
"eventCategory": "Management"
}

```

RevokeGrant

下列範例顯示呼叫 [RevokeGrant](#) 操作所產生的 AWS CloudTrail 日誌項目。如需有關撤銷授予的詳細資訊，請參閱 [淘汰和撤銷授予](#)。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2022-09-01T19:35:17Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "RevokeGrant",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "grantId": "abcde1237f76e4ba7987489ac329fbfba6ad343d6f7075dbd1ef191f0120514a"
  },
  "responseElements": null,
  "requestID": "59d94c03-c5b7-428d-ae6e-f2c4b47d2917",
  "eventID": "07a23a39-6526-4ae2-b31e-d35fbe9e24ee",
  "readOnly": false,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ]
}

```

```
],  
  "eventType": "AwsApiCall",  
  "managementEvent": true,  
  "recipientAccountId": "111122223333",  
  "eventCategory": "Management"  
}
```

RotateKey

這些範例顯示輪換之操作的 AWS CloudTrail 日誌項目 AWS KMS keys。如需輪換 KMS 金鑰的詳細資訊，請參閱 [輪換 AWS KMS keys](#)。

下列範例顯示 操作的 CloudTrail 日誌項目，該操作會輪換啟用自動金鑰輪換的對稱加密 KMS 金鑰。如需啟用自動輪換的資訊，請參閱 [輪換 AWS KMS keys](#)。

如需記錄 EnableKeyRotation 操作的 CloudTrail 日誌項目範例，請參閱 [EnableKeyRotation](#)。

```
{  
  "eventVersion": "1.11",  
  "userIdentity": {  
    "accountId": "111122223333",  
    "invokedBy": "AWS Internal"  
  },  
  "eventTime": "2025-05-20T20:44:37Z",  
  "eventSource": "kms.amazonaws.com",  
  "eventName": "RotateKey",  
  "awsRegion": "us-west-2",  
  "sourceIPAddress": "AWS Internal",  
  "userAgent": "AWS Internal",  
  "requestParameters": null,  
  "responseElements": null,  
  "eventID": "a24b3967-ddad-417f-9b22-2332b918db06",  
  "readOnly": false,  
  "resources": [  
    {  
      "accountId": "111122223333",  
      "type": "AWS::KMS::Key",  
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"  
    }  
  ],  
  "eventType": "AwsServiceEvent",  
  "managementEvent": true,  
}
```

```

    "recipientAccountId": "111122223333",
    "serviceEventDetails": {
      "rotationType": "AUTOMATIC",
      "keyOrigin": "AWS_KMS",
      "previousKeyMaterialId":
"123456789abcdef0123456789abcdef0123456789abcdef0",
      "currentKeyMaterialId":
"96083e4fb6dbc41d77578a213a6b6669c044dd4c143e96755396d2bf11fd6068",
      "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "eventCategory": "Management"
  }
}

```

下列範例顯示由 [RotateKeyOnDemand](#) 操作啟動的隨需輪換的 CloudTrail 日誌項目。如需隨需輪換對稱加密 KMS 金鑰的詳細資訊，請參閱 [執行隨需金鑰輪換](#)。

如需記錄 RotateKeyOnDemand 操作的 CloudTrail 日誌項目範例，請參閱 [RotateKeyOnDemand](#)。

```

{
  "eventVersion": "1.11",
  "userIdentity": {
    "accountId": "111122223333",
    "invokedBy": "AWS Internal"
  },
  "eventTime": "2025-05-20T20:44:37Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "RotateKey",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "AWS Internal",
  "userAgent": "AWS Internal",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "a24b3967-ddad-417f-9b22-2332b918db06",
  "readOnly": false,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ],
  "eventType": "AwsServiceEvent",

```

```

    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "serviceEventDetails": {
      "rotationType": "ON_DEMAND",
      "keyOrigin": "EXTERNAL",
      "previousKeyMaterialId":
"123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0",
      "currentKeyMaterialId":
"96083e4fb6dbc41d77578a213a6b6669c044dd4c143e96755396d2bf11fd6068",
      "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "eventCategory": "Management"
  }
}

```

RotateKeyOnDemand

下列範例顯示 [RotateKeyOnDemand](#) 操作的 AWS CloudTrail 日誌項目。如需輪換金鑰時所寫入的 CloudTrail 日誌項目範例，請參閱 [RotateKey](#)。如需 KMS 金鑰金鑰金鑰材料隨需輪換的詳細資訊，請參閱 [執行隨需金鑰輪換](#)。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2024-02-20T17:41:57Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "RotateKeyOnDemand",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "responseElements": {
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "requestID": "9e1dee86-eb84-42fd-8f25-e3fc7dbb32c8",
}

```

```

    "eventID": "00a09fbc-20d6-4a58-9b92-7da85984ab77",
    "readOnly": false,
    "resources": [
      {
        "accountId": "111122223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
      "tlsVersion": "TLSv1.2",
      "cipherSuite": "ECDHE-RSA-AES256-GCM-SHA384",
      "clientProvidedHostHeader": "kms.us-east-1.amazonaws.com"
    }
  }
}

```

ScheduleKeyDeletion

這些範例顯示 [ScheduleKeyDeletion](#) 操作的 AWS CloudTrail 日誌項目。

如需刪除金鑰時所寫入的 CloudTrail 日誌項目範例，請參閱 [DeleteKey](#)。如需刪除 AWS KMS keys 的資訊，請參閱 [刪除 AWS KMS key](#)。

以下範例會記錄單一區域 KMS 金鑰的 ScheduleKeyDeletion 請求。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2021-03-23T18:58:30Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "ScheduleKeyDeletion",

```

```

    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
      "pendingWindowInDays": 20,
      "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": {
      "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "keyState": "PendingDeletion",
      "deletionDate": "Apr 12, 2021 18:58:30 PM"
    },
    "requestID": "ee408f36-ea01-422b-ac14-b0f147c68334",
    "eventID": "3c4226b0-1e81-48a8-a333-7fa5f3cbd118",
    "readOnly": false,
    "resources": [
      {
        "accountId": "111122223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
  }
}

```

以下範例會記錄具有複本金鑰之多區域 KMS 金鑰的 ScheduleKeyDeletion 請求。

在刪除其所有複本金鑰之前，AWS KMS 不會刪除多區域金鑰，因此在 responseElements 欄位中，keyState 為 PendingReplicaDeletion，而 deletionDate 欄位為 省略。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },

```



```

    "eventTime": "2021-10-28T17:59:05Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "ScheduleKeyDeletion",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "pendingWindowInDays": 30,
        "keyId": "mrk-1234abcd12ab34cd56ef1234567890ab"
    },
    "responseElements": {
        "keyId": "arn:aws:kms:us-west-2:111122223333:key/mrk-1234abcd12ab34cd56ef1234567890ab",
        "keyState": "PendingReplicaDeletion",
        "pendingWindowInDays": 30
    },
    "requestID": "12341411-d846-42a6-a476-b1cbe3011f89",
    "eventID": "abcda5f-396d-494c-9380-0c47860df5f1",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/mrk-1234abcd12ab34cd56ef1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}

```

下列範例會記錄 AWS CloudHSM [自訂金鑰存放區中 KMS 金鑰](#) 的 ScheduleKeyDeletion 請求。

```

{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",

```

```

        "userName": "Alice"
    },
    "eventTime": "2021-10-26T23:25:25Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "ScheduleKeyDeletion",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321",
        "pendingWindowInDays": 30
    },
    "responseElements": {
        "keyId": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321",
        "deletionDate": "Nov 2, 2021, 11:25:25 PM",
        "keyState": "PendingDeletion",
        "pendingWindowInDays": 30
    },
    "additionalEventData": {
        "customKeyStoreId": "cks-1234567890abcdef0",
        "clusterId": "cluster-1a23b4cdefg",
        "backingKeys": "[{\\"backingKeyId\\":\\"backing-key-id\\"}]"
    },
    "requestID": "abcd9f60-2c9c-4a0b-a456-d5d998f7f321",
    "eventID": "ca01996a-01b0-4edd-bbbb-25d7b6d1a6fa",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}

```

符號

這些範例顯示 [Sign](#) 操作的 AWS CloudTrail 日誌項目。

以下範例顯示 [Sign](#) 操作的 CloudTrail 日誌項目，該操作使用非對稱 RSA KMS 金鑰產生檔案的數位簽章。

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2022-03-07T22:36:44Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "Sign",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "messageType": "RAW",
    "keyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
    "signingAlgorithm": "RSASSA_PKCS1_V1_5_SHA_256"
  },
  "responseElements": null,
  "requestID": "8d0b35e0-46cf-48b9-be99-bf2ebc9ab9fb",
  "eventID": "107b3cac-b125-4556-9702-12a2b9afc7f7",
  "readOnly": true,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
```

```
"eventCategory": "Management"
}
```

SynchronizeMultiRegionKey

下列範例顯示 AWS KMS 同步[多區域金鑰](#)時產生的 AWS CloudTrail 日誌項目。同步涉及跨區域呼叫，將多區域主要金鑰的[共用屬性](#)複製到其複本金鑰。會定期 AWS KMS 同步多區域金鑰，以確保所有相關多區域金鑰具有相同的金鑰材料。

CloudTrail 日誌項目的 `resources` 元素包含多區域主要金鑰的金鑰 ARN，包括其 AWS 區域。相關的多區域複本金鑰及其區域不會列在此日誌項目中。

在 2022 年 12 月或之後記錄的此操作的 CloudTrail 日誌項目會在 `responseElements.keyId` 值中包含受影響 KMS 金鑰的金鑰 ARN，即使此操作不會傳回金鑰 ARN。

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "accountId": "111122223333",
    "invokedBy": "AWS Internal"
  },
  "eventTime": "2020-11-18T02:04:37Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "SynchronizeMultiRegionKey",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "AWS Internal",
  "userAgent": "AWS Internal",
  "requestParameters": null,
  "responseElements": {
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "requestID": "12345681-de97-42e9-bed0-b02ae1abd8dc",
  "eventID": "abcdec99-2b5c-4670-9521-ddb8f031e146",
  "readOnly": false,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ],
}
```

```
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "111122223333",
"eventCategory": "Management"
}
```

TagResource

下列範例顯示呼叫 [TagResource](#) 操作的 AWS CloudTrail 日誌項目，以新增標籤索引鍵為 Department 且標籤值為 IT 的標籤。

對於輪換金鑰時，寫入 UntagResource CloudTrail 日誌項目的範例，請參閱 [UntagResource](#)。如需標記的相關資訊 AWS KMS keys，請參閱 [中的標籤 AWS KMS](#)。

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2020-07-01T21:19:25Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "TagResource",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "tags": [
      {
        "tagKey": "Department",
        "tagValue": "IT"
      }
    ]
  },
  "responseElements": null,
  "requestID": "b942584a-f77d-4787-9feb-b9c5be6e746d",
  "eventID": "0a091b9b-0df5-4cf9-b667-6f2879532b8f",
}
```

```

    "readOnly": false,
    "resources": [
      {
        "accountId": "111122223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
  }
}

```

UntagResource

下列範例顯示呼叫 [UntagResource](#) 操作的 AWS CloudTrail 日誌項目，以刪除標籤索引鍵為 的標籤Dept。

在 2022 年 12 月或之後記錄的此操作的 CloudTrail 日誌項目會在 `responseElements.keyId` 值中包含受影響 KMS 金鑰的金鑰 ARN，即使此操作不會傳回金鑰 ARN。

對於 TagResource CloudTrail 日誌項目的範例，請參閱 [TagResource](#)。如需標記 AWS KMS keys 的詳細資訊，請參閱 [中的標籤 AWS KMS](#)。

```

{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2020-07-01T21:19:19Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "UntagResource",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",

```

```

        "tagKeys": [
            "Dept"
        ],
        "responseElements": {
            "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        },
        "requestID": "cb1d507b-6015-47f4-812b-179713af8068",
        "eventID": "0b00f4b0-036e-411d-aa75-87eb4a35a4b3",
        "readOnly": false,
        "resources": [
            {
                "accountId": "111122223333",
                "type": "AWS::KMS::Key",
                "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
            }
        ],
        "eventType": "AwsApiCall",
        "recipientAccountId": "111122223333"
    }
}

```

UpdateAlias

下列範例顯示 [UpdateAlias](#) 操作的 AWS CloudTrail 日誌項目。resources 元素包含別名和 KMS 金鑰資源的欄位。如需在 中建立別名的詳細資訊 AWS KMS，請參閱 [建立別名](#)。

在 2022 年 12 月或之後記錄的此操作的 CloudTrail 日誌項目會在 responseElements.keyId 值中包含受影響 KMS 金鑰的金鑰 ARN，即使此操作不會傳回金鑰 ARN。

```

{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2020-11-13T23:18:15Z",
    "eventSource": "kms.amazonaws.com",

```

```

    "eventName": "UpdateAlias",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "aliasName": "alias/my_alias",
        "targetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": {
        "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "d9472f40-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "f72d3993-864f-48d6-8f16-e26e1ae8dff0",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:alias/my_alias"
        },
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}

```

UpdateCustomKeyStore

下列範例顯示呼叫 [UpdateCustomKeyStore](#) 操作以更新自訂金鑰存放區的叢集 ID 所產生的 AWS CloudTrail 日誌項目。如需編輯自訂金鑰存放區的詳細資訊，請參閱 [編輯 AWS CloudHSM 金鑰存放區設定](#)。

```

{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",

```



```

    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2021-10-21T20:17:32Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "UpdateCustomKeyStore",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "customKeyId": "cks-1234567890abcdef0",
    "clusterId": "cluster-1a23b4cdefg"
  },
  "responseElements": null,
  "additionalEventData": {
    "customKeyName": "ExampleKeyStore",
    "clusterId": "cluster-1a23b4cdefg"
  },
  "requestID": "abcde9e1-f1a3-4460-a423-577fb6e695c9",
  "eventID": "114b61b9-0ea6-47f5-a9d2-4f2bdd0017d5",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333"
}

```

UpdateKeyDescription

下列範例顯示呼叫 [UpdateKeyDescription](#) 操作所產生的 AWS CloudTrail 日誌項目。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2022-09-01T19:22:40Z",

```

```

    "eventSource": "kms.amazonaws.com",
    "eventName": "UpdateKeyDescription",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
        "description": "New key description"
    },
    "responseElements": null,
    "requestID": "8c3c1f8b-336d-4896-b034-4eb9916bc9b3",
    "eventID": "f5f3d548-2e9e-4658-8427-9dcb5b1ea791",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}

```

UpdatePrimaryRegion

下列範例顯示呼叫[多區域金鑰](#)上的 [UpdatePrimaryRegion](#) 操作所產生的 AWS CloudTrail 日誌項目。

UpdatePrimaryRegion 操作會寫入兩條 CloudTrail 日誌項目：一個在區域中具有轉換為複本金鑰的多區域主要金鑰，另一個在區域中具有轉換為主要金鑰的多區域複本金鑰。

在 2022 年 12 月或之後記錄的此操作的 CloudTrail 日誌項目會在 `responseElements.keyId` 值中包含受影響 KMS 金鑰的金鑰 ARN，即使此操作不會傳回金鑰 ARN。

以下範例顯示區域中 UpdatePrimaryRegion CloudTrail 日誌項目，其中多區域金鑰從主要金鑰變更為複本金鑰 (us-west-2)。primaryRegion 欄位顯示現在託管主要金鑰的區域 (ap-northeast-1)。

```

{
    "eventVersion": "1.08",
    "userIdentity": {

```

```

        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2021-03-10T20:23:37Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "UpdatePrimaryRegion",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyId": "mrk-1234abcd12ab34cd56ef1234567890ab",
        "primaryRegion": "ap-northeast-1"
    },
    "responseElements": {
        "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "ee408f36-ea01-422b-ac14-b0f147c68334",
    "eventID": "3c4226b0-1e81-48a8-a333-7fa5f3cbd118",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/mrk-1234abcd12ab34cd56ef1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "111122223333"
}

```

以下範例顯示區域中 UpdatePrimaryRegion CloudTrail 日誌項目，其中多區域金鑰從複本金鑰變更為主要金鑰 (ap-northeast-1)。此日誌項目無法識別先前的主要區域。

```

{
    "eventVersion": "1.08",

```

```

    "userIdentity": {
      "type": "IAMUser",
      "principalId": "EX_PRINCIPAL_ID",
      "arn": "arn:aws:iam::111122223333:user/Alice",
      "accountId": "111122223333",
      "accessKeyId": "EXAMPLE_KEY_ID",
      "userName": "Alice",
      "invokedBy": "kms.amazonaws.com"
    },
    "eventTime": "2021-03-10T20:23:37Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "UpdatePrimaryRegion",
    "awsRegion": "ap-northeast-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
      "keyId": "arn:aws:kms:ap-northeast-1:111122223333:key/mrk-1234abcd12ab34cd56ef1234567890ab",
      "primaryRegion": "ap-northeast-1"
    },
    "responseElements": {
      "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "requestID": "ee408f36-ea01-422b-ac14-b0f147c68334",
    "eventID": "091e6be5-737f-43c6-8431-e3679d6d0619",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "111122223333"
  }
}

```

VerifyMac

下列範例顯示 [VerifyMac](#) 操作的 AWS CloudTrail 日誌項目。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",

```

```

    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2022-03-31T19:25:54Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "VerifyMac",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "macAlgorithm": "HMAC_SHA_384",
    "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "responseElements": null,
  "requestID": "f35da560-edff-4d6e-9b40-fb306fa9ef1e",
  "eventID": "6b464487-6dea-44cd-84ad-225d7450c975",
  "readOnly": true,
  "resources": [
    {
      "accountId": "111122223333",
      "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333",
  "eventCategory": "Management"
}

```

確認

這些範例顯示[驗證](#)操作的 AWS CloudTrail 日誌項目。

以下範例顯示 [Verify](#) 操作的 CloudTrail 日誌項目，該操作使用非對稱 RSA KMS 金鑰驗證數位簽章。

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",

```

```

    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2022-03-07T22:50:41Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "Verify",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "signingAlgorithm": "RSASSA_PKCS1_V1_5_SHA_256",
    "keyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
    "messageType": "RAW"
  },
  "responseElements": null,
  "requestID": "c73ab82a-af82-4750-ae2c-b6bb790e9c28",
  "eventID": "3b4331cd-5b7b-4de5-bf5f-82ec22f0dac0",
  "readOnly": true,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management"
}

```

Amazon EC2 範例 1

以下範例記錄 IAM 主體在 Amazon EC2 管理主控台中使用預設磁碟區金鑰來建立加密的磁碟區。

以下範例顯示 CloudTrail 日誌項目，其中使用者 Alice 在 Amazon EC2 管理主控台中使用預設磁碟區金鑰來建立加密的磁碟區。EC2 日誌檔記錄包含 volumeId 欄位，其值為 "vol-13439757"。AWS KMS 記錄包含值為 encryptionContext 的欄位 "aws:ebs:id": "vol-13439757"。同樣地，兩個記錄之間的 principalId 和 accountId 相符。記錄反映建立加密的磁碟區會產生用來加密磁碟區內容的資料金鑰。

```
{
```

```
"Records": [  
  {  
    "eventVersion": "1.02",  
    "userIdentity": {  
      "type": "IAMUser",  
      "principalId": "EX_PRINCIPAL_ID",  
      "arn": "arn:aws:iam::111122223333:user/Alice",  
      "accountId": "111122223333",  
      "accessKeyId": "EXAMPLE_KEY_ID",  
      "userName": "Alice"  
    },  
    "eventTime": "2014-11-05T20:50:18Z",  
    "eventSource": "ec2.amazonaws.com",  
    "eventName": "CreateVolume",  
    "awsRegion": "us-east-1",  
    "sourceIPAddress": "192.0.2.0",  
    "userAgent": "AWS Internal",  
    "requestParameters": {  
      "size": "10",  
      "zone": "us-east-1a",  
      "volumeType": "gp2",  
      "encrypted": true  
    },  
    "responseElements": {  
      "volumeId": "vol-13439757",  
      "size": "10",  
      "zone": "us-east-1a",  
      "status": "creating",  
      "createTime": 1415220618876,  
      "volumeType": "gp2",  
      "iops": 30,  
      "encrypted": true  
    },  
    "requestID": "1565210e-73d0-4912-854c-b15ed349e526",  
    "eventID": "a3447186-135f-4b00-8424-bc41f1a93b4f",  
    "eventType": "AwsApiCall",  
    "recipientAccountId": "123456789012"  
  },  
  {  
    "eventVersion": "1.02",  
    "userIdentity": {  
      "type": "IAMUser",  
      "principalId": "EX_PRINCIPAL_ID",  
      "arn": "arn:aws:iam::111122223333:user/Alice",
```

```

    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2014-11-05T20:50:19Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GenerateDataKeyWithoutPlaintext",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "&AWS; Internal",
  "requestParameters": {
    "encryptionContext": {
      "aws:ebs:id": "vol-13439757"
    },
    "numberOfBytes": 64,
    "keyId": "alias/aws/ebs"
  },
  "responseElements": null,
  "requestID": "create-123456789012-758241111-1415220618",
  "eventID": "4bd2a696-d833-48cc-b72c-05e61b608399",
  "readOnly": true,
  "resources": [
    {
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "accountId": "111122223333"
    }
  ],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
]
}

```

Amazon EC2 範例 2

在下列範例中，執行 Amazon EC2 執行個體的 IAM 主體會建立並掛載在 KMS 金鑰下加密的資料磁碟區。此動作會產生多筆 CloudTrail 日誌記錄。如需 Amazon EBS 和加密的詳細資訊，請參閱 [Amazon EBS 加密的需求](#)。

建立磁碟區時，代表客戶的 Amazon EC2 會從 AWS KMS () 取得加密的資料金鑰 `GenerateDataKeyWithoutPlaintext`。然後它建立一個授予權限 (`CreateGrant`)，允許它解密資料密鑰。掛載磁碟區時，Amazon EC2 會呼叫 AWS KMS 來解密資料金鑰 (`Decrypt`)。

Amazon EC2 執行個體的 `instanceId` ("i-81e2f56c") 會顯示在 `RunInstances` 事件中。相同執行個體 ID 會使 `granteePrincipal` 授權限定所建立的授權 ("111122223333:aws:ec2-infrastructure:i-81e2f56c") 和假設角色是 `Decrypt` 呼叫中的委託人 ("arn:aws:sts::111122223333:assumed-role/aws:ec2-infrastructure/i-81e2f56c")。

保護資料磁碟區的 KMS 金鑰金鑰 [ARN](#) `arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab` 會出現在所有三個 AWS KMS 呼叫中 (`CreateGrant`、`GenerateDataKeyWithoutPlaintext` 和 `Decrypt`)。

```
{
  "Records": [
    {
      "eventVersion": "1.02",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
      },
      "eventTime": "2014-11-05T21:35:27Z",
      "eventSource": "ec2.amazonaws.com",
      "eventName": "RunInstances",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "AWS Internal",
      "requestParameters": {
        "instancesSet": {
          "items": [
            {
              "imageId": "ami-b66ed3de",
              "minCount": 1,
              "maxCount": 1
            }
          ]
        },
        "groupSet": {
          "items": [
            {
              "groupId": "sg-98b6e0f2"
            }
          ]
        }
      }
    }
  ]
}
```

```

    }
  ]
},
"instanceType": "m3.medium",
"blockDeviceMapping": {
  "items": [
    {
      "deviceName": "/dev/xvda",
      "ebs": {
        "volumeSize": 8,
        "deleteOnTermination": true,
        "volumeType": "gp2"
      }
    },
    {
      "deviceName": "/dev/sdb",
      "ebs": {
        "volumeSize": 8,
        "deleteOnTermination": false,
        "volumeType": "gp2",
        "encrypted": true
      }
    }
  ]
},
"monitoring": {
  "enabled": false
},
"disableApiTermination": false,
"instanceInitiatedShutdownBehavior": "stop",
"clientToken": "XdKUT141516171819",
"ebsOptimized": false
},
"responseElements": {
  "reservationId": "r-5ebc9f74",
  "ownerId": "111122223333",
  "groupSet": {
    "items": [
      {
        "groupId": "sg-98b6e0f2",
        "groupName": "launch-wizard-2"
      }
    ]
  }
},

```

```
"instancesSet": {
  "items": [
    {
      "instanceId": "i-81e2f56c",
      "imageId": "ami-b66ed3de",
      "instanceState": {
        "code": 0,
        "name": "pending"
      },
      "amiLaunchIndex": 0,
      "productCodes": {

      },
      "instanceType": "m3.medium",
      "launchTime": 1415223328000,
      "placement": {
        "availabilityZone": "us-east-1a",
        "tenancy": "default"
      },
      "monitoring": {
        "state": "disabled"
      },
      "stateReason": {
        "code": "pending",
        "message": "pending"
      },
      "architecture": "x86_64",
      "rootDeviceType": "ebs",
      "rootDeviceName": "/dev/xvda",
      "blockDeviceMapping": {

      },
      "virtualizationType": "hvm",
      "hypervisor": "xen",
      "clientToken": "XdKUT1415223327917",
      "groupSet": {
        "items": [
          {
            "groupId": "sg-98b6e0f2",
            "groupName": "launch-wizard-2"
          }
        ]
      },
      "networkInterfaceSet": {
```

```

        },
        "ebsOptimized": false
    }
]
}
},
"requestID": "41c4b4f7-8bce-4773-bf0e-5ae3bb5cbce2",
"eventID": "cd75a605-2fee-4fda-b847-9c3d330ebaae",
"eventType": "AwsApiCall",
"recipientAccountId": "111122223333"
},
{
    "eventVersion": "1.02",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111122223333:user/Alice",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2014-11-05T21:35:35Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "constraints": {
            "encryptionContextSubset": {
                "aws:ebs:id": "vol-f67bafb2"
            }
        },
        "granteePrincipal": "111122223333:aws:ec2-infrastructure:i-81e2f56c",
        "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": {
        "grantId": "abcde1237f76e4ba7987489ac329fbfba6ad343d6f7075dbd1ef191f0120514a"
    },
    "requestID": "41c4b4f7-8bce-4773-bf0e-5ae3bb5cbce2",
    "eventID": "c1ad79e3-0d3f-402a-b119-d5c31d7c6a6c",
    "readOnly": false,

```

```

    "resources": [
      {
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "accountId": "111122223333"
      }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
  },
  {
    "eventVersion": "1.02",
    "userIdentity": {
      "type": "IAMUser",
      "principalId": "EX_PRINCIPAL_ID",
      "arn": "arn:aws:iam::111122223333:user/Alice",
      "accountId": "111122223333",
      "accessKeyId": "EXAMPLE_KEY_ID",
      "userName": "Alice"
    },
    "eventTime": "2014-11-05T21:35:32Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyWithoutPlaintext",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
      "encryptionContext": {
        "aws:ebs:id": "vol-f67bafb2"
      },
      "numberOfBytes": 64,
      "keyId": "alias/aws/ebs"
    },
    "responseElements": null,
    "requestID": "create-111122223333-758247346-1415223332",
    "eventID": "ac3cab10-ce93-4953-9d62-0b6e5cba651d",
    "readOnly": true,
    "resources": [
      {
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "accountId": "111122223333"
      }
    ],

```

```

    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
  },
  {
    "eventVersion": "1.02",
    "userIdentity": {
      "type": "AssumedRole",
      "principalId": "111122223333:aws:ec2-infrastructure:i-81e2f56c",
      "arn": "arn:aws:sts::111122223333:assumed-role/aws:ec2-infrastructure/
i-81e2f56c",
      "accountId": "111122223333",
      "accessKeyId": "",
      "sessionContext": {
        "attributes": {
          "mfaAuthenticated": "false",
          "creationDate": "2014-11-05T21:35:38Z"
        }
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "111122223333:aws:ec2-infrastructure",
        "arn": "arn:aws:iam::111122223333:role/aws:ec2-infrastructure",
        "accountId": "111122223333",
        "userName": "aws:ec2-infrastructure"
      }
    }
  },
  "eventTime": "2014-11-05T21:35:47Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "Decrypt",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "requestParameters": {
    "encryptionContext": {
      "aws:ebs:id": "vol-f67bafb2"
    }
  },
  "responseElements": null,
  "requestID": "b4b27883-6533-11e4-b4d9-751f1761e9e5",
  "eventID": "edb65380-0a3e-4123-bbc8-3d1b7cff49b0",
  "readOnly": true,
  "resources": [
    {
      "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",

```

```
        "accountId": "111122223333"
      }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
  }
]
```

使用 Amazon CloudWatch 監控 KMS 金鑰

您可以使用 AWS KMS keys [Amazon CloudWatch](#) 來監控您的，Amazon CloudWatch 是一種 AWS 服務，可收集原始資料並將其處理 AWS KMS 為可讀且近乎即時的指標。這些資料會保留 2 週，因此您可以存取歷史資訊，更加了解您 KMS 金鑰的使用情形及其隨時間的變更。

您可以使用 Amazon CloudWatch 對您的重要事件發出提醒，例如下列事件。

- KMS 金鑰中匯入的金鑰材料接近其過期日期。
- 仍然會使用等待刪除的 KMS 金鑰。
- KMS 金鑰中的金鑰材料會自動輪換。
- 已刪除 KMS 金鑰。

您也可以建立 [Amazon CloudWatch](#) 警示，在您的請求率達到配額值的特定百分比時發出提醒。如需詳細資訊，請參閱 AWS 安全部落格中的[使用 Service Quotas 和 Amazon CloudWatch 管理您的 AWS KMS API 請求率](#)。

AWS KMS 指標和維度

AWS KMS 預先定義 Amazon CloudWatch 指標，讓您更輕鬆地監控關鍵資料並建立警示。您可以使用 AWS 管理主控台 和 Amazon CloudWatch API 檢視 AWS KMS 指標。

本節列出每個 AWS KMS 指標和每個指標的維度，並提供一些根據這些指標和維度建立 CloudWatch 警示的基本指導。

Note

維度群組名稱：

若要在 Amazon CloudWatch 主控台中檢視指標，請在 Metrics (指標) 區段中選取維度群組名稱。然後，您可以按 Metric name (指標名稱) 進行篩選。本主題包括每個 AWS KMS 指標的指標名稱和維度群組名稱。

您可以使用 AWS 管理主控台 和 Amazon CloudWatch API 檢視 AWS KMS 指標。如需詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的[檢視可用的指標](#)。

主題

- [SuccessfulRequest](#)
- [SecondsUntilKeyMaterialExpiration](#)
- [CloudHSMKeyStoreThrottle](#)
- [ExternalKeyStoreThrottle](#)
- [XksProxyCertificateDaysToExpire](#)
- [XksProxyCredentialAge](#)
- [XksProxyErrors](#)
- [XksExternalKeyManagerStates](#)
- [XksProxyLatency](#)

SuccessfulRequest

在特定 KMS 金鑰上成功請求密碼編譯操作的次數。透過使用 SuccessfulRequest 指標，您可以將金鑰層級篩選套用至 CloudWatch 中的 AWS KMS API 用量。此指標的 Sum 統計資料會定義期間內成功請求的總數。

使用此指標來識別哪些 KMS 金鑰消耗請求配額的最大部分，或對 API 費用的貢獻最大。您也可以根據 SuccessfulRequest 指標建立 CloudWatch 警示，以通知您異常 AWS KMS 的 API 使用模式。這些提醒有助於識別效率不佳的工作流程，這些工作流程可能會意外超過您的請求配額或產生非預期的費用。

的維度 SuccessfulRequest

維度	Description
KeyArn	每個 KMS 金鑰的值。

維度	Description
作業	每個 AWS KMS API 操作的值。此指標僅適用於密碼編譯操作。

對於 [ReEncrypt](#) 操作，SuccessfulRequest 指標包含來源和目的地 KMS 金鑰的維度。

維度	Description
SourceKey Arn	解密加密文字的 KMS 金鑰值。
DestinationKeyArn	重新加密資料的 KMS 金鑰值。
作業	每個 AWS KMS API 操作的值，在此案例中為 ReEncrypt。

SecondsUntilKeyMaterialExpiration

在 KMS [金鑰中最早過期的匯入金鑰材料](#) 之前剩餘的秒數。此指標僅對具有匯入金鑰材料 (EXTERNAL 的 [金鑰材料來源](#)) 和到期日的 KMS 金鑰有效。

使用此指標來追蹤在您最早過期的匯入金鑰材料過期之前剩餘的時間。當該時間低於您定義的閾值時，您應該重新匯入具有新過期日期的金鑰材料，以保持 KMS 金鑰可用。該 SecondsUntilKeyMaterialExpiration 指標專用於 KMS 金鑰。您無法使用此指標來監控多個 KMS 金鑰或您可能在將來建立的 KMS 金鑰。如需建立 CloudWatch 警示以監控此指標的說明，請參閱 [為匯入金鑰材料過期建立 CloudWatch 警示](#)。

此指標最實用的統計數字是 Minimum，其會告訴您指定統計週期內所有資料點之剩餘時間的最小數量。此指標的唯一有效單位是 Seconds。

維度群組名稱：Per-Key Metrics (每個金鑰指標)

SecondsUntilKeyMaterialExpiration 的維度

維度	描述；與 相關 AWS
KeyId	每個 KMS 金鑰的值。

當您[排程刪除](#) KMS 金鑰時，AWS KMS 會在刪除 KMS 金鑰之前強制執行等待期間。您可以利用等待期間來確保您現在或未來都不需要該 KMS 金鑰。您也可以設定 CloudWatch 警示來警告您有人或應用程式在等待期間嘗試在[密碼編譯操作](#)中使用 KMS 金鑰。如果您收到這類警示的通知，您可能需要取消刪除 KMS 金鑰。

如需說明，請參閱[建立偵測 KMS 金鑰待刪除使用情況的警示](#)。

CloudHSMKeyStoreThrottle

AWS KMS 調節（回應）的每個金鑰存放區中 KMS AWS CloudHSM 金鑰的密碼編譯操作請求數量 `ThrottlingException`。此指標僅適用於 AWS CloudHSM 金鑰存放區。

`CloudHSMKeyStoreThrottle` 指標僅適用於金鑰存放區中的 AWS CloudHSM KMS 金鑰，也僅適用於[密碼編譯操作](#)的請求。當請求速率超過 AWS CloudHSM 金鑰存放區的[自訂金鑰存放區請求配額](#)時，AWS KMS [會調節這些](#)請求。此指標也包含 AWS CloudHSM 叢集的限流。

維度群組名稱：Keystore Throttle Metrics (金鑰存放區限流指標)

維度	Description
CustomKeyStoreId	每個 AWS CloudHSM 金鑰存放區的值。
KmsOperation	每個 AWS KMS API 操作的值。此指標僅適用於金鑰存放區中 KMS AWS CloudHSM 金鑰的密碼編譯操作。
KeySpec	每個 KMS 金鑰類型的值。金鑰存放區中 KMS 金鑰唯一支援的 AWS CloudHSM 金鑰規格 是 SYMMETRIC_DEFAULT。

ExternalKeyStoreThrottle

AWS KMS 調節的每個外部金鑰存放區中 KMS 金鑰的密碼編譯操作請求數量（回應 `ThrottlingException`）。此指標只適用於[外部金鑰存放區](#)。

`ExternalKeyStoreThrottle` 指標僅適用於外部金鑰存放區中的 KMS 金鑰，也僅適用於[密碼編譯操作](#)的請求。當請求速率超過外部[金鑰存放區的自訂金鑰存放區請求配額](#)時，AWS KMS [會調節這些](#)請求。此指標不包括外部金鑰存放區代理或外部金鑰管理器的限流。

使用此指標來檢閱和調整自訂金鑰存放區請求配額的值。如果此指標指出 AWS KMS 經常調節您對這些 KMS 金鑰的請求，您可以考慮請求增加自訂金鑰存放區請求配額值。如需相關說明，請參閱《Service Quotas 使用者指南》中的[請求提升配額](#)。

如果您經常收到 `KMSInvalidStateException` 錯誤，並且訊息說明「因為請求率非常高」而拒絕請求，或「因為外部金鑰存放區代理未及時回應」而拒絕請求，則可能表示您的外部金鑰管理器或外部金鑰存放區代理無法跟上目前的請求率。如果可能，請降低您的請求率。您也可以考慮請求減少自訂金鑰存放區請求配額值。減少此配額值可能會增加限流（和 `ExternalKeyStoreThrottle` 指標值），但表示在將多餘的請求傳送到外部金鑰存放區代理或外部金鑰管理器之前 AWS KMS，會快速拒絕這些請求。若要請求減少配額，請造訪 [AWS 支援中心](#) 並建立案例。

維度群組名稱：Keystore Throttle Metrics (金鑰存放區限流指標)

維度	Description
CustomKeyStoreId	每個外部金鑰存放區的值。
KmsOperation	每個 AWS KMS API 操作的值。此指標僅適用於外部金鑰存放區中 KMS 金鑰的密碼編譯操作。
KeySpec	每個 KMS 金鑰類型的值。外部金鑰存放區中 KMS 金鑰的唯一受支援的 金鑰規格 為 SYMMETRIC_DEFAULT。

XksProxyCertificateDaysToExpire

[外部金鑰存放區代理端點](#) (XksProxyUriEndpoint) 的 TLS 憑證到期前的天數。此指標只適用於[外部金鑰存放區](#)。

使用此指標建立 CloudWatch 警示，通知您 TLS 憑證即將到期。當憑證過期時，AWS KMS 無法與外部金鑰存放區代理通訊。在您續約憑證之前，外部金鑰存放區中受 KMS 金鑰保護的所有資料都無法存取。

憑證警示可防止憑證過期，這可能使您無法存取已加密的資源。設定警示可讓組織有時間在憑證過期之前續約憑證。

維度群組名稱：XKS Proxy Certificate Metrics (XKS 代理憑證指標)

維度	Description
CustomKeyStoreId	每個外部金鑰存放區的值。
CertificateName	TLS 憑證中的主體名稱 (CN)。

您可以根據外部金鑰存放區和外部金鑰存放區中 KMS 金鑰的指標來建立 CloudWatch 警示。如需說明，請參閱[監控外部金鑰存放區](#)。

XksProxyCredentialAge

自當前外部金鑰存放區[代理身分驗證憑證](#) (XksProxyAuthenticationCredential) 與外部金鑰存放區相關聯之後的天數。當您在建立或更新外部金鑰存放區時輸入身分驗證憑證，此計數便會開始。此指標只適用於[外部金鑰存放區](#)。

此值旨在提醒您身分驗證憑證的有效期。但是，由於我們會在您將憑證與外部金鑰存放區建立關聯時開始計數，而不是在外部金鑰存放區代理上建立身分驗證憑證時，因此這可能不是代理上憑證有效期的準確指標。

使用此指標建立 CloudWatch 警示，提醒您輪換外部金鑰存放區代理身分驗證憑證。

維度群組名稱：Per-Keystore Metrics (每個金鑰存放區指標)

維度	Description
CustomKeyStoreId	每個外部金鑰存放區的值。

您可以根據外部金鑰存放區和外部金鑰存放區中 KMS 金鑰的指標來建立 CloudWatch 警示。如需說明，請參閱[監控外部金鑰存放區](#)。

XksProxyErrors

與外部[金鑰存放區代理](#) AWS KMS 請求相關的例外狀況數目。此計數包含外部金鑰存放區代理傳回的例外狀況，AWS KMS 以及外部金鑰存放區代理未在 250 毫秒逾時間隔 AWS KMS 內回應時發生的逾時錯誤。此指標只適用於[外部金鑰存放區](#)。

使用此指標可追蹤外部金鑰存放區中 KMS 金鑰的錯誤率。它顯示了最常見的錯誤，因此您可以優先考慮工程工作。例如，產生高速率不可重試錯誤的 KMS 金鑰可能表示外部金鑰存放區的組態有問題。若要檢視外部金鑰存放區組態，請參閱 [檢視外部金鑰存放區](#)。若要編輯外部金鑰存放區設定，請參閱 [編輯外部金鑰存放區屬性](#)。

維度群組名稱：XKS Proxy Error Metrics (XKS 代理錯誤指標)

維度	Description
CustomKeyStoreId	每個外部金鑰存放區的值。
KmsOperation	產生 XKS 代理請求的每個 AWS KMS API 操作的值。
XksOperation	每個 外部金鑰存放區代理 API 操作 的值。
KeySpec	每個 KMS 金鑰類型的值。外部金鑰存放區中 KMS 金鑰的唯一受支援的 金鑰規格 為 SYMMETRIC_DEFAULT。
ErrorType	數值： - 可重試的錯誤：可能是暫時性的，例如網路錯誤。 - 不可重試的錯誤：可能表示自訂金鑰存放區組態或外部元件有問題。 - N/A：請求成功；沒有錯誤
ExceptionName	數值： - 例外狀況的名稱 - 無：請求成功；沒有錯誤

您可以根據外部金鑰存放區和外部金鑰存放區中 KMS 金鑰的指標來建立 CloudWatch 警示。如需說明，請參閱[監控外部金鑰存放區](#)。

XksExternalKeyManagerStates

下列每個運作狀態中的[外部金鑰管理器執行個體](#)的數目計數：Active、Degraded 和 Unavailable。此指標的資訊來自與每個外部金鑰存放區關聯的外部金鑰存放區代理。此指標只適用於[外部金鑰存放區](#)。

以下是與外部金鑰存放區相關聯之外部金鑰管理器執行個體的運作狀態。每個外部金鑰存放區代理可能會使用不同的指標來測量外部金鑰管理器的運作狀態。如需詳細資訊，請參閱外部金鑰存放區代理的文件。

- Active：外部金鑰管理器運作正常。
- Degraded：外部金鑰管理器運作不正常，但仍可提供流量
- Unavailable：外部金鑰管理器無法提供流量。

使用此指標建立 CloudWatch 警示，以提醒您存在降級和無法使用的外部金鑰管理器執行個體。若要確定處於每個狀態的外部金鑰管理器執行個體，請參閱外部金鑰存放區代理日誌。

維度群組名稱：XKS External Key Manager Metrics (XKS 外部金鑰管理器指標)

維度	Description
CustomKeyStoreId	每個外部金鑰存放區的值。
XksExternalKeyManagerState	每個運作狀態的值。

您可以根據外部金鑰存放區和外部金鑰存放區中 KMS 金鑰的指標來建立 CloudWatch 警示。如需說明，請參閱[監控外部金鑰存放區](#)。

XksProxyLatency

外部金鑰存放區代理回應 AWS KMS 請求所需的毫秒數。如果請求逾時，則記錄的值為 250 毫秒逾時限制。此指標只適用於[外部金鑰存放區](#)。

使用此指標可評估外部金鑰存放區代理和外部金鑰管理器的效能。例如，如果代理在加密和解密操作中經常逾時，請諮詢您的外部代理管理員。

回應緩慢也可能表示您的外部金鑰管理器無法處理目前的請求流量。AWS KMS 建議外部金鑰管理器每秒最多可處理 1800 個密碼編譯操作的請求。如果您的外部金鑰管理器無法處理每秒 1800 個請求，請考慮請求減少[自訂金鑰存放區中 KMS 金鑰的請求配額](#)。使用外部金鑰存放區中的 KMS 金鑰進行密碼編譯操作的請求會快速檢錯，並發生[限流例外狀況](#)，而不是被外部金鑰存放區代理或外部金鑰管理器處理並拒絕。

維度群組名稱：XKS Proxy Latency Metrics (XKS 代理延遲指標)

維度	Description
CustomKeyStoreId	每個外部金鑰存放區的值。
KmsOperation	產生 XKS 代理請求的每個 AWS KMS API 操作的值。
XksOperation	每個 外部金鑰存放區代理 API 操作 的值。
KeySpec	每個 KMS 金鑰類型的值。外部金鑰存放區中 KMS 金鑰的唯一受支援的 金鑰規格 為 SYMMETRIC_DEFAULT。

您可以根據外部金鑰存放區和外部金鑰存放區中 KMS 金鑰的指標來建立 CloudWatch 警示。如需說明，請參閱[監控外部金鑰存放區](#)。

為匯入金鑰材料過期建立 CloudWatch 警示

您可以建立 CloudWatch 警示，在 KMS 金鑰中的匯入金鑰資料臨近其到期時間時通知您。例如，警示會在到期時間不到 30 天時通知您。

當您[匯入金鑰資料至 KMS 金鑰時](#)，您可以選擇指定該金鑰資料的到期日期和時間。當金鑰材料過期時，會 AWS KMS 刪除金鑰材料，而 KMS 金鑰變得無法使用。若要再次使用 KMS 金鑰，您必須[重新匯入金鑰資料](#)。但是，如果您在金鑰資料到期前重新匯入，則可以避免中斷使用該 KMS 金鑰的程序。

此警示使用針對 KMS AWS KMS 金鑰發佈至 CloudWatch 的[SecondsUntilKeyMaterialExpires](#)指標，其匯入的金鑰材料已過期。每個警示都會使用此指標來監控特定 KMS 金鑰的匯入金鑰資料。您無法為具有到期金鑰資料的所有 KMS 金鑰建立單一警示，也不能為您未來可能建立的 KMS 金鑰建立警示。

需求

監控匯入金鑰資料到期的 CloudWatch 警示需要下列資源。

- 具有到期匯入金鑰資料的 KMS 金鑰。
- Amazon SNS 主題。如需詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的[建立 Amazon SNS 主題](#)。

建立警示

按照[建立以靜態閾值為基礎的 CloudWatch 警示](#)中的指示，使用以下所需值。對於其他欄位，請接受預設值，並按要求提供名稱。

欄位	Value
選取指標	選擇 KMS，然後選擇 Per-Key Metrics (每個金鑰指標)。 選擇包含 KMS 金鑰和 SecondsUntilKeyMaterialExpires 指標的資料列。然後選擇 Select metric (選取指標)。 對於具有會到期的匯入金鑰資料之 KMS 金鑰的指標，Metrics (指標) 清單只會顯示 SecondsUntilKeyMaterialExpires 指標。如果您在帳戶和區域中沒有具有這些屬性的 KMS 金鑰，則此清單為空白。
統計數字	下限
Period	1 分鐘
閾值類型	靜態
Whenever ...	每當####大於 1 時

為外部金鑰存放區建立 CloudWatch 警示

您可以根據外部金鑰存放區指標建立 Amazon CloudWatch 警示，以便在指標值超過您指定的閾值時通知您。警示可將訊息傳送至 [Amazon Simple Notification Service \(Amazon SNS\) 主題](#) 或 [Amazon EC2 Auto Scaling 政策](#)。如需 CloudWatch 警示的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的[使用 Amazon CloudWatch 警示](#)

在建立 Amazon CloudWatch 警示之前，您需要 Amazon SNS 主題。如需詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的[建立 Amazon SNS 主題](#)。

主題

- [建立憑證過期的警示](#)
- [建立回應逾時的警示](#)
- [建立可重試錯誤的警示](#)
- [針對不可重試的錯誤建立警示](#)

建立憑證過期的警示

此警示會使用 AWS KMS 發佈至 CloudWatch 的 [XksProxyCertificateDaysToExpire](#) 指標來記錄與外部金鑰存放區代理端點關聯的 TLS 憑證的預期到期日。您無法為您帳戶中的所有外部金鑰存放區建立單一警示，也不能為您將來可能建立的外部金鑰存放區建立警示。

我們建議設定警示，以便在憑證設定到期的前 10 天提醒您，但您應該設定最符合您需求的閾值。

建立警示

按照[建立以靜態閾值為基礎的 CloudWatch 警示](#)中的指示，使用以下所需值。對於其他欄位，請接受預設值，並按要求提供名稱。

欄位	Value
選取指標	選擇 KMS，然後選擇 XKS Proxy Certificate Metrics (XKS 代理憑證指標)。 選取您想要監控的 XksProxyCertificateName 旁的核取方塊。 然後選擇 Select metric (選取指標)。
統計數字	下限
Period	5 分鐘
閾值類型	靜態
Whenever ...	每當 XksProxyCertificateDaysToExpire Lower 10 時。

建立回應逾時的警示

此警示使用發佈至 CloudWatch AWS KMS 的 [XksProxyLatency](#) 指標來記錄外部金鑰存放區代理回應 AWS KMS 請求所需的毫秒數。您無法為您帳戶中的所有外部金鑰存放區建立單一警示，也不能為您將來可能建立的外部金鑰存放區建立警示。

AWS KMS 預期外部金鑰存放區代理會在 250 毫秒內回應每個請求。我們建議您設定警示，以便在外部金鑰存放區代理的回應時間超過 200 毫秒時提醒您，但您應該設定最符合您需求的閾值。

建立警示

按照[建立以靜態閾值為基礎的 CloudWatch 警示](#)中的指示，使用以下所需值。對於其他欄位，請接受預設值，並按要求提供名稱。

欄位	Value
選取指標	選擇 KMS，然後選擇 XKS Proxy Latency Metrics (XKS 代理延遲指標)。 選取您想要監控的 KmsOperation 旁的核取方塊。 然後選擇 Select metric (選取指標)。
統計數字	平均數
Period	5 分鐘
閾值類型	靜態
Whenever ...	每當 XksProxyLatency Greater 200 時。

建立可重試錯誤的警示

此警示使用發佈至 CloudWatch AWS KMS 的 [XksProxyErrors](#) 指標，記錄與外部金鑰存放區代理 AWS KMS 請求相關的例外狀況數目。您無法為您帳戶中的所有外部金鑰存放區建立單一警示，也不能為您將來可能建立的外部金鑰存放區建立警示。

可重試的錯誤會降低您的可靠性百分比，並可指示網路錯誤。我們建議您設定警示，以便在一分鐘內記錄五個以上的可重試錯誤時提醒您，但您應該設定最符合您需求的閾值。

按照[建立以靜態閾值為基礎的 CloudWatch 警示](#)中的指示，使用以下所需值。對於其他欄位，請接受預設值，並按要求提供名稱。

欄位	Value
選取指標	選擇 Queries (查詢) 索引標籤。 將 Namespace (命名空間) 選為 AWS/KMS。 針對 Metric name (指標名稱)，輸入 SUM(XksProxyErrors) 。 針對 Filter by (篩選依據)，輸入 ErrorType = Retryable 。 選擇執行。然後選擇 Select metric (選取指標)。
標籤	#####
Period	1 分鐘
閾值類型	靜態
Whenever ...	每當 q1 Greater 5 時。

針對不可重試的錯誤建立警示

此警示使用發佈至 CloudWatch AWS KMS 的 [XksProxyErrors](#) 指標，記錄與外部金鑰存放區代理 AWS KMS 請求相關的例外狀況數目。您無法為您帳戶中的所有外部金鑰存放區建立單一警示，也不能為您將來可能建立的外部金鑰存放區建立警示。

不可重試的錯誤表示外部金鑰存放區的組態有問題。我們建議您設定警示，以便在一分鐘內記錄五個以上的不可重試錯誤時提醒您，但您應該設定最符合您需求的閾值。

按照[建立以靜態閾值為基礎的 CloudWatch 警示](#)中的指示，使用以下所需值。對於其他欄位，請接受預設值，並按要求提供名稱。

欄位	Value
選取指標	選擇 Queries (查詢) 索引標籤。 將 Namespace (命名空間) 選為 AWS/KMS。 針對 Metric name (指標名稱)，輸入 SUM(XksProxyErrors) 。

欄位	Value
	針對 Filter by (篩選依據)，輸入 <code>ErrorType = Non-retryable</code> 。
	選擇執行。然後選擇 Select metric (選取指標)。
標籤	#####
Period	1 分鐘
閾值類型	靜態
Whenever ...	每當 q1 Greater 5 時。

使用 Amazon EventBridge 監控 KMS 金鑰

您可以使用 Amazon EventBridge (原稱為 Amazon CloudWatch Events)，提醒您注意 KMS 金鑰生命週期中的下列重要事件。

- KMS 金鑰中的金鑰材料會自動輪換或隨需輪換。
- KMS 金鑰中所匯入的金鑰資料過期。
- 已排程刪除的 KMS 金鑰已被刪除。

AWS KMS 與 Amazon EventBridge 整合，以通知您影響 KMS 金鑰的重要事件。每個事件以 [JSON \(JavaScript 物件標記法\)](#) 來表示，並包含事件名稱、事件的發生日期和時間，以及受影響的項目。您可以收集這些事件並建立規則，將它們路由到一或多個目標，例如 AWS Lambda 函數、Amazon SNS 主題、Amazon SQS 佇列、Amazon Kinesis Data Streams 中的串流或內建目標。

如需搭配其他類型的事件使用 EventBridge 的詳細資訊，包括記錄讀取/寫入 API 請求 AWS CloudTrail 時由發出的事件，請參閱 [Amazon EventBridge 使用者指南](#)。

下列主題說明 AWS KMS 產生的 EventBridge 事件。

KMS CMK Rotation (KMS CMK 輪換)

AWS KMS 支援對稱加密 KMS 金鑰中金鑰材料的 [自動和隨需輪換](#)。

每當 AWS KMS 輪換金鑰材料時，就會傳送 KMS CMK Rotation 事件至 EventBridge。會盡最大努力 AWS KMS 產生此事件。

以下為此事件的範例。

```
{
  "version": "0",
  "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
  "detail-type": "KMS CMK Rotation",
  "source": "aws.kms",
  "account": "111122223333",
  "time": "2025-05-23T03:11:54Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  ],
  "detail": {
    "key-id": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "key-origin": "AWS_KMS",
    "rotation-type": "ON_DEMAND",
    "previous-key-material-id":
"123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0",
    "current-key-material-id":
"96083e4fb6dbc41d77578a213a6b6669c044dd4c143e96755396d2bf11fd6068"
  }
}
```

KMS Imported Key Material Expiration (KMS 匯入的金鑰資料過期)

當您[匯入金鑰材料至 KMS 金鑰時](#)，您可以選擇指定該金鑰材料的過期時間。當金鑰材料過期時，會 AWS KMS 刪除金鑰材料，並將對應的 KMS Imported Key Material Expiration 事件傳送至 EventBridge。會盡最大努力 AWS KMS 產生此事件。

以下為此事件的範例。

```
{
  "version": "0",
  "id": "9da9af57-9253-4406-87cb-7cc400e43465",
  "detail-type": "KMS Imported Key Material Expiration",
  "source": "aws.kms",
  "account": "111122223333",
  "time": "2025-05-23T03:11:54Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  ]
}
```

```
],
"detail": {
  "key-id": "1234abcd-12ab-34cd-56ef-1234567890ab",
  "key-material-id":
"123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef0"
}
}
```

KMS CMK Deletion (KMS CMK 的刪除)

當您[排程刪除](#) KMS 金鑰時，AWS KMS 會在刪除 KMS 金鑰之前強制執行等待期間。等待期間結束後，AWS KMS 會刪除 KMS 金鑰，並傳送 KMS CMK Deletion 事件至 EventBridge。AWS KMS 會確保此 EventBridge 事件。由於重試，它可能會在幾秒鐘內產生多個刪除相同 KMS 金鑰的事件。

以下為此事件的範例。

```
{
  "version": "0",
  "id": "e9ce3425-7d22-412a-a699-e7a5fc3fbc9a",
  "detail-type": "KMS CMK Deletion",
  "source": "aws.kms",
  "account": "111122223333",
  "time": "2025-05-23T03:11:54Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  ],
  "detail": {
    "key-id": "1234abcd-12ab-34cd-56ef-1234567890ab"
  }
}
```

中的別名 AWS KMS

別名是 AWS KMS key 的易記名稱。例如，別名可讓您將 KMS 金鑰稱為 test-key，而不是 1234abcd-12ab-34cd-56ef-1234567890ab。

您可以使用別名在 AWS KMS 主控台、[DescribeKey](#) 操作和[密碼編譯操作](#)中識別 KMS 金鑰，例如 [Encrypt](#) 和 [GenerateDataKey](#)。別名也可以很容易地識別 [AWS 受管金鑰](#)。這些 KMS 金鑰的別名一律具有 aws/<service-name> 格式。例如，AWS 受管金鑰 適用於 Amazon DynamoDB 的別名為 aws/dynamodb。您可以為專案建立類似的別名標準，例如在別名前面加上專案或品類的名稱。

您也可以根據其別名允許和拒絕存取 KMS 金鑰，而不需編輯政策或管理授權。此功能是[屬性型存取控制](#) (ABAC) AWS KMS 支援的一部分。如需詳細資訊，請參閱[使用別名來控制對 KMS 金鑰的存取](#)。

別名的大部分功能來自於您隨時變更與別名相關聯之 KMS 金鑰的能力。別名可以讓您的程式碼更容易撰寫和維護。例如，假設您使用別名來指代特定的 KMS 金鑰，而且您想要變更 KMS 金鑰。在這種情況下，只要將別名與不同的 KMS 金鑰建立關聯即可。您不需要變更程式碼。

別名也可以更容易在不同的 AWS 區域中重複使用相同的程式碼。在多個區域中建立具有相同名稱的別名，並將每個別名與其區域中的 KMS 金鑰產生關聯。當程式碼在每個區域中執行時，別名是指該區域中相關聯的 KMS 金鑰。如需範例，請參閱 [了解如何在應用程式中使用別名](#)。

您可以在 AWS KMS 主控台中使用 [CreateAlias](#) API 或使用 [AWS::KMS::Alias CloudFormation 範本](#) 來建立 KMS 金鑰的別名。

AWS KMS API 可完全控制每個帳戶和區域中的別名。API 包含用於建立別名 ([CreateAlias](#))、檢視別名名稱和別名 ARN ([ListAliases](#))、變更與別名相關聯的 KMS 金鑰 ([UpdateAlias](#)) 和刪除別名 ([DeleteAlias](#)) 的操作。

別名的運作方式

了解別名如何在 AWS KMS 運作。

別名是獨立的 AWS 資源

別名不是 KMS 金鑰的屬性。您對別名所採取的動作不會影響其關聯的 KMS 金鑰。您可以為 KMS 金鑰建立別名，然後更新別名，讓其與不同的 KMS 金鑰相關聯。您甚至可以刪除別名，而不會對相關聯的 KMS 金鑰造成任何影響。不過，如果您刪除 KMS 金鑰，則會刪除與該 KMS 金鑰相關聯的所有別名。

如果您在 IAM 政策中指定別名作為資源，則政策會指代別名，而不是關聯的 KMS 金鑰。
每個別名都有兩種格式

當您建立別名時，您可以指定別名 name。AWS KMS creates the alias ARN。

- [別名 ARN](#) 是唯一可識別別名的 Amazon Resource Name (ARN)。

```
# Alias ARN
arn:aws:kms:us-west-2:111122223333:alias/<alias-name>
```

- 在帳戶和區域中，[別名名稱](#)必須是唯一的。在 AWS KMS API 中，別名名稱一律以 為字首 alias/。在 AWS KMS 主控台中省略該字首。

```
# Alias name
alias/<alias-name>
```

別名並非秘密

別名可能在 CloudTrail 日誌與其他輸出中以純文字顯示。請勿在別名名稱包含機密或敏感資訊。
每個別名一次都與一個 KMS 金鑰相關聯

別名及其 KMS 金鑰必須位於相同的帳戶和區域中。

您可以將別名與相同 AWS 帳戶 和區域中的任何[客戶受管金鑰](#)建立關聯。不過，您沒有將別名與[AWS 受管金鑰](#)相關聯的許可。

例如，此 [ListAliases](#) 輸出顯示 test-key 別名僅與一個目標 KMS 金鑰相關聯，這由 TargetKeyId 屬性表示。

```
{
  "AliasName": "alias/test-key",
  "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/test-key",
  "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
  "CreationDate": 1593622000.191,
  "LastUpdatedDate": 1593622000.191
}
```

多個別名可以與相同的 KMS 金鑰建立關聯

例如，您可以將 test-key 和 project-key 別名與相同的 KMS 金鑰相關聯。

```
{
```



```
"AliasName": "alias/test-key",
"AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/test-key",
"TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
"CreationDate": 1593622000.191,
"LastUpdatedDate": 1593622000.191
},
{
  "AliasName": "alias/project-key",
  "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/project-key",
  "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
  "CreationDate": 1516435200.399,
  "LastUpdatedDate": 1516435200.399
}
```

別名在帳戶和區域中必須是唯一的。

例如，您在每個帳戶和區域只能有一個 test-key 別名。別名區分大小寫，但只有大小寫不同的別名很容易出錯。您無法變更別名名稱。但是，您可以刪除別名，並使用所需名稱建立新別名。

但是，您可以在不同區域中使用相同的名稱建立別名。

例如，您可以在美國東部 (維吉尼亞北部) 有一個 finance-key 別名和在歐洲 (法蘭克福) 有一個 finance-key 別名。每個別名都會與其區域中的 KMS 金鑰相關聯。如果您的程式碼引用了類似 alias/finance-key 的別名名稱，則可以在多個區域中執行。在每個區域中，它會使用不同的 KMS 金鑰。如需詳細資訊，請參閱[了解如何在應用程式中使用別名](#)。

您可以變更與別名相關聯的 KMS 金鑰

您可以使用 [UpdateAlias](#) 操作將別名與不同的 KMS 金鑰相關聯。例如，如果 finance-key 別名與 1234abcd-12ab-34cd-56ef-1234567890ab KMS 金鑰相關聯，則您可以對其進行更新，使其與 0987dcba-09fe-87dc-65ba-ab0987654321 KMS 金鑰相關聯。

不過，目前的和新的 KMS 金鑰必須是相同類型 (兩個皆為對稱或兩個皆為非對稱或兩個皆為 HMAC)，並且必須具有相同的[金鑰使用情形](#) (ENCRYPT_DECRYPT 或 SIGN_VERIFY 或 GENERATE_VERIFY_MAC)。此限制可避免使用別名之程式碼中的錯誤。如果您必須將別名與不同類型的金鑰建立關聯，且已降低風險，則可以刪除並重新建立別名。

某些 KMS 金鑰沒有別名

在 AWS KMS 主控台中建立 KMS 金鑰時，您必須為其提供新的別名。但是，當使用 [CreateKey](#) 操作來建立 KMS 金鑰時，不需要別名。此外，您可利用 [UpdateAlias](#) 操作來變更關聯別名的 KMS 金鑰，以及 [DeleteAlias](#) 操作來刪除別名。因此，某些 KMS 金鑰可能有數個別名，有些金鑰可能沒有。

AWS 會在您的帳戶中建立別名

AWS 會在您的帳戶中為 建立別名 [AWS 受管金鑰](#)。這些別名有 `alias/aws/<service-name>` 格式的名稱，例如 `alias/aws/s3`。

有些別名沒有 KMS AWS 金鑰。當您開始使用服務 AWS 受管金鑰 時，這些預先定義的別名通常與相關聯。

使用別名來識別 KMS 金鑰

您可以使用 [別名名稱](#) 或 [別名 ARN](#) 來識別 [密碼編譯操作](#) 中的 KMS 金鑰、[DescribeKey](#) 和 [GetPublicKey](#)。(如果 [KMS 金鑰位於不同的 AWS 帳戶](#)，您必須使用其 [金鑰 ARN](#) 或別名 ARN。) 別名在其他 AWS KMS 操作中不是 KMS 金鑰的有效識別符。如需每個 AWS KMS API 操作之有效 [金鑰識別符](#) 的相關資訊，請參閱 AWS Key Management Service API 參考中的 `KeyId` 參數說明。

您不能使用別名名稱或別名 ARN 來 [識別 IAM 政策中的 KMS 金鑰](#)。若要根據其別名控制對 KMS 金鑰的存取權，請使用 [kms:RequestAlias](#) 或 [kms:ResourceAliases](#) 條件索引鍵。如需詳細資訊，請參閱 [適用於的 ABAC AWS KMS](#)。

控制對別名的存取

當您建立或變更別名時，會影響別名及其關聯的 KMS 金鑰。因此，管理別名的委託人必須具有對別名 and 所有受影響之 KMS 金鑰呼叫別名操作的許可。您可以使用 [金鑰政策](#)、[IAM 政策](#) 和 [授權](#) 來提供這些許可。

Note

授予委託人管理標籤和別名的許可時，請務必謹慎。變更標記或別名可允許或拒絕客戶受管金鑰的許可。如需詳細資訊，請參閱 [適用於的 ABAC AWS KMS](#) 和 [使用別名來控制對 KMS 金鑰的存取](#)。

如需控制所有 AWS KMS 操作存取的資訊，請參閱 [許可參考](#)。

建立和管理別名的許可如下所示。

kms:CreateAlias

若要建立別名，委託人需要別名和相關聯的 KMS 金鑰的下列許可。

- 別名的 `kms:CreateAlias`。在連接至允許建立別名之委託人的 IAM 政策中提供此許可。

下列範例政策陳述式會指定 `Resource` 元素中的特定別名。但是，您可以列出多個別名 ARN 或指定別名模式，例如 `"test*"`。您也可以指定 `"*"` 的 `Resource` 值以允許委託人在帳戶和區域中建立任何別名。建立別名的許可也可以包含在帳戶和區域中所有資源的 `kms:Create*` 許可中。

```
{
  "Sid": "IAMPolicyForAnAlias",
  "Effect": "Allow",
  "Action": [
    "kms:CreateAlias",
    "kms:UpdateAlias",
    "kms>DeleteAlias"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:alias/test-key"
}
```

- KMS 金鑰的 `kms:CreateAlias`。必須在金鑰政策或在從金鑰政策委派的 IAM 政策中提供此許可。

```
{
  "Sid": "Key policy for 1234abcd-12ab-34cd-56ef-1234567890ab",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:user/KMSAdminUser"},
  "Action": [
    "kms:CreateAlias",
    "kms:DescribeKey"
  ],
  "Resource": "*"
}
```

您可以使用條件索引鍵來限制可與別名建立關聯的 KMS 金鑰。例如，您可使用 [kms:KeySpec](#) 條件索引鍵，以允許委託人只在非對稱 KMS 金鑰上建立別名。如需您可用於限制 KMS 金鑰資料之 `kms:CreateAlias` 許可的條件索引鍵完整清單，請參閱 [AWS KMS 許可](#)。

kms:ListAliases

若要列出帳戶和區域中的別名，委託人必須具有 IAM 政策的 `kms:ListAliases` 許可。由於此政策與任何特定 KMS 金鑰或別名資源無關，因此政策中資源元素的值必須為 `"*"`。

例如，下列 IAM 政策陳述式會授予委託人許可，以列出帳戶和區域中的所有 KMS 金鑰和別名。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:ListKeys",
      "kms:ListAliases"
    ],
    "Resource": "*"
  }
}
```

kms:UpdateAlias

若要變更與別名相關聯的 KMS 金鑰，委託人需要三個許可元素：一個用於別名、一個用於當前 KMS 金鑰，另一個用於新的 KMS 金鑰。

例如，假設您想要將 test-key 別名從金鑰 ID 為 1234abcd-12ab-34cd-56ef-1234567890ab 的 KMS 金鑰變更為金鑰 ID 為 0987dcba-09fe-87dc-65ba-ab0987654321 的 KMS 金鑰。在這種情況下，請包含類似於本節範例的政策陳述式。

- 別名的 kms:UpdateAlias。您可以在連接至委託人的 IAM 政策中提供此許可。下列 IAM 政策會指定特定別名。但是，您可以列出多個別名 ARN 或指定別名模式，例如 "test*"。您也可以指定 "*" 的 Resource 值以允許委託人更新帳戶和區域中的任何別名。

```
{
  "Sid": "IAMPolicyForAnAlias",
  "Effect": "Allow",
  "Action": [
    "kms:UpdateAlias",
    "kms:ListAliases",
    "kms:ListKeys"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:alias/test-key"
}
```

- 目前與別名相關聯之 KMS 金鑰的 kms:UpdateAlias。必須在金鑰政策或在從金鑰政策委派的 IAM 政策中提供此許可。

```
{
  "Sid": "Key policy for 1234abcd-12ab-34cd-56ef-1234567890ab",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:user/KMSAdminUser"},
  "Action": [
    "kms:UpdateAlias",
    "kms:DescribeKey"
  ],
  "Resource": "*"
}
```

- 操作與別名相關聯之 KMS 金鑰的 `kms:UpdateAlias`。必須在金鑰政策或在從金鑰政策委派的 IAM 政策中提供此許可。

```
{
  "Sid": "Key policy for 0987dcba-09fe-87dc-65ba-ab0987654321",
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::111122223333:user/KMSAdminUser"},
  "Action": [
    "kms:UpdateAlias",
    "kms:DescribeKey"
  ],
  "Resource": "*"
}
```

您可以使用條件索引鍵來限制 `UpdateAlias` 操作中的其中一個 KMS 金鑰，或者兩個金鑰皆限制。例如，您可以使用 [kms:ResourceAliases](#) 條件索引鍵，以允許委託人只在目標 KMS 金鑰已經有特定別名時更新別名。如需您可用於限制 KMS 金鑰資源之 `kms:UpdateAlias` 許可的條件索引鍵完整清單，請參閱 [AWS KMS 許可](#)。

kms:DeleteAlias

若要刪除別名，委託人需要別名和相關聯的 KMS 金鑰的許可。

一如往常，在授予委託人刪除資源的許可時，您應該小心謹慎。不過，刪除別名並不會影響相關聯的 KMS 金鑰。雖然這可能會在依賴別名的應用程式中造成故障，但是如果錯誤地刪除了別名，則您可以重新建立別名。

- 別名的 `kms:DeleteAlias`。在連接至允許刪除別名之委託人的 IAM 政策中提供此許可。

下列範例政策陳述式會指定 Resource 元素中的別名。但是，您可以列出多個別名 ARN 或指定別名模式，例如 "test*"，您也可以指定 "*" 的 Resource 值以允許委託人刪除帳戶和區域中的任何別名。

```
{
  "Sid": "IAMPolicyForAnAlias",
  "Effect": "Allow",
  "Action": [
    "kms:CreateAlias",
    "kms:UpdateAlias",
    "kms:DeleteAlias"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:alias/test-key"
}
```

- 相關聯 KMS 金鑰的 kms:DeleteAlias。必須在金鑰政策或在從金鑰政策委派的 IAM 政策中提供此許可。

```
{
  "Sid": "Key policy for 1234abcd-12ab-34cd-56ef-1234567890ab",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:user/KMSAdminUser"
  },
  "Action": [
    "kms:CreateAlias",
    "kms:UpdateAlias",
    "kms:DeleteAlias",
    "kms:DescribeKey"
  ],
  "Resource": "*"
}
```

限制別名許可

當資源為 KMS 金鑰時，您可以使用條件索引鍵來限制別名許可。例如，以下 IAM 政策允許在特定帳戶和區域中對 KMS 金鑰執行別名操作。不過，它會使用 [kms:KeyOrigin](#) 條件金鑰，進一步限制 KMS 金鑰的許可與金鑰材料 AWS KMS。

如需可用來限制 KMS 金鑰資源別名許可的條件索引鍵完整清單，請參閱 [AWS KMS 許可](#)。

```
{
  "Sid": "IAMPolicyKeyPermissions",
  "Effect": "Allow",
  "Resource": "arn:aws:kms:us-west-2:111122223333:key/*",
  "Action": [
    "kms:CreateAlias",
    "kms:UpdateAlias",
    "kms>DeleteAlias"
  ],
  "Condition": {
    "StringEquals": {
      "kms:KeyOrigin": "AWS_KMS"
    }
  }
}
```

您無法在資源為別名的政策陳述式中使用條件索引鍵。若要限制委託人可以管理的別名，請使用 IAM 政策陳述式 (可控制對別名的存取) 之 `Resource` 元素的值。例如，下列政策陳述式允許主體在 AWS 帳戶 和 區域中建立、更新或刪除任何別名，除非別名以 開頭 `Restricted`。

```
{
  "Sid": "IAMPolicyForAnAliasAllow",
  "Effect": "Allow",
  "Action": [
    "kms:CreateAlias",
    "kms:UpdateAlias",
    "kms>DeleteAlias"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:alias/*"
},
{
  "Sid": "IAMPolicyForAnAliasDeny",
  "Effect": "Deny",
  "Action": [
    "kms:CreateAlias",
    "kms:UpdateAlias",
    "kms>DeleteAlias"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:alias/Restricted*"
}
```

建立別名

您可以在 AWS KMS 主控台或使用 AWS KMS API 操作建立別名。

別名必須是 1-256 個字元的字串。它只能包含英數字元、斜線 (/)、底線 (_) 和破折號 (-)。客戶受管金鑰的別名名稱不能以 alias/aws/ 開頭。此 alias/aws/ 字首已保留供 AWS 受管金鑰使用。

您可以為新的 KMS 金鑰或現有的 KMS 金鑰建立別名。您可以新增別名，以便在專案或應用程式中使用特定的 KMS 金鑰。

您也可以使用 AWS CloudFormation 範本來建立 KMS 金鑰的別名。如需詳細資訊，請參閱 AWS CloudFormation 《使用者指南》中的 [AWS::KMS::Alias](#)。

使用 AWS KMS 主控台

在 AWS KMS 主控台中[建立 KMS 金鑰](#)時，您必須為新的 KMS 金鑰建立別名。若要為現有的 KMS 金鑰建立別名，請使用 KMS 金鑰詳細資訊頁面上的 Aliases (別名) 索引標籤。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。您無法管理 AWS 受管金鑰 或 的別名 AWS 擁有的金鑰。
4. 在資料表中，選擇 KMS 金鑰的金鑰 ID 或別名。然後，在 KMS 金鑰詳細資訊頁面上，選擇 Aliases (別名) 索引標籤。

如果 KMS 金鑰有多個別名，Aliases (別名) 資料行會顯示一個別名和別名摘要，例如 (+n 等)。選擇別名摘要會直接帶您前往 KMS 金鑰詳細資訊頁面上的 Aliases (別名) 索引標籤。

5. 在 Aliases (別名) 索引標籤中，選擇 Create alias (建立別名)。輸入別名名稱，然後選擇 Create alias (建立別名)。

Important

請勿在此欄位包含機密或敏感資訊。此欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

 Note

請勿加入 `alias/` 字首。主控台會自動執行此操作。如果您輸入 `alias/ExampleAlias`，實際的別名名稱將是 `alias/alias/ExampleAlias`。

使用 AWS KMS API

若要建立別名，請使用 [CreateAlias](#) 操作。不同於在主控台中建立 KMS 金鑰的程序，[CreateKey](#) 操作不會為新的 KMS 金鑰建立別名。

 Important

請勿在此欄位包含機密或敏感資訊。此欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

您可以使用 `CreateAlias` 操作為沒有別名的新 KMS 金鑰建立別名。您也可以使用 `CreateAlias` 操作將別名新增至任何現有的 KMS 金鑰，或重新建立意外刪除的別名。

在 AWS KMS API 操作中，別名名稱必須以 `alias/` 開頭，後面接著名稱，例如 `alias/ExampleAlias`。別名在帳戶和區域中必須是唯一的。若要尋找已在使用中的別名，請使用 [ListAliases](#) 操作。別名名稱區分大小寫。

`TargetKeyId` 可以為相同 AWS 區域中的任一項 [客戶受管金鑰](#)。若要識別 KMS 金鑰，請使用它的 [金鑰 ID](#) 或 [金鑰 ARN](#)。您無法使用另一個別名。

下列範例會建立 `example-key` 別名，並將其與指定的 KMS 金鑰建立關聯。這些範例使用 AWS Command Line Interface (AWS CLI)。如需多種程式設計語言的範例，請參閱 [CreateAlias 搭配 AWS SDK 或 CLI 使用](#)。

```
$ aws kms create-alias \  
  --alias-name alias/example-key \  
  --target-key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

`CreateAlias` 不會傳回任何輸出。若要查看新別名，請使用 `ListAliases` 操作。如需詳細資訊，請參閱 [使用 AWS KMS API](#)。

尋找 KMS 金鑰的別名名稱和別名 ARN

別名可讓您在 AWS KMS 主控台中輕鬆識別 KMS 金鑰。您可以在 AWS KMS 主控台或使用 [ListAliases](#) 操作來檢視 KMS 金鑰的別名。[DescribeKey](#) 操作 (會傳回 KMS 金鑰的屬性) 不包含別名。

下列程序示範如何使用 AWS KMS 主控台和 AWS KMS API 檢視和識別與 KMS 金鑰相關聯的別名。AWS KMS API 範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

使用 AWS KMS 主控台

AWS KMS 主控台會顯示與 KMS 金鑰相關聯的別名。

1. 在 <https://console.aws.amazon.com/kms> 開啟 AWS KMS 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 若要檢視您所建立及管理帳戶中的金鑰，請在導覽窗格中選擇 Customer managed keys (客戶受管金鑰)。若要檢視您帳戶中為您 AWS 建立和管理的金鑰，請在導覽窗格中選擇 AWS 受管金鑰。
4. Aliases (別名) 欄會顯示每個 KMS 金鑰的別名。如果 KMS 金鑰沒有別名，則會在 Aliases (別名) 資料欄中使用破折號 (-) 顯示。

如果 KMS 金鑰有多個別名，Aliases (別名) 資料欄也有別名摘要，例如 (+n more) (+n 等)。例如，下列 KMS 金鑰有兩個別名，其中一個是 key-test。

若要尋找 KMS 金鑰所有別名的別名名稱和別名 ARN，請使用 Aliases (別名) 索引標籤。

- 若要直接移至 Aliases (別名) 欄中的 Aliases (別名) 索引標籤，請選擇別名摘要 (+n 等)。只有在 KMS 金鑰具有多個別名時，才會顯示別名摘要。
- 或者，選擇 KMS 金鑰的別名或金鑰 ID (這會開啟 KMS 金鑰的詳細資訊頁面)，然後選擇 Aliases (別名) 索引標籤。索引標籤位於 General configuration (一般組態) 區段。

Customer managed keys (16)				Key actions ▼	Create key
Filter keys by aliases, key ID, or key type				< 1 2 >	⚙️
☐	Aliases ▼	Key ID ▼	Status		
☐	key-test (+1 more)	1234abcd-12ab-34cd-56ef-1234567890ab	Enabled		
☐	-	1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d	Enabled		

5. Aliases (別名) 索引標籤會顯示 KMS 金鑰所有別名的別名名稱和別名 ARN。您也可以在此索引標籤上建立並刪除 KMS 金鑰的別名。

Key policy	Cryptographic configuration	Key material	Tags	Public key	Aliases
Aliases Info Filter by Alias name Delete Create new alias < 1 >					
☐	Alias name	Alias ARN			
☐	key-test	arn:aws:kms:us-east-1:111122223333:alias/key-test			
☐	project-key	arn:aws:kms:us-east-1:111122223333:alias/project-key			

AWS 受管金鑰

您可以使用別名來辨識 AWS 受管金鑰，如本範例 AWS 受管金鑰頁面所示。AWS 受管金鑰 的別名一律具有以下格式：`aws/<service-name>`。例如，AWS 受管金鑰 適用於 Amazon DynamoDB 的別名為 `aws/dynamodb`。

AWS managed keys (9)	
Filter keys by alias or key ID	
Alias	
aws/dynamodb	
aws/ebs	
aws/lightsail	
aws/rds	
aws/s3	
aws/secretsmanager	
aws/ssm	
aws/workmail	
aws/xray	

使用 AWS KMS API

[ListAliases](#) 操作會傳回帳戶和區域中別名的別名名稱和別名 ARN。輸出包含客戶受管金鑰的 AWS 受管金鑰 和 別名。AWS 受管金鑰 別名具有格式 `aws/<service-name>`，例如 `aws/dynamodb`。

回應可能也包含沒有 `TargetKeyId` 欄位的別名。這些是 AWS 已建立但尚未與 KMS 金鑰建立關聯的預先定義別名。

```
$ aws kms list-aliases
{
  "Aliases": [
    {
      "AliasName": "alias/access-key",
      "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/access-key",
      "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
      "CreationDate": 1516435200.399,
      "LastUpdatedDate": 1516435200.399
    },
    {
      "AliasName": "alias/ECC-P521-Sign",
      "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/ECC-P521-Sign",
      "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
      "CreationDate": 1693622000.704,
      "LastUpdatedDate": 1693622000.704
    },
  ],
}
```

```

    "AliasName": "alias/ImportedKey",
    "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/ImportedKey",
    "TargetKeyId": "1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d",
    "CreationDate": 1493622000.704,
    "LastUpdatedDate": 1521097200.235
  },
  {
    "AliasName": "alias/finance-project",
    "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/finance-project",
    "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
    "CreationDate": 1604958290.014,
    "LastUpdatedDate": 1604958290.014
  },
  {
    "AliasName": "alias/aws/dynamodb",
    "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/aws/dynamodb",
    "TargetKeyId": "0987ab65-43cd-21ef-09ab-87654321cdef",
    "CreationDate": 1521097200.454,
    "LastUpdatedDate": 1521097200.454
  },
  {
    "AliasName": "alias/aws/ebs",
    "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/aws/ebs",
    "TargetKeyId": "abcd1234-09fe-ef90-09fe-ab0987654321",
    "CreationDate": 1466518990.200,
    "LastUpdatedDate": 1466518990.200
  }
]
}

```

若要取得與特定 KMS 金鑰關聯的所有別名，請使用 `ListAliases` 操作的選用 `KeyId` 參數。`KeyId` 參數採用 KMS 金鑰的[金鑰 ID](#) 或[金鑰 ARN](#)。

此範例會將所有別名與 `0987dcba-09fe-87dc-65ba-ab0987654321` KMS 金鑰關聯。

```

$ aws kms list-aliases --key-id 0987dcba-09fe-87dc-65ba-ab0987654321
{
  "Aliases": [
    {
      "AliasName": "alias/access-key",
      "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/access-key",
      "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
      "CreationDate": "2018-01-20T15:23:10.194000-07:00",

```

```

        "LastUpdatedDate": "2018-01-20T15:23:10.194000-07:00"
      },
      {
        "AliasName": "alias/finance-project",
        "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/finance-project",
        "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
        "CreationDate": 1604958290.014,
        "LastUpdatedDate": 1604958290.014
      }
    ]
  }
}

```

KeyId 參數不會採用萬用字元，但您可以使用程式設計語言的功能來篩選回應。

例如，下列 AWS CLI 命令只會取得 的別名 AWS 受管金鑰。

```
$ aws kms list-aliases --query 'Aliases[?starts_with(AliasName, `alias/aws/`)]'
```

下列命令只會取得 access-key 別名。別名名稱區分大小寫。

```
$ aws kms list-aliases --query 'Aliases[?AliasName==`alias/access-key`]'
[
  {
    "AliasName": "alias/access-key",
    "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/access-key",
    "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
    "CreationDate": "2018-01-20T15:23:10.194000-07:00",
    "LastUpdatedDate": "2018-01-20T15:23:10.194000-07:00"
  }
]
```

更新別名

由於別名是獨立的資源，因此您可以變更與別名相關聯的 KMS 金鑰。例如，如果 test-key 別名與一個 KMS 金鑰相關聯，則您可以使用 [UpdateAlias](#) 操作，將其與不同的 KMS 金鑰建立關聯。這是[手動輪換 KMS 金鑰](#)的幾種方式之一，不會變更其金鑰材料。您也可以更新 KMS 金鑰，以便針對新資源使用 KMS 金鑰的應用程式現在使用不同的 KMS 金鑰。

您無法在 AWS KMS 主控台中更新別名。您也無法使用 UpdateAlias (或任何其他操作) 來變更別名名稱。若要變更別名名稱，請刪除目前別名，然後為 KMS 金鑰建立新的別名。

更新別名時，當前 KMS 金鑰和新 KMS 金鑰必須為相同類型 (兩者皆為對稱或非對稱或 HMAC)。也必須有相同的金鑰用途 (ENCRYPT_DECRYPT 或 SIGN_VERIFY 或 GENERATE_VERIFY_MAC)。這項限制可防止使用別名的程式碼中出現密碼編譯錯誤。

下列範例透過使用 [ListAliases](#) 操作開始，以顯示 test-key 別名目前與 KMS 金鑰 1234abcd-12ab-34cd-56ef-1234567890ab 建立關聯。

```
$ aws kms list-aliases --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
{
  "Aliases": [
    {
      "AliasName": "alias/test-key",
      "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/test-key",
      "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
      "CreationDate": 1593622000.191,
      "LastUpdatedDate": 1593622000.191
    }
  ]
}
```

接下來，它使用 UpdateAlias 操作來將與 test-key 別名關聯的 KMS 金鑰變更為 KMS 金鑰 0987dcba-09fe-87dc-65ba-ab0987654321。您不需要指定目前關聯的 KMS 金鑰，只需指定新的 (「目標」) KMS 金鑰。別名名稱區分大小寫。

```
$ aws kms update-alias --alias-name 'alias/test-key' --target-key-id
0987dcba-09fe-87dc-65ba-ab0987654321
```

若要驗證別名是否已與目標 KMS 金鑰關聯，請再次使用 ListAliases 操作。此 AWS CLI 命令使用 --query 參數僅取得 test-key 別名。TargetKeyId 和 LastUpdatedDate 欄位會更新。

```
$ aws kms list-aliases --query 'Aliases[?AliasName==`alias/test-key`]'
[
  {
    "AliasName": "alias/test-key",
    "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/test-key",
    "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321",
    "CreationDate": 1593622000.191,
    "LastUpdatedDate": 1604958290.154
  }
]
```

刪除別名

您可以在 AWS KMS 主控台或使用 [DeleteAlias](#) 操作來刪除別名。在刪除別名前，請確認別名未處於使用中狀態。雖然刪除別名並不會影響相關聯的 KMS 金鑰，但它可能會對使用別名的任何應用程式造成問題。如果不小心刪除了別名，您可以建立具有相同名稱的新別名，並將其與相同或不同的 KMS 金鑰產生關聯。

如果刪除 KMS 金鑰，則會刪除與該 KMS 金鑰相關聯的所有別名。

使用 AWS KMS 主控台

若要刪除 AWS KMS 主控台內的別名，請使用 KMS 金鑰詳細資訊頁面上的別名索引標籤。您可以一次刪除 KMS 金鑰的多個別名。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。您無法管理 AWS 受管金鑰 或 的別名 AWS 擁有的金鑰。
4. 在資料表中，選擇 KMS 金鑰的金鑰 ID 或別名。然後，在 KMS 金鑰詳細資訊頁面上，選擇 Aliases (別名) 索引標籤。

如果 KMS 金鑰有多個別名，Aliases (別名) 資料行會顯示一個別名和別名摘要，例如 (+n 等)。選擇別名摘要會直接帶您前往 KMS 金鑰詳細資訊頁面上的 Aliases (別名) 索引標籤。

5. 在 Aliases (別名) 索引標籤上，選取您要刪除之別名旁邊的核取方塊。然後選擇刪除。

使用 AWS KMS API

若要刪除別名，請使用 [DeleteAlias](#) 操作。此操作一次會刪除一個別名。別名名稱區分大小寫，且前面必須加上 alias/ 字首。

例如，以下命令會刪除 test-key 別名。此命令不會傳回任何輸出。

```
$ aws kms delete-alias --alias-name alias/test-key
```

若要確認是否已刪除別名，請使用 [ListAliases](#) 操作。下列命令使用 中的 --query 參數 AWS CLI 來僅取得 test-key 別名。回應中的空括號表示 ListAliases 回應中未包含 test-key 別名。若要消除括號，請使用 --output text 參數和值。


```
$ aws kms list-aliases --query 'Aliases[?AliasName==`alias/test-key`]'  
[]
```

使用別名來控制對 KMS 金鑰的存取

您可以根據與 KMS 金鑰相關聯的別名來控制對 KMS 金鑰的存取。若要執行這項操作，請使用 [kms:RequestAlias](#) 和 [kms:ResourceAliases](#) 條件索引鍵。此功能是 [屬性型存取控制](#) (ABAC) AWS KMS 支援的一部分。

`kms:RequestAlias` 條件索引鍵允許或拒絕根據請求中的別名存取 KMS 金鑰。`kms:ResourceAliases` 條件索引鍵會根據與 KMS 金鑰相關聯的別名，允許或拒絕存取 KMS 金鑰。

這些功能不允許您使用政策陳述式中 `resource` 元素的別名來識別 KMS 金鑰。當別名是 `resource` 元素的值時，政策會套用至別名資源，而不會套用至可能與其相關聯的任何 KMS 金鑰。

Note

可能最多需要五分鐘才能將標籤和別名變更體現在 KMS 金鑰授權上。最近的變更可能會在 API 操作中可見，然後才會影響授權。

使用別名控制 KMS 金鑰的存取時，請考慮下列事項：

- 使用別名來強化 [最低權限存取](#) 的最佳實務。僅為 IAM 委託人提供他們必須使用或管理之 KMS 金鑰所需的許可。例如，使用別名來識別專案所使用的 KMS 金鑰。然後授予專案小組僅將 KMS 金鑰與專案別名搭配使用的許可。
- 要謹慎地授予委託人 `kms:CreateAlias`、`kms:UpdateAlias` 或 `kms>DeleteAlias` 許可，讓其新增、編輯和刪除別名。當您使用別名來控制對 KMS 金鑰的存取時，變更別名可授予委託人使用 KMS 金鑰 (否則其沒有使用的許可) 的許可。它也可以拒絕存取其他委託人執行任務所需的 KMS 金鑰。
- 檢閱 中目前具有管理別名許可 AWS 帳戶 的主體，並視需要調整許可。沒有變更主要政策或建立授權之許可的重要管理員可以控制 KMS 金鑰的存取，如果他們擁有管理別名的許可。

例如，重要管理員的 [主控台預設重要政策](#) 會包括 `kms:CreateAlias`、`kms>DeleteAlias` 和 `kms:UpdateAlias` 許可。IAM 政策可能會為您的 AWS 帳戶中的所有 KMS 金鑰提供別名許可。例如，[AWSKeyManagementServicePowerUser](#) 受管政策允許委託人建立、刪除和列出所有 KMS 金鑰的別名，但不予以更新。

- 在設定依賴別名的政策之前，請檢閱您 中 KMS 金鑰上的別名 AWS 帳戶。請確定您的政策僅適用於您想要包含的別名。使用 [CloudTrail 日誌](#) 和 [CloudWatch 警示](#)，提醒您可能會影響 KMS 金鑰存取的別名變更。此外，[ListAliases](#) 回應包含每個別名的建立日期和上次更新日期。
- 別名政策條件使用模式比對；其不會繫結至別名的特定執行個體。使用別名型條件索引鍵的政策會影響所有符合模式的新別名和現有別名。如果您刪除並重新建立符合政策條件的別名，則條件會套用至新別名，就像舊別名一樣。

`kms:RequestAlias` 條件索引鍵依賴於操作請求中明確指定的別名。`kms:ResourceAliases` 條件索引鍵取決於與 KMS 金鑰相關聯的別名，即使其並沒有出現在請求中。

kms:RequestAlias

根據識別請求中 KMS 金鑰的別名，允許或拒絕存取 KMS 金鑰。您可以使用 [金鑰政策](#) 或 IAM 政策中的 `kms:RequestAlias` 條件索引鍵。它適用於使用別名來識別請求中 KMS 金鑰的操作，也就是 [密碼編譯操作](#)、[DescribeKey](#)，以及 [GetPublicKey](#)。其對別名操作無效，例如 [CreateAlias](#) 或 [DeleteAlias](#)。

Important

不建議使用具有 `kms:RequestAlias` 條件 Deny 的政策陳述式，因為發起人可以使用 [金鑰 ID](#)、[金鑰 ARN](#) 或不同的別名來識別請求中的 KMS 金鑰。若要根據與 KMS 金鑰相關聯的別名拒絕存取，請改用 `kms:ResourceAliases` 條件金鑰。

在條件索引鍵中，指定 [別名名稱](#) 或別名名稱模式。您不能指定 [別名 ARN](#)。

例如，下列金鑰政策陳述式可讓委託人對 KMS 金鑰使用指定的操作。僅在請求使用包含 `alpha` 來識別 KMS 金鑰的別名時，許可才有效。

```
{
  "Sid": "Key policy using a request alias condition",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/alpha-developer"
  },
  "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey*",
    "kms:DescribeKey"
  ],
```

```

"Resource": "*",
"Condition": {
  "StringLike": {
    "kms:RequestAlias": "alias/*alpha*"
  }
}
}

```

下列授權委託人提出的範例請求將滿足條件。不過，使用[金鑰 ID](#) 的請求、[金鑰 ARN](#)，或者不同的別名將無法滿足條件，即使這些值識別了相同的 KMS 索引鍵。

```

$ aws kms describe-key --key-id "arn:aws:kms:us-west-2:111122223333:alias/project-alpha"

```

kms:ResourceAliases

根據與 KMS 金鑰相關聯的別名允許或拒絕存取 KMS 金鑰，即使請求中未使用別名。[kms:ResourceAliases](#) 條件索引鍵可讓您指定別名或別名模式，例如 `alias/test*`，因此您可以在 IAM 政策中使用其來控制對相同區域中數個 KMS 金鑰的存取。它適用於使用 KMS 金鑰的任何 AWS KMS 操作。

例如，下列 IAM 政策可讓委託人在 KMS 金鑰上呼叫指定的操作。AWS 帳戶不過，此許可僅適用於與開頭為 `restricted` 之別名關聯的 KMS 金鑰。

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AliasBasedIAMPolicy",
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": [
        "arn:aws:kms:*:111122223333:key/*",
        "arn:aws:kms:*:444455556666:key/*"
      ]
    }
  ]
}

```

```
    ],
    "Condition": {
      "ForAnyValue:StringLike": {
        "kms:ResourceAliases": "alias/restricted*"
      }
    }
  }
}
```

`kms:ResourceAliases` 條件是資源的條件，而不是請求。因此，未指定別名的請求仍然可以滿足條件。

下列範例請求 (指定相符別名) 會滿足條件。

```
$ aws kms enable-key-rotation --key-id "alias/restricted-project"
```

不過，下列範例請求也符合條件，前提是指定的 KMS 金鑰具有以 `restricted` 開頭的別名，即使別名未在請求中使用。

```
$ aws kms enable-key-rotation --key-id "1234abcd-12ab-34cd-56ef-1234567890ab"
```

了解如何在應用程式中使用別名

您可以使用別名來代表應用程式程式碼中的 KMS 金鑰。KeyId 參數 AWS KMS [加密操作 DescribeKey](#) 和 [GetPublicKey](#) 接受別名名稱或別名 ARN。

例如，下列 `GenerateDataKey` 命令會使用別名名稱 (`alias/finance`) 來識別 KMS 金鑰。別名名稱為 `KeyId` 參數的值。

```
$ aws kms generate-data-key --key-id alias/finance --key-spec AES_256
```

如果 KMS 金鑰位於不同的 AWS 帳戶，您必須在這些操作中使用金鑰 ARN 或別名 ARN。使用別名 ARN 時，請記住 KMS 金鑰的別名是在擁有 KMS 金鑰的帳戶中定義的，且在每個區域中可能會有所不同。如需尋找別名 ARN 的說明，請參閱 [尋找 KMS 金鑰的別名名稱和別名 ARN](#)。

例如，下列 `GenerateDataKey` 命令會使用不位於呼叫者帳戶中的 KMS 金鑰。ExampleAlias 別名與指定帳戶和區域中的 KMS 金鑰相關聯。

```
$ aws kms generate-data-key --key-id arn:aws:kms:us-west-2:444455556666:alias/  
ExampleAlias --key-spec AES_256
```

其中一個最強大的別名用途是在多個 AWS 區域中執行之應用程式中使用。例如，您可能有使用 RSA [非對稱 KMS 金鑰](#)的全域應用程式，以進行簽章和驗證。

- 在美國西部 (奧勒岡) (us-west-2) 中，您想要使用 `arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab`。
- 在歐洲 (法蘭克福) (eu-central-1) 中，您想要使用 `arn:aws:kms:eu-central-1:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321`
- 在亞太區域 (新加坡) (ap-southeast-1) 中，您想要使用 `arn:aws:kms:ap-southeast-1:111122223333:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d`。

您可以在每個區域中建立不同版本的應用程式，或使用字典或 switch 陳述式為每個區域選取正確的 KMS 金鑰。但是，更輕鬆的方式是在每個區域中建立具有相同別名名稱的別名。請記住，別名名稱區分大小寫。

```
aws --region us-west-2 kms create-alias \  
    --alias-name alias/new-app \  
    --key-id arn:aws:kms:us-  
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab  
  
aws --region eu-central-1 kms create-alias \  
    --alias-name alias/new-app \  
    --key-id arn:aws:kms:eu-central-1:111122223333:key/0987dcba-09fe-87dc-65ba-  
ab0987654321  
  
aws --region ap-southeast-1 kms create-alias \  
    --alias-name alias/new-app \  
    --key-id arn:aws:kms:ap-  
southeast-1:111122223333:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d
```

然後，在程式碼中使用別名。當程式碼在每個區域中執行時，別名會參照其在該區域中關聯的 KMS 金鑰。例如，此程式碼會使用別名名稱呼叫 [Sign](#) 操作。

```
aws kms sign --key-id alias/new-app \  
    --message $message \  
    --message-type RAW \  
    --signing-algorithm RSASSA_PSS_SHA_384
```

不過，風險是有可能會刪除或更新別名，以與不同的 KMS 金鑰相關聯。在這種情況下，應用程式嘗試使用別名名稱驗證簽章將會失敗，您可能需要重新建立或更新別名。

若要降低此風險，請謹慎授予委託人管理您在應用程式中使用之別名的許可。如需詳細資訊，請參閱[控制對別名的存取](#)。

對於在多個 AWS 區域中加密資料的應用程式，還有幾個其他解決方案，包括 [AWS Encryption SDK](#)。

在 AWS CloudTrail 日誌中尋找別名

您可以使用別名來代表 AWS KMS API 操作 AWS KMS key 中的。當您這麼做時，KMS 金鑰的別名和金鑰 ARN 會記錄在事件的 AWS CloudTrail 日誌項目中。別名會顯示在 `requestParameters` 欄位。金鑰 ARN 會顯示在 `resources` 欄位。即使 AWS 服務在您的帳戶 AWS 受管金鑰中使用，也是如此。

例如，下列 [GenerateDataKey](#) 請求會使用 `project-key` 別名來代表 KMS 金鑰。

```
$ aws kms generate-data-key --key-id alias/project-key --key-spec AES_256
```

當此請求記錄在 CloudTrail 日誌中時，日誌項目會包含實際使用 KMS 金鑰的別名和金鑰 ARN。

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "ABCDE",
    "arn": "arn:aws:iam::111122223333:role/ProjectDev",
    "accountId": "111122223333",
    "accessKeyId": "FFHIJ",
    "userName": "example-dev"
  },
  "eventTime": "2020-06-29T23:36:41Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GenerateDataKey",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "205.205.123.000",
  "userAgent": "aws-cli/1.18.89 Python/3.6.10
Linux/4.9.217-0.1.ac.205.84.332.metal1.x86_64 boto3/1.17.12",
  "requestParameters": {
    "keyId": "alias/project-key",
    "keySpec": "AES_256"
  }
}
```

```
    },
    "responseElements": null,
    "requestID": "d93f57f5-d4c5-4bab-8139-5a1f7824a363",
    "eventID": "d63001e2-dbc6-4aae-90cb-e5370aca7125",
    "readOnly": true,
    "resources": [
      {
        "accountId": "111122223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-  
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
  }
}
```

如需有關在 CloudTrail 日誌中記錄 AWS KMS 操作的詳細資訊，請參閱 [使用記錄 AWS KMS API 呼叫 AWS CloudTrail](#)。

中的標籤 AWS KMS

標籤是您可以指派（或 AWS 可指派）給 AWS 資源的選用中繼資料標籤。每個標籤皆包含標籤索引鍵和標籤值，它們都是區分大小寫的字串。此標籤值可以是空 (null) 字串。資源上的每個標籤都必須有不同的標籤索引鍵，但您可以將相同的標籤新增至多個 AWS 資源。每個資源最多可以有 50 個使用者建立的標籤。

請勿在標籤金鑰或標籤值包含機密或敏感資訊。許多 都可以存取標籤 AWS 服務，包括帳單。

在 中 AWS KMS，您可以在建立 KMS 金鑰時將標籤新增至客戶受管金鑰，並標記或取消標記現有的 KMS 金鑰，除非它們正在 [等待刪除](#)。您無法在其他 中標記別名 AWS 受管金鑰AWS 擁有的金鑰、自訂金鑰存放區或 KMS 金鑰 AWS 帳戶。標籤是選用的，但它們可以非常有用。

例如，您可以將 "Project"="Alpha" 標籤新增至您用於 Alpha 專案的所有 KMS 金鑰和 Amazon S3 儲存貯體。

```
TagKey    = "Project"
TagValue  = "Alpha"
```

如需標籤的一般資訊，包括格式和語法，請參閱《》中的 [標記 AWS 資源](#) Amazon Web Services 一般參考。

標籤可協助您執行以下操作：

- 識別和組織您的 AWS 資源。許多 AWS 服務支援標記，因此您可以將相同的標籤指派給來自不同服務的資源，以指出資源相關。例如，您可以將相同的標籤指派給 KMS 金鑰和 Amazon Elastic Block Store (Amazon EBS) 磁碟區或 AWS Secrets Manager 秘密。您也可以使用標籤來識別 KMS 金鑰以進行自動化。
- 追蹤您的 AWS 成本。當您將標籤新增至 AWS 資源時，AWS 會產生成本分配報告，其中包含依標籤彙總的用量和成本。您可以使用此功能來追蹤專案、應用程式或成本中心 AWS KMS 的成本。

如需有關使用成本配置標籤的詳細資訊，請參閱《AWS Billing 使用者指南》中的 [使用成本分配標籤](#)。如需標籤鍵和標籤值規則的相關資訊，請參閱《AWS Billing 使用者指南》中的 [使用者定義的標籤限制](#)。

- 控制對 AWS 資源的存取。允許和拒絕根據 KMS 金鑰的標籤存取，是 [屬性型存取控制](#) (ABAC) AWS KMS 支援的一部分。如需 AWS KMS keys 根據標籤控制對存取的資訊，請參閱 [使用標籤來控制對 KMS 金鑰的存取](#)。如需使用標籤控制 AWS 資源存取的一般資訊，請參閱《IAM 使用者指南》中的 [使用資源標籤控制 AWS 資源存取](#)。

AWS KMS 當您使用 [TagResource](#)、[UntagResource](#) 或 [ListResourceTags](#) 操作時，會將項目寫入您的 AWS CloudTrail 日誌。

主題

- [控制對標籤的存取](#)
- [將標籤新增至 KMS 金鑰](#)
- [編輯與 KMS 金鑰相關聯的標籤](#)
- [移除與 KMS 金鑰相關聯的標籤](#)
- [檢視與 KMS 金鑰相關聯的標籤](#)
- [使用標籤來控制對 KMS 金鑰的存取](#)

控制對標籤的存取

若要在 AWS KMS 主控台或使用 API 新增、檢視和刪除標籤，委託人需要標記許可。您可以在[金鑰政策](#)中提供這些許可。您也可以 IAM 政策 (包括 [VPC 端點政策](#)) 中提供，但僅當[金鑰政策允許](#)時。[AWSKeyManagementServicePowerUser](#) 受管政策允許主體在帳戶可存取的所有 KMS 金鑰上標記、取消標記和列出標籤。

您也可以使用標籤的 AWS 全域條件索引鍵來限制這些許可。在 AWS KMS 中，這些條件可以控制對標記操作的存取，例如 [TagResource](#) 和 [UntagResource](#)。

Note

授予主體管理標籤和別名的許可時，請務必謹慎。變更標記或別名可允許或拒絕客戶受管金鑰的許可。如需詳細資訊，請參閱 [適用於的 ABAC AWS KMS](#) 和 [使用標籤來控制對 KMS 金鑰的存取](#)。

如需政策和詳細資訊，請參閱《IAM 使用者指南》中的[根據標籤索引鍵控制存取](#)。

建立和管理標籤的許可如下所示。

kms:TagResource

允許主體新增或編輯標籤。若要在建立 KMS 金鑰時新增標籤，主體必須在 IAM 政策中具有不限於特定 KMS 金鑰的許可。

kms:ListResourceTags

允許主體檢視 KMS 金鑰上的標籤。

kms:UntagResource

允許主體從 KMS 金鑰刪除標籤。

標記政策中的許可

您可以在金鑰政策或 IAM 政策中提供標記許可。例如，以下範例金鑰政策會給予精選使用者標記 KMS 金鑰的許可。它為所有可以擔任範例管理員或開發人員角色的使用者提供檢視標籤的許可。

JSON

```
{
  "Version": "2012-10-17",
  "Id": "example-key-policy",
  "Statement": [
    {
      "Sid": "EnableIAMUserPermissions",
      "Effect": "Allow",
      "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "AllowAllTaggingPermissions",
      "Effect": "Allow",
      "Principal": {"AWS": [
        "arn:aws:iam::111122223333:user/LeadAdmin",
        "arn:aws:iam::111122223333:user/SupportLead"
      ]},
      "Action": [
        "kms:TagResource",
        "kms:ListResourceTags",
        "kms:UntagResource"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowRolesViewTags",
```

```

    "Effect": "Allow",
    "Principal": {"AWS": [
        "arn:aws:iam::111122223333:role/Administrator",
        "arn:aws:iam::111122223333:role/Developer"
    ]},
    "Action": "kms:ListResourceTags",
    "Resource": "*"
  }
]
}

```

若要授予主體在多個 KMS 金鑰上的標記許可，您可以使用 IAM 政策。若要讓此政策生效，每個 KMS 金鑰的金鑰政策必須允許帳戶使用 IAM 政策來控制對 KMS 金鑰的存取。

例如，以下 IAM 政策允許主體建立 KMS 金鑰。它也允許其在指定帳戶中的所有 KMS 金鑰上建立和管理標籤。這種組合允許主體使用 [CreateKey](#) 操作的 [標籤](#) 參數在建立 KMS 金鑰時新增標籤。

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPolicyCreateKeys",
      "Effect": "Allow",
      "Action": "kms:CreateKey",
      "Resource": "*"
    },
    {
      "Sid": "IAMPolicyTags",
      "Effect": "Allow",
      "Action": [
        "kms:TagResource",
        "kms:UntagResource",
        "kms:ListResourceTags"
      ],
      "Resource": "arn:aws:kms:*:111122223333:key/*"
    }
  ]
}

```

限制標籤許可

您可以使用[政策條件](#)來限制標記許可。下列政策條件可套用至 `kms:TagResource` 和 `kms:UntagResource` 許可。例如，您可以使用 `aws:RequestTag/tag-key` 條件，允許主體僅新增特定標籤，或防止主體新增具有特定標籤索引鍵的標籤。或者，您可以使用 `kms:KeyOrigin` 條件，以防止主體使用[匯入金鑰材料](#)標記或取消標記 KMS 金鑰。

- [aws:RequestTag](#)
- [aws:ResourceTag/tag-key](#) (僅限 IAM 政策)
- [aws:TagKeys](#)
- [kms:CallerAccount](#)
- [kms:KeySpec](#)
- [kms:KeyUsage](#)
- [kms:KeyOrigin](#)
- [kms:ViaService](#)

最佳實務的做法是，當您使用標籤來控制 KMS 金鑰的存取時，請使用 `aws:RequestTag/tag-key` 或 `aws:TagKeys` 條件鍵來確定允許哪些標籤 (或標籤索引鍵)。

例如，下列 IAM 政策與前一個類似。不過，此政策允許主體建立標籤 (`TagResource`) 並僅為具有 `Project` 標籤索引鍵的標籤刪除標籤 (`UntagResource`)。

由於 `TagResource` 和 `UntagResource` 請求可以包含多個標籤，您必須指定具有 [aws:TagKeys](#) 條件的 `ForAllValues` 或 `ForAnyValue` 集合運算子。`ForAnyValue` 運算子會要求請求中的至少一個標籤索引鍵與政策中的標籤索引鍵相符。`ForAllValues` 運算子會要求請求中的所有標籤索引鍵與政策中的其中一個標籤索引鍵相符。`ForAllValues` 運算子也會傳回 `true`，如果在請求中沒有標籤，但 `TagResource` 和 `UntagResource` 會在未指定標籤時失敗。如需集合運算子的詳細資訊，請參閱《IAM 使用者指南》中的[使用多個索引鍵和值](#)。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPolicyCreateKey",
      "Effect": "Allow",
```

```

    "Action": "kms:CreateKey",
    "Resource": "*"
  },
  {
    "Sid": "IAMPolicyViewAllTags",
    "Effect": "Allow",
    "Action": "kms:ListResourceTags",
    "Resource": "arn:aws:kms:*:111122223333:key/*"
  },
  {
    "Sid": "IAMPolicyManageTags",
    "Effect": "Allow",
    "Action": [
      "kms:TagResource",
      "kms:UntagResource"
    ],
    "Resource": "arn:aws:kms:*:111122223333:key/*",
    "Condition": {
      "ForAllValues:StringEquals": {"aws:TagKeys": "Project"}
    }
  }
]
}

```

將標籤新增至 KMS 金鑰

標籤有助於識別和組織您的 AWS 資源。您可以在[建立 KMS 金鑰時將標籤新增至客戶受管金鑰](#)，或將標籤新增至現有的 KMS 金鑰。您無法標記 AWS 受管金鑰。

下列程序示範如何使用 主控台和 AWS KMS API 將 AWS KMS 標籤新增至客戶受管金鑰。AWS KMS API 範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

主題

- [建立 KMS 金鑰時新增標籤](#)
- [將標籤新增至現有的 KMS 金鑰](#)

建立 KMS 金鑰時新增標籤

當您使用 AWS KMS 主控台或 [CreateKey](#) 操作建立金鑰時，您可以將標籤新增至 KMS 金鑰。若要在建立 KMS 金鑰時新增標籤，除了建立 KMS 金鑰所需的 `kms:TagResource` 許可之外，您還必須在

IAM 政策中擁有許可。此許可至少必須涵蓋帳戶和區域中的所有 KMS 金鑰。如需詳細資訊，請參閱[控制對標籤的存取](#)。

使用 AWS KMS 主控台

若要在主控台中建立 KMS 金鑰時新增標籤，除了標記和建立 KMS 金鑰所需的許可之外，您還必須具有在主控台中檢視 KMS 金鑰所需的許可。此許可至少必須涵蓋帳戶和區域中的所有 KMS 金鑰。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。(您無法管理 AWS 受管金鑰的標籤)
4. 選擇金鑰類型，然後選擇 Next (下一頁)。
5. 輸入別名和選用描述。
6. 輸入標籤索引鍵和選用的標籤值。若要新增其他標籤，請選擇 Add tag (新增標籤)。若要移除標籤，請選擇 Remove (移除)。標記完新的 KMS 金鑰後，選擇 Next (下一頁)。
7. 完成建立 KMS 金鑰。

使用 AWS KMS API

若要在使用 [CreateKey](#) 操作建立金鑰時指定標籤，請使用 操作的 Tags 參數。

CreateKey 的 Tags 參數值是區分大小寫的標籤鍵和標籤值對的集合。KMS 金鑰上的每個標籤必須有不同的標籤名稱。標籤值可以為 null 或空字串。

例如，下列 AWS CLI 命令會使用 Project:Alpha 標籤建立對稱加密 KMS 金鑰。指定多個索引鍵/值組時，請使用空格來分隔每一組。

```
$ aws kms create-key --tags TagKey=Project,TagValue=Alpha
```

此命令成功時，會傳回包含新 KMS 金鑰相關資訊的 KeyMetadata 物件。但是，KeyMetadata 不包含標籤。若要取得標籤，請使用 [ListResourceTags](#) 操作。

將標籤新增至現有的 KMS 金鑰

您可以在 AWS KMS 主控台或使用 [TagResource](#) 操作，將標籤新增至現有的客戶受管 KMS 金鑰。若要新增標籤，您需要在 KMS 金鑰上標記許可。您可以從 KMS 金鑰的金鑰政策取得此許可，或者如果金鑰政策允許，則從包含 KMS 金鑰的 IAM 政策取得此許可。

使用 AWS KMS 主控台

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。(您無法管理 AWS 受管金鑰的標籤)
4. 您可以使用資料表篩選器，只顯示具有特定標籤的 KMS 金鑰。如需詳細資訊，請參閱[使用 AWS KMS 主控台檢視標籤](#)。
5. 選取 KMS 金鑰別名旁的核取方塊。
6. 選擇 Key actions (金鑰動作)、Add or edit tags (新增或編輯標籤)。
7. 在 KMS 金鑰的詳細資訊頁面上，選擇 Tags (標籤) 索引標籤。
 - 若要建立您的第一個標籤，請選擇 Create tag (建立標籤)，輸入標籤索引鍵 (必要) 和標籤值 (選用)，然後選擇 Save (儲存)。

如果您將標籤值保留空白，則實際標籤值為 null 或空字串。
 - 若要新增標籤，請選擇 Edit (編輯)，選擇 Add tag (新增標籤)，輸入標籤索引鍵和標籤值，然後選擇 Save (儲存)。
8. 若要儲存您所做的變更，請選擇 Save changes (儲存變更)。

使用 AWS KMS API

[TagResource](#) 操作會新增一或多個標籤至 KMS 金鑰。您無法使用此操作在不同的 中新增標籤 AWS 帳戶。您也可以使用 TagResource 操作來編輯現有的標籤。如需詳細資訊，請參閱[the section called “編輯標籤”](#)。

若要新增標籤，請指定新標籤索引鍵和標籤值。KMS 金鑰上的每個標籤必須有不同的標籤索引鍵。標籤值可以為 null 或空字串。

例如，下列命令會將 **Purpose** 和 **Department** 標籤新增至範例 KMS 金鑰。

```
$ aws kms tag-resource \  
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
    --tags TagKey=Purpose,TagValue=Pretest TagKey=Department,TagValue=Finance
```

當此命令成功時，不會傳回任何輸出。若要檢視 KMS 金鑰上的標籤，請使用 [ListResourceTags](#) 操作。

編輯與 KMS 金鑰相關聯的標籤

標籤有助於識別和組織您的 AWS 資源。您可以在 AWS KMS 主控台或使用 [TagResource](#) 操作，編輯與客戶受管 KMS 金鑰相關聯的標籤。您無法編輯的標籤 AWS 受管金鑰。

下列程序示範如何編輯與 KMS 金鑰相關聯的標籤。AWS KMS API 範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

使用 AWS KMS 主控台

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。(您無法編輯的標籤 AWS 受管金鑰)
4. 您可以使用資料表篩選器，只顯示具有特定標籤的 KMS 金鑰。如需詳細資訊，請參閱[使用 AWS KMS 主控台檢視標籤](#)。
5. 選取 KMS 金鑰別名旁的核取方塊。
6. 選擇 Key actions (金鑰動作)、Add or edit tags (新增或編輯標籤)。
7. 在 KMS 金鑰的詳細資訊頁面上，選擇 Tags (標籤) 索引標籤。
 - 若要變更標籤的名稱或值，請選擇 Edit (編輯)，完成您要的變更，然後選擇 Save (儲存)。
8. 若要儲存您所做的變更，請選擇 Save changes (儲存變更)。

使用 AWS KMS API

[TagResource](#) 操作會將一或多個標籤新增至客戶受管金鑰；不過，您也可以使用 TagResource 來變更現有標籤的標籤值。您無法使用此操作新增或編輯不同 AWS 帳戶中的標籤。

若要編輯標籤，請指定現有標籤索引鍵和新標籤值。KMS 金鑰上的每個標籤必須有不同的標籤索引鍵。標籤值可以為 null 或空字串。

例如，這個命令會將 Purpose 標籤的值從 Pretest 變更為 Test。

```
$ aws kms tag-resource \  
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
    --tags TagKey=Purpose,TagValue=Test
```

移除與 KMS 金鑰相關聯的標籤

標籤有助於識別和組織您的 AWS 資源。您可以在 AWS KMS 主控台或使用 [UntagResource](#) 操作，移除與客戶受管 KMS 金鑰相關聯的標籤。您無法編輯或移除的標籤 AWS 受管金鑰。

下列程序示範如何從 KMS 金鑰移除標籤。AWS KMS API 範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

使用 AWS KMS 主控台

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。(您無法管理 AWS 受管金鑰的標籤)
4. 您可以使用資料表篩選器，只顯示具有特定標籤的 KMS 金鑰。如需詳細資訊，請參閱[使用 AWS KMS 主控台檢視標籤](#)。
5. 選取 KMS 金鑰別名旁的核取方塊。
6. 選擇 Key actions (金鑰動作)、Add or edit tags (新增或編輯標籤)。
7. 在 KMS 金鑰的詳細資訊頁面上，選擇 Tags (標籤) 索引標籤。
 - 若要刪除標籤，請選擇 Edit (編輯)。在標籤列，選擇 Remove (移除)，然後選擇 Save (儲存)。
8. 若要儲存您所做的變更，請選擇 Save changes (儲存變更)。

使用 AWS KMS API

[UntagResource](#) 操作會從 KMS 金鑰刪除標籤。若要識別要刪除的標籤，請指定標籤索引鍵。您無法使用此操作刪除不同 AWS 帳戶中 KMS 金鑰的標籤。

成功時，`UntagResource` 操作不會傳回任何輸出。此外，如果在 KMS 金鑰上找不到指定的標籤金鑰，則其不會擲回例外狀況或傳回回應。若要確認操作已運作，請使用 [ListResourceTags](#) 操作。

例如，此命令會從指定的 KMS 金鑰刪除 **Purpose** 標籤及其值。

```
$ aws kms untag-resource --key-id 1234abcd-12ab-34cd-56ef-1234567890ab --tag-keys Purpose
```

檢視與 KMS 金鑰相關聯的標籤

標籤有助於識別和組織您的 AWS 資源。您可以在 AWS KMS 主控台或使用 [ListResourceTags](#) 操作，檢視與客戶受管 KMS 金鑰相關聯的標籤。

下列程序示範如何尋找與特定 KMS 金鑰相關聯的標籤。AWS KMS API 範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

使用 AWS KMS 主控台

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。(您無法管理 AWS 受管金鑰的標籤)
4. 您可以使用資料表篩選器，只顯示具有特定標籤的 KMS 金鑰。

若只要顯示具有特定標籤的 KMS 金鑰，請選擇篩選方塊、選擇標籤索引鍵，然後從實際標籤值中選擇。您也可以輸入全部或部分的標籤值。

產生的資料表會顯示具有所選標籤的所有 KMS 金鑰。但不會顯示標籤。若要查看標籤，請選擇 KMS 金鑰的金鑰 ID 或別名，然後在其詳細資訊頁面上選擇 Tags (標籤) 索引標籤。索引標籤會顯示在 General Configuration (一般組態) 區段下。

此篩選條件需要標籤索引鍵和標籤值。只輸入標籤索引鍵或只輸入其值，就無法找到 KMS 金鑰。若要依標籤索引鍵或值的全部或部分篩選標籤，請使用 [ListResourceTags](#) 操作來取得標記的 KMS 金鑰，然後使用程式設計語言的篩選功能。

5. 選取 KMS 金鑰別名旁的核取方塊。
6. 選擇 Key actions (金鑰動作)、Add or edit tags (新增或編輯標籤)。
7. 在 KMS 金鑰的詳細資訊頁面上，選擇 Tags (標籤) 索引標籤。

使用 AWS KMS API

[ListResourceTags](#) 操作會取得 KMS 金鑰的標籤。KeyId 參數是必要參數。您無法使用此操作檢視不同 AWS 帳戶中 KMS 金鑰的標籤。

例如，下列命令會取得範例 KMS 金鑰的標籤。

```
$ aws kms list-resource-tags --key-id 1234abcd-12ab-34cd-56ef-1234567890ab

{
  "Truncated": false,
  "Tags": [
    {
      "TagKey": "Project",
      "TagValue": "Alpha"
    },
    {
      "TagKey": "Purpose",
      "TagValue": "Test"
    },
    {
      "TagKey": "Department",
      "TagValue": "Finance"
    }
  ]
}
```

使用標籤來控制對 KMS 金鑰的存取

您可以 AWS KMS keys 根據 KMS 金鑰上的標籤控制對的存取。例如，您可以撰寫 IAM 政策，允許主體僅啟用和停用具有特定標籤的 KMS 金鑰。或者，您可以使用 IAM 政策來防止主體在密碼編譯操作中使用 KMS 金鑰，除非 KMS 金鑰有特定的標籤。

此功能是[屬性型存取控制](#) (ABAC) AWS KMS 支援的一部分。如需有關使用標籤控制資源 AWS 存取的資訊，請參閱 [《IAM 使用者指南》中的 ABAC 是什麼 AWS？](#) 以及 [使用資源標籤控制 AWS 資源存取](#)。如需解決與 ABAC 相關之存取問題的說明，請參閱 [對適用於的 ABAC 進行故障診斷 AWS KMS](#)。

Note

可能最多需要五分鐘才能將標籤和別名變更體現在 KMS 金鑰授權上。最近的變更可能會在 API 操作中可見，然後才會影響授權。

AWS KMS 支援 [aws : ResourceTag/tag-key 全域條件內容金鑰](#)，可讓您根據 KMS 金鑰上的標籤控制對 KMS 金鑰的存取。由於多個 KMS 金鑰可以具有相同的標籤，所以此功能可讓您將許可套用至一組精選 KMS 金鑰。您也可以透過變更其標籤，輕鬆變更集合中的 KMS 金鑰。

在 中 AWS KMS，只有在 IAM 政策中支援 `aws:ResourceTag/tag-key` 條件金鑰。只適用於單一 KMS 金鑰的金鑰政策，或不使用特定 KMS 金鑰的操作 (例如 [ListKeys](#) 或 [ListAliases](#) 操作) 皆不支援此功能。

使用標籤控制存取可提供一種簡單、可擴展且靈活的方式來管理許可。不過，如果沒有正確設計和管理，它可能會意外允許或拒絕存取您的 KMS 金鑰。如果您使用標籤來控制存取，請考慮下列實務。

- 使用標籤來強化[最低權限存取](#)的最佳實務。僅為 IAM 主體提供他們必須使用或管理之 KMS 金鑰所需的許可。例如，使用標籤來標註專案所使用的 KMS 金鑰。然後授予專案小組僅將 KMS 金鑰與專案標籤搭配使用的許可。
- 要謹慎地授予主體 `kms:TagResource` 和 `kms:UntagResource` 許可，讓其新增、編輯和刪除別名。當您使用標籤來控制對 KMS 金鑰的存取時，變更標籤可授予主體使用 KMS 金鑰 (否則其沒有使用的許可) 的許可。它也可以拒絕存取其他主體執行任務所需的 KMS 金鑰。如果他們有管理標籤的許可，則沒有變更主要政策或建立授予之許可的金鑰管理員可以控制對 KMS 金鑰的存取。

如果可能，請使用政策條件，例如 `aws:RequestTag/tag-key` 或 `aws:TagKeys`，將[主體的標記許可限制](#)為特定 KMS 金鑰上的特定標籤或標籤模式。

- 檢閱 中目前具有標記和取消標記許可 AWS 帳戶的主體，並視需要進行調整。例如，主控台[金鑰管理員的預設金鑰政策](#)包含該 KMS 金鑰的 `kms:TagResource` 和 `kms:UntagResource` 許可。IAM 政策可能允許所有 KMS 金鑰的標記和取消標記許可。例如，[AWSKeyManagementServicePowerUser](#) 受管政策允許主體在所有 KMS 金鑰上標記、取消標記和列出標籤。
- 在設定取決於標籤的政策之前，請檢閱 中 KMS 金鑰上的標籤 AWS 帳戶。請確定您的政策僅適用於您想要包含的標籤。使用 [CloudTrail 日誌](#) 和 [CloudWatch 警示](#)，提醒您可能會影響 KMS 金鑰存取的標籤變更。
- 標籤型政策條件使用模式比對；其不會繫結至標籤的特定執行個體。使用標籤型條件索引鍵的政策會影響所有符合模式的新標籤和現有標籤。如果您刪除並重新建立符合政策條件的標籤，則條件會套用到新標籤，就像舊標籤一樣。

例如，請考慮以下 IAM 政策。它允許主體會僅針對亞太區域 (新加坡) 區域和擁有 "Project"="Alpha" 標籤之帳戶中的 KMS 金鑰呼叫 [GenerateDataKeyWithoutPlaintext](#) 和 [Decrypt](#) 操作。您可以將此政策連接至 Alpha 專案範例中的角色。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPolicyWithResourceTag",
      "Effect": "Allow",
      "Action": [
        "kms:GenerateDataKeyWithoutPlaintext",
        "kms:Decrypt"
      ],
      "Resource": "arn:aws:kms:ap-southeast-1:111122223333:key/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Project": "Alpha"
        }
      }
    }
  ]
}
```

以下 IAM 政策範例允許主體使用帳戶中的任何 KMS 金鑰來進行密碼編譯操作。但其禁止主體針對具有 "Type"="Reserved" 標籤或不具有 "Type" 標籤的 KMS 金鑰使用這些密碼編譯操作。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMAllowCryptographicOperations",
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:GenerateDataKey*",
        "kms:Decrypt",
        "kms:ReEncrypt*"
      ],
      "Resource": "arn:aws:kms:*:111122223333:key/*"
    }
  ]
}
```

```

    },
    {
      "Sid": "IAMDenyOnTag",
      "Effect": "Deny",
      "Action": [
        "kms:Encrypt",
        "kms:GenerateDataKey*",
        "kms:Decrypt",
        "kms:ReEncrypt*"
      ],
      "Resource": "arn:aws:kms:*:111122223333:key/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Type": "Reserved"
        }
      }
    },
    {
      "Sid": "IAMDenyNoTag",
      "Effect": "Deny",
      "Action": [
        "kms:Encrypt",
        "kms:GenerateDataKey*",
        "kms:Decrypt",
        "kms:ReEncrypt*"
      ],
      "Resource": "arn:aws:kms:*:111122223333:key/*",
      "Condition": {
        "Null": {
          "aws:ResourceTag/Type": "true"
        }
      }
    }
  ]
}

```

金鑰存放區

金鑰存放區是存放和使用密碼編譯金鑰的安全位置。中的預設金鑰存放區 AWS KMS 也支援產生和管理其存放金鑰的方法。根據預設，AWS KMS keys 您在 中建立之 的密碼編譯金鑰材料 AWS KMS 會在 中產生，並由 [FIPS 140-3 密碼編譯模組驗證計劃](#)的硬體安全模組 (HSMs) 保護。KMS 金鑰的金鑰材料永遠不會讓 HSM 處於未加密狀態。

AWS KMS 支援多種類型的金鑰存放區，可在使用 AWS KMS 建立和管理加密金鑰時保護您的金鑰材料。提供的所有金鑰存放區選項 AWS KMS 都會在安全層級 3 的 FIPS 140-3 下持續驗證，旨在防止任何人存取您的純文字金鑰或在沒有您許可的情況下使用這些金鑰存放區選項，包括 AWS 運算子。

AWS KMS 標準金鑰存放區

根據預設，會使用標準 AWS KMS HSM 建立 KMS 金鑰。此 HSM 類型可視為 HSMs 的多租戶機群，可讓您從角度管理最具可擴展性、最低成本且最簡單的金鑰存放區。如果您要建立 KMS 金鑰以用於一或多個，AWS 服務 讓 服務可以代表您加密資料，則您將建立對稱金鑰。如果您使用 KMS 金鑰進行自己的應用程式設計，您可以選擇建立對稱加密金鑰、非對稱金鑰或 HMAC 金鑰。

在標準金鑰存放區選項中，會 AWS KMS 建立您的金鑰，然後在服務內部管理的金鑰下對其進行加密。然後，將多個加密版本的金鑰存放在專為耐用性設計的系統中。在標準金鑰存放區類型中產生和保護您的金鑰材料，可讓您充分利用的 AWS 可擴展性、可用性和耐用性，AWS KMS 並降低金鑰存放區的營運負擔和成本。

AWS KMS 具有匯入金鑰材料的標準金鑰存放區

與其 AWS KMS 同時要求 產生和存放指定金鑰的唯一副本，您可以選擇將金鑰材料匯入 AWS KMS，允許您產生自己的 256 位元對稱加密金鑰、RSA 或橢圓曲線 (ECC) 金鑰或雜湊型訊息驗證碼 (HMAC) 金鑰，並將其套用至 KMS 金鑰識別符 (keyId)。這有時稱為自帶金鑰 (BYOK)。從本機金鑰管理系統匯入的金鑰材料必須使用 發行的公有金鑰 AWS KMS、支援的密碼編譯包裝演算法，以及 提供的以時間為基礎的匯入字符來保護 AWS KMS。此程序會驗證您的加密匯入金鑰只有在離開您的環境後，才能由 AWS KMS HSM 解密。

如果您在系統周圍有產生金鑰的特定需求，或想要在 外部複製金鑰 AWS 做為備份，則匯入的金鑰材料可能很有用。請注意，您必須負責匯入金鑰材料的整體可用性和耐久性。雖然 AWS KMS 有匯入金鑰的副本，而且 會在您需要時保持高可用性，但匯入的金鑰提供特殊 API 以供刪除 – `DeleteImportedKeyMaterial`。此 API 會立即刪除 AWS KMS 具有的匯入金鑰材料的所有副本，沒有 AWS 讓 復原金鑰的選項。此外，您可以在匯入的金鑰上設定過期時間，之後金鑰將無法使用。若要讓金鑰在 中再次有用 AWS KMS，您必須重新匯入金鑰材料，並將其指派給相同的 keyId。此匯入金鑰

的刪除動作與代表您 AWS KMS 產生和存放的標準金鑰不同。在標準案例中，金鑰刪除程序具有強制的等待期間，其中排定刪除的金鑰會先封鎖使用。此動作可讓您從任何可能需要該金鑰才能存取資料的應用程式 AWS 或服務，查看日誌中的存取遭拒錯誤。如果您看到這類存取請求，您可以選擇取消排定的刪除並重新啟用金鑰。在可設定的等待期間（介於 7 到 30 天之間）之後，只有 KMS 才會實際刪除金鑰材料、keyID 以及與金鑰相關聯的所有中繼資料。如需可用性和耐用性的詳細資訊，請參閱《AWS KMS 開發人員指南》中的[保護匯入的金鑰材料](#)。

匯入的金鑰材料有一些額外的限制需要注意。由於 AWS KMS 無法產生新的金鑰材料，因此無法設定匯入金鑰的自動輪換。您需要使用新的 keyID 建立新的 KMS 金鑰，然後匯入新的金鑰材料以實現有效的輪換。此外，在匯入的對稱金鑰 AWS KMS 下在 中建立的加密文字，無法使用 外部的金鑰本機複本輕鬆解密 AWS。這是因為 使用的已驗證加密格式會將其他中繼資料 AWS KMS 附加到加密文字，以在解密操作期間保證加密文字是由預期的 KMS 金鑰在先前的加密操作下建立。大多數外部密碼編譯系統都不會了解如何剖析此中繼資料，以取得原始加密文字的存取權，以便能夠使用其對稱金鑰的副本。在匯入的非對稱金鑰（例如 RSA 或 ECC）下建立的加密文字，可以在 外部 AWS KMS 與金鑰的相符（公有或私有）部分搭配使用，因為 不會將其他中繼資料新增至 AWS KMS 加密文字。

AWS KMS 自訂金鑰存放區

然而，如果需要更多的 HSM 控制權，您可以建立自訂金鑰存放區。

自訂金鑰存放區是 內的金鑰存放區 AWS KMS，由您擁有和管理的 外部金鑰管理員 AWS KMS 提供支援。自訂金鑰存放區結合了方便且全面的金鑰管理界面 AWS KMS，以及擁有和控制金鑰材料和密碼編譯操作的能力。當您使用自訂金鑰存放區中的 KMS 金鑰時，您的金鑰管理器將使用您的密碼編譯金鑰執行密碼編譯操作。因此，您需要對密碼編譯金鑰的可用性和持久性以及 HSM 的操作承擔更多責任。

擁有 HSMs 可能有助於滿足尚未允許標準 KMS 金鑰存放區等多租戶 Web 服務保存密碼編譯金鑰的特定法規要求。自訂金鑰存放區不比使用受管 HSMs KMS AWS 金鑰存放區更安全，但它們具有不同的（和更高的）管理和成本影響。因此，您對密碼編譯金鑰的可用性和持久性以及 HSMs 的操作承擔更多責任。無論您使用標準金鑰存放區搭配 AWS KMS HSMs 還是自訂金鑰存放區，服務的設計都會讓包括 AWS 員工在內的任何人都無法擷取您的純文字金鑰，或在未經您許可的情況下使用這些金鑰存放區。AWS KMS 支援兩種類型的自訂金鑰存放區、AWS CloudHSM 金鑰存放區和外部金鑰存放區。

不支援的功能

AWS KMS 不支援自訂金鑰存放區中的下列功能。

- [非對稱 KMS 金鑰](#)
- [HMAC KMS 金鑰](#)

- [含有匯入金鑰資料的 KMS 金鑰](#)
- [自動金鑰輪換](#)
- [多區域金鑰](#)

AWS CloudHSM 金鑰存放區

您可以在[AWS CloudHSM](#)金鑰存放區中建立 KMS 金鑰，其中會產生根使用者金鑰，並儲存在您擁有和管理的 AWS CloudHSM 叢集中。針對某些密碼編譯操作 AWS KMS 使用金鑰的請求會轉送到您的 AWS CloudHSM 叢集，以執行操作。當 AWS CloudHSM 叢集由託管時 AWS，它是由您直接管理和操作的單一租用戶解決方案。您擁有 AWS CloudHSM 叢集中 KMS 金鑰的大部分可用性和效能。若要查看 AWS CloudHSM 自訂金鑰存放區是否適合您的需求，請參閱 [AWS KMS 安全部落格中的 是否適合您？自訂金鑰存放區](#)。AWS

外部金鑰存放區

您可以將設定為 AWS KMS 使用外部金鑰存放區 (XKS)，其中會產生、存放根使用者金鑰，並在外部的金鑰管理系統中使用 AWS 雲端。AWS KMS 使用金鑰進行某些密碼編譯操作的請求會轉送至外部託管系統，以執行操作。具體而言，請求會轉送到您網路中的 XKS Proxy，然後將請求轉送到您使用的任何密碼編譯系統。XKS Proxy 是一種開放原始碼規格，任何人都可以與之整合。許多商業金鑰管理廠商支援 XKS Proxy 規格。由於外部金鑰存放區由您或某些第三方託管，因此您擁有系統中金鑰的所有可用性、耐用性和效能。若要查看外部金鑰存放區是否適合您的需求，請參閱 AWS 新聞部落格上的[公告 AWS KMS 外部金鑰存放區 \(XKS\)](#)。

AWS CloudHSM 金鑰存放區

AWS CloudHSM 金鑰存放區是由[AWS CloudHSM 叢集](#)支援的[自訂金鑰存放區](#)。當您在自訂金鑰存放 AWS KMS key 區中建立時，會在您擁有和管理的 AWS CloudHSM 叢集中 AWS KMS 產生和存放 KMS 金鑰的不可擷取金鑰材料。當您使用自訂金鑰存放區中的 KMS 金鑰時，[密碼編譯操作](#)是在叢集的 HSM 中執行。此功能結合了便利性和廣泛整合，AWS KMS 以及中 AWS CloudHSM 叢集的新增控制 AWS 帳戶。

AWS KMS 提供建立、使用和管理自訂金鑰存放區的完整主控台和 API 支援。您使用自訂金鑰存放區中 KMS 金鑰的方式，與您使用任何 KMS 金鑰的方式相同。例如，您可以使用 KMS 金鑰來產生資料金鑰並加密資料。您也可以將自訂金鑰存放區中的 KMS 金鑰與支援客戶受管金鑰 AWS 的服務搭配使用。

我是否需要自訂金鑰存放區？

對於大多數使用者而言，受 [FIPS 140-3 驗證的密碼編譯模組](#) 保護的預設 AWS KMS 金鑰存放區可滿足其安全需求。您不需要多一層維護責任或依賴額外的服務。

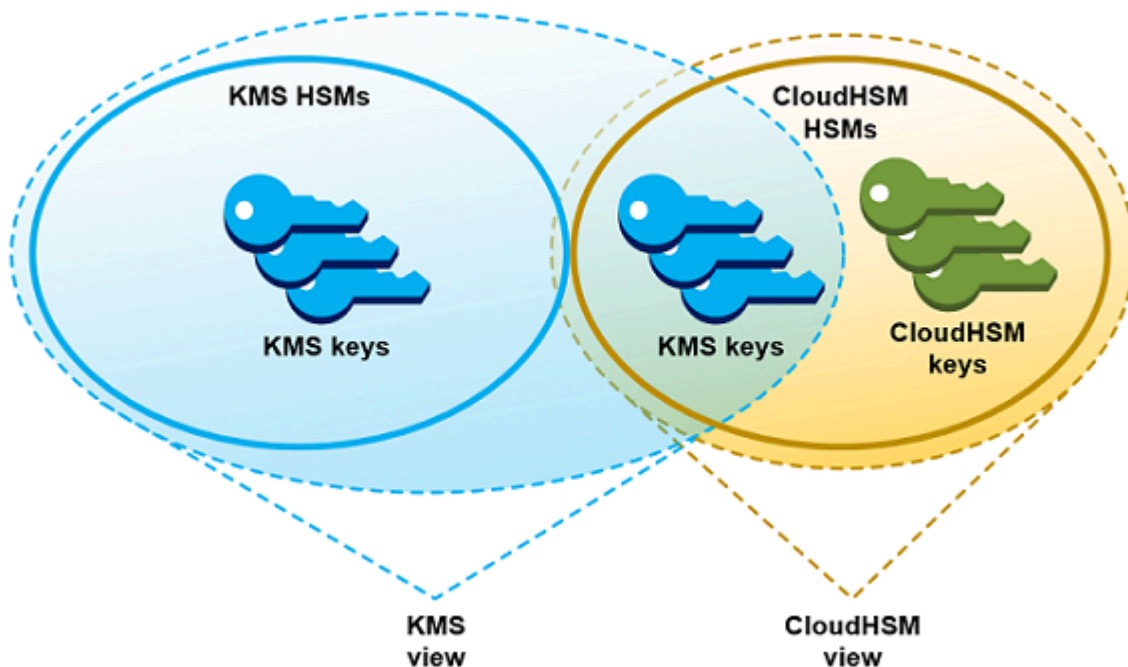
不過，如果您的組織有下列要求，您可能會考慮建立自訂金鑰存放區：

- 您在單一租用戶 HSM 或您可直接控制的 HSM 有明確需要受到保護的金鑰。
- 您需要能夠立即從中移除金鑰材料 AWS KMS。
- 您需要能夠獨立於 AWS KMS 或 稽核金鑰的所有使用 AWS CloudTrail。

自訂金鑰存放區如何運作？

每個自訂金鑰存放區都與 中的 AWS CloudHSM 叢集相關聯 AWS 帳戶。當您將自訂金鑰存放區連接到其叢集時，會 AWS KMS 建立網路基礎設施以支援連線。然後，它會使用叢集中 [專用加密使用者](#) 的登入資料，登入叢集中的金鑰 AWS CloudHSM 用戶端。

您可以在 中建立和管理自訂金鑰存放區 AWS KMS，並在其中建立和管理 HSM 叢集 AWS CloudHSM。當您在 AWS KMS 自訂金鑰存放區中建立時，您可以檢視和管理其中的 KMS 金鑰 AWS KMS。但您也可以 AWS CloudHSM 中檢視和管理其金鑰資料，就像您對叢集內的其他金鑰所做的一樣。



您可以使用自訂金鑰存放區 AWS KMS 中產生的金鑰材料來 [建立對稱加密 KMS](#) 金鑰。然後使用相同的技術來檢視和管理您用於金鑰存放區中 KMS 金鑰的自訂 AWS KMS 金鑰存放區中的 KMS 金鑰。您

可以使用 IAM 和金鑰政策控制存取、建立標籤和別名、啟用和停用 KMS 金鑰，以及排程金鑰刪除。您可以使用 KMS 金鑰進行[密碼編譯操作](#)，並將其與整合的 AWS 服務搭配使用 AWS KMS。

此外，您可以完全控制 AWS CloudHSM 叢集，包括建立和刪除 HSMs 和管理備份。您可以使用 AWS CloudHSM 用戶端和支援的軟體程式庫來檢視、稽核和管理 KMS 金鑰的金鑰材料。當自訂金鑰存放區中斷連接時，AWS KMS 無法存取，且使用者無法使用自訂金鑰存放區中的 KMS 金鑰進行密碼編譯操作。多一道這種控制讓自訂金鑰存放區成為有需要的組織的強大解決方案。

從何處開始？

若要建立和管理 AWS CloudHSM 金鑰存放區，您可以使用 AWS KMS 和 的功能 AWS CloudHSM。

1. 從開始 AWS CloudHSM。[建立作用中 AWS CloudHSM 叢集](#)或選取現有的叢集。叢集必須有至少兩個作用中 HSM 在不同的可用區域。然後為 AWS KMS 建立該叢集中的[專用加密使用者 \(CU\) 帳戶](#)。
2. 在中 AWS KMS，建立與所選 AWS CloudHSM 叢集相關聯的自訂金鑰存放區。AWS KMS 提供完整的管理介面，可讓您建立、檢視、編輯和刪除自訂金鑰存放區。[???](#)
3. 當您準備好使用自訂金鑰存放區時，[請將其連線至相關聯的 AWS CloudHSM 叢集](#)。會 AWS KMS 建立其支援連線所需的網路基礎設施。接著會使用專用加密使用者帳戶來登入叢集，以便能夠在叢集內產生和管理金鑰資料。
4. 現在，您可以[在自訂金鑰存放區建立對稱加密 KMS 金鑰](#)。只需要在建立 KMS 金鑰時指定自訂金鑰存放區。

如果您在任何時候受阻，您可以在[對自訂金鑰存放區進行故障診斷](#)主題中尋找說明。如果找不到問題的解答，請使用本指南每一頁底部的意見回饋連結，或將問題張貼到 [AWS Key Management Service 開發論壇](#)。

配額

AWS KMS 在每個 AWS 帳戶 和 區域中最多允許 [10 個自訂金鑰存放區](#)，包括[AWS CloudHSM 金鑰存放區](#)和[外部金鑰存放區](#)，無論其連線狀態為何。此外，在 [金鑰存放區中使用 KMS AWS CloudHSM 金鑰](#)有 AWS KMS 請求配額。

定價

如需 AWS KMS 自訂金鑰存放區中自訂金鑰存放區和客戶受管金鑰成本的資訊，請參閱 [AWS Key Management Service 定價](#)。如需 AWS CloudHSM 叢集和 HSMs 成本的相關資訊，請參閱 [AWS CloudHSM 定價](#)。

區域

AWS KMS 在 AWS KMS 支援的所有 中支援 AWS CloudHSM 金鑰存放 AWS 區域 區，但亞太區域（墨爾本）、中國（北京）、中國（寧夏）和歐洲（西班牙）除外。

不支援的功能

AWS KMS 不支援自訂金鑰存放區中的下列功能。

- [非對稱 KMS 金鑰](#)
- [HMAC KMS 金鑰](#)
- [含有匯入金鑰資料的 KMS 金鑰](#)
- [自動金鑰輪換](#)
- [多區域金鑰](#)

AWS CloudHSM 金鑰存放區概念

本主題說明 AWS CloudHSM 金鑰存放區中使用的一些術語和概念。

AWS CloudHSM 金鑰存放區

AWS CloudHSM 金鑰存放區是與您 AWS CloudHSM 擁有和管理的叢集相關聯的[自訂金鑰存放區](#)。AWS CloudHSM 叢集由 [FIPS 140-2 或 FIPS 140-3Level 3](#) 認證的硬體安全模組 (HSMs) 提供支援。

當您在 AWS CloudHSM 金鑰存放區中建立 KMS 金鑰時，會在相關聯的 AWS CloudHSM 叢集中 AWS KMS 產生 256 位元、持久性、不可匯出的進階加密標準 (AES) 對稱金鑰。金鑰材料離開您的 HSM 時一定會加密。當您在 金鑰存放區中使用 KMS AWS CloudHSM 金鑰時，密碼編譯操作會在叢集的 HSMs 中執行。

AWS CloudHSM 金鑰存放區結合了 的便利且全面的金鑰管理界面 AWS KMS，以及 AWS CloudHSM 叢集在 中提供的額外控制項 AWS 帳戶。此整合功能可讓您在 中建立、管理和使用 KMS 金鑰，AWS KMS 同時完全控制存放其金鑰資料的 HSMs，包括管理叢集、HSMs 和備份。您可以使用 AWS KMS 主控台和 APIs 來管理金鑰存放區及其 KMS AWS CloudHSM 金鑰。您也可以使用 AWS CloudHSM 主控台、APIs、用戶端軟體和相關聯的軟體程式庫來管理相關聯的叢集。

您可以[檢視和管理](#) AWS CloudHSM 金鑰存放區、[編輯其屬性](#)，以及[連接](#)和[中斷](#)其與其相關聯 AWS CloudHSM 叢集的連線。如果您需要[刪除金鑰 AWS CloudHSM 存放區](#)，您必須先透過排程刪除並等

待寬限期到期，來刪除 AWS CloudHSM 金鑰存放區中的 KMS 金鑰。刪除金鑰 AWS CloudHSM 存放區會從 移除資源 AWS KMS，但不會影響 AWS CloudHSM 叢集。

AWS CloudHSM 叢集

每個 AWS CloudHSM 金鑰存放區都與一個 AWS CloudHSM 叢集相關聯。當您在 AWS CloudHSM 金鑰存放區 AWS KMS key 中建立時，AWS KMS 會在相關聯的叢集中建立其金鑰材料。在您的 AWS CloudHSM 金鑰存放區中使用 KMS 金鑰時，密碼編譯操作會在關聯的叢集中執行。

每個 AWS CloudHSM 叢集只能與一個 AWS CloudHSM 金鑰存放區相關聯。您選擇的叢集無法與另一個 AWS CloudHSM 金鑰存放區相關聯，也無法與其他 AWS CloudHSM 金鑰存放區相關聯的叢集共用備份歷史記錄。叢集必須初始化並處於作用中狀態，而且必須位於與 AWS CloudHSM 金鑰存放區相同的 AWS 帳戶和區域。您可以建立新的叢集或使用現有的叢集。AWS KMS 不需要獨佔使用叢集。若要在金鑰存放區中建立 KMS AWS CloudHSM 金鑰，其關聯的叢集必須至少包含兩個作用中 HSMs。所有其他操作只需要一個 HSM。

您可以在建立 AWS CloudHSM 金鑰存放區時指定 AWS CloudHSM 叢集，而且無法變更它。不過，您可以替代與原始叢集共用備份歷史記錄的任何叢集。這可讓您刪除叢集，並在必要時將它從其中一個備份建立的叢集取代。您可保有相關聯 AWS CloudHSM 叢集的完整控制，讓您可以管理使用者和金鑰、建立和刪除 HSM，並使用和管理備份。

當您準備好使用 AWS CloudHSM 金鑰存放區時，您可以將其連線到其相關聯的 AWS CloudHSM 叢集。您可以隨時連接和中斷連接您的自訂金鑰存放區。連接自訂金鑰存放區時，您可以建立和使用它的 KMS 金鑰。中斷連線時，您可以檢視和管理 AWS CloudHSM 金鑰存放區及其 KMS 金鑰。但是，您無法建立新的 KMS 金鑰或使用 AWS CloudHSM 金鑰存放區中的 KMS 金鑰進行密碼編譯操作。

kmsuser 加密使用者

若要代表您在相關聯的 AWS CloudHSM 叢集中建立和管理金鑰材料，AWS KMS 會在名為 `kmsuser` 的叢集中使用專用 AWS CloudHSM [加密使用者 \(CU\)](#) `kmsuser`。kmsuser CU 是標準 CU 帳戶，會自動同步到叢集中的所有 HSM，並且儲存在叢集備份中。

在建立 AWS CloudHSM 金鑰存放區之前，您可以使用 CloudHSM [kmsuser CLI 中的使用者建立命令，在叢集中建立 CU 帳戶](#)。AWS CloudHSM 然後，當您 [建立 AWS CloudHSM 金鑰存放區](#) 時，您會提供 kmsuser 帳戶密碼給 AWS KMS。當您 [連接自訂金鑰存放區](#) 時，會以 CU kmsuser 身分 AWS KMS 登入叢集，並輪換其密碼。在安全存放 kmsuser 密碼之前，會先 AWS KMS 加密您的密碼。當密碼輪換時，新密碼會加密並以相同的方式存放。

AWS KMS kmsuser 只要 AWS CloudHSM 金鑰存放區已連線，就會保持登入狀態。您不應該將此 CU 帳戶用於其他用途。不過，您對於 kmsuser CU 帳戶會保有最終控制。您可以隨時 [找到擁有的金](#)

[鑰](#)。kmsuser如有需要，您可以[中斷連線自訂金鑰存放區](#)，變更 kmsuser 密碼，[以 kmsuser 身分登入叢集](#)，以及檢視和管理 kmsuser 擁有的金鑰。

如需建立您的 kmsuser CU 帳戶的相關指示，請參閱[建立 kmsuser 加密使用者](#)。

金鑰存放區中的 KMS AWS CloudHSM 金鑰

您可以使用 AWS KMS 或 AWS KMS API 在 AWS CloudHSM 金鑰存放 AWS KMS keys 區中建立。您可以使用與您在任何 KMS 金鑰上所使用的相同技術。唯一的差別是您必須識別 AWS CloudHSM 金鑰存放區，並指定金鑰材料的來源為 AWS CloudHSM 叢集。

當您在[AWS CloudHSM 金鑰存放區中建立 KMS 金鑰](#)時，會在 AWS KMS 建立 KMS 金鑰，AWS KMS 並在其相關聯的叢集中產生 256 位元、持久性、不可匯出的進階加密標準 (AES) 對稱金鑰材料。當您在密碼編譯操作中使用 AWS KMS 金鑰時，操作會在 AWS CloudHSM 叢集中使用以叢集為基礎的 AES 金鑰來執行。雖然 AWS CloudHSM 支援不同類型的對稱和非對稱金鑰，但 AWS CloudHSM 金鑰存放區僅支援 AES 對稱加密金鑰。

您可以在 AWS KMS 主控台中檢視 AWS CloudHSM 金鑰存放區中的 KMS 金鑰，並使用主控台選項來顯示自訂金鑰存放區 ID。您也可以使用 [DescribeKey](#) 操作來尋找 AWS CloudHSM 金鑰存放區 ID 和 AWS CloudHSM 叢集 ID。

AWS CloudHSM 金鑰存放區中的 KMS 金鑰的運作方式與任何 KMS 金鑰相同 AWS KMS。獲授權使用者需要相同權限才能使用和管理 KMS 金鑰。您可以使用相同的主控台程序和 API 操作來檢視和管理金鑰存放區中的 KMS AWS CloudHSM 金鑰。這些包含啟用和停用 KMS 金鑰、建立和使用標籤和別名，以及設定和變更 IAM 和金鑰政策。您可以使用 AWS CloudHSM 金鑰存放區中的 KMS 金鑰進行密碼編譯操作，並將其與支援使用客戶受管金鑰的[整合 AWS 服務](#)搭配使用。不過，您無法啟用[自動金鑰輪換](#)或將[金鑰材料匯入](#)金鑰存放區中的 KMS AWS CloudHSM 金鑰。

您也可以使用相同的程序來[排程刪除](#)金鑰存放區中的 KMS AWS CloudHSM 金鑰。在等待期間到期後，會從 KMS AWS KMS 刪除 KMS 金鑰。然後，會盡最大努力從關聯的 AWS CloudHSM 叢集中刪除 KMS 金鑰的金鑰材料。不過，您可能需要手動從叢集及其備份[刪除遺棄的金鑰材料](#)。

控制對 AWS CloudHSM 金鑰存放區的存取

您可以使用 IAM 政策來控制對 AWS CloudHSM 金鑰存放區和 AWS CloudHSM 叢集的存取。您可以使用金鑰政策、IAM 政策和授權來控制對 AWS CloudHSM 金鑰存放區 AWS KMS keys 中的存取。建議您只提供使用者、群組和角色可能執行的任務所需的許可給他們。

為了支援您的 AWS CloudHSM 金鑰存放區，AWS KMS 需要取得叢集相關資訊 AWS CloudHSM 的許可。它還需要許可，才能建立將金鑰存放區連接到其 AWS CloudHSM 叢集的網路基礎設施 AWS CloudHSM。若要取得這些許可，會在您的 AWS KMS 建立

AWSServiceRoleForKeyManagementServiceCustomKeyStores 服務連結角色 AWS 帳戶。如需詳細資訊，請參閱[授權 AWS KMS 管理 AWS CloudHSM 和 Amazon EC2 資源](#)。

設計您的 AWS CloudHSM 金鑰存放區時，請確定使用和管理金鑰存放區的委託人只有他們所需的許可。下列清單說明 AWS CloudHSM 金鑰存放區管理員和使用者所需的最低許可。

- 建立和管理 AWS CloudHSM 金鑰存放區的委託人需要下列許可才能使用 AWS CloudHSM 金鑰存放區 API 操作。
 - `cloudhsm:DescribeClusters`
 - `kms:CreateCustomKeyStore`
 - `kms:ConnectCustomKeyStore`
 - `kms>DeleteCustomKeyStore`
 - `kms:DescribeCustomKeyStores`
 - `kms:DisconnectCustomKeyStore`
 - `kms:UpdateCustomKeyStore`
 - `iam:CreateServiceLinkedRole`
- 建立和管理與您 AWS CloudHSM 金鑰存放區相關聯 AWS CloudHSM 叢集的委託人需要建立和初始化 AWS CloudHSM 叢集的許可。這包括要有許可來建立或使用 Amazon Virtual Private Cloud (VPC)、建立子網路，以及建立 Amazon EC2 執行個體。他們可能還需要建立和刪除 HSM，以及管理備份。如需必要許可的清單，請參閱《AWS CloudHSM 使用指南》中 [AWS CloudHSM 的身分與存取管理](#)。
- 在 AWS CloudHSM 金鑰存放區 AWS KMS keys 中建立和管理的委託人需要與建立和管理任何 KMS 金鑰的委託人[相同的許可](#) AWS KMS。金鑰存放區中 KMS [金鑰的預設金鑰政策](#)與 中 KMS 金鑰的預設金鑰政策相同 AWS KMS。AWS CloudHSM [屬性型存取控制](#) (ABAC) 使用標籤和別名來控制對 KMS 金鑰的存取，也適用於 AWS CloudHSM 金鑰存放區中的 KMS 金鑰。
- 在您的 AWS CloudHSM 金鑰存放區中使用 KMS 金鑰進行[密碼編譯操作](#)的委託人需要許可，才能使用 KMS 金鑰執行密碼編譯操作，例如 [kms:Decrypt](#)。您可以在金鑰政策或 IAM 政策中提供這些許可。但是，它們不需要任何額外的許可，即可在 金鑰存放區中使用 KMS AWS CloudHSM 金鑰。

建立 AWS CloudHSM 金鑰存放區

您可以在帳戶中建立一或多個 AWS CloudHSM 金鑰存放區。每個 AWS CloudHSM 金鑰存放區都與相同 AWS 帳戶 和 區域中的一個 AWS CloudHSM 叢集相關聯。在您建立 AWS CloudHSM 金鑰存放區之前，您需要[備妥先決條件](#)。然後，您必須先[將](#) AWS CloudHSM 金鑰存放區連接到其 AWS CloudHSM 叢集，才能使用金鑰存放區。

備註

KMS 無法透過 IPv6 與 AWS CloudHSM 金鑰存放區通訊。

如果您嘗試建立與現有中斷連線 AWS CloudHSM 金鑰存放區具有所有相同屬性值的金鑰存放區，AWS KMS 不會建立新的 AWS CloudHSM 金鑰存放區，也不會擲回例外狀況或顯示錯誤。反之，會將重複項目 AWS KMS 識別為重試的可能結果，並傳回現有 AWS CloudHSM 金鑰存放區的 ID。

您不需要立即連接您的 AWS CloudHSM 金鑰存放區。在準備使用之前可以維持在中斷連線狀態。不過，若要確認是否已正確設定，您可能需要[將其連接](#)、[檢視其連接狀態](#)，然後[中斷連接](#)。

主題

- [備妥先決條件](#)
- [建立新的 AWS CloudHSM 金鑰存放區](#)

備妥先決條件

每個 AWS CloudHSM 金鑰存放區都由 AWS CloudHSM 叢集支援。若要建立 AWS CloudHSM 金鑰存放區，您必須指定尚未與另一個金鑰存放區建立關聯的作用中 AWS CloudHSM 叢集。您也需要在叢集的 HSMs 中建立專用加密使用者 (CU)，AWS KMS 可用來代表您建立和管理金鑰。

建立 AWS CloudHSM 金鑰存放區之前，請執行下列動作：

選取 AWS CloudHSM 叢集

每個 AWS CloudHSM 金鑰存放區[只會與一個 AWS CloudHSM 叢集相關聯](#)。當您 AWS KMS keys 在 AWS CloudHSM 金鑰存放區中建立時，會在其中 AWS KMS 建立 KMS 金鑰中繼資料，例如 ID 和 Amazon Resource Name (ARN) AWS KMS。接著，它會在相關聯叢集的 HSM 中建立金鑰材料。您可以[建立新的 AWS CloudHSM 叢集](#)或使用現有的叢集。AWS KMS 不需要叢集的專屬存取權。

您選取的 AWS CloudHSM 叢集會永久與 AWS CloudHSM 金鑰存放區建立關聯。建立 AWS CloudHSM 金鑰存放區後，您可以[變更相關聯叢集的叢集 ID](#)，但您指定的叢集必須與原始叢集共用備份歷史記錄。若要使用不相關的叢集，您需要建立新的 AWS CloudHSM 金鑰存放區。

您選取的 AWS CloudHSM 叢集必須具有下列特性：

- 叢集必須為作用中。

您必須建立叢集、初始化叢集、為您的平台安裝 AWS CloudHSM 用戶端軟體，然後啟用叢集。如需詳細說明，請參閱《AWS CloudHSM 使用者指南》中的 [AWS CloudHSM 入門](#)。

- 未啟用相互 TLS (mTLS)。

KMS 不支援叢集的 mTLS。不得啟用此設定。

- 叢集必須與金鑰存放區位於相同的帳戶和區域。AWS CloudHSM 您無法將某個區域中的 AWS CloudHSM 金鑰存放區與不同區域中的叢集建立關聯。若要在多個區域中建立金鑰基礎設施，您必須在每個區域中建立 AWS CloudHSM 金鑰存放區和叢集。
- 叢集不能與相同帳戶和區域中的其他自訂金鑰存放區關聯。帳戶和區域中的每個 AWS CloudHSM 金鑰存放區都必須與不同的 AWS CloudHSM 叢集相關聯。您不能指定已經與自訂金鑰存放區相關聯的叢集，或與相關聯叢集共用備份歷史記錄的叢集。共用備份歷史記錄的叢集具有相同的叢集憑證。若要檢視叢集的叢集憑證，請使用 AWS CloudHSM 主控台或 [DescribeClusters](#) 操作。

如果您將 [AWS CloudHSM 叢集備份至不同區域](#)，則此叢集會被視為不同的叢集，您可以將備份與該區域中的自訂金鑰存放區關聯。不過，兩個自訂金鑰存放區中的 KMS 金鑰無法互通，即使它們具有相同的後端 key。AWS KMS binds 中繼資料到加密文字，因此只能由加密它的 KMS 金鑰進行解密。

- 必須在區域中至少兩個可用區域中，為叢集設定[私有子網路](#)。由於 AWS CloudHSM 並非所有可用區域都支援，我們建議您在該區域的所有可用區域中建立私有子網路。您不能對現有叢集重新設定子網路，但可以[從備份建立叢集](#)，並於叢集組態中使用不同的子網路。

Important

建立 AWS CloudHSM 金鑰存放區後，請勿刪除為其 AWS CloudHSM 叢集設定的任何私有子網路。如果在叢集組態中 AWS KMS 找不到所有子網路，[嘗試連線至自訂金鑰存放區](#)會失敗並顯示SUBNET_NOT_FOUND連線錯誤狀態。如需詳細資訊，請參閱[如何修正連線失敗](#)。

- [叢集 \(\) 的安全群組](#)必須包含連接埠 2223-2225 上允許透過 IPv4 TCP 流量的傳入規則和傳出規則。cloudhsm-cluster-*<cluster-id>*-sg傳入規則的來源和傳出規則的目的地必須符合安全群組 ID。當您建立叢集時依預設會設定這些規則。請勿刪除或變更。
- 叢集必須包含至少兩個作用中 HSM 在不同的可用區域。若要驗證 HSMs 的數量，請使用 AWS CloudHSM 主控台或 [DescribeClusters](#) 操作。如有需要，您可以[新增 HSM](#)。

尋找信任起點憑證

建立自訂金鑰存放區時，您必須將 AWS CloudHSM 叢集的信任錨點憑證上傳至 AWS KMS。AWS KMS 需要信任錨點憑證，才能將 AWS CloudHSM 金鑰存放區連線至其相關聯的 AWS CloudHSM 叢集。

每個作用中 AWS CloudHSM 叢集都有信任錨點憑證。當您[初始化叢集](#)時，您需要產生此憑證，將它儲存在 `customerCA.crt` 檔案中，然後複製到連接至叢集的主機。

建立的 **kmsuser** 加密使用者 AWS KMS

若要管理您的 AWS CloudHSM 金鑰存放區，會 AWS KMS 登入所選叢集中的 [kmsuser 加密使用者](#) (CU) 帳戶。建立 AWS CloudHSM 金鑰存放區之前，您必須建立 kmsuser CU。然後，當您建立 AWS CloudHSM 金鑰存放區時，請提供密碼 kmsuser AWS KMS。每當您將 AWS CloudHSM 金鑰存放區連接到其相關聯的 AWS CloudHSM 叢集時，都會 AWS KMS 以登入 kmsuser 並輪換 kmsuser 密碼。

Important

當您建立 kmsuser CU 時，請勿指定 2FA 選項。如果您這麼做，AWS KMS 則無法登入，且您的 AWS CloudHSM 金鑰存放區無法連線至此 AWS CloudHSM 叢集。一旦指定 2FA 就無法復原。您必須刪除 CU 再重新建立。

備註

下列程序使用 AWS CloudHSM 用戶端 SDK 5 命令列工具 [CloudHSM CLI](#)。CloudHSM CLI `key-handle` 會取代為 `key-reference`。

2025 年 1 月 1 日，AWS CloudHSM 將結束對用戶端 SDK 3 命令列工具、CloudHSM 管理公用程式 (CMU) 和金鑰管理公用程式 (KMU) 的支援。如需用戶端 SDK 3 命令列工具和用戶端 SDK 5 命令列工具之間差異的詳細資訊，請參閱 AWS CloudHSM 《使用者指南》中的 [從用戶端 SDK 3 CMU 和 KMU 遷移至用戶端 SDK 5 CloudHSM CLI](#)。

1. 請遵循 AWS CloudHSM 使用者指南的 [CloudHSM 命令列界面入門 \(CLI\)](#) 主題中所述的入門程序。
2. 使用 [使用者建立](#) 命令來建立名為 `CUkmsuser`。

密碼必須包含 7 到 32 個英數字元。區分大小寫，且不能包含任何特殊字元。

下列範例命令會建立 kmsuser CU。

```
aws-cloudhsm > user create --username kmsuser --role crypto-user
Enter password:
Confirm password:
{
  "error_code": 0,
  "data": {
    "username": "kmsuser",
    "role": "crypto-user"
  }
}
```

建立新的 AWS CloudHSM 金鑰存放區

[組合先決條件](#)後，您可以在 AWS KMS 主控台中或使用 [CreateCustomKeyStore](#) 操作建立新的 AWS CloudHSM 金鑰存放區。

使用 AWS KMS 主控台

當您在 中建立 AWS CloudHSM 金鑰存放區時 AWS 管理主控台，您可以在工作流程中新增和建立[先決條件](#)。不過，如果事先備妥，則程序會更快。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格依次選擇自訂金鑰存放區、AWS CloudHSM 金鑰存放區。
4. 選擇 Create a key store (建立金鑰存放區)。
5. 輸入自訂金鑰存放區的易記名稱。該名稱在帳戶的所有自訂金鑰存放區中必須是唯一的。

Important

請勿在此欄位包含機密或敏感資訊。此欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

6. 選取 AWS CloudHSM 金鑰存放區的[AWS CloudHSM 叢集](#)。或者，若要建立新的 AWS CloudHSM 叢集，請選擇建立 AWS CloudHSM 叢集連結。

選單會顯示您帳戶和區域中尚未與 AWS CloudHSM 金鑰存放區建立關聯的 AWS CloudHSM 叢集。叢集必須[滿足要求](#)，才能與自訂金鑰存放區建立關聯。

7. 選擇選擇檔案，然後上傳所選 AWS CloudHSM 叢集的信任錨點憑證。這是 `customerCA.crt` 檔案，您在[初始化叢集](#)時所建立。
8. 輸入您在所選叢集中建立的 [kmsuser 加密使用者](#) (CU) 的密碼。
9. 選擇建立。

當程序成功時，新的 AWS CloudHSM 金鑰存放區會出現在帳戶和區域中的 AWS CloudHSM 金鑰存放區清單中。如果不成功，則會出現錯誤訊息來描述問題，並提供如何修正的說明。如果您需要更多協助，請參閱[對自訂金鑰存放區進行故障診斷](#)。

如果您嘗試使用與現有中斷連線 AWS CloudHSM 的 AWS CloudHSM 金鑰存放區相同的所有屬性值來建立金鑰存放區，AWS KMS 不會建立新的 AWS CloudHSM 金鑰存放區，也不會擲回例外狀況或顯示錯誤。反之，會將重複項目 AWS KMS 識別為重試的可能結果，並傳回現有 AWS CloudHSM 金鑰存放區的 ID。

下一步：不會自動連接新的 AWS CloudHSM 金鑰存放區。您必須先將自訂金鑰存放區連接到其相關聯的 AWS CloudHSM 叢集，才能在 AWS CloudHSM 金鑰存放區 AWS KMS keys 中建立。[???](#)

使用 AWS KMS API

您可以使用 [CreateCustomKeyStore](#) 操作來建立新的 AWS CloudHSM 金鑰存放區，該存放區與帳戶和區域中的 AWS CloudHSM 叢集相關聯。以下範例使用 AWS Command Line Interface (AWS CLI)，但您可以使用任何支援的程式設計語言。

`CreateCustomKeyStore` 操作需要以下參數值。

- `CustomKeyStoreName` – 自訂金鑰存放區在帳戶中唯一的易用名稱。

Important

請勿在此欄位包含機密或敏感資訊。此欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

- `CloudHsmClusterId` – 滿足 AWS CloudHSM [金鑰存放區需求](#)的叢集的 AWS CloudHSM 叢集 ID。
- `KeyStorePassword` – 所指定叢集內的 `kmsuser` CU 帳戶的密碼。
- `TrustAnchorCertificate` – 您在[初始化叢集](#)時所建立 `customerCA.crt` 檔案的內容。

以下範例使用虛構的叢集 ID。執行命令之前，請換成有效的叢集 ID。

```
$ aws kms create-custom-key-store
  --custom-key-store-name ExampleCloudHSMKeyStore \
  --cloud-hsm-cluster-id cluster-1a23b4cdefg \
  --key-store-password kmsPswd \
  --trust-anchor-certificate <certificate-goes-here>
```

如果您使用的是 AWS CLI，您可以指定信任錨點憑證檔案，而不是其內容。在下列範例中，customerCA.crt 檔案位於根目錄。

```
$ aws kms create-custom-key-store
  --custom-key-store-name ExampleCloudHSMKeyStore \
  --cloud-hsm-cluster-id cluster-1a23b4cdefg \
  --key-store-password kmsPswd \
  --trust-anchor-certificate file://customerCA.crt
```

當操作成功時，CreateCustomKeyStore 會傳回自訂金鑰存放區 ID，如下回應範例所示。

```
{
  "CustomKeyStoreId": cks-1234567890abcdef0
}
```

如果操作失敗，請修正例外狀況所指出的錯誤，然後重試。如需其他說明，請參閱[對自訂金鑰存放區進行故障診斷](#)。

如果您嘗試使用與現有中斷連線 AWS CloudHSM 的 AWS CloudHSM 金鑰存放區相同的所有屬性值來建立金鑰存放區，AWS KMS 不會建立新的 AWS CloudHSM 金鑰存放區，也不會擲回例外狀況或顯示錯誤。反之，會將重複項目 AWS KMS 識別為重試的可能結果，並傳回現有 AWS CloudHSM 金鑰存放區的 ID。

下一步：若要使用 AWS CloudHSM 金鑰存放區，[請將其連接到其 AWS CloudHSM 叢集](#)。

檢視 AWS CloudHSM 金鑰存放區

您可以使用 AWS KMS 主控台或 [DescribeCustomKeyStores](#) 操作來檢視每個帳戶和區域中的 AWS CloudHSM 金鑰存放區。

使用 AWS KMS 主控台

當您在 中檢視 AWS CloudHSM 金鑰存放區時 AWS 管理主控台，您可以看到以下內容：

- 自訂金鑰存放區名稱和 ID
- 關聯 AWS CloudHSM 叢集的 ID
- 叢集中 HSM 的數量
- 目前的連接狀態

連線狀態 (狀態) 值為已中斷連線，表示自訂金鑰存放區是新的且從未連線，或是刻意[中斷與其 AWS CloudHSM 叢集的連線](#)。不過，如果您嘗試在連接的自訂金鑰存放區中使用 KMS 金鑰失敗，這可能表示自訂金鑰存放區或其 AWS CloudHSM 叢集發生問題。如需協助，請參閱[如何修正失效的 KMS 金鑰](#)。

若要檢視指定帳戶和區域中的 AWS CloudHSM 金鑰存放區，請使用下列程序。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格依次選擇自訂金鑰存放區、AWS CloudHSM 金鑰存放區。

若要自訂顯示，請按一下出現在 Create key store (建立金鑰存放區) 按鈕下的齒輪圖示。

使用 AWS KMS API

若要檢視您的 AWS CloudHSM 金鑰存放區，請使用 [DescribeCustomKeyStores](#) 操作。在預設情況下，此操作會傳回帳戶和區域中的所有自訂金鑰存放區。但是，您可以使用 CustomKeyId 或 CustomKeyName 參數 (但不能同時使用) 來限制對特定自訂金鑰存放區的輸出。對於 AWS CloudHSM 金鑰存放區，輸出包含自訂金鑰存放區 ID 和名稱、自訂金鑰存放區類型、相關聯 AWS CloudHSM 叢集的 ID 和連線狀態。如果連接狀態指出錯誤，則輸出也會包含描述錯誤原因的錯誤碼。

本節中的範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

例如，以下命令會傳回帳戶和區域中的所有自訂金鑰存放區。您可以使用 Limit 和 Marker 參數來切換輸出中的自訂金鑰存放區頁面。

```
$ aws kms describe-custom-key-stores
```

以下範例命令使用 CustomKeyName 參數來取得僅具有 ExampleCloudHSMKeyStore 易記名稱的自訂金鑰存放區。您可以在每個命令中使用 CustomKeyName 或 CustomKeyId 參數 (但不可同時使用)。

下列範例輸出代表連接到其 AWS CloudHSM 叢集的 AWS CloudHSM 金鑰存放區。

 Note

CustomKeyStoreType 欄位已新增至DescribeCustomKeyStores回應，以區分 AWS CloudHSM 金鑰存放區與外部金鑰存放區。

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleCloudHSMKeyStore
{
  "CustomKeyStores": [
    {
      "CloudHsmClusterId": "cluster-1a23b4cdefg",
      "ConnectionState": "CONNECTED",
      "CreationDate": "1.499288695918E9",
      "CustomKeyId": "cks-1234567890abcdef0",
      "CustomKeyStoreName": "ExampleCloudHSMKeyStore",
      "CustomKeyStoreType": "AWS_CLOUDHSM",
      "TrustAnchorCertificate": "<certificate appears here>"
    }
  ]
}
```

ConnectionState 的 Disconnected 表示自訂金鑰存放區從未連線，或刻意與其 [AWS CloudHSM 叢集中斷](#)連線。不過，如果嘗試在連線的金鑰存放區中使用 KMS AWS CloudHSM 金鑰失敗，這可能表示 AWS CloudHSM 金鑰存放區或其 AWS CloudHSM 叢集發生問題。如需協助，請參閱 [如何修正失效的 KMS 金鑰](#)。

如果自訂金鑰存放區的 ConnectionState 為 FAILED，則 DescribeCustomKeyStores 回應會包含一個 ConnectionErrorCode 元素，解釋錯誤的原因。

例如，在以下輸出中，INVALID_CREDENTIALS 值指出自訂金鑰存放區連接失敗，因為 [kmsuser](#) 密碼無效。如需此錯誤和其他連接錯誤失敗的協助，請參閱 [對自訂金鑰存放區進行故障診斷](#)。

```
$ aws kms describe-custom-key-stores --custom-key-store-id cks-1234567890abcdef0
{
  "CustomKeyStores": [
    {
      "CloudHsmClusterId": "cluster-1a23b4cdefg",
      "ConnectionErrorCode": "INVALID_CREDENTIALS",

```

```
    "ConnectionState": "FAILED",
    "CustomKeyStoreId": "cks-1234567890abcdef0",
    "CustomKeyStoreName": "ExampleCloudHSMKeyStore",
    "CustomKeyStoreType": "AWS_CLOUDHSM",
    "CreationDate": "1.499288695918E9",
    "TrustAnchorCertificate": "<certificate appears here>"
  }
]
```

進一步了解：

- [檢視外部金鑰存放區](#)
- [識別金鑰存放區中的 KMS AWS CloudHSM 金鑰](#)
- [使用 記錄 AWS KMS API 呼叫 AWS CloudTrail](#)

編輯 AWS CloudHSM 金鑰存放區設定

您可以變更現有 AWS CloudHSM 金鑰存放區的設定。自訂金鑰存放區必須中斷其 AWS CloudHSM 叢集的連線。

若要編輯 AWS CloudHSM 金鑰存放區設定：

1. [將自訂金鑰存放區](#)從其 AWS CloudHSM 叢集中斷連線。

當自訂金鑰存放區中斷連線時，您無法在自訂金鑰存放區中建立 AWS KMS keys (KMS 金鑰)，也無法使用其包含的 KMS 金鑰進行[密碼編譯操作](#)。

2. 編輯一或多個 AWS CloudHSM 金鑰存放區設定。

您可以編輯自訂金鑰存放區中的以下設定：

自訂金鑰存放區的易記名稱。

輸入新的易記名稱。新名稱在 中的所有自訂金鑰存放區中必須是唯一的 AWS 帳戶。

Important

請勿在此欄位包含機密或敏感資訊。此欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。

相關聯叢集的 AWS CloudHSM 叢集 ID。

編輯此值以取代原始叢集的相關 AWS CloudHSM 叢集。如果自訂金鑰存放區叢集 AWS CloudHSM 損毀或刪除，您可以使用此功能來修復自訂金鑰存放區。

指定與原始 AWS CloudHSM 叢集共用備份歷史記錄的叢集，並滿足與自訂金鑰存放區關聯的要求，包括在不同的可用區域中的兩個作用中 HSMs。共用備份歷史記錄的叢集具有相同的叢集憑證。若要檢視叢集的叢集憑證，請使用 [DescribeClusters](#) 操作。您無法使用編輯功能，將自訂金鑰存放區與不相關的 AWS CloudHSM 叢集產生關聯。

[kmsuser 加密使用者](#) (CU) 的目前密碼。

告知叢集中 kmsuser AWS CloudHSM CU 的 AWS KMS 目前密碼。此動作不會變更 AWS CloudHSM 叢集中 kmsuser CU 的密碼。

如果您變更 AWS CloudHSM 叢集中 kmsuser CU 的密碼，請使用此功能來告知 AWS KMS 新密碼 kmsuser。否則，AWS KMS 會無法登入叢集，並且將自訂金鑰存放區連接到叢集的所有嘗試都會失敗。

3. [將自訂金鑰存放區重新連接](#)至其 AWS CloudHSM 叢集。

編輯您的金鑰存放區設定

您可以在 AWS KMS 主控台或使用 [UpdateCustomKeyStore](#) 操作來編輯 AWS CloudHSM 金鑰存放區設定。

使用 AWS KMS 主控台

編輯 AWS CloudHSM 金鑰存放區時，您可以變更任何 或可設定的值。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格依次選擇自訂金鑰存放區、AWS CloudHSM 金鑰存放區。
4. 選擇您要編輯的 AWS CloudHSM 金鑰存放區資料列。

如 Connection state (連接狀態) 欄的值不是 DISCONNECTED (已中斷連接)，則在編輯之前，您必須先中斷連接自訂金鑰存放區。(從 Key store actions (金鑰存放區動作) 選單中，選擇 Disconnect (中斷連接)。)

當 AWS CloudHSM 金鑰存放區中斷連線時，您可以管理 AWS CloudHSM 金鑰存放區及其 KMS 金鑰，但無法在 AWS CloudHSM 金鑰存放區中建立或使用 KMS 金鑰。

5. 從 Key store actions (金鑰存放區動作) 選單中，選擇 Edit (編輯)。

6. 執行下列其中一或多個動作。

- 輸入自訂金鑰存放區的易記名稱。
- 輸入相關叢集的 AWS CloudHSM 叢集 ID。
- 在關聯的 AWS CloudHSM 叢集中輸入 `kmsuser` 加密使用者的目前密碼。

7. 選擇儲存。

當程序成功時，會出現訊息描述您編輯的設定。當操作失敗時，會出現錯誤訊息，其中描述問題並提供如何修正的協助。如果您需要更多協助，請參閱[對自訂金鑰存放區進行故障診斷](#)。

8. [重新連接自訂金鑰存放區。](#)

若要使用 AWS CloudHSM 金鑰存放區，您必須在編輯後重新連線。您可以將 AWS CloudHSM 金鑰存放區保持中斷連接。但在中斷連線時，您無法在 AWS CloudHSM 金鑰存放區中建立 KMS 金鑰，或在[密碼編譯操作](#)中使用 AWS CloudHSM 金鑰存放區中的 KMS 金鑰。

使用 AWS KMS API

若要變更 AWS CloudHSM 金鑰存放區的屬性，請使用 [UpdateCustomKeyStore](#) 操作。您可以在相同命令中變更自訂金鑰存放區的多個屬性。如果操作成功，會 AWS KMS 傳回 HTTP 200 回應和沒有屬性的 JSON 物件。若要確認變更已生效，請使用 [DescribeCustomKeyStores](#) 操作。

本節中的範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

首先使用 [DisconnectCustomKeyStore](#) 將自訂金鑰存放區與其叢集中斷連線。AWS CloudHSM 將範例自訂金鑰存放區 ID `cks-1234567890abcdef0` 以實際 ID 取代。

```
$ aws kms disconnect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

第一個範例使用 [UpdateCustomKeyStore](#) 將 AWS CloudHSM 金鑰存放區的易記名稱變更為 `DevelopmentKeys`。命令使用 `CustomKeyId` 參數來識別 AWS CloudHSM 金鑰存放區，並使用 `CustomKeyName` 來指定自訂金鑰存放區的新名稱。

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 --new-custom-key-store-name DevelopmentKeys
```

下列範例會將與 AWS CloudHSM 金鑰存放區相關聯的叢集變更為相同叢集的另一個備份。命令使用 CustomKeyStoreId 參數來識別 AWS CloudHSM 金鑰存放區，並使用 CloudHsmClusterId 參數來指定新的叢集 ID。

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 --cloud-hsm-cluster-id cluster-1a23b4cdefg
```

下列範例 AWS KMS 會告知目前的kmsuser密碼為 ExamplePassword。命令使用 CustomKeyStoreId 參數來識別 AWS CloudHSM 金鑰存放區，並使用 KeyStorePassword 參數來指定目前的密碼。

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 --key-store-password ExamplePassword
```

最後一個命令會將 AWS CloudHSM 金鑰存放區重新連線至其 AWS CloudHSM 叢集。您可以將自訂金鑰存放區保持在中斷連線狀態，但必須先將其連線，才能建立新的 KMS 金鑰或將現有的 KMS 金鑰用於[密碼編譯操作](#)。將範例自訂金鑰存放區 ID 以實際 ID 取代。

```
$ aws kms connect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

連接 AWS CloudHSM 金鑰存放區

新 AWS CloudHSM 金鑰存放區未連線。在 AWS CloudHSM 金鑰存放區 AWS KMS keys 中建立和使用之前，您需要將其連接到其相關聯的 AWS CloudHSM 叢集。您可以隨時連接和中斷連接您的 AWS CloudHSM 金鑰存放區，並[檢視其連線狀態](#)。

您不需要連接 AWS CloudHSM 金鑰存放區。您可以讓 AWS CloudHSM 金鑰存放區無限期處於中斷連線狀態，並只在您需要使用它時才連接。不過，您可能希望定期測試連接，以驗證設定正確並且可連接。

Note

AWS CloudHSM 只有在金鑰存放區從未連線或您明確中斷DISCONNECTED連線時，金鑰存放區才會有連線狀態。如果您的 AWS CloudHSM 金鑰存放區連線狀態為 `CONNECTED`但您無

法使用它，請確定其相關聯的 AWS CloudHSM 叢集處於作用中狀態，且至少包含一個作用中 HSMs。如需連線失敗的協助，請參閱 [the section called “對自訂金鑰存放區進行故障診斷”](#)。

當您連接 AWS CloudHSM 金鑰存放區時，會 AWS KMS 尋找相關聯的 AWS CloudHSM 叢集、連接至該叢集、以 [kmsuser 加密使用者](#) (CU) 身分登入 AWS CloudHSM 用戶端，然後只要 AWS CloudHSM 金鑰存放區已連線，就會輪換登入 AWS CloudHSM 用戶端 kmsuser 的 password. AWS KMS remains.

若要建立連線，會在叢集的虛擬私有雲端 (VPC) kms-*<custom key store ID>* 中 AWS KMS 建立名為 [的安全群組](#)。安全群組具有單一規則，允許來自叢集安全群組的傳入流量。AWS KMS 也會在叢集私有子網路的每個可用區域中建立 [彈性網路介面](#) (ENI)。會將 ENIs AWS KMS 新增至 kms-*<cluster ID>* 安全群組和叢集的安全群組。每個 ENI 的描述為 KMS managed ENI for cluster *<cluster-ID>*。

連接程序可能需要很長的時間才能完成；最多 20 分鐘。

連接 AWS CloudHSM 金鑰存放區之前，請確認其符合要求。

- 其關聯的 AWS CloudHSM 叢集必須至少包含一個作用中的 HSM。若要尋找叢集中的 HSMs 數量，請在 AWS CloudHSM 主控台中檢視叢集，或使用 [DescribeClusters](#) 操作。如有需要，您可以 [新增 HSM](#)。
- 叢集必須擁有 [kmsuser 加密使用者](#) (CU) 帳戶，但當您連接 AWS CloudHSM 金鑰存放區時，該 CU 無法登入叢集。如需登出的說明，請參閱 [如何登出和重新連線](#)。
- AWS CloudHSM 金鑰存放區的連線狀態不能是 DISCONNECTING 或 FAILED。若要檢視連線狀態，請使用 AWS KMS 主控台或 [DescribeCustomKeyStores](#) 回應。如果連接狀態為 FAILED，請中斷連接自訂金鑰存放區，解決問題，然後進行連接。

如需連線失敗的協助，請參閱 [如何修正連線失敗](#)。

當您的 AWS CloudHSM 金鑰存放區連線時，您可以在 [其中建立 KMS 金鑰](#)，並在 [密碼編譯操作](#) 中使用現有的 KMS 金鑰。

連接並重新連線至您的 AWS CloudHSM 金鑰存放區

您可以在 AWS KMS 主控台或使用 [ConnectCustomKeyStore](#) 操作來連接或重新連接您的 AWS CloudHSM 金鑰存放區。

使用 AWS KMS 主控台

若要連接 中的 AWS CloudHSM 金鑰存放區 AWS 管理主控台，請先從自訂 AWS CloudHSM 金鑰存放區頁面選取金鑰存放區。此連接程序最長可能需要 20 分鐘的時間才能完成。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格依次選擇自訂金鑰存放區、AWS CloudHSM 金鑰存放區。
4. 選擇您要連線的 AWS CloudHSM 金鑰存放區資料列。

如果 AWS CloudHSM 金鑰存放區的連線狀態為失敗，您必須在連接之前[中斷連接自訂金鑰存放區](#)。

5. 從 Key store actions (金鑰存放區動作) 選單中，選擇 Connect (連接)。

AWS KMS 會開始連接自訂金鑰存放區的程序。它會找到相關的 AWS CloudHSM 叢集，建置所需的網路基礎設施，連接它，以 kmsuser CU 身分登入 AWS CloudHSM 叢集，並輪換 kmsuser 密碼。當操作完成時，連接狀態會變為 CONNECTED (已連接)。

如果部署失敗，說明失敗原因的錯誤訊息會顯示。在您嘗試再次連線之前，[請先檢視金鑰存放區的連線狀態](#)。AWS CloudHSM 如狀態為 FAILED (失敗)，在重新連接之前，您必須先[中斷連接自訂金鑰存放區](#)。如果您需要協助，請參閱[對自訂金鑰存放區進行故障診斷](#)。

下一步：[the section called “在 金鑰存放區中建立 KMS AWS CloudHSM 金鑰”](#)。

使用 AWS KMS API

若要連接中斷連接的 AWS CloudHSM 金鑰存放區，請使用 [ConnectCustomKeyStore](#) 操作。相關聯的 AWS CloudHSM 叢集必須至少包含一個作用中的 HSM，且連線狀態不能為 FAILED。

連接程序可能需要很長的時間才能完成；最多 20 分鐘。除非其快速失敗，否則 操作會傳回 HTTP 200 回應和不帶屬性的 JSON 物件。不過，這個初始回應並不表示連接已成功。若要判斷自訂金鑰存放區的連接狀態，請參閱 [DescribeCustomKeyStores](#) 回應。

本節中的範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

若要識別 AWS CloudHSM 金鑰存放區，請使用其自訂金鑰存放區 ID。您可以在主控台的 Custom key stores (自訂金鑰存放區) 頁面查找 ID，也可以使用不帶參數的 [DescribeCustomKeyStores](#) 操作。執行此範例之前，請將範例 ID 以有效的 ID 取代。

```
$ aws kms connect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

若要驗證 AWS CloudHSM 金鑰存放區是否已連線，請使用 [DescribeCustomKeyStores](#) 操作。在預設情況下，此操作會傳回您帳戶和區域中的所有自訂金鑰存放區。但是，您可以使用 CustomKeyId 或 CustomKeyName 參數 (但不能同時使用) 來限制對特定自訂金鑰存放區的回應。此 ConnectionState 值 CONNECTED 指出自訂的金鑰存放區連接到它的 AWS CloudHSM 叢集。

Note

CustomKeyType 欄位已新增至 DescribeCustomKeyStores 回應，以區分 AWS CloudHSM 金鑰存放區與外部金鑰存放區。

```
$ aws kms describe-custom-key-stores --custom-key-store-id cks-1234567890abcdef0
{
  "CustomKeyStores": [
    {
      "CustomKeyId": "cks-1234567890abcdef0",
      "CustomKeyName": "ExampleCloudHSMKeyStore",
      "CloudHsmClusterId": "cluster-1a23b4cdefg",
      "CustomKeyType": "AWS_CLOUDHSM",
      "TrustAnchorCertificate": "<certificate string appears here>",
      "CreationDate": "1.499288695918E9",
      "ConnectionState": "CONNECTED"
    }
  ],
}
```

如果 ConnectionState 值為 failed，ConnectionErrorCode 元素會指出失敗的原因。在此情況下，在您的帳戶中 AWS KMS 找不到 AWS CloudHSM 叢集 ID 為 cluster-1a23b4cdefg。如果已刪除叢集，則可以從原始叢集的[備份還原叢集](#)，然後[編輯自訂金鑰存放區的叢集 ID](#)。如需回應連接錯誤代碼的說明，請參閱 [如何修正連線失敗](#)。

```
$ aws kms describe-custom-key-stores --custom-key-store-id cks-1234567890abcdef0
{
  "CustomKeyStores": [
    {
      "CustomKeyId": "cks-1234567890abcdef0",
      "CustomKeyName": "ExampleKeyStore",
      "CloudHsmClusterId": "cluster-1a23b4cdefg",
      "CustomKeyType": "AWS_CLOUDHSM",
    }
  ],
}
```



```
"TrustAnchorCertificate": "<certificate string appears here>",
"CreationDate": "1.499288695918E9",
"ConnectionState": "FAILED"
"ConnectionErrorCode": "CLUSTER_NOT_FOUND"
],
}
```

中斷連接 AWS CloudHSM 金鑰存放區

當您中斷連接 AWS CloudHSM 金鑰存放區時，會 AWS KMS 登出 AWS CloudHSM 用戶端、中斷與相關聯 AWS CloudHSM 叢集的連線，並移除其為支援連線而建立的網路基礎設施。

當 AWS CloudHSM 金鑰存放區中斷連線時，您可以管理 AWS CloudHSM 金鑰存放區及其 KMS 金鑰，但無法在 AWS CloudHSM 金鑰存放區中建立或使用 KMS 金鑰。金鑰存放區的連接狀態為 DISCONNECTED，並且自訂金鑰存放區中 KMS 金鑰的[金鑰狀態](#)為 Unavailable，除非其為 PendingDeletion。您可以隨時重新連接 AWS CloudHSM 金鑰存放區。

Note

AWS CloudHSM 只有在金鑰存放區從未連線或您明確中斷 DISCONNECTED 連線時，金鑰存放區才會有連線狀態。如果您的 AWS CloudHSM 金鑰存放區連線狀態為，CONNECTED 但您無法使用它，請確定其相關聯的 AWS CloudHSM 叢集處於作用中狀態，且至少包含一個作用中 HSMs。如需連線失敗的協助，請參閱 [the section called “對自訂金鑰存放區進行故障診斷”](#)。

當您中斷連接自訂金鑰存放區時，該金鑰存放區中的 KMS 金鑰會立即變成無法使用 (視最終一致性而定)。不過，使用受 KMS 金鑰保護之[資料金鑰](#)所加密的資源不會受影響，除非再次使用 KMS 金鑰 (例如解密資料金鑰)。此問題會影響 AWS 服務，其中許多服務會使用資料金鑰來保護您的資源。如需詳細資訊，請參閱[無法使用的 KMS 金鑰如何影響資料金鑰](#)。

Note

當自訂金鑰存放區中斷連接時，所有在自訂金鑰存放區中建立 KMS 金鑰的嘗試，或在密碼編譯操作中使用現有 KMS 金鑰的嘗試，均會失敗。此動作可防止使用者存放和存取敏感資料。

若要更好地預估中斷連接您的自訂金鑰存放區的效果，請在自訂金鑰存放區中[識別 KMS 金鑰](#)，並[判斷其過去的使用情形](#)。

您可能會因為下列原因中斷連接 AWS CloudHSM 金鑰存放區：

- 為了輪換 **kmsuser** 密碼。AWS KMS 會在每次連接到 AWS CloudHSM 叢集時變更 kmsuser 密碼。若要強制密碼輪換，只需中斷連接並重新連接。
- 稽核叢集中 KMS 金鑰的金鑰材料。AWS CloudHSM 當您中斷連接自訂金鑰存放區時，會 AWS KMS 登出 AWS CloudHSM 用戶端中的 [kmsuser 加密使用者帳戶](#)。這可讓您以 kmsuser CU 身分登入叢集，並稽核和管理 KMS 金鑰的金鑰材料。
- 立即停用 AWS CloudHSM 金鑰存放區中的所有 KMS 金鑰。您可以使用 AWS 管理主控台 或 [DisableKey](#) 操作，在 AWS CloudHSM 金鑰存放區中 [停用和重新啟用 KMS](#) 金鑰。這些操作會快速完成，但一次只能對一個 KMS 金鑰進行。中斷連接 AWS CloudHSM 金鑰存放區會立即將 AWS CloudHSM 金鑰存放區中所有 KMS 金鑰的金鑰狀態變更為 Unavailable，以防止在任何密碼編譯操作中使用它們。
- 為了修復失敗的連接嘗試。如果嘗試連接 AWS CloudHSM 金鑰存放區失敗（自訂金鑰存放區的連線狀態為 FAILED），您必須先中斷連接 AWS CloudHSM 金鑰存放區，然後再嘗試再次連接。

中斷連接您的 AWS CloudHSM 金鑰存放區

您可以在 AWS KMS 主控台或使用 [DisconnectCustomKeyStore](#) 操作中斷連接 AWS CloudHSM 金鑰存放區。

使用 AWS KMS 主控台中斷連線

若要中斷 AWS KMS 主控台中已連線的 AWS CloudHSM 金鑰存放區，請先從自訂 AWS CloudHSM 金鑰存放區頁面中選擇金鑰存放區。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格依次選擇自訂金鑰存放區、AWS CloudHSM 金鑰存放區。
4. 選擇您想要中斷連接的外部金鑰存放區列。
5. 從 Key store actions (金鑰存放區動作) 選單中，選擇 Disconnect (中斷連接)。

當操作完成時，連接狀態會從 Disconnecting (正在中斷連接) 變為 Disconnected (已中斷連接)。如果操作失敗，會出現錯誤訊息，其中描述問題並提供如何修正的協助。如果您需要更多協助，請參閱 [對自訂金鑰存放區進行故障診斷](#)。

使用 AWS KMS API 中斷連線

若要中斷連接已連線的 AWS CloudHSM 金鑰存放區，請使用 [DisconnectCustomKeyStore](#) 操作。如果操作成功，會 AWS KMS 傳回 HTTP 200 回應和沒有屬性的 JSON 物件。

本節中的範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

此範例會中斷 AWS CloudHSM 金鑰存放區的連線。執行此範例之前，請將範例 ID 以有效的 ID 取代。

```
$ aws kms disconnect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

若要確認 AWS CloudHSM 金鑰存放區已中斷連線，請使用 [DescribeCustomKeyStores](#) 操作。在預設情況下，此操作會傳回您帳戶和區域中的所有自訂金鑰存放區。但是，您可以使用 CustomKeyId 和 CustomKeyName 參數 (但不能同時使用) 來限制對特定自訂金鑰存放區的回應。ConnectionState 的值 DISCONNECTED 表示此範例 AWS CloudHSM 金鑰存放區未連線到其 AWS CloudHSM 叢集。

```
$ aws kms describe-custom-key-stores --custom-key-store-id cks-1234567890abcdef0
{
  "CustomKeyStores": [
    {
      "CloudHsmClusterId": "cluster-1a23b4cdefg",
      "ConnectionState": "DISCONNECTED",
      "CreationDate": "1.499288695918E9",
      "CustomKeyId": "cks-1234567890abcdef0",
      "CustomKeyName": "ExampleKeyStore",
      "CustomKeyType": "AWS_CLOUDHSM",
      "TrustAnchorCertificate": "<certificate string appears here>"
    }
  ],
}
```

刪除 AWS CloudHSM 金鑰存放區

當您刪除 AWS CloudHSM 金鑰存放區時，會從 KMS AWS KMS 刪除 AWS CloudHSM 有關金鑰存放區的所有中繼資料，包括其與 AWS CloudHSM 叢集關聯的資訊。此操作不會影響 AWS CloudHSM 叢集、其 HSMs 或其使用者。您可以建立與相同 AWS CloudHSM 叢集相關聯的新 AWS CloudHSM 金鑰存放區，但無法復原刪除操作。

您只能刪除與其 AWS CloudHSM 叢集中斷連線且不包含任何的 AWS CloudHSM 金鑰存放區 AWS KMS keys。刪除自訂金鑰存放區之前，請執行下列動作。

- 驗證您將不再需要使用金鑰存放區中的任何 KMS 金鑰進行任何[密碼編譯操作](#)。然後，[排程從金鑰存放區刪除](#)所有 KMS 金鑰。如需在 金鑰存放區中尋找 KMS AWS CloudHSM 金鑰的說明，請參閱 [在金鑰存放區中尋找 KMS AWS CloudHSM 金鑰](#)。
- 確認已刪除所有 KMS 金鑰。若要檢視 金鑰存放區中的 KMS AWS CloudHSM 金鑰，請參閱 [the section called “識別金鑰存放區中的 KMS AWS CloudHSM 金鑰”](#)。
- [中斷 AWS CloudHSM 金鑰存放區與其叢集的連線](#)。AWS CloudHSM

請考慮[中斷其與其相關聯叢集的連線](#)，而不是刪除 AWS CloudHSM 金鑰存放區。AWS CloudHSM 當 AWS CloudHSM 金鑰存放區中斷連線時，您可以管理 AWS CloudHSM 金鑰存放區及其 AWS KMS keys。但您無法在金鑰存放區中建立或使用 KMS AWS CloudHSM 金鑰。您可以隨時重新連接 AWS CloudHSM 金鑰存放區。

刪除您的 AWS CloudHSM 金鑰存放區

您可以在 AWS KMS 主控台或使用 [DeleteCustomKeyStore](#) 操作來刪除 AWS CloudHSM 金鑰存放區。

使用 AWS KMS 主控台

若要刪除 中的 AWS CloudHSM 金鑰存放區 AWS 管理主控台，請先從自訂 AWS CloudHSM 金鑰存放區頁面中選取金鑰存放區。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格依次選擇自訂金鑰存放區、AWS CloudHSM 金鑰存放區。
4. 尋找代表您要刪除之 AWS CloudHSM 金鑰存放區的列。如果 AWS CloudHSM 金鑰存放區的連線狀態未中斷連線，您必須在刪除[AWS CloudHSM 金鑰存放區之前中斷連線](#)。
5. 從 Key store actions (金鑰存放區動作) 選單中，選擇 Delete (刪除)。

當操作完成時，會顯示成功訊息，且 AWS CloudHSM 金鑰存放區不會再出現在金鑰存放區清單中。如果操作失敗，就會出現錯誤訊息，其中描述問題並提供如何修正的協助。如果您需要更多協助，請參閱[對自訂金鑰存放區進行故障診斷](#)。

使用 AWS KMS API

若要刪除 AWS CloudHSM 金鑰存放區，請使用 [DeleteCustomKeyStore](#) 操作。如果操作成功，會 AWS KMS 傳回 HTTP 200 回應和沒有屬性的 JSON 物件。

若要開始，請確認 AWS CloudHSM 金鑰存放區不包含任何 AWS KMS keys。您無法刪除包含 KMS 金鑰的自訂金鑰存放區。第一個範例命令使用 [ListKeys](#) 和 [DescribeKey](#)，以範例 *cks-1234567890abcdef0* 自訂金鑰存放區 ID 在 AWS CloudHSM 金鑰存放區 AWS KMS keys 中搜尋。在此情況下，命令不會傳回任何 KMS 金鑰。如果有傳回，請使用 [ScheduleKeyDeletion](#) 操作來排程每個 KMS 金鑰的刪除。

Bash

```
for key in $(aws kms list-keys --query 'Keys[*].KeyId' --output text) ;  
do aws kms describe-key --key-id $key |  
grep '"CustomKeyIdStoreId": "cks-1234567890abcdef0"' --context 100; done
```

PowerShell

```
PS C:\> Get-KMSKeyList | Get-KMSKey | where CustomKeyIdStoreId -eq  
'cks-1234567890abcdef0'
```

接著，中斷連接 AWS CloudHSM 金鑰存放區。此範例命令使用 [DisconnectCustomKeyStore](#) 操作來中斷 AWS CloudHSM 金鑰存放區與其 AWS CloudHSM 叢集的連線。執行此命令之前，請將範例自訂金鑰存放區 ID 以有效的 ID 取代。

Bash

```
$ aws kms disconnect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

PowerShell

```
PS C:\> Disconnect-KMSCustomKeyStore -CustomKeyIdStoreId cks-1234567890abcdef0
```

自訂金鑰存放區中斷連接之後，您就可以使用 [DeleteCustomKeyStore](#) 操作將其刪除。

Bash

```
$ aws kms delete-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

PowerShell

```
PS C:\> Remove-KMSCustomKeyStore -CustomKeyIdStoreId cks-1234567890abcdef0
```

對自訂金鑰存放區進行故障診斷

AWS CloudHSM 金鑰存放區旨在提供可用性和彈性。不過，您可能需要修復一些錯誤條件，才能維持 AWS CloudHSM 金鑰存放區的運作。

主題

- [如何修正無法使用的 KMS 金鑰](#)
- [如何修正失效的 KMS 金鑰](#)
- [如何修正連線失敗](#)
- [如何回應密碼編譯操作失敗](#)
- [如何修正無效的 kmsuser 登入資料](#)
- [如何刪除遺棄的金鑰材料](#)
- [如何復原 KMS 金鑰已刪除的金鑰材料](#)
- [如何以 kmsuser 身分登入](#)

如何修正無法使用的 KMS 金鑰

金鑰存放區 AWS KMS keys 中 [的金鑰狀態](#)通常是 Enabled。AWS CloudHSM 如同所有 KMS 金鑰，當您停用金鑰存放區中的 KMS 金鑰或排定刪除金鑰時，AWS CloudHSM 金鑰狀態會變更。不過，與其他 KMS 金鑰不同，自訂金鑰存放區中的 KMS 金鑰也可以有 Unavailable 的[金鑰狀態](#)。

Unavailable 的金鑰狀態表示自訂金鑰存放區中的 KMS 金鑰是特意[中斷連接](#)，並且會在失敗時 (如果有) 嘗試重新連接。當 KMS 金鑰無法使用時，您可以檢視和管理 KMS 金鑰，但無法將它用於[密碼編譯操作](#)。

若要尋找 KMS 金鑰的金鑰狀態，請在 Customer managed keys (客戶受管金鑰) 頁面上，檢視 KMS 金鑰的 Status (狀態) 欄位。或使用 [DescribeKey](#) 操作並檢視回應中的 KeyState 元素。如需詳細資訊，請參閱[識別和檢視金鑰](#)。

中斷連線的自訂金鑰存放區中的 KMS 金鑰會有 Unavailable 或 PendingDeletion 的金鑰狀態。排程要從自訂金鑰存放區刪除的 KMS 金鑰會有 Pending Deletion 金鑰狀態，即使自訂金鑰存放區中斷連接時亦然。這可讓您取消排程的金鑰刪除，而無需重新連接自訂金鑰存放區。

若要修正無法使用的 KMS 金鑰，請[重新連線自訂金鑰存放區](#)。自訂金鑰存放區重新連接之後，自訂金鑰存放區中 KMS 金鑰的金鑰狀態會自動還原到其先前的狀態，例如 Enabled 或 Disabled。等待刪除的 KMS 金鑰會保持在 PendingDeletion 狀態。不過，當問題存在時，[啟用和停用無法使用的 KMS 金鑰](#)不會變更它的金鑰狀態。啟用或停用動作僅在金鑰可供使用時生效。

如需失敗的連接的協助，請參閱[如何修正連線失敗](#)。

如何修正失效的 KMS 金鑰

在 AWS CloudHSM 金鑰存放區中建立和使用 KMS 金鑰的問題可能是由 AWS CloudHSM 金鑰存放區、其關聯的 AWS CloudHSM 叢集、KMS 金鑰或其金鑰材料的問題所造成。

當 AWS CloudHSM 金鑰存放區與其 AWS CloudHSM 叢集中斷連線時，自訂金鑰存放區中 KMS 金鑰的金鑰狀態為 Unavailable。在中斷連線的金鑰存放區中建立 KMS AWS CloudHSM 金鑰的所有請求都會傳回 CustomKeyStoreInvalidStateException 例外狀況。加密、解密、重新加密或產生資料金鑰的所有請求會傳回 KMSInvalidStateException 例外狀況。若要修正此問題，請[重新連接 AWS CloudHSM 金鑰存放區](#)。

不過，即使 AWS CloudHSM 金鑰狀態為 Enabled 且金鑰存放區的連線狀態為 `Connected`，您嘗試在金鑰存放區中使用 KMS AWS CloudHSM 金鑰進行[密碼編譯操作](#)仍可能會失敗。這可能是因為以下任何情況所造成。

- 可能已從相關聯的 AWS CloudHSM 叢集刪除 KMS 金鑰的金鑰材料。若要調查，請[尋找 KMS 金鑰之金鑰材料的金鑰 ID](#)，並視需要嘗試[復原金鑰材料](#)。
- 所有 HSMs 都已從與 AWS CloudHSM 金鑰存放區相關聯的 AWS CloudHSM 叢集中刪除。若要在密碼編譯操作的 AWS CloudHSM 金鑰存放區中使用 KMS 金鑰，其 AWS CloudHSM 叢集必須至少包含一個作用中的 HSM。若要驗證 AWS CloudHSM 叢集中 HSMs 的數量和狀態，請[使用 AWS CloudHSM 主控台](#)或 [DescribeClusters](#) 操作。若要將 HSM 新增至叢集，請使用 AWS CloudHSM 主控台或 [CreateHsm](#) 操作。
- 已刪除與 AWS CloudHSM 金鑰存放區相關聯的 AWS CloudHSM 叢集。若要修正此問題，請從與原始叢集相關的備份（例如，原始叢集的備份，或用於建立原始叢集的備份）[建立叢集](#)。然後在自訂金鑰存放區設定中[編輯叢集 ID](#)。如需說明，請參閱[如何復原 KMS 金鑰已刪除的金鑰材料](#)。
- 與自訂金鑰存放區相關聯的 AWS CloudHSM 叢集沒有任何可用的 PKCS #11 工作階段。這通常發生在高爆量流量期間，此時需要額外的的工作階段來服務流量。若要回應帶有關於 PKCS #11 工作階段錯誤訊息的 KMSInternalException，請退回並重試請求。

如何修正連線失敗

如果您嘗試將 [AWS CloudHSM 金鑰存放區](#) 連接至其 AWS CloudHSM 叢集，但操作失敗，則 AWS CloudHSM 金鑰存放區的連線狀態會變更為 FAILED。若要尋找 AWS CloudHSM 金鑰存放區的連線狀態，請使用 AWS KMS 主控台或 [DescribeCustomKeyStores](#) 操作。

或者，由於很容易偵測到叢集組態錯誤，一些連接嘗試會很快失敗。在這種情況下，連接狀態仍然是 DISCONNECTED。這些失敗會傳回錯誤訊息或 [例外狀況](#) 來說明嘗試失敗的原因。檢閱例外狀況描述和 [叢集需求](#)、修正問題、視需要 [更新 AWS CloudHSM 金鑰存放區](#)，然後再次嘗試連線。

當連接狀態為 FAILED 時，請執行 [DescribeCustomKeyStores](#) 操作，並查看回應中的 ConnectionErrorCode 元素。

Note

當 AWS CloudHSM 金鑰存放區的連線狀態為 FAILED 時，您必須先 [中斷連接 AWS CloudHSM 金鑰存放區](#)，然後再嘗試重新連線。您無法以連線狀態 FAILED 連接 AWS CloudHSM 金鑰存放區。

- CLUSTER_NOT_FOUND 表示 AWS KMS 找不到具有指定 AWS CloudHSM 叢集 ID 的叢集。發生此情況可能是因為提供給 API 操作的是錯誤的叢集 ID 或是已刪除叢集而無法取代。若要修正此錯誤，請驗證叢集 ID，例如使用 AWS CloudHSM 主控台或 [DescribeClusters](#) 操作。如果已刪除叢集，請從原始叢集的 [最近備份建立叢集](#)。然後，[中斷連接 AWS CloudHSM 金鑰存放區](#)、[編輯 AWS CloudHSM 金鑰存放區](#) 叢集 ID 設定，然後將 [AWS CloudHSM 金鑰存放區重新](#) 連線至叢集。
- INSUFFICIENT_CLOUDHSM_HSMs 表示相關聯的 AWS CloudHSM 叢集不包含任何 HSMs。若要連接，叢集必須至少有一個 HSM。若要尋找叢集中的 HSM 數量，請使用 [DescribeClusters](#) 操作。若要解決此錯誤，請 [新增至少一個 HSM](#) 到叢集。如果您新增多個 HSM，最好在不同的可用區域建立它們。
- INSUFFICIENT_FREE_ADDRESSES_IN_SUBNET 表示 AWS KMS 無法將 AWS CloudHSM 金鑰存放區連接至其 AWS CloudHSM 叢集，因為至少一個 [與叢集相關聯的私有子網路](#) 沒有任何可用的 IP 地址。AWS CloudHSM 金鑰存放區連線在每個相關聯的私有子網路中需要一個可用的 IP 地址，但最好有兩個。

您 [無法新增 IP 地址](#) (CIDR 區塊) 至現有的子網路。如果可能，請移動或刪除子網路中使用 IP 地址的其他資源，例如未使用的 EC2 執行個體或彈性網路介面。否則，您可以從 [最近備份的叢集建立叢集](#)，該 AWS CloudHSM 叢集具有新的或現有的私有子網路，這些子網路具有 [更多可用的地址空間](#)。然後，若要將新叢集與您的 AWS CloudHSM 金鑰存放區建立關聯，請 [中斷連接自訂金鑰存放區](#)，將 [金鑰存放區的叢集 ID 變更為](#) 新叢集的 ID，然後再次嘗試連線。AWS CloudHSM

Tip

若要避免 [重設密碼 kmsuser](#)，請使用 AWS CloudHSM 叢集的最新備份。

- `INTERNAL_ERROR` 表示 AWS KMS 因內部錯誤而無法完成請求。重試 請求。對於 `ConnectCustomKeyStore` 請求，請先中斷連接 AWS CloudHSM 金鑰存放區，然後再嘗試再次連線。
- `INVALID_CREDENTIALS` 表示 AWS KMS 無法登入相關聯的 AWS CloudHSM 叢集，因為它沒有正確的 `kmsuser` 帳戶密碼。如需此錯誤的協助，請參閱 [如何修正無效的 `kmsuser` 登入資料](#)。
- `NETWORK_ERRORS` 通常指出暫時性的網路問題。 [中斷連接 AWS CloudHSM 金鑰存放區](#)，等待幾分鐘，然後再次嘗試連線。
- `SUBNET_NOT_FOUND` 表示 AWS CloudHSM 叢集組態中至少有一個子網路已刪除。如果 AWS KMS 找不到叢集組態中的所有子網路，會嘗試將 AWS CloudHSM 金鑰存放區連線至 AWS CloudHSM 叢集失敗。

若要修正此錯誤，[請從相同叢集的最近備份建立](#) AWS CloudHSM 叢集。(此程序會建立具有 VPC 和私有子網路的新叢集組態)。請確認新叢集符合 [自訂金鑰存放區的需求](#)，並記下新的叢集 ID。然後，若要將新叢集與您的 AWS CloudHSM 金鑰存放區建立關聯，[請中斷連接自訂金鑰存放區](#)，將 AWS CloudHSM 金鑰存放區的 [叢集 ID 變更為](#) 新叢集的 ID，然後再次嘗試連線。

 Tip

若要避免 [重設密碼 `kmsuser`](#)，請使用 AWS CloudHSM 叢集的最新備份。

- `USER_LOCKED_OUT` 指出 [kmsuser 加密使用者 \(CU\) 帳戶](#) 因為有太多失敗的密碼嘗試，已鎖定在相關聯的 AWS CloudHSM 叢集之外。如需此錯誤的協助，請參閱 [如何修正無效的 `kmsuser` 登入資料](#)。

若要修正此錯誤，[請中斷連接 AWS CloudHSM 金鑰存放區](#)，並使用 CloudHSM CLI 中的 [使用者 `change-password`](#) 命令來變更 `kmsuser` 帳戶密碼。然後編輯自訂金鑰存放區的 [kmsuser 密碼設定](#)，並嘗試重新連接。如需協助，請使用 [如何修正無效的 `kmsuser` 登入資料](#) 主題中描述的程序。

- `USER_LOGGED_IN` 表示 CU `kmsuser` 帳戶已登入相關聯的 AWS CloudHSM 叢集。這 AWS KMS 可防止 輪換 `kmsuser` 帳戶密碼和登入叢集。若要修正這個錯誤，請將 `kmsuser` CU 登出叢集。如果您變更 `kmsuser` 密碼以登入叢集，也必須更新金鑰存放區的 AWS CloudHSM 金鑰存放區密碼值。如需協助，請參閱 [如何登出和重新連線](#)。
- `USER_NOT_FOUND` 表示 在關聯的 AWS CloudHSM 叢集中 AWS KMS 找不到 `kmsuser` CU 帳戶。若要修正此錯誤，請在叢集中 [建立 `kmsuser` CU 帳戶](#)，然後 [更新金鑰存放區的金鑰存放區密碼值](#)。AWS CloudHSM 如需協助，請參閱 [如何修正無效的 `kmsuser` 登入資料](#)。

如何回應密碼編譯操作失敗

在自訂金鑰存放區採用 KMS 金鑰的密碼編譯操作可能失敗，並顯示 `KMSInvalidStateException`。下列錯誤訊息可能會伴隨 `KMSInvalidStateException`。

KMS 無法與 CloudHSM 叢集通訊。這可能是暫時性的網路問題。如果您重複看到此錯誤，請確認 AWS CloudHSM 叢集 VPC 的網路 ACLs 和安全群組規則正確無誤。

- 雖然這是 HTTPS 400 錯誤，但可能是因為暫時性網路問題所造成的。若要回應，請先重試請求。不過，如果繼續失敗，請檢查聯網元件的組態。此錯誤很可能是因為聯網元件設定錯誤所造成，例如防火牆規則或 VPC 安全群組規則封鎖傳出流量。例如，KMS 無法透過 IPv6 與 AWS CloudHSM 叢集通訊。如需先決條件的詳細資訊，請參閱 [建立 AWS CloudHSM 金鑰存放區](#)。

KMS 無法與您的 AWS CloudHSM 叢集通訊，因為 `kmsuser` 已鎖定。如果您重複看到此錯誤，請中斷連接 AWS CloudHSM 金鑰存放區並重設 `kmsuser` 帳戶密碼。更新自訂金鑰存放區的 `kmsuser` 密碼並重試請求。

- 此錯誤訊息指出 [kmsuser 加密使用者 \(CU\) 帳戶](#) 因密碼嘗試多次失敗，已鎖定在關聯的 AWS CloudHSM 叢集之外。如需此錯誤的協助，請參閱 [如何中斷連線和登入](#)。

如何修正無效的 `kmsuser` 登入資料

當您[連接 AWS CloudHSM 金鑰存放區](#)時，會以[kmsuser 加密使用者 \(CU\)](#) 身分 AWS KMS 登入相關聯的 AWS CloudHSM 叢集。它會保持登入狀態，直到 AWS CloudHSM 金鑰存放區中斷連線為止。[DescribeCustomKeyStores](#) 回應顯示 `FAILED` 的 `ConnectionState` 和 `INVALID_CREDENTIALS` 的 `ConnectionErrorCode` 值，如以下範例所示。

如果您中斷連接 AWS CloudHSM 金鑰存放區並變更 `kmsuser` 密碼，AWS KMS 無法使用 CU `kmsuser` 帳戶的登入資料登入 AWS CloudHSM 叢集。因此，所有連線 AWS CloudHSM 金鑰存放區的嘗試都會失敗。`DescribeCustomKeyStores` 回應顯示 `FAILED` 的 `ConnectionState` 和 `INVALID_CREDENTIALS` 的 `ConnectionErrorCode` 值，如以下範例所示。

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleKeyStore
{
```



```

"CustomKeyStores": [
  "CloudHsmClusterId": "cluster-1a23b4cdefg",
  "ConnectionErrorCode": "INVALID_CREDENTIALS"
  "CustomKeyStoreId": "cks-1234567890abcdef0",
  "CustomKeyStoreName": "ExampleKeyStore",
  "TrustAnchorCertificate": "<certificate string appears here>",
  "CreationDate": "1.499288695918E9",
  "ConnectionState": "FAILED"
],
}

```

此外，在以不正確的密碼嘗試登入叢集失敗五次之後，AWS CloudHSM 會鎖定該使用者帳戶。若要登入叢集，您必須變更帳戶的密碼。

如果嘗試以 CU kmsuser 身分登入叢集時 AWS KMS 取得鎖定回應，連線 AWS CloudHSM 金鑰存放區的請求會失敗。[DescribeCustomKeyStores](#) 回應包含 FAILED 的 ConnectionState 和 USER_LOCKED_OUT 的 ConnectionErrorCode 值，如以下範例所示。

```

$ aws kms describe-custom-key-stores --custom-key-store-name ExampleKeyStore
{
  "CustomKeyStores": [
    "CloudHsmClusterId": "cluster-1a23b4cdefg",
    "ConnectionErrorCode": "USER_LOCKED_OUT"
    "CustomKeyStoreId": "cks-1234567890abcdef0",
    "CustomKeyStoreName": "ExampleKeyStore",
    "TrustAnchorCertificate": "<certificate string appears here>",
    "CreationDate": "1.499288695918E9",
    "ConnectionState": "FAILED"
  ],
}

```

若要修正任一個情況，請使用下列程序。

1. [中斷連接 AWS CloudHSM 金鑰存放區](#)。
2. 執行 [DescribeCustomKeyStores](#) 操作，並檢視回應中 ConnectionErrorCode 元素的值。
 - 如果 ConnectionErrorCode 值為 INVALID_CREDENTIALS，請判斷 kmsuser 帳戶目前的密碼。如有必要，請使用 CloudHSM CLI 中的 [使用者 change-password](#) 命令，將密碼設定為已知值。
 - 如果 ConnectionErrorCode 值為 USER_LOCKED_OUT，您必須使用 CloudHSM CLI 中的 [使用者 change-password](#) 命令來變更 kmsuser 密碼。

3. [編輯 kmsuser 密碼設定](#)，讓它符合叢集中目前的 kmsuser 密碼。此動作可告知 AWS KMS 要使用哪個密碼來登入叢集。它不會變更叢集中的 kmsuser 密碼。
4. [連接自訂金鑰存放區](#)。

如何刪除遺棄的金鑰材料

排程從金鑰存放區刪除 KMS AWS CloudHSM 金鑰之後，您可能需要從相關聯的 AWS CloudHSM 叢集手動刪除對應的金鑰材料。

當您在 AWS CloudHSM 金鑰存放區中建立 KMS 金鑰時，會在 AWS KMS 建立 KMS 金鑰中繼資料，AWS KMS 並在相關聯的 AWS CloudHSM 叢集中產生金鑰材料。當您排程刪除 AWS CloudHSM 金鑰存放區中的 KMS 金鑰時，在等待期間之後，會 AWS KMS 刪除 KMS 金鑰中繼資料。然後，AWS KMS 會盡最大努力從 AWS CloudHSM 叢集中刪除對應的金鑰材料。如果 AWS KMS 無法存取叢集，則嘗試可能會失敗，例如當它與 AWS CloudHSM 金鑰存放區中斷連線或 kmsuser 密碼變更時。AWS KMS 不會嘗試從叢集備份中刪除金鑰材料。

AWS KMS 會在 AWS CloudTrail 日誌 DeleteKey 的事件項目中報告嘗試從叢集刪除金鑰材料的結果。它會出現在 additionalEventData 元素的 backingKeysDeletionStatus 元素中，如以下範例項目所示。項目也包含 KMS 金鑰 ARN、AWS CloudHSM 叢集 ID 和金鑰材料的 ID (backing-key-id)。

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "accountId": "111122223333",
    "invokedBy": "AWS Internal"
  },
  "eventTime": "2021-12-10T14:23:51Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "DeleteKey",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "AWS Internal",
  "userAgent": "AWS Internal",
  "requestParameters": null,
  "responseElements": {
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "additionalEventData": {
    "customKeyStoreId": "cks-1234567890abcdef0",
    "clusterId": "cluster-1a23b4cdefg",
```

```

    "backingKeys": "[{\"backingKeyId\":\"backing-key-id\"}]",
    "backingKeysDeletionStatus": "[{\"backingKeyId\":\"backing-key-id\",
    \"deletionStatus\":\"FAILURE\"}]"
  },
  "eventID": "c21f1f47-f52b-4ffe-bff0-6d994403cf40",
  "readOnly": false,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:eu-west-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ],
  "eventType": "AwsServiceEvent",
  "recipientAccountId": "111122223333",
  "managementEvent": true,
  "eventCategory": "Management"
}

```

備註

下列程序使用 AWS CloudHSM 用戶端 SDK 5 命令列工具 [CloudHSM CLI](#)。CloudHSM CLI `key-handle` 會取代為 `key-reference`。

2025 年 1 月 1 日，AWS CloudHSM 將結束對用戶端 SDK 3 命令列工具、CloudHSM 管理公用程式 (CMU) 和金鑰管理公用程式 (KMU) 的支援。如需用戶端 SDK 3 命令列工具和用戶端 SDK 5 命令列工具之間差異的詳細資訊，請參閱 AWS CloudHSM 《使用者指南》中的 [從用戶端 SDK 3 CMU 和 KMU 遷移至用戶端 SDK 5 CloudHSM CLI](#)。

下列程序示範如何從相關聯的 AWS CloudHSM 叢集刪除孤立的金鑰材料。

1. 如果 AWS CloudHSM 金鑰存放區尚未中斷連線，請中斷連線，然後[登入](#)，如中所述[如何中斷連線和登入](#)。

Note

當自訂金鑰存放區中斷連接時，所有在自訂金鑰存放區中建立 KMS 金鑰的嘗試，或在密碼編譯操作中使用現有 KMS 金鑰的嘗試，均會失敗。此動作可防止使用者存放和存取敏感資料。

2. 使用 CloudHSM CLI 中的[金鑰刪除](#)命令，從叢集中的 HSMs 刪除金鑰。

AWS CloudHSM 金鑰存放區中具有 KMS 金鑰之密碼編譯操作的所有 CloudTrail 日誌項目都包含具有 customKeyStoreId 和 additionalEventData 的欄位 backingKey。backingKeyId 欄位中傳回的值是 CloudHSM 金鑰 id 屬性。建議您透過 篩選金鑰刪除操作 id，以刪除您在 CloudTrail 日誌中識別的孤立金鑰材料。

AWS CloudHSM 會將 backingKeyId 值識別為十六進位值。若要依 篩選 id，您必須在前面 backingKeyId 加上 0x。例如，如果 CloudTrail 日誌 backingKeyId 中的是 1a2b3c45678abcdef，您會依 篩選 **0x1a2b3c45678abcdef**。

下列範例會從叢集中的 HSMs 刪除金鑰。backing-key-id 會列在 CloudTrail 日誌項目中。執行此命令之前，請將範例取代 backing-key-id 為您的帳戶中有效的範例。

```
aws-cloudhsm key delete --filter attr.id="0x<backing-key-id>"
{
  "error_code": 0,
  "data": {
    "message": "Key deleted successfully"
  }
}
```

3. 如中所述登出並重新連接 AWS CloudHSM 金鑰存放區[如何登出和重新連線](#)。

如何復原 KMS 金鑰已刪除的金鑰材料

如果刪除 AWS KMS key 的金鑰材料，則 KMS 金鑰無法使用，而且無法在 KMS 金鑰下加密的所有加密文字都無法解密。如果從關聯的 AWS CloudHSM 叢集中刪除金鑰存放區中 KMS 金鑰的 AWS CloudHSM 金鑰材料，就會發生這種情況。不過，復原金鑰材料可能可行。

當您在 AWS CloudHSM 金鑰存放區中建立 AWS KMS key (KMS 金鑰) 時，會 AWS KMS 登入相關聯的 AWS CloudHSM 叢集，並建立 KMS 金鑰的金鑰材料。它也會將密碼變更為只有它知道的值，只要 AWS CloudHSM 金鑰存放區已連線，就會保持登入狀態。由於只有金鑰擁有者 (也就是建立金鑰的 CU) 可以刪除金鑰，幾乎不可能會從 HSM 中不慎刪除金鑰。

不過，如果某個 KMS 金鑰的金鑰材料已從叢集中的 HSM 刪除，KMS 金鑰之金鑰狀態最終會變更為 UNAVAILABLE。如果您嘗試將該 KMS 金鑰用於密碼編譯操作，則該操作會失敗，出現 KMSInvalidStateException 例外狀況。最重要的是，使用該 KMS 金鑰加密的任何資料會無法解密。

在某些情況下，您可以透過[從包含金鑰材料的備份建立叢集](#)來復原已刪除的金鑰材料。此策略僅在當該金鑰存在時並且在刪除它之前建立了至少一個備份的情況下才有用。

使用以下程序來復原金鑰材料。

1. 尋找包含金鑰材料的叢集備份。該備份也必須包含您需要支援叢集及其加密資料的所有使用者和金鑰。

使用 [DescribeBackups](#) 操作來列出叢集的備份。然後使用備份時間戳記來協助您選取備份。若要限制與 AWS CloudHSM 金鑰存放區相關聯的叢集輸出，請使用 `Filters` 參數，如下列範例所示。

```
$ aws cloudhsmv2 describe-backups --filters clusterIds=<cluster ID>
{
  "Backups": [
    {
      "ClusterId": "cluster-1a23b4cdefg",
      "BackupId": "backup-9g87f6edcba",
      "CreateTimestamp": 1536667238.328,
      "BackupState": "READY"
    },
    ...
  ]
}
```

2. [從選取的備份建立叢集](#)。驗證備份包含已刪除的金鑰和叢集所需的其他使用者和金鑰。
3. [中斷連接 AWS CloudHSM 金鑰存放區](#)，以便編輯其屬性。
4. [編輯金鑰存放區的叢集 ID](#)。AWS CloudHSM 輸入您從備份建立之叢集的叢集 ID。由於該叢集與原始叢集共用備份歷史記錄，新叢集 ID 應該是有效的。
5. [重新連接 AWS CloudHSM 金鑰存放區](#)。

如何以 `kmsuser` 身分登入

若要為您的金鑰存放區在 AWS CloudHSM 叢集中建立和管理 AWS CloudHSM 金鑰材料，AWS KMS 會使用 [kmsuser 加密使用者 \(CU\) 帳戶](#)。您可以在叢集中[建立 kmsuser CU 帳戶](#)，並在建立 AWS CloudHSM 金鑰存放區 AWS KMS 時提供其密碼給。

一般而言，會 AWS KMS 管理 `kmsuser` 帳戶。不過，對於某些任務，您需要中斷連接 AWS CloudHSM 金鑰存放區、以 CU `kmsuser` 身分登入叢集，並使用 [CloudHSM 命令列界面 \(CLI\)](#)。

Note

當自訂金鑰存放區中斷連接時，所有在自訂金鑰存放區中建立 KMS 金鑰的嘗試，或在密碼編譯操作中使用現有 KMS 金鑰的嘗試，均會失敗。此動作可防止使用者存放和存取敏感資料。

本主題說明如何[中斷連接您的 AWS CloudHSM 金鑰存放區並以身分登入 kmsuser](#)、執行 AWS CloudHSM 命令列工具，以及[登出並重新連接您的 AWS CloudHSM 金鑰存放區](#)。

主題

- [如何中斷連線和登入](#)
- [如何登出和重新連線](#)

如何中斷連線和登入

每次都需要使用下列程序，以 kmsuser 加密使用者身分登入相關聯的叢集。

備註

下列程序使用 AWS CloudHSM 用戶端 SDK 5 命令列工具 [CloudHSM CLI](#)。CloudHSM CLI key-handle 會取代為 key-reference。

2025 年 1 月 1 日，AWS CloudHSM 將結束對用戶端 SDK 3 命令列工具、CloudHSM 管理公用程式 (CMU) 和金鑰管理公用程式 (KMU) 的支援。如需用戶端 SDK 3 命令列工具和用戶端 SDK 5 命令列工具之間差異的詳細資訊，請參閱 AWS CloudHSM 《使用者指南》中的[從用戶端 SDK 3 CMU 和 KMU 遷移至用戶端 SDK 5 CloudHSM CLI](#)。

1. 如果 AWS CloudHSM 金鑰存放區尚未中斷連線，請中斷其連線。您可以使用 AWS KMS 主控台或 AWS KMS API。

當您的 AWS CloudHSM 金鑰連線時，AWS KMS 會以身分登入 kmsuser。如此會防止您以 kmsuser 身分登入或變更 kmsuser 密碼。

例如，此命令使用 [DisconnectCustomKeyStore](#) 來中斷連接範例金鑰存放區。將範例 AWS CloudHSM 金鑰存放區 ID 取代為有效的 ID。

```
$ aws kms disconnect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

2. 使用登入命令以管理員身分登入。使用 AWS CloudHSM 使用者指南[的使用 CloudHSM CLI](#) 一節中所述的程序。

```
aws-cloudhsm > login --username admin --role admin
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "admin",
    "role": "admin"
  }
}
```

3. 使用 CloudHSM CLI 中的[使用者 change-password](#) 命令，將kmsuser帳戶的密碼變更為您知道的密碼。(在您連接 AWS CloudHSM 金鑰存放區時AWS KMS 輪換密碼。)密碼必須包含 7 到 32 個英數字元。區分大小寫，且不能包含任何特殊字元。
4. kmsuser 使用您設定的密碼登入為。如需詳細說明，請參閱AWS CloudHSM 《使用者指南》中的[使用 CloudHSM CLI](#) 一節。

```
aws-cloudhsm > login --username kmsuser --role crypto-user
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "kmsuser",
    "role": "crypto-user"
  }
}
```

如何登出和重新連線

每次您需要以kmsuser加密使用者身分登出並重新連接金鑰存放區時，請使用下列程序。

備註

下列程序使用 AWS CloudHSM 用戶端 SDK 5 命令列工具 [CloudHSM CLI](#)。CloudHSM CLI key-handle會取代為 key-reference。

2025 年 1 月 1 日，AWS CloudHSM 將結束對用戶端 SDK 3 命令列工具、CloudHSM 管理公用程式 (CMU) 和金鑰管理公用程式 (KMU) 的支援。如需用戶端 SDK 3 命令列工具和用戶端

SDK 5 命令列工具之間差異的詳細資訊，請參閱AWS CloudHSM 《使用者指南》中的[從用戶端 SDK 3 CMU 和 KMU 遷移至用戶端 SDK 5 CloudHSM CLI](#)。

1. 執行任務，然後使用 CloudHSM CLI 中的[登出](#)命令來登出。如果您不登出，嘗試重新連接您的 AWS CloudHSM 金鑰存放區將會失敗。

```
aws-cloudhsm  logout
{
  "error_code": 0,
  "data": "Logout successful"
}
```

2. [編輯自訂金鑰存放區的 kmsuser 密碼設定](#)。

這會告知叢集kmsuser中的 AWS KMS 目前密碼。如果您省略此步驟，AWS KMS 將無法以身分登入叢集kmsuser，而且所有重新連線自訂金鑰存放區的嘗試都會失敗。您可以使用 AWS KMS 主控台或 [UpdateCustomKeyStore](#) 操作的 KeyStorePassword 參數。

例如，此命令 AWS KMS 會告知目前的密碼為 tempPassword。將範例密碼以實際密碼取代。

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 --key-store-password tempPassword
```

3. 將 AWS KMS 金鑰存放區重新連線至其 AWS CloudHSM 叢集。將範例 AWS CloudHSM 金鑰存放區 ID 取代為有效的 ID。在連線過程中，會將kmsuser密碼 AWS KMS 變更為只有它知道的值。

[ConnectCustomKeyStore](#) 操作會很快傳回，但連接程序可能需要一段很長的時間。初始回應並不表示連接程序已成功。

```
$ aws kms connect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

4. 使用 [DescribeCustomKeyStores](#) 操作來驗證 AWS CloudHSM 金鑰存放區是否已連線。將範例 AWS CloudHSM 金鑰存放區 ID 取代為有效的 ID。

在此範例中，連線狀態欄位會顯示 AWS CloudHSM 金鑰存放區現在已連線。

```
$ aws kms describe-custom-key-stores --custom-key-store-id cks-1234567890abcdef0
{
```



```
"CustomKeyStores": [  
  "CustomKeyStoreId": "cks-1234567890abcdef0",  
  "CustomKeyStoreName": "ExampleKeyStore",  
  "CloudHsmClusterId": "cluster-1a23b4cdefg",  
  "TrustAnchorCertificate": "<certificate string appears here>",  
  "CreationDate": "1.499288695918E9",  
  "ConnectionState": "CONNECTED"  
],  
}
```

外部金鑰存放區

外部金鑰存放區可讓您使用 外部的密碼編譯金鑰來保護 AWS 資源 AWS。此進階功能專為受管制的工作負載而設計，您必須使用儲存在您控制的外部金鑰管理系統中的加密金鑰來保護這些工作負載。外部金鑰存放區支援[AWS 數位主權承諾](#)，讓您對 中的資料進行主權控制 AWS，包括能夠使用您擁有的金鑰材料進行加密，以及在外部控制 AWS。

外部金鑰存放區是由您擁有和管理的外部金鑰管理器支援的[自訂金鑰存放區](#) AWS。您的外部金鑰管理器可以是實體或虛擬硬體安全模組 (HSM)，也可以是任何能夠產生和使用密碼編譯金鑰的硬體型或軟體型系統。在外部金鑰存放區中使用 KMS 金鑰的加密和解密操作由外部金鑰管理器使用您的密碼編譯金鑰材料執行，這項功能稱為 Hold Your Own Keys (HYOK)。

AWS KMS 永遠不會直接與您的外部金鑰管理器互動，也無法建立、檢視、管理或刪除您的金鑰。反之，只會與您提供的[外部金鑰存放區代理](#) (XKS 代理) 軟體 AWS KMS 互動。您的外部金鑰存放區代理會媒介 AWS KMS 與外部金鑰管理器之間的所有通訊。它將所有請求從 傳輸 AWS KMS 到您的外部金鑰管理器，並將來自外部金鑰管理器的回應傳輸回 AWS KMS。外部金鑰存放區代理也會將來自 的一般請求轉譯 AWS KMS 為外部金鑰管理器可以理解的廠商特定格式，讓您可以將外部金鑰存放區與來自各種廠商的金鑰管理器搭配使用。

您可以在外部金鑰存放區中使用 KMS 金鑰進行用戶端加密，包括 [AWS Encryption SDK](#)。但是，外部金鑰存放區是伺服器端加密的重要資源，可讓您在 外部 AWS 服務 使用密碼編譯金鑰來保護多個 中的 AWS 資源 AWS。AWS 服務 支援對稱加密的[客戶受管金鑰](#)也支援外部金鑰存放區中的 KMS 金鑰。如需服務支援詳細資訊，請參閱 [AWS 服務整合](#)。

外部金鑰存放區可讓您 AWS KMS 將 用於受管制的工作負載，其中加密金鑰必須在外部存放和使用 AWS。但其與標準共同責任模式有很大不同，並且需要額外的運營負擔。對於大多數客戶而言，可用性和延遲的風險更大，將超過外部金鑰存放區的安全優勢。

外部金鑰存放區可讓您控制信任的根源。只有使用您控制的外部金鑰管理器，才能解密使用外部金鑰存放區中 KMS 金鑰加密的資料。如果您暫時撤銷對外部金鑰管理器的存取，例如中斷外部金鑰存放區連線，或中斷外部金鑰管理器與外部金鑰存放區代理的連線，會 AWS 失去對密碼編譯金鑰的所有存取，直到您將其還原為止。在該間隔期間，不能解密使用 KMS 金鑰加密的密文。如果您永久撤銷對外部金鑰管理器的存取權，則使用外部金鑰存放區中的 KMS 金鑰加密的所有密文都將無法復原。唯一的例外是短暫快取受 KMS [金鑰保護的資料](#) 金鑰 AWS 的服務。這些資料金鑰會繼續運作，直到您停用資源或快取到期為止。如需詳細資訊，請參閱[無法使用的 KMS 金鑰如何影響資料金鑰](#)。

外部金鑰存放區會解除封鎖受管制工作負載的少數使用案例，其中加密金鑰必須完全由您控制且無法存取 AWS。但是，這是雲端基礎架構操作方式的重大變化，也是共同責任模式的明顯轉變。對於大多數工作負載而言，額外的操作負擔以及可用性和效能的更大風險將超過外部金鑰存放區感知的安全優勢。

我需要外部金鑰存放區嗎？

對於大多數使用者而言，受 [FIPS 140-3 安全層級 3 驗證的硬體安全模組](#) 保護的預設 AWS KMS 金鑰存放區可滿足其安全、控制和法規要求。外部金鑰存放區使用者會產生巨大的成本、維護和疑難排解負擔，以及延遲、可用性和可靠性風險。

考量外部金鑰存放區時，請花一些時間了解替代方案，包括您擁有和管理的 AWS CloudHSM 叢集支援的 [AWS CloudHSM 金鑰存放區](#)，以及在您自己的 HSM 中產生且可視需要從 KMS 金鑰中刪除的具有 [匯入金鑰材料](#) 的 KMS 金鑰。特別是，匯入有效期非常短的金鑰材料可能會提供類似的控制層級，而不會造成效能或可用性風險。

如果您有下列需求，外部金鑰存放區可能是您組織的正確解決方案：

- 您需要在內部部署金鑰管理器或您 AWS 控制範圍之外的金鑰管理器中使用密碼編譯金鑰。
- 必須證明您的密碼編譯金鑰在雲端之外完全由您控制。
- 您必須使用具有獨立授權的密碼編譯金鑰來進行加密和解密。
- 金鑰資料必須放在次要、獨立的稽核路徑。

如果您選擇外部金鑰存放區，則請將其使用限制於需要利用 AWS 之外的密碼編譯金鑰來保護的工作負載。

共同責任模型

標準 KMS 金鑰使用在 AWS KMS 擁有和管理 HSMs 中產生和使用的金鑰材料。您可以在 KMS 金鑰上建立存取控制政策，並設定 AWS 服務使用 KMS 金鑰來保護資源。AWS KMS 會負責 KMS 金鑰中金鑰材料的安全性、可用性、延遲和耐久性。

外部金鑰存放區中的 KMS 金鑰依賴於外部金鑰管理器中的金鑰材料和操作。因此，責任的天平會朝著您的方向移動。您要負責外部金鑰管理器中密碼編譯金鑰的安全性、可靠性、耐用性和效能。AWS KMS 負責立即回應請求，並與外部金鑰存放區代理通訊，並維護我們的安全標準。為了確保每個外部金鑰存放區加密文字至少與標準 AWS KMS 加密文字一樣強，AWS KMS 首先使用 KMS 金鑰特有的 AWS KMS 金鑰材料加密所有純文字，然後將其傳送到您的外部金鑰管理器，以使用外部金鑰加密，此程序稱為雙重加密。因此，無論是 AWS KMS 還是外部金鑰材料擁有者都不能單獨解密雙重加密的密文。

您有責任維護符合法規與效能標準的外部金鑰管理器、供應及維護符合[AWS KMS 外部金鑰存放區代理 API 規格](#)的外部金鑰存放區代理，並確保金鑰資料的可用性與持久性。此外，您也必須建立、設定和維護外部金鑰存放區。當由您維護的元件導致的錯誤發生時，您必須準備好識別和解決錯誤，以便 AWS 服務可以存取您的資源，而不會過度中斷。AWS KMS 提供[故障診斷指引](#)，以協助您判斷問題原因和最可能的解決方法。

檢閱外部金鑰存放區記錄 AWS KMS 的 [Amazon CloudWatch 指標和維度](#)。AWS KMS 強烈建議您建立 CloudWatch 警示來監控外部金鑰存放區，以便在發生效能和操作問題的早期跡象之前對其進行偵測。

有什麼變化？

外部金鑰存放區僅支援對稱加密 KMS 金鑰。在其中 AWS KMS，您使用和管理外部金鑰存放區中的 KMS 金鑰的方式與管理其他[客戶受管金鑰](#)的方式大致相同，包括[設定存取控制政策和監控金鑰使用](#)。您可以使用具有相同參數的相同 API，在用於任何 KMS 金鑰的外部金鑰存放區中使用 KMS 金鑰來請求密碼編譯操作。定價也與標準 KMS 金鑰相同。如需詳細資訊，請參閱 [外部金鑰存放區中的 KMS 金鑰](#)和 [AWS Key Management Service 定價](#)。

但是，對於外部金鑰存放區，以下原則發生了變化：

- 您負責金鑰操作的可用性、耐久性和延遲。
- 您負責開發、購買、操作和授權外部金鑰管理器系統的所有費用。
- 您可以從 AWS KMS 對外部金鑰存放區代理實作所有請求的[獨立授權](#)。
- 您可以監控、稽核和記錄外部金鑰存放區代理的所有操作，以及與 AWS KMS 請求相關的外部金鑰管理器的所有操作。

從何處開始？

若要建立和管理外部金鑰存放區，您需要[選擇外部金鑰存放區代理連接選項](#)、[備妥先決條件](#)以及[建立和設定外部金鑰存放區](#)。

配額

AWS KMS 在每個 AWS 帳戶 和 區域中最多允許 [10 個自訂金鑰存放區](#)，包括[AWS CloudHSM 金鑰存放區](#)和[外部金鑰存放區](#)，無論其連線狀態為何。此外，[在外部金鑰存放區中使用 KMS 金鑰](#)有 AWS KMS 請求配額。

如果您為外部金鑰存放區代理選擇 [VPC 代理連接](#)，則必要元件 (例如 VPC、子網路 and 網路負載平衡器) 可能也會有配額。如需這些配額的相關資訊，請使用 [Service Quotas 主控台](#)。

區域

若要將網路延遲降到最低，請在離[外部金鑰管理器](#)最近的 AWS 區域 中建立外部金鑰存放區元件。如果可能，請選擇網路封包來回時間 (RTT) 為 35 毫秒或更短的區域。

除了中國（北京）和中國（寧夏）以外 AWS 區域 AWS KMS，支援的所有 都支援外部金鑰存放區。

不支援的功能

AWS KMS 不支援自訂金鑰存放區中的下列功能。

- [非對稱 KMS 金鑰](#)
- [HMAC KMS 金鑰](#)
- [含有匯入金鑰資料的 KMS 金鑰](#)
- [自動金鑰輪換](#)
- [多區域金鑰](#)

進一步了解：

- 在 AWS 新聞部落格中[宣告 AWS KMS 外部金鑰存放區](#)。

外部金鑰存放區概念

了解外部金鑰存放區中使用的基本術語和概念。

外部金鑰存放區

外部金鑰存放區是您 AWS 擁有和管理的外部金鑰管理器支援的 AWS KMS [自訂金鑰存放區](#)。外部金鑰存放區中的每個 KMS 金鑰都與外部金鑰管理器中的[外部金鑰](#)相關聯。當您在外部金鑰存放區中使用 KMS 金鑰進行加密或解密時，外部金鑰管理器將使用外部金鑰執行操作，這種安排稱為 Hold your Own Keys (HYOK)。此功能專為需要在其外部金鑰管理器中維護密碼編譯金鑰的組織而設計。

外部金鑰存放區可確保保護您的 AWS 資源的密碼編譯金鑰和操作會保留在您的外部金鑰管理器中，由您控制。AWS KMS 會傳送請求給您的外部金鑰管理器來加密和解密資料，但 AWS KMS 無法建立、刪除或管理任何外部金鑰。從 AWS KMS 到外部金鑰管理器的所有請求，都是由您提供、擁有和管理[的外部金鑰存放區代理](#)軟體元件所媒介。

AWS 支援 AWS KMS [客戶受管金鑰](#)的服務可以使用外部金鑰存放區中的 KMS 金鑰來保護您的資料。因此，使用外部金鑰管理器中的加密操作，您的資料最終會受到金鑰保護。

與標準 KMS 金鑰相比，外部金鑰存放區中的 KMS 金鑰具有本質上不同的信任模型、[共同責任安排](#)和效能預期。透過外部金鑰存放區，您負責金鑰材料和密碼編譯操作的安全性和完整性。外部金鑰存放區中 KMS 金鑰的可用性和延遲會受到硬體、軟體、網路元件以及 AWS KMS 與外部金鑰管理器之間距離的影響。您也可能會對外部金鑰管理器以及外部金鑰管理器與之通訊所需的聯網和負載平衡基礎設施產生額外的成本 AWS KMS

您可以使用外部金鑰存放區作為更廣泛資料保護策略的一部分。對於您保護的每個 AWS 資源，您可以決定哪些需要外部金鑰存放區中的 KMS 金鑰，以及哪些金鑰可以受到標準 KMS 金鑰的保護。這可讓您靈活地為特定資料分類、應用程式或專案選擇 KMS 金鑰。

外部金鑰管理器

外部金鑰管理器是 AWS 外部的元件，其可產生 256 位元 AES 對稱金鑰並執行對稱加密和解密。外部金鑰存放區的外部金鑰管理器可以是實體硬體安全模組 (HSM)、虛擬 HSM，或是含有或不含 HSM 元件的軟體金鑰管理器。它可以位於外部的任何位置 AWS，包括在內部部署、本機或遠端資料中心或任何雲端中。您的外部金鑰存放區可由單一外部金鑰管理器或多個共用密碼編譯金鑰的相關金鑰管理器執行個體 (例如 HSM 叢集) 進行支援。外部金鑰存放區的設計是為了支援來自不同廠商的各種外部管理器。如需連線至外部金鑰管理器的詳細資訊，請參閱 [選擇外部金鑰存放區代理連線選項](#)。

外部金鑰

外部金鑰存放區中的每個 KMS 金鑰都與[外部金鑰管理器](#)中稱為外部金鑰的密碼編譯金鑰相關聯。當您在外部金鑰存放區中使用 KMS 金鑰進行加密或解密時，[外部金鑰管理器](#)將使用外部金鑰執行密碼編譯操作。

⚠ Warning

外部金鑰對於 KMS 金鑰的操作至關重要。如果遺失或刪除外部金鑰，則使用關聯之 KMS 金鑰加密的密文將無法復原。

對於外部金鑰存放區，外部金鑰必須是已啟用且可執行加密和解密動作的 256 位元 AES 金鑰。如需外部金鑰需求的詳細資訊，請參閱 [外部金鑰存放區中 KMS 金鑰的要求](#)。

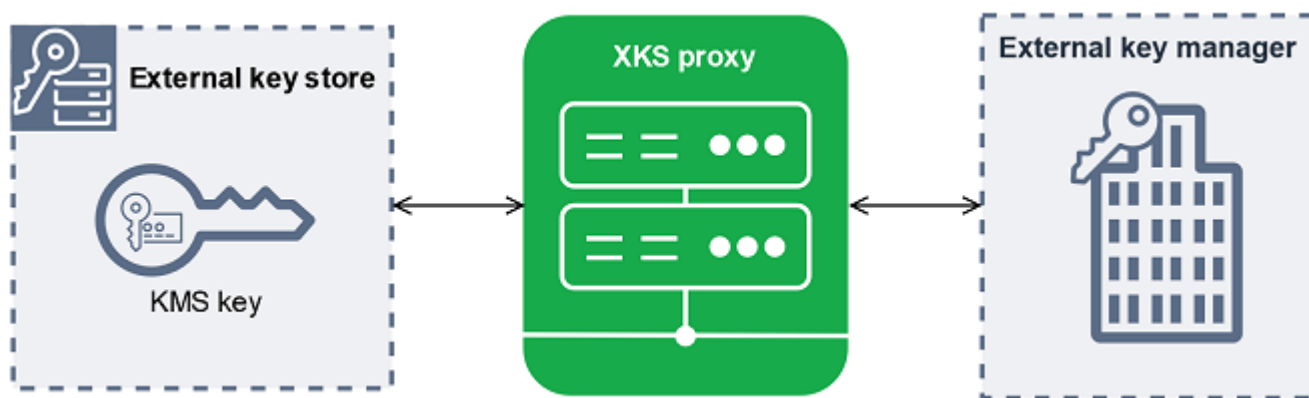
AWS KMS 無法建立、刪除或管理任何外部金鑰。您的密碼編譯金鑰材料永遠不會離開外部金鑰管理器。在外部金鑰存放區中建立 KMS 金鑰時，您需要提供外部金鑰的 ID (XksKeyId)。您無法變更與 KMS 金鑰相關聯的外部金鑰 ID，不過您的外部金鑰管理器可以輪換與外部金鑰 ID 相關聯的金鑰材料。

除了外部金鑰之外，外部金鑰存放區中的 KMS 金鑰也包含 AWS KMS 金鑰材料。受 KMS 金鑰保護的資料會先 AWS KMS 使用 AWS KMS 金鑰材料加密，然後使用外部金鑰由外部金鑰管理器加密。此[雙重加密](#)程序可確保受 KMS 金鑰保護的密文始終至少與僅受 AWS KMS 保護的密文一樣強。

許多密碼編譯金鑰具有不同類型的識別符。在外部金鑰存放區中建立 KMS 金鑰時，請提供[外部金鑰存放區代理](#)用來參照外部金鑰的外部金鑰 ID。如果使用錯誤的識別符，則您嘗試在外部金鑰存放區中建立 KMS 金鑰會失敗。

外部金鑰存放區代理

外部金鑰存放區代理 ("XKS Proxy") 是客戶擁有且客戶受管的軟體應用程式，可協調 AWS KMS 和外部金鑰管理器之間的所有通訊。它也會將一般 AWS KMS 請求轉譯為廠商特定的外部金鑰管理器了解的格式。外部金鑰存放區需要外部金鑰存放區代理。每個外部金鑰存放區與一個外部金鑰存放區代理相關聯。



AWS KMS 無法建立、刪除或管理任何外部金鑰。您的密碼編譯金鑰材料永遠不會離開您的外部金鑰管理器。AWS KMS 與外部金鑰管理器之間的所有通訊均由外部金鑰存放區代理媒介。AWS KMS 會將請求傳送至外部金鑰存放區代理，並接收來自外部金鑰存放區代理的回應。外部金鑰存放區代理負責將請求從 傳輸 AWS KMS 到外部金鑰管理器，以及將回應從外部金鑰管理器傳輸回 AWS KMS。

您擁有並管理外部金鑰存放區的外部金鑰存放區代理，並負責其維護和操作。您可以根據發佈或從廠商購買代理應用程式的開放原始碼[外部金鑰存放區代理 API 規格來開發外部金鑰存放區代理](#)。AWS KMS 您的外部金鑰存放區代理可能包含在您的外部金鑰管理器中。為了支援代理開發，AWS KMS 也提供範例外部金鑰存放區代理 ([aws-kms-xks-proxy](#)) 和測試用戶端 ([xks-kms-xksproxy-test-client](#))，以驗證外部金鑰存放區代理是否符合規格。

為了進行身分驗證 AWS KMS，代理會使用伺服器端 TLS 憑證。若要對您的代理進行身分驗證，會使用 SigV4 [代理身分驗證憑證](#)，向外部金鑰存放區代理 AWS KMS 簽署所有請求。

您的外部金鑰存放區代理必須支援 HTTP/1.1 或更新版本，以及 TLS 1.2 或更新版本，且至少包含下列其中一個加密套件：

- TLS_AES_256_GCM_SHA384 (TLS 1.3)
- TLS_CHACHA20_POLY1305_SHA256 (TLS 1.3)

 Note

AWS GovCloud (US) Region 不支援 TLS_CHACHA20_POLY1305_SHA256。

- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (TLS 1.2)
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (TLS 1.2)

若要在外部金鑰存放區中建立和使用 KMS 金鑰，您必須首先將[外部金鑰存放區連接](#)至其外部金鑰存放區代理。您也可以視需求中斷外部金鑰存放區與其代理的連接。當您這麼做時，外部金鑰存放區中的所有 KMS 金鑰都變得[無法使用](#)；其無法用於任何加密操作。

外部金鑰存放區代理連接

外部金鑰存放區代理連線 (「XKS 代理連線」) 說明 AWS KMS 用來與外部金鑰存放區代理通訊的方法。

您可以在建立外部金鑰存放區時指定代理連接選項，其會成為外部金鑰存放區的屬性。您可以透過更新自訂金鑰存放區屬性來變更代理連接選項，但您必須確定外部金鑰存放區代理仍可存取相同的外部金鑰。

AWS KMS 支援下列連線選項。

- [公有端點連線](#) — 透過網際網路將外部金鑰存放區代理的請求 AWS KMS 傳送至您控制的公有端點。此選項的建立和維護非常簡單，但可能無法滿足每個安裝的安全需求。
- [VPC 端點服務連線](#) — AWS KMS 將請求傳送至您建立和維護的 Amazon Virtual Private Cloud (Amazon VPC) 端點服務。您可以在 Amazon VPC 內託管外部金鑰存放區代理，或在之外託管外部金鑰存放區代理，AWS 並僅使用 Amazon VPC 進行通訊。您也可以將外部金鑰存放區連線至另一個擁有的 Amazon VPC 端點服務 AWS 帳戶。

如需有關外部金鑰存放區代理連接選項的詳細資訊，請參閱 [選擇外部金鑰存放區代理連線選項](#)。

外部金鑰存放區代理身分驗證憑證

若要向外部金鑰存放區代理進行身分驗證，會使用 [Signature V4 \(SigV4\)](#) 身分驗證憑證，將所有請求 AWS KMS 簽署至外部金鑰存放區代理。您可以在代理上建立和維護身分驗證憑證，然後在建立外部存放區 AWS KMS 時將此憑證提供給。

Note

AWS KMS 用來簽署 XKS 代理請求的 SigV4 登入資料與中與 AWS Identity and Access Management 主體相關聯的任何 SigV4 登入資料無關 AWS 帳戶。請勿針對外部金鑰存放區代理重複使用任何 IAM SigV4 憑證。

每個代理身分驗證憑證有兩部分。建立外部金鑰存放區或更新外部金鑰存放區的身分驗證憑證時，您必須同時提供這兩個部分。

- 存取金鑰 ID：識別私密存取金鑰。您可以提供純文字形式的 ID。
- 私密存取金鑰：登入資料的私密部分。在儲存登入資料之前，會在登入資料中 AWS KMS 加密私密存取金鑰。

您可以隨時[編輯憑證設定](#)，例如當您輸入錯誤的值時、變更代理上的憑證時或者代理輪換憑證時。如需外部金鑰存放區代理身分驗證的技術詳細資訊 AWS KMS，請參閱 AWS KMS 外部金鑰存放區代理 API 規格中的身分[驗證](#)。

若要讓您在不斷外部金鑰存放區中使用 KMS 金鑰 AWS 服務的的情況下輪換登入資料，建議您的外部金鑰存放區代理支援至少兩個有效的身分驗證登入資料 AWS KMS。這可確保當您為 AWS KMS 提供新憑證時，先前的憑證可繼續運作。

為了協助您追蹤代理身分驗證憑證的存留期，AWS KMS 會定義 Amazon CloudWatch 指標 [XksProxyCredentialAge](#)。您可以使用此指標建立 CloudWatch 警示，在憑證有效期達到您建立的閾值時通知您。

代理 API

若要支援 AWS KMS 外部金鑰存放區，[外部金鑰存放區代理](#) 必須實作必要的代理 APIs，如 [AWS KMS 外部金鑰存放區代理 API 規格](#) 所述。這些代理 API 請求是唯一 AWS KMS 傳送到代理的請求。雖然您永遠不會直接傳送這些請求，但了解其可能有助於您解決外部金鑰存放區或其代理可能出現的任何問題。例如，在外部金鑰存放區的 [Amazon CloudWatch 指標](#) 中 AWS KMS 包含這些 API 呼叫延遲和成功率的相關資訊。如需詳細資訊，請參閱 [監控外部金鑰存放區](#)。

下表列出並說明每個代理 API。它還包括 AWS KMS 觸發對代理 API 呼叫的操作，以及與代理 API 相關的任何 AWS KMS 操作例外狀況。

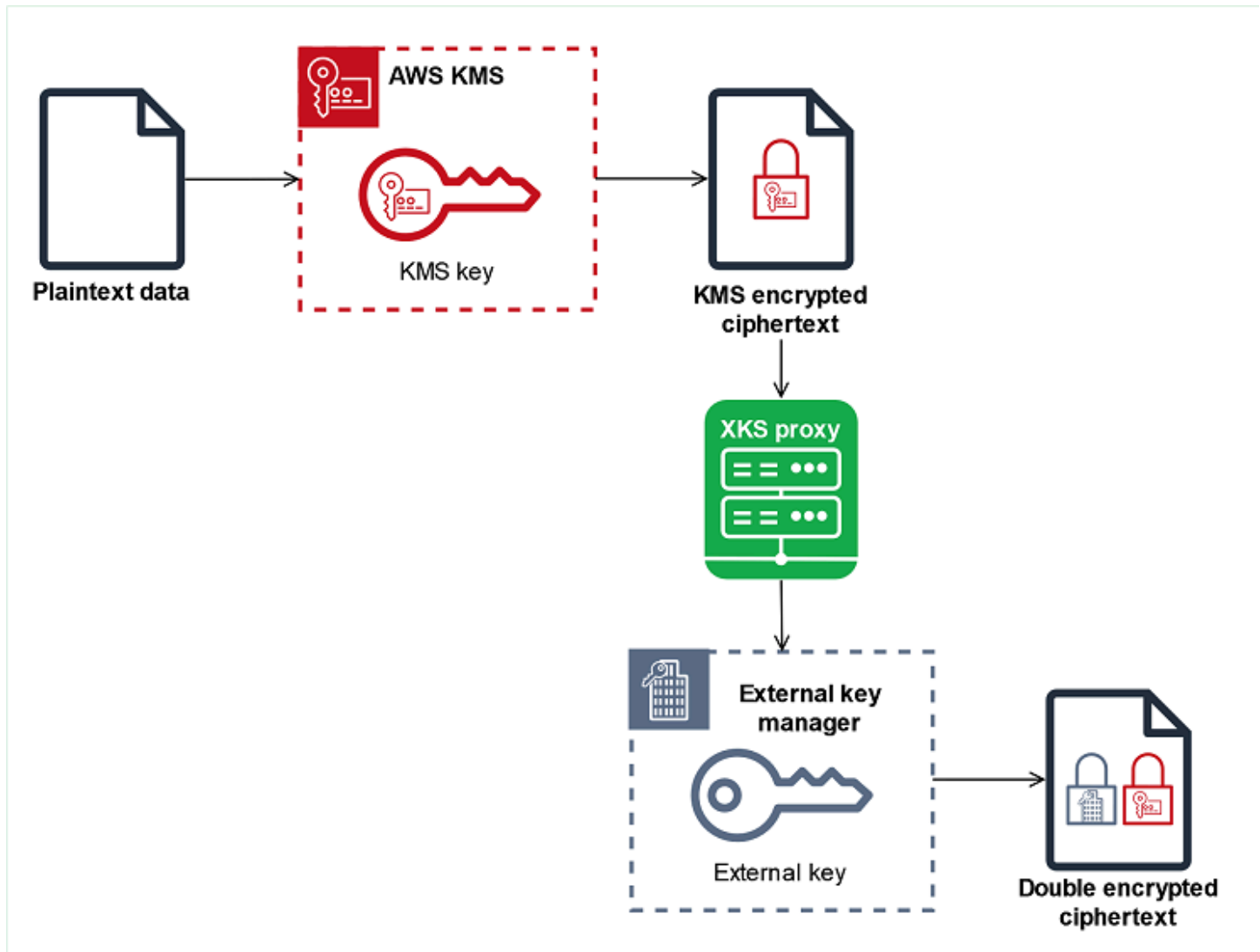
代理 API	Description	相關 AWS KMS 操作
解密	AWS KMS 會傳送要解密的加密文字，以及要使用的 外部金鑰 ID。所需的加密演算法是 AES_GCM。	Decrypt 、 ReEncrypt
加密	AWS KMS 會傳送要加密的資料，以及要使用的 外部金鑰 ID。所需的加密演算法是 AES_GCM。	Encrypt 、 GenerateDataKey 、 GenerateDataKeyWithoutPlaintext 、 ReEncrypt
GetHealthStatus	AWS KMS 會請求有關代理和外部金鑰管理器狀態的資訊。 每個外部金鑰管理器的狀態可以為以下其中一種。 - Active：狀態良好；可以提供流量 - Degraded：狀態不良，但可提供流量 - Unavailable：狀態不良；不能提供流量	CreateCustomKeyStore (用於 公有端點連接)、ConnectCustomKeyStore (用於 VPC 端點服務連接) 如果所有外部金鑰管理器執行個體都是 Unavailable，則嘗試建立或連接金鑰存放區會失敗，且出現 XksProxyUriUnreachableException。

代理 API	Description	相關 AWS KMS 操作
GetKeyMetadata	AWS KMS 會請求與外部金鑰存放區中的 KMS 金鑰相關聯的外部金鑰相關資訊。 回應包括金鑰規格 (AES_256)、金鑰用法 ([ENCRYPT, DECRYPT]) 以及外部金鑰是否為 ENABLED 或 DISABLED。	CreateKey 如果金鑰規格不是 AES_256 , 或金鑰用法不是 [ENCRYPT, DECRYPT] , 或狀態為 DISABLED , 則 CreateKey 操作會失敗, 且會出現 XksKeyInvalidConfigurationException 。

雙重加密

透過外部金鑰存放區中 KMS 金鑰加密的資料會加密兩次。首先, 會使用 KMS AWS KMS 金鑰特定的金鑰材料來 AWS KMS 加密資料。然後, [外部金鑰管理器](#)使用[外部金鑰](#)對 AWS KMS加密的密文進行加密。此過程稱為雙重加密。

雙重加密可確保由外部金鑰存放區中 KMS 金鑰加密的密文至少與使用標準 KMS 金鑰加密的密文一樣強。它也可以保護從 傳輸 AWS KMS 到外部金鑰存放區代理的純文字。使用雙重加密, 您可以完全控制您的密文。如果您透過外部代理永久撤銷對外部金鑰的 AWS 存取權, 則 AWS 中剩餘的任何密文都會被有效地加密銷毀。



若要啟用雙重加密，外部金鑰存放區中的每個 KMS 金鑰都有兩個密碼編譯備份金鑰：

- KMS AWS KMS 金鑰的唯一金鑰材料。系統會產生此金鑰材料，且僅使用於 AWS KMS [經 FIPS 140-3 安全層級 3](#) 認證的硬體安全模組 (HSMs)。
- 外部金鑰管理器中的[外部金鑰](#)。

雙重加密具有以下效果：

- AWS KMS 無法在沒有透過外部金鑰存放區代理存取外部金鑰的情況下，解密由外部金鑰存放區中 KMS 金鑰加密的任何加密文字。
- 您無法解密 KMS 金鑰在 外部的外部金鑰存放區中加密的任何加密文字 AWS，即使您擁有其外部金鑰材料。

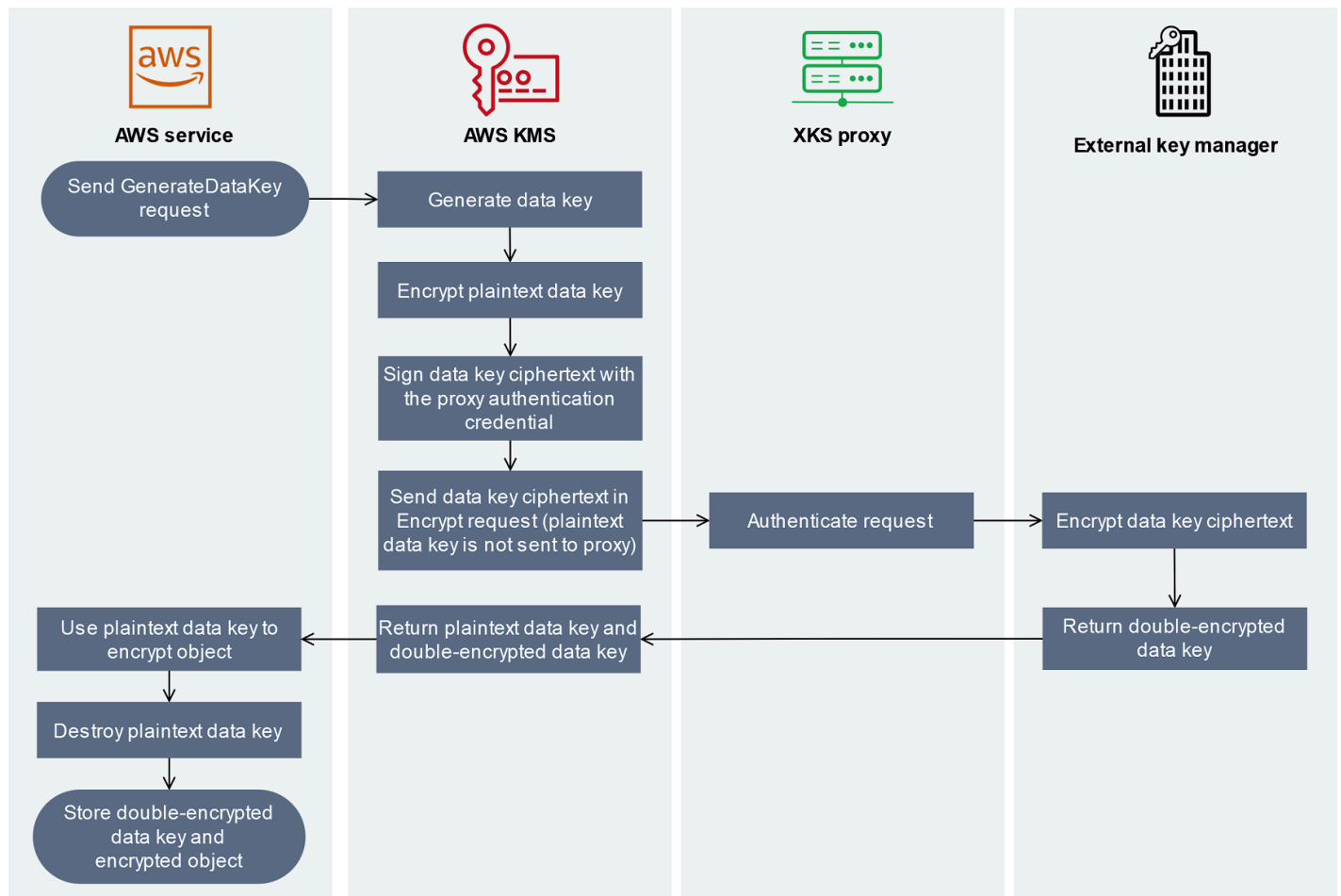
- 您無法重新建立從外部金鑰存放區刪除的 KMS 金鑰，即使您擁有其外部金鑰材料。每個 KMS 金鑰都擁有包含在對稱密文中的唯一中繼資料。新的 KMS 金鑰無法解密由原始金鑰加密的密文，即使其使用相同的外部金鑰材料。

如需做法中的雙重加密範例，請參閱 [外部金鑰存放區的運作方式](#)。

外部金鑰存放區的運作方式

您的[外部金鑰存放區](#)、[外部金鑰存放區代理](#)和[外部金鑰管理器](#)會共同運作，以保護您的 AWS 資源。以下程序描述了典型 AWS 服務的加密工作流程，該服務使用由 KMS 金鑰保護的唯一資料金鑰對每個物件進行加密。在此情況下，您已選擇外部金鑰存放區中的 KMS 金鑰來保護物件。此範例顯示 AWS KMS 如何使用[雙重加密](#)來保護傳輸中的資料金鑰，並確保外部金鑰存放區中 KMS 金鑰所產生的加密文字至少與使用金鑰材料的標準對稱 KMS 金鑰加密的加密文字一樣強 AWS KMS。

與 AWS 服務整合的每個實際所使用的加密方法 AWS KMS 會有所不同。如需詳細資訊，請參閱 AWS 服務文件「安全性」一章中的「資料保護」主題。



1. 您可以將新物件新增至 AWS 服務 資源。若要加密物件，AWS KMS 會使用外部金鑰存放區中的 KMS 金鑰，將 [GenerateDataKey](#) 請求 AWS 服務 傳送至。
2. AWS KMS 會產生 256 位元對稱資料金鑰，並準備透過外部金鑰存放區代理將純文字資料金鑰的副本傳送至外部金鑰管理器。透過使用與外部金鑰存放區中的 KMS 金鑰相關聯的 [AWS KMS 金鑰材料](#) 加密純文字資料金鑰，AWS KMS 開始 [雙重加密](#) 程序。
3. AWS KMS 會將 [加密](#) 請求傳送至與外部金鑰存放區相關聯的外部金鑰存放區代理。請求包含要加密的資料金鑰加密文字，以及與 KMS 金鑰相關聯的 [外部金鑰](#) ID。會使用外部金鑰存放區代理的 [代理身分驗證憑證](#) 來 AWS KMS 簽署請求。

資料金鑰的純文字複本不會傳送至外部金鑰存放區代理。

4. 外部金鑰存放區代理會驗證請求，然後將加密請求傳送至外部金鑰管理器。

某些外部金鑰存放區代理也會實作選用的 [授權政策](#)，只允許選取的主體在特定條件下執行操作。

5. 您的外部金鑰管理器會使用指定的外部金鑰加密資料金鑰密文。外部金鑰管理器會將雙重加密的資料金鑰傳回至您的外部金鑰存放區代理，它會將其傳回 AWS KMS。
6. AWS KMS 會將純文字資料金鑰和該資料金鑰的雙重加密複本傳回 AWS 服務。
7. AWS 服務 使用純文字資料金鑰來加密資源物件、銷毀純文字資料金鑰，以及將加密的資料金鑰與加密的物件一起存放。

有些 AWS 服務 可能會快取純文字資料金鑰，以用於多個物件，或在資源使用時重複使用。如需詳細資訊，請參閱 [無法使用的 KMS 金鑰如何影響資料金鑰](#)。

若要解密加密的物件，AWS 服務 必須在 [解密](#) 請求 AWS KMS 中將加密的資料金鑰傳回。若要解密加密的資料金鑰，AWS KMS 必須使用外部金鑰的 ID 將加密的資料金鑰傳回外部金鑰存放區代理。如果對外部金鑰存放區代理的解密請求因任何原因失敗，AWS KMS 則無法解密加密的資料金鑰，且 AWS 服務 無法解密加密的物件。

控制對外部金鑰存放區的存取

與標準 KMS 金鑰搭配使用的所有 AWS KMS 存取控制功能 — [金鑰政策](#)、[IAM 政策和授權](#) — 對外部金鑰存放區中的 KMS 金鑰的運作方式相同。您可以使用 IAM 政策來控制對建立和管理外部金鑰存放區之 API 操作的存取。您可以使用 IAM 政策和金鑰政策來控制對外部金鑰存放區 AWS KMS keys 中的存取。您也可以為 AWS 組織和 VPC 端點 [政策使用服務控制](#) 政策，以控制對外部金鑰存放區中 KMS 金鑰的存取。 <https://docs.aws.amazon.com/vpc/latest/privatelink/interface-endpoints.html#edit-vpc-endpoint-policy>

建議您僅向使用者和角色提供執行任務所需的許可。

主題

- [授權外部金鑰存放區管理器](#)
- [授權外部金鑰存放區中 KMS 金鑰的使用者](#)
- [授權 與您的外部金鑰存放區代理 AWS KMS 通訊](#)
- [外部金鑰存放區代理授權 \(選用\)](#)
- [mTLS 身分驗證 \(已棄用\)](#)

授權外部金鑰存放區管理器

建立及管理外部金鑰存放區的主體需要自訂金鑰存放區操作許可。以下清單說明了外部金鑰存放區管理器所需的最低許可。由於自訂金鑰存放區不是 AWS 資源，因此您無法為其他 中的主體提供外部金鑰存放區的許可 AWS 帳戶。

- kms:CreateCustomKeyStore
- kms:DescribeCustomKeyStores
- kms:ConnectCustomKeyStore
- kms:DisconnectCustomKeyStore
- kms:UpdateCustomKeyStore
- kms>DeleteCustomKeyStore

若要使用 [Amazon VPC 端點服務連線](#) 建立外部金鑰存放區，且 VPC 端點服務由不同的 擁有 AWS 帳戶，您還需要下列許可：

- ec2:DescribeVPCEndpointServices

建立外部金鑰存放區的主體需要許可，才能建立及設定外部金鑰存放區元件。主體只能在自己的帳戶中建立外部金鑰存放區。若要建立具有 [VPC 端點服務連接](#) 的外部金鑰存放區，主體必須具有建立下列元件的許可：

- Amazon VPC
- 公有和私有子網路
- 網路負載平衡器和目標群組
- Amazon VPC 端點服務

如需詳細資訊，請參閱 [Amazon VPC 的身分和存取管理](#)、[VPC 端點和 VPC 端點服務的身分和存取管理](#) 以及 [Elastic Load Balancing API 許可](#)。

授權外部金鑰存放區中 KMS 金鑰的使用者

在外部金鑰存放區 AWS KMS keys 中建立和管理的委託人需要與建立和管理任何 KMS 金鑰的委託人 [相同的許可](#) AWS KMS。外部金鑰存放區中 KMS 金鑰的 [預設金鑰政策](#) 與 AWS KMS 中 KMS 金鑰的預設金鑰政策完全相同。[屬性型存取控制](#) (ABAC) 使用標籤和別名來控制對 KMS 金鑰的存取，對外部金鑰存放區中的 KMS 金鑰也有效。

將自訂金鑰存放區中的 KMS 金鑰用於 [密碼編譯操作](#) 的委託人，需要許可對 KMS 金鑰執行密碼編譯操作，例如 [kms:Decrypt](#)。您可以在 IAM 或金鑰政策中提供這些許可。但是，他們使用自訂金鑰存放區中的 KMS 金鑰並不需要任何額外的許可。

若要設定僅適用於外部金鑰存放區中 KMS 金鑰的許可，請使用值為 EXTERNAL_KEY_STORE 的 [kms:KeyOrigin](#) 政策條件。您可以使用此條件來限制 [kms:CreateKey](#) 許可或 KMS 金鑰資源專屬的任何許可。例如，下列 IAM 政策允許其所連接的身分，以便對帳戶中的所有 KMS 金鑰呼叫指定的操作，前提是 KMS 金鑰位於外部金鑰存放區中。請注意，您可以將許可限制為外部金鑰存放區中的 KMS 金鑰，以及 中的 KMS 金鑰 AWS 帳戶，但不能限制為帳戶中的任何特定外部金鑰存放區。

```
{
  "Sid": "AllowKeysInExternalKeyStores",
  "Effect": "Allow",
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:DescribeKey"
  ],
  "Resource": "arn:aws:kms:us-west-2:111122223333:key/*",
  "Condition": {
    "StringEquals": {
      "kms:KeyOrigin": "EXTERNAL_KEY_STORE"
    }
  }
}
```


授權 與您的外部金鑰存放區代理 AWS KMS 通訊

AWS KMS 只會透過您提供的[外部金鑰存放區代理](#)與您的外部金鑰管理器通訊。AWS KMS 會使用 [Signature 第 4 版 \(SigV4\) 程序](#)與您指定的[外部金鑰存放區代理身分驗證憑證](#)簽署請求，以驗證您的代理。如果您為外部金鑰存放區代理使用[公有端點連線](#)，AWS KMS 不需要任何其他許可。

不過，如果您使用的是 [VPC 端點服務連線](#)，您必須授予 AWS KMS 許可，才能建立 Amazon VPC 端點服務的介面端點。無論外部金鑰存放區代理位於您的 VPC 中，還是外部金鑰存放區代理位於其他地方，但使用 VPC 端點服務與 通訊，都需要此許可 AWS KMS。

若要允許 AWS KMS 建立介面端點，請使用 [Amazon VPC 主控台](#)或 [ModifyVpcEndpointServicePermissions](#) 操作。允許下列主體的許可：`cks.kms.<region>.amazonaws.com`。

如果您的 Amazon VPC 端點服務是由 AWS 帳戶 與 AWS 帳戶 擁有外部金鑰存放區 (XKS) 不同的所擁有，您也需要允許 XKS 存取 VPC 端點服務。若要這麼做，[請允許將 XKS AWS 帳戶 ID 列為 Amazon VPC 端點服務的主體](#)。

Same AWS 帳戶

當您的 VPC 端點服務擁有 AWS 帳戶 與外部金鑰存放區相同的 時，您必須 AWS KMS 將 新增至 VPC 端點服務的允許委託人清單。

下列範例使用 [modify-vpc-endpoint-service-permissions](#) AWS CLI 命令，AWS KMS 允許 連線到美國西部（奧勒岡）(us-west-2) 區域中指定的 VPC 端點服務。使用此命令之前，請將 Amazon VPC 服務 ID 和 AWS 區域 取代為您的組態的有效值。

```
modify-vpc-endpoint-service-permissions
--service-id vpce-svc-12abc34567def0987
--add-allowed-principals '["cks.kms.us-west-2.amazonaws.com"]'
```

若要移除此許可，請使用 [Amazon VPC 主控台](#)或具有 `RemoveAllowedPrincipals` 參數的 [ModifyVpcEndpointServicePermissions](#)。

Cross AWS 帳戶

當另一個 VPC 端點服務擁有時 AWS 帳戶，您必須將 AWS KMS 和外部金鑰存放區新增至 VPC 端點服務的允許主體清單。

下列範例使用 [modify-vpc-endpoint-service-permissions](#) AWS CLI 命令來允許 AWS KMS 和外部金鑰存放區 (XKS) 連線到美國西部（奧勒岡）(us-west-2) 區域中指定的 VPC 端點服

務。使用此命令之前，請將 Amazon VPC 服務 ID、AWS 區域和 IAM 主體 ARN 取代為您的組態的有效值。IAM 主體應該替換為 XKS 擁有者中的主體 AWS 帳戶。

在此範例中，`arn:aws:iam::123456789012:role/cks_role` 是 XKS 擁有者帳戶中的 IAM 主體，用於建立、更新或連接 XKS 到您的 VPC 端點服務。如果您想要允許 XKS 擁有者帳戶中的所有主體存取您的 VPC 端點服務，您可以指定 `arn:aws:iam::123456789012:root`。

```
modify-vpc-endpoint-service-permissions
--service-id vpce-svc-12abc34567def0987
--add-allowed-principals '["cks.kms.us-west-2.amazonaws.com",
"arn:aws:iam::123456789012:role/cks_role"]'
```

若要移除此許可，請使用 [Amazon VPC 主控台](#) 或具有 `RemoveAllowedPrincipals` 參數的 [ModifyVpcEndpointServicePermissions](#)。

外部金鑰存放區代理授權 (選用)

某些外部金鑰存放區代理會實作使用其外部金鑰的授權需求。允許外部金鑰存放區代理 (但不是必需的) 來設計和實作授權方案，該方案允許特定使用者僅在特定條件下請求特定操作。例如，代理可能會設定為允許使用者 A 使用特定的外部金鑰進行加密，但無法使用其進行解密。

Proxy 授權獨立於所有外部金鑰存放區代理 AWS KMS 所需的 [SigV4-based 代理身分驗證](#)。其也獨立於金鑰政策、IAM 政策和授予，其可授權存取會影響外部金鑰存放區或其 KMS 金鑰的操作。

若要啟用外部金鑰存放區代理的授權，會在每個 [代理 API 請求](#) 中 AWS KMS 包含中繼資料，包括發起人、KMS 金鑰、AWS KMS 操作、AWS 服務 (如果有的話)。外部金鑰代理 API 第 1 版 (v1) 的請求中繼資料如下所示。

```
"requestMetadata": {
  "awsPrincipalArn": string,
  "awsSourceVpc": string, // optional
  "awsSourceVpce": string, // optional
  "kmsKeyArn": string,
  "kmsOperation": string,
  "kmsRequestId": string,
  "kmsViaService": string // optional
}
```

例如，您可以設定代理以允許來自特定委託人 (`awsPrincipalArn`) 的請求，但僅限於由特定 AWS 服務 () 代表委託人提出請求時 `kmsViaService`。

如果代理授權失敗，相關 AWS KMS 操作會失敗，並顯示說明錯誤的訊息。如需詳細資訊，請參閱 [代理授權問題](#)。

mTLS 身分驗證（已棄用）

本指南的先前版本提及交互 Transport Layer Security (mTLS) 做為驗證請求的選用次要身分驗證機制 AWS KMS。透過 mTLS，雙方 (AWS KMS 做為用戶端和 XKS 代理做為伺服器) 透過 TLS 頻道進行通訊會使用憑證彼此驗證。

不過，[Chrome 根計劃政策（第 4.2.2 節）的變更](#)禁止 Chrome 根存放區中包含的公開信任根 CAs 在 2026 年 6 月 15 日之後使用 clientAuth 延伸金鑰使用 (EKU) 延伸發行憑證。因此，AWS KMS 無法再從 [Amazon Trust Services](#) 取得適用於 mTLS 的用戶端憑證。2026 年 3 月 16 日 AWS KMS 之後，用於在中建立新的外部金鑰存放區的任何 XKS 代理都不需要 mTLS。2026 年 6 月 15 日之後，任何設定為需要 mTLS 的 XKS 代理將無法與通訊 AWS KMS。客戶必須依賴 SigV4 身分驗證來驗證請求是否來自 AWS KMS。如需詳細資訊，請參閱[外部金鑰存放區代理身分驗證憑證](#)。

選擇外部金鑰存放區代理連線選項

在建立外部金鑰存放區之前，請選擇連線選項，以決定如何與外部金鑰存放區元件 AWS KMS 通訊。您選擇的連接選項會決定規劃程序的剩餘部分。

如果您要建立外部金鑰存放區，您需要判斷如何與[外部金鑰存放區代理](#) AWS KMS 通訊。此選項將決定您需要哪些元件，以及如何設定這些元件。AWS KMS 支援下列連線選項。

- [公有端點連接](#)
- [VPC 端點服務連接](#)

選擇可滿足您的效能和安全目標的選項。

開始之前，[請確認您需要外部金鑰存放區](#)。大多數客戶都可以使用金鑰材料支援的 KMS AWS KMS 金鑰。

考量事項

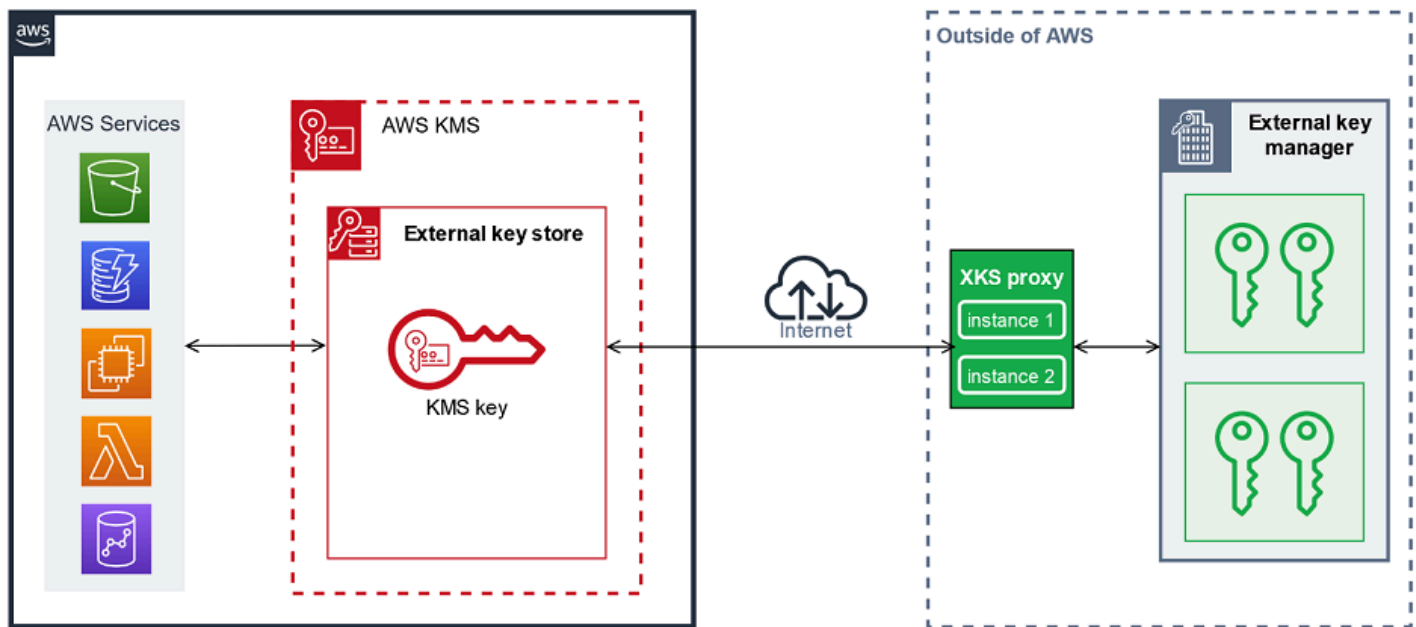
- 如果您的外部金鑰存放區代理內建於外部金鑰管理器中，則可能已預先決定您的連接。如需相關指導，請參閱外部金鑰管理器或外部金鑰存放區代理的文件。
- 即使在正操作的外部金鑰存放區中，您也可以[變更外部金鑰存放區代理連接選項](#)。但是，必須仔細規劃和執行此程序，以最大限度地減少中斷、避免錯誤並確保持續存取對您的資料進行加密的密碼編譯金鑰。

公有端點連接

AWS KMS 使用公有端點透過網際網路連線至外部金鑰存放區代理 (XKS 代理)。

此連接選項更易於設定和維護，並且可與某些金鑰管理模式完美配合。但是，其可能無法滿足某些組織的安全要求。

XKS proxy connected by a public endpoint



需求

如果您選擇公有端點連接，則需要下列項目。

- 您的外部金鑰存放區代理在公開可路由端點必須可存取。
- 您可對多個外部金鑰存放區使用相同的公有端點，前提是它們使用不同的[代理 URI 路徑值](#)。
- 您無法將相同的端點用於具有公有端點連線的外部金鑰存放區，以及任何具有 VPC 端點服務連線的外部金鑰存放區 AWS 區域，即使金鑰存放區位於不同的 AWS 帳戶。
- 您必須取得由外部金鑰存放區支援的公有憑證授權機構核發的 TLS 憑證。如需清單，請參閱[受信任的憑證授權機構](#)。

TLS 憑證上的主體通用名稱 (CN) 必須與外部金鑰存放區代理之[代理 URI 端點](#)中的網域名稱相符。例如，如果公有端點為 `https://myproxy.xks.example.com`，則 TLS 憑證上的通用名稱必須為 `myproxy.xks.example.com` 或 `*.xks.example.com`。

- 確保 AWS KMS 和外部金鑰存放區代理之間的任何防火牆允許進出 Proxy 上連接埠 443 的流量。透過 IPv4 在連接埠 443 上 AWS KMS 通訊。此值不可設定。

如需外部金鑰存放區的所有要求，請參閱[備妥先決條件](#)。

VPC 端點服務連接

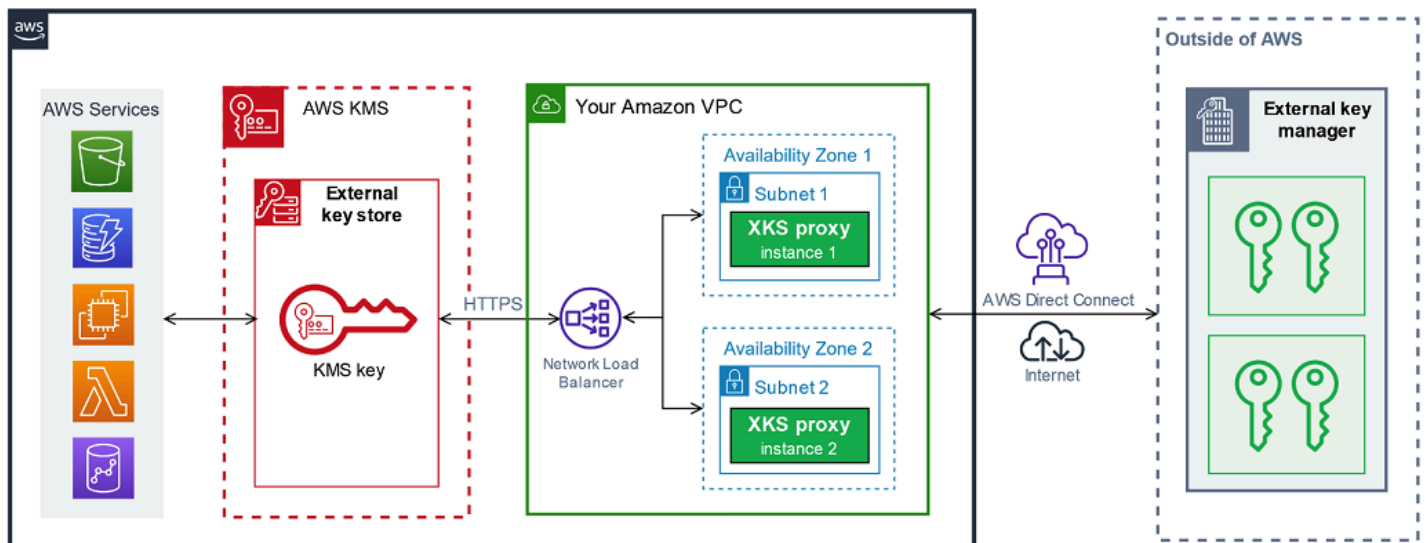
AWS KMS 透過建立介面端點到您建立和設定的 Amazon VPC 端點服務，連接到外部金鑰存放區代理 (XKS 代理)。您必須負責[建立 VPC 端點服務](#)，並將 VPC 連接至外部金鑰管理器。

您的端點服務可以使用任何[支援的網路到 Amazon VPC 選項](#)進行通訊，包括 [AWS Direct Connect](#)。

此連接選項的設定和維護更為複雜。但它使用 AWS PrivateLink，可讓 AWS KMS 私下連線至您的 Amazon VPC 和外部金鑰存放區代理，而無需使用公有網際網路。

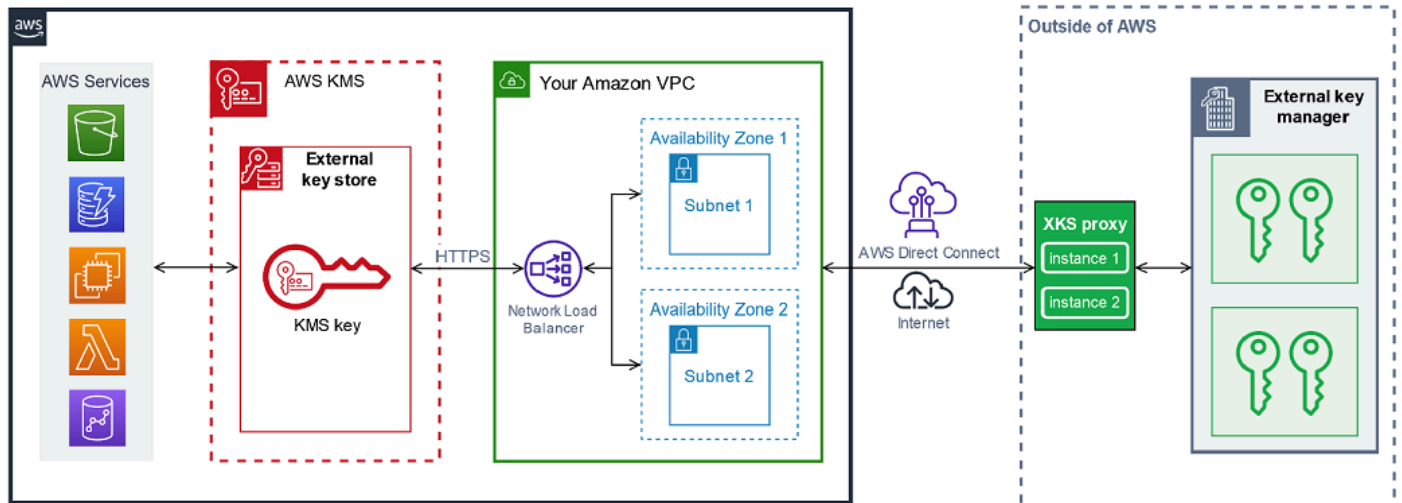
您可以在 Amazon VPC 中找到外部金鑰存放區代理。

XKS proxy hosted in Amazon VPC



或者，您可以在 外部找到外部金鑰存放區代理 AWS 雲端，並僅將 Amazon VPC 端點服務用於與 進行安全通訊 AWS KMS。

XKS proxy connected via Amazon VPC endpoint service



您也可以將外部金鑰存放區連接到另一個擁有的 Amazon VPC 端點服務 AWS 帳戶。兩者 AWS 帳戶都需要[必要的許可](#)，才能允許 AWS KMS 和 VPC 端點服務之間的通訊。

進一步了解：

- 檢閱建立外部金鑰存放區的程序，包括[備妥先決條件](#)。它將幫助您確保在建立外部金鑰存放區時具有所需的所有元件。
- 了解如何[控制對外部金鑰存放區的存取](#)，包括外部金鑰存放區管理員和使用者所需的許可。
- 了解為外部金鑰存放區 AWS KMS 記錄的 [Amazon CloudWatch 指標和維度](#)。強烈建議您建立警示來監控外部金鑰存放區，以便可偵測到效能和操作問題的早期跡象。

設定 VPC 端點服務連線

使用本節中的指引來建立和設定使用 [VPC 端點服務連線](#) 的外部金鑰存放區所需的 AWS 資源和相關元件。針對此連接選項列出的資源是[所有外部金鑰存放區所需資源](#)的補充。建立並設定所需資源後，您可以[建立外部金鑰存放區](#)。

您可以在 Amazon VPC 中找到外部金鑰存放區代理，或在外部找到代理 AWS，並使用 VPC 端點服務進行通訊。

開始之前，[請確認您需要外部金鑰存放區](#)。大多數客戶都可以使用金鑰材料支援的 KMS AWS KMS 金鑰。

 Note

VPC 端點服務連接所需的一些元素可能包含在外部金鑰管理器中。此外，您的軟體可能還有其他組態要求。在建立和設定本節中的 AWS 資源之前，請參閱您的代理和金鑰管理器文件。

主題

- [VPC 端點服務連接的要求](#)
- [步驟 1：建立 Amazon VPC 和子網路](#)
- [步驟 2：建立目標群組](#)
- [步驟 3：建立網路負載平衡器](#)
- [步驟 4：建立 VPC 端點服務](#)
- [步驟 5：驗證您的私有 DNS 名稱網域](#)
- [步驟 6：授權 AWS KMS 連線至 VPC 端點服務](#)

VPC 端點服務連接的要求

如果您為外部金鑰存放區選擇 VPC 端點服務連接，則需要下列資源。

- 連接到外部金鑰管理器的 Amazon VPC。其在兩個不同可用區域中必須擁有至少兩個私有[子網路](#)。

您可以將現有的 Amazon VPC 用於外部金鑰存放區，前提是其[符合與外部金鑰存放區搭配使用的要求](#)。多個外部金鑰存放區可以共用 Amazon VPC，但是每個外部金鑰存放區都必須擁有自己的 VPC 端點服務和私有 DNS 名稱。

- [由 AWS PrivateLink 提供支援的 Amazon VPC 端點服務](#)，具有[網路負載平衡器](#)和[目標群組](#)。

端點服務無法要求接受。此外，您必須新增 AWS KMS 作為允許的主體。這可讓 AWS KMS 建立介面端點，以便與您的外部金鑰存放區代理通訊。

- VPC 端點服務的私有 DNS 名稱在其 AWS 區域中是唯一的。

私有 DNS 名稱必須是較高層級公有網域的子網域。例如，如果私有 DNS 名為 myproxy-private.xks.example.com，則其必須是公有網域的子網域，例如 xks.example.com 或 example.com。

您必須[驗證私有 DNS 名稱的 DNS 網域的擁有權](#)。

- [支援的公有憑證授權機構](#)為您的外部金鑰存放區代理核發的 TLS 憑證。

TLS 憑證上的主體通用名稱 (CN) 必須與私有 DNS 名稱相符。例如，如果私有 DNS 名稱為 `myproxy-private.xks.example.com`，則 TLS 憑證上的 CN 必須為 `myproxy-private.xks.example.com` 或 `*.xks.example.com`。

- 若要將網路延遲降至最低，請在最接近[外部金鑰管理器](#)的[支援 AWS 區域](#)中建立 AWS 元件。如果可能，請選擇網路封包來回時間 (RTT) 為 35 毫秒或更短的區域。

如需外部金鑰存放區的所有要求，請參閱[備妥先決條件](#)。

步驟 1：建立 Amazon VPC 和子網路

VPC 端點服務連接需要一個 Amazon VPC，其使用至少兩個私有子網路連接到外部金鑰管理器。您可以建立 Amazon VPC，或使用符合外部金鑰存放區要求的現有 Amazon VPC。如需有關建立新 Amazon VPC 的說明，請參閱《Amazon 虛擬私有雲端使用者指南》中的[建立 VPC](#)。

Amazon VPC 的要求

Amazon VPC 端點服務必須具有下列屬性，才能使用外部金鑰存放區。

- 必須位於[支援的區域中](#)，做為外部金鑰存放區。
- 需要至少兩個私有子網路，每個處於不同的可用區域。
- Amazon VPC 的私有 IP 地址範圍不得與託管[外部金鑰管理器](#)之資料中心的私有 IP 地址範圍重疊。
- 所有元件都必須使用 IPv4。

您有許多選項可將 Amazon VPC 連接到外部金鑰存放區代理。選擇可滿足您的效能和安全需求的選項。如需清單，請參閱[將您的 VPC 連接到其他網路](#)和[網路到 Amazon VPC 連接選項](#)。如需詳細資訊，請參閱[Direct Connect](#)和[AWS Site-to-Site VPN 使用者指南](#)。

為您的外部金鑰存放區建立 Amazon VPC

遵循以下指示建立外部金鑰存放區的 Amazon VPC。只有在選擇[VPC 端點服務連接](#)選項時，才需要 Amazon VPC。您可以使用符合外部金鑰存放區要求的現有 Amazon VPC。

遵循[建立 VPC、子網路和其他 VPC 資源](#)主題中的指示，使用以下所需值。對於其他欄位，請接受預設值，並按要求提供名稱。

欄位	Value
IPv4 CIDR 區塊	輸入 VPC 的 IP 地址。Amazon VPC 的私有 IP 地址範圍不得與託管 外部金鑰管理器 之資料中心的私有 IP 地址範圍重疊。
可用區域 (AZ) 的數量	2 或以上
公有子網路數量	需要空值 (0)
私有子網路數量	每個 AZ 一個
NAT 閘道	需要空值。
VPC 端點	需要空值。
Enable DNS hostnames (啟用 DNS 主機名稱)	是
啟用 DNS 解析	是

請務必測試 VPC 通訊。例如，如果您的外部金鑰存放區代理不在 Amazon VPC 中，則請在 Amazon VPC 中建立 Amazon EC2 執行個體，確認 Amazon VPC 可以與您的外部金鑰存放區代理通訊。

將 VPC 連接到外部金鑰管理器

使用 Amazon VPC 支援的任何[網路連接選項](#)，將 VPC 連接到託管外部金鑰管理器的資料中心。確保 VPC 中的 Amazon EC2 執行個體 (或外部金鑰存放區代理 (如果位於 VPC 中)) 可以與資料中心和外部金鑰管理器通訊。

步驟 2：建立目標群組

建立必要的 VPC 端點服務之前，請先建立其必要元件、網路負載平衡器 (NLB) 和目標群組。網路負載平衡器 (NLB) 會在多個狀態良好的目標之間分佈請求，其中任何一個都可以為請求提供服務。在此步驟中，您會為外部金鑰存放區代理建立至少具有兩個主機的目標群組，並向目標群組註冊 IP 地址。

遵循[設定目標群組](#)主題中的指示，使用以下所需值。對於其他欄位，請接受預設值，並按要求提供名稱。

欄位	Value
Target type (目標類型)	IP 位址
通訊協定	TCP
站點	443
IP 地址類型	IPv4
VPC	選擇您要在其中為外部金鑰存放區建立 VPC 端點服務的 VPC。
運作狀態檢查通訊協定和路徑	您的運作狀態檢查通訊協定和路徑會因外部金鑰存放區代理組態而有所不同。請參閱外部金鑰管理器或外部金鑰存放區代理的文件。 如需有關針對目標群組設定運作狀態檢查的一般資訊，請參閱《Network Load Balancer 之 Elastic Load Balancing 使用者指南》中的 目標群組運作狀態檢查 。
網路	其他私有 IP 地址
IPv4 地址	外部金鑰存放區代理的私有地址
連接埠	443

步驟 3：建立網路負載平衡器

網路負載平衡器會將網路流量 (包括從 AWS KMS 到外部金鑰存放區代理的請求) 分佈至設定的目標。

遵循[設定負載平衡器和接聽程式](#)主題中的指示，設定和新增接聽程式，並使用下列必要值建立負載平衡器。對於其他欄位，請接受預設值，並按要求提供名稱。

欄位	Value
Scheme	內部 (Internal)
IP 地址類型	IPv4
網路映射	選擇您要在其中為外部金鑰存放區建立 VPC 端點服務的 VPC。

欄位	Value
映射	選擇您為 VPC 子網路設定的兩個可用區域 (至少兩個)。驗證子網路名稱和私有 IP 地址。
通訊協定	TCP
站點	443
預設動作：轉送至	選擇網路負載平衡器的 目標群組 。

步驟 4：建立 VPC 端點服務

通常，您會建立服務的端點。不過，當您建立 VPC 端點服務時，您是提供者，並為您的服務 AWS KMS 建立端點。對於外部金鑰存放區，請使用您在上一個步驟中建立的網路負載平衡器建立 VPC 端點服務。VPC 端點服務可以 AWS 帳戶 與您的外部金鑰存放區或不同的 位於相同的 中 AWS 帳戶。

多個外部金鑰存放區可以共用 Amazon VPC，但是每個外部金鑰存放區都必須擁有自己的 VPC 端點服務和私有 DNS 名稱。

遵循[建立端點服務](#)主題中的指示，使用下列必要值建立 VPC 端點服務。對於其他欄位，請接受預設值，並按要求提供名稱。

欄位	Value
負載平衡器類型	網路
可用的負載平衡器	選擇您在上一步驟中建立的 網路負載平衡器 。 如果您的新負載平衡器未出現在清單中，請確認它處於作用中狀態。負載平衡器狀態可能需要幾分鐘才能從佈建變更為作用中。
需要接受	False。取消勾選核取方塊。 不需要接受。若未手動接受，AWS KMS 則無法連線至 VPC 端點服務。如果需要接受，嘗試 建立外部金鑰存放區 會失敗，並出現 XksProxyInvalidConfigurationException 例外狀況。

欄位	Value
啟用私有 DNS 名稱	將私有 DNS 名稱與服務建立關聯
私有 DNS 名稱	輸入在其 AWS 區域中唯一的私有 DNS 名稱。 私有 DNS 名稱必須是較高層級公有網域的子網域。例如，如果私有 DNS 名稱為 <code>myproxy-private.xks.example.com</code>，則其必須是公有網域的子網域，例如 <code>xks.example.com</code> 或 <code>example.com</code>。 此私有 DNS 名稱必須與外部金鑰存放區代理上設定的 TLS 憑證中的主體通用名稱 (CN) 相符。例如，如果私有 DNS 名為 <code>myproxy-private.xks.example.com</code>，則 TLS 憑證上的 CN 必須為 <code>myproxy-private.xks.example.com</code> 或 <code>*.xks.example.com</code>。 如果憑證和私有 DNS 名稱不相符，則嘗試將外部金鑰存放區連接至其外部金鑰存放區代理會失敗，並顯示 <code>XKS_PROXY_INVALID_TLS_CONFIGURATION</code> 的连接錯誤代碼。如需詳細資訊，請參閱一般組態錯誤。
支援的 IP 地址類型	IPv4

步驟 5：驗證您的私有 DNS 名稱網域

建立 VPC 端點服務時，其網域驗證狀態為 `pendingVerification`。在使用 VPC 端點服務建立外部金鑰存放區之前，此狀態必須為 `verified`。若要確認您擁有與私有 DNS 名稱相關聯的網域，您必須在公有 DNS 伺服器中建立 TXT 記錄。

例如，如果 VPC 端點服務的私有 DNS 名為 `myproxy-private.xks.example.com`，您必須在公有網域中建立 TXT 記錄，例如 `xks.example.com` 或 `example.com`，以 `public. AWS PrivateLink looks` 為準，先在上，`xks.example.com` 再於上建立 TXT 記錄 `example.com`。

Tip

新增 TXT 記錄之後，Domain verification status (網域驗證狀態) 值可能需要幾分鐘才能從 `pendingVerification` 變更為 `verify`。

若要開始，請使用下列其中一種方法查找網域的驗證狀態。有效值為 `verified`、`pendingVerification` 和 `failed`。

- 在 [Amazon VPC 主控台](#) 中，選擇 Endpoint services (端點服務)，然後選擇您的端點服務。在詳細資訊窗格中，請查看 Domain verification status (網域驗證狀態)。
- 使用 [DescribeVpcEndpointServiceConfigurations](#) 操作。State 值位於 `ServiceConfigurations.PrivateDnsNameConfiguration.State` 欄位中。

如果驗證狀態不是 `verified`，請遵循[網域擁有權驗證](#)主題中的指示，將 TXT 記錄新增至網域的 DNS 伺服器，並確認 TXT 記錄已發佈。然後再次檢查您的驗證狀態。

您不需要為私有 DNS 網域名稱建立 A 記錄。當為您的 VPC 端點服務 AWS KMS 建立界面端點時，AWS PrivateLink 會自動建立託管區域，其中包含 AWS KMS VPC 中私有網域名稱所需的 A 記錄。對於具有 VPC 端點服務連接的外部金鑰存放區，當您[將外部金鑰存放區連接至](#)其外部金鑰存放區代理時，會發生這種情況。

步驟 6：授權 AWS KMS 連線至 VPC 端點服務

請參閱下列管理 Amazon VPC 端點服務許可的程序。每個步驟取決於外部金鑰存放區、VPC 端點服務和之間的連線和組態 AWS 帳戶。

Same AWS 帳戶

當您的 VPC 端點服務擁有 AWS 帳戶與外部金鑰存放區相同的時，您必須將 AWS KMS 新增至 VPC 端點服務的允許委託人清單。這可讓 AWS KMS 建立 VPC 端點服務的界面端點。如果 AWS KMS 不是允許的委託人，則嘗試建立外部金鑰存放區將會失敗，但有 `XksProxyVpcEndpointServiceNotFoundException` 例外。

遵循《AWS PrivateLink 指南》中的[管理許可](#)主題。使用以下所需值。

欄位	Value
AWS KMS ARN	<code>cks.kms.<region>.amazonaws.com</code> 例如 <code>cks.kms.us-east-1.amazonaws.com</code>

Cross AWS 帳戶

當您的 VPC 端點服務由其他人擁有時 AWS 帳戶，您必須同時將 AWS KMS 和您的帳戶新增至允許委託人清單。這可讓 AWS KMS 和您的外部金鑰存放區建立 VPC 端點服務的介面端點。如果 AWS KMS 不是允許的主體，嘗試建立外部金鑰存放區將會失敗，並出現 `XksProxyVpcEndpointServiceNotFoundException` 例外狀況。您需要提供外部金鑰存放區所在的 AWS 帳戶 ARN。

遵循《AWS PrivateLink 指南》中的[管理許可](#)主題。使用以下所需值。

欄位	Value
AWS KMS ARN	<code>cks.kms.<region>.amazonaws.com</code> 例如 <code>cks.kms.us-east-1.amazonaws.com</code>
AWS 帳戶 ARN	<code>arn:aws:iam:: 111122223333 :role/role_name</code> 例如 <code>arn:aws:iam:: 123456789012 :role/cks_role</code>

下一頁: [建立外部金鑰存放區](#)

建立外部金鑰存放區

您可以在每個 AWS 帳戶 和區域中建立一或多個外部金鑰存放區。每個外部金鑰存放區都必須與外部金鑰管理器相關聯 AWS，以及一個外部金鑰存放區代理 (XKS 代理)，其會媒介 AWS KMS 和外部金鑰管理器之間的通訊。如需詳細資訊，請參閱[選擇外部金鑰存放區代理連線選項](#)。開始之前，[請確認您需要外部金鑰存放區](#)。大多數客戶都可以使用金鑰材料支援的 KMS AWS KMS 金鑰。

Tip

有些外部金鑰管理器會提供更簡單的方法來建立外部金鑰存放區。如需詳細資訊，請參閱外部金鑰管理器文件。

在您建立外部金鑰存放區之前，您需要[備妥先決條件](#)。在建立過程中，您可以指定外部金鑰存放區的屬性。最重要的是，您可以指出 中的外部金鑰存放區是否 AWS KMS 使用[公有端點](#)或 [VPC 端點服務](#)連線到其外部金鑰存放區代理。您也可以指定連線詳細資訊，包括代理的 URI 端點，以及 將 API 請求 AWS KMS 傳送至代理的代理端點內路徑。

考量事項

- KMS 無法透過 IPv6 與外部金鑰存放區通訊。
- 如果您使用公有端點連線，請確定 AWS KMS 可以使用 HTTPS 連線透過網際網路與您的代理通訊。這包括在外部金鑰存放區代理上設定 TLS，並確保 AWS KMS 和代理之間的任何防火牆允許往返代理上連接埠 443 的 IPv4 流量。建立具有公有端點連線的外部金鑰存放區時，會透過將狀態請求傳送至外部金鑰存放區代理來 AWS KMS 測試連線。此測試會驗證端點是否可連接，以及您的外部金鑰存放區代理是否將接受使用[外部金鑰存放區代理身分驗證憑證](#)簽署的請求。如果此測試請求失敗，則建立外部金鑰存放區的操作失敗。
- 如果您使用 VPC 端點服務連接，則請確定網路負載平衡器、私有 DNS 名稱和 VPC 端點服務設定正確且可運作。如果外部金鑰存放區代理不在 VPC 中，您需要確保 VPC 端點服務可以與外部金鑰存放區代理通訊。（當您[將外部金鑰存放區](#)連線至其外部金鑰存放區代理時 AWS KMS，測試 VPC 端點服務連線。）
- AWS KMS 會記錄 [Amazon CloudWatch 指標和維度](#)，尤其是外部金鑰存放區。根據其中一些指標監控圖形會顯示在每個外部金鑰存放區的 AWS KMS 主控台中。我們強烈建議您使用這些指標來建立監控外部金鑰存放區的警示。這些警示會在效能和操作問題的相關跡象發生之前提醒您。如需說明，請參閱[監控外部金鑰存放區](#)。
- 外部金鑰存放區受[資源配額](#)限制。在外部金鑰存放區中使用 KMS 金鑰會受到[請求配額](#)的限制。在設計外部金鑰存放區實作之前，請檢閱這些配額。

Note

檢閱您的組態，了解可能導致其無法運作的循環相依性。

例如，如果您使用 AWS 資源建立外部金鑰存放區代理，請確定操作代理不需要在透過該代理存取的外部金鑰存放區中使用 KMS 金鑰。

在中斷連接的狀態下建立所有新的外部金鑰存放區。在外部金鑰存放區中建立 KMS 金鑰之前，必須先[將其連接](#)至外部金鑰存放區代理。若要變更外部金鑰存放區的屬性，請[編輯外部金鑰存放區設定](#)。

主題

- [備妥先決條件](#)
- [建立新的外部金鑰存放區](#)

備妥先決條件

在建立外部金鑰存放區之前，您需要組合必要的元件，包括您將用來支援外部金鑰存放區的[外部金鑰管理器](#)，以及將 AWS KMS 請求轉譯為外部金鑰管理器可理解格式的[外部金鑰存放區代理](#)。

所有外部金鑰存放區都需要下列元件。除了這些元件之外，您還需要提供其他元件來支援您選擇的[外部金鑰存放區代理連接選項](#)。

Tip

您的外部金鑰管理器可能包含其中一些元件，也可能會為您設定這些元件。如需詳細資訊，請參閱外部金鑰管理器文件。

如果您要在 AWS KMS 主控台中建立外部金鑰存放區，您可以選擇上傳以 JSON 為基礎的[代理組態檔案](#)，以指定[代理 URI 路徑](#)和[代理身分驗證憑證](#)。某些外部金鑰存放區代理會為您產生此檔案。如需詳細資訊，請參閱外部金鑰存放區代理或外部金鑰管理器的文件。

外部金鑰管理器

每個外部金鑰存放區至少需要一個[外部金鑰管理器](#)執行個體。這可以是實體或虛擬硬體安全模組 (HSM)，也可以是金鑰管理軟體。

您可以使用單一金鑰管理器，但建議有至少兩個相關的金鑰管理器執行個體共用密碼編譯金鑰以進行備援。外部金鑰存放區不需要獨佔使用外部金鑰管理器。不過，外部金鑰管理器必須具有處理來自使用外部金鑰存放區中 KMS 金鑰之 AWS 服務的預期加密和解密請求頻率的容量。您的外部金鑰管理器應設定為每秒最多可處理 1800 個請求，並在每個請求的 250 毫秒逾時內回應。我們建議您將外部金鑰管理器靠近，AWS 區域 以便網路往返時間 (RTT) 為 35 毫秒或更短。

如果您的外部金鑰存放區代理允許，則您可以變更與外部金鑰存放區代理關聯的外部金鑰管理器，但新的外部金鑰管理器必須是具有相同金鑰材料的備份或快照。如果您與 KMS 金鑰相關聯的外部金鑰不再可供外部金鑰存放區代理使用，AWS KMS 則無法解密使用 KMS 金鑰加密的加密文字。

外部金鑰管理器必須可供外部金鑰存放區代理存取。如果來自代理的 [GetHealthStatus](#) 回應報告所有外部金鑰管理器執行個體都為 Unavailable，則建立外部金鑰存放區的所有嘗試都會失敗，並顯示 [XksProxyUriUnreachableException](#)。

外部金鑰存放區代理

您必須指定[外部金鑰存放區代理](#) (XKS 代理)，其符合 [AWS KMS 外部金鑰存放區代理規格](#)中的設計需求。您可以開發或購買外部金鑰存放區代理，或使用外部金鑰存放區代理提供或內建於外部金鑰管理

器中。AWS KMS 建議將外部金鑰存放區代理設定為每秒處理最多 1800 個請求，並在每個請求的 250 毫秒逾時內回應。我們建議您將外部金鑰管理器靠近 AWS 區域，以便網路往返時間 (RTT) 為 35 毫秒或更短。

您可以將外部金鑰存放區代理用於多個外部金鑰存放區，但是每個外部金鑰存放區在其請求的外部金鑰存放區代理內必須具有唯一的 URI 端點和路徑。

如果您使用的是 VPC 端點服務連接，則可以在 Amazon VPC 中找到外部金鑰存放區代理，但這不是必需的。您可以在外部找到代理 AWS，例如在私有資料中心，並僅使用 VPC 端點服務與代理通訊。

代理身分驗證憑證

若要建立外部金鑰存放區，您必須指定外部金鑰存放區代理身分驗證憑證 (XksProxyAuthenticationCredential)。

您必須在外部金鑰存放區 proxy AWS KMS 上建立的 [身分驗證憑證](#) (XksProxyAuthenticationCredential)。透過使用 [Signature 第 4 版 \(SigV4\) 程序](#) 搭配外部金鑰存放區 AWS KMS 代理身分驗證憑證來簽署代理的請求，以對您的代理進行身分驗證。您可以在建立外部金鑰存放區時指定身分驗證憑證，而且您可以隨時 [將其變更](#)。如果代理輪換憑證，則請務必更新外部金鑰存放區的憑證值。

代理身分驗證憑證有兩部分。您必須為外部金鑰存放區提供這兩部分。

- 存取金鑰 ID：識別私密存取金鑰。您可以提供純文字形式的 ID。
- 私密存取金鑰：Credential 的私密部分。在儲存私密存取金鑰之前，會在登入資料中 AWS KMS 加密私密存取金鑰。

AWS KMS 用來簽署外部金鑰存放區代理請求的 SigV4 登入資料與您 AWS 帳戶中任何主體相關聯的任何 AWS Identity and Access Management SigV4 登入資料無關。請勿針對外部金鑰存放區代理重複使用任何 IAM SigV4 憑證。

代理連接

若要建立外部金鑰存放區，您必須指定外部金鑰存放區代理連接選項 (XksProxyConnectivity)。

AWS KMS 可以使用 [公有端點](#) 或 [Amazon Virtual Private Cloud \(Amazon VPC\) 端點服務](#) 與您的外部金鑰存放區代理通訊。雖然公有端點的設定和維護較為簡單，但可能不符合每個安裝的安全需求。如果選擇 Amazon VPC 端點服務連接選項，則必須建立和維護所需元件，包括在兩個不同可用區域中至少具有兩個子網路的 Amazon VPC、具有網路負載平衡器和目標群組的 VPC 端點服務，以及 VPC 端點服務的私有 DNS 名稱。

您可以變更外部金鑰存放區的[代理連接選項](#)。但是，您必須確保與外部金鑰存放區中 KMS 金鑰相關聯的金鑰材料持續可用。否則，AWS KMS 無法解密使用這些 KMS 金鑰加密的任何加密文字。

如需決定哪個代理連接選項最適合您的外部金鑰存放區的說明，請參閱[選擇外部金鑰存放區代理連線選項](#)。如需建立和設定 VPC 端點服務連接的說明，請參閱[設定 VPC 端點服務連線](#)。

代理 URI 端點

若要建立外部金鑰存放區，您必須指定 AWS KMS 用來將請求傳送至外部金鑰存放區代理的端點 (XksProxyUriEndpoint)。

通訊協定必須是連接埠 443 上透過 IPv4 的 HTTPS。AWS KMS communicates。請勿在代理 URI 端點值中指定連接埠。

- [公有端點連接](#) – 指定外部金鑰存放區代理的公開可用端點。在建立外部金鑰存放區之前，必須可連接此端點。
- [VPC 端點服務連接](#) – 指定 https://，後跟 VPC 端點服務的私有 DNS 名稱。

在外部金鑰存放區代理上設定的 TLS 伺服器憑證必須與外部金鑰存放區代理 URI 端點中的網域名稱相符，並由外部金鑰存放區支援的憑證授權機構發行。如需清單，請參閱[受信任的憑證授權機構](#)。在發行 TLS 憑證之前，您的憑證授權機構將要求提供網域擁有權證明。

TLS 憑證上的主體通用名稱 (CN) 必須與私有 DNS 名稱相符。例如，如果私有 DNS 名稱為 myproxy-private.xks.example.com，則 TLS 憑證上的 CN 必須為 myproxy-private.xks.example.com 或 *.xks.example.com。

您可以[變更代理 URI 端點](#)，但請確定外部金鑰存放區代理可存取與外部金鑰存放區中 KMS 金鑰相關聯的金鑰材料。否則，AWS KMS 無法解密使用這些 KMS 金鑰加密的任何加密文字。

唯一性要求

- 組合的代理 URI 端點 (XksProxyUriEndpoint) 和代理 URI 路徑 (XksProxyUriPath) 值在 AWS 帳戶 和 區域中必須唯一。
- 具有公有端點連接的外部金鑰存放區可以共用相同的代理 URI 端點，前提是其具有不同的代理 URI 路徑值。
- 具有公有端點連線能力的外部金鑰存放區無法使用與具有相同 VPC 端點服務連線能力的任何外部金鑰存放區相同的代理 URI 端點值 AWS 區域，即使金鑰存放區不同 AWS 帳戶。
- 具有 VPC 端點連接的每個外部金鑰存放區都必須有自己的私有 DNS 名稱。代理 URI 端點（私有 DNS 名稱）在 AWS 帳戶 和 區域中必須是唯一的。

代理 URI 路徑

若要建立外部金鑰存放區，您必須在外部金鑰存放區代理中指定[所需代理 API](#) 的基本路徑。該值必須以 / 開頭，且必須以 /kms/xks/v1 結尾，其中 v1 表示外部金鑰存放區代理的 AWS KMS API 版本。此路徑可以在必要元素之間包含選用字首，例如 /example-prefix/kms/xks/v1。若要尋找此值，請參閱外部金鑰存放區代理的文件。

AWS KMS 會將代理請求傳送到代理 URI 端點和代理 URI 路徑串連指定的地址。例如，如果代理 URI 端點為 `https://myproxy.xks.example.com` 且代理 URI 路徑為 `/kms/xks/v1`，會將其代理 API 請求 AWS KMS 傳送至 `https://myproxy.xks.example.com/kms/xks/v1`。

您可以[變更代理 URI 路徑](#)，但請確定外部金鑰存放區代理可存取與外部金鑰存放區中 KMS 金鑰相關聯的金鑰材料。否則，AWS KMS 無法解密使用這些 KMS 金鑰加密的任何加密文字。

唯一性要求

- 組合的代理 URI 端點 (XksProxyUriEndpoint) 和代理 URI 路徑 (XksProxyUriPath) 值在 AWS 帳戶 和區域中必須唯一。

VPC 端點服務

指定用於與外部金鑰存放區代理通訊的 Amazon VPC 端點服務名稱。只有使用 VPC 端點服務連接的外部金鑰存放區才需要此元件。如需為外部金鑰存放區設定和配置 VPC 端點服務的說明，請參閱[設定 VPC 端點服務連線](#)。

VPC 端點服務必須具有下列屬性：

- VPC 端點服務可以位於與 AWS 帳戶 外部金鑰存放區相同的或不同的 中。
 - VPC 端點服務必須位於與外部金鑰存放區 AWS 區域 相同的 中。
 - 如果 VPC 端點服務位於不同的 中，您將需要提供 VPC 端點服務的 AWS 帳戶 ID AWS 帳戶。
- 其必須擁有已連接到至少兩個子網路的網路負載平衡器 (NLB)，每個位於不同的可用區域中。
- VPC 端點服務的允許主體清單必須包含區域的 AWS KMS 服務主體：`cks.kms.<region>.amazonaws.com`，例如 `cks.kms.us-east-1.amazonaws.com`。
 - 如果您的 Amazon VPC 端點服務是由 AWS 帳戶 與 AWS 帳戶 擁有外部金鑰存放區 (XKS) 不同的所擁有，您也需要允許 XKS 存取 VPC 端點服務。若要這麼做，[請允許將 XKS AWS 帳戶 ID 列為 Amazon VPC 端點服務的主體](#)。
- 其不能要求接受連接請求。

- 其在較高級別的公有網域中必須具有私有 DNS 名稱。例如，在公有 `xks.example.com` 網域中擁有私有 DNS 名稱 `myproxy-private.xks.example.com`。

具有 VPC 端點服務連接之外部金鑰存放區的私有 DNS 名稱在其 AWS 區域中必須唯一。

- 私有 DNS 名稱網域的[網域驗證狀態](#)必須為 `verified`。
- 在外部金鑰存放區代理上設定的 TLS 伺服器憑證必須指定可連接端點的私有 DNS 主機名稱。

唯一性要求

- 具有 VPC 端點連接的外部金鑰存放區可以共用 Amazon VPC，但是每個外部金鑰存放區都必須有自己的 VPC 端點服務和私有 DNS 名稱。

代理組態檔案

代理組態檔案是一個可選的基於 JSON 的檔案，其中包含外部金鑰存放區的[代理 URI 路徑](#)和[代理身分驗證憑證](#)屬性的值。在 AWS KMS 主控台中建立或[編輯外部金鑰存放區](#)時，您可以上傳代理組態檔案，以提供外部金鑰存放區的組態值。使用此檔案可避免輸入和貼上錯誤，並確保外部金鑰存放區中的值與外部金鑰存放區代理中的值相符。

代理組態檔案由外部金鑰存放區代理產生。若要了解您的外部金鑰存放區代理是否提供代理組態檔案，請參閱您的外部金鑰存放區代理文件。

以下是具有虛構值的正確代理組態檔案範例。

```
{
  "XksProxyUriPath": "/example-prefix/kms/xks/v1",
  "XksProxyAuthenticationCredential": {
    "AccessKeyId": "ABCDE12345670EXAMPLE",
    "RawSecretAccessKey": "0000EXAMPLEFA5FT0mCc3DrGUe2sti527BitkQ0Zr9M09+vE="
  }
}
```

您只能在 AWS KMS 主控台中建立或編輯外部金鑰存放區時上傳代理組態檔案。您無法將其與 [CreateCustomKeyStore](#) 或 [UpdateCustomKeyStore](#) 操作搭配使用，但您可以使用代理組態檔案中的值來確保您的參數值正確無誤。

建立新的外部金鑰存放區

組合必要的先決條件後，您可以在 AWS KMS 主控台中或使用 [CreateCustomKeyStore](#) 操作建立新的外部金鑰存放區。

使用 AWS KMS 主控台

在建立外部金鑰存放區之前，[請選擇您的代理連線類型](#)，並確保您已建立並設定所有必要的[元件](#)。如果您需要尋找任何必要值的說明，請參閱外部金鑰存放區代理或金鑰管理軟體的文件。

Note

在 中建立外部金鑰存放區時 AWS 管理主控台，您可以上傳 JSON 型代理組態檔案，其中包含[代理 URI 路徑](#)和[代理身分驗證憑證](#)的值。某些代理會為您產生此檔案。這不是必要的。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，依次選擇 Custom key stores (自訂金鑰存放區)、External key stores (外部金鑰存放區)。
4. 選擇 Create external key store (建立外部金鑰存放區)。
5. 輸入外部金鑰存放區的易記名稱。該名稱在帳戶的所有外部金鑰存放區中必須唯一。

Important

請勿在此欄位包含機密或敏感資訊。此欄位可能以純文字顯示於 CloudTrail 日誌與其他輸出。

6. 選擇您的[代理連接](#)類型。

您的代理連接選項會決定外部金鑰存放區代理[所需的元件](#)。如需進行此選擇的說明，請參閱 [選擇外部金鑰存放區代理連線選項](#)。

- a. 如果您的 VPC 端點服務位於不同的 中，請選取跨帳戶 VPC 端點服務 AWS 帳戶。然後在 VPC 端點服務擁有者帳戶 AWS 帳戶 ID 欄位中輸入 VPC 端點擁有者的 ID。
- b. 選擇或輸入此外部金鑰存放區的 [VPC 端點服務](#) 名稱。只有在外部金鑰存放區代理連線類型為 VPC 端點服務時，才會顯示此步驟。

VPC 端點服務及其 VPC 必須滿足外部金鑰存放區的要求。如需詳細資訊，請參閱[the section called “備妥先決條件”](#)。

7. 選擇或輸入此外部金鑰存放區的 [VPC 端點服務](#) 名稱。只有當外部金鑰存放區代理連接類型為 VPC endpoint service (VPC 端點服務) 時，此步驟才會出現。

VPC 端點服務及其 VPC 必須滿足外部金鑰存放區的要求。如需詳細資訊，請參閱[the section called “備妥先決條件”](#)。

8. 輸入您的[代理 URI 端點](#)。通訊協定必須是連接埠 443 上透過 IPv4 的 HTTPS。AWS KMS communicates。請勿在代理 URI 端點值中指定連接埠。

如果 AWS KMS 辨識您在上一個步驟中指定的 VPC 端點服務，它會為您完成此欄位。

若為公有端點連接，請輸入公開可用的端點 URI。若為 VPC 端點連接，請輸入 https://，後跟 VPC 端點服務的私有 DNS 名稱。

9. 若要輸入[代理 URI 路徑](#)字首和[代理身分驗證憑證](#)的值，請上傳代理組態檔案，或手動輸入值。
 - 如果您有可選的[代理組態檔案](#)，其中包含[代理 URI 路徑](#)和[代理身分驗證憑證](#)的值，則請選擇 Upload configuration file (上傳組態檔案)。遵循步驟上傳檔案。

上傳檔案時，主控台會在可編輯欄位中顯示檔案中的值。您可以立即變更值，或在建立外部金鑰存放區後[編輯這些值](#)。

若要顯示私密存取金鑰的值，請選擇 Show secret access key (顯示私密存取金鑰)。

- 如果您沒有代理組態檔案，則可以手動輸入代理 URI 路徑和代理身分驗證憑證。
 - a. 如果您沒有代理組態檔案，則可以手動輸入代理 URI。主控台會提供必要的 /kms/xks/v1 值。

如果您的[代理 URI 路徑](#)包含可選字首，例如 */example-prefix/kms/xks/v1* 中的 example-prefix，則請在 Proxy URI path prefix (代理 URI 路徑字首) 欄位中輸入該字首。否則，請將欄位保留空白。

- b. 如果您沒有代理組態檔案，則可以手動輸入[代理身分驗證憑證](#)。存取金鑰 ID 和私密存取金鑰都是必要項目。
 - 在 Proxy credential: Access key ID (代理憑證：存取金鑰 ID) 中，輸入代理身分驗證憑證的存取金鑰 ID。存取金鑰 ID 可識別私密存取金鑰。
 - 在 Proxy credential: Secret access key (代理憑證：私密存取金鑰) 中，輸入代理身分驗證憑證的私密存取金鑰。

若要顯示私密存取金鑰的值，請選擇 Show secret access key (顯示私密存取金鑰)。

此程序不會設定或變更您在外部金鑰存放區代理上建立的身分驗證憑證。其只是將這些值與您的外部金鑰存放區相關聯。如需有關設定、變更及輪換代理身分驗證憑證的資訊，請參閱外部金鑰存放區代理或金鑰管理軟體的文件。

如果您的代理身分驗證憑證發生變更，則請編輯外部金鑰存放區的[憑證設定](#)。

10. 選擇 Create external key store (建立外部金鑰存放區)。

當程序成功時，新的外部金鑰存放區會出現在帳戶和區域的外部金鑰存放區清單中。如果不成功，則會出現錯誤訊息來描述問題，並提供如何修正的說明。如果您需要更多協助，請參閱[外部金鑰的 CreateKey 錯誤](#)。

下一步：不會自動連接新的外部金鑰存放區。您必須先[將外部金鑰存放區](#)連接到其外部金鑰存放區代理，才能 AWS KMS keys 在外部金鑰存放區中建立。

使用 AWS KMS API

您可以使用建 [CreateCustomKeyStore](#) 操作來建立新的外部金鑰存放區。如需尋找必要參數值的說明，請參閱外部金鑰存放區代理或金鑰管理軟體的文件。

Tip

在使用 CreateCustomKeyStore 操作時，您無法上傳[代理組態檔案](#)。但是，您可以使用代理組態檔案中的值來確保參數值正確無誤。

若要建立外部金鑰存放區，CreateCustomKeyStore 操作需要下列參數值。

- CustomKeyStoreName – 外部金鑰存放區在帳戶中唯一的易用名稱。

Important

請勿在此欄位包含機密或敏感資訊。此欄位可能以純文字顯示於 CloudTrail 日誌與其他輸出。

- CustomKeyStoreType – 指定 EXTERNAL_KEY_STORE。

- [XksProxyConnectivity](#) – 指定 PUBLIC_ENDPOINT 或 VPC_ENDPOINT_SERVICE。
- [XksProxyAuthenticationCredential](#) – 指定存取金鑰 ID 和私密存取金鑰 ID。
- [XksProxyUriEndpoint](#) – AWS KMS 用於與外部金鑰存放區代理通訊的端點。
- [XksProxyUriPath](#) – 連接代理 API 的代理內路徑。
- [XksProxyVpcEndpointServiceName](#) – 僅當 XksProxyConnectivity 值為 VPC_ENDPOINT_SERVICE 時才需要。

Note

如果您使用 1.0 AWS CLI 版，請在指定具有 HTTP 或 HTTPS 值的參數之前執行下列命令，例如 XksProxyUriEndpoint 參數。

```
aws configure set cli_follow_urlparam false
```

否則，AWS CLI 版本 1.0 會將參數值取代為在該 URI 地址找到的內容，造成下列錯誤：

```
Error parsing parameter '--xks-proxy-uri-endpoint': Unable to retrieve
https:// : received non 200 status code of 404
```

下列範例使用虛構值。在執行命令之前，請將其取代為外部金鑰存放區的有效值。

建立具有公有端點連接的外部金鑰存放區。

```
$ aws kms create-custom-key-store
  --custom-key-store-name ExampleExternalKeyStorePublic \
  --custom-key-store-type EXTERNAL_KEY_STORE \
  --xks-proxy-connectivity PUBLIC_ENDPOINT \
  --xks-proxy-uri-endpoint https://myproxy.xks.example.com \
  --xks-proxy-uri-path /kms/xks/v1 \
  --xks-proxy-authentication-credential
AccessKeyId=<value>,RawSecretAccessKey=<value>
```

建立具有 VPC 端點服務連接的外部金鑰存放區。

```
$ aws kms create-custom-key-store
  --custom-key-store-name ExampleExternalKeyStoreVPC \
```

```
--custom-key-store-type EXTERNAL_KEY_STORE \  
--xks-proxy-connectivity VPC_ENDPOINT_SERVICE \  
--xks-proxy-vpc-endpoint-service-name com.amazonaws.vpce.us-east-1.vpce-svc-  
example \  
--xks-proxy-uri-endpoint https://myproxy-private.xks.example.com \  
--xks-proxy-uri-path /kms/xks/v1 \  
--xks-proxy-authentication-credential  
AccessKeyId=<value>,RawSecretAccessKey=<value>
```

當操作成功時，CreateCustomKeyStore 會傳回自訂金鑰存放區 ID，如下回應範例所示。

```
{  
  "CustomKeyId": cks-1234567890abcdef0  
}
```

如果操作失敗，請修正例外狀況所指出的錯誤，然後重試。如需其他說明，請參閱[外部金鑰存放區故障診斷](#)。

下一步：若要使用外部金鑰存放區，請[將其連接至其外部金鑰存放區代理](#)。

編輯外部金鑰存放區屬性

您可以編輯現有外部金鑰存放區的所選屬性。

您可以在連接或中斷連接外部金鑰存放區時編輯某些屬性。對於其他屬性，您必須先[中斷外部金鑰存放區](#)與其外部金鑰存放區代理的連接。外部金鑰存放區的[連接狀態](#)必須是 DISCONNECTED。當外部金鑰存放區中斷連接時，您可以管理金鑰存放區和其 KMS 金鑰，但無法在外部金鑰存放區中建立或使用 KMS 金鑰。若要尋找外部金鑰存放區的[連接狀態](#)，請使用 [DescribeCustomKeyStores](#) 操作，或參閱外部金鑰存放區詳細資訊頁面上的 General configuration (一般組態) 區段。

更新外部金鑰存放區屬性之前，AWS KMS 會使用新值將 [GetHealthStatus](#) 請求傳送至外部金鑰存放區代理。如果請求成功，表示您可以使用更新後的屬性值連接至外部金鑰存放區代理並進行身分驗證。如果請求失敗，編輯操作將失敗，並顯示識別錯誤的例外狀況。

編輯操作完成時，外部金鑰存放區的更新屬性值會顯示在 AWS KMS 主控台和 DescribeCustomKeyStores 回應中。但是，變更需要 5 分鐘才能完全生效。

如果您在 AWS KMS 主控台中編輯外部金鑰存放區，您可以選擇上傳 JSON 型[代理組態檔案](#)，以指定[代理 URI 路徑](#)和[代理身分驗證憑證](#)。某些外部金鑰存放區代理會為您產生此檔案。如需詳細資訊，請參閱外部金鑰存放區代理或外部金鑰管理器的文件。

⚠ Warning

更新的屬性值必須將外部金鑰存放區連接至與先前值相同的外部金鑰管理器的代理，或使用相同密碼編譯金鑰之外部金鑰管理器的備份或快照。如果您的外部金鑰存放區永久無法存取與其 KMS 金鑰相關聯的外部金鑰，則使用這些外部金鑰加密的密文將無法復原。特別是，變更外部金鑰存放區的代理連線可能會 AWS KMS 阻止 存取您的外部金鑰。

💡 Tip

有些外部金鑰管理器會提供更簡單的方法來編輯外部金鑰存放區屬性。如需詳細資訊，請參閱外部金鑰管理器文件。

您可以變更外部金鑰存放區的下列屬性。

可編輯的外部金鑰存放區屬性	任何連接狀態	需要已中斷連接狀態
自訂金鑰存放區名稱		
自訂金鑰存放區的必要易記名稱。		
 Important 請勿在此欄位包含機密或敏感資訊。此欄位可能在 CloudTrail 日誌與其他輸出中以純文字顯示。		
代理身分驗證憑證 (XksProxyAuthenticationCredential) (您必須同時指定存取金鑰 ID 和私密存取金鑰，即使您只變更一個元素。)		
代理 URI 路徑 (XksProxyUriPath)		
代理連接 (XksProxyConnectivity)		

可編輯的外部金鑰存放區屬性	任何連接狀態	需要已中斷連接狀態
(您也必須更新代理 URI 端點。如果您要變更為 VPC 端點服務連接，則必須指定代理 VPC 端點服務名稱。)		
代理 URI 端點 (XksProxyUriEndpoint) 如果變更代理端點 URI，則您可能也需要變更相關的 TLS 憑證。		✓
代理 VPC 端點服務名稱 (XksProxyVpcEndpointServiceName) (VPC 端點服務連接需要此欄位)		✓
VPC 端點服務擁有者 (XksProxyVpcEndpointServiceOwner) (VPC 端點服務連接需要此欄位)		✓

編輯外部金鑰存放區的屬性

您可以在 AWS KMS 主控台或使用 [UpdateCustomKeyStore](#) 操作來編輯外部金鑰存放區的屬性。

使用 AWS KMS 主控台

編輯金鑰存放區時，您可以變更任何可編輯的值。某些變更需要將外部金鑰存放區與其外部金鑰存放區代理中斷連接。

如果您正在編輯代理 URI 路徑或代理身分驗證憑證，則可以輸入新值或上傳包含新值的外部金鑰存放區 [代理組態檔案](#)。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，依次選擇 Custom key stores (自訂金鑰存放區)、External key stores (外部金鑰存放區)。
4. 選擇您要編輯的金鑰存放區。

- 如果需要，將外部金鑰存放區與其外部金鑰存放區代理中斷連接。從 Key store actions (金鑰存放區動作) 選單中，選擇 Disconnect (中斷連接)。
5. 從 Key store actions (金鑰存放區動作) 選單中，選擇 Edit (編輯)。
 6. 變更一個或多個可編輯的外部金鑰存放區屬性。您也可以上傳外部金鑰存放區[代理組態檔案](#)，其中包含代理 URI 路徑和代理身分驗證憑證的值。即使檔案中指定的某些值尚未變更，您仍可使用代理組態檔案。
 7. 選擇 Update external key store (更新外部金鑰存放區)。
 8. 檢閱警告，如果您決定繼續，請確認警告，然後選擇 Update external key store (更新外部金鑰存放區)。

當程序成功時，會出現訊息描述您編輯的屬性。當操作失敗時，會出現錯誤訊息，其中描述問題並提供如何修正的協助。

9. 如有必要，請重新連接外部金鑰存放區。從 Key store actions (金鑰存放區動作) 選單中，選擇 Connect (連接)。

您可以將外部金鑰存放區保持中斷連接。中斷連接時，您無法在外部金鑰存放區中建立 KMS 金鑰，並且無法在[密碼編譯操作](#)中使用外部金鑰存放區中的 KMS 金鑰。

使用 AWS KMS API

若要變更外部金鑰存放區的屬性，請使用 [UpdateCustomKeyStore](#) 操作。您可以在相同操作中變更外部金鑰存放區的多個屬性。如果操作成功，會 AWS KMS 傳回 HTTP 200 回應和沒有屬性的 JSON 物件。

使用 CustomKeyId 參數可識別外部金鑰存放區。使用其他參數來變更屬性。您無法將[代理組態檔案](#)用於 UpdateCustomKeyStore 操作。代理組態檔案僅主控台支援 AWS KMS。不過，您可以使用代理組態檔案來協助您判斷外部金鑰存放區代理的正確參數值。

本節中的範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

在開始之前，[如果需要](#)，將[外部金鑰存放區](#)與其外部金鑰存放區代理中斷連接。更新之後，[如果需要](#)，您可以將[外部金鑰存放區重新連接](#)至其外部金鑰存放區代理。您可以將外部金鑰存放區保持在中斷連接狀態，但必須先重新將其連接，才能在金鑰存放區中建立新的 KMS 金鑰或使用金鑰存放區中的現有 KMS 金鑰進行密碼編譯操作。

Note

如果您使用 1.0 AWS CLI 版，請在指定具有 HTTP 或 HTTPS 值的參數之前執行下列命令，例如 `XksProxyUriEndpoint` 參數。

```
aws configure set cli_follow_urlparam false
```

否則，AWS CLI 版本 1.0 會將參數值取代為在該 URI 地址找到的內容，造成下列錯誤：

```
Error parsing parameter '--xks-proxy-uri-endpoint': Unable to retrieve
https:// : received non 200 status code of 404
```

變更外部金鑰存放區的名稱

第一個範例使用 [UpdateCustomKeyStore](#) 操作，將外部金鑰存放區的易記名稱變更為 `XksKeyStore`。此命令使用 `CustomKeyStoreId` 參數來識別自訂金鑰存放區和 `CustomKeyStoreName` 來指定自訂金鑰存放區的新名稱。將所有範例值取代為外部金鑰存放區的實際值。

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 --new-
custom-key-store-name XksKeyStore
```

變更代理身分驗證憑證

下列範例會更新 AWS KMS 用來驗證外部金鑰存放區代理的代理身分驗證憑證。如果在代理上輪換憑證，則可以使用類似這樣的命令來更新憑證。

請先更新外部金鑰存放區代理上的憑證。然後使用此功能將變更報告給 AWS KMS。（您的代理會短暫支援舊憑證和新憑證，讓您可以有時間在其中更新憑證 AWS KMS。）

您必須在憑證中同時指定存取金鑰 ID 和私密存取金鑰，即使只變更一個值。

前兩個命令設定變數來保留憑證值。UpdateCustomKeyStore 操作使用 `CustomKeyStoreId` 參數識別外部金鑰存放區。其會使用 `XksProxyAuthenticationCredential` 參數及其 `AccessKeyId` 和 `RawSecretAccessKey` 欄位來指定新的憑證。將所有範例值取代為外部金鑰存放區的實際值。

```
$ accessKeyID=access key id
$ secretAccessKey=secret access key
```

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 \
  --xks-proxy-authentication-credential \
    AccessKeyId=$accessKeyId,RawSecretAccessKey=$secretAccessKey
```

變更代理 URI 路徑

下列範例會更新代理 URI 路徑 (XksProxyUriPath)。代理 URI 端點和代理 URI 路徑的組合在 AWS 帳戶 和 區域中必須是唯一的。將所有範例值取代為外部金鑰存放區的實際值。

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 \
  --xks-proxy-uri-path /kms/xks/v1
```

變更為 VPC 端點服務連接

下列範例使用 [UpdateCustomKeyStore](#) 操作，將外部金鑰存放區代理連接類型變更為 VPC_ENDPOINT_SERVICE。若要進行此變更，您必須指定 VPC 端點服務連接的必要值，包括 VPC 端點服務名稱 (XksProxyVpcEndpointServiceName) 和包含 VPC 端點服務之私有 DNS 名稱的代理 URI 端點 (XksProxyUriEndpoint) 值。將所有範例值取代為外部金鑰存放區的實際值。

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 \
  --xks-proxy-connectivity "VPC_ENDPOINT_SERVICE" \
  --xks-proxy-uri-endpoint https://myproxy-private.xks.example.com \
  --xks-proxy-vpc-endpoint-service-name com.amazonaws.vpce.us-east-1.vpce-  
svc-example
```

變更為公有端點連接

下列範例會將外部金鑰存放區代理連接類型變更為 PUBLIC_ENDPOINT。當您進行此變更時，必須更新代理 URI 端點 (XksProxyUriEndpoint) 值。將所有範例值取代為外部金鑰存放區的實際值。

Note

與公有端點連接相比，VPC 端點連接可提供更高的安全性。在變更為公有端點連接之前，請考慮其他選項，包括在內部部署中尋找外部金鑰存放區代理，以及僅將 VPC 用於通訊。

```
$ aws kms update-custom-key-store --custom-key-store-id cks-1234567890abcdef0 \
  --xks-proxy-connectivity "PUBLIC_ENDPOINT" \
```

```
--xks-proxy-uri-endpoint https://myproxy.xks.example.com
```

檢視外部金鑰存放區

您可以使用 AWS KMS 主控台或使用 [DescribeCustomKeyStores](#) 操作來檢視每個帳戶和區域中的外部金鑰存放區。

檢視外部金鑰存放區時，您可以查看下列項目：

- 有關金鑰存放區的基本資訊，包括其易記名稱、ID、金鑰存放區類型和建立日期。
- [外部金鑰存放區代理](#)的組態資訊，包括[連接類型](#)、[代理 URI 端點](#)和[路徑](#)，以及當前[代理身分驗證憑證](#)的[存取金鑰 ID](#)。
- 如果外部金鑰存放區代理使用 [VPC 端點服務連接](#)，則主控台會顯示 VPC 端點服務的名稱。
- 當前的[連接狀態](#)。

Note

連接狀態值 Disconnected (已中斷連接) 指示外部金鑰存放區從未連接，或它是特意從其外部金鑰存放區代理中斷連接。不過，如果嘗試在已連接的外部金鑰存放區中使用 KMS 金鑰失敗，可能表示外部金鑰存放區或其代理發生問題。如需協助，請參閱 [外部金鑰存放區連接錯誤](#)。

- 具有 [Amazon CloudWatch 指標](#)圖形的 [Monitoring](#) (監控) 區段旨在協助您偵測並解決外部金鑰存放區的問題。如需解譯圖形、在規劃和疑難排解中使用圖形、以及根據圖形中的指標建立 CloudWatch 警示的說明，請參閱 [監控外部金鑰存放區](#)。

外部金鑰存放區屬性

AWS KMS 主控台和 [DescribeCustomKeyStores](#) 回應中會顯示外部金鑰存放區的下列屬性。

自訂金鑰存放區屬性

下列值會出現在每個自訂金鑰存放區詳細資訊頁面的一般組態區段中。這些屬性適用於所有自訂金鑰存放區，包括 AWS CloudHSM 金鑰存放區和外部金鑰存放區。

自訂金鑰存放區 ID

AWS KMS 指派給自訂金鑰存放區的唯一 ID。

自訂金鑰存放區名稱

您在建立自訂金鑰存放區時指派給它的易記名稱。您可隨時變更此值。

自訂金鑰存放區類型

自訂金鑰存放區的類型。有效值為 AWS CloudHSM (AWS_CLOUDHSM) 或外部金鑰存放區 (EXTERNAL_KEY_STORE)。您無法在建立自訂金鑰存放區之後變更類型。

Creation (建立) 日期

建立自訂金鑰存放區的日期。此日期顯示為 AWS 區域的本地時間。

連線狀態

指示自訂金鑰存放區是否已連接至其備份金鑰存放區。只有當自訂金鑰存放區從未連接至其備份金鑰存放區，或已故意中斷連接時，連接狀態才為 DISCONNECTED。如需詳細資訊，請參閱[the section called “連線狀態”](#)。

外部金鑰存放區組態屬性

下列值會顯示在每個外部金鑰存放區之詳細資訊頁面的 External key store proxy configuration (外部金鑰存放區代理組態) 區段以及 [DescribeCustomKeyStores](#) 回應的 XksProxyConfiguration 元素中。如需每個欄位的詳細說明，包括唯一性要求以及決定每個欄位正確值的說明，請參閱「建立外部金鑰存放區」主題中的 [the section called “備妥先決條件”](#)。

代理連接

指示外部金鑰存放區是使用[公有端點連接](#)還是[VPC 端點服務連接](#)。

代理 URI 端點

AWS KMS 用來連線至[外部金鑰存放區代理](#)的端點。

代理 URI 路徑

AWS KMS 傳送代理 [API 請求的代理](#) URI 端點路徑。

代理憑證：存取金鑰 ID

您在外部金鑰存放區代理上建立的[代理身分驗證憑證](#)的一部分。存取金鑰 ID 可識別憑證中的私密存取金鑰。

AWS KMS 使用 SigV4 簽署程序和代理身分驗證憑證來簽署其對外部金鑰存放區代理的請求。簽章中的登入資料可讓外部金鑰存放區代理代表您從中驗證請求 AWS KMS。

VPC 端點服務名稱

支援外部金鑰存放區的 Amazon VPC 端點服務名稱。只有當外部金鑰存放區使用 [VPC 端點服務連接](#)時，才會顯示此值。您可以在 VPC 中找到外部金鑰存放區代理，或使用 VPC 端點服務與外部金鑰存放區代理進行安全通訊。

VPC 端點服務擁有者 ID

支援外部金鑰存放區的 Amazon VPC 端點服務的 ID。只有當外部金鑰存放區使用 [VPC 端點服務連接](#)時，才會顯示此值。您可以在 VPC 中找到外部金鑰存放區代理，或使用 VPC 端點服務與外部金鑰存放區代理進行安全通訊。

檢視外部金鑰存放區屬性

您可以在 AWS KMS 主控台或使用 [DescribeCustomKeyStores](#) 操作來檢視外部金鑰存放區及其相關聯的屬性。

使用 AWS KMS 主控台

若要檢視指定帳戶和區域中的外部金鑰存放區，請使用下列程序。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，依次選擇 Custom key stores (自訂金鑰存放區)、External key stores (外部金鑰存放區)。
4. 若要檢視有關外部金鑰存放區的詳細資訊，請選擇金鑰存放區名稱。

使用 AWS KMS API

若要檢視您的外部金鑰存放區，請使用 [DescribeCustomKeyStores](#) 操作。在預設情況下，此操作會傳回帳戶和區域中的所有自訂金鑰存放區。但是，您可以使用 CustomKeyId 或 CustomKeyName 參數 (但不能同時使用) 來限制對特定自訂金鑰存放區的輸出。

若為自訂金鑰存放區，則輸出包含自訂金鑰存放區 ID、名稱和類型以及金鑰存放區的[連接狀態](#)。如果連接狀態為 FAILED，則輸出也會包含描述錯誤原因的 ConnectionErrorCode。如需解譯外部金鑰存放區的 ConnectionErrorCode 說明，請參閱 [外部金鑰存放區的連接錯誤代碼](#)。

若為外部金鑰存放區，輸出還包括 XksProxyConfiguration 元素。此元素包括[連接類型](#)、[代理 URI 端點](#)、[代理 URI 路徑](#)以及[代理身分驗證憑證](#)的存取金鑰 ID。

本節中的範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

例如，以下命令會傳回帳戶和區域中的所有自訂金鑰存放區。您可以使用 Limit 和 Marker 參數來切換輸出中的自訂金鑰存放區頁面。

```
$ aws kms describe-custom-key-stores
```

以下命令使用 CustomKeyStoreName 參數來取得僅具有 ExampleXksPublic 易記名稱的範例外部金鑰存放區。此範例金鑰存放區使用公有端點連接。它連接到其外部金鑰存放區代理。

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleXksPublic
{
  "CustomKeyStores": [
    {
      "CustomKeyId": "cks-1234567890abcdef0",
      "CustomKeyName": "ExampleXksPublic",
      "ConnectionState": "CONNECTED",
      "CreationDate": "2022-12-14T20:17:36.419000+00:00",
      "CustomKeyType": "EXTERNAL_KEY_STORE",
      "XksProxyConfiguration": {
        "AccessKeyId": "ABCDE12345670EXAMPLE",
        "Connectivity": "PUBLIC_ENDPOINT",
        "UriEndpoint": "https://xks.example.com:6443",
        "UriPath": "/example/prefix/kms/xks/v1"
      }
    }
  ]
}
```

下列命令取得具有 VPC 端點服務連接的範例外部金鑰存放區。在此範例中，外部金鑰存放區會連接至其外部金鑰存放區代理。

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleXksVpc
{
  "CustomKeyStores": [
    {
      "CustomKeyId": "cks-9876543210fedcba9",
      "CustomKeyName": "ExampleXksVpc",
      "ConnectionState": "CONNECTED",
      "CreationDate": "2022-12-13T18:34:10.675000+00:00",
      "CustomKeyType": "EXTERNAL_KEY_STORE",

```

```

    "XksProxyConfiguration": {
      "AccessKeyId": "ABCDE98765432EXAMPLE",
      "Connectivity": "VPC_ENDPOINT_SERVICE",
      "UriEndpoint": "https://example-proxy-uri-endpoint-vpc",
      "UriPath": "/example/prefix/kms/xks/v1",
      "VpcEndpointServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-example"
    }
  }
]
}

```

ConnectionState 的 **Disconnected** 值指示外部金鑰存放區從未連接，或它是特意從其外部金鑰存放區代理中斷連接。不過，如果嘗試在已連接的外部金鑰存放區中使用 KMS 金鑰失敗，可能表示外部金鑰存放區代理或其他外部元件發生問題。

如果外部金鑰存放區的 **ConnectionState** 為 **FAILED**，則 **DescribeCustomKeyStores** 回應會包含一個 **ConnectionErrorCode** 元素，解釋錯誤的原因。

例如，在以下輸出中，**XKS_PROXY_TIMED_OUT** 值指出 AWS KMS 可以連線至外部金鑰存放區代理，但連線失敗，因為外部金鑰存放區代理未在分配 AWS KMS 的時間內回應。如果您重複看到此連接錯誤代碼，則請通知您的外部金鑰存放區代理廠商。如需此錯誤和其他連接錯誤失敗的協助，請參閱[外部金鑰存放區故障診斷](#)。

```

$ aws kms describe-custom-key-stores --custom-key-store-name ExampleXksVpc
{
  "CustomKeyStores": [
    {
      "CustomKeyId": "cks-9876543210fedcba9",
      "CustomKeyName": "ExampleXksVpc",
      "ConnectionState": "FAILED",
      "ConnectionErrorCode": "XKS_PROXY_TIMED_OUT",
      "CreationDate": "2022-12-13T18:34:10.675000+00:00",
      "CustomKeyType": "EXTERNAL_KEY_STORE",
      "XksProxyConfiguration": {
        "AccessKeyId": "ABCDE98765432EXAMPLE",
        "Connectivity": "VPC_ENDPOINT_SERVICE",
        "UriEndpoint": "https://example-proxy-uri-endpoint-vpc",
        "UriPath": "/example/prefix/kms/xks/v1",
        "VpcEndpointServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-example"
      }
    }
  ]
}

```

```
}
```

監控外部金鑰存放區

AWS KMS 會收集與外部金鑰存放區每次互動的指標，並將其發佈到您的 CloudWatch 帳戶中。這些指標用於在每個外部金鑰存放區之詳細資訊頁面的監控區段中產生圖表。下列主題詳細說明如何使用圖表來識別會影響外部金鑰存放區的操作和組態問題，並進行疑難排解。建議您使用 CloudWatch 指標來設定警示，以便在外部金鑰存放區未按預期執行時通知您。如需詳細資訊，請參閱[透過 Amazon CloudWatch 進行監控](#)。

主題

- [檢視圖表](#)
- [解釋圖表](#)

檢視圖表

您可以使用不同的詳細資料層級檢視圖表。依預設，每個圖表使用三小時的時間範圍和五分鐘的彙總期間。您可以在主控台內調整圖表檢視，但是當外部金鑰存放區詳細資訊頁面關閉或瀏覽器重新整理時，您的變更將還原為預設設定。如需有關 Amazon CloudWatch 術語的說明，請參閱[Amazon CloudWatch 概念](#)。

檢視資料點詳細資訊

每個圖表中的資料是依 [AWS KMS 指標](#) 收集的。若要檢視有關特定資料點的詳細資訊，請將滑鼠暫停在折線圖上的資料點上。這會顯示一個彈出式視窗，其中包含有關衍生出該圖表之指標的詳細資訊。每個清單項目都會顯示在該資料點記錄的維度值。如果該資料點的維度值沒有可用的指標資料，彈出式視窗會顯示空值 (-)。有些圖表會記錄單一資料點的多個維度和值。其他圖表 (例如 [可靠性圖表](#)) 會使用指標收集的資料來計算唯一值。每個清單項目都與不同的折線圖顏色相關聯。

修改時間範圍

若要修改 [時間範圍](#)，請選取監控區段右上角其中一種預先定義的時間範圍。預先定義的時間範圍為 1 小時到 1 週 (1h (1 小時)、3h (3 小時)、12h (12 小時)、1d (1 天)、3d (3 天) 或 1w (1 週))。這會調整所有圖表的時間範圍。如果您想要檢視不同時間範圍內的特定圖表，或想要設定自訂時間範圍，請放大圖表或在 Amazon CloudWatch 主控台中檢視它。

放大圖表

您可以使用[迷你地圖縮放功能](#)，將焦點放在折線圖的某個區段以及圖表的某些部分上，而無需放大和縮小檢視之間進行變更。例如，您可以使用迷你地圖縮放功能，將焦點放在圖表中的峰值上，以便在同一時間軸中將尖峰與監控區段中的其他圖表進行比較。

1. 選擇並拖曳要聚焦的圖表區域，然後放開拖曳。
2. 若要重設縮放，請選擇 Reset zoom (重設縮放) 圖示，這看起來像一個帶有減號 (-) 符號的放大鏡。

放大圖表

若要放大圖表，請選取單個圖表右上角的選單圖示，然後選擇 Enlarge (放大)。您也可以選取當您將游標停留在圖表上時顯示在選單圖示旁邊的放大圖示。

放大圖表可讓您透過指定不同的時段、自訂時間範圍或重新整理間隔來進一步修改圖表檢視。當您關閉放大的檢視時，這些變更將恢復為預設設定。

修改期間

1. 選擇 Period options (期間選項) 功能表。依預設，此選單會顯示值：5 分鐘。
2. 選擇一個期間，預先定義的期間從 1 秒到 30 天不等。

例如，您可以選擇一分鐘檢視，這在疑難排解時非常有用。或者，選擇較不精細的一小時檢視。當檢視更廣泛的時間範圍 (例如 3 天) 時，您可以看到隨時間變化的趨勢。如需詳細資訊，請參閱和《Amazon CloudWatch 使用者指南》中的[期間](#)。

修改時間範圍或時區

1. 選取其中一個預先定義的時間範圍，範圍從 1 小時到 1 週不等 (1h (1 小時)、3h (3 小時)、12h (12 小時)、1d (1 天)、3d (3 天) 1w (1 週))。或者，您也可以選擇 Custom (自訂) 來設定您自己的時間範圍。
2. 選擇 Custom (自訂)。
 - a. Time range: (時間範圍：) 選取方塊左上角的 Absolute (絕對值) 索引標籤。使用行事曆選擇器或文字欄位方塊來指定時間範圍。
 - b. Time zone: (時區：) 選擇方塊右上角的下拉式選單。您可以將時區變更為 UTC 或者 Local time zone (本機時區)。
3. 指定時間範圍後，選擇 Apply (套用)。

修改圖表中資料重新整理的頻率

1. 選擇右上角的 Refresh options (重新整理選項) 選單。
2. 選擇重新整理間隔 (Off (關閉)、10 Seconds (10 秒)、1 Minute (1 分鐘)、2 Minutes (2 分鐘)、5 Minutes (5 分鐘) 或 15 Minutes (15 分鐘))。

在 Amazon CloudWatch 主控台中檢視圖表

監控區段中的圖表衍生自 AWS KMS 發佈到 Amazon CloudWatch 的預先定義指標。您可以在 CloudWatch 主控台中開啟它們，並將它們儲存到 CloudWatch 儀表板。如果您有多個外部金鑰存放區，則可以在 CloudWatch 中開啟各自的圖表，並將其儲存到單一儀表板，以比較其運作狀態和使用量。

新增至 CloudWatch 儀表板

選取右上角的 Add to dashboard (新增至儀表板)，將所有圖表新增至 Amazon CloudWatch 儀表板。您可以選取現有的儀表板或建立新的儀表板。如需使用此儀表板建立圖表和警示之自訂檢視的相關資訊，請參閱《Amazon CloudWatch 使用者指南》中的[使用 Amazon CloudWatch 儀表板](#)。

在 CloudWatch 指標中檢視

選取單個圖表右上角的選單圖示，然後選擇 View in metrics (在指標中檢視) 以在 Amazon CloudWatch 主控台中檢視此圖表。在 CloudWatch 主控台中，您可以將此單個圖表新增至儀表板，並修改時間範圍、期間和重新整理間隔。如需詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的[圖表指標](#)。

解釋圖表

AWS KMS 提供數個圖形來監控 AWS KMS 主控台內外部金鑰存放區的運作狀態。這些圖表會自動設定並衍生自 [AWS KMS 指標](#)。

作為您對外部金鑰存放區和外部金鑰進行呼叫的一部分來收集圖表資料。您可能會在未進行任何呼叫的時間範圍內看到資料填入圖表，此資料來自代表您 AWS KMS 進行的定期 GetHealthStatus 呼叫，以檢查外部金鑰存放區代理和外部金鑰管理器的狀態。如果您的圖表顯示 No data available (沒有資料可用) 訊息，則在該時間範圍內沒有記錄任何呼叫，或者您的外部金鑰存放區處於 [DISCONNECTED](#) 狀態。透過將[檢視調整](#)為更寬的時間範圍，可確定外部金鑰存放區中斷連接的時間。

主題

- [請求總數](#)
- [可靠性](#)

- [延遲](#)
- [前 5 個例外狀況](#)
- [憑證到期天數](#)

請求總數

在指定時間範圍內，針對特定外部金鑰存放區收到的 AWS KMS 請求總數。使用此圖表來確定是否有限流風險。

AWS KMS 建議您的外部金鑰管理器每秒最多能夠處理 1800 個密碼編譯操作請求。如果五分鐘內的呼叫達到 540,000 個，則有限流風險。

您可以在使用指標 AWS KMS 調節的外部金鑰存放區中，監控 KMS [ExternalKeyStoreThrottle](#) 金鑰上密碼編譯操作的請求數量。

如果您經常收到 `KMSInvalidStateException` 錯誤，並且訊息說明「因為請求率非常高」而拒絕請求，則可能表示您的外部金鑰管理器或外部金鑰存放區代理無法跟上目前的請求率。如果可能，請降低您的請求率。您也可以考慮請求減少自訂金鑰存放區請求配額值。減少此配額值可能會增加限流，但表示在將多餘的請求傳送到外部金鑰存放區代理或外部金鑰管理器之前 AWS KMS，會快速拒絕這些請求。若要請求減少配額，請造訪 [AWS 支援中心](#) 並建立案例。

總請求圖表衍生自 [XksProxyErrors](#) 指標，其會收集 AWS KMS 從外部金鑰存放區代理接收的成功與失敗回應的相關資料。當您[檢視特定資料點](#)時，快顯視窗會顯示 `CustomKeyStoreId` 維度的值，以及在該資料點記錄的 AWS KMS 請求總數。`CustomKeyStoreId` 將永遠是相同的。

可靠性

外部金鑰存放區代理傳回成功回應或不可重試錯誤的 AWS KMS 請求百分比。使用此圖表來評估外部金鑰存放區代理的操作運作狀態。

當圖表顯示的值小於 100% 時，表示代理未回應或回應時出現可重試錯誤。這可能表示網路出現問題、外部金鑰存放區代理或外部金鑰管理器運行緩慢或實作錯誤。

如果請求包含錯誤的憑證並且您的代理回應時出現 `AuthenticationFailedException`，則圖表仍會顯示 100% 的可靠性，因為代理在[外部金鑰存放區代理請求](#)中發現錯誤值，因此預期會發生失敗。如果可靠性圖表的百分比為 100%，則外部金鑰存放區代理會按預期回應。如果圖表顯示的值小於 100%，則代理回應時會出現可重試錯誤或逾時。例如，如果由於請求率非常高，代理回應時出現 `ThrottlingException`，則會顯示較低的可靠性百分比，因為代理無法識別導致它失敗的請求中的特定問題。這是因為可重試的錯誤很可能是暫時性的問題，可以透過重試請求來解決。

下列錯誤回應會降低可靠性百分比。您可以使用 [前 5 個例外狀況](#) 圖表和 [XksProxyErrors](#) 指標進一步監控代理傳回每個可重試錯誤的頻率。

- `InternalException`
- `DependencyTimeoutException`
- `ThrottlingException`
- `XksProxyUnreachableException`

可靠性圖表衍生自 [XksProxyErrors](#) 指標，該指標會收集從外部金鑰存放區代理 AWS KMS 接收的成功和失敗回應資料。只有當回應的 `ErrorType` 值為 `Retryable` 時，才會降低可靠性百分比。當您[檢視特定資料點](#)時，快顯視窗會顯示維 `CustomKeyStoreId` 度的值，以及在該資料點記錄之 AWS KMS 請求的可靠性百分比。`CustomKeyStoreId` 將永遠是相同的。

我們建議您使用 [XksProxyErrors](#) 指標來建立 CloudWatch 警示，以便在一分鐘內記錄超過五個可重試的錯誤時提醒您，從而通知您存在潛在的網路問題。如需詳細資訊，請參閱[建立可重試錯誤的警示](#)。

延遲

外部金鑰存放區代理回應 AWS KMS 請求所需的毫秒數。使用此圖表可評估外部金鑰存放區代理和外部金鑰管理器的效能。

AWS KMS 預期外部金鑰存放區代理會在 250 毫秒內回應每個請求。在網路逾時的情況下，AWS KMS 會重試一次請求。如果代理第二次失敗，則記錄的延遲是兩次請求嘗試的合併逾時限制，圖表將顯示約 500 毫秒。在代理未在 250 毫秒逾時限制內進行回應的所有其他情況下，記錄的延遲為 250 毫秒。如果代理在加密和解密操作中經常逾時，請諮詢您的外部代理管理員。如需解決延遲問題的說明，請參閱 [延遲和逾時錯誤](#)。

回應緩慢也可能表示您的外部金鑰管理器無法處理目前的請求流量。AWS KMS 建議外部金鑰管理器每秒最多可處理 1800 個密碼編譯操作的請求。如果您的外部金鑰管理器無法處理每秒 1800 個請求，請考慮請求減少[自訂金鑰存放區中 KMS 金鑰的請求配額](#)。使用外部金鑰存放區中的 KMS 金鑰進行密碼編譯操作的請求會快速檢錯，並發生[限流例外狀況](#)，而不是被外部金鑰存放區代理或外部金鑰管理器處理並拒絕。

延遲圖表衍生自 [XksProxyLatency](#) 指標。當您[檢視特定資料點](#)時，彈出式視窗會顯示相應的 `KmsOperation` 和 `XksOperation` 維度值，以及在該資料點記錄的平均操作延遲。清單項目會從最高延遲到最低延遲排序。

建議您使用 [XksProxyLatency](#) 指標來建立 CloudWatch 警示，在您的延遲臨近逾時限制時通知您。如需詳細資訊，請參閱[建立回應逾時的警示](#)。

前 5 個例外狀況

指定時間範圍內失敗的密碼編譯和管理操作的前五個例外狀況。使用此圖表可追蹤最常見的錯誤，因此您可以優先考慮工程工作。

此計數包含從外部金鑰存放區代理 AWS KMS 收到的例外 `XksProxyUnreachableException` 狀況，以及當無法與外部金鑰存放區代理建立通訊時，在內部 AWS KMS 傳回的例外狀況。

高比率的可重試錯誤可能表示存在網路錯誤，而高比率的不可重試錯誤可能表示外部金鑰存放區的組態有問題。例如，中的峰值 `AuthenticationFailedExceptions` 表示在 中設定的身分驗證憑證 AWS KMS 與外部金鑰存放區代理之間的差異。若要檢視外部金鑰存放區組態，請參閱 [檢視外部金鑰存放區](#)。若要編輯外部金鑰存放區設定，請參閱 [編輯外部金鑰存放區屬性](#)。

從外部金鑰存放區代理 AWS KMS 接收的例外狀況與 操作失敗時 AWS KMS 傳回的例外狀況不同。AWS KMS 加密操作 `KMSInvalidStateException` 會針對與外部金鑰存放區之外部組態或連線狀態相關的所有失敗傳回。若要識別問題，請使用隨附的錯誤訊息文字。

下表顯示會出現在前 5 個例外狀況圖表中的例外狀況，以及 AWS KMS 傳回給您的對應例外狀況。

錯誤類型	圖表中顯示的例外狀況	AWS KMS 傳回給您的例外狀況
不可重試	AccessDeniedException 如需故障診斷協助，請參閱 代理授權問題 。	CustomKeyStoreInvalidStateException 回應 <code>CreateKey</code> 操作。 KMSInvalidStateException 回應密碼編譯操作。
不可重試	AuthenticationFailedException 如需故障診斷協助，請參閱 身分驗證憑證錯誤 。	XksProxyIncorrectAuthenticationCredentialException 回應 <code>CreateCustomKeyStore</code> 和 <code>UpdateCustomKeyStore</code> 操作。 CustomKeyStoreInvalidStateException 回應 <code>CreateKey</code> 操作。

錯誤類型	圖表中顯示的例外狀況	AWS KMS 傳回給您的例外狀況
可重試	DependencyTimeoutException 如需故障診斷協助，請參閱 延遲和逾時錯誤 。	KMSInvalidStateException 回應密碼編譯操作。
		XksProxyUriUnreachableException 回應 CreateCustomKeyStore 和 UpdateCustomKeyStore 操作。
		CustomKeyStoreInvalidStateException 回應 CreateKey 操作。 KMSInvalidStateException 回應密碼編譯操作。
可重試	InternalException 外部金鑰存放區代理拒絕了請求，因為其無法與外部金鑰管理器通訊。確認外部金鑰存放區代理組態正確，以及外部金鑰管理器可使用。	XksProxyInvalidResponseException 回應 CreateCustomKeyStore 和 UpdateCustomKeyStore 操作。 CustomKeyStoreInvalidStateException 回應 CreateKey 操作。 KMSInvalidStateException 回應密碼編譯操作。
不可重試	InvalidCiphertextException 如需故障診斷協助，請參閱 解密錯誤 。	KMSInvalidStateException 回應密碼編譯操作。

錯誤類型	圖表中顯示的例外狀況	AWS KMS 傳回給您的例外狀況
不可重試	InvalidKeyUsageException 如需故障診斷協助，請參閱 外部金鑰的密碼編譯操作錯誤 。	XksKeyInvalidConfigurationException 回應 CreateKey 操作。 KMSInvalidStateException 回應密碼編譯操作。
不可重試	InvalidStateException 如需故障診斷協助，請參閱 外部金鑰的密碼編譯操作錯誤 。	XksKeyInvalidConfigurationException 回應 CreateKey 操作。 KMSInvalidStateException 回應密碼編譯操作。
不可重試	InvalidUriPathException 如需故障診斷協助，請參閱 一般組態錯誤 。	XksProxyInvalidConfigurationException 回應 CreateCustomKeyStore 和 UpdateCustomKeyStore 操作。 CustomKeyStoreInvalidStateException 回應 CreateKey 操作。 KMSInvalidStateException 回應密碼編譯操作。
不可重試	KeyNotFoundException 如需故障診斷協助，請參閱 外部金鑰錯誤 。	XksKeyNotFoundException 回應 CreateKey 操作。 KMSInvalidStateException 回應密碼編譯操作。

錯誤類型	圖表中顯示的例外狀況	AWS KMS 傳回給您的例外狀況
可重試	ThrottlingException 由於請求率非常高，所以外部金鑰存放區代理拒絕了請求。減少使用此外部金鑰存放區中的 KMS 金鑰進行呼叫的頻率。	XksProxyUriUnreachableException 回應 CreateCustomKeyStore 和 UpdateCustomKeyStore 操作。 CustomKeyStoreInvalidStateException 回應 CreateKey 操作。 KMSInvalidStateException 回應密碼編譯操作。
不可重試	UnsupportedOperationException 如需故障診斷協助，請參閱 外部金鑰的密碼編譯操作錯誤 。	XksKeyInvalidResponseException 回應 CreateKey 操作。 KMSInvalidStateException 回應密碼編譯操作。
不可重試	ValidationException 如需故障診斷協助，請參閱 代理問題 。	XksProxyInvalidResponseException 回應 CreateCustomKeyStore 和 UpdateCustomKeyStore 操作。 CustomKeyStoreInvalidStateException 回應 CreateKey 操作。 KMSInvalidStateException 回應密碼編譯操作。

錯誤類型	圖表中顯示的例外狀況	AWS KMS 傳回給您的例外狀況
可重試	XksProxyUnreachableException 如果您重複看到此錯誤，則請確認外部金鑰存放區代理處於作用中狀態且已連接至網路，以及外部金鑰存放區中的 URI 路徑和端點 URI 或 VPC 服務名稱正確無誤。	XksProxyUriUnreachableException 回應 CreateCustomKeyStore 和 UpdateCustomKeyStore 操作。 CustomKeyStoreInvalidStateException 回應 CreateKey 操作。 KMSInvalidStateException 回應密碼編譯操作。

前 5 個例外狀況圖衍生自 [XksProxyErrors](#) 指標。當您[檢視特定資料點](#)時，彈出式視窗會顯示 ExceptionName 維度值，以及在該資料點記錄的例外狀況次數。這五個清單項目從最常見的例外狀況到最不常見的例外狀況排序。

建議您使用 [XksProxyErrors](#) 指標來建立 CloudWatch 警示，以便在一分鐘內記錄超過五個不可重試的錯誤時提醒您，從而通知您存在潛在的組態問題。如需詳細資訊，請參閱[針對不可重試的錯誤建立警示](#)。

憑證到期天數

外部金鑰存放區代理端點 (XksProxyUriEndpoint) 的 TLS 憑證到期前的天數。使用此圖表來監控 TLS 憑證即將到期。

當憑證過期時，AWS KMS 無法與外部金鑰存放區代理通訊。在您續約憑證之前，外部金鑰存放區中受 KMS 金鑰保護的所有資料都無法存取。

憑證到期天數圖表衍生自 [XksProxyCertificateDaysToExpire](#) 指標。強烈建議使用此指標來建立 CloudWatch 警示，通知您即將到期。憑證到期可能會阻礙您存取加密資源。設定警示可讓組織有時間在憑證過期之前續約憑證。如需詳細資訊，請參閱[建立憑證過期的警示](#)。

連接和中斷連接外部金鑰存放區

新的外部金鑰存放區未連接。若要在外部金鑰存放區 AWS KMS keys 中建立和使用，您需要將外部金鑰存放區連接到其[外部金鑰存放區代理](#)。您可以隨時連接和中斷連接您的外部金鑰存放區，並且[檢視其連接狀態](#)。

當您的外部金鑰存放區中斷連線時，AWS KMS 無法與您的外部金鑰存放區代理通訊。因此，您可以檢視和管理外部金鑰存放區和其現有 KMS 金鑰。但是，您無法在外部金鑰存放區中建立 KMS 金鑰，或在密碼編譯操作中使用其 KMS 金鑰。您可能需要在某些時候中斷連接外部金鑰存放區，例如在編輯其屬性時，但需要進行相應的規劃。中斷連接金鑰存放區可能會中斷使用其 KMS 金鑰之 AWS 服務的操作。

您不需要連接您的外部金鑰存放區。您可以將外部金鑰存放區無限期保留在中斷連接狀態，並只在您需要使用它時連接它。不過，您可能希望定期測試連接，以驗證設定正確並且可連接。

當您中斷連接自訂金鑰存放區時，該金鑰存放區中的 KMS 金鑰會立即變成無法使用 (視最終一致性而定)。不過，使用受 KMS 金鑰保護之[資料金鑰](#)所加密的資源不會受影響，除非再次使用 KMS 金鑰 (例如解密資料金鑰)。此問題會影響 AWS 服務，其中許多服務會使用資料金鑰來保護您的資源。如需詳細資訊，請參閱[無法使用的 KMS 金鑰如何影響資料金鑰](#)。

Note

僅當金鑰存放區從未連接或明確中斷連接時，外部金鑰存放區才會處於 DISCONNECTED 狀態。CONNECTED 狀態並不表示外部金鑰存放區或其支援元件正在高效運作。如需外部金鑰存放區元件效能的相關資訊，請參閱每個外部金鑰存放區詳細資訊頁面之 Monitoring (監控) 區段中的圖表。如需詳細資訊，請參閱[監控外部金鑰存放區](#)。

您的外部金鑰管理器可能會提供其他方法來停止和重新啟動 AWS KMS 外部金鑰存放區與外部金鑰存放區代理之間的通訊，或外部金鑰存放區代理與外部金鑰管理器之間的通訊。如需詳細資訊，請參閱外部金鑰管理器文件。

主題

- [連線狀態](#)
- [連接外部金鑰存放區](#)
- [中斷連接外部金鑰存放區](#)

連線狀態

連接和中斷連接會變更自訂金鑰存放區的連接狀態。AWS CloudHSM 金鑰存放區和外部金鑰存放區的連線狀態值相同。

若要檢視自訂金鑰存放區的連線狀態，請使用 [DescribeCustomKeyStores](#) 操作或 AWS KMS 主控台。Connection state (連接狀態) 會顯示在每個自訂金鑰存放區資料表、每個自訂金鑰存放區詳細資料頁面的 General configuration (一般組態) 區段以及自訂金鑰存放區中 KMS 金鑰的 Cryptographic configuration (密碼編譯組態) 索引標籤中。如需詳細資訊，請參閱 [檢視 AWS CloudHSM 金鑰存放區](#) 和 [檢視外部金鑰存放區](#)。

自訂金鑰存放區可以有下列其中一個連接狀態：

- **CONNECTED**：自訂金鑰存放區已連接至其備份金鑰存放區。您可在自訂金鑰存放區中建立和使用 KMS 金鑰。

金鑰存放區的後端 AWS CloudHSM 金鑰存放區是其相關聯的 AWS CloudHSM 叢集。外部金鑰存放區的備份金鑰存放區是外部金鑰存放區代理及其支援的外部金鑰管理器。

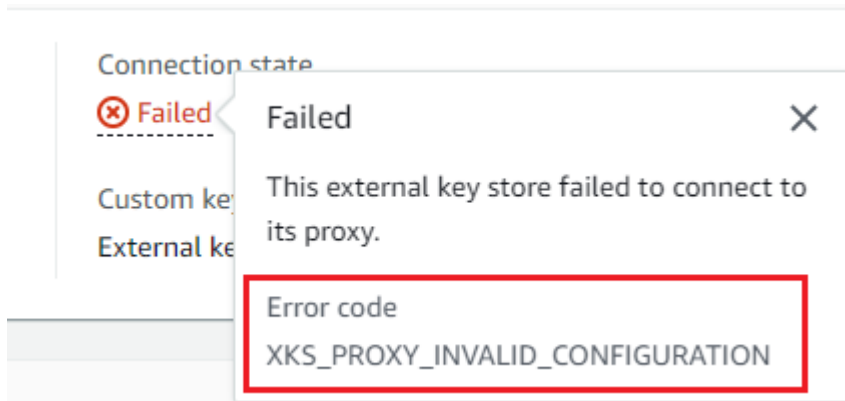
CONNECTED 狀態表示連接成功，且自訂金鑰存放區尚未有意中斷連接。這並不表示連接運作正常。如需有關與 AWS CloudHSM 金鑰存放區相關聯之 AWS CloudHSM 叢集狀態的資訊，請參閱 AWS CloudHSM 《使用者指南》中的[取得的 CloudWatch 指標 AWS CloudHSM](#)。如需有關外部金鑰存放區狀態和操作的相關資訊，請參閱每個外部金鑰存放區詳細資訊頁面的監控區段中的圖表。如需詳細資訊，請參閱[監控外部金鑰存放區](#)。

- **CONNECTING**：連接自訂金鑰存放區的程序正在進行中。這是暫時的狀態。
- **DISCONNECTED**：自訂金鑰存放區從未連接到其備份，或使用 AWS KMS 主控台或 [DisconnectCustomKeyStore](#) 操作刻意中斷連線。
- **DISCONNECTING**：中斷連接自訂金鑰存放區的程序正在進行中。這是暫時的狀態。
- **FAILED**：嘗試連接自訂金鑰存放區失敗。[DescribeCustomKeyStores](#) 回應中的 ConnectionErrorCode 指出該問題。

若要連接自訂金鑰存放區，其連接狀態必須為 DISCONNECTED。如果連接狀態為 FAILED，則請使用 ConnectionErrorCode 來識別並解決問題。請先中斷連接自訂金鑰存放區，然後再重試連接。如需連線失敗的協助，請參閱 [外部金鑰存放區連接錯誤](#)。如需有關回應連接錯誤代碼的說明，請參閱 [外部金鑰存放區的連接錯誤代碼](#)。

若要檢視連接錯誤代碼：

- 在 [DescribeCustomKeyStores](#) 回應中，檢視 ConnectionErrorCode 元素的值。只有當 ConnectionState 為 FAILED 時，此元素才會出現在 DescribeCustomKeyStores 回應中。
- 若要在 AWS KMS 主控台中檢視連線錯誤代碼，請在外部金鑰存放區的詳細資訊頁面上，並將滑鼠游標暫留在失敗值上。



連接外部金鑰存放區

外部金鑰存放區連接到其外部金鑰存放區代理時，您可以[在外部金鑰存放區中建立 KMS 金鑰](#)，並在[密碼編譯操作](#)中使用其現有的 KMS 金鑰。

將外部金鑰存放區連接至其外部金鑰存放區代理的程序會有所不同，這取決於外部金鑰存放區的連接。

- 使用[公有端點連接](#)來連接外部金鑰存放區時，AWS KMS 會將 [GetHealthStatus 請求](#)傳送至外部金鑰存放區代理，以驗證[代理 URI 端點](#)、[代理 URI 路徑](#)和[代理身分驗證憑證](#)。來自代理的成功回應可確認[代理 URI 端點](#)和[代理 URI 路徑](#)是正確且可存取的，而且代理驗證了使用外部金鑰存放區之[代理身分驗證憑證](#)所簽署的請求。
- 當您將具有 [VPC 端點服務的外部金鑰存放區](#)連線至其外部金鑰存放區代理時，會 AWS KMS 執行下列動作：
 - 確認在[代理 URI 端點](#)中指定之私有 DNS 名稱的網域[已經過驗證](#)。
 - 建立從 AWS KMS VPC 到 VPC 端點服務的介面端點。
 - 為代理 URI 端點中指定的私有 DNS 名稱建立私有託管區域
 - 將 [GetHealthStatus 請求](#)傳送至外部金鑰存放區代理。來自代理的成功回應可確認[代理 URI 端點](#)和[代理 URI 路徑](#)是正確且可存取的，而且代理驗證了使用外部金鑰存放區之[代理身分驗證憑證](#)所簽署的請求。

連接操作開始了連接自訂金鑰存放區的過程，但將外部金鑰存放區連接到其外部代理大約需要五分鐘。來自連接操作的成功回應並不表示外部金鑰存放區已連接。若要確認連線成功，請使用 AWS KMS 主控台或 [DescribeCustomKeyStores](#) 操作來檢視金鑰存放區外部的連線狀態。

當連線狀態為 FAILED，連線錯誤代碼會顯示在 AWS KMS 主控台中，並新增至 DescribeCustomKeyStore 回應。如需有關解譯連接錯誤代碼的說明，請參閱 [外部金鑰存放區的連接錯誤代碼](#)。

連接並重新連線至您的外部金鑰存放區

您可以在 AWS KMS 主控台或使用 [ConnectCustomKeyStore](#) 操作來連接或重新連接外部金鑰存放區。

使用 AWS KMS 主控台

您可以使用 AWS KMS 主控台將外部金鑰存放區連接到其外部金鑰存放區代理。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，依次選擇 Custom key stores (自訂金鑰存放區)、External key stores (外部金鑰存放區)。
4. 選擇您想要連接的外部金鑰存放區資料列。

如果外部金鑰存放區的連接狀態為 FAILED (失敗)，則在連接之前，必須[中斷連接外部金鑰存放區](#)。

5. 從 Key store actions (金鑰存放區動作) 選單中，選擇 Connect (連接)。

連接程序通常需要大約五分鐘才能完成。操作完成時，連接狀態會變更為 CONNECTED (已連接)。

如果連接狀態為 Failed (失敗)，則請將滑鼠游標移至連接狀態上，以查看連接錯誤代碼，其中會說明錯誤原因。如需有關回應連接錯誤代碼的說明，請參閱 [外部金鑰存放區的連接錯誤代碼](#)。若要連接具有 Failed (失敗) 連接狀態的外部金鑰存放區，您必須先[中斷連接自訂金鑰存放區](#)。

使用 AWS KMS API

若要連接已中斷連接的外部金鑰存放區，請使用 [ConnectCustomKeyStore](#) 操作。

在連接之前，外部金鑰存放區的連接狀態必須為 DISCONNECTED。如果當前連接狀態為 FAILED，則請[中斷連接外部金鑰存放區](#)，然後將其連接。

連接程序需要大約五分鐘才能完成。除非它快速失敗，否則 `ConnectCustomKeyStore` 會傳回 HTTP 200 回應和不帶屬性的 JSON 物件。不過，這個初始回應並不表示連接已成功。若要判斷外部金鑰存放區是否已連接，請參閱 [DescribeCustomKeyStores](#) 回應中的連接狀態。

本節中的範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

若要識別外部金鑰存放區，請使用其自訂金鑰存放區 ID。您可以在主控台的 Custom key stores (自訂金鑰存放區) 頁面或使用 [DescribeCustomKeyStores](#) 操作來找到 ID。執行此範例之前，請將範例 ID 以有效的 ID 取代。

```
$ aws kms connect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

`ConnectCustomKeyStore` 操作不會傳回其回應中的 `ConnectionState`。若要驗證已連接外部金鑰存放區，請使用 [DescribeCustomKeyStores](#) 操作。在預設情況下，此操作會傳回您帳戶和區域中的所有自訂金鑰存放區。但是，您可以使用 `CustomKeyStoreId` 或 `CustomKeyStoreName` 參數 (但不能同時使用) 來限制對特定自訂金鑰存放區的回應。`CONNECTED` 的 `ConnectionState` 值表示外部金鑰存放區已連接至其外部金鑰存放區代理。

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleXksVpc
{
  "CustomKeyStores": [
    {
      "CustomKeyStoreId": "cks-9876543210fedcba9",
      "CustomKeyStoreName": "ExampleXksVpc",
      "ConnectionState": "CONNECTED",
      "CreationDate": "2022-12-13T18:34:10.675000+00:00",
      "CustomKeyStoreType": "EXTERNAL_KEY_STORE",
      "XksProxyConfiguration": {
        "AccessKeyId": "ABCDE98765432EXAMPLE",
        "Connectivity": "VPC_ENDPOINT_SERVICE",
        "UriEndpoint": "https://example-proxy-uri-endpoint-vpc",
        "UriPath": "/example/prefix/kms/xks/v1",
        "VpcEndpointServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-example"
      }
    }
  ]
}
```

如果 `DescribeCustomKeyStores` 回應中的 `ConnectionState` 值為 `FAILED`，則 `ConnectionErrorCode` 元素會指出失敗的原因。

在下列範例中，XKS_VPC_ENDPOINT_SERVICE_NOT_FOUND的值ConnectionErrorCode表示 AWS KMS 找不到用於與外部金鑰存放區代理通訊的 VPC 端點服務。確認

XksProxyVpcEndpointServiceName 正確、AWS KMS 服務主體是 Amazon VPC 端點服務上允許的主體，而且 VPC 端點服務不需要接受連線請求。如需回應連接錯誤代碼的說明，請參閱 [外部金鑰存放區的連接錯誤代碼](#)。

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleXksVpc
{
  "CustomKeyStores": [
    {
      "CustomKeyStoreId": "cks-9876543210fedcba9",
      "CustomKeyStoreName": "ExampleXksVpc",
      "ConnectionState": "FAILED",
      "ConnectionErrorCode": "XKS_VPC_ENDPOINT_SERVICE_NOT_FOUND",
      "CreationDate": "2022-12-13T18:34:10.675000+00:00",
      "CustomKeyStoreType": "EXTERNAL_KEY_STORE",
      "XksProxyConfiguration": {
        "AccessKeyId": "ABCDE98765432EXAMPLE",
        "Connectivity": "VPC_ENDPOINT_SERVICE",
        "UriEndpoint": "https://example-proxy-uri-endpoint-vpc",
        "UriPath": "/example/prefix/kms/xks/v1",
        "VpcEndpointServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-example"
      }
    }
  ]
}
```

中斷連接外部金鑰存放區

當您中斷具有 [VPC 端點服務連接](#) 的外部金鑰存放區與其外部金鑰存放區代理的連接時，AWS KMS 會刪除其與 VPC 端點服務的介面端點，並移除其為支援連接而建立的網路基礎設施。具有公有端點連接的外部金鑰存放區不需要同等程序。此動作不會影響 VPC 端點服務或其任何支援元件，也不會影響外部金鑰存放區代理或任何外部元件。

當外部金鑰存放區中斷連線時，AWS KMS 不會將任何請求傳送至外部金鑰存放區代理。外部金鑰存放區的連接狀態為 DISCONNECTED。中斷連接的外部金鑰存放區中的 KMS 金鑰處於 [UNAVAILABLE 金鑰狀態](#) (除非其為 [待刪除](#))，這表示其無法用於密碼編譯操作。但是，您仍可以檢視和管理外部金鑰存放區和其現有 KMS 金鑰。

中斷連接狀態被設計為暫時且可還原的。您可以隨時重新連接外部金鑰存放區。通常，不需要重新設定。但是，如果相關聯的外部金鑰存放區代理的任何屬性在中斷連接時發生變更，例如其[代理身分驗證憑證](#)的輪換，則必須先[編輯外部金鑰存放區設定](#)，才能重新連接。

Note

當自訂金鑰存放區中斷連接時，所有在自訂金鑰存放區中建立 KMS 金鑰的嘗試，或在密碼編譯操作中使用現有 KMS 金鑰的嘗試，均會失敗。此動作可防止使用者存放和存取敏感資料。

若要更好地預估中斷連接您的外部金鑰存放區的效果，請在外部金鑰存放區中識別 KMS 金鑰，並判斷其過去的使用情形。

您可能中斷連接外部金鑰存放區的原因如下所示：

- 編輯其屬性。在連接外部金鑰存放區時，您可以編輯自訂金鑰存放區名稱、代理 URI 路徑和代理身分驗證憑證。但是，若要編輯代理連接類型、代理 URI 端點或 VPC 端點服務名稱，您必須先中斷連接外部金鑰存放區。如需詳細資訊，請參閱[編輯外部金鑰存放區屬性](#)。
- 停止與外部金鑰存放區代理之間的所有通訊。AWS KMS 您也可以停用端點或 VPC 端點服務，停止 AWS KMS 與代理之間的通訊。此外，您的外部金鑰存放區代理或金鑰管理軟體可能會提供額外的機制，AWS KMS 以防止與代理通訊，或防止代理存取您的外部金鑰管理器。
- 停用外部金鑰存放區中的所有 KMS 金鑰。您可以使用 AWS KMS 主控台或 [DisableKey](#) 操作，在外部金鑰存放區中[停用和重新啟用 KMS](#) 金鑰。這些操作會快速完成 (視最終一致性而定)，但一次只能對一個 KMS 金鑰進行。中斷連接外部金鑰存放區會將外部金鑰存放區中所有 KMS 金鑰的金鑰狀態變更為 Unavailable，這會防止在任何密碼編譯操作中對其進行使用。
- 為了修復失敗的連接嘗試。如果嘗試連接外部金鑰存放區失敗 (自訂金鑰存放區的連接狀態為 FAILED)，您必須先中斷連接外部金鑰存放區，之後再嘗試重新連接。

中斷連接外部金鑰存放區

您可以在 AWS KMS 主控台或使用 [DisconnectCustomKeyStore](#) 操作中斷連接外部金鑰存放區。

使用 AWS KMS 主控台

您可以使用 AWS KMS 主控台將外部金鑰存放區連接到其外部金鑰存放區代理。此程序約需 5 分鐘才能完成。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。

2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，依次選擇 Custom key stores (自訂金鑰存放區)、External key stores (外部金鑰存放區)。
4. 選擇您想要中斷連接之外部金鑰存放區的資料列。
5. 從 Key store actions (金鑰存放區動作) 選單中，選擇 Disconnect (中斷連接)。

操作完成時，連接狀態會從 DISCONNECTING (中斷連接中) 變更為 DISCONNECTED (已中斷連接)。如果操作失敗，會出現錯誤訊息，其中描述問題並提供如何修正的協助。如果您需要更多協助，請參閱[外部金鑰存放區連接錯誤](#)。

使用 AWS KMS API

若要中斷連接已連接的外部金鑰存放區，請使用 [DisconnectCustomKeyStore](#) 操作。如果操作成功，會 AWS KMS 傳回 HTTP 200 回應和沒有屬性的 JSON 物件。該程序需要大約五分鐘才能完成。若要查找外部金鑰存放區的連接狀態，請使用 [DescribeCustomKeyStores](#) 操作。

本節中的範例使用 [AWS Command Line Interface \(AWS CLI\)](#)，但您可以使用任何支援的程式設計語言。

此範例中斷連接具有 VPC 端點服務連接的外部金鑰存放區。執行此範例之前，請將範例自訂金鑰存放區 ID 取代為有效的 ID。

```
$ aws kms disconnect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

若要驗證已中斷連接外部金鑰存放區，請使用 [DescribeCustomKeyStores](#) 操作。在預設情況下，此操作會傳回您帳戶和區域中的所有自訂金鑰存放區。但是，您可以使用 CustomKeyId 和 CustomKeyName 參數 (但不能同時使用) 來限制對特定自訂金鑰存放區的回應。DISCONNECTED 的 ConnectionState 值表示此範例外部金鑰存放區不再連接至其外部金鑰存放區代理。

```
$ aws kms describe-custom-key-stores --custom-key-store-name ExampleXksVpc
{
  "CustomKeyStores": [
    {
      "CustomKeyId": "cks-9876543210fedcba9",
      "CustomKeyName": "ExampleXksVpc",
      "ConnectionState": "DISCONNECTED",
      "CreationDate": "2022-12-13T18:34:10.675000+00:00",
      "CustomKeyType": "EXTERNAL_KEY_STORE",
```

```
"XksProxyConfiguration": {
  "AccessKeyId": "ABCDE98765432EXAMPLE",
  "Connectivity": "VPC_ENDPOINT_SERVICE",
  "UriEndpoint": "https://example-proxy-uri-endpoint-vpc",
  "UriPath": "/example/prefix/kms/xks/v1",
  "VpcEndpointServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-example"
}
}
```

刪除外部金鑰存放區

當您刪除外部金鑰存放區時，會從 AWS KMS 中刪除有關外部金鑰存放區的所有中繼資料 AWS KMS，包括有關其外部金鑰存放區代理的資訊。此操作不會影響[外部金鑰存放區代理](#)、[外部金鑰管理器](#)、[外部金鑰](#)或您建立以支援外部金鑰存放區的任何 AWS 資源，例如 Amazon VPC 或 VPC 端點服務。

刪除外部金鑰存放區之前，您必須[刪除金鑰存放區中的所有 KMS 金鑰](#)，並從其外部金鑰存放區代理中[中斷連接金鑰存放區](#)。否則，嘗試刪除金鑰存放區會失敗。

刪除外部金鑰存放區是不可復原的，但您可以建立新的外部金鑰存放區，並將其與相同的外部金鑰存放區代理和外部金鑰管理器產生關聯。不過，您無法在外部金鑰存放區中重新建立對稱加密 KMS 金鑰，即使您可存取相同的外部金鑰材料。在每個 KMS 金鑰特有的對稱加密文字中 AWS KMS 包含中繼資料。此安全功能可以確保僅已加密資料的 KMS 金鑰可以解密。

請考慮將其中斷連接，而不是刪除外部金鑰存放區。當外部金鑰存放區中斷連線時，您可以管理外部金鑰存放區及其，AWS KMS keys 但無法在外部金鑰存放區中建立或使用 KMS 金鑰。您可以隨時重新連接外部金鑰存放區，並繼續使用其 KMS 金鑰來加密和解密資料。中斷連接的外部金鑰存放區代理或其無法使用的 KMS 金鑰無需支付任何費用。

您可以在 AWS KMS 主控台或使用 [DeleteCustomKeyStore](#) 操作來刪除外部金鑰存放區。

使用 AWS KMS 主控台

您可以使用 AWS KMS 主控台來刪除外部金鑰存放區。

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/kms> 的 AWS Key Management Service (AWS KMS) 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。

3. 在導覽窗格中，依次選擇 Custom key stores (自訂金鑰存放區)、External key stores (外部金鑰存放區)。
4. 尋找代表您要刪除之外部金鑰存放區的資料列。如果外部金鑰存放區的連接狀態不是 DISCONNECTED (已中斷連接)，則在刪除之前，必須[中斷連接外部金鑰存放區](#)。
5. 從 Key store actions (金鑰存放區動作) 選單中，選擇 Delete (刪除)。

當操作完成時，就會出現成功訊息，並且外部金鑰存放區將不再顯示在金鑰存放區清單中。如果操作失敗，就會出現錯誤訊息，其中描述問題並提供如何修正的協助。如果您需要更多協助，請參閱[外部金鑰存放區故障診斷](#)。

使用 AWS KMS API

若要刪除外部金鑰存放區，請使用 [DeleteCustomKeyStore](#) 操作。如果操作成功，會 AWS KMS 傳回 HTTP 200 回應和沒有屬性的 JSON 物件。

若要開始，請中斷連接外部金鑰存放區。執行此命令之前，請將範例自訂金鑰存放區 ID 以有效的 ID 取代。

```
$ aws kms disconnect-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

外部金鑰存放區中斷連接之後，您就可以使用 [DeleteCustomKeyStore](#) 操作將其刪除。

```
$ aws kms delete-custom-key-store --custom-key-store-id cks-1234567890abcdef0
```

若要確認已刪除外部金鑰存放區，請使用 [DescribeCustomKeyStores](#) 操作。

```
$ aws kms describe-custom-key-stores

{
  "CustomKeyStores": []
}
```

如果您指定不再存在的自訂金鑰存放區名稱或 ID，會 AWS KMS 傳回 CustomKeyStoreNotFoundException 例外狀況。

```
$ aws kms describe-custom-key-stores --custom-key-store-id cks-1234567890abcdef0
```

```
An error occurred (CustomKeyStoreNotFoundException) when calling the
DescribeCustomKeyStore operation:
```


外部金鑰存放區故障診斷

大多數外部金鑰存放區問題的解決方法，是以每次例外狀況時 AWS KMS 顯示的錯誤訊息表示，或是透過嘗試[將外部金鑰存放區連線至](#)其外部金鑰存放區代理失敗時 AWS KMS 傳回的[連線錯誤代碼](#)表示。但是，有些問題有點複雜。

診斷外部金鑰存放區的問題時，請先找出原因。這將縮小補救措施的範圍，並使您更有效地進行故障排除。

- AWS KMS — 問題可能位於其中 AWS KMS，例如[外部金鑰存放區組態](#)中的值不正確。
- 外部 — 問題可能源自外部 AWS KMS，包括外部金鑰存放區代理、外部金鑰管理器、外部金鑰或 VPC 端點服務的組態或操作問題。
- 聯網 — 這可能是連線或聯網的問題，例如代理端點、連接埠、IP 堆疊或私有 DNS 名稱或網域的問題。

Note

當外部金鑰存放區上的管理操作失敗時，其會產生數個不同的例外狀況。但是 AWS KMS，密碼編譯操作 `KMSInvalidStateException` 會傳回與外部金鑰存放區外部組態或連線狀態相關的所有失敗。若要識別問題，請使用隨附的錯誤訊息文字。

連接程序完成之前，[ConnectCustomKeyStore](#) 操作會迅速成功。若要確定連接程序是否成功，請檢視外部金鑰存放區的[連接狀態](#)。如果連接程序失敗，則 AWS KMS 會傳回[連接錯誤代碼](#)，說明原因並建議解決方法。

主題

- [外部金鑰存放區的故障診斷工具](#)
- [組態錯誤](#)
- [外部金鑰存放區連接錯誤](#)
- [延遲和逾時錯誤](#)
- [身分驗證憑證錯誤](#)
- [金鑰狀態錯誤](#)
- [解密錯誤](#)
- [外部金鑰錯誤](#)

- [代理問題](#)
- [代理授權問題](#)

外部金鑰存放區的故障診斷工具

AWS KMS 提供數種工具，協助您識別和解決外部金鑰存放區及其金鑰的問題。將這些工具與外部金鑰存放區代理和外部金鑰管理器隨附的工具搭配使用。

Note

您的外部金鑰存放區代理和外部金鑰管理器可能會提供更簡單的方法來建立和維護外部金鑰存放區及其 KMS 金鑰。如需詳細資訊，請參閱外部工具的文件。

AWS KMS 例外狀況和錯誤訊息

AWS KMS 提供有關其遇到的任何問題的詳細錯誤訊息。您可以在 [AWS Key Management Service API 參考](#) 和 AWS SDKs 中找到有關 AWS KMS 例外狀況的其他資訊。即使您使用 AWS KMS 主控台，也可能會發現這些參考很有幫助。例如，請參閱 CreateCustomKeyStores 操作的 [錯誤清單](#)。

為了最佳化外部金鑰存放區代理的效能，AWS KMS 會根據代理的可靠性，在 5 分鐘的指定彙總期間內傳回例外狀況。如果發生 500 內部伺服器錯誤、503 服務無法使用或連線逾時，具有高可靠性的代理會傳回 KMSInternalException 並觸發自動重試，以確保請求最終成功。不過，可靠性低的代理會傳回 KMSInvalidStateException。如需詳細資訊，請參閱 [監控外部金鑰存放區](#)。

如果問題出現在不同的 AWS 服務中，例如當您使用外部金鑰存放區中的 KMS 金鑰來保護其他服務中的資源時 AWS，該 AWS 服務可能會提供其他資訊，以協助您識別問題。如果 AWS 服務不提供訊息，您可以在記錄 KMS 金鑰使用的 [CloudTrail 日誌](#) 中檢視錯誤訊息。

[CloudTrail 日誌](#)

每個 AWS KMS API 操作，包括 AWS KMS 主控台內的動作，都會記錄在 AWS CloudTrail log 中。會 AWS KMS 記錄日誌項目，以取得成功和失敗的操作。對於失敗的操作，日誌項目會包含 AWS KMS 例外狀況名稱 (errorCode) 和錯誤訊息 (errorMessage)。您可以使用此資訊來協助您識別和解決錯誤。如需範例，請參閱 [使用外部金鑰存放區中的 KMS 金鑰進行解密失敗](#)。

日誌項目也包含請求 ID。如果請求到達外部金鑰存放區代理，則您可以使用日誌項目中的請求 ID，在代理日誌中尋找相應的請求 (如果代理提供的話)。

CloudWatch 指標

AWS KMS 會記錄有關外部金鑰存放區操作和效能的詳細 Amazon CloudWatch 指標，包括延遲、調節、代理錯誤、外部金鑰管理器狀態、TLS 憑證過期前的天數，以及代理身分驗證憑證的回報存留期。您可以使用這些指標為外部金鑰存放區和 CloudWatch 警示的操作開發資料模型，在即將發生的問題出現之前提醒您。

Important

AWS KMS 建議您建立 CloudWatch 警示來監控外部金鑰存放區指標。這些警報會在問題發生之前提醒您早期跡象。

監控圖表

AWS KMS 會在 AWS KMS 主控台中每個外部金鑰存放區的詳細資訊頁面上顯示外部金鑰存放區 CloudWatch 指標的圖形。您可以使用圖表中的資料來協助尋找錯誤來源、偵測即將發生的問題、建立基準以及強化 CloudWatch 警示閾值。如需有關解釋監控圖表及使用其資料的詳細資訊，請參閱 [監控外部金鑰存放區](#)。

顯示外部金鑰存放區和 KMS 金鑰

AWS KMS 會在 AWS KMS 主控台的外部金鑰存放區中，以及回應 [DescribeCustomKeyStores](#) 和 [DescribeKey](#) 操作時，顯示外部金鑰存放區和 KMS 金鑰的詳細資訊。這些顯示包括外部金鑰存放區和 KMS 金鑰的特殊欄位，以及可用於進行故障診斷的資訊，例如外部金鑰存放區的[連接狀態](#)，以及與 KMS 金鑰相關聯的外部金鑰 ID。如需詳細資訊，請參閱[檢視外部金鑰存放區](#)。

XKS 代理測試用戶端

AWS KMS 提供開放原始碼測試用戶端，可驗證外部金鑰存放區代理是否符合[AWS KMS 外部金鑰存放區代理 API 規格](#)。您可以使用此測試用戶端來識別並解決外部金鑰存放區代理的問題。

組態錯誤

建立外部金鑰存放區時，您可以指定組成外部金鑰存放區組態的屬性值，例如[代理身分驗證憑證](#)、[代理 URI 端點](#)、[代理 URI 路徑](#)以及 [VPC 端點服務名稱](#)。當 AWS KMS 偵測到屬性值中的錯誤時，操作會失敗，並傳回指出錯誤值的錯誤。

許多組態問題可以透過修正不正確的值來解決。您可以修正無效的代理 URI 路徑或代理身分驗證憑證，而無需中斷連接外部金鑰存放區。如需這些值的定義，包括唯一性要求，請參閱 [備妥先決條件](#)。如需有關更新這些值的指示，請參閱 [編輯外部金鑰存放區屬性](#)。

若要避免代理 URI 路徑和代理身分驗證憑證值發生錯誤，在建立或更新外部金鑰存放區時，請將[代理組態檔案](#)上傳至 AWS KMS 主控台。這是一個基於 JSON 的檔案，其中包含代理 URI 路徑和代理身分驗證憑證值，由外部金鑰存放區代理或外部金鑰管理器提供。您無法將代理組態檔案與 AWS KMS API 操作搭配使用，但您可以使用檔案中的值來協助您為 API 請求提供符合代理中值的參數值。

一般組態錯誤

例外狀況：CustomKeyStoreInvalidStateException
(CreateKey)、KMSInvalidStateException (密碼編譯操作)、XksProxyInvalidConfigurationException (管理操作，CreateKey 除外)

連接錯誤代

碼：XKS_PROXY_INVALID_CONFIGURATION、XKS_PROXY_INVALID_TLS_CONFIGURATION

對於具有[公有端點連線](#)的外部金鑰存放區，會在您建立和更新外部金鑰存放區時 AWS KMS 測試屬性值。對於具有[VPC 端點服務連接](#)的外部金鑰存放區，AWS KMS 會在連接和更新外部金鑰存放區時測試屬性值。

Note

即使嘗試將外部金鑰存放區連接至其外部金鑰存放區代理失敗，非同步的 ConnectCustomKeyStore 操作仍可能成功。在這種情況下，沒有例外狀況，但是外部金鑰存放區的連接狀態為 Failed (失敗)，連接錯誤代碼會說明錯誤訊息。如需詳細資訊，請參閱[外部金鑰存放區連接錯誤](#)。

如果 AWS KMS 偵測到屬性值中的錯誤，操作會失敗，並 XksProxyInvalidConfigurationException 傳回下列其中一個錯誤訊息。

由於 URI 路徑無效，外部金鑰存放區代理拒絕請求。驗證外部金鑰存放區的 URI 路徑，並在必要時更新。

- [代理 URI 路徑](#)是 AWS KMS 請求代理 APIs 的基本路徑。如果此路徑不正確，則對代理的所有請求都會失敗。若要檢視外部金鑰存放區的[當前代理 URI 路徑](#)，請使用 AWS KMS 主控台或 DescribeCustomKeyStores 操作。若要尋找正確的代理 URI 路徑，請參閱您的外部金鑰存放區代理文件。如需有關修正代理 URI 路徑值的說明，請參閱[編輯外部金鑰存放區屬性](#)。
- 外部金鑰存放區代理的代理 URI 路徑可能會隨著外部金鑰存放區代理或外部金鑰管理器的更新而變更。如需有關這些變更的資訊，請參閱外部金鑰存放區代理或外部金鑰管理器的文件。

XKS_PROXY_INVALID_TLS_CONFIGURATION

AWS KMS 無法建立與外部金鑰存放區代理的 TLS 連接。驗證 TLS 組態，包括其憑證。

- 所有外部金鑰存放區代理都需要 TLS 憑證。TLS 憑證必須由外部金鑰存放區支援的公有憑證授權機構核發。如需支援的 CA 清單，請參閱「AWS KMS 外部金鑰存放區代理 API 規格」中的[受信任憑證授權機構](#)。
- 若為公有端點連接，TLS 憑證上的主體通用名稱 (CN) 必須與外部金鑰存放區代理之[代理 URI 端點](#)中的網域名稱相符。例如，如果公有端點為 `https://myproxy.xks.example.com`，則 TLS 憑證上的通用名稱必須為 `myproxy.xks.example.com` 或 `*.xks.example.com`。
- 若為 VPC 端點服務連接，TLS 憑證上的主體通用名稱 (CN) 必須與[VPC 端點服務](#)的私有 DNS 名稱相符。例如，如果私有 DNS 名為 `myproxy-private.xks.example.com`，則 TLS 憑證上的通用名稱必須為 `myproxy-private.xks.example.com` 或 `*.xks.example.com`。
- TLS 憑證不能過期。若要取得 TLS 憑證的到期日，請使用 SSL 工具，例如 [OpenSSL](#)。若要監控與外部金鑰存放區關聯的 TLS 憑證的到期日，請使用 [XksProxyCertificateDaysToExpire](#) CloudWatch 指標。TLS 憑證過期日期的天數也會顯示在 AWS KMS 主控台的[監控區段](#)中。
- 如果您使用的是[公有端點連接](#)，則請使用 SSL 測試工具來測試您的 SSL 組態。TLS 連接錯誤可能是因為不正確的憑證鏈結所導致。

VPC 端點服務連接組態錯誤

例外狀況

況：XksProxyVpcEndpointServiceNotFoundException、XksProxyVpcEndpointServiceInvalid

除了一般連線問題之外，使用 VPC 端點服務連線建立、連線或更新外部金鑰存放區時，您可能會遇到下列問題。在[建立](#)、[連線](#)和[更新](#)外部金鑰存放區時，使用 VPC 端點服務連線 AWS KMS 測試外部金鑰存放區的屬性值。當管理操作因組態錯誤而失敗時，會產生下列例外狀況：

XksProxyVpcEndpointServiceNotFoundException

原因可能為下列之一：

- 不正確的 VPC 端點服務名稱。請確認外部金鑰存放區的 VPC 端點服務名稱正確且符合外部金鑰存放區的代理 URI 端點值。若要查找 VPC 端點服務名稱，請使用 [Amazon VPC 主控台](#)或 [DescribeVpcEndpointServices](#) 操作。若要尋找現有外部金鑰存放區的 VPC 端點服務名稱和代理

URI 端點，請使用 AWS KMS 主控台或 [DescribeCustomKeyStores](#) 操作。如需詳細資訊，請參閱[檢視外部金鑰存放區](#)。

- VPC 端點服務可能位於與外部金鑰存放區 AWS 區域不同的中。請確認 VPC 端點服務與外部金鑰存放區處於相同區域中。(區域名稱的外部名稱 (例如 us-east-1) 是 VPC 端點服務名稱的一部分，例如 com.amazonaws.vpce.us-east-1.vpce-svc-example)。如需外部金鑰存放區之 VPC 端點服務的要求清單，請參閱 [VPC 端點服務](#)。您無法將 VPC 端點服務或外部金鑰存放區移至不同區域。不過，您可以在與 VPC 端點服務相同的區域中建立新的外部金鑰存放區。如需詳細資訊，請參閱 [設定 VPC 端點服務連線](#) 和 [建立外部金鑰存放區](#)。
- AWS KMS 不是 VPC 端點服務的允許主體。VPC 端點服務的 Allow principals (允許主體) 清單必須包含 cks.kms.<region>.amazonaws.com 值，例如 cks.kms.eu-west-3.amazonaws.com。如需有關新增此值的指示，請參閱《AWS PrivateLink 指南》中的[管理許可](#)。

XksProxyVpcEndpointServiceInvalidConfigurationException

當 VPC 端點服務無法滿足下列其中一項要求時，就會發生此錯誤：

- VPC 需要至少兩個私有子網路，每個處於不同的可用區域。如需有關將子網路新增至 VPC 的說明，請參閱《Amazon VPC 使用者指南》中的[在 VPC 中建立子網路](#)。
- 您的 [VPC 端點服務類型](#) 必須使用網路負載平衡器，而非閘道負載平衡器。
- VPC 端點服務不要求接受 (Acceptance required (要求接受) 必須為 False)。如果需要手動接受每個連線請求，AWS KMS 則無法使用 VPC 端點服務連線到外部金鑰存放區代理。如需詳細資訊，請參閱《AWS PrivateLink 指南》中的[接受或拒絕連接請求](#)。
- VPC 端點服務必須具有私有 DNS 名稱，它是公有網域的子網域。例如，如果私有 DNS 名為 https://myproxy-private.xks.example.com，則 xks.example.com 或 example.com 網域必須具有公有 DNS 伺服器。若要檢視或變更 VPC 端點服務的私有 DNS 名稱，請參閱《AWS PrivateLink 指南》中的[管理 VPC 端點服務的 DNS 名稱](#)。
- 私有 DNS 名稱網域的 Domain verification status (網域驗證狀態) 必須為 verified。若要檢視和更新私有 DNS 名稱網域的驗證狀態，請參閱 [步驟 5：驗證您的私有 DNS 名稱網域](#)。新增必要的文字記錄後，可能需要幾分鐘的時間才會顯示更新的驗證狀態。

Note

只有當私有 DNS 網域是公有網域的子網域時，才能驗證它。否則，即使新增所需的 TXT 記錄後，私有 DNS 網域的驗證狀態也不會變更。

- 確保 AWS KMS 和外部金鑰存放區代理之間的任何防火牆允許進出 Proxy 上連接埠 443 的流量。透過 IPv4 在連接埠 443 上 AWS KMS 通訊。此值不可設定。
- VPC 端點服務的私有 DNS 名稱必須與外部金鑰存放區的[代理 URI 端點](#)值相符。對於具有 VPC 端點服務連接的外部金鑰存放區，代理 URI 端點必須為 `https://`，後面為 VPC 端點服務的私有 DNS 名稱。若要檢視代理 URI 端點值，請參閱 [檢視外部金鑰存放區](#)。若要變更代理 URI 端點值，請參閱 [編輯外部金鑰存放區屬性](#)。

外部金鑰存放區連接錯誤

[將外部金鑰存放區連接至](#)其外部金鑰存放區代理的程序大約需要五分鐘才能完成。除非其快速失敗，否則 `ConnectCustomKeyStore` 操作會傳回 HTTP 200 回應和不帶屬性的 JSON 物件。不過，這個初始回應並不表示連接已成功。若要判斷外部金鑰存放區是否已連接，請參閱其[連接狀態](#)。如果連線失敗，外部金鑰存放區的連線狀態會變更為 `FAILED` 並 AWS KMS 傳回[連線錯誤代碼](#)，說明失敗的原因。

Note

如果自訂金鑰存放區的狀態為 `FAILED`，在嘗試重新連接之前，您必須中斷連接自訂金鑰存放區。您無法連接具有 `FAILED` 連接狀態的自訂金鑰存放區。

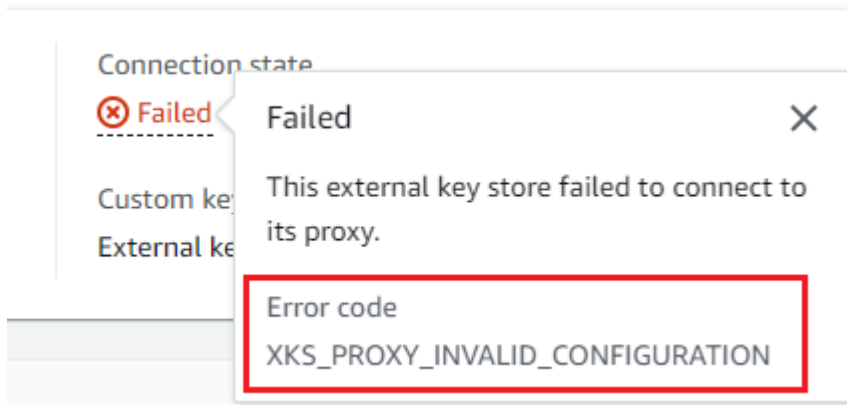
若要檢視外部金鑰存放區的連接狀態：

- 在 [DescribeCustomKeyStores](#) 回應中，檢視 `ConnectionState` 元素的值。
- 在 AWS KMS 主控台中，連線狀態會出現在外部金鑰存放區資料表中。此外，在每個外部金鑰存放區的詳細資訊頁面上，`Connection state` (連接狀態) 會顯示在 `General configuration` (一般組態) 區段中。

當連接狀態為 `FAILED` 時，連接錯誤代碼有助於解釋錯誤。

若要檢視連接錯誤代碼：

- 在 [DescribeCustomKeyStores](#) 回應中，檢視 `ConnectionErrorCode` 元素的值。只有當 `ConnectionState` 為 `FAILED` 時，此元素才會出現在 `DescribeCustomKeyStores` 回應中。
- 若要在 AWS KMS 主控台中檢視連線錯誤代碼，請在外部金鑰存放區的詳細資訊頁面上，並將滑鼠游標暫留在失敗值上。



外部金鑰存放區的連接錯誤代碼

下列連接錯誤代碼適用於外部金鑰存放區

INTERNAL_ERROR

AWS KMS 由於內部錯誤，無法完成請求。重試 請求。若為 `ConnectCustomKeyStore` 請求，請先中斷連接自訂金鑰存放區，再重試連接。

INVALID_CREDENTIALS

在指定的外部金鑰存放區代理上，一個或兩個 `XksProxyAuthenticationCredential` 值無效。

NETWORK_ERRORS

網路錯誤 AWS KMS 導致無法將自訂金鑰存放區連接至其後端金鑰存放區。

XKS_PROXY_ACCESS_DENIED

AWS KMS 請求會被拒絕存取外部金鑰存放區代理。如果外部金鑰存放區代理有授權規則，請確認其允許 AWS KMS 代表您與代理通訊。

XKS_PROXY_INVALID_CONFIGURATION

組態錯誤導致外部金鑰存放區無法連接到其代理。驗證 `XksProxyUriPath` 的值。

XKS_PROXY_INVALID_RESPONSE

AWS KMS 無法解譯來自外部金鑰存放區代理的回應。如果您重複看到此連接錯誤代碼，則請通知您的外部金鑰存放區代理廠商。

XKS_PROXY_INVALID_TLS_CONFIGURATION

AWS KMS 無法連線至外部金鑰存放區代理，因為 TLS 組態無效。確認外部金鑰存放區代理支援 TLS 1.2 或 1.3。此外，請確認 TLS 憑證尚未過期，與 XksProxyUriEndpoint 值中的主機名稱相符，並且由[受信任憑證授權機構](#)清單中所包含的受信任憑證授權機構簽署。

XKS_PROXY_NOT_REACHABLE

AWS KMS 無法與您的外部金鑰存放區代理通訊。確認 XksProxyUriEndpoint 和 XksProxyUriPath 正確無誤。使用外部金鑰存放區代理的工具來驗證代理處於作用中狀態且可在其網路上使用。此外，請確認您的外部金鑰管理器執行個體運作正常。如果代理報告所有外部金鑰管理器執行個體都無法使用，則連接嘗試失敗，並顯示此連接錯誤代碼。

XKS_PROXY_TIMED_OUT

AWS KMS 可以連線至外部金鑰存放區代理，但代理在分配 AWS KMS 的時間內不會回應。如果您重複看到此連接錯誤代碼，則請通知您的外部金鑰存放區代理廠商。

XKS_VPC_ENDPOINT_SERVICE_INVALID_CONFIGURATION

Amazon VPC 端點服務組態不符合 AWS KMS 外部金鑰存放區的要求。

- VPC 端點服務必須是呼叫者 AWS 帳戶中的介面端點的端點服務。
- 其必須擁有已連接到至少兩個子網路的網路負載平衡器 (NLB)，每個位於不同的可用區域中。
- Allow principals 清單必須包含區域的 AWS KMS 服務主體，cks.kms.<region>.amazonaws.com 例如 cks.kms.us-east-1.amazonaws.com。
- 其不得要求[接受](#)連接請求。
- 其必須擁有私有 DNS 名稱。具有 VPC_ENDPOINT_SERVICE 連接的外部金鑰存放區的私有 DNS 名稱在其 AWS 區域中必須唯一。
- 私有 DNS 名稱網域的[驗證狀態](#)必須為 verified。
- [TLS 憑證](#)會指定可連接端點的私有 DNS 主機名稱。

XKS_VPC_ENDPOINT_SERVICE_NOT_FOUND

AWS KMS 找不到用於與外部金鑰存放區代理通訊的 VPC 端點服務。確認 XksProxyVpcEndpointServiceName 正確無誤，且 AWS KMS 服務主體在 Amazon VPC 端點服務上具有服務消費者許可。

延遲和逾時錯誤

例外狀況：CustomKeyStoreInvalidStateException
(CreateKey)、KMSInvalidStateException (密碼編譯操作)、XksProxyUriUnreachableException (管理操作)

[連接錯誤代碼](#)：XKS_PROXY_NOT_REACHABLE、XKS_PROXY_TIMED_OUT

當 AWS KMS 無法在 250 毫秒的逾時間隔內聯絡代理時，它會傳回例外狀況。CreateCustomKeyStore 並 UpdateCustomKeyStore 傳回 XksProxyUriUnreachableException。密碼編譯操作會傳回標準，KMSInvalidStateException 並顯示描述問題的錯誤訊息。如果 ConnectCustomKeyStore 失敗，會 AWS KMS 傳回描述問題的[連線錯誤代碼](#)。

逾時錯誤可能是暫時性的問題，可透過重試請求來解決。如果問題依然存在，請確認外部金鑰存放區代理處於作用中狀態且已連接至網路，並且其代理 URI 端點、代理 URI 路徑以及 VPC 端點服務名稱 (如果有的話) 在外部金鑰存放區中正確無誤。此外，確認您的外部金鑰管理器靠近 AWS 區域 外部金鑰存放區的。如果需要更新任何這些值，請參閱 [編輯外部金鑰存放區屬性](#)。

若要追蹤延遲模式，請使用 AWS KMS 主控台之 [Monitoring \(監控\) 區段](#) 中的 [XksProxyLatency](#) CloudWatch 指標和 Average latency (平均延遲) 圖表 (基於該指標)。您的外部金鑰存放區代理也可能會產生追蹤延遲和逾時的日誌和指標。

XksProxyUriUnreachableException

AWS KMS 無法與外部金鑰存放區代理通訊。這可能是暫時性的網路問題。如果您重複看到此錯誤，請確認外部金鑰存放區代理處於作用中狀態且已連接至網路，並且其端點 URI 在外部金鑰存放區中正確無誤。

- 外部金鑰存放區代理未在 250 毫秒逾時間隔內回應 AWS KMS 代理 API 請求。這可能表示代理發生暫時性的網路問題或者操作或效能問題。如果重試無法解決問題，請通知您的外部金鑰存放區代理管理員。

延遲和逾時錯誤通常顯示為連接失敗。當 [ConnectCustomKeyStore](#) 操作失敗時，外部金鑰存放區的連線狀態會變更為 `FAILED` 並 AWS KMS 傳回說明錯誤的連線錯誤代碼。如需連接錯誤代碼和解決錯誤的建議清單，請參閱 [外部金鑰存放區的連接錯誤代碼](#)。All custom key stores (所有自訂金鑰存放區) 和 External key stores (外部金鑰存放區) 的連接代碼清單適用於外部金鑰存放區。下列連接錯誤與延遲和逾時有關。

XKS_PROXY_NOT_REACHABLE

-或-

CustomKeyStoreInvalidStateException , KMSInvalidStateException ,
XksProxyUriUnreachableException

AWS KMS 無法與外部金鑰存放區代理通訊。請確認外部金鑰存放區代理處於作用中狀態且已連接至網路，以及外部金鑰存放區中的 URI 路徑和端點 URI 或 VPC 服務名稱正確無誤。

此錯誤可能發生的原因如下：

- 外部金鑰存放區代理處於非作用中狀態，或未連接至網路。
- 外部金鑰存放區組態中的[代理 URI 端點](#)、[代理 URI 路徑](#)或 [VPC 端點服務名稱](#) (如果適用) 值中有錯誤。若要檢視外部金鑰存放區組態，請使用 [DescribeCustomKeyStores](#) 操作，或在 [主控台中檢視外部金鑰存放區的詳細資訊頁面](#)。AWS KMS
- AWS KMS 與外部金鑰存放區 proxy 之間的網路路徑上可能發生網路組態錯誤，例如連接埠錯誤。透過 IPv4 與連接埠 443 上的外部金鑰存放區代理 AWS KMS 通訊。此值不可設定。
- 當外部金鑰存放區代理報告 (在 [GetHealthStatus](#) 回應中) 所有外部金鑰管理器執行個體都為 UNAVAILABLE 時，[ConnectCustomKeyStore](#) 操作會失敗，並顯示值為 XKS_PROXY_NOT_REACHABLE 的 ConnectionErrorCode。如需說明，請參閱外部金鑰管理器文件。
- 此錯誤可能是因為外部金鑰管理器與 AWS 區域 具有外部金鑰存放區的 之間有很長的實體距離。AWS 區域 與外部金鑰管理器之間的 ping 延遲 (網路往返時間 (RTT)) 不應超過 35 毫秒。您可能必須在 AWS 區域 更接近外部金鑰管理器的 中建立外部金鑰存放區，或將外部金鑰管理器移至更接近的資料中心 AWS 區域。

XKS_PROXY_TIMED_OUT

-或-

CustomKeyStoreInvalidStateException , KMSInvalidStateException ,
XksProxyUriUnreachableException

AWS KMS 拒絕請求，因為外部金鑰存放區代理沒有及時回應。重試 請求。如果您重複看到此錯誤，請向外部金鑰存放區代理管理員報告。

此錯誤可能發生的原因如下：

- 此錯誤可能是由於外部金鑰管理器與外部金鑰存放區代理之間有很長的實際距離。如果可能，請將外部金鑰存放區代理移至更接近外部金鑰管理器的位置。
- 當代理無法處理來自的請求量和頻率時，可能會發生逾時錯誤 AWS KMS。如果 CloudWatch 指標指示持續存在問題，則請通知您的外部金鑰存放區代理管理員。
- 當外部金鑰管理器與外部金鑰存放區的 Amazon VPC 之間的連接未正常運作時，可能會發生逾時錯誤。如果您使用的是 AWS Direct Connect，請確認您的 VPC 和外部金鑰管理器可以有效通訊。如需解決任何問題的協助，請參閱 Direct Connect 《使用者指南》中的[故障診斷 AWS Direct Connect](#)。

XKS_PROXY_TIMED_OUT

-或-

CustomKeyStoreInvalidStateException, KMSInvalidStateException, XksProxyUriUnreachableException

外部金鑰存放區代理沒有在規定時間內回應請求。重試 請求。如果您重複看到此錯誤，請向外部金鑰存放區代理管理員報告。

- 此錯誤可能是由於外部金鑰管理器與外部金鑰存放區代理之間有很長的實際距離。如果可能，請將外部金鑰存放區代理移至更接近外部金鑰管理器的位置。

身分驗證憑證錯誤

例外狀況：CustomKeyStoreInvalidStateException

(CreateKey)、KMSInvalidStateException (密碼編譯操

作)、XksProxyIncorrectAuthenticationCredentialException (CreateKey 以外的管理操作)

您可以在外部金鑰存放區代理 AWS KMS 上建立和維護的身分驗證憑證。然後在建立外部金鑰存放區時 AWS KMS 告知登入資料值。若要變更身分驗證憑證，請在外部金鑰存放區代理上進行變更。然後，更新外部金鑰存放區的[憑證](#)。如果代理輪換憑證，則必須更新外部金鑰存放區的[憑證](#)。

如果外部金鑰存放區代理無法驗證使用外部金鑰存放區的[代理身分驗證憑證](#)簽署的請求，則效果取決於請求：

- CreateCustomKeyStore 和 UpdateCustomKeyStore 因 XksProxyIncorrectAuthenticationCredentialException 失敗。
- ConnectCustomKeyStore 成功，但連接失敗。連接狀態為 FAILED，連接錯誤代碼為 INVALID_CREDENTIALS。如需詳細資訊，請參閱[外部金鑰存放區連接錯誤](#)。
- 密碼編譯操作 KMSInvalidStateException 會傳回外部金鑰存放區中的所有外部組態錯誤和連線狀態錯誤。隨附的錯誤訊息描述了問題。

外部金鑰存放區代理拒絕了請求，因為它無法對 AWS KMS 進行身分驗證。驗證外部金鑰存放區的憑證，並在必要時更新。

此錯誤可能發生的原因如下：

- 外部金鑰存放區的存取金鑰 ID 或私密存取金鑰與外部金鑰存放區代理上建立的值不符。

若要修正此錯誤，請更新外部金鑰存放區的[代理身分驗證憑證](#)。您可以在不中斷外部金鑰存放區連接的情況下進行此變更。

- AWS KMS 與外部金鑰存放區代理之間的反向代理可能會以使 SigV4 簽章失效的方式操作 HTTP 標頭。若要修正此錯誤，請通知代理管理員。

金鑰狀態錯誤

例外狀況：KMSInvalidStateException

KMSInvalidStateException 用於自訂金鑰存放區中 KMS 金鑰的兩種不同用途。

- 當管理操作 (例如 CancelKeyDeletion) 失敗並傳回此例外狀況時，表示 KMS 金鑰的[金鑰狀態](#)與操作不相容。
- 當自訂金鑰存放區中 KMS 金鑰的[密碼編譯操作](#)因 KMSInvalidStateException 失敗時，可能表示 KMS 金鑰的金鑰狀態有問題。但是 AWS KMS，密碼編譯操作 KMSInvalidStateException 會傳回外部金鑰存放區中的所有外部組態錯誤和連線狀態錯誤。若要識別問題，請使用例外狀況隨附的錯誤訊息。

若要尋找 AWS KMS API 操作所需的金鑰狀態，請參閱[金鑰的 AWS KMS 金鑰狀態](#)。若要尋找 KMS 金鑰的金鑰狀態，請在 Customer managed keys (客戶受管金鑰) 頁面上，檢視 KMS 金鑰的 Status

(狀態) 欄位。或使用 [DescribeKey](#) 操作並檢視回應中的 KeyState 元素。如需詳細資訊，請參閱[識別和檢視金鑰](#)。

 Note

外部金鑰存放區中 KMS 金鑰的金鑰狀態不會指示與其關聯的[外部金鑰](#)的任何狀態資訊。如需有關外部金鑰狀態的資訊，請使用外部金鑰管理器和外部金鑰存放區代理工具。CustomKeyStoreInvalidStateException 指的是外部金鑰存放區的[連接狀態](#)，而不是 KMS 金鑰的[金鑰狀態](#)。

自訂存放區中 KMS 金鑰的密碼編譯操作可能會失敗，因為 KMS 金鑰的金鑰狀態為 Unavailable 或 PendingDeletion。(已停用的金鑰會傳回 DisabledException。)

- 只有當您刻意在 AWS KMS 主控台中停用 KMS 金鑰或使用 [DisableKey](#) 操作時，KMS 金鑰才會有金鑰 Disabled 狀態。當 KMS 金鑰被停用時，您可以檢視和管理金鑰，但無法在密碼編譯操作中使用它。若要修正此問題，請啟用金鑰。如需詳細資訊，請參閱[啟用和停用金鑰](#)。
- 當外部金鑰存放區與其外部金鑰存放區代理中斷連接時，KMS 金鑰的金鑰狀態為 Unavailable。若要修正無法使用的 KMS 金鑰，請[重新連接外部金鑰存放區](#)。外部金鑰存放區重新連接之後，外部金鑰存放區中 KMS 金鑰的金鑰狀態會自動還原到其先前的狀態，例如 Enabled 或 Disabled。

當 KMS 金鑰已排程刪除且處於等待期間時，KMS 金鑰的金鑰狀態為 PendingDeletion。正在等待刪除的 KMS 金鑰的金鑰狀態錯誤表示不應該刪除該金鑰，因為它正用於加密，或者需要它進行解密。若要重新啟用 KMS 金鑰，請取消已排程的刪除，然後[啟用金鑰](#)。如需詳細資訊，請參閱[排程金鑰刪除](#)。

解密錯誤

例外狀況：KMSInvalidStateException

當外部金鑰存放區中使用 KMS 金鑰的[解密](#)操作失敗時，會 AWS KMS 傳回 KMSInvalidStateException 密碼編譯操作用於外部金鑰存放區上所有外部組態錯誤和連線狀態錯誤的標準。指出問題的錯誤訊息。

若要解密使用[雙重加密](#)來加密的密文，外部金鑰管理器會首先使用外部金鑰來解密密文外層。然後使用 AWS KMS KMS AWS KMS 金鑰中的金鑰材料來解密加密文字的內層。外部金鑰管理器或 AWS KMS 可拒絕無效或損毀的密文。

解密失敗時，`KMSInvalidStateException` 會伴隨下列錯誤訊息。它指出請求中的密文或選用的加密內容有問題。

外部金鑰存放區代理拒絕請求，因為指定的密文或其他已驗證的資料已損毀、遺失或無效。

- 當外部金鑰存放區代理或外部金鑰管理器報告加密文字或其加密內容無效時，通常表示傳送 `Decrypt` 請求中的加密文字或加密內容有問題 AWS KMS。對於 `Decrypt` 操作，AWS KMS 會傳送與 `Decrypt` 請求中接收到的相同加密文字和加密內容給代理。

此錯誤可能是由傳輸過程中的網路問題所引起，例如翻轉位。重試 `Decrypt` 請求。如果問題仍然存在，請確認密文沒有被更改或損壞。此外，請確認 `Decrypt` 請求中的加密內容與加密資料的請求中的加密內容 AWS KMS 相符。

外部金鑰存放區代理提交用於解密的密文或加密內容已損毀、遺失或無效。

- 當 AWS KMS 拒絕從代理收到的加密文字時，表示外部金鑰管理器或代理傳回無效或損毀的加密文字 AWS KMS。

此錯誤可能是由傳輸過程中的網路問題所引起，例如翻轉位。重試 `Decrypt` 請求。如果問題仍然存在，請確認外部金鑰管理器是否正常運作，而且外部金鑰存放區代理不會在傳回給外部金鑰管理器之前變更從外部金鑰管理器收到的加密文字 AWS KMS。

外部金鑰錯誤

[外部金鑰](#) 是外部金鑰管理器中的密碼編譯金鑰，可作為 KMS 金鑰的外部金鑰材料。AWS KMS 無法直接存取外部金鑰。它必須要求外部金鑰管理器 (透過外部金鑰存放區代理) 使用外部金鑰來加密資料或解密密文。

在外部金鑰存放區中建立 KMS 金鑰時，您可以在其外部金鑰管理器中指定外部金鑰的 ID。您無法在建立 KMS 金鑰之後變更外部金鑰 ID。為了避免 KMS 金鑰發生問題，`CreateKey` 操作會要求外部金鑰存放區代理驗證外部金鑰的 ID 和組態。如果外部金鑰 [不滿足](#) 與 KMS 金鑰搭配使用的要求，則 `CreateKey` 操作會失敗，並顯示可識別問題的例外狀況和錯誤訊息。

不過，建立 KMS 金鑰之後可能會發生問題。如果密碼編譯操作由於外部金鑰問題而失敗，則操作會失敗並傳回 `KMSInvalidStateException` 和指出問題的錯誤訊息。

外部金鑰的 CreateKey 錯誤

例外狀

況：XksKeyAlreadyInUseException、XksKeyNotFoundException、XksKeyInvalidConfigurationException

[CreateKey](#) 操作會嘗試驗證您在 External key ID (外部金鑰 ID) (主控台) 或 XksKeyId (API) 參數中提供的外部金鑰的 ID 和屬性。此做法的設計目的在於在您嘗試搭配使用外部金鑰和 KMS 金鑰之前，提前偵測錯誤。

使用中的外部金鑰

外部金鑰存放區中的每個 KMS 金鑰都必須使用不同的外部金鑰。當 CreateKey 識別出 KMS 金鑰的外部金鑰 ID (xKSKeyID) 在外部金鑰存放區中不唯一時，就會失敗並顯示 XksKeyAlreadyInUseException。

如果您為相同外部金鑰使用多個 ID，CreateKey 將無法識別重複內容。不過，具有相同外部金鑰的 KMS 金鑰無法互通，因為它們具有不同的 AWS KMS 金鑰材料和中繼資料。

找不到外部金鑰

當外部金鑰存放區代理報告它無法使用 KMS 金鑰的外部金鑰 ID (xKSKeyID) 找到外部金鑰時，CreateKey 操作會失敗，並傳回 XksKeyNotFoundException 及下列錯誤訊息。

外部金鑰存放區代理拒絕了請求，因為它找不到外部金鑰。

此錯誤可能發生的原因如下：

- KMS 金鑰的外部金鑰 ID (XksKeyId) 可能無效。若要查找外部金鑰代理用來識別外部金鑰的 ID，請參閱外部金鑰存放區代理或外部金鑰管理器文件。
- 可能已從外部金鑰管理器刪除外部金鑰。若要進行調查，請使用外部金鑰管理器工具。如果外部金鑰已永久刪除，請搭配 KMS 金鑰使用其他外部金鑰。如需外部金鑰的清單或要求，請參閱 [外部金鑰存放區中 KMS 金鑰的要求](#)。

未滿足外部金鑰要求

當外部金鑰存放區代理報告外部金鑰[不符合](#)與 KMS 金鑰搭配使用的要求時，CreateKey 操作會失敗，並傳回 XksKeyInvalidConfigurationException 及下列其中一個錯誤訊息。

外部金鑰的金鑰規格必須為 AES_256。指定外部金鑰的金鑰規格為 *<key-spec>* 。

- 外部金鑰必須是 256 位元對稱加密金鑰，金鑰規格為 AES_256。如果指定的外部金鑰是不同的類型，請指定符合此要求的外部金鑰 ID。

外部金鑰的狀態必須為 ENABLED。指定外部金鑰的狀態為 *<status>*。

- 必須在外部金鑰管理器中啟用外部金鑰。如果指定的外部金鑰未啟用，請使用外部金鑰管理器工具來啟用它，或指定已啟用的外部金鑰。

外部金鑰的金鑰用途必須包括 ENCRYPT 和 DECRYPT。指定外部金鑰的金鑰用途是 *<key-usage>*。

- 外部金鑰必須設定為在外部金鑰管理器中進行加密和解密。如果指定的外部金鑰不包含這些操作，請使用外部金鑰管理器工具變更操作，或指定其他外部金鑰。

外部金鑰的密碼編譯操作錯誤

例外狀況：KMSInvalidStateException

當外部金鑰存放區代理找不到與 KMS 金鑰相關聯的外部金鑰時，或外部金鑰[不滿足](#)與 KMS 金鑰搭配使用的要求時，密碼編譯操作會失敗。

與建立 KMS 金鑰之前偵測到的外部金鑰問題相比，在密碼編譯操作期間偵測到的外部金鑰問題更難解決。您無法在建立 KMS 金鑰之後變更外部金鑰 ID。如果 KMS 金鑰尚未加密任何資料，您可以刪除 KMS 金鑰，然後使用不同的外部金鑰 ID 建立新金鑰。不過，任何其他 KMS 金鑰都無法解密使用 KMS 金鑰產生的加密文字，即使是具有相同外部金鑰的金鑰，因為金鑰會有不同的金鑰中繼資料和不同的 AWS KMS 金鑰材料。相反，盡可能使用外部金鑰管理器工具來解決外部金鑰問題。

當外部金鑰存放區代理報告外部金鑰問題時，密碼編譯操作會傳回 KMSInvalidStateException 以及可識別問題的錯誤訊息。

找不到外部金鑰

當外部金鑰存放區代理報告其無法使用 KMS 金鑰的外部金鑰 ID (xKSKeyID) 找到外部金鑰時，密碼編譯操作會傳回 `KMSInvalidStateException` 以及下列錯誤訊息。

外部金鑰存放區代理拒絕了請求，因為它找不到外部金鑰。

此錯誤可能發生的原因如下：

- KMS 金鑰的外部金鑰 ID (XksKeyId) 已不再有效。

若要查找與 KMS 金鑰相關聯的外部金鑰 ID，[請檢視 KMS 金鑰的詳細資訊](#)。若要查找外部金鑰代理用來識別外部金鑰的 ID，請參閱外部金鑰存放區代理或外部金鑰管理器文件。

AWS KMS 在外部金鑰存放區中建立 KMS 金鑰時，會驗證外部金鑰 ID。不過，ID 可能會變得無效，特別是如果外部金鑰 ID 值是別名或可變名稱。您不能變更與現有 KMS 金鑰相關聯的外部金鑰 ID。若要解密使用 KMS 金鑰加密的任何密文，您必須將外部金鑰與現有的外部金鑰 ID 重新關聯。

如果您尚未使用 KMS 金鑰加密資料，您可以使用有效的外部金鑰 ID 建立新的 KMS 金鑰。但是，如果您已使用 KMS 金鑰產生密文，即使使用相同的外部金鑰，也無法使用任何其他 KMS 金鑰來解密密文。

- 可能已從外部金鑰管理器刪除外部金鑰。若要進行調查，請使用外部金鑰管理器工具。如果可能，請嘗試從外部金鑰管理器的複本或備份中[復原金鑰材料](#)。如果永久刪除外部金鑰，則使用關聯的 KMS 金鑰加密的任何密文都無法復原。

外部金鑰組態錯誤

當外部金鑰存放區代理報告外部金鑰[不符合](#)與 KMS 金鑰搭配使用的要求時，密碼編譯操作會傳回 `KMSInvalidStateException` 及下列其中一個錯誤訊息。

外部金鑰存放區代理拒絕了請求，因為外部金鑰不支持請求的操作。

- 外部金鑰必須同時支援加密和解密。如果金鑰用途不包含加密和解密，請使用外部金鑰管理器工具來變更金鑰用途。

外部金鑰存放區代理拒絕了請求，因為外部金鑰管理器中未啟用外部金鑰。

- 外部金鑰必須已啟用並且可用於外部金鑰管理器。如果外部金鑰的狀態不是 `Enabled`，請使用外部金鑰管理器工具來啟用它。

代理問題

例外狀況：

`CustomKeyStoreInvalidStateException (CreateKey)`、`KMSInvalidStateException (密碼編譯操`

`作)`、`UnsupportedOperationException`、`XksProxyUriUnreachableException`、`XksProxyInvalid (CreateKey 以外的管理操作)`

外部金鑰存放區代理會媒介 AWS KMS 和外部金鑰管理器之間的所有通訊。它會將一般 AWS KMS 請求轉譯為外部金鑰管理器可以了解的格式。如果外部金鑰存放區代理不符合 [AWS KMS 外部金鑰存放區代理 API 規格](#)，或無法正常運作或無法與通訊 AWS KMS，您將無法在外部金鑰存放區中建立或使用 KMS 金鑰。

雖然許多錯誤提到外部金鑰存放區代理，因為它在外部金鑰存放區架構中起著關鍵作用，但這些問題可能源於外部金鑰管理器或外部金鑰。

本節中的問題與外部金鑰存放區代理的設計或操作問題有關。解決這些問題可能需要變更代理軟體。請諮詢您的代理管理員。為了協助診斷代理問題，AWS KMS 會提供 [XKS Proxy Text Client](#)，它是一個開放原始碼測試用戶端，可驗證您的外部金鑰存放區代理是否符合 [AWS KMS 外部金鑰存放區代理 API 規格](#)。

`CustomKeyStoreInvalidStateException`、`KMSInvalidStateException` 或 `XksProxyUriUnreachableException`

外部金鑰存放區代理處於運作不佳狀態。如果您重複看到此訊息，請通知外部金鑰存放區代理管理員。

- 此錯誤表示外部金鑰存放區代理中存在操作問題或軟體錯誤。您可以找到產生每個錯誤的 AWS KMS API 操作的 CloudTrail 日誌項目。重試操作可能會解決此錯誤。但是，如果問題仍然存在，請通知您的外部金鑰存放區代理管理員。
- 當外部金鑰存放區代理報告 (在 [GetHealthStatus](#) 回應中) 所有外部金鑰管理器執行個體為 `UNAVAILABLE` 時，嘗試建立或更新外部金鑰存放區會失敗，並顯示此例外狀況。如果此錯誤仍然存在，請參閱外部金鑰管理器文件。

`CustomKeyStoreInvalidStateException`、`KMSInvalidStateException` 或 `XksProxyInvalidResponseException`

AWS KMS 無法解譯來自外部金鑰存放區代理的回應。如果您重複看到此錯誤，請諮詢您的外部金鑰存放區代理管理員。

- AWS KMS 當代理傳回 AWS KMS 無法剖析或解譯的未定義回應時，操作會產生此例外狀況。由於暫時的外部問題或偶發的網絡錯誤，偶爾會發生此錯誤。但是，如果它仍然存在，則可能表示外部金鑰存放區代理不符合 [AWS KMS 外部金鑰存放區代理 API 規格](#)。通知您的外部金鑰存放區管理員或廠商。

`CustomKeyStoreInvalidStateException`、`KMSInvalidStateException` 或 `UnsupportedOperationException`

外部金鑰存放區代理拒絕了請求，因為其不支援請求的密碼編譯操作。

- 外部金鑰存放區代理應支援 [AWS KMS 外部金鑰存放區代理 API 規格](#) 中定義的所有 [代理 API](#)。此錯誤表示代理不支援與請求相關的操作。通知您的外部金鑰存放區管理員或廠商。

代理授權問題

例外狀況：CustomKeyStoreInvalidStateException、KMSInvalidStateException

某些外部金鑰存放區代理會實作使用其外部金鑰的授權需求。允許外部金鑰存放區代理 (但不是必需的) 來設計和實作授權方案，該方案允許特定使用者在特定條件下請求特定操作。例如，代理可能允許使用者使用特定的外部金鑰進行加密，但無法使用它進行解密。如需詳細資訊，請參閱 [外部金鑰存放區代理授權 \(選用\)](#)。

Proxy 授權是以中繼資料為基礎，這些中繼資料 AWS KMS 包含在對 Proxy 的請求中。只有當請求來自 VPC 端點且呼叫者與 KMS 金鑰位於相同帳戶時，`awsSourceVpc` 和 `awsSourceVpce` 欄位才會包含在中繼資料中。

```
"requestMetadata": {
  "awsPrincipalArn": string,
  "awsSourceVpc": string, // optional
  "awsSourceVpce": string, // optional
```

```
"kmsKeyArn": string,  
"kmsOperation": string,  
"kmsRequestId": string,  
"kmsViaService": string // optional  
}
```

當代理因授權失敗而拒絕請求時，相關 AWS KMS 操作會失敗。CreateKey 會傳回 CustomKeyStoreInvalidStateException。AWS KMS cryptographic 操作會傳回 KMSInvalidStateException。兩者都使用下列錯誤訊息：

外部金鑰存放區代理拒絕存取該操作。確認使用者和外部金鑰都已獲得針對此操作的授權，然後再試一次請求。

- 若要解決錯誤，請使用外部金鑰管理器或外部金鑰存放區代理工具來確定授權失敗的原因。然後，更新導致未授權請求的程序，或使用外部金鑰存放區代理工具來更新授權政策。您無法在 AWS KMS 中解決此錯誤。

的安全性 AWS Key Management Service

的雲端安全 AWS 是最高優先順序。身為 AWS 客戶，您可以受益於資料中心和網路架構，這些架構專為滿足最安全敏感組織的需求而建置。

安全性是 AWS 與您之間共同責任。[共同責任模型](#)將其描述為雲端的安全性和雲端中的安全性：

- 雲端的安全性 – AWS 負責保護在 AWS Cloud 中執行 AWS 服務的基礎設施。AWS 也為您提供可安全使用的服務。在[AWS 合規計劃](#)合規中，第三方稽核人員會定期測試和驗證我們安全的有效性。若要了解適用於 AWS Key Management Service (AWS KMS) 的合規計劃，請參閱[AWS 合規計劃的服務範圍](#)。
- 雲端的安全性 – 您的責任取決於您使用 AWS 的服務。此外 AWS KMS，除了您的組態和使用之外 AWS KMS keys，您還需負責其他因素，包括資料的敏感度、您的公司的需求，以及適用的法律和法規

本文件可協助您了解如何在使用時套用共同責任模型 AWS Key Management Service。它說明如何設定 AWS KMS 以符合您的安全與合規目標。

主題

- [中的資料保護 AWS Key Management Service](#)
- [的身分和存取管理 AWS Key Management Service](#)
- [在 中記錄和監控 AWS Key Management Service](#)
- [的合規驗證 AWS Key Management Service](#)
- [中的彈性 AWS Key Management Service](#)
- [中的基礎設施安全 AWS Key Management Service](#)

中的資料保護 AWS Key Management Service

AWS Key Management Service 會存放和保護您的加密金鑰，使其高度可用，同時為您提供強大且靈活的存取控制。

主題

- [保護金鑰資料](#)
- [資料加密](#)
- [網際網路流量隱私權](#)

保護金鑰資料

根據預設，AWS KMS 會產生並保護 KMS 金鑰的密碼編譯金鑰材料。此外，為在外部建立和保護的金鑰材料 AWS KMS 提供選項 AWS KMS。

保護中產生的金鑰材料 AWS KMS

根據預設，當您建立 KMS 金鑰時，AWS KMS 會產生並保護 KMS 金鑰的密碼編譯資料。

為了保護 KMS 金鑰的金鑰材料，AWS KMS 仰賴 [FIPS 140-3 安全層級 3 驗證硬體](#) 安全模組 (HSMs) 的分散式機群。每個 AWS KMS HSM 都是專用、獨立的硬體設備，旨在提供專用密碼編譯函數，以滿足的安全性和可擴展性要求 AWS KMS。(在中國區域中 AWS KMS 使用的 HSMs 經過 [OSCCA](#) 認證，並符合所有相關中國法規，但未經 FIPS 140-3 密碼編譯模組驗證計畫驗證。)

根據預設，KMS 金鑰的金鑰資料會在 HSM 產生時加密。金鑰資料僅在 HSM 揮發性記憶體內進行解密，且解密時間僅限密碼編譯操作運用金鑰資料所需的數毫秒。每當金鑰資料未處於作用中使用狀態時，就會在 HSM 加密，並傳輸至 [高持性](#) (99.999999999%) 且低延遲持續性儲存體，其會在此儲存體保持獨立並與 HSM 隔離。純文字金鑰資料永遠不會離開 HSM [安全界限](#)；永遠不會寫入磁碟或保存於任何儲存媒體。(唯一例外是非對稱金鑰對的公有金鑰，其非秘密。)

AWS 宣告 作為基本安全原則，在任何類型中都不會與純文字密碼編譯金鑰材料進行人工互動 AWS 服務。沒有任何機制可讓包括 AWS 服務 運算子在內的任何人檢視、存取或匯出純文字金鑰材料。即使在災難性故障及災難復原事件，此政策也適用。中的純文字客戶金鑰材料 AWS KMS 僅用於 AWS KMS FIPS 140-3 驗證 HSMs 內的密碼編譯操作，僅用於回應客戶或其委派人對服務提出的授權請求。

對於 [客戶受管金鑰](#)，建立金鑰 AWS 帳戶 的是金鑰的唯一且不可轉移的擁有者。擁有帳戶對控制金鑰存取權的授權政策具完整且獨佔控制權。對於 AWS 受管金鑰，AWS 帳戶 可完全控制授權 請求的 IAM 政策 AWS 服務。

保護在外部產生的金鑰材料 AWS KMS

AWS KMS 提供在 中產生之金鑰材料的替代方案 AWS KMS。

[自訂金鑰存放區](#)是選用 AWS KMS 功能，可讓您建立 KMS 金鑰，其由外部產生和使用的金鑰材料提供支援 AWS KMS。[AWS CloudHSM 金鑰存放區](#)中的 KMS 金鑰由您控制的 AWS CloudHSM 硬體安全模組中的金鑰支援。這些 HSMs 已經過 [FIPS 140-2 安全層級 3 或 140-3 安全層級 3](#) 認證。[外部金鑰存放區中的 KMS 金鑰](#)由您在外部控制和管理的 [外部金鑰管理器](#)中的金鑰提供支援 AWS，例如私有資料中心中的實體 HSM。

另一個選用功能可讓您為 KMS 金鑰[匯入金鑰材料](#)。為了保護傳輸中的匯入金鑰材料 AWS KMS，您可以使用 AWS KMS HSM 中產生的 RSA 金鑰對中的公有金鑰來加密金鑰材料。匯入的金鑰材料會在 AWS KMS HSM 中解密，並在 HSM 中的對稱金鑰下重新加密。如同所有 AWS KMS 金鑰材料，純文字匯入的金鑰材料永遠不會讓 HSMs 未加密。然而，提供金鑰資料的客戶需負責安全使用、持久性及 AWS KMS 外部的金鑰資料維護。

資料加密

中的資料 AWS KMS 包含 AWS KMS keys 及其代表的加密金鑰材料。此金鑰材料僅以純文字形式存在於 AWS KMS 硬體安全模組 (HSM)，且僅在使用時存在。否則，金鑰材料會加密並存放在耐久的持久性儲存裝置中。

為 KMS 金鑰 AWS KMS 產生的金鑰材料絕不會讓 HSMs 的 AWS KMS 界限處於未加密狀態。它不會在任何 AWS KMS API 操作中匯出或傳輸。[多區域金鑰](#)例外，其中 AWS KMS 使用跨區域複寫機制，將多區域金鑰的金鑰材料從 HSM 複製到 AWS 區域 HSM AWS 區域。如需詳細資訊，請參閱 AWS Key Management Service 密碼編譯詳細資訊中[多區域金鑰的複寫程序](#)。

主題

- [靜態加密](#)
- [傳輸中加密](#)

靜態加密

AWS KMS [在 FIPS 140-3 安全層級 3](#) 相容硬體安全模組 (HSMs) AWS KMS keys 中產生的金鑰材料。唯一的例外是中國區域，其中 AWS KMS 用來產生 KMS 金鑰 HSMs 符合所有相關中國法規，但未經 FIPS 140-3 密碼編譯模組驗證計畫驗證。不使用時，金鑰資料會由 HSM 金鑰加密，並寫入耐久的持久性儲存裝置。KMS 金鑰的金鑰材料和保護金鑰材料的加密金鑰永遠不會讓 HSM 以純文字形式出現。

KMS 金鑰的金鑰材料加密和管理完全由 AWS KMS 處理。

如需詳細資訊，請參閱[在 AWS KMS keys](#)密碼編譯詳細資訊中使用 AWS Key Management Service。

傳輸中加密

為 KMS 金鑰 AWS KMS 產生的金鑰材料絕不會在 AWS KMS API 操作中匯出或傳輸。AWS KMS 會使用[金鑰識別符](#)來代表 API 操作中的 KMS 金鑰。同樣地，KMS 金鑰 AWS KMS [自訂金鑰存放區的金鑰](#)材料不可匯出，且絕不會在 AWS KMS 或 AWS CloudHSM API 操作中傳輸。

不過，某些 AWS KMS API 操作會傳回[資料金鑰](#)。此外，客戶可以使用 API 操作為所選的 KMS 金鑰[匯入金鑰材料](#)。

所有 AWS KMS API 呼叫都必須使用 Transport Layer Security (TLS) 進行簽署和傳輸。在所有區域中 AWS KMS 需要 TLS 1.2 並建議 TLS 1.3。AWS KMS 也支援所有區域中 AWS KMS 服務端點的混合式後量子 TLS，中國區域除外。AWS KMS 不支援 FIPS 端點的混合後量子 TLS AWS GovCloud (US)。呼叫 AWS KMS 還需要支援完整轉寄密碼的現代加密套件，這表示任何機密 (例如私有金鑰) 的洩露也不會影響工作階段金鑰。

如果您在 AWS 透過命令列界面或 API 存取時需要 FIPS 140-3 驗證的密碼編譯模組，請使用 FIPS 端點。若要使用標準 AWS KMS 端點或 AWS KMS FIPS 端點，用戶端必須支援 TLS 1.2 或更新版本。如需有關 FIPS 和 FIPS 端點的更多相關資訊，請參閱[聯邦資訊處理標準 \(FIPS\) 140-3](#)。如需 AWS KMS FIPS 端點的清單，請參閱 [AWS Key Management Service 中的端點和配額](#) AWS 一般參考。

在已驗證的加密機制中，AWS KMS 服務主機和 HSMs 之間的通訊會使用橢圓曲線密碼編譯 (ECC) 和進階加密標準 (AES) 進行保護。如需詳細資訊，請參閱 AWS Key Management Service 密碼編譯詳細資訊中的[內部通訊安全](#)。

網際網路流量隱私權

AWS KMS 支援 AWS 管理主控台 和一組 API 操作，可讓您在密碼編譯操作中建立和管理 AWS KMS keys 和使用它們。

AWS KMS 支援從私有網路到 的兩個網路連線選項 AWS。

- 網際網路上的 IPsec VPN 連接
- [AWS Direct Connect](#)，透過標準乙太網路光纖纜線將內部網路連結至 Direct Connect 位置。

所有 AWS KMS API 呼叫都必須使用 Transport Layer Security (TLS) 簽署和傳輸。這些呼叫還需要支援[完整轉寄密碼](#)的現代加密套件。只能透過 AWS 內部網路從已知 AWS KMS API 主機傳輸到存放 KMS 金鑰之金鑰資料的硬體安全模組 (HSMs)。

若要 AWS KMS 從虛擬私有雲端 (VPC) 直接連線至，而不透過公有網際網路傳送流量，請使用採用技術的 VPC 端點[AWS PrivateLink](#)。如需詳細資訊，請參閱[AWS KMS 透過 VPC 端點連線至](#)。

AWS KMS 也支援 Transport Layer Security (TLS) 網路加密通訊協定的[混合式後量子金鑰交換](#)選項。連線至 AWS KMS API 端點時，您可以將此選項與 TLS 搭配使用。

的身分和存取管理 AWS Key Management Service

AWS Identity and Access Management (IAM) 可協助您安全地控制對 AWS 資源的存取。管理員可控制誰可以驗證（登入）和授權（具有許可）使用 AWS KMS 資源。如需詳細資訊，請參閱[搭配使用 IAM 政策 AWS KMS](#)。

[金鑰政策](#)是控制 KMS 金鑰存取的主要機制 AWS KMS。每個 KMS 金鑰都必須有一個金鑰政策。您也可以使用 [IAM 政策](#)和[授予](#)，以及金鑰政策，以控制對 KMS 金鑰的存取。如需詳細資訊，請參閱[KMS 金鑰存取和許可](#)。

如果您使用的是 Amazon Virtual Private Cloud (Amazon VPC)，則可以[建立介面 VPC 端點](#)以採用 AWS KMS 技術[AWS PrivateLink](#)。您也可以使用 VPC 端點政策來判斷哪些主體可以存取您的 AWS KMS 端點、他們可以進行哪些 API 呼叫，以及他們可以存取哪些 KMS 金鑰。

主題

- [AWS 的 受管政策 AWS Key Management Service](#)
- [使用的服務連結角色 AWS KMS](#)

AWS 的 受管政策 AWS Key Management Service

AWS 受管政策是由 AWS AWS 受管政策建立和管理的獨立政策旨在為許多常用案例提供許可，以便您可以開始將許可指派給使用者、群組和角色。

請記住，AWS 受管政策可能不會授予特定使用案例的最低權限許可，因為這些許可可供所有 AWS 客戶使用。我們建議您定義特定於使用案例的[客戶管理政策](#)，以便進一步減少許可。

您無法變更 AWS 受管政策中定義的許可。如果 AWS 更新受 AWS 管政策中定義的許可，則更新會影響政策連接的所有委託人身分（使用者、群組和角色）。AWS 服務 當新的 啟動或新的 API 操作可用於現有服務時，AWS 最有可能更新 AWS 受管政策。

如需詳細資訊，請參閱 IAM 使用者指南中的 [AWS 受管政策](#)。

AWS 受管政策：AWSKeyManagementServicePowerUser

您可將 AWSKeyManagementServicePowerUser 政策連接到 IAM 身分。

您可以使用 AWSKeyManagementServicePowerUser 受管政策，為您帳戶中的 IAM 主體提供進階使用者的許可。進階使用者可以建立 KMS 金鑰、使用和管理其建立的 KMS 金鑰，以及檢視所有 KMS

金鑰和 IAM 身分識別。擁有 `AWSKeyManagementServicePowerUser` 受管政策的主體也可以從其他來源取得許可，包括金鑰政策、其他 IAM 政策和授予。

`AWSKeyManagementServicePowerUser` 是 AWS 受管 IAM 政策。如需 AWS 受管政策的詳細資訊，請參閱《IAM 使用者指南》中的 [AWS 受管政策](#)。

 Note

此政策中特定於 KMS 金鑰的許可，例如 `kms:TagResource` 和 `kms:GetKeyRotationStatus`，僅當該 KMS 金鑰的金鑰政策[明確允許 AWS 帳戶使用 IAM 政策](#)來控制對金鑰的存取時才有效。若要確定許可是否特定於 KMS 金鑰，請參閱 [AWS KMS 許可](#) 並查找資源資料欄中的 KMS 金鑰。

這項政策為進階使用者提供針對任何 KMS 金鑰的許可，搭配允許操作的金鑰政策。對於跨帳戶許可，例如 `kms:DescribeKey` 和 `kms:ListGrants`，這可能包括不受信任的 AWS 帳戶中的 KMS 金鑰。如需詳細資訊，請參閱 [IAM 政策的最佳實務](#) 和 [允許其他帳戶中的使用者使用 KMS 金鑰](#)。若要確定許可是否對其他帳戶中的 KMS 金鑰有效，請參閱 [AWS KMS 許可](#) 並查找跨帳戶使用資料欄中的是。

若要允許主體在不發生錯誤的情況下檢視 AWS KMS 主控台，主體需要 `tag : GetResources` 許可，此許可不包含在 `AWSKeyManagementServicePowerUser` 政策中。您可以在單獨的 IAM 政策中允許此許可。

[AWSKeyManagementServicePowerUser](#) 受管 IAM 政策包含以下許可。

- 允許主體建立 KMS 金鑰。由於此程序包含設定金鑰政策，進階使用者可以授予自己和其他人使用和管理其所建立之 KMS 金鑰的許可。
- 允許主體對所有 KMS 金鑰建立和刪除[別名](#)和[標籤](#)。變更標籤或別名可允許或拒絕使用和管理 KMS 金鑰的許可。如需詳細資訊，請參閱[適用於的 ABAC AWS KMS](#)。
- 允許主體取得有關所有 KMS 金鑰的詳細資訊，包括金鑰 ARN、密碼編譯組態、金鑰政策、別名、標籤和[輪換狀態](#)。
- 允許主體列出 IAM 使用者、群組和角色。
- 此政策不允許主體使用或管理其未建立的 KMS 金鑰。然而，它們可以變更所有 KMS 金鑰上的別名和標籤，這可能允許或拒絕使用或管理 KMS 金鑰的許可。

若要檢視此政策的許可，請參閱《AWS 受管政策參考》中的 [AWSKeyManagementServicePowerUser](#)。

AWS 受管政策：AWSServiceRoleForKeyManagementServiceCustomKeyStores

您不得將 `AWSServiceRoleForKeyManagementServiceCustomKeyStores` 連接到 IAM 實體。此政策會連接到服務連結角色，該角色授予 AWS KMS 許可來檢視與 AWS CloudHSM 金鑰存放區相關聯的 AWS CloudHSM 叢集，並建立網路以支援自訂金鑰存放區與其 AWS CloudHSM 叢集之間的連線。如需詳細資訊，請參閱[授權 AWS KMS 管理 AWS CloudHSM 和 Amazon EC2 資源](#)。

AWS 受管政策：AWSServiceRoleForKeyManagementServiceMultiRegionKeys

您不得將 `AWSServiceRoleForKeyManagementServiceMultiRegionKeys` 連接到 IAM 實體。此政策連接到服務連結角色，該角色授予 AWS KMS 許可，以將多區域主要金鑰的金鑰材料的任何變更同步至其複本金鑰。如需詳細資訊，請參閱[授權 AWS KMS 同步多區域金鑰](#)。

AWS KMS AWS 受管政策的更新

檢視自此服務開始追蹤這些變更 AWS KMS 以來，AWS 受管政策更新的詳細資訊。如需有關此頁面變更的自動提醒，請訂閱 AWS KMS [文件歷史紀錄](#) 頁面的 RSS 摘要。

變更	描述	Date
AWSKeyManagementServiceMultiRegionKeysServiceRolePolicy – 更新現有政策	AWS KMS 已將陳述式 ID (Sid) 欄位新增至政策版本 v2 中的 受管政策。	2024 年 11 月 21 日
AWSKeyManagementServiceCustomKeyStoresServiceRolePolicy – 更新現有政策	AWS KMS 新增 <code>ec2:DescribeVpcs</code> 、 <code>ec2:DescribeNetworkAcls</code> 和 <code>ec2:DescribeNetworkInterfaces</code> 許可，以監控包含 AWS CloudHSM 叢集的 VPC 中的變更，以便 AWS KMS 可以在失敗時提供明確的錯誤訊息。	2023 年 11 月 10 日
AWS KMS 開始追蹤變更	AWS KMS 已開始追蹤其 AWS 受管政策的變更。	2023 年 11 月 10 日

使用的服務連結角色 AWS KMS

AWS Key Management Service use AWS Identity and Access Management (IAM) [服務連結角色](#)。服務連結角色是直接連結至其中的唯一 IAM 角色類型 AWS KMS。服務連結角色是由 定義，AWS KMS 並包含該服務代表您呼叫其他 AWS 服務所需的所有許可。

服務連結角色可讓您更 AWS KMS 輕鬆地設定，因為您不必手動新增必要的許可。AWS KMS 會定義其服務連結角色的許可，除非另有定義，否則只能 AWS KMS 擔任其角色。定義的許可包括信任政策和許可政策，且該許可政策無法附加至其他 IAM 實體。

您必須先刪除相關的資源，才能刪除服務連結角色。這可保護您的 AWS KMS 資源，因為您不會不小心移除存取資源的許可。

如需關於支援服務連結角色的其他服務的資訊，請參閱[可搭配 IAM 運作的AWS 服務](#)，並尋找 Service-Linked Role (服務連結角色) 欄顯示為 Yes (是) 的服務。選擇具有連結的是，以檢視該服務的服務連結角色文件。

若要檢視本主題中所討論之服務連結角色更新的詳細資訊，請參閱 [AWS KMS AWS 受管政策的更新](#)。

主題

- [授權 AWS KMS 管理 AWS CloudHSM 和 Amazon EC2 資源](#)
- [授權 AWS KMS 同步多區域金鑰](#)

授權 AWS KMS 管理 AWS CloudHSM 和 Amazon EC2 資源

為了支援您的 AWS CloudHSM 金鑰存放區，AWS KMS 需要取得 AWS CloudHSM 叢集相關資訊的許可。它還需要許可，才能建立將 AWS CloudHSM 金鑰存放區連接到其 AWS CloudHSM 叢集的網路基礎設施。若要取得這些許可，會在您的 中 AWS KMS 建立 AWSServiceRoleForKeyManagementServiceCustomKeyStores 服務連結角色 AWS 帳戶。建立 AWS CloudHSM 金鑰存放區的使用者必須具有允許他們建立服務連結角色的 iam:CreateServiceLinkedRole 許可。

若要檢視 AWSKeyManagementServiceCustomKeyStoresServiceRolePolicy 受管政策更新的詳細資訊，請參閱 [AWS KMS AWS 受管政策的更新](#)。

主題

- [關於 AWS KMS 服務連結角色](#)
- [建立服務連結角色](#)
- [編輯服務連結角色描述](#)

- [刪除服務連結角色](#)

關於 AWS KMS 服務連結角色

[服務連結角色](#)是 IAM 角色，提供一項 AWS 服務代表您呼叫其他 AWS 服務的許可。它旨在讓您更輕鬆地使用多個整合 AWS 服務的功能，而無需建立和維護複雜的 IAM 政策。如需詳細資訊，請參閱[使用的服務連結角色 AWS KMS](#)。

對於 AWS CloudHSM 金鑰存放區，會使用 `AWSKeyManagementServiceCustomKeyStores` 受管政策 AWS KMS 建立 `AWSKeyManagementServiceCustomKeyStoresServiceRolePolicy` 服務連結角色。 `AWSServiceRoleForKeyManagementServiceCustomKeyStores` 此政策將下列許可授予此角色：

- [cloudhsm : Describe*](#) – 偵測連接到自訂金鑰存放區的 AWS CloudHSM 叢集中的變更。
- [ec2 : CreateSecurityGroup](#) – 當您[連接 AWS CloudHSM 金鑰存放區](#)來建立安全群組，以啟用 AWS KMS 和 AWS CloudHSM 叢集之間的網路流量流。
- [ec2 : AuthorizeSecurityGroupIngress](#) – 當您[連接 AWS CloudHSM 金鑰存放區](#)以允許網路 AWS KMS 從 存取包含 AWS CloudHSM 叢集的 VPC 時使用。
- [ec2 : CreateNetworkInterface](#) – 當您[連接 AWS CloudHSM 金鑰存放區](#)以建立用於在 AWS KMS 和 AWS CloudHSM 叢集之間通訊的網路介面時使用。
- [ec2 : RevokeSecurityGroupEgress](#) – 當您[連接 AWS CloudHSM 金鑰存放區](#)以從 AWS KMS 建立的安全群組中移除所有傳出規則時使用。
- [ec2 : DeleteSecurityGroup](#) – 用於[中斷連接 AWS CloudHSM 金鑰存放區](#)，以刪除連接 AWS CloudHSM 金鑰存放區時建立的安全群組。
- [ec2 : DescribeSecurityGroups](#) – 用於監控在包含 AWS CloudHSM 叢集的 VPC 中 AWS KMS 建立的安全群組中的變更，以便在失敗時 AWS KMS 提供明確的錯誤訊息。
- [ec2 : DescribeVpcs](#) – 用於監控包含 AWS CloudHSM 叢集的 VPC 中的變更，以便 AWS KMS 可以在失敗時提供明確的錯誤訊息。
- [ec2 : DescribeNetworkAcls](#) – 用於監控包含 AWS CloudHSM 叢集之 VPC 的網路 ACLs 中的變更，以便在發生故障時 AWS KMS 提供明確的錯誤訊息。
- [ec2 : DescribeNetworkInterfaces](#) – 用於監控在包含 AWS CloudHSM 叢集的 VPC 中 AWS KMS 建立之網路介面的變更，AWS KMS 以便 可以在失敗時提供明確的錯誤訊息。

JSON

```
{
```

```

"Version": "2012-10-17",
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "cloudhsm:Describe*",
      "ec2:CreateNetworkInterface",
      "ec2:AuthorizeSecurityGroupIngress",
      "ec2:CreateSecurityGroup",
      "ec2:DescribeSecurityGroups",
      "ec2:RevokeSecurityGroupEgress",
      "ec2>DeleteSecurityGroup",
      "ec2:DescribeVpcs",
      "ec2:DescribeNetworkAcls",
      "ec2:DescribeNetworkInterfaces"
    ],
    "Resource": "*"
  }
]
}

```

由於 `AWSServiceRoleForKeyManagementServiceCustomKeyStores` 服務連結角色僅信任 `cks.kms.amazonaws.com`，因此 AWS KMS 只能擔任此服務連結角色。此角色僅限於檢視 AWS CloudHSM 叢集和將 AWS CloudHSM 金鑰存放區連接至其相關聯 AWS CloudHSM 叢集 AWS KMS 所需的操作。它不會給予 AWS KMS 任何其他許可。例如，AWS KMS 沒有建立、管理或刪除 AWS CloudHSM 叢集、HSMs 或備份的許可。

區域

如同 AWS CloudHSM 金鑰存放區功能，`AWSServiceRoleForKeyManagementServiceCustomKeyStores` 角色支援 AWS KMS 和 AWS CloudHSM 可用的所有 AWS 區域。如需 AWS 區域 每個服務支援的清單，請參閱《》中的 [AWS Key Management Service 端點和配額](#) 以及 [AWS CloudHSM 端點和配額](#) Amazon Web Services 一般參考。

如需 AWS 服務如何使用服務連結角色的詳細資訊，請參閱《IAM 使用者指南》中的 [使用服務連結角色](#)。

建立服務連結角色

AWS KMS 當您建立 AWS CloudHSM 金鑰存放區時，如果角色尚不存在，會自動在 AWS 帳戶中建立 `AWSServiceRoleForKeyManagementServiceCustomKeyStores` 服務連結角色。您無法直接建立或重新建立此服務連結角色。

編輯服務連結角色描述

您無法編輯 `AWSServiceRoleForKeyManagementServiceCustomKeyStores` 服務連結角色中的角色名稱或政策陳述式，但可以編輯角色描述。如需相關說明，請參閱《IAM 使用者指南》中的[編輯服務連結角色](#)。

刪除服務連結角色

AWS KMS 不會從 AWS 帳戶刪除 `AWSServiceRoleForKeyManagementServiceCustomKeyStores` 服務連結角色，即使您已刪除所有 [AWS CloudHSM 金鑰存放區](#)。雖然目前沒有刪除 `AWSServiceRoleForKeyManagementServiceCustomKeyStores` 服務連結角色的程序，但除非您有作用中的 AWS CloudHSM 金鑰存放區，否則 AWS KMS 不會擔任此角色或使用其許可。

授權 AWS KMS 同步多區域金鑰

若要支援[多區域金鑰](#)，AWS KMS 需要將多區域主要金鑰的[共用屬性](#)與其複本金鑰同步的許可。若要取得這些許可，會在您的 AWS KMS 建立 `AWSServiceRoleForKeyManagementServiceMultiRegionKeys` 服務連結角色 AWS 帳戶。建立多區域金鑰的使用者必須具有允許他們建立服務連結角色的 `iam:CreateServiceLinkedRole` 許可。

您可以檢視 [SynchronizeMultiRegionKey](#) CloudTrail 事件，該事件會記錄 AWS CloudTrail 日誌中的 AWS KMS 同步共用屬性。

若要檢視 `AWSKeyManagementServiceMultiRegionKeysServiceRolePolicy` 受管政策更新的詳細資訊，請參閱 [AWS KMS AWS 受管政策的更新](#)。

主題

- [關於多區域金鑰的服務連結角色](#)
- [建立服務連結角色](#)
- [編輯服務連結角色描述](#)
- [刪除服務連結角色](#)

關於多區域金鑰的服務連結角色

[服務連結角色](#)是 IAM 角色，提供一項 AWS 服務代表您呼叫其他 AWS 服務的許可。它旨在讓您更輕鬆地使用多個整合 AWS 服務的功能，而無需建立和維護複雜的 IAM 政策。

對於多區域金鑰，會使用 `AWSKeyManagementServiceMultiRegionKeys` 受管政策 AWS KMS 建立 `AWSKeyManagementServiceMultiRegionKeysServiceRolePolicy` 服務連結角色。 `AWSServiceRoleForKeyManagementServiceMultiRegionKeys` 此政策為角色提供 `kms:SynchronizeMultiRegionKey` 許可，允許其同步多區域金鑰的共用屬性。

由於 `AWSServiceRoleForKeyManagementServiceMultiRegionKeys` 服務連結角色僅信任 `mrk.kms.amazonaws.com`，因此 AWS KMS 只能擔任此服務連結角色。此角色僅限於同步多區域共用屬性 AWS KMS 所需的操作。它不會給予 AWS KMS 任何其他許可。例如，AWS KMS 沒有建立、複寫或刪除任何 KMS 金鑰的許可。

如需 AWS 服務如何使用服務連結角色的詳細資訊，請參閱《IAM 使用者指南》中的[使用服務連結角色](#)。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "KMSSynchronizeMultiRegionKey",
      "Effect": "Allow",
      "Action": [
        "kms:SynchronizeMultiRegionKey"
      ],
      "Resource": "*"
    }
  ]
}
```

建立服務連結角色

AWS KMS 如果角色尚未存在，當您建立多區域金鑰 AWS 帳戶時，會自動在中建立 `AWSServiceRoleForKeyManagementServiceMultiRegionKeys` 服務連結角色。您無法直接建立或重新建立此服務連結角色。

編輯服務連結角色描述

您無法編輯 `AWSServiceRoleForKeyManagementServiceMultiRegionKeys` 服務連結角色中的角色名稱或政策陳述式，但可以編輯角色描述。如需相關說明，請參閱《IAM 使用者指南》中的[編輯服務連線角色](#)。

刪除服務連結角色

AWS KMS 不會從刪除 `AWSServiceRoleForKeyManagementServiceMultiRegionKeys` 服務連結角色，AWS 帳戶 而且您無法刪除它。不過，除非您的 AWS 帳戶 和 區域中有多個區域金鑰，否則 AWS KMS 不會擔任 `AWSServiceRoleForKeyManagementServiceMultiRegionKeys` 角色或使用其任何許可。

在中記錄和監控 AWS Key Management Service

監控主要用於了解 AWS KMS 中 AWS KMS keys 的可用性、狀態和使用情形。監控有助於維護 AWS 解決方案的安全性、可靠性、可用性和效能。AWS 提供多種工具來監控 KMS 金鑰。

AWS CloudTrail 日誌

對 AWS KMS API 操作的每次呼叫都會擷取為 AWS CloudTrail 日誌中的事件。這些日誌會記錄來自 AWS KMS 主控台的所有 API 呼叫，以及 AWS KMS 和其他 AWS 服務的呼叫。跨帳戶 API 呼叫，例如在不同 AWS 帳戶中使用 KMS 金鑰的呼叫，會記錄在兩者帳戶的 CloudTrail 日誌中。

故障診斷或稽核時，您可以使用日誌來重建 KMS 金鑰的生命週期。您也可以密碼編譯操作中檢視其 KMS 金鑰的管理和使用。如需詳細資訊，請參閱[the section called “使用記錄 AWS CloudTrail”](#)。

Amazon CloudWatch Logs

監控、存放和存取來自 和其他來源 AWS CloudTrail 的日誌檔案。如需更多資訊，請參閱 [Amazon CloudWatch 使用者指南](#)。

對於 AWS KMS，CloudWatch 會存放有用的資訊，協助您防止 KMS 金鑰及其保護的資源發生問題。如需詳細資訊，請參閱[the section called “使用 CloudWatch 監控金鑰”](#)。

Amazon EventBridge

AWS KMS 當您的 KMS 金鑰[輪換或刪除](#)，或 KMS 金鑰中的[匯入金鑰材料](#)過期時，會產生 EventBridge 事件。搜尋 AWS KMS 事件 (API 操作) 並將其路由至一或多個目標函數或串流，以擷取狀態資訊。如需詳細資訊，請參閱 [the section called “使用 Amazon EventBridge 監控金鑰”](#) 和 [《Amazon EventBridge 使用者指南》](#)。

Amazon CloudWatch 指標

您可以使用 CloudWatch 指標來監控 KMS 金鑰，該指標會收集原始資料並將其處理 AWS KMS 為效能指標。資料會以兩週的時間間隔記錄，以便您可以檢視目前和歷史記錄資訊的趨勢。這可協助您了解 KMS 金鑰的使用方式，及其使用方式隨著時間的變化。如需使用 CloudWatch 指標監控 KMS 金鑰的詳細資訊，請參閱 [AWS KMS 指標和維度](#)。

Amazon CloudWatch 警示

監看您指定期間的單一指標變更。然後，根據在數個期間與閾值相關的指標值，執行動作。例如，您可以建立 CloudWatch 警示，當有人嘗試使用排在密碼編譯操作中刪除的 KMS 金鑰時觸發該警示。這表示 KMS 金鑰仍在使用中，可能不應該刪除。如需詳細資訊，請參閱 [the section called “建立警示”](#)。

AWS Security Hub CSPM

您可以使用 監控您的 AWS KMS 用量，以確保安全產業標準和最佳實務合規性 AWS Security Hub CSPM。Security Hub CSPM 使用安全控制來評估資源組態和安全標準，以協助您遵守各種合規架構。如需詳細資訊，請參閱《AWS Security Hub 使用者指南》的 [AWS Key Management Service 控制](#)。

的合規驗證 AWS Key Management Service

在多個合規計畫 AWS Key Management Service 中，第三方稽核人員會評估 的安全與 AWS 合規。這些計畫包括 SOC、PCI、FedRAMP、HIPAA 等等。

主題

- [合規和安全文件](#)
- [進一步了解](#)

合規和安全文件

以下合規和安全文件涵蓋在內 AWS KMS。若要檢視，請使用 [AWS Artifact](#)。

- Cloud Computing Compliance Controls Catalogue (C5)
- ISO 27001:2013 適用性聲明 (SoA)
- ISO 27001:2013 認證
- ISO 27017:2015 適用性聲明 (SoA)

- ISO 27017:2015 認證
- ISO 27018:2015 適用性聲明 (SoA)
- ISO 27018:2014 認證
- ISO 9001:2015 認證
- PCI DSS 合規聲明文件 (AOC) 與責任摘要
- 服務組織控制 (SOC) 1 報告
- 服務組織控制 (SOC) 2 報告
- 服務組織控制 (SOC) 2 報告，針對機密性
- FedRAMP-High

如需使用 的說明 AWS Artifact，請參閱在 [AWS Artifact 中下載報告](#)。

進一步了解

您使用 時的合規責任 AWS KMS 取決於資料的機密性、您公司的合規目標，以及適用的法律和法規。如果您使用 AWS KMS 時必須遵守已發佈的標準，AWS 會提供資源來協助：

- [AWS 合規計劃範圍內的服務](#) – 此頁面列出特定合規計劃範圍內 AWS 的服務。如需一般資訊，請參閱 [AWS 合規計劃](#)。
- [安全與合規快速入門指南](#) – 這些部署指南討論架構考量，並提供在其中部署以安全與合規為重心的基準環境的步驟 AWS。
- [AWS 合規資源](#) – 此工作手冊和指南集合可能適用於您的產業和位置。
- [AWS Config](#) – AWS 此服務會評估資源組態符合內部實務、產業準則和法規的程度。
- [AWS Security Hub CSPM](#) – AWS 此服務可讓您全面檢視其中的安全狀態 AWS。Security Hub CSPM 使用安全控制來評估您的 AWS 資源，並檢查您的是否符合安全產業標準和最佳實務。如需支援的服務和控制項清單，請參閱 [Security Hub CSPM 控制項參考](#)。

中的彈性 AWS Key Management Service

AWS 全球基礎設施是以 AWS 區域 和 可用區域為基礎建置。AWS 區域 提供多個實體隔離和隔離的可用區域，這些可用區域與低延遲、高輸送量和高備援聯網連接。透過可用區域，您所設計與操作的應用程式和資料庫，就能夠在可用區域之間自動容錯移轉，而不會發生中斷。可用區域的可用性、容錯能力和擴充能力，均較單一或多個資料中心的傳統基礎設施還高。

除了 AWS 全球基礎設施之外，AWS KMS 還提供數種功能，以協助支援您的資料彈性和備份需求。如需 AWS 區域 和可用區域的詳細資訊，請參閱 [AWS 全球基礎設施](#)。

區域隔離

AWS Key Management Service (AWS KMS) 是一種自我維持的區域服務，可在所有 中使用 AWS 區域。的區域隔離設計 AWS KMS 可確保一個 中的可用性問題 AWS 區域 不會影響任何其他區域中 AWS KMS 的操作。AWS KMS 旨在確保零計劃停機時間，且所有軟體更新和擴展操作都能無縫且無法理解地執行。

AWS KMS [服務水準協議](#) (SLA) 包含所有 KMS APIs 99.999% 的服務承諾。為了履行此承諾，AWS KMS 確保執行 API 請求所需的所有資料和授權資訊在接收請求的所有區域主機上都可用。

AWS KMS 基礎設施會在每個區域中至少三個可用區域 (AZs) 中複寫。為了確保多個主機故障不會影響 AWS KMS 效能，AWS KMS 旨在為來自區域中任何 AZs 的客戶流量提供服務。

您對 KMS 金鑰的屬性或許可所做的變更將複製到區域中的所有主機，以確保區域中的任何主機都可以正確處理後續請求。使用 KMS 金鑰[進行密碼編譯操作](#)的請求會轉送至 AWS KMS 硬體安全模組 (HSMs) 的機群，其中任何一個都可以使用 KMS 金鑰執行操作。

多租用戶設計

的多租戶設計 AWS KMS 使其能夠實現 99.999% 的可用性 SLA，並維持高請求率，同時保護金鑰和資料的機密性。

部署多個完整性強制執行機制，以確保您為密碼編譯操作指定的 KMS 金鑰始終是使用的金鑰。

您的 KMS 金鑰的純文字金鑰資料受到廣泛保護。金鑰資料一經建立就會在 HSM 中加密，並且加密的金鑰資料會立即移至安全、低延遲的儲存中。系統會在 HSM 內擷取並解密已加密的金鑰，以便及時使用。純文字金鑰僅在完成密碼編譯操作所需的時間內保留在 HSM 記憶體中。然後在 HSM 中對其進行重新加密，並將加密的金鑰傳回至儲存體。純文字金鑰資料永遠不會離開 HSM；它永遠不會寫入持久性儲存。

中的彈性最佳實務 AWS KMS

若要最佳化 AWS KMS 資源的彈性，請考慮下列策略。

- 若要支援備份和災難復原策略，請考慮多區域金鑰，它們是在一個 AWS 區域 中建立的 KMS 金鑰，並僅複寫至您指定的區域。使用多區域金鑰，您可以在 AWS 區域（同一分割區內）之間移動加密的資源，而無須暴露純文字，並在需要時在其任何目的地區域中解密資源。相關的多區域金鑰是可相

互操作的，因為它們共用相同的金鑰資料和金鑰 ID，但它們具有獨立的金鑰策略來實現高解析度存取控制。如需詳細資訊，請參閱 [AWS KMS 中的多區域金鑰](#)。

- 若要保護多租戶服務中的金鑰 AWS KMS，請務必使用存取控制，包括[金鑰政策](#)和 [IAM 政策](#)。此外，您可以使用採用技術的 AWS KMS VPC 介面端點，將請求傳送至 AWS PrivateLink。當您這麼做時，Amazon VPC 和 之間的所有通訊 AWS KMS 都會完全在 AWS 網路中，使用僅限您 VPC 的專用 AWS KMS 端點進行。您可以透過使用 [VPC 端點政策](#) 建立額外授權層來進一步保護這些請求。如需詳細資訊，請參閱[透過 VPC 端點連接至 AWS KMS](#)。

中的基礎設施安全 AWS Key Management Service

作為受管服務，AWS Key Management Service (AWS KMS) 受到 [Amazon Web Services：安全程序概觀](#) 中所述的 AWS 全球網路安全程序的保護。

若要 AWS KMS 透過網路存取，您可以呼叫 AWS KMS API [AWS Key Management Service 參考中所述的 API](#) 操作。AWS KMS 需要 TLS 1.2，並建議在所有區域中使用 TLS 1.3。AWS KMS 也支援所有區域中 AWS KMS 服務端點的混合式後量子 TLS，中國區域除外。AWS KMS 不支援 FIPS 端點的混合後量子 TLS AWS GovCloud (US)。若要採用[標準 AWS KMS 端點](#)或 [AWS KMS FIPS 端點](#)，用戶端必須支援 TLS 1.2 或更新版本。用戶端也必須支援具備完美轉送私密 (PFS) 的密碼套件，例如臨時 Diffie-Hellman (DHE) 或橢圓曲線臨時 Diffie-Hellman (ECDHE)。現代系統 (如 Java 7 和更新版本) 大多會支援這些模式。

此外，請求必須使用存取金鑰 ID 和與 IAM 主體相關聯的私密存取金鑰來簽署。或者，您可以透過 [AWS Security Token Service](#) (AWS STS) 來產生暫時安全憑證來簽署請求。

您可以從任何網路位置呼叫這些 API 操作，但 AWS KMS 支援全域政策條件，可讓您根據來源 IP 地址、VPC 和 VPC 端點控制對 KMS 金鑰的存取。您可以在金鑰政策和 IAM 政策中使用這些條件索引鍵。不過，這些條件可能會 AWS 阻止 代表您使用 KMS 金鑰。如需詳細資訊，請參閱[AWS 全域條件索引鍵](#)。

例如，下列金鑰政策陳述式允許可擔任該KMSTestRole角色的使用者 AWS KMS key 將其用於指定的[密碼編譯操作](#)，除非來源 IP 地址是政策中指定的其中一個 IP 地址。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
```

```
"Principal": {"AWS":
"arn:aws:iam::111122223333:role/KMSTestRole"},
"Action": [
  "kms:Encrypt",
  "kms:Decrypt",
  "kms:ReEncrypt*",
  "kms:GenerateDataKey*",
  "kms:DescribeKey"
],
"Resource": "*",
"Condition": {
  "NotIpAddress": {
    "aws:SourceIp": [
      "192.0.2.0/24",
      "203.0.113.0/24"
    ]
  }
}
}
```

實體主機的隔離

AWS KMS 使用之實體基礎設施的安全受到 [Amazon Web Services：安全程序概觀](#) 之實體和環境安全一節中所述的控制。您可以在上一節所列的合規報告和第三方稽核問題清單中找到更多詳細資訊。

AWS KMS 由專用強化硬體安全模組 (HSMs) 支援，其設計具有特定控制，可抵禦實體攻擊。HSM 是不具有虛擬化層 (例如 Hypervisor) 的實體裝置，可在數個邏輯租用戶之間共用實體裝置。AWS KMS keys 金鑰材料只會存放在 HSM 的揮發性記憶體中，而且只有在使用 KMS 金鑰時才會存放。當 HSM 移出操作狀態時，此記憶體會被清除，包括預期和非預期的關機和重設。如需 AWS KMS HSMs 操作的詳細資訊，請參閱 [AWS Key Management Service 密碼編譯詳細資訊](#)。

配額

為了讓所有使用者都能 AWS KMS 回應和執行，AWS KMS 套用兩種類型的配額、資源配額和請求配額。每一種配額都會針對每個 AWS 帳戶的每個區域獨立計算。

所有 AWS KMS 配額都是可調整的，[隨需輪換資源配額](#)和[AWS CloudHSM 金鑰存放區請求配額](#)除外。若要請求提高配額，請參閱 [《Service Quotas 使用者指南》](#) 中的請求提高配額。若要請求減少配額、變更 Service Quotas 中未列出的配額，或在 AWS KMS 無法使用 AWS 區域 Service Quotas 的中變更配額，請造訪 [AWS 支援中心](#) 並建立案例。

主題

- [資源配額](#)
- [請求配額](#)
- [調節 AWS KMS 請求](#)

資源配額

AWS KMS 建立資源配額，以確保它可以為所有客戶提供快速且彈性的服務。有些資源配額僅適用於您建立的資源，但不適用於 AWS 服務為您建立的資源。您使用但並非位於您 AWS 帳戶中的資源 (例如 [AWS 擁有的金鑰](#)) 不會計入這些配額中。

如果您已超出資源限制，請求建立該類型的其他資源會產生 `LimitExceededException` 錯誤訊息。

所有 AWS KMS 資源配額都是可調整的，[隨需輪換資源配額](#)除外。若要請求提高配額，請參閱 [《Service Quotas 使用者指南》](#) 中的請求提高配額。若要請求減少配額、變更 Service Quotas 中未列出的配額，或在 AWS KMS 無法使用 AWS 區域 Service Quotas 的中變更配額，請造訪 [AWS 支援中心](#) 並建立案例。

下表列出並說明每個 AWS 帳戶 和 區域中 AWS KMS 的資源配額。

配額名稱	預設值	適用對象	可調整
AWS KMS keys	100,000	客戶自管金鑰	是
每個 KMS 金鑰的別名	50	客戶建立的別名	是
每個 KMS 金鑰的授予	50,000	客戶自管金鑰	是

配額名稱	預設值	適用對象	可調整
自訂金鑰存放區資源配額	10	AWS 帳戶 和 區域	是
隨需輪換	25	客戶自管金鑰	否

除了資源配額之外，AWS KMS 還會使用請求配額來確保服務的回應能力。如需詳細資訊，請參閱[the section called “請求配額”](#)。

AWS KMS keys : 100,000 個

在您 AWS 帳戶的每個區域，您最多可有 100,000 個[客戶受管金鑰](#)。此配額適用於所有 AWS 區域中所有的客戶受管金鑰，與其[金鑰規格](#)或[金鑰狀態](#)無關。每個 KMS 金鑰都會被視為一個資源。[AWS 受管金鑰](#) 和 [AWS 擁有的金鑰](#) 不會計入此配額。

每個 KMS 金鑰的別名：50

您最多可以將 50 個[別名](#)與每個[客戶受管金鑰](#)關聯。與 AWS 關聯的別名[AWS 受管金鑰](#)不會計入此配額。[建立](#)或[更新](#)別名時，您可能會遇到此配額。

Note

僅當 KMS 金鑰符合此配額時，[kms:ResourceAliases](#) 條件才會有效。如果 KMS 金鑰超過此配額，則會拒絕透過 `kms:ResourceAliases` 條件授權使用 KMS 金鑰的委託人存取 KMS 金鑰。如需詳細資訊，請參閱[因別名配額而拒絕存取](#)。

每個 KMS 金鑰配額的別名會取代每個區域配額的別名，限制 `an. AWS KMS has` 每個區域配額的別名總數 AWS 帳戶。

每個 KMS 金鑰的授予：50,000

每個[客戶受管金鑰](#)最多可有 50,000 個[授予](#)，包括與 [AWS KMS 整合的 AWS 服務](#)所建立的授予。此配額不適用於 [AWS 受管金鑰](#) 或 [AWS 擁有的金鑰](#)。

此配額的其中一個效果是，您無法同時執行超過 50,000 個使用相同 KMS 金鑰的授予授權操作。達到配額後，只有在作用中的授予已淘汰或撤銷時，您才能在 KMS 金鑰上建立新的授予。

例如，當您將 Amazon Elastic Block Store (Amazon EBS) 磁碟區連接到 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體時，磁碟區便會解密，讓您可以讀取。為了取得解密資料的許可，Amazon EBS 會為每個磁碟區建立授予。因此，如果您所有的 Amazon EBS 磁碟區都使用相同的 KMS 金鑰，您無法一次連接超過 50,000 個磁碟區。

自訂金鑰存放區資源配額：10

您可以在每個 AWS 帳戶和區域中建立最多 10 個 [自訂金鑰存放區](#)。如果您嘗試建立更多，則 [CreateCustomKeyStore](#) 操作會失敗。

此配額適用於每個帳戶和區域中自訂金鑰存放區的總數，包括 [AWS CloudHSM 金鑰存放區](#) 和 [外部金鑰存放區](#)，無論其連接狀態為何。

隨需輪換：25

您可以執行 [隨需金鑰輪換](#)，每個 KMS 金鑰最多 25 次。如果您嘗試執行更多隨需輪換，則 [RotateKeyOnDemand](#) 操作會失敗。

此配額不可調整。您無法透過使用 Service Quotas 或在其中建立案例來增加配額 AWS 支援。為了防止達到隨需輪換配額，我們建議您盡可能使用 [自動金鑰輪換](#)。

請求配額

AWS KMS 會為每秒請求的 API 操作數量建立配額。請求配額會因 API 操作、AWS 區域和其他因素而有所不同，例如 KMS 金鑰類型。當您超過 API 請求配額時，AWS KMS [會調節請求](#)。

所有 AWS KMS 請求配額都是可調整的，但 [AWS CloudHSM 金鑰存放區請求配額](#) 除外。若要請求提高配額，請參閱 [《Service Quotas 使用者指南》](#) 中的請求提高配額。若要請求減少配額、變更 Service Quotas 中未列出的配額，或在 AWS KMS 無法使用 AWS 區域 Service Quotas 的中變更配額，請造訪 [AWS 支援中心](#) 並建立案例。

如果正在超出 [GenerateDataKey](#) 操作的請求配額，請考慮使用 AWS Encryption SDK 的 [資料金鑰快速取](#) 功能。重複使用資料金鑰可能會降低請求的頻率 AWS KMS。

除了請求配額之外，AWS KMS 還會使用資源配額來確保所有使用者的容量。如需詳細資訊，請參閱 [資源配額](#)。

若要檢視請求率的趨勢，請使用 [Service Quotas 主控台](#)。您也可以建立 [Amazon CloudWatch](#) 警示，在您的請求率達到配額值的特定百分比時發出提醒。如需詳細資訊，請參閱 AWS 安全部落格中的 [使用 Service Quotas 和 Amazon CloudWatch 管理您的 AWS KMS API 請求率](#)。

主題

- [請求每個 AWS KMS API 操作的配額](#)
- [套用請求配額](#)
- [密碼編譯操作的共用配額](#)
- [代表您進行的 API 請求](#)
- [跨帳戶請求](#)
- [自訂金鑰存放區請求配額](#)

請求每個 AWS KMS API 操作的配額

此資料表列出 [Service Quotas](#) 配額代碼，以及每個 AWS KMS 請求配額的預設值。所有 AWS KMS 請求配額都是可調整的，但[AWS CloudHSM 金鑰存放區請求配額](#)除外。

Note

您可能需要水平或垂直捲動，才能查看此資料表中的所有資料。

配額名稱	預設值 (每秒請求數)
Cryptographic operations (symmetric) request rate 適用對象： • Decrypt • Encrypt • GenerateDataKey • GenerateDataKeyWithoutPlainText • GenerateMac • GenerateRandom • ReEncrypt • VerifyMac	這些共用配額會隨 AWS 區域 和請求中使用的 KMS 金鑰類型而有所不同。每個配額會分別計算。 • 10,000 (共用) • 下列區域的 20,000 (共用)： • 美國東部 (俄亥俄)，us-east-2 • 亞太區域 (新加坡) ap-southeast-1 • 亞太區域 (雪梨)，ap-southeast-2 • 亞太區域 (東京)，ap-northeast-1 • 歐洲 (法蘭克福)，eu-central-1 • 歐洲 (倫敦)，eu-west-2 • 下列區域的 100,000 (共用)： • 美國東部 (維吉尼亞北部)，us-east-1

配額名稱	預設值 (每秒請求數)
	• 美國西部 (奧勒岡) , us-west-2 • 歐洲 (愛爾蘭) , eu-west-1
Cryptographic operations (RSA) request rate 適用對象： • Decrypt • Encrypt • ReEncrypt • Sign • Verify	RSA KMS 金鑰 1 , 000 (共用)
Cryptographic operations (ML-DSA) request rate 適用對象： • Sign • Verify	ML-DSA KMS 金鑰 1 , 000 (共用)

配額名稱	預設值 (每秒請求數)
Cryptographic operations (ECC and SM2) request rate 適用對象： - Decrypt- 僅支援 SM2 (僅限中國區域) KMS 金鑰 - DeriveSharedSecret - Encrypt- 僅支援 SM2 (僅限中國區域) KMS 金鑰 - ReEncrypt - 僅支援 SM2 (僅限中國區域) KMS 金鑰 - Sign - Verify	1,000 (共用) 代表橢圓曲線 (ECC) 和 SM2 (僅限中國區域) KMS 金鑰
Custom key store request quotas 適用對象： - Decrypt - DeriveSharedSecret - Encrypt - GenerateDataKey - GenerateDataKeyWithoutPlaintext - GenerateRandom - ReEncrypt	會針對每個自訂金鑰存放區分別計算自訂金鑰存放區請求配額 - 每個 AWS CloudHSM 金鑰存放區 1,800 個 (共用) - 每個外部金鑰存放區 1800 個 (共用)
CancelKeyDeletion request rate	5
ConnectCustomKeyStore request rate	5
CreateAlias request rate	5

配額名稱	預設值 (每秒請求數)
CreateCustomKeyStore request rate	5
CreateGrant request rate	50
CreateKey request rate	5
DeleteAlias request rate	15
DeleteCustomKeyStore request rate	5
DeleteImportedKeyMaterial request rate	15
DescribeCustomKeyStores request rate	5
DescribeKey request rate	2000
DisableKey request rate	5
DisableKeyRotation request rate	5
DisconnectCustomKeyStore request rate	5
EnableKey request rate	5
EnableKeyRotation request rate	15
GenerateDataKeyPair (ECC_NIST_P256) request rate	100
適用對象：	
• GenerateDataKeyPair • GenerateDataKeyPairWithoutPlaintext	

配額名稱	預設值 (每秒請求數)
<code>GenerateDataKeyPair (ECC_NIST_P384) request rate</code> 適用對象： • <code>GenerateDataKeyPair</code> • <code>GenerateDataKeyPairWithoutPlaintext</code>	100
<code>GenerateDataKeyPair (ECC_NIST_P521) request rate</code> 適用對象： • <code>GenerateDataKeyPair</code> • <code>GenerateDataKeyPairWithoutPlaintext</code>	100
<code>GenerateDataKeyPair (ECC_SECG_P256K1) request rate</code> 適用對象： • <code>GenerateDataKeyPair</code> • <code>GenerateDataKeyPairWithoutPlaintext</code>	100
<code>GenerateDataKeyPair (ECC_NIST_EDWARDS25519) request rate</code> 適用對象： • <code>GenerateDataKeyPair</code> • <code>GenerateDataKeyPairWithoutPlaintext</code>	100

配額名稱	預設值 (每秒請求數)
GenerateDataKeyPair (RSA_2048) request rate 適用對象 : • GenerateDataKeyPair • GenerateDataKeyPairWithoutPlaintext	20
GenerateDataKeyPair (RSA_3072) request rate 適用對象 : • GenerateDataKeyPair • GenerateDataKeyPairWithoutPlaintext	4
GenerateDataKeyPair (RSA_4096) request rate 適用對象 : • GenerateDataKeyPair • GenerateDataKeyPairWithoutPlaintext	1
GenerateDataKeyPair (SM2 – China Regions only) request rate 適用對象 : • GenerateDataKeyPair • GenerateDataKeyPairWithoutPlaintext	25
GetKeyPolicy request rate	1000

配額名稱	預設值 (每秒請求數)
GetKeyRotationStatus request rate	1000
GetParametersForImport request rate	1
GetPublicKey request rate	2000
ImportKeyMaterial request rate	15
ListAliases request rate	500
ListGrants request rate	100
ListKeyPolicies request rate	100
ListKeys request rate	500
ListKeyRotations request rate	100
ListResourceTags request rate	2000
ListRetirableGrants request rate	100
PutKeyPolicy request rate	15
ReplicateKey request rate	5
ReplicateKey 操作在主要金鑰的區域中計為一個 ReplicateKey 請求，在複本區域中計為兩個 CreateKey 請求。其中一個 CreateKey 請求是試執行，以在建立金鑰之前偵測潛在的問題。	
RetireGrant request rate	50
RevokeGrant request rate	50
RotateKeyOnDemand request rate	5

配額名稱	預設值 (每秒請求數)
ScheduleKeyDeletion request rate	15
TagResource request rate	10
UntagResource request rate	5
UpdateAlias request rate	5
UpdateCustomKeyStore request rate	5
UpdateKeyDescription request rate	5
UpdatePrimaryRegion request rate	5
UpdatePrimaryRegion 操作會計為兩個 UpdatePrimaryRegion 請求；每兩個受影 響的區域中一個請求。	

套用請求配額

檢閱請求配額時，請注意以下資訊。

- 請求配額同時適用於[客戶受管金鑰](#)和 [AWS 受管金鑰](#)。使用 [AWS 擁有的金鑰](#) 不會計入您的請求配額 AWS 帳戶，即使它們用於保護您帳戶中的資源。
- 請求配額適用於傳送至 FIPS 端點和非 FIPS 端點的請求。如需 AWS KMS 服務端點的清單，請參閱中的[AWS Key Management Service 端點和配額](#) AWS 一般參考。
- 調節是以區域中所有類型 KMS 金鑰的所有請求為基礎。此總計包含來自 中所有主體的請求 AWS 帳戶，包括代表您來自 AWS 服務的請求。
- 每個請求配額都是獨立計算的。例如，[CreateKey](#) 操作的請求不會對 [CreateAlias](#) 操作的請求配額產生任何影響。如果您的 CreateAlias 請求遭到調節，您的 CreateKey 請求仍然可以成功完成。
- 雖然密碼編譯操作會共用配額，但是共用的配額會獨立於其他操作配額之外進行計算。例如，對 [Encrypt](#) 和 [Decrypt](#) 操作的呼叫會共用請求配額，但是該配額是獨立於管理操作 (例如 [EnableKey](#)) 的配額之外。例如，在歐洲 (倫敦) 區域，您可以在對稱 KMS 金鑰上執行 10,000 次密碼編譯操作，加上每秒 5 次 EnableKey 操作，而仍不會遭到調節。

密碼編譯操作的共用配額

AWS KMS [密碼編譯操作](#) 共用請求配額。您可以請求 KMS 金鑰所支援密碼編譯操作的任何組合，只要密碼編譯操作的總數不會超過該 KMS 金鑰類型的請求配額即可。例外狀況是 [GenerateDataKeyPair](#) 和 [GenerateDataKeyPairWithoutPlaintext](#)，它們共用個別的配額。

不同 KMS 金鑰類型的配額也會獨立計算。每個配額適用於 AWS 帳戶 和 區域中這些操作的所有請求，每個 1 秒間隔具有指定的金鑰類型。

- 密碼編譯操作 (對稱) 請求率是帳戶和區域中使用對稱 KMS 金鑰進行加密編譯操作的共用請求配額。此配額適用於具有對稱加密金鑰和 HMAC 金鑰 (兩者皆為對稱金鑰) 的密碼編譯操作。

例如，您可能在 中使用 [對稱 KMS 金鑰](#)，AWS 區域 共用配額為每秒 10,000 個請求。當您每秒發出 7,000 個 [GenerateDataKey](#) 請求和每秒發出 2,000 個 [解密](#) 請求時，AWS KMS 不會調節您的請求。但是，當您每秒發出 9,500 次 [GenerateDataKey](#) 請求及 1,000 次 [Encrypt](#) 請求時，AWS KMS 便會進行調節，因為其超過了共用配額。

[自訂金鑰存放區](#) 中 [對稱加密 KMS 金鑰](#) 的密碼編譯操作會計入帳戶的密碼編譯操作 (對稱) 請求率以及自訂金鑰存放區的 [自訂金鑰存放區請求配額](#)。

- 密碼編譯操作 (RSA) 請求率是使用 [RSA 非對稱 KMS 金鑰](#) 進行加密編譯操作的共用請求配額。

例如，請求配額為每秒 1,000 個操作，您可以使用可加密和 [解密](#) 的 RSA KMS 金鑰發出 400 個 [加密](#) 請求和 200 個解密請求，加上 250 個 [Sign](#) 請求和 150 個使用可簽署和驗證的 RSA KMS 金鑰 [驗證](#) 請求。

- 密碼編譯操作 (ECC) 請求率是使用 [橢圓曲線 \(ECC\) 非對稱 KMS 金鑰](#) 和 [SM 非對稱 KMS 金鑰](#) 進行密碼編譯操作的共用請求配額。

例如，如果請求配額為每秒 1,000 個操作，您可以使用可簽署和驗證的 ECC KMS 金鑰發出 400 個 Sign 請求和 200 個 Verify 請求，以及使用可簽署和驗證的 SM2 KMS 金鑰發出 250 個 [Sign](#) 請求和 150 個 [Verify](#) 請求。

- 自訂金鑰存放區請求配額是自訂金鑰存放區中 KMS 金鑰之密碼編譯操作的共用請求配額。此配額會針對每個自訂金鑰存放區分別計算。

[自訂金鑰存放區](#) 中 [對稱加密 KMS 金鑰](#) 的密碼編譯操作會計入帳戶的密碼編譯操作 (對稱) 請求率以及自訂金鑰存放區的 [自訂金鑰存放區請求配額](#)。

不同金鑰類型的配額也會獨立計算。例如，在亞太區域 (新加坡) 區域中，如果使用對稱和非對稱 KMS 金鑰，您最多可以使用對稱 KMS 金鑰 (包括 HMAC 金鑰) 每秒進行 10,000 次呼叫，加上使用 RSA 非

對稱 KMS 金鑰每秒進行最多 500 次額外的呼叫，再加上使用 ECC 型 KMS 金鑰每秒進行最多 300 次額外的請求。

代表您進行的 API 請求

您可以直接提出 API 請求，或使用整合式 AWS 服務 AWS KMS 代表您向 提出 API 請求。此配額同時適用於這兩種請求。

例如，您可能將資料存放在使用伺服器端加密搭配 KMS 金鑰 (SSE-KMS) 的 Amazon S3 上。每次上傳或下載使用 SSE-KMS 加密的 S3 物件時，Amazon S3 都會 AWS KMS 代表您向 發出 `GenerateDataKey` (上傳) 或 `Decrypt` (下載) 請求。這些請求會計入您的配額，因此如果您超過每秒 5,500 個 (或 10,000 或 50,000 個，視您的 而定 AWS 區域) 上傳或下載使用 SSE-KMS 加密的 S3 物件，則 會 AWS KMS 調節請求。

跨帳戶請求

當一個 中的應用程式 AWS 帳戶 使用不同帳戶擁有的 KMS 金鑰時，稱為跨帳戶請求。對於跨帳戶請求，AWS KMS 會調節發出請求的帳戶，而不是擁有 KMS 金鑰的帳戶。例如，如果帳戶 A 中的應用程式使用帳戶 B 中的 KMS 金鑰，使用該 KMS 金鑰就會套用帳戶 A 中的配額。

自訂金鑰存放區請求配額

AWS KMS 在 [自訂金鑰存放區](#) 中的 KMS 金鑰上維護 [密碼編譯操作](#) 的請求配額。會針對每個自訂金鑰存放區分別計算這些請求配額。

自訂金鑰存放區請求配額	每個自訂金鑰存放區的預設值 (每秒請求數)	可調整
AWS CloudHSM 金鑰存放區 請求配額	1800	否
外部金鑰存放區 請求配額	1800	否

Note

AWS KMS [自訂金鑰存放區請求配額](#) 不會出現在 Service Quotas 主控台中。您無法使用 Service Quotas API 操作來檢視或管理這些配額。

如果與 AWS CloudHSM 金鑰存放區相關聯的 AWS CloudHSM 叢集正在處理許多命令，包括與自訂金鑰存放區無關的命令，您可能會遇到 `ThrottlingException` 以 `lower-than-expected` 速率取得 AWS KMS。如果發生這種情況，請將請求率降低為 AWS KMS、減少不相關的負載，或為您的 AWS CloudHSM 金鑰存放區使用專用 AWS CloudHSM 叢集。

AWS KMS 會報告 [ExternalKeyStoreThrottle](#) CloudWatch 指標中外部金鑰存放區請求的限流。您可以使用此指標來檢視限流模式、建立警示以及調整外部金鑰存放區請求配額。

自訂金鑰存放區 KMS 金鑰的 [密碼編譯操作](#) 請求會計入兩個配額：

- 密碼編譯操作 (對稱) 請求率配額 (每個帳戶)

自訂金鑰存放區中 KMS 金鑰的密碼編譯操作請求會計入每個 AWS 帳戶 和區域的 `Cryptographic operations (symmetric) request rate` 配額。例如，在美國東部 (維吉尼亞北部) (`us-east-1`)，每個對對稱加密 KMS 金鑰每秒 AWS 帳戶 最多可以有 100,000 個請求，包括在自訂金鑰存放區中使用 KMS 金鑰的請求。

- 自訂金鑰存放區請求配額 (每個自訂金鑰存放區)

自訂金鑰存放區中 KMS 金鑰的密碼編譯操作請求也會計入每秒 1,800 次 `Custom key store request quota` 操作中。會針對每個自訂金鑰存放區分別計算這些配額。它們可能包括來自多個 AWS 帳戶 在自訂金鑰存放區中使用 KMS 金鑰的請求。

例如，美國東部 (維吉尼亞北部) (`us-east-1`) 區域中自訂金鑰存放區 (任一類型) 中 KMS 金鑰的 [加密](#) 操作會計入其 `Cryptographic operations (symmetric) request rate` 帳戶和區域的帳戶層級配額 (每秒 100,000 個請求)，以及自訂金鑰存放區的 `Custom key store request quota` (每秒 1,800 個請求)。不過，針對自訂金鑰存放區中 KMS 金鑰的管理操作 (例如 [PutKeyPolicy](#)) 的請求只會套用至其帳戶層級配額 (每秒 15 個請求)。

調節 AWS KMS 請求

為了確保 AWS KMS 可以為所有客戶的 API 請求提供快速且可靠的回應，它會調節超過特定界限的 API 請求。

當 AWS KMS 拒絕原本可能有效的請求，並傳回如下的 `ThrottlingException` 錯誤時，就會發生限流。

```
You have exceeded the rate at which you may call KMS. Reduce the frequency of your calls.
```

(Service: AWSKMS; Status Code: 400; Error Code: ThrottlingException; Request ID: <ID>)

AWS KMS 針對下列條件調節請求。

- 每秒請求的速率超過帳戶和區域的 AWS KMS [請求配額](#)。

例如，如果您帳戶中的使用者在一秒內提交 1000 個 DescribeKey 請求，會 AWS KMS 調節該秒中的所有後續 DescribeKey 請求。

若要回應調節，請使用 [退避與重試策略](#)。此策略會自動針對 AWS SDKs 中的 HTTP 400 錯誤實作。

- 變更相同 KMS 金鑰狀態之高載或持續高水準的請求率。這種情況通常稱為「快速鍵」。

例如，如果您帳戶中的應用程式傳送相同 KMS 金鑰的持久性 EnableKey 和 DisableKey 請求，會 AWS KMS 調節請求。即使對於 EnableKey 和 DisableKey 操作，請求未超過請求每秒次數的請求限制，也會發生此調節。

若要回應調節，請調整您的應用程式邏輯，讓其只提出必要的請求，或合併多個函數的請求。

- 當與金鑰 [AWS CloudHSM 存放區](#) 相關聯的叢集正在處理許多命令，包括與 AWS CloudHSM 金鑰存放區無關的命令時 AWS CloudHSM，金鑰存放區中 KMS AWS CloudHSM 金鑰的操作請求可能會 lower-than-expected 速率調節。

(當 AWS CloudHSM 叢集沒有可用的 PKCS #11 工作階段時，AWS KMS 不會再調節對金鑰存放區中 KMS AWS CloudHSM 金鑰的操作請求。相反地，它會擲回 KMSInternalException，並建議您重試請求。)

若要檢視請求率的趨勢，請使用 [Service Quotas 主控台](#)。您也可以建立 [Amazon CloudWatch](#) 警示，在您的請求率達到配額值的特定百分比時發出提醒。如需詳細資訊，請參閱 AWS 安全部落格中的 [使用 Service Quotas 和 Amazon CloudWatch 管理您的 AWS KMS API 請求率](#)。

所有 AWS KMS 配額都是可調整的，[隨需輪換資源配額](#) 和 [AWS CloudHSM 金鑰存放區請求配額](#) 除外。若要請求提高配額，請參閱 [《Service Quotas 使用者指南》](#) 中的請求提高配額。若要請求減少配額、變更 Service Quotas 中未列出的配額，或在 AWS KMS 無法使用 AWS 區域 Service Quotas 的中變更配額，請造訪 [AWS 支援 中心](#) 並建立案例。

Note

AWS KMS [自訂金鑰存放區請求配額](#) 不會出現在 Service Quotas 主控台中。您無法使用 Service Quotas API 操作來檢視或管理這些配額。

AWS KMS 使用 AWS SDKs 程式碼範例

下列程式碼範例示範如何 AWS KMS 搭配 AWS 軟體開發套件 (SDK) 使用。

基本概念是程式碼範例，這些範例說明如何在服務內執行基本操作。

Actions 是大型程式的程式碼摘錄，必須在內容中執行。雖然動作會告訴您如何呼叫個別服務函數，但您可以在其相關情境中查看內容中的動作。

案例是向您展示如何呼叫服務中的多個函數或與其他 AWS 服務組合來完成特定任務的程式碼範例。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

程式碼範例

- [AWS KMS 使用 AWS SDKs 的基本範例](#)
 - [您好 AWS Key Management Service](#)
 - [AWS KMS 使用 AWS SDK 了解 的基本概念](#)
 - [AWS KMS 使用 AWS SDKs 的動作](#)
 - [CreateAlias 搭配 AWS SDK 或 CLI 使用](#)
 - [CreateGrant 搭配 AWS SDK 或 CLI 使用](#)
 - [CreateKey 搭配 AWS SDK 或 CLI 使用](#)
 - [Decrypt 搭配 AWS SDK 或 CLI 使用](#)
 - [DeleteAlias 搭配 AWS SDK 或 CLI 使用](#)
 - [DescribeKey 搭配 AWS SDK 或 CLI 使用](#)
 - [DisableKey 搭配 AWS SDK 或 CLI 使用](#)
 - [EnableKey 搭配 AWS SDK 或 CLI 使用](#)
 - [EnableKeyRotation 搭配 AWS SDK 或 CLI 使用](#)
 - [Encrypt 搭配 AWS SDK 或 CLI 使用](#)
 - [GenerateDataKey 搭配 AWS SDK 或 CLI 使用](#)
 - [GenerateDataKeyWithoutPlaintext 搭配 AWS SDK 或 CLI 使用](#)
 - [GenerateRandom 搭配 AWS SDK 或 CLI 使用](#)
 - [GetKeyPolicy 搭配 AWS SDK 或 CLI 使用](#)
 - [ListAliases 搭配 AWS SDK 或 CLI 使用](#)

- [ListGrants 搭配 AWS SDK 或 CLI 使用](#)
- [ListKeyPolicies 搭配 AWS SDK 或 CLI 使用](#)
- [ListKeys 搭配 AWS SDK 或 CLI 使用](#)
- [PutKeyPolicy 搭配 AWS SDK 或 CLI 使用](#)
- [ReEncrypt 搭配 AWS SDK 或 CLI 使用](#)
- [RetireGrant 搭配 AWS SDK 或 CLI 使用](#)
- [RevokeGrant 搭配 AWS SDK 或 CLI 使用](#)
- [ScheduleKeyDeletion 搭配 AWS SDK 或 CLI 使用](#)
- [Sign 搭配 AWS SDK 或 CLI 使用](#)
- [TagResource 搭配 AWS SDK 或 CLI 使用](#)
- [UpdateAlias 搭配 AWS SDK 或 CLI 使用](#)
- [Verify 搭配 AWS SDK 或 CLI 使用](#)
- [AWS KMS 使用 AWS SDKs 案例](#)
 - [使用 AWS Command Line Interface v2 使用 DynamoDB 資料表加密](#)

AWS KMS 使用 AWS SDKs 的基本範例

下列程式碼範例示範如何 AWS Key Management Service 搭配 AWS SDKs 使用的基本概念。

範例

- [您好 AWS Key Management Service](#)
- [AWS KMS 使用 AWS SDK 了解的基本概念](#)
- [AWS KMS 使用 AWS SDKs 的動作](#)
 - [CreateAlias 搭配 AWS SDK 或 CLI 使用](#)
 - [CreateGrant 搭配 AWS SDK 或 CLI 使用](#)
 - [CreateKey 搭配 AWS SDK 或 CLI 使用](#)
 - [Decrypt 搭配 AWS SDK 或 CLI 使用](#)
 - [DeleteAlias 搭配 AWS SDK 或 CLI 使用](#)
 - [DescribeKey 搭配 AWS SDK 或 CLI 使用](#)
 - [DisableKey 搭配 AWS SDK 或 CLI 使用](#)
 - [EnableKey 搭配 AWS SDK 或 CLI 使用](#)

- [EnableKeyRotation 搭配 AWS SDK 或 CLI 使用](#)
- [Encrypt 搭配 AWS SDK 或 CLI 使用](#)
- [GenerateDataKey 搭配 AWS SDK 或 CLI 使用](#)
- [GenerateDataKeyWithoutPlaintext 搭配 AWS SDK 或 CLI 使用](#)
- [GenerateRandom 搭配 AWS SDK 或 CLI 使用](#)
- [GetKeyPolicy 搭配 AWS SDK 或 CLI 使用](#)
- [ListAliases 搭配 AWS SDK 或 CLI 使用](#)
- [ListGrants 搭配 AWS SDK 或 CLI 使用](#)
- [ListKeyPolicies 搭配 AWS SDK 或 CLI 使用](#)
- [ListKeys 搭配 AWS SDK 或 CLI 使用](#)
- [PutKeyPolicy 搭配 AWS SDK 或 CLI 使用](#)
- [ReEncrypt 搭配 AWS SDK 或 CLI 使用](#)
- [RetireGrant 搭配 AWS SDK 或 CLI 使用](#)
- [RevokeGrant 搭配 AWS SDK 或 CLI 使用](#)
- [ScheduleKeyDeletion 搭配 AWS SDK 或 CLI 使用](#)
- [Sign 搭配 AWS SDK 或 CLI 使用](#)
- [TagResource 搭配 AWS SDK 或 CLI 使用](#)
- [UpdateAlias 搭配 AWS SDK 或 CLI 使用](#)
- [Verify 搭配 AWS SDK 或 CLI 使用](#)

您好 AWS Key Management Service

下列程式碼範例示範如何開始使用 AWS Key Management Service。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.services.kms.KmsAsyncClient;
import software.amazon.awssdk.services.kms.model.ListKeysRequest;
import software.amazon.awssdk.services.kms.paginators.ListKeysPublisher;
import java.util.concurrent.CompletableFuture;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class HelloKMS {
    public static void main(String[] args) {
        listAllKeys();
    }

    public static void listAllKeys() {
        KmsAsyncClient kmsAsyncClient = KmsAsyncClient.builder()
            .build();
        ListKeysRequest listKeysRequest = ListKeysRequest.builder()
            .limit(15)
            .build();

        /**
         * The `subscribe` method is required when using paginator methods in the
         * AWS SDK
         * because paginator methods return an instance of a `ListKeysPublisher`,
         * which is
         * based on a reactive stream. This allows asynchronous retrieval of
         * paginated
         * results as they become available. By subscribing to the stream, we can
         * process
         * each page of results as they are emitted.
         */
        ListKeysPublisher keysPublisher =
            kmsAsyncClient.listKeysPaginator(listKeysRequest);
        CompletableFuture<Void> future = keysPublisher
            .subscribe(r -> r.keys().forEach(key ->
                System.out.println("The key ARN is: " + key.keyArn() + ". The key
                Id is: " + key.keyId())))
```



```
        .whenComplete((result, exception) -> {
            if (exception != null) {
                System.err.println("Error occurred: " +
exception.getMessage());
            } else {
                System.out.println("Successfully listed all keys.");
            }
        });

        try {
            future.join();
        } catch (Exception e) {
            System.err.println("Failed to list keys: " + e.getMessage());
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ListKeys](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
include "vendor/autoload.php";

use Aws\Kms\KmsClient;

echo "This file shows how to connect to the KmsClient, uses a paginator to get
the keys for the account, and lists the KeyIds for up to 10 keys.\n";

$client = new KmsClient([]);

$pageLength = 10; // Change this value to change the number of records shown, or
to break up the result into pages.
```

```
$keys = [];  
$keysPaginator = $client->getPaginator("ListKeys", ['Limit' => $pageLength]);  
foreach($keysPaginator as $page){  
    foreach($page['Keys'] as $index => $key){  
        echo "The $index index Key's ID is: {$key['KeyId']}\n";  
    }  
    echo "End of page one of results. Alter the \$pageLength variable to see more  
results.\n";  
    break;  
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [ListKeys](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    oo_result = lo_kms->listkeys( ).  
    MESSAGE 'Retrieved KMS keys list.' TYPE 'I'.  
CATCH /aws1/cx_kmskmsinternalex.  
    MESSAGE 'An internal error occurred.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [ListKeys](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

AWS KMS 使用 AWS SDK 了解 的基本概念

下列程式碼範例示範如何：

- 建立 KMS 金鑰。
- 列出您帳戶的 KMS 金鑰，並取得其詳細資訊。
- 啟用和停用 KMS 金鑰。
- 產生可用於用戶端加密的對稱資料金鑰。
- 產生用於數位簽署資料的非對稱金鑰。
- 標籤索引鍵。
- 刪除 KMS 金鑰。

Java

適用於 Java 2.x 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

在命令提示中執行案例。

```
import software.amazon.awssdk.core.SdkBytes;
import software.amazon.awssdk.regions.Region;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.kms.model.AlreadyExistsException;
import software.amazon.awssdk.services.kms.model.DisabledException;
import software.amazon.awssdk.services.kms.model.EnableKeyRotationResponse;
import software.amazon.awssdk.services.kms.model.KmsException;
import software.amazon.awssdk.services.kms.model.NotFoundException;
import software.amazon.awssdk.services.kms.model.RevokeGrantResponse;
import java.util.List;
import java.util.Scanner;
import java.util.concurrent.CompletableFuture;
import java.util.concurrent.CompletionException;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */

public class KMSScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
    "-");
    private static String accountId = "";

    private static final Logger logger =
    LoggerFactory.getLogger(KMSScenario.class);

    static KMSActions kmsActions = new KMSActions();

    static Scanner scanner = new Scanner(System.in);

    static String aliasName = "alias/dev-encryption-key";

    public static void main(String[] args) {
        final String usage = ""
            Usage: <granteePrincipal>

            Where:
                granteePrincipal - The principal (user, service account, or group)
to whom the grant or permission is being given.
            "";

        if (args.length != 1) {
            logger.info(usage);
            return;
        }
        String granteePrincipal = args[0];
        String policyName = "default";

        accountId = kmsActions.getAccountId();
        String keyDesc = "Created by the AWS KMS API";

        logger.info(DASHES);
    }
}
```

```

        logger.info("""
            Welcome to the AWS Key Management SDK Basics scenario.

            This program demonstrates how to interact with AWS Key Management
            using the AWS SDK for Java (v2).
            The AWS Key Management Service (KMS) is a secure and highly available
            service that allows you to create
            and manage AWS KMS keys and control their use across a wide range of
            AWS services and applications.
            KMS provides a centralized and unified approach to managing
            encryption keys, making it easier to meet your
            data protection and regulatory compliance requirements.

            This Basics scenario creates two key types:

            - A symmetric encryption key is used to encrypt and decrypt data.
            - An asymmetric key used to digitally sign data.

            Let's get started...
            """);
        waitForInputToContinue(scanner);

        try {
            // Run the methods that belong to this scenario.
            String targetKeyId = runScenario(granteePrincipal, keyDesc, policyName);
            requestDeleteResources(aliasName, targetKeyId);

        } catch (Throwable rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
        }
    }

    private static String runScenario(String granteePrincipal, String keyDesc,
String policyName) throws Throwable {
        logger.info(DASHES);
        logger.info("1. Create a symmetric KMS key\n");
        logger.info("First, the program will creates a symmetric KMS key that you
can used to encrypt and decrypt data.");
    }

```

```

        waitForInputToContinue(scanner);
        String targetKeyId;
        try {
            CompletableFuture<String> futureKeyId =
kmsActions.createKeyAsync(keyDesc);
            targetKeyId = futureKeyId.join();
            logger.info("A symmetric key was successfully created " +
targetKeyId);

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
            throw cause;
        }
        waitForInputToContinue(scanner);

        logger.info(DASHES);
        logger.info("""
            2. Enable a KMS key

            By default, when the SDK creates an AWS key, it is enabled. The next
            bit of code checks to
            determine if the key is enabled.
            """);
        waitForInputToContinue(scanner);
        boolean isEnabled;
        try {
            CompletableFuture<Boolean> futureIsKeyEnabled =
kmsActions.isKeyEnabledAsync(targetKeyId);
            isEnabled = futureIsKeyEnabled.join();
            logger.info("Is the key enabled? {}", isEnabled);

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
        }
    }
}

```

```

        }
        throw cause;
    }

    if (!isEnabled)
        try {
            CompletableFuture<Void> future =
kmsActions.enableKeyAsync(targetKeyId);
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error
code {}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " +
rt.getMessage());
            }
            throw cause;
        }
    waitForInputToContinue(scanner);

    logger.info(DASHES);
    logger.info("3. Encrypt data using the symmetric KMS key");
    String plaintext = "Hello, AWS KMS!";
    logger.info("""
        One of the main uses of symmetric keys is to encrypt and decrypt
data.

        Next, the code encrypts the string {} with the SYMMETRIC_DEFAULT
encryption algorithm.
        """, plaintext);
    waitForInputToContinue(scanner);
    SdkBytes encryptedData;
    try {
        CompletableFuture<SdkBytes> future =
kmsActions.encryptDataAsync(targetKeyId, plaintext);
        encryptedData = future.join();

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof DisabledException kmsDisabledEx) {

```

```

        logger.info("KMS error occurred due to a disabled
key: Error message: {}, Error code {}", kmsDisabledEx.getMessage(),
kmsDisabledEx.awsErrorDetails().errorCode());
    } else {
        logger.info("An unexpected error occurred: " + rt.getMessage());
    }
    deleteKey(targetKeyId);
    throw cause;
}
waitForInputToContinue(scanner);

logger.info(DASHES);
logger.info("4. Create an alias");
logger.info("""

    The alias name should be prefixed with 'alias/'.
    The default, 'alias/dev-encryption-key'.
    """);
waitForInputToContinue(scanner);

try {
    CompletableFuture<Void> future =
kmsActions.createCustomAliasAsync(targetKeyId, aliasName);
    future.join();

} catch (RuntimeException rt) {
    Throwable cause = rt.getCause();
    if (cause instanceof AlreadyExistsException kmsExistsEx) {
        if (kmsExistsEx.getMessage().contains("already exists")) {
            logger.info("The alias '" + aliasName + "' already exists.
Moving on...");
        }
    } else {
        logger.error("An unexpected error occurred: " + rt.getMessage(),
rt);

        deleteKey(targetKeyId);
        throw cause;
    }
}
waitForInputToContinue(scanner);

logger.info(DASHES);
logger.info("5. List all of your aliases");
waitForInputToContinue(scanner);

```



```

    try {
        CompletableFuture<Object> future = kmsActions.listAllAliasesAsync();
        future.join();

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof KmsException kmsEx) {
            logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
        } else {
            logger.info("An unexpected error occurred: " + rt.getMessage());
        }
        deleteAliasName(aliasName);
        deleteKey(targetKeyId);
        throw cause;
    }
    waitForInputToContinue(scanner);

    logger.info(DASHES);
    logger.info("6. Enable automatic rotation of the KMS key");
    logger.info("""

        By default, when the SDK enables automatic rotation of a KMS key,
        KMS rotates the key material of the KMS key one year (approximately
365 days) from the enable date and every year
        thereafter.
        """);
    waitForInputToContinue(scanner);
    try {
        CompletableFuture<EnableKeyRotationResponse> future =
kmsActions.enableKeyRotationAsync(targetKeyId);
        future.join();

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof KmsException kmsEx) {
            logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
        } else {
            logger.info("An unexpected error occurred: " + rt.getMessage());
        }
        deleteAliasName(aliasName);
        deleteKey(targetKeyId);
        throw cause;
    }

```

```

    }
    waitForInputToContinue(scanner);

    logger.info(DASHES);
    logger.info("""
        7. Create a grant

        A grant is a policy instrument that allows Amazon Web Services
principals to use KMS keys.
        It also can allow them to view a KMS key (DescribeKey) and create and
manage grants.
        When authorizing access to a KMS key, grants are considered along
with key policies and IAM policies.
        """);

    waitForInputToContinue(scanner);
    String grantId = null;
    try {
        CompletableFuture<String> futureGrantId =
kmsActions.grantKeyAsync(targetKeyId, granteePrincipal);
        grantId = futureGrantId.join();

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof KmsException kmsEx) {
            logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
        } else {
            logger.info("An unexpected error occurred: " + rt.getMessage());
        }
        deleteKey(targetKeyId);
        throw cause;
    }
    waitForInputToContinue(scanner);
    logger.info(DASHES);

    logger.info(DASHES);
    logger.info("8. List grants for the KMS key");
    waitForInputToContinue(scanner);
    try {
        CompletableFuture<Object> future =
kmsActions.displayGrantIdsAsync(targetKeyId);
        future.join();
    }

```

```

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof KmsException kmsEx) {
            logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
        } else {
            logger.info("An unexpected error occurred: " + rt.getMessage());
        }
        deleteAliasName(aliasName);
        deleteKey(targetKeyId);
        throw cause;
    }
    waitForInputToContinue(scanner);

    logger.info(DASHES);
    logger.info("9. Revoke the grant");
    logger.info("""
        The revocation of a grant immediately removes the permissions and
        access that the grant had provided.
        This means that any principal (user, role, or service) that was
        granted access to perform specific
        KMS operations on a KMS key will no longer be able to perform those
        operations.
        """);
    waitForInputToContinue(scanner);
    try {
        CompletableFuture<RevokeGrantResponse> future =
kmsActions.revokeKeyGrantAsync(targetKeyId, grantId);
        future.join();

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof KmsException kmsEx) {
            if (kmsEx.getMessage().contains("Grant does not exist")) {
                logger.info("The grant ID '" + grantId + "' does not exist.
Moving on...");
            } else {
                logger.info("KMS error occurred: Error message: {}, Error
code {}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
                throw cause;
            }
        } else {
            logger.info("An unexpected error occurred: " + rt.getMessage());
            deleteAliasName(aliasName);

```

```

        deleteKey(targetKeyId);
        throw cause;
    }
}
waitForInputToContinue(scanner);

logger.info(DASHES);
logger.info("10. Decrypt the data\n");
logger.info("""
    Lets decrypt the data that was encrypted in an early step.
    The code uses the same key to decrypt the string that we encrypted
earlier in the program.
    """);
waitForInputToContinue(scanner);
String decryptedData = "";
try {
    CompletableFuture<String> future =
kmsActions.decryptDataAsync(encryptedData, targetKeyId);
    decryptedData = future.join();
    logger.info("Decrypted data: " + decryptedData);

} catch (RuntimeException rt) {
    Throwable cause = rt.getCause();
    if (cause instanceof KmsException kmsEx) {
        logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
    } else {
        logger.info("An unexpected error occurred: " + rt.getMessage());
    }
    deleteAliasName(aliasName);
    deleteKey(targetKeyId);
    throw cause;
}
logger.info("Decrypted text is: " + decryptedData);
waitForInputToContinue(scanner);

logger.info(DASHES);
logger.info("11. Replace a key policy\n");
logger.info("""
    A key policy is a resource policy for a KMS key. Key policies are the
primary way to control
    access to KMS keys. Every KMS key must have exactly one key policy.
The statements in the key policy

```

determine who has permission to use the KMS key and how they can use it.

You can also use IAM policies and grants to control access to the KMS key, but every KMS key must have a key policy.

By default, when you create a key by using the SDK, a policy is created that gives the AWS account that owns the KMS key full access to the KMS key.

Let's try to replace the automatically created policy with the following policy.

```
        "Version": "2012-10-17",
        "Statement": [{
            "Effect": "Allow",
            "Principal": {"AWS": "arn:aws:iam::000000000000:root"},
            "Action": "kms:*",
            "Resource": "*"
        }]
    """);

waitForInputToContinue(scanner);
try {
    CompletableFuture<Boolean> future =
kmsActions.replacePolicyAsync(targetKeyId, policyName, accountId);
    boolean success = future.join();
    if (success) {
        logger.info("Key policy replacement succeeded.");
    } else {
        logger.error("Key policy replacement failed.");
    }
} catch (RuntimeException rt) {
    Throwable cause = rt.getCause();
    if (cause instanceof KmsException kmsEx) {
        logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
    } else {
        logger.info("An unexpected error occurred: " + rt.getMessage());
    }
    deleteAliasName(aliasName);
    deleteKey(targetKeyId);
}
```

```

        throw cause;
    }
    waitForInputToContinue(scanner);

    logger.info(DASHES);
    logger.info("12. Get the key policy\n");
    logger.info("The next bit of code that runs gets the key policy to make
sure it exists.");
    waitForInputToContinue(scanner);
    try {
        CompletableFuture<String> future =
kmsActions.getKeyPolicyAsync(targetKeyId, policyName);
        String policy = future.join();
        if (!policy.isEmpty()) {
            logger.info("Retrieved policy: " + policy);
        }

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof KmsException kmsEx) {
            logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
        } else {
            logger.info("An unexpected error occurred: " + rt.getMessage());
        }
        deleteAliasName(aliasName);
        deleteKey(targetKeyId);
        throw cause;
    }
    waitForInputToContinue(scanner);

    logger.info(DASHES);
    logger.info("13. Create an asymmetric KMS key and sign your data\n");
    logger.info(""""
        Signing your data with an AWS key can provide several benefits that
make it an attractive option
        for your data signing needs. By using an AWS KMS key, you can
leverage the
        security controls and compliance features provided by AWS,
        which can help you meet various regulatory requirements and enhance
the overall security posture
        of your organization.
        """);
    waitForInputToContinue(scanner);

```

```

    try {
        CompletableFuture<Boolean> future = kmsActions.signVerifyDataAsync();
        boolean success = future.join();
        if (success) {
            logger.info("Sign and verify data operation succeeded.");
        } else {
            logger.error("Sign and verify data operation failed.");
        }
    }

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof KmsException kmsEx) {
            logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
        } else {
            logger.info("An unexpected error occurred: " + rt.getMessage());
        }
        deleteAliasName(aliasName);
        deleteKey(targetKeyId);
        throw cause;
    }
    waitForInputToContinue(scanner);

    logger.info(DASHES);
    logger.info("14. Tag your symmetric KMS Key\n");
    logger.info("""
        By using tags, you can improve the overall management, security, and
governance of your
        KMS keys, making it easier to organize, track, and control access to
your encrypted data within
        your AWS environment
        """);
    waitForInputToContinue(scanner);
    try {
        CompletableFuture<Void> future =
kmsActions.tagKMSKeyAsync(targetKeyId);
        future.join();

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof KmsException kmsEx) {
            logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
        } else {

```

```

        logger.info("An unexpected error occurred: " + rt.getMessage());
    }
    deleteAliasName(aliasName);
    deleteKey(targetKeyId);
    throw cause;
}
waitForInputToContinue(scanner);
return targetKeyId;
}

// Deletes KMS resources with user input.
private static void requestDeleteResources(String aliasName, String
targetKeyId) {
    logger.info(DASHES);
    logger.info("15. Schedule the deletion of the KMS key\n");
    logger.info("""
        By default, KMS applies a waiting period of 30 days,
        but you can specify a waiting period of 7-30 days. When this
operation is successful,
        the key state of the KMS key changes to PendingDeletion and the key
can't be used in any
        cryptographic operations. It remains in this state for the duration
of the waiting period.

        Deleting a KMS key is a destructive and potentially dangerous
operation. When a KMS key is deleted,
        all data that was encrypted under the KMS key is unrecoverable.
        """);
    logger.info("Would you like to delete the Key Management resources? (y/
n)");
    String delAns = scanner.nextLine().trim();
    if (delAns.equalsIgnoreCase("y")) {
        logger.info("You selected to delete the AWS KMS resources.");
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<Void> future =
kmsActions.deleteSpecificAliasAsync(aliasName);
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error
code {}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());

```



```
        } else {
            logger.info("An unexpected error occurred: " +
rt.getMessage());
        }
    }
    waitForInputToContinue(scanner);
    try {
        CompletableFuture<Void> future =
kmsActions.disableKeyAsync(targetKeyId);
        future.join();

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof KmsException kmsEx) {
            logger.info("KMS error occurred: Error message: {}, Error
code {}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
        } else {
            logger.info("An unexpected error occurred: " +
rt.getMessage());
        }
    }

    try {
        CompletableFuture<Void> future =
kmsActions.deleteKeyAsync(targetKeyId);
        future.join();

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof KmsException kmsEx) {
            logger.info("KMS error occurred: Error message: {}, Error
code {}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
        } else {
            logger.info("An unexpected error occurred: " +
rt.getMessage());
        }
    }

    } else {
        logger.info("The Key Management resources will not be deleted");
    }

    logger.info(DASHES);
    logger.info("This concludes the AWS Key Management SDK scenario");
```

```
        logger.info(DASHES);
    }

    // This method is invoked from Exceptions to clean up the resources.
    private static void deleteKey(String targetKeyId) {
        try {
            CompletableFuture<Void> future =
kmsActions.disableKeyAsync(targetKeyId);
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
        }

        try {
            CompletableFuture<Void> future =
kmsActions.deleteKeyAsync(targetKeyId);
            future.join();

        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof KmsException kmsEx) {
                logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
            } else {
                logger.info("An unexpected error occurred: " + rt.getMessage());
            }
        }
    }

    // This method is invoked from Exceptions to clean up the resources.
    private static void deleteAliasName(String aliasName) {
        try {
            CompletableFuture<Void> future =
kmsActions.deleteSpecificAliasAsync(aliasName);
            future.join();

        } catch (RuntimeException rt) {
```

```

        Throwable cause = rt.getCause();
        if (cause instanceof KmsException kmsEx) {
            logger.info("KMS error occurred: Error message: {}, Error code
{}", kmsEx.getMessage(), kmsEx.awsErrorDetails().errorCode());
        } else {
            logger.info("An unexpected error occurred: " + rt.getMessage());
        }
    }
}

private static void waitForInputToContinue(Scanner scanner) {
    while (true) {
        logger.info("");
        logger.info("Enter 'c' followed by <ENTER> to continue:");
        String input = scanner.nextLine();

        if (input.trim().equalsIgnoreCase("c")) {
            logger.info("Continuing with the program...");
            logger.info("");
            break;
        } else {
            // Handle invalid input.
            logger.info("Invalid input. Please try again.");
        }
    }
}
}
}

```

定義包裝 KMS 動作的類別。

```

public class KMSActions {
    private static final Logger logger =
LoggerFactory.getLogger(KMSActions.class);
    private static KmsAsyncClient kmsAsyncClient;

    /**
     * Retrieves an asynchronous AWS Key Management Service (KMS) client.
     * <p>
     * This method creates and returns a singleton instance of the KMS async
     client, with the following configurations:
     * <ul>
     * <li>Max concurrency: 100</li>

```

```

*   <li>Connection timeout: 60 seconds</li>
*   <li>Read timeout: 60 seconds</li>
*   <li>Write timeout: 60 seconds</li>
*   <li>API call timeout: 2 minutes</li>
*   <li>API call attempt timeout: 90 seconds</li>
*   <li>Retry policy: up to 3 retries</li>
*   <li>Credentials provider: environment variable credentials provider</li>
* </ul>
* <p>
*   If the client instance has already been created, it is returned instead of
creating a new one.
*
*   @return the KMS async client instance
*/
private static KmsAsyncClient getAsyncClient() {
    if (kmsAsyncClient == null) {
        SdkAsyncHttpClient httpClient = NettyNioAsyncHttpClient.builder()
            .maxConcurrency(100)
            .connectionTimeout(Duration.ofSeconds(60))
            .readTimeout(Duration.ofSeconds(60))
            .writeTimeout(Duration.ofSeconds(60))
            .build();

        ClientOverrideConfiguration overrideConfig =
ClientOverrideConfiguration.builder()
            .apiCallTimeout(Duration.ofMinutes(2))
            .apiCallAttemptTimeout(Duration.ofSeconds(90))
            .retryPolicy(RetryPolicy.builder()
                .numRetries(3)
                .build())
            .build();

        kmsAsyncClient = KmsAsyncClient.builder()
            .httpClient(httpClient)
            .overrideConfiguration(overrideConfig)
            .build();
    }
    return kmsAsyncClient;
}

/**
*   Creates a new symmetric encryption key asynchronously.
*
*   @param keyDesc the description of the key to be created

```

```

    * @return a {@link CompletableFuture} that completes with the ID of the
    newly created key
    * @throws RuntimeException if an error occurs while creating the key
    */
    public CompletableFuture<String> createKeyAsync(String keyDesc) {
        CreateKeyRequest keyRequest = CreateKeyRequest.builder()
            .description(keyDesc)
            .keySpec(KeySpec.SYMMETRIC_DEFAULT)
            .keyUsage(KeyUsageType.ENCRYPT_DECRYPT)
            .build();

        return getAsyncClient().createKey(keyRequest)
            .thenApply(resp -> resp.keyMetadata().keyId())
            .exceptionally(ex -> {
                throw new RuntimeException("An error occurred while creating the
key: " + ex.getMessage(), ex);
            });
    }

    /**
     * Asynchronously checks if a specified key is enabled.
     *
     * @param keyId the ID of the key to check
     * @return a {@link CompletableFuture} that, when completed, indicates
whether the key is enabled or not
     *
     * @throws RuntimeException if an exception occurs while checking the key
state
     */
    public CompletableFuture<Boolean> isKeyEnabledAsync(String keyId) {
        DescribeKeyRequest keyRequest = DescribeKeyRequest.builder()
            .keyId(keyId)
            .build();

        CompletableFuture<DescribeKeyResponse> responseFuture =
getAsyncClient().describeKey(keyRequest);
        return responseFuture.whenComplete((resp, ex) -> {
            if (resp != null) {
                KeyState keyState = resp.keyMetadata().keyState();
                if (keyState == KeyState.ENABLED) {
                    logger.info("The key is enabled.");
                } else {
                    logger.info("The key is not enabled. Key state: {}",
keyState);
                }
            }
        });
    }

```

```

        }
    } else {
        throw new RuntimeException(ex);
    }
}).thenApply(resp -> resp.keyMetadata().keyState() == KeyState.ENABLED);
}

/**
 * Asynchronously enables the specified key.
 *
 * @param keyId the ID of the key to enable
 * @return a {@link CompletableFuture} that completes when the key has been
 * enabled
 */
public CompletableFuture<Void> enableKeyAsync(String keyId) {
    EnableKeyRequest enableKeyRequest = EnableKeyRequest.builder()
        .keyId(keyId)
        .build();

    CompletableFuture<EnableKeyResponse> responseFuture =
getAsyncClient().enableKey(enableKeyRequest);
    responseFuture.whenComplete((response, exception) -> {
        if (exception == null) {
            logger.info("Key with ID [{}] has been enabled.", keyId);
        } else {
            if (exception instanceof KmsException kmsEx) {
                throw new RuntimeException("KMS error occurred while enabling
key: " + kmsEx.getMessage(), kmsEx);
            } else {
                throw new RuntimeException("An unexpected error occurred
while enabling key: " + exception.getMessage(), exception);
            }
        }
    });

    return responseFuture.thenApply(response -> null);
}

/**
 * Encrypts the given text asynchronously using the specified KMS client and
 * key ID.
 *
 * @param keyId the ID of the KMS key to use for encryption
 * @param text the text to encrypt

```

```

    * @return a CompletableFuture that completes with the encrypted data as an
    SdkBytes object
    */
    public CompletableFuture<SdkBytes> encryptDataAsync(String keyId, String
text) {
        SdkBytes myBytes = SdkBytes.fromUtf8String(text);
        EncryptRequest encryptRequest = EncryptRequest.builder()
            .keyId(keyId)
            .plaintext(myBytes)
            .build();

        CompletableFuture<EncryptResponse> responseFuture =
getAsyncClient().encrypt(encryptRequest).toCompletableFuture();
        return responseFuture.whenComplete((response, ex) -> {
            if (response != null) {
                String algorithm = response.encryptionAlgorithm().toString();
                logger.info("The string was encrypted with algorithm {}.\"",
algorithm);
            } else {
                throw new RuntimeException(ex);
            }
        }).thenApply(EncryptResponse::ciphertextBlob);
    }

    /**
     * Creates a custom alias for the specified target key asynchronously.
     *
     * @param targetKeyId the ID of the target key for the alias
     * @param aliasName the name of the alias to create
     * @return a {@link CompletableFuture} that completes when the alias creation
     operation is finished
     */
    public CompletableFuture<Void> createCustomAliasAsync(String targetKeyId,
String aliasName) {
        CreateAliasRequest aliasRequest = CreateAliasRequest.builder()
            .aliasName(aliasName)
            .targetKeyId(targetKeyId)
            .build();

        CompletableFuture<CreateAliasResponse> responseFuture =
getAsyncClient().createAlias(aliasRequest);
        responseFuture.whenComplete((response, exception) -> {
            if (exception == null) {
                logger.info("{} was successfully created.", aliasName);
            }
        });
    }

```

```

        } else {
            if (exception instanceof ResourceExistsException) {
                logger.info("Alias [{}] already exists. Moving on...",
aliasName);
            } else if (exception instanceof KmsException kmsEx) {
                throw new RuntimeException("KMS error occurred while creating
alias: " + kmsEx.getMessage(), kmsEx);
            } else {
                throw new RuntimeException("An unexpected error occurred
while creating alias: " + exception.getMessage(), exception);
            }
        }
    });

    return responseFuture.thenApply(response -> null);
}

/**
 * Asynchronously lists all the aliases in the current AWS account.
 *
 * @return a {@link CompletableFuture} that completes when the list of
aliases has been processed
 */
public CompletableFuture<Object> listAllAliasesAsync() {
    ListAliasesRequest aliasesRequest = ListAliasesRequest.builder()
        .limit(15)
        .build();

    ListAliasesPublisher paginator =
getAsyncClient().listAliasesPaginator(aliasesRequest);
    return paginator.subscribe(response -> {
        response.aliases().forEach(alias ->
            logger.info("The alias name is: " + alias.aliasName())
        );
    })
        .thenApply(v -> null)
        .exceptionally(ex -> {
            if (ex.getCause() instanceof KmsException) {
                KmsException e = (KmsException) ex.getCause();
                throw new RuntimeException("A KMS exception occurred: " +
e.getMessage());
            } else {
                throw new RuntimeException("An unexpected error occurred: " +
ex.getMessage());
            }
        });
}

```



```

        }
    });
}

/**
 * Enables key rotation asynchronously for the specified key ID.
 *
 * @param keyId the ID of the key for which to enable key rotation
 * @return a CompletableFuture that represents the asynchronous operation of
 * enabling key rotation
 * @throws RuntimeException if there was an error enabling key rotation,
 * either due to a KMS exception or an unexpected error
 */
public CompletableFuture<EnableKeyRotationResponse>
enableKeyRotationAsync(String keyId) {
    EnableKeyRotationRequest enableKeyRotationRequest =
    EnableKeyRotationRequest.builder()
        .keyId(keyId)
        .build();

    CompletableFuture<EnableKeyRotationResponse> responseFuture =
    getAsyncClient().enableKeyRotation(enableKeyRotationRequest);
    responseFuture.whenComplete((response, exception) -> {
        if (exception == null) {
            logger.info("Key rotation has been enabled for key with id [{}]",
keyId);
        } else {
            if (exception instanceof KmsException kmsEx) {
                throw new RuntimeException("Failed to enable key rotation: "
+ kmsEx.getMessage(), kmsEx);
            } else {
                throw new RuntimeException("An unexpected error occurred: " +
exception.getMessage(), exception);
            }
        }
    });

    return responseFuture;
}

/**
 * Grants permissions to a specified principal on a customer master key (CMK)
 * asynchronously.
 *

```

```
    * @param keyId          The unique identifier for the customer master key
    (CMK) that the grant applies to.
    * @param granteePrincipal The principal that is given permission to perform
    the operations that the grant permits on the CMK.
    * @return A {@link CompletableFuture} that, when completed, contains the ID
    of the created grant.
    * @throws RuntimeException If an error occurs during the grant creation
    process.
    */
    public CompletableFuture<String> grantKeyAsync(String keyId, String
    granteePrincipal) {
        List<GrantOperation> grantPermissions = List.of(
            GrantOperation.ENCRYPT,
            GrantOperation.DECRYPT,
            GrantOperation.DESCRIBE_KEY
        );

        CreateGrantRequest grantRequest = CreateGrantRequest.builder()
            .keyId(keyId)
            .name("grant1")
            .granteePrincipal(granteePrincipal)
            .operations(grantPermissions)
            .build();

        CompletableFuture<CreateGrantResponse> responseFuture =
        getAsyncClient().createGrant(grantRequest);
        responseFuture.whenComplete((response, ex) -> {
            if (ex == null) {
                logger.info("Grant created successfully with ID: " +
                response.grantId());
            } else {
                if (ex instanceof KmsException kmsEx) {
                    throw new RuntimeException("Failed to create grant: " +
                    kmsEx.getMessage(), kmsEx);
                } else {
                    throw new RuntimeException("An unexpected error occurred: " +
                    ex.getMessage(), ex);
                }
            }
        });

        return responseFuture.thenApply(CreateGrantResponse::grantId);
    }
}
```

```

/**
 * Asynchronously displays the grant IDs for the specified key ID.
 *
 * @param keyId the ID of the AWS KMS key for which to list the grants
 * @return a {@link CompletableFuture} that, when completed, will be null
if the operation succeeded, or will throw a {@link RuntimeException} if the
operation failed
 * @throws RuntimeException if there was an error listing the grants, either
due to an {@link KmsException} or an unexpected error
 */
public CompletableFuture<Object> displayGrantIdsAsync(String keyId) {
    ListGrantsRequest grantsRequest = ListGrantsRequest.builder()
        .keyId(keyId)
        .limit(15)
        .build();

    ListGrantsPublisher paginator =
getAsyncClient().listGrantsPaginator(grantsRequest);
    return paginator.subscribe(response -> {
        response.grants().forEach(grant -> {
            logger.info("The grant Id is: " + grant.grantId());
        });
    })
        .thenApply(v -> null)
        .exceptionally(ex -> {
            Throwable cause = ex.getCause();
            if (cause instanceof KmsException) {
                throw new RuntimeException("Failed to list grants: " +
cause.getMessage(), cause);
            } else {
                throw new RuntimeException("An unexpected error occurred: " +
cause.getMessage(), cause);
            }
        });
}

/**
 * Revokes a grant for the specified AWS KMS key asynchronously.
 *
 * @param keyId The ID or key ARN of the AWS KMS key.
 * @param grantId The identifier of the grant to be revoked.
 * @return A {@link CompletableFuture} representing the asynchronous
operation of revoking the grant.

```

```

    *      The {@link CompletableFuture} will complete with a {@link
    RevokeGrantResponse} object
    *      if the operation is successful, or with a {@code null} value if an
    error occurs.
    */
    public CompletableFuture<RevokeGrantResponse> revokeKeyGrantAsync(String
    keyId, String grantId) {
        RevokeGrantRequest grantRequest = RevokeGrantRequest.builder()
            .keyId(keyId)
            .grantId(grantId)
            .build();

        CompletableFuture<RevokeGrantResponse> responseFuture =
    getAsyncClient().revokeGrant(grantRequest);
        responseFuture.whenComplete((response, exception) -> {
            if (exception == null) {
                logger.info("Grant ID: [" + grantId + "] was successfully
    revoked!");
            } else {
                if (exception instanceof KmsException kmsEx) {
                    if (kmsEx.getMessage().contains("Grant does not exist")) {
                        logger.info("The grant ID '" + grantId + "' does not
    exist. Moving on...");
                    } else {
                        throw new RuntimeException("KMS error occurred: " +
    kmsEx.getMessage(), kmsEx);
                    }
                } else {
                    throw new RuntimeException("An unexpected error occurred: " +
    exception.getMessage(), exception);
                }
            }
        });

        return responseFuture;
    }

    /**
    * Asynchronously decrypts the given encrypted data using the specified key
    ID.
    *
    * @param encryptedData The encrypted data to be decrypted.
    * @param keyId The ID of the key to be used for decryption.

```

```

    * @return A CompletableFuture that, when completed, will contain the
    decrypted data as a String.
    *
    * If an error occurs during the decryption process, the
    CompletableFuture will complete
    *
    * exceptionally with the error, and the method will return an empty
    String.
    */
    public CompletableFuture<String> decryptDataAsync(SdkBytes encryptedData,
String keyId) {
        DecryptRequest decryptRequest = DecryptRequest.builder()
            .ciphertextBlob(encryptedData)
            .keyId(keyId)
            .build();

        CompletableFuture<DecryptResponse> responseFuture =
getAsyncClient().decrypt(decryptRequest);
        responseFuture.whenComplete((decryptResponse, exception) -> {
            if (exception == null) {
                logger.info("Data decrypted successfully for key ID: " + keyId);
            } else {
                if (exception instanceof KmsException kmsEx) {
                    throw new RuntimeException("KMS error occurred while
decrypting data: " + kmsEx.getMessage(), kmsEx);
                } else {
                    throw new RuntimeException("An unexpected error occurred
while decrypting data: " + exception.getMessage(), exception);
                }
            }
        });

        return responseFuture.thenApply(decryptResponse ->
decryptResponse.plaintext().asString(StandardCharsets.UTF_8));
    }

    /**
     * Asynchronously replaces the policy for the specified KMS key.
     *
     * @param keyId      the ID of the KMS key to replace the policy for
     * @param policyName the name of the policy to be replaced
     * @param accountId  the AWS account ID to be used in the policy
     * @return a {@link CompletableFuture} that completes with a boolean
     indicating
     *
     * whether the policy replacement was successful or not
     */

```

```

    public CompletableFuture<Boolean> replacePolicyAsync(String keyId, String
policyName, String accountId) {
        String policy = ""
        {
            "Version": "2012-10-17",
            "Statement": [{
                "Effect": "Allow",
                "Principal": {"AWS": "arn:aws:iam::%s:root"},
                "Action": "kms:*",
                "Resource": "*"
            }]
        }
        ""formatted(accountId);

        PutKeyPolicyRequest keyPolicyRequest = PutKeyPolicyRequest.builder()
            .keyId(keyId)
            .policyName(policyName)
            .policy(policy)
            .build();

        // First, get the current policy to check if it exists
        return getAsyncClient().getKeyPolicy(r ->
r.keyId(keyId).policyName(policyName))
            .thenCompose(response -> {
                logger.info("Current policy exists. Replacing it...");
                return getAsyncClient().putKeyPolicy(keyPolicyRequest);
            })
            .thenApply(putPolicyResponse -> {
                logger.info("The key policy has been replaced.");
                return true;
            })
            .exceptionally(throwable -> {
                if (throwable.getCause() instanceof LimitExceededException) {
                    logger.error("Cannot replace policy, as only one policy is
allowed per key.");
                    return false;
                }
                throw new RuntimeException("Error replacing policy", throwable);
            });
    }

    /**

```

```

    * Asynchronously retrieves the key policy for the specified key ID and
    policy name.
    *
    * @param keyId      the ID of the AWS KMS key for which to retrieve the
    policy
    * @param policyName the name of the key policy to retrieve
    * @return a {@link CompletableFuture} that, when completed, contains the key
    policy as a {@link String}
    */
    public CompletableFuture<String> getKeyPolicyAsync(String keyId, String
    policyName) {
        GetKeyPolicyRequest policyRequest = GetKeyPolicyRequest.builder()
            .keyId(keyId)
            .policyName(policyName)
            .build();

        return getAsyncClient().getKeyPolicy(policyRequest)
            .thenApply(response -> {
                String policy = response.policy();
                logger.info("The response is: " + policy);
                return policy;
            })
            .exceptionally(ex -> {
                throw new RuntimeException("Failed to get key policy", ex);
            });
    }

    /**
     * Asynchronously signs and verifies data using AWS KMS.
     *
     * <p>The method performs the following steps:</p>
     * <ol>
     *     <li>Creates an AWS KMS key with the specified key spec, key usage, and
     origin.</li>
     *     <li>Signs the provided message using the created KMS key and the
     RSASSA-PSS-SHA-256 algorithm.</li>
     *     <li>Verifies the signature of the message using the created KMS key
     and the RSASSA-PSS-SHA-256 algorithm.</li>
     * </ol>
     *
     * @return a {@link CompletableFuture} that completes with the result of the
     signature verification,
     *         {@code true} if the signature is valid, {@code false} otherwise.
     * @throws KmsException if any error occurs during the KMS operations.

```

```

    * @throws RuntimeException if an unexpected error occurs.
    */
    public CompletableFuture<Boolean> signVerifyDataAsync() {
        String signMessage = "Here is the message that will be digitally signed";

        // Create an AWS KMS key used to digitally sign data.
        CreateKeyRequest createKeyRequest = CreateKeyRequest.builder()
            .keySpec(KeySpec.RSA_2048)
            .keyUsage(KeyUsageType.SIGN_VERIFY)
            .origin(OriginType.AWS_KMS)
            .build();

        return getAsyncClient().createKey(createKeyRequest)
            .thenCompose(createKeyResponse -> {
                String keyId = createKeyResponse.keyMetadata().keyId();

                SdkBytes messageBytes = SdkBytes.fromString(signMessage,
                    Charset.defaultCharset());
                SignRequest signRequest = SignRequest.builder()
                    .keyId(keyId)
                    .message(messageBytes)
                    .signingAlgorithm(SigningAlgorithmSpec.RSASSA_PSS_SHA_256)
                    .build();

                return getAsyncClient().sign(signRequest)
                    .thenCompose(signResponse -> {
                        byte[] signedBytes =
signResponse.signature().asByteArray();

                        VerifyRequest verifyRequest = VerifyRequest.builder()
                            .keyId(keyId)

                            .message(SdkBytes.fromByteArray(signMessage.getBytes(Charset.defaultCharset())))

                            .signature(SdkBytes.fromByteBuffer(ByteBuffer.wrap(signedBytes)))

                            .signingAlgorithm(SigningAlgorithmSpec.RSASSA_PSS_SHA_256)
                                .build();

                        return getAsyncClient().verify(verifyRequest)
                            .thenApply(verifyResponse -> {
                                return (boolean) verifyResponse.signatureValid();
                            });
                    });
            });
    }

```



```
        })
        .exceptionally(throwable -> {
            throw new RuntimeException("Failed to sign or verify data",
throwable);
        });
    }

    /**
     * Asynchronously tags a KMS key with a specific tag.
     *
     * @param keyId the ID of the KMS key to be tagged
     * @return a {@link CompletableFuture} that completes when the tagging
operation is finished
     */
    public CompletableFuture<Void> tagKMSKeyAsync(String keyId) {
        Tag tag = Tag.builder()
            .tagKey("Environment")
            .tagValue("Production")
            .build();

        TagResourceRequest tagResourceRequest = TagResourceRequest.builder()
            .keyId(keyId)
            .tags(tag)
            .build();

        return getAsyncClient().tagResource(tagResourceRequest)
            .thenRun(() -> {
                logger.info("{} key was tagged", keyId);
            })
            .exceptionally(throwable -> {
                throw new RuntimeException("Failed to tag the KMS key",
throwable);
            });
    }

    /**
     * Deletes a specific KMS alias asynchronously.
     *
     * @param aliasName the name of the alias to be deleted
     * @return a {@link CompletableFuture} representing the asynchronous
operation of deleting the specified alias
     */
    public CompletableFuture<Void> deleteSpecificAliasAsync(String aliasName) {
        DeleteAliasRequest deleteAliasRequest = DeleteAliasRequest.builder()
```

```

        .aliasName(aliasName)
        .build();

    return getAsyncClient().deleteAlias(deleteAliasRequest)
        .thenRun(() -> {
            logger.info("Alias {} has been deleted successfully", aliasName);
        })
        .exceptionally(throwable -> {
            throw new RuntimeException("Failed to delete alias: " +
aliasName, throwable);
        });
    }

    /**
     * Asynchronously disables the specified AWS Key Management Service (KMS)
key.
     *
     * @param keyId the ID or Amazon Resource Name (ARN) of the KMS key to be
disabled
     * @return a CompletableFuture that, when completed, indicates that the key
has been disabled successfully
     */
    public CompletableFuture<Void> disableKeyAsync(String keyId) {
        DisableKeyRequest keyRequest = DisableKeyRequest.builder()
            .keyId(keyId)
            .build();

        return getAsyncClient().disableKey(keyRequest)
            .thenRun(() -> {
                logger.info("Key {} has been disabled successfully",keyId);
            })
            .exceptionally(throwable -> {
                throw new RuntimeException("Failed to disable key: " + keyId,
throwable);
            });
    }

    /**
     * Deletes a KMS key asynchronously.
     *
     * <p><strong>Warning:</strong> Deleting a KMS key is a destructive and
potentially dangerous operation.
     *
     * When a KMS key is deleted, all data that was encrypted under the KMS key
becomes unrecoverable.

```

```
* This means that any files, databases, or other data that were encrypted
using the deleted KMS key
* will become permanently inaccessible. Exercise extreme caution when
deleting KMS keys.</p>
*
* @param keyId the ID of the KMS key to delete
* @return a {@link CompletableFuture} that completes when the key deletion
is scheduled
*/
public CompletableFuture<Void> deleteKeyAsync(String keyId) {
    ScheduleKeyDeletionRequest deletionRequest =
ScheduleKeyDeletionRequest.builder()
        .keyId(keyId)
        .pendingWindowInDays(7)
        .build();

    return getAsyncClient().scheduleKeyDeletion(deletionRequest)
        .thenRun(() -> {
            logger.info("Key {} will be deleted in 7 days", keyId);
        })
        .exceptionally(throwable -> {
            throw new RuntimeException("Failed to schedule key deletion for
key ID: " + keyId, throwable);
        });
}

public String getAccountId(){
    try (StsClient stsClient = StsClient.create()){
        GetCallerIdentityResponse callerIdentity =
stsClient.getCallerIdentity();
        return callerIdentity.account();
    }
}
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
 - [CreateAlias](#)
 - [CreateGrant](#)
 - [CreateKey](#)
 - [解密](#)

- [DescribeKey](#)
- [DisableKey](#)
- [EnableKey](#)
- [加密](#)
- [GetKeyPolicy](#)
- [ListAliases](#)
- [ListGrants](#)
- [ListKeys](#)
- [RevokeGrant](#)
- [ScheduleKeyDeletion](#)
- [符號](#)
- [TagResource](#)

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
echo "\n";
echo "-----\n";
echo <<<WELCOME
```

Welcome to the AWS Key Management Service SDK Basics scenario.

This program demonstrates how to interact with AWS Key Management Service using the AWS SDK for PHP (v3).

The AWS Key Management Service (KMS) is a secure and highly available service that allows you to create

and manage AWS KMS keys and control their use across a wide range of AWS services and applications.

KMS provides a centralized and unified approach to managing encryption keys, making it easier to meet your

data protection and regulatory compliance requirements.

This KMS Basics scenario creates two key types:

- A symmetric encryption key is used to encrypt and decrypt data.
- An asymmetric key used to digitally sign data.

Let's get started...\n

WELCOME;

```
    echo "-----\n";
    $this->pressEnter();

    $this->kmsClient = new KmsClient([]);
    // Initialize the KmsService class with the client. This allows you to
    override any defaults in the client before giving it to the service class.
    $this->kmsService = new KmsService($this->kmsClient);

    // 1. Create a symmetric KMS key.
    echo "\n";
    echo "1. Create a symmetric KMS key.\n";
    echo "First, we will create a symmetric KMS key that is used to encrypt
and decrypt data by invoking createKey().\n";
    $this->pressEnter();

    $key = $this->kmsService->createKey();
    $this->resources['symmetricKey'] = $key['KeyId'];
    echo "Created a customer key with ARN {$key['Arn']}. \n";
    $this->pressEnter();

    // 2. Enable a KMS key.
    echo "\n";
    echo "2. Enable a KMS key.\n";
    echo "By default when you create an AWS key, it is enabled. The code
checks to
determine if the key is enabled. If it is not enabled, the code enables it.\n";
    $this->pressEnter();

    $keyInfo = $this->kmsService->describeKey($key['KeyId']);
    if(!$keyInfo['Enabled']){
        echo "The key was not enabled, so we will enable it.\n";
        $this->pressEnter();
        $this->kmsService->enableKey($key['KeyId']);
        echo "The key was successfully enabled.\n";
    }else{
```

```
        echo "The key was already enabled, so there was no need to enable it.
\n";
    }
    $this->pressEnter();

    // 3. Encrypt data using the symmetric KMS key.
    echo "\n";
    echo "3. Encrypt data using the symmetric KMS key.\n";
    echo "One of the main uses of symmetric keys is to encrypt and decrypt
data.\n";
    echo "Next, we'll encrypt the string 'Hello, AWS KMS!' with the
SYMMETRIC_DEFAULT encryption algorithm.\n";
    $this->pressEnter();
    $text = "Hello, AWS KMS!";
    $encryption = $this->kmsService->encrypt($key['KeyId'], $text);
    echo "The plaintext data was successfully encrypted with the algorithm:
{$encryption['EncryptionAlgorithm']}. \n";
    $this->pressEnter();

    // 4. Create an alias.
    echo "\n";
    echo "4. Create an alias.\n";
    $aliasInput = testable_readline("Please enter an alias prefixed with
\n\"alias/\n" or press enter to use a default value: ");
    if($aliasInput == ""){
        $aliasInput = "alias/dev-encryption-key";
    }
    $this->kmsService->createAlias($key['KeyId'], $aliasInput);
    $this->resources['alias'] = $aliasInput;
    echo "The alias \"$aliasInput\" was successfully created.\n";
    $this->pressEnter();

    // 5. List all of your aliases.
    $aliasPageSize = 10;
    echo "\n";
    echo "5. List all of your aliases, up to $aliasPageSize.\n";
    $this->pressEnter();
    $aliasPaginator = $this->kmsService->listAliases();
    foreach($aliasPaginator as $pages){
        foreach($pages['Aliases'] as $alias){
            echo $alias['AliasName'] . "\n";
        }
        break;
    }
}
```

```

$this->pressEnter();

// 6. Enable automatic rotation of the KMS key.
echo "\n";
echo "6. Enable automatic rotation of the KMS key.\n";
echo "By default, when the SDK enables automatic rotation of a KMS key,
KMS rotates the key material of the KMS key one year (approximately 365 days)
from the enable date and every year
thereafter.";
$this->pressEnter();
$this->kmsService->enableKeyRotation($key['KeyId']);
echo "The key's rotation was successfully set for key:
{$key['KeyId']}\n";
$this->pressEnter();

// 7. Create a grant.
echo "7. Create a grant.\n";
echo "\n";
echo "A grant is a policy instrument that allows Amazon Web Services
principals to use KMS keys.
It also can allow them to view a KMS key (DescribeKey) and create and manage
grants.
When authorizing access to a KMS key, grants are considered along with key
policies and IAM policies.\n";
$granteeARN = testable_readline("Please enter the Amazon Resource Name
(ARN) of an Amazon Web Services principal. Valid principals include Amazon
Web Services accounts, IAM users, IAM roles, federated users, and assumed
role users. For help with the ARN syntax for a principal, see IAM ARNs in the
Identity and Access Management User Guide. \nTo skip this step, press enter
without any other values: ");
if($granteeARN){
    $operations = [
        "ENCRYPT",
        "DECRYPT",
        "DESCRIBE_KEY",
    ];
    $grant = $this->kmsService->createGrant($key['KeyId'], $granteeARN,
    $operations);
    echo "The grant Id is: {$grant['GrantId']}\n";
}else{
    echo "Steps 7, 8, and 9 will be skipped.\n";
}
$this->pressEnter();

```

```
// 8. List grants for the KMS key.
if($granteeARN){
    echo "8. List grants for the KMS key.\n\n";
    $grantsPaginator = $this->kmsService->listGrants($key['KeyId']);
    foreach($grantsPaginator as $page){
        foreach($page['Grants'] as $grant){
            echo $grant['GrantId'] . "\n";
        }
    }
}else{
    echo "Skipping step 8...\n";
}
$this->pressEnter();

// 9. Revoke the grant.
if($granteeARN) {
    echo "\n";
    echo "9. Revoke the grant.\n";
    $this->pressEnter();
    $this->kmsService->revokeGrant($grant['GrantId'], $keyInfo['KeyId']);
    echo "{$grant['GrantId']} was successfully revoked!\n";
}else{
    echo "Skipping step 9...\n";
}
$this->pressEnter();

// 10. Decrypt the data.
echo "\n";
echo "10. Decrypt the data.\n";
echo "Let's decrypt the data that was encrypted before.\n";
echo "We'll use the same key to decrypt the string that we encrypted
earlier in the program.\n";
$this->pressEnter();
$decryption = $this->kmsService->decrypt($keyInfo['KeyId'],
$encryption['CiphertextBlob'], $encryption['EncryptionAlgorithm']);
echo "The decrypted text is: {$decryption['Plaintext']}\n";
$this->pressEnter();

// 11. Replace a Key Policy.
echo "\n";
echo "11. Replace a Key Policy.\n";
echo "A key policy is a resource policy for a KMS key. Key policies are
the primary way to control access to KMS keys.\n";
```



```

        echo "Every KMS key must have exactly one key policy. The statements in
the key policy determine who has permission to use the KMS key and how they can
use it.\n";
        echo " You can also use IAM policies and grants to control access to the
KMS key, but every KMS key must have a key policy.\n";
        echo "We will replace the key's policy with a new one:\n";
        $stsClient = new StsClient([]);
        $result = $stsClient->getCallerIdentity();
        $accountId = $result['Account'];
        $keyPolicy = <<< KEYPOLICY
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Principal": {"AWS": "arn:aws:iam::$accountId:root"},
        "Action": "kms:*",
        "Resource": "*"
    }]
}
KEYPOLICY;
        echo $keyPolicy;
        $this->pressEnter();
        $this->kmsService->putKeyPolicy($keyInfo['KeyId'], $keyPolicy);
        echo "The Key Policy was successfully replaced!\n";
        $this->pressEnter();

        // 12. Retrieve the key policy.
        echo "\n";
        echo "12. Retrieve the key policy.\n";
        echo "Let's get some information about the new policy and print it to the
screen.\n";
        $this->pressEnter();
        $policyInfo = $this->kmsService->getKeyPolicy($keyInfo['KeyId']);
        echo "We got the info! Here is the policy: \n";
        echo $policyInfo['Policy'] . "\n";
        $this->pressEnter();

        // 13. Create an asymmetric KMS key and sign data.
        echo "\n";
        echo "13. Create an asymmetric KMS key and sign data.\n";
        echo "Signing your data with an AWS key can provide several benefits that
make it an attractive option for your data signing needs.\n";

```

```

    echo "By using an AWS KMS key, you can leverage the security controls and
    compliance features provided by AWS, which can help you meet various regulatory
    requirements and enhance the overall security posture of your organization.\n";
    echo "First we'll create the asymmetric key.\n";
    $this->pressEnter();
    $keySpec = "RSA_2048";
    $keyUsage = "SIGN_VERIFY";
    $asymmetricKey = $this->kmsService->createKey($keySpec, $keyUsage);
    $this->resources['asymmetricKey'] = $asymmetricKey['KeyId'];
    echo "Created the key with ID: {$asymmetricKey['KeyId']}\n";
    echo "Next, we'll sign the data.\n";
    $this->pressEnter();
    $algorithm = "RSASSA_PSS_SHA_256";
    $sign = $this->kmsService->sign($asymmetricKey['KeyId'], $text,
    $algorithm);
    $verify = $this->kmsService->verify($asymmetricKey['KeyId'], $text,
    $sign['Signature'], $algorithm);
    echo "Signature verification result: {$sign['signature']}\n";
    $this->pressEnter();

    // 14. Tag the symmetric KMS key.
    echo "\n";
    echo "14. Tag the symmetric KMS key.\n";
    echo "By using tags, you can improve the overall management, security,
    and governance of your KMS keys, making it easier to organize, track, and
    control access to your encrypted data within your AWS environment.\n";
    echo "Let's tag our symmetric key as Environment->Production\n";
    $this->pressEnter();
    $this->kmsService->tagResource($key['KeyId'], [
        [
            'TagKey' => "Environment",
            'TagValue' => "Production",
        ],
    ]);
    echo "The key was successfully tagged!\n";
    $this->pressEnter();

    // 15. Schedule the deletion of the KMS key
    echo "\n";
    echo "15. Schedule the deletion of the KMS key.\n";
    echo "By default, KMS applies a waiting period of 30 days, but you can
    specify a waiting period of 7-30 days.\n";

```

```

        echo "When this operation is successful, the key state of the KMS key
changes to PendingDeletion and the key can't be used in any cryptographic
operations.\n";
        echo "It remains in this state for the duration of the waiting period.\n
\n";

        echo "Deleting a KMS key is a destructive and potentially dangerous
operation. When a KMS key is deleted, all data that was encrypted under the KMS
key is unrecoverable.\n\n";

        $cleanUp = testable_readline("Would you like to delete the resources
created during this scenario, including the keys? (y/n): ");
        if($cleanUp == "Y" || $cleanUp == "y"){
            $this->cleanUp();
        }

        echo
"-----
\n";
        echo "This concludes the AWS Key Management SDK Basics scenario\n";
        echo
"-----
\n";

namespace Kms;

use Aws\Kms\Exception\KmsException;
use Aws\Kms\KmsClient;
use Aws\Result;
use Aws\ResultPaginator;
use AwsUtilities\AWSServiceClass;

class KmsService extends AWSServiceClass
{
    protected KmsClient $client;
    protected bool $verbose;

    /**
     * @param KmsClient|null $client
     * @param bool $verbose
     */

```

```

public function __construct(KmsClient $client = null, bool $verbose = false)
{
    $this->verbose = $verbose;
    if($client){
        $this->client = $client;
        return;
    }
    $this->client = new KmsClient([]);
}

/**
 * @param string $keySpec
 * @param string $keyUsage
 * @param string $description
 * @return array
 */
public function createKey(string $keySpec = "", string $keyUsage = "", string
$description = "Created by the SDK for PHP")
{
    $parameters = ['Description' => $description];
    if($keySpec && $keyUsage){
        $parameters['KeySpec'] = $keySpec;
        $parameters['KeyUsage'] = $keyUsage;
    }
    try {
        $result = $this->client->createKey($parameters);
        return $result['KeyMetadata'];
    }catch(KmsException $caught){
        // Check for error specific to createKey operations
        if ($caught->getAwsErrorMessage() == "LimitExceededException"){
            echo "The request was rejected because a quota was exceeded. For
more information, see Quotas in the Key Management Service Developer Guide.";
        }
        throw $caught;
    }
}

/**
 * @param string $keyId
 * @param string $ciphertext
 * @param string $algorithm

```

```
* @return Result
*/
public function decrypt(string $keyId, string $ciphertext, string $algorithm
= "SYMMETRIC_DEFAULT")
{
    try{
        return $this->client->decrypt([
            'CiphertextBlob' => $ciphertext,
            'EncryptionAlgorithm' => $algorithm,
            'KeyId' => $keyId,
        ]);
    }catch(KmsException $caught){
        echo "There was a problem decrypting the data: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
}

/**
 * @param string $keyId
 * @param string $text
 * @return Result
 */
public function encrypt(string $keyId, string $text)
{
    try {
        return $this->client->encrypt([
            'KeyId' => $keyId,
            'Plaintext' => $text,
        ]);
    }catch(KmsException $caught){
        if($caught->getAwsErrorMessage() == "DisabledException"){
            echo "The request was rejected because the specified KMS key is
not enabled.\n";
        }
        throw $caught;
    }
}

/**
```

```

    * @param string $keyId
    * @param int $limit
    * @return ResultPaginator
    */
    public function listAliases(string $keyId = "", int $limit = 0)
    {
        $args = [];
        if($keyId){
            $args['KeyId'] = $keyId;
        }
        if($limit){
            $args['Limit'] = $limit;
        }
        try{
            return $this->client->getPaginator("ListAliases", $args);
        }catch(KmsException $caught){
            if($caught->getAwsErrorMessage() == "InvalidMarkerException"){
                echo "The request was rejected because the marker that specifies
where pagination should next begin is not valid.\n";
            }
            throw $caught;
        }
    }

    /**
    * @param string $keyId
    * @param string $alias
    * @return void
    */
    public function createAlias(string $keyId, string $alias)
    {
        try{
            $this->client->createAlias([
                'TargetKeyId' => $keyId,
                'AliasName' => $alias,
            ]);
        }catch (KmsException $caught){
            if($caught->getAwsErrorMessage() == "InvalidAliasNameException"){
                echo "The request was rejected because the specified alias name
is not valid.";
            }
            throw $caught;
        }
    }

```

```

    }
}

/**
 * @param string $keyId
 * @param string $granteePrincipal
 * @param array $operations
 * @param array $grantTokens
 * @return Result
 */
public function createGrant(string $keyId, string $granteePrincipal, array
$operations, array $grantTokens = [])
{
    $args = [
        'KeyId' => $keyId,
        'GranteePrincipal' => $granteePrincipal,
        'Operations' => $operations,
    ];
    if($grantTokens){
        $args['GrantTokens'] = $grantTokens;
    }
    try{
        return $this->client->createGrant($args);
    }catch(KmsException $caught){
        if($caught->getAwsErrorMessage() == "InvalidGrantTokenException"){
            echo "The request was rejected because the specified grant token
is not valid.\n";
        }
        throw $caught;
    }
}

/**
 * @param string $keyId
 * @return array
 */
public function describeKey(string $keyId)
{
    try {
        $result = $this->client->describeKey([

```

```

        "KeyId" => $keyId,
    ]);
    return $result['KeyMetadata'];
} catch (KmsException $caught){
    if($caught->getAwsErrorMessage() == "NotFoundException"){
        echo "The request was rejected because the specified entity or
resource could not be found.\n";
    }
    throw $caught;
}
}

/**
 * @param string $keyId
 * @return void
 */
public function disableKey(string $keyId)
{
    try {
        $this->client->disableKey([
            'KeyId' => $keyId,
        ]);
    } catch (KmsException $caught){
        echo "There was a problem disabling the key: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
}

/**
 * @param string $keyId
 * @return void
 */
public function enableKey(string $keyId)
{
    try {
        $this->client->enableKey([
            'KeyId' => $keyId,
        ]);
    } catch (KmsException $caught){

```



```

        if($caught->getAwsErrorMessage() == "NotFoundException"){
            echo "The request was rejected because the specified entity or
resource could not be found.\n";
        }
        throw $caught;
    }
}

/**
 * @return array
 */
public function listKeys()
{
    try {
        $contents = [];
        $paginator = $this->client->getPaginator("ListKeys");
        foreach($paginator as $result){
            foreach ($result['Content'] as $object) {
                $contents[] = $object;
            }
        }
        return $contents;
    }catch(KmsException $caught){
        echo "There was a problem listing the keys: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
}

/**
 * @param string $keyId
 * @return Result
 */
public function listGrants(string $keyId)
{
    try{
        return $this->client->listGrants([
            'KeyId' => $keyId,
        ]);
    }catch(KmsException $caught){

```

```

        if($caught->getAwsErrorMessage() == "NotFoundException"){
            echo "    The request was rejected because the specified entity
or resource could not be found.\n";
        }
        throw $caught;
    }
}

/**
 * @param string $keyId
 * @return Result
 */
public function getKeyPolicy(string $keyId)
{
    try {
        return $this->client->getKeyPolicy([
            'KeyId' => $keyId,
        ]);
    }catch(KmsException $caught){
        echo "There was a problem getting the key policy: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
}

/**
 * @param string $grantId
 * @param string $keyId
 * @return void
 */
public function revokeGrant(string $grantId, string $keyId)
{
    try{
        $this->client->revokeGrant([
            'GrantId' => $grantId,
            'KeyId' => $keyId,
        ]);
    }catch(KmsException $caught){
        echo "There was a problem with revoking the grant: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
}

```

```

    }

    /**
     * @param string $keyId
     * @param int $pendingWindowInDays
     * @return void
     */
    public function scheduleKeyDeletion(string $keyId, int $pendingWindowInDays =
7)
    {
        try {
            $this->client->scheduleKeyDeletion([
                'KeyId' => $keyId,
                'PendingWindowInDays' => $pendingWindowInDays,
            ]);
        } catch (KmsException $caught){
            echo "There was a problem scheduling the key deletion: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }

    /**
     * @param string $keyId
     * @param array $tags
     * @return void
     */
    public function tagResource(string $keyId, array $tags)
    {
        try {
            $this->client->tagResource([
                'KeyId' => $keyId,
                'Tags' => $tags,
            ]);
        } catch (KmsException $caught){
            echo "There was a problem applying the tag(s): {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }
}

```

```

    /**
     * @param string $keyId
     * @param string $message
     * @param string $algorithm
     * @return Result
     */
    public function sign(string $keyId, string $message, string $algorithm)
    {
        try {
            return $this->client->sign([
                'KeyId' => $keyId,
                'Message' => $message,
                'SigningAlgorithm' => $algorithm,
            ]);
        } catch (KmsException $caught){
            echo "There was a problem signing the data: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }

    /**
     * @param string $keyId
     * @param int $rotationPeriodInDays
     * @return void
     */
    public function enableKeyRotation(string $keyId, int $rotationPeriodInDays =
365)
    {
        try{
            $this->client->enableKeyRotation([
                'KeyId' => $keyId,
                'RotationPeriodInDays' => $rotationPeriodInDays,
            ]);
        } catch (KmsException $caught){
            if($caught->getAwsErrorMessage() == "NotFoundException"){
                echo "The request was rejected because the specified entity or
resource could not be found.\n";
            }
        }
    }

```

```
        throw $caught;
    }
}

/**
 * @param string $keyId
 * @param string $policy
 * @return void
 */
public function putKeyPolicy(string $keyId, string $policy)
{
    try {
        $this->client->putKeyPolicy([
            'KeyId' => $keyId,
            'Policy' => $policy,
        ]);
    } catch (KmsException $caught) {
        echo "There was a problem replacing the key policy: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
}

/**
 * @param string $aliasName
 * @return void
 */
public function deleteAlias(string $aliasName)
{
    try {
        $this->client->deleteAlias([
            'AliasName' => $aliasName,
        ]);
    } catch (KmsException $caught) {
        echo "There was a problem deleting the alias: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
}
```

```
/**
 * @param string $keyId
 * @param string $message
 * @param string $signature
 * @param string $signingAlgorithm
 * @return bool
 */
public function verify(string $keyId, string $message, string $signature,
string $signingAlgorithm)
{
    try {
        $result = $this->client->verify([
            'KeyId' => $keyId,
            'Message' => $message,
            'Signature' => $signature,
            'SigningAlgorithm' => $signingAlgorithm,
        ]);
        return $result['SignatureValid'];
    } catch (KmsException $caught){
        echo "There was a problem verifying the signature: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的下列主題。

- [CreateAlias](#)
- [CreateGrant](#)
- [CreateKey](#)
- [解密](#)
- [DescribeKey](#)
- [DisableKey](#)
- [EnableKey](#)

- [加密](#)
- [GetKeyPolicy](#)
- [ListAliases](#)
- [ListGrants](#)
- [ListKeys](#)
- [RevokeGrant](#)
- [ScheduleKeyDeletion](#)
- [符號](#)
- [TagResource](#)

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class KMSScenario:
    """Runs an interactive scenario that shows how to get started with KMS."""

    def __init__(
        self,
        key_manager: KeyManager,
        key_encryption: KeyEncrypt,
        alias_manager: AliasManager,
        grant_manager: GrantManager,
        key_policy: KeyPolicy,
    ):
        self.key_manager = key_manager
        self.key_encryption = key_encryption
        self.alias_manager = alias_manager
        self.grant_manager = grant_manager
        self.key_policy = key_policy
        self.key_id = ""
        self.alias_name = ""
```

```

        self.asymmetric_key_id = ""

    def kms_scenario(self):
        key_description = "Created by the AWS KMS API"

        print(DASHES)
        print(
            """
Welcome to the AWS Key Management SDK Basics scenario.

This program demonstrates how to interact with AWS Key Management using the AWS
SDK for Python (Boto3).
The AWS Key Management Service (KMS) is a secure and highly available service
that allows you to create
and manage AWS KMS keys and control their use across a wide range of AWS services
and applications.
KMS provides a centralized and unified approach to managing encryption keys,
making it easier to meet your
data protection and regulatory compliance requirements.

This Basics scenario creates two key types:

- A symmetric encryption key is used to encrypt and decrypt data.
- An asymmetric key used to digitally sign data.

Let's get started...
        """
        )
        q.ask("Press Enter to continue...")

        print(DASHES)
        print(f"1. Create a symmetric KMS key\n")
        print(
            f"First, the program will creates a symmetric KMS key that you can
used to encrypt and decrypt data."
        )
        q.ask("Press Enter to continue...")
        self.key_id = self.key_manager.create_key(key_description)["KeyId"]
        print(f"A symmetric key was successfully created {self.key_id}.")
        q.ask("Press Enter to continue...")
        print(DASHES)
        print(
            """
2. Enable a KMS key

```


By default, when the SDK creates an AWS key, it is enabled. The next bit of code checks to determine if the key is enabled.

```

    """
    )
    q.ask("Press Enter to continue...")
    is_enabled = self.is_key_enabled(self.key_id)
    print(f"Is the key enabled? {is_enabled}")
    if not is_enabled:
        self.key_manager.enable_key(self.key_id)
    q.ask("Press Enter to continue...")
    print(DASHES)
    print(f"3. Encrypt data using the symmetric KMS key")
    plain_text = "Hello, AWS KMS!"
    print(
        f"""

```

One of the main uses of symmetric keys is to encrypt and decrypt data. Next, the code encrypts the string "{plain_text}" with the SYMMETRIC_DEFAULT encryption algorithm.

```

    """
    )
    q.ask("Press Enter to continue...")
    encrypted_text = self.key_encryption.encrypt(self.key_id, plain_text)
    print(DASHES)
    print(f"4. Create an alias")
    print(
        """

```

Now, the program will create an alias for the KMS key. An alias is a friendly name that you can associate with a KMS key. The alias name should be prefixed with 'alias/'.

```

    """
    )
    alias_name = q.ask("Enter an alias name: ", q.non_empty)
    self.alias_manager.create_alias(self.key_id, alias_name)
    print(f"{alias_name} was successfully created.")
    self.alias_name = alias_name
    print(DASHES)
    print(f"5. List all of your aliases")
    q.ask("Press Enter to continue...")
    self.alias_manager.list_aliases(10)
    q.ask("Press Enter to continue...")
    print(DASHES)
    print(f"6. Enable automatic rotation of the KMS key")

```

```
print(
    """
```

By default, when the SDK enables automatic rotation of a KMS key, KMS rotates the key material of the KMS key one year (approximately 365 days) from the enable date and every year thereafter.

```
    """
)
q.ask("Press Enter to continue...")
self.key_manager.enable_key_rotation(self.key_id)
print(DASHES)
print(f"Key rotation has been enabled for key with id {self.key_id}")
print(
    """
```

7. Create a grant

A grant is a policy instrument that allows Amazon Web Services principals to use KMS keys.

It also can allow them to view a KMS key (DescribeKey) and create and manage grants.

When authorizing access to a KMS key, grants are considered along with key policies and IAM policies.

```
    """
)
print(
    """
```

To create a grant you must specify a `account_id`. To specify the grantee `account_id`, use the Amazon Resource Name (ARN) of an AWS `account_id`. Valid principals include AWS accounts, IAM users, IAM roles, federated users, and assumed role users.

```
    """
)
account_id = q.ask(
    "Enter an account_id, or press enter to skip creating a grant... "
)
grant = None
if account_id != "":
    grant = self.grant_manager.create_grant(
        self.key_id,
        account_id,
        [
            "Encrypt",
```

```

        "Decrypt",
        "DescribeKey",
    ],
)
print(f"Grant created successfully with ID: {grant['GrantId']}")

q.ask("Press Enter to continue...")
print(DASHES)
print(DASHES)
print(f"8. List grants for the KMS key")
q.ask("Press Enter to continue...")
self.grant_manager.list_grants(self.key_id)
q.ask("Press Enter to continue...")
print(DASHES)
print(f"9. Revoke the grant")
print(
    """

```

The revocation of a grant immediately removes the permissions and access that the grant had provided.

This means that any account_id (user, role, or service) that was granted access to perform specific

KMS operations on a KMS key will no longer be able to perform those operations.

```

    """
)
q.ask("Press Enter to continue...")

if grant is not None:
    self.grant_manager.revoke_grant(self.key_id, grant["GrantId"])
    print(f"Grant ID: {grant['GrantId']} was successfully revoked!")

q.ask("Press Enter to continue...")
print(DASHES)
print(f"10. Decrypt the data\n")
print(
    """

```

Lets decrypt the data that was encrypted in an early step.

The code uses the same key to decrypt the string that we encrypted earlier in the program.

```

    """
)
q.ask("Press Enter to continue...")
decrypted_data = self.key_encryption.decrypt(self.key_id, encrypted_text)
print(f"Data decrypted successfully for key ID: {self.key_id}")
print(f"Decrypted data: {decrypted_data}")

```

```

q.ask("Press Enter to continue...")
print(DASHES)
print(f"11. Replace a key policy\n")
print(
    """

```

A key policy is a resource policy for a KMS key. Key policies are the primary way to control access to KMS keys. Every KMS key must have exactly one key policy. The statements in the key policy determine who has permission to use the KMS key and how they can use it. You can also use IAM policies and grants to control access to the KMS key, but every KMS key must have a key policy.

By default, when you create a key by using the SDK, a policy is created that gives the AWS account that owns the KMS key full access to the KMS key.

Let's try to replace the automatically created policy with the following policy.

```

{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"AWS": "arn:aws:iam::000000000000:root"},
    "Action": "kms:*",
    "Resource": "*"
  }]
}

"""
)
account_id = q.ask("Enter your account ID or press enter to skip: ")
if account_id != "":
    policy = {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"AWS": f"arn:aws:iam::{account_id}:root"},
                "Action": "kms:*",
                "Resource": "*",
            }
        ],
    }
}

```

```

        self.key_policy.set_new_policy(self.key_id, policy)
        print("Key policy replacement succeeded.")
        q.ask("Press Enter to continue...")
    else:
        print("Skipping replacing the key policy.")

    print(DASHES)
    print(f"12. Get the key policy\n")
    print(
        f"The next bit of code that runs gets the key policy to make sure it
exists."
    )
    q.ask("Press Enter to continue...")
    policy = self.key_policy.get_policy(self.key_id)
    print(f"The key policy is: {policy}")

    q.ask("Press Enter to continue...")
    print(DASHES)
    print(f"13. Create an asymmetric KMS key and sign your data\n")
    print(
        """
        Signing your data with an AWS key can provide several benefits that make
it an attractive option
        for your data signing needs. By using an AWS KMS key, you can leverage
the
        security controls and compliance features provided by AWS,
        which can help you meet various regulatory requirements and enhance the
overall security posture
of your organization.
        """
    )
    q.ask("Press Enter to continue...")
    print(f"Sign and verify data operation succeeded.")
    self.asymmetric_key_id = self.key_manager.create_asymmetric_key()
    message = "Here is the message that will be digitally signed"
    signature = self.key_encryption.sign(self.asymmetric_key_id, message)
    if self.key_encryption.verify(self.asymmetric_key_id, message,
signature):
        print("Signature verification succeeded.")
    else:
        print("Signature verification failed.")

    q.ask("Press Enter to continue...")
    print(DASHES)

```

```

        print(f"14. Tag your symmetric KMS Key\n")
        print(
            """
            By using tags, you can improve the overall management, security, and
            governance of your
            KMS keys, making it easier to organize, track, and control access to your
            encrypted data within
            your AWS environment
            """
        )
        q.ask("Press Enter to continue...")
        self.key_manager.tag_resource(self.key_id, "Environment", "Production")
        self.clean_up()

def is_key_enabled(self, key_id: str) -> bool:
    """
    Check if the key is enabled or not.

    :param key_id: The key to check.
    :return: True if the key is enabled, otherwise False.
    """
    response = self.key_manager.describe_key(key_id)
    return response["Enabled"] is True

def clean_up(self):
    """
    Delete resources created by this scenario.
    """
    if self.alias_name != "":
        print(f"Deleting the alias {self.alias_name}.")
        self.alias_manager.delete_alias(self.alias_name)
    window = 7 # The window in days for a scheduled deletion.
    if self.key_id != "":
        print(
            """
            Warning:
            Deleting a KMS key is a destructive and potentially dangerous operation. When a
            KMS key is deleted,
            all data that was encrypted under the KMS key is unrecoverable.
            """
        )
        if q.ask(
            f"Do you want to delete the key with ID {self.key_id} (y/n)?",
            q.is_yesno,

```

```

        ):
            print(
                f"The key {self.key_id} will be deleted with a window of
{window} days. You can cancel the deletion before"
            )
            print("the window expires.")
            self.key_manager.delete_key(self.key_id, window)
            self.key_id = ""

    if self.asymmetric_key_id != "":
        if q.ask(
            f"Do you want to delete the asymmetric key with ID
{self.asymmetric_key_id} (y/n)?",
            q.is_yesno,
        ):
            print(
                f"The key {self.asymmetric_key_id} will be deleted with a
window of {window} days. You can cancel the deletion before"
            )
            print("the window expires.")
            self.key_manager.delete_key(self.asymmetric_key_id, window)
            self.asymmetric_key_id = ""

if __name__ == "__main__":
    kms_scenario = None
    try:
        kms_client = boto3.client("kms")
        a_key_manager = KeyManager(kms_client)
        a_key_encrypt = KeyEncrypt(kms_client)
        an_alias_manager = AliasManager(kms_client)
        a_grant_manager = GrantManager(kms_client)
        a_key_policy = KeyPolicy(kms_client)
        kms_scenario = KMSScenario(
            key_manager=a_key_manager,
            key_encryption=a_key_encrypt,
            alias_manager=an_alias_manager,
            grant_manager=a_grant_manager,
            key_policy=a_key_policy,
        )
        kms_scenario.kms_scenario()
    except Exception:
        logging.exception("Something went wrong with the demo!")
    if kms_scenario is not None:

```

```
kms_scenario.clean_up()
```

適用於 KMS 金鑰管理的包裝函式類別和方法。

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
    def from_client(cls) -> "KeyManager":
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def create_key(self, key_description: str) -> dict[str, any]:
        """
        Creates a key with a user-provided description.

        :param key_description: A description for the key.
        :return: The key ID.
        """
        try:
            key = self.kms_client.create_key(Description=key_description)
            ["KeyMetadata"]
            self.created_keys.append(key)
            return key
        except ClientError as err:
            logging.error(
                "Couldn't create your key. Here's why: %s",
                err.response["Error"]["Message"],
            )
            raise
```



```
def describe_key(self, key_id: str) -> dict[str, any]:
    """
    Describes a key.

    :param key_id: The ARN or ID of the key to describe.
    :return: Information about the key.
    """

    try:
        key = self.kms_client.describe_key(KeyId=key_id)["KeyMetadata"]
        return key
    except ClientError as err:
        logging.error(
            "Couldn't get key '%s'. Here's why: %s",
            key_id,
            err.response["Error"]["Message"],
        )
        raise

def enable_key_rotation(self, key_id: str) -> None:
    """
    Enables rotation for a key.

    :param key_id: The ARN or ID of the key to enable rotation for.
    """
    try:
        self.kms_client.enable_key_rotation(KeyId=key_id)
    except ClientError as err:
        logging.error(
            "Couldn't enable rotation for key '%s'. Here's why: %s",
            key_id,
            err.response["Error"]["Message"],
        )
        raise

def create_asymmetric_key(self) -> str:
    """
    Creates an asymmetric key in AWS KMS for signing messages.

    :return: The ID of the created key.
    """
    try:
```

```

        key = self.kms_client.create_key(
            KeySpec="RSA_2048", KeyUsage="SIGN_VERIFY", Origin="AWS_KMS"
        )["KeyMetadata"]
        self.created_keys.append(key)
        return key["KeyId"]
    except ClientError as err:
        logger.error(
            "Couldn't create your key. Here's why: %s",
            err.response["Error"]["Message"],
        )
        raise

def tag_resource(self, key_id: str, tag_key: str, tag_value: str) -> None:
    """
    Add or edit tags on a customer managed key.

    :param key_id: The ARN or ID of the key to enable rotation for.
    :param tag_key: Key for the tag.
    :param tag_value: Value for the tag.
    """
    try:
        self.kms_client.tag_resource(
            KeyId=key_id, Tags=[{"TagKey": tag_key, "TagValue": tag_value}]
        )
    except ClientError as err:
        logging.error(
            "Couldn't add a tag for the key '%s'. Here's why: %s",
            key_id,
            err.response["Error"]["Message"],
        )
        raise

def delete_key(self, key_id: str, window: int) -> None:
    """
    Deletes a list of keys.

    Warning:
    Deleting a KMS key is a destructive and potentially dangerous operation.
    When a KMS key is deleted,
    all data that was encrypted under the KMS key is unrecoverable.

    :param key_id: The ARN or ID of the key to delete.

```

```

        :param window: The waiting period, in days, before the KMS key is
        deleted.
        """

        try:
            self.kms_client.schedule_key_deletion(
                KeyId=key_id, PendingWindowInDays=window
            )
        except ClientError as err:
            logging.error(
                "Couldn't delete key %s. Here's why: %s",
                key_id,
                err.response["Error"]["Message"],
            )
            raise

```

適用於 KMS 金鑰別名的包裝函式類別和方法。

```

class AliasManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_key = None

    @classmethod
    def from_client(cls) -> "AliasManager":
        """
        Creates an AliasManager instance with a default KMS client.

        :return: An instance of AliasManager initialized with the default KMS
        client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def create_alias(self, key_id: str, alias: str) -> None:
        """
        Creates an alias for the specified key.

        :param key_id: The ARN or ID of a key to give an alias.
        :param alias: The alias to assign to the key.

```

```

    """
    try:
        self.kms_client.create_alias(AliasName=alias, TargetKeyId=key_id)
    except ClientError as err:
        if err.response["Error"]["Code"] == "AlreadyExistsException":
            logger.error(
                "Could not create the alias %s because it already exists.",
key_id
            )
        else:
            logger.error(
                "Couldn't encrypt text. Here's why: %s",
                err.response["Error"]["Message"],
            )
            raise

def list_aliases(self, page_size: int) -> None:
    """
    Lists aliases for the current account.
    :param page_size: The number of aliases to list per page.
    """
    try:
        alias_paginator = self.kms_client.get_paginator("list_aliases")
        for alias_page in alias_paginator.paginate(
            PaginationConfig={"PageSize": page_size}
        ):
            print(f"Here are {page_size} aliases:")
            pprint(alias_page["Aliases"])
            if alias_page["Truncated"]:
                answer = input(
                    f"Do you want to see the next {page_size} aliases (y/n)?
"
                )
                if answer.lower() != "y":
                    break
            else:
                print("That's all your aliases!")
    except ClientError as err:
        logging.error(
            "Couldn't list your aliases. Here's why: %s",
            err.response["Error"]["Message"],
        )
        raise

```

```

def delete_alias(self, alias: str) -> None:
    """
    Deletes an alias.

    :param alias: The alias to delete.
    """
    try:
        self.kms_client.delete_alias(AliasName=alias)
    except ClientError as err:
        logger.error(
            "Couldn't delete alias %s. Here's why: %s",
            alias,
            err.response["Error"]["Message"],
        )
        raise

```

適用於 KMS 金鑰加密的包裝函式類別和方法。

```

class KeyEncrypt:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyEncrypt":
        """
        Creates a KeyEncrypt instance with a default KMS client.

        :return: An instance of KeyEncrypt initialized with the default KMS
        client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def encrypt(self, key_id: str, text: str) -> bytes:
        """
        Encrypts text by using the specified key.

```

```

        :param key_id: The ARN or ID of the key to use for encryption.
        :param text: The text to encrypt.
        :return: The encrypted version of the text.
        """
        try:
            response = self.kms_client.encrypt(KeyId=key_id,
Plaintext=text.encode())
            print(
                f"The string was encrypted with algorithm
{response['EncryptionAlgorithm']}"
            )
            return response["CiphertextBlob"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "DisabledException":
                logger.error(
                    "Could not encrypt because the key %s is disabled.", key_id
                )
            else:
                logger.error(
                    "Couldn't encrypt text. Here's why: %s",
                    err.response["Error"]["Message"],
                )
            raise

def decrypt(self, key_id: str, cipher_text: bytes) -> str:
    """
    Decrypts text previously encrypted with a key.

    :param key_id: The ARN or ID of the key used to decrypt the data.
    :param cipher_text: The encrypted text to decrypt.
    :return: The decrypted text.
    """
    try:
        return self.kms_client.decrypt(KeyId=key_id,
CiphertextBlob=cipher_text)[
            "Plaintext"
        ].decode()
    except ClientError as err:
        logger.error(
            "Couldn't decrypt your ciphertext. Here's why: %s",
            err.response["Error"]["Message"],
        )
        raise

```

```
def sign(self, key_id: str, message: str) -> str:
    """
    Signs a message with a key.

    :param key_id: The ARN or ID of the key to use for signing.
    :param message: The message to sign.
    :return: The signature of the message.
    """
    try:
        return self.kms_client.sign(
            KeyId=key_id,
            Message=message.encode(),
            SigningAlgorithm="RSASSA_PSS_SHA_256",
        )["Signature"]
    except ClientError as err:
        logger.error(
            "Couldn't sign your message. Here's why: %s",
            err.response["Error"]["Message"],
        )
        raise

def verify(self, key_id: str, message: str, signature: str) -> bool:
    """
    Verifies a signature against a message.

    :param key_id: The ARN or ID of the key used to sign the message.
    :param message: The message to verify.
    :param signature: The signature to verify.
    :return: True when the signature matches the message, otherwise False.
    """
    try:
        response = self.kms_client.verify(
            KeyId=key_id,
            Message=message.encode(),
            Signature=signature,
            SigningAlgorithm="RSASSA_PSS_SHA_256",
        )
        valid = response["SignatureValid"]
        print(f"The signature is {'valid' if valid else 'invalid'}.")
        return valid
    except ClientError as err:
```

```

        if err.response["Error"]["Code"] == "SignatureDoesNotMatchException":
            print("The signature is not valid.")
        else:
            logger.error(
                "Couldn't verify your signature. Here's why: %s",
                err.response["Error"]["Message"],
            )
        raise

```

適用於 KMS 金鑰授權的包裝函式類別和方法。

```

class GrantManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "GrantManager":
        """
        Creates a GrantManager instance with a default KMS client.

        :return: An instance of GrantManager initialized with the default KMS
        client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def create_grant(
        self, key_id: str, principal: str, operations: [str]
    ) -> dict[str, str]:
        """
        Creates a grant for a key that lets a principal generate a symmetric data
        encryption key.

        :param key_id: The ARN or ID of the key.
        :param principal: The principal to grant permission to.
        :param operations: The operations to grant permission for.
        :return: The grant that is created.
        """
        try:
            return self.kms_client.create_grant(

```



```

        KeyId=key_id,
        GranteePrincipal=principal,
        Operations=operations,
    )
except ClientError as err:
    logger.error(
        "Couldn't create a grant on key %s. Here's why: %s",
        key_id,
        err.response["Error"]["Message"],
    )
    raise

def list_grants(self, key_id):
    """
    Lists grants for a key.

    :param key_id: The ARN or ID of the key to query.
    :return: The grants for the key.
    """
    try:
        paginator = self.kms_client.get_paginator("list_grants")
        grants = []
        page_iterator = paginator.paginate(KeyId=key_id)
        for page in page_iterator:
            grants.extend(page["Grants"])

        print(f"Grants for key {key_id}:")
        pprint(grants)
        return grants
    except ClientError as err:
        logger.error(
            "Couldn't list grants for key %s. Here's why: %s",
            key_id,
            err.response["Error"]["Message"],
        )
        raise

def revoke_grant(self, key_id: str, grant_id: str) -> None:
    """
    Revokes a grant so that it can no longer be used.

    :param key_id: The ARN or ID of the key associated with the grant.

```

```

:param grant_id: The ID of the grant to revoke.
"""
try:
    self.kms_client.revoke_grant(KeyId=key_id, GrantId=grant_id)
except ClientError as err:
    logger.error(
        "Couldn't revoke grant %s. Here's why: %s",
        grant_id,
        err.response["Error"]["Message"],
    )
    raise

```

適用於 KMS 金鑰政策的包裝函式類別和方法。

```

class KeyPolicy:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyPolicy":
        """
        Creates a KeyPolicy instance with a default KMS client.

        :return: An instance of KeyPolicy initialized with the default KMS
        client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def set_new_policy(self, key_id: str, policy: dict[str, any]) -> None:
        """
        Sets the policy of a key. Setting a policy entirely overwrites the
        existing
        policy, so care is taken to add a statement to the existing list of
        statements
        rather than simply writing a new policy.

        :param key_id: The ARN or ID of the key to set the policy to.

```

```

        :param policy: A new key policy. The key policy must allow the calling
        principal to make a subsequent
            PutKeyPolicy request on the KMS key. This reduces the risk
        that the KMS key becomes unmanageable
        """

        try:
            self.kms_client.put_key_policy(KeyId=key_id,
            Policy=json.dumps(policy))
        except ClientError as err:
            logger.error(
                "Couldn't set policy for key %s. Here's why %s",
                key_id,
                err.response["Error"]["Message"],
            )
            raise

def get_policy(self, key_id: str) -> dict[str, str]:
    """
    Gets the policy of a key.

    :param key_id: The ARN or ID of the key to query.
    :return: The key policy as a dict.
    """
    if key_id != "":
        try:
            response = self.kms_client.get_key_policy(
                KeyId=key_id,
            )
            policy = json.loads(response["Policy"])
        except ClientError as err:
            logger.error(
                "Couldn't get policy for key %s. Here's why: %s",
                key_id,
                err.response["Error"]["Message"],
            )
            raise
        else:
            pprint(policy)
            return policy
    else:
        print("Skipping get policy demo.")

```

- 如需 API 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS SDK API 參考》中的下列主題。
 - [CreateAlias](#)
 - [CreateGrant](#)
 - [CreateKey](#)
 - [解密](#)
 - [DescribeKey](#)
 - [DisableKey](#)
 - [EnableKey](#)
 - [加密](#)
 - [GetKeyPolicy](#)
 - [ListAliases](#)
 - [ListGrants](#)
 - [ListKeys](#)
 - [RevokeGrant](#)
 - [ScheduleKeyDeletion](#)
 - [符號](#)
 - [TagResource](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

AWS KMS 使用 AWS SDKs 的動作

下列程式碼範例示範如何使用 AWS SDKs 執行個別 AWS KMS 動作。每個範例均包含 GitHub 的連結，您可以在連結中找到設定和執行程式碼的相關說明。

這些摘錄會呼叫 AWS KMS API，是必須在內容中執行之大型程式的程式碼摘錄。您可以在 [AWS KMS 使用 AWS SDKs 案例](#) 中查看內容中的動作。

下列範例僅包含最常使用的動作。如需完整清單，請參閱《[AWS Key Management Service API 參考](#)》。

範例

- [CreateAlias 搭配 AWS SDK 或 CLI 使用](#)
- [CreateGrant 搭配 AWS SDK 或 CLI 使用](#)
- [CreateKey 搭配 AWS SDK 或 CLI 使用](#)
- [Decrypt 搭配 AWS SDK 或 CLI 使用](#)
- [DeleteAlias 搭配 AWS SDK 或 CLI 使用](#)
- [DescribeKey 搭配 AWS SDK 或 CLI 使用](#)
- [DisableKey 搭配 AWS SDK 或 CLI 使用](#)
- [EnableKey 搭配 AWS SDK 或 CLI 使用](#)
- [EnableKeyRotation 搭配 AWS SDK 或 CLI 使用](#)
- [Encrypt 搭配 AWS SDK 或 CLI 使用](#)
- [GenerateDataKey 搭配 AWS SDK 或 CLI 使用](#)
- [GenerateDataKeyWithoutPlaintext 搭配 AWS SDK 或 CLI 使用](#)
- [GenerateRandom 搭配 AWS SDK 或 CLI 使用](#)
- [GetKeyPolicy 搭配 AWS SDK 或 CLI 使用](#)
- [ListAliases 搭配 AWS SDK 或 CLI 使用](#)
- [ListGrants 搭配 AWS SDK 或 CLI 使用](#)
- [ListKeyPolicies 搭配 AWS SDK 或 CLI 使用](#)
- [ListKeys 搭配 AWS SDK 或 CLI 使用](#)
- [PutKeyPolicy 搭配 AWS SDK 或 CLI 使用](#)
- [ReEncrypt 搭配 AWS SDK 或 CLI 使用](#)
- [RetireGrant 搭配 AWS SDK 或 CLI 使用](#)
- [RevokeGrant 搭配 AWS SDK 或 CLI 使用](#)
- [ScheduleKeyDeletion 搭配 AWS SDK 或 CLI 使用](#)
- [Sign 搭配 AWS SDK 或 CLI 使用](#)
- [TagResource 搭配 AWS SDK 或 CLI 使用](#)
- [UpdateAlias 搭配 AWS SDK 或 CLI 使用](#)
- [Verify 搭配 AWS SDK 或 CLI 使用](#)

CreateAlias 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CreateAlias。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon.KeyManagementService;
using Amazon.KeyManagementService.Model;

/// <summary>
/// Creates an alias for an AWS Key Management Service (AWS KMS) key.
/// </summary>
public class CreateAlias
{
    public static async Task Main()
    {
        var client = new AmazonKeyManagementServiceClient();

        // The alias name must start with alias/ and can be
        // up to 256 alphanumeric characters long.
        var aliasName = "alias/ExampleAlias";

        // The value supplied as the TargetKeyId can be either
        // the key ID or key Amazon Resource Name (ARN) of the
        // AWS KMS key.
        var keyId = "1234abcd-12ab-34cd-56ef-1234567890ab";
```

```
var request = new CreateAliasRequest
{
    AliasName = aliasName,
    TargetKeyId = keyId,
};

var response = await client.CreateAliasAsync(request);

if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
{
    Console.WriteLine($"Alias, {aliasName}, successfully created.");
}
else
{
    Console.WriteLine($"Could not create alias.");
}
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [CreateAlias](#)。

CLI

AWS CLI

建立 KMS 金鑰的別名

下列 create-alias 命令會為金鑰 ID 1234abcd-12ab-34cd-56ef-1234567890ab 所識別的 KMS 金鑰建立一個名為 example-alias 的別名。

別名的開頭不可為 alias/。請勿使用以 開頭的別名；alias/aws 這些名稱會保留供使用 AWS。

```
aws kms create-alias \
    --alias-name alias/example-alias \
    --target-key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

此命令不會傳回任何輸出。若要查看新的別名，請使用 list-aliases 命令。

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的 [使用別名](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [CreateAlias](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Creates a custom alias for the specified target key asynchronously.
 *
 * @param targetKeyId the ID of the target key for the alias
 * @param aliasName the name of the alias to create
 * @return a {@link CompletableFuture} that completes when the alias creation
 * operation is finished
 */
public CompletableFuture<Void> createCustomAliasAsync(String targetKeyId,
String aliasName) {
    CreateAliasRequest aliasRequest = CreateAliasRequest.builder()
        .aliasName(aliasName)
        .targetKeyId(targetKeyId)
        .build();

    CompletableFuture<CreateAliasResponse> responseFuture =
getAsyncClient().createAlias(aliasRequest);
    responseFuture.whenComplete((response, exception) -> {
        if (exception == null) {
            logger.info("{} was successfully created.", aliasName);
        } else {
            if (exception instanceof ResourceExistsException) {
                logger.info("Alias [{}] already exists. Moving on...",
aliasName);
            } else if (exception instanceof KmsException kmsEx) {
                throw new RuntimeException("KMS error occurred while creating
alias: " + kmsEx.getMessage(), kmsEx);
            } else {
                throw new RuntimeException("An unexpected error occurred
while creating alias: " + exception.getMessage(), exception);
            }
        }
    });
}
```



```
        }
    }
});

return responseFuture.thenApply(response -> null);
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [CreateAlias](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun createCustomAlias(
    targetKeyIdVal: String?,
    aliasNameVal: String?,
) {
    val request =
        CreateAliasRequest {
            aliasName = aliasNameVal
            targetKeyId = targetKeyIdVal
        }

    KmsClient.fromEnvironment { region = "us-west-2" }.use { kmsClient ->
        kmsClient.createAlias(request)
        println("$aliasNameVal was successfully created")
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [CreateAlias](#)。

PHP

適用於 PHP 的 SDK

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * @param string $keyId
 * @param string $alias
 * @return void
 */
public function createAlias(string $keyId, string $alias)
{
    try{
        $this->client->createAlias([
            'TargetKeyId' => $keyId,
            'AliasName' => $alias,
        ]);
    }catch (KmsException $caught){
        if($caught->getAwsErrorMessage() == "InvalidAliasNameException"){
            echo "The request was rejected because the specified alias name
is not valid.";
        }
        throw $caught;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [CreateAlias](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class AliasManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_key = None

    @classmethod
    def from_client(cls) -> "AliasManager":
        """
        Creates an AliasManager instance with a default KMS client.

        :return: An instance of AliasManager initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def create_alias(self, key_id: str, alias: str) -> None:
        """
        Creates an alias for the specified key.

        :param key_id: The ARN or ID of a key to give an alias.
        :param alias: The alias to assign to the key.
        """
        try:
            self.kms_client.create_alias(AliasName=alias, TargetKeyId=key_id)
        except ClientError as err:
            if err.response["Error"]["Code"] == "AlreadyExistsException":
                logger.error(
                    "Could not create the alias %s because it already exists.",
key_id
                )
```

```

else:
    logger.error(
        "Couldn't encrypt text. Here's why: %s",
        err.response["Error"]["Message"],
    )
    raise

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [CreateAlias](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    " iv_alias_name = 'alias/my-key-alias'
    " iv_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'
    lo_kms->createalias(
        iv_aliasname = iv_alias_name
        iv_targetkeyid = iv_key_id
    ).
    MESSAGE 'Alias created successfully.' TYPE 'I'.
CATCH /aws1/cx_kmsalreadyexistsex.
    MESSAGE 'Alias already exists.' TYPE 'E'.
CATCH /aws1/cx_kmsnotfoundexception.
    MESSAGE 'Key not found.' TYPE 'E'.
CATCH /aws1/cx_kmsinvalidaliasnameex.
    MESSAGE 'Invalid alias name.' TYPE 'E'.
CATCH /aws1/cx_kmskmsinternalex.
    MESSAGE 'An internal error occurred.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [CreateAlias](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreateGrant 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CreateGrant。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
public static async Task Main()
{
    var client = new AmazonKeyManagementServiceClient();

    // The identity that is given permission to perform the operations
    // specified in the grant.
    var grantee = "arn:aws:iam::111122223333:role/ExampleRole";

    // The identifier of the AWS KMS key to which the grant applies. You
    // can use the key ID or the Amazon Resource Name (ARN) of the KMS
key.
    var keyId = "7c9eccc2-38cb-4c4f-9db3-766ee8dd3ad4";

    var request = new CreateGrantRequest
    {
```

```

        GranteePrincipal = grantee,
        KeyId = keyId,

        // A list of operations that the grant allows.
        Operations = new List<string>
        {
            "Encrypt",
            "Decrypt",
        },
    };

    var response = await client.CreateGrantAsync(request);

    string grantId = response.GrantId; // The unique identifier of the
grant.
    string grantToken = response.GrantToken; // The grant token.

    Console.WriteLine($"Id: {grantId}, Token: {grantToken}");
}
}

```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [CreateGrant](#)。

CLI

AWS CLI

建立授權

下列 create-grant 範例會建立授權，其允許 exampleUser 使用者在 1234abcd-12ab-34cd-56ef-1234567890ab 範例 KMS 金鑰上使用 decrypt 命令。淘汰主體是 adminRole 角色。授權使用 EncryptionContextSubset 授權限制條件，允許只在 decrypt 請求中的加密內容包含 "Department": "IT" 鍵值對時，才授予此權限。

```

aws kms create-grant \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --grantee-principal arn:aws:iam::123456789012:user/exampleUser \
  --operations Decrypt \
  --constraints EncryptionContextSubset={Department=IT} \
  --retiring-principal arn:aws:iam::123456789012:role/adminRole

```

輸出：

```
{
  "GrantId":
    "1a2b3c4d2f5e69f440bae30eaec9570bb1fb7358824f9ddfa1aa5a0dab1a59b2",
  "GrantToken": "<grant token here>"
}
```

若要檢視授權的詳細資訊，請使用 `list-grants` 命令。

如需詳細資訊，請參閱 AWS Key Management Service 開發人員指南中的 [AWS KMS 中的授權](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [CreateGrant](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Grants permissions to a specified principal on a customer master key (CMK)
 * asynchronously.
 *
 * @param keyId          The unique identifier for the customer master key
 * (CMK) that the grant applies to.
 * @param granteePrincipal The principal that is given permission to perform
 * the operations that the grant permits on the CMK.
 * @return A {@link CompletableFuture} that, when completed, contains the ID
 * of the created grant.
 * @throws RuntimeException If an error occurs during the grant creation
 * process.
 */
public CompletableFuture<String> grantKeyAsync(String keyId, String
granteePrincipal) {
    List<GrantOperation> grantPermissions = List.of(
        GrantOperation.ENCRYPT,
```

```
        GrantOperation.DECRYPT,
        GrantOperation.DESCRIBE_KEY
    );

    CreateGrantRequest grantRequest = CreateGrantRequest.builder()
        .keyId(keyId)
        .name("grant1")
        .granteePrincipal(granteePrincipal)
        .operations(grantPermissions)
        .build();

    CompletableFuture<CreateGrantResponse> responseFuture =
        getAsyncClient().createGrant(grantRequest);
    responseFuture.whenComplete((response, ex) -> {
        if (ex == null) {
            logger.info("Grant created successfully with ID: " +
                response.grantId());
        } else {
            if (ex instanceof KmsException kmsEx) {
                throw new RuntimeException("Failed to create grant: " +
                    kmsEx.getMessage(), kmsEx);
            } else {
                throw new RuntimeException("An unexpected error occurred: " +
                    ex.getMessage(), ex);
            }
        }
    });

    return responseFuture.thenApply(CreateGrantResponse::grantId);
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [CreateGrant](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。


```

suspend fun createNewGrant(
    keyIdVal: String?,
    granteePrincipalVal: String?,
    operation: String,
): String? {
    val operationOb = GrantOperation.fromValue(operation)
    val grantOperationList = ArrayList<GrantOperation>()
    grantOperationList.add(operationOb)

    val request =
        CreateGrantRequest {
            keyId = keyIdVal
            granteePrincipal = granteePrincipalVal
            operations = grantOperationList
        }

    KmsClient.fromEnvironment { region = "us-west-2" }.use { kmsClient ->
        val response = kmsClient.createGrant(request)
        return response.grantId
    }
}

```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [CreateGrant](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

/**
 * @param string $keyId
 * @param string $granteePrincipal
 * @param array $operations
 * @param array $grantTokens

```

```

    * @return Result
    */
    public function createGrant(string $keyId, string $granteePrincipal, array
$operations, array $grantTokens = [])
    {
        $args = [
            'KeyId' => $keyId,
            'GranteePrincipal' => $granteePrincipal,
            'Operations' => $operations,
        ];
        if($grantTokens){
            $args['GrantTokens'] = $grantTokens;
        }
        try{
            return $this->client->createGrant($args);
        }catch(KmsException $caught){
            if($caught->getAwsErrorMessage() == "InvalidGrantTokenException"){
                echo "The request was rejected because the specified grant token
is not valid.\n";
            }
            throw $caught;
        }
    }
}

```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [CreateGrant](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

class GrantManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client

```

```
@classmethod
def from_client(cls) -> "GrantManager":
    """
    Creates a GrantManager instance with a default KMS client.

    :return: An instance of GrantManager initialized with the default KMS
client.
    """
    kms_client = boto3.client("kms")
    return cls(kms_client)

def create_grant(
    self, key_id: str, principal: str, operations: [str]
) -> dict[str, str]:
    """
    Creates a grant for a key that lets a principal generate a symmetric data
    encryption key.

    :param key_id: The ARN or ID of the key.
    :param principal: The principal to grant permission to.
    :param operations: The operations to grant permission for.
    :return: The grant that is created.
    """
    try:
        return self.kms_client.create_grant(
            KeyId=key_id,
            GranteePrincipal=principal,
            Operations=operations,
        )
    except ClientError as err:
        logger.error(
            "Couldn't create a grant on key %s. Here's why: %s",
            key_id,
            err.response["Error"]["Message"],
        )
        raise
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [CreateGrant](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    " iv_key_id = 'arn:aws:kms:us-  
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'  
    " iv_grantee_principal = 'arn:aws:iam::123456789012:role/my-role'  
    " it_operations contains 'Encrypt', 'Decrypt', 'GenerateDataKey'  
    oo_result = lo_kms->creategrant(  
        iv_keyid = iv_key_id  
        iv_granteeprincipal = iv_grantee_principal  
        it_operations = it_operations  
    ).  
    MESSAGE 'Grant created successfully.' TYPE 'I'.  
CATCH /aws1/cx_kmsdisabledexception.  
    MESSAGE 'The key is disabled.' TYPE 'E'.  
CATCH /aws1/cx_kmsnotfoundexception.  
    MESSAGE 'Key not found.' TYPE 'E'.  
CATCH /aws1/cx_kmskmsinternalex.  
    MESSAGE 'An internal error occurred.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [CreateGrant](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

CreateKey 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 CreateKey。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)
- [使用資料表加密](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon.KeyManagementService;
using Amazon.KeyManagementService.Model;

/// <summary>
/// Shows how to create a new AWS Key Management Service (AWS KMS)
/// key.
/// </summary>
public class CreateKey
{
    public static async Task Main()
    {
        // Note that if you need to create a Key in an AWS Region
        // other than the Region defined for the default user, you need to
        // pass the Region to the client constructor.
        var client = new AmazonKeyManagementServiceClient();

        // The call to CreateKeyAsync will create a symmetrical AWS KMS
        // key. For more information about symmetrical and asymmetrical
        // keys, see:
        //
        // https://docs.aws.amazon.com/kms/latest/developerguide/symm-asymm-
        choose.html
        var response = await client.CreateKeyAsync(new CreateKeyRequest());
```

```
key.        // The KeyMetadata object contains information about the new AWS KMS
            KeyMetadata keyMetadata = response.KeyMetadata;

            if (keyMetadata is not null)
            {
                Console.WriteLine($"KMS Key: {keyMetadata.KeyId} was successfully
created.");
            }
            else
            {
                Console.WriteLine("Could not create KMS Key.");
            }
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [CreateKey](#)。

CLI

AWS CLI

範例 1：在 KMS 中建立客戶受管 AWS KMS 金鑰

以下 create-key 範例會建立對稱加密 KMS 金鑰。

若要建立基本 KMS 金鑰 (對稱加密金鑰)，您不需要指定任何參數。這些參數的預設值會建立對稱加密金鑰。

由於此命令未指定金鑰政策，KMS 金鑰會取得適用於以程式設計方式建立之 KMS 金鑰的[預設金鑰政策](#)。若要檢視金鑰政策，請使用 get-key-policy 命令。若要變更金鑰政策，請使用 put-key-policy 命令。

```
aws kms create-key
```

create-key 命令會傳回金鑰中繼資料，包括新 KMS 金鑰的金鑰 ID 和 ARN。您可以使用這些值來識別其他 KMS 操作中的 AWS KMS 金鑰。輸出不包含標籤。若要檢視 KMS 金鑰的標籤，請使用 list-resource-tags command。

輸出：

```
{
  "KeyMetadata": {
    "AWSAccountId": "111122223333",
    "Arn": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "CreationDate": "2017-07-05T14:04:55-07:00",
    "CurrentKeyMaterialId":
      "0b7fd7ddbac6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "Description": "",
    "Enabled": true,
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",
    "KeySpec": "SYMMETRIC_DEFAULT",
    "KeyState": "Enabled",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "MultiRegion": false,
    "Origin": "AWS_KMS"
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ]
  }
}
```

注意：create-key 命令不允許您指定別名。若要為新的 KMS 金鑰建立別名，請使用 create-alias 命令。

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的[建立金鑰](#)。

範例 2：建立用於加密和解密的非對稱 RSA KMS 金鑰

下列 create-key 範例會建立 KMS 金鑰，其中包含用於加密和解密的非對稱 RSA 金鑰對。建立金鑰之後，就無法變更金鑰規格和金鑰用量：

```
aws kms create-key \
  --key-spec RSA_4096 \
  --key-usage ENCRYPT_DECRYPT
```

輸出：

```
{
```

```

    "KeyMetadata": {
      "Arn": "arn:aws:kms:us-east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "AWSAccountId": "111122223333",
      "CreationDate": "2021-04-05T14:04:55-07:00",
      "CustomerMasterKeySpec": "RSA_4096",
      "Description": "",
      "Enabled": true,
      "EncryptionAlgorithms": [
        "RSAES_OAEP_SHA_1",
        "RSAES_OAEP_SHA_256"
      ],
      "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
      "KeyManager": "CUSTOMER",
      "KeySpec": "RSA_4096",
      "KeyState": "Enabled",
      "KeyUsage": "ENCRYPT_DECRYPT",
      "MultiRegion": false,
      "Origin": "AWS_KMS"
    }
  }
}

```

如需詳細資訊，請參閱 [Key Management Service 開發人員指南中的 AWS KMS 中的非對稱金鑰](#)。AWS

範例 3：建立用於簽署和驗證的非對稱橢圓曲線 KMS 金鑰

建立非對稱 KMS 金鑰，其中包含用於簽署和驗證的非對稱橢圓曲線 (ECC) 金鑰對。即使 SIGN_VERIFY 是 ECC KMS 金鑰的唯一有效值，仍需要 --key-usage 參數。建立金鑰之後，就無法變更金鑰規格和金鑰用量：

```

aws kms create-key \
  --key-spec ECC_NIST_P521 \
  --key-usage SIGN_VERIFY

```

輸出：

```

{
  "KeyMetadata": {
    "Arn": "arn:aws:kms:us-east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "AWSAccountId": "111122223333",

```



```

    "CreationDate": "2019-12-02T07:48:55-07:00",
    "CustomerMasterKeySpec": "ECC_NIST_P521",
    "Description": "",
    "Enabled": true,
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",
    "KeySpec": "ECC_NIST_P521",
    "KeyState": "Enabled",
    "KeyUsage": "SIGN_VERIFY",
    "MultiRegion": false,
    "Origin": "AWS_KMS",
    "SigningAlgorithms": [
        "ECDSA_SHA_512"
    ]
  }
}

```

如需詳細資訊，請參閱 [Key Management Service 開發人員指南](#) 中的 [AWS KMS 中的非對稱金鑰](#)。AWS

範例 4：建立用於簽署和驗證的非對稱 ML-DSA KMS 金鑰

此範例會建立用於簽署和驗證的模組格線數位簽章演算法 (ML-DSA) 金鑰。即使 SIGN_VERIFY 是 ML-DSA 金鑰的唯一有效值，仍需要 key-usage 參數。

```

aws kms create-key \
  --key-spec ML_DSA_65 \
  --key-usage SIGN_VERIFY

```

輸出：

```

{
  "KeyMetadata": {
    "Arn": "arn:aws:kms:us-east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "AWSAccountId": "111122223333",
    "CreationDate": "2019-12-02T07:48:55-07:00",
    "Description": "",
    "Enabled": true,
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",
    "KeySpec": "ML_DSA_65",
    "KeyState": "Enabled",

```

```

        "KeyUsage": "SIGN_VERIFY",
        "MultiRegion": false,
        "Origin": "AWS_KMS",
        "SigningAlgorithms": [
            "ML_DSA_SHAKE_256"
        ]
    }
}

```

如需詳細資訊，請參閱 [Key Management Service 開發人員指南](#) 中的 [AWS KMS 中的非對稱金鑰](#)。AWS

範例 5：建立 HMAC KMS 金鑰

以下 create-key 範例會建立 384 位元 HMAC KMS 金鑰。--key-usage 參數的 GENERATE_VERIFY_MAC 值是必要的，即使它是 HMAC KMS 金鑰的唯一有效值。

```

aws kms create-key \
  --key-spec HMAC_384 \
  --key-usage GENERATE_VERIFY_MAC

```

輸出：

```

{
  "KeyMetadata": {
    "Arn": "arn:aws:kms:us-east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "AWSAccountId": "111122223333",
    "CreationDate": "2022-04-05T14:04:55-07:00",
    "CustomerMasterKeySpec": "HMAC_384",
    "Description": "",
    "Enabled": true,
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",
    "KeySpec": "HMAC_384",
    "KeyState": "Enabled",
    "KeyUsage": "GENERATE_VERIFY_MAC",
    "MacAlgorithms": [
      "HMAC_SHA_384"
    ],
    "MultiRegion": false,
    "Origin": "AWS_KMS"
  }
}

```

```
}  
}
```

如需詳細資訊，請參閱《[金鑰管理服務開發人員指南](#)》中的 [AWS KMS 中的 HMAC 金鑰](#)。
AWS

範例 6：建立多區域主要 KMS 金鑰

以下 create-key 範例會建立多區域主要對稱加密金鑰。由於所有參數的預設值都會建立對稱加密金鑰，此 KMS 金鑰只需要 --multi-region 參數。在 AWS CLI 中，若要指出布林值參數為 true，只要指定參數名稱即可。

```
aws kms create-key \  
  --multi-region
```

輸出：

```
{  
  "KeyMetadata": {  
    "Arn": "arn:aws:kms:us-west-2:111122223333:key/  
mrk-1234abcd12ab34cd56ef12345678990ab",  
    "AWSAccountId": "111122223333",  
    "CreationDate": "2021-09-02T016:15:21-09:00",  
    "CurrentKeyMaterialId":  
"0b7fd7ddb6c6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",  
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",  
    "Description": "",  
    "Enabled": true,  
    "EncryptionAlgorithms": [  
      "SYMMETRIC_DEFAULT"  
    ],  
    "KeyId": "mrk-1234abcd12ab34cd56ef12345678990ab",  
    "KeyManager": "CUSTOMER",  
    "KeySpec": "SYMMETRIC_DEFAULT",  
    "KeyState": "Enabled",  
    "KeyUsage": "ENCRYPT_DECRYPT",  
    "MultiRegion": true,  
    "MultiRegionConfiguration": {  
      "MultiRegionKeyType": "PRIMARY",  
      "PrimaryKey": {  
        "Arn": "arn:aws:kms:us-west-2:111122223333:key/  
mrk-1234abcd12ab34cd56ef12345678990ab",
```

```
        "Region": "us-west-2"
      },
      "ReplicaKeys": []
    },
    "Origin": "AWS_KMS"
  }
}
```

如需詳細資訊，請參閱 [Key Management Service 開發人員指南中的 AWS KMS 中的非對稱金鑰](#)。AWS

範例 7：為匯入的金鑰材料建立 KMS 金鑰

下列 create-key 範例會建立不含金鑰材料的 KMS 金鑰。完成操作後，您可以將自己的金鑰材料匯入 KMS 金鑰。若要建立此 KMS 金鑰，請將 --origin 參數設定為 EXTERNAL。

```
aws kms create-key \
  --origin EXTERNAL
```

輸出：

```
{
  "KeyMetadata": {
    "Arn": "arn:aws:kms:us-east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "AWSAccountId": "111122223333",
    "CreationDate": "2019-12-02T07:48:55-07:00",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "Description": "",
    "Enabled": false,
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ],
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyManager": "CUSTOMER",
    "KeySpec": "SYMMETRIC_DEFAULT",
    "KeyState": "PendingImport",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "MultiRegion": false,
    "Origin": "EXTERNAL"
  }
}
```

如需詳細資訊，請參閱 [Key Management Service 開發人員指南](#) 中的在 AWS KMS 金鑰中匯入金鑰材料。AWS

範例 6：在 an AWS CloudHSM 金鑰存放區中建立 KMS 金鑰

下列 create-key 範例會在指定的 AWS CloudHSM 金鑰存放區中建立 KMS 金鑰。操作會在 KMS 中建立 AWS KMS 金鑰及其中繼資料，並在與自訂金鑰存放區相關聯的 AWS CloudHSM 叢集中建立金鑰材料。--custom-key-store-id 和 --origin 是必要參數。

```
aws kms create-key \  
  --origin AWS_CLOUDHSM \  
  --custom-key-store-id cks-1234567890abcdef0
```

輸出：

```
{  
  "KeyMetadata": {  
    "Arn": "arn:aws:kms:us-east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",  
    "AWSAccountId": "111122223333",  
    "CloudHsmClusterId": "cluster-1a23b4cdefg",  
    "CreationDate": "2019-12-02T07:48:55-07:00",  
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",  
    "CustomKeyId": "cks-1234567890abcdef0",  
    "Description": "",  
    "Enabled": true,  
    "EncryptionAlgorithms": [  
      "SYMMETRIC_DEFAULT"  
    ],  
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",  
    "KeyManager": "CUSTOMER",  
    "KeySpec": "SYMMETRIC_DEFAULT",  
    "KeyState": "Enabled",  
    "KeyUsage": "ENCRYPT_DECRYPT",  
    "MultiRegion": false,  
    "Origin": "AWS_CLOUDHSM"  
  }  
}
```

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的 [AWS CloudHSM 金鑰存放區](#)。

範例 8：在外部金鑰存放區中建立 KMS 金鑰

下列 `create-key` 範例會在指定的外部金鑰存放區中建立 KMS 金鑰。此命令需要用到 `--custom-key-store-id`、`--origin` 和 `--xks-key-id` 參數。

`--xks-key-id` 參數會在外部金鑰管理員中，指定現有對稱加密金鑰的 ID。此金鑰用作 KMS 金鑰的外部金鑰材料。`--origin` 參數的值必須是 `EXTERNAL_KEY_STORE`。`custom-key-store-id` 參數必須識別連接到其外部金鑰存放區代理的外部金鑰存放區。

```
aws kms create-key \  
  --origin EXTERNAL_KEY_STORE \  
  --custom-key-store-id cks-9876543210fedcba9 \  
  --xks-key-id bb8562717f809024
```

輸出：

```
{  
  "KeyMetadata": {  
    "Arn": "arn:aws:kms:us-east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",  
    "AWSAccountId": "111122223333",  
    "CreationDate": "2022-12-02T07:48:55-07:00",  
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",  
    "CustomKeyId": "cks-9876543210fedcba9",  
    "Description": "",  
    "Enabled": true,  
    "EncryptionAlgorithms": [  
      "SYMMETRIC_DEFAULT"  
    ],  
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",  
    "KeyManager": "CUSTOMER",  
    "KeySpec": "SYMMETRIC_DEFAULT",  
    "KeyState": "Enabled",  
    "KeyUsage": "ENCRYPT_DECRYPT",  
    "MultiRegion": false,  
    "Origin": "EXTERNAL_KEY_STORE",  
    "XksKeyConfiguration": {  
      "Id": "bb8562717f809024"  
    }  
  }  
}
```

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的[外部金鑰存放區](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [CreateKey](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Creates a new symmetric encryption key asynchronously.
 *
 * @param keyDesc the description of the key to be created
 * @return a {@link CompletableFuture} that completes with the ID of the
 * newly created key
 * @throws RuntimeException if an error occurs while creating the key
 */
public CompletableFuture<String> createKeyAsync(String keyDesc) {
    CreateKeyRequest keyRequest = CreateKeyRequest.builder()
        .description(keyDesc)
        .keySpec(KeySpec.SYMMETRIC_DEFAULT)
        .keyUsage(KeyUsageType.ENCRYPT_DECRYPT)
        .build();

    return getAsyncClient().createKey(keyRequest)
        .thenApply(resp -> resp.keyMetadata().keyId())
        .exceptionally(ex -> {
            throw new RuntimeException("An error occurred while creating the
key: " + ex.getMessage(), ex);
        });
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [CreateKey](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun createKey(keyDesc: String?): String? {
    val request =
        CreateKeyRequest {
            description = keyDesc
            customerMasterKeySpec = CustomerMasterKeySpec.SymmetricDefault
            keyUsage = KeyUsageType.fromValue("ENCRYPT_DECRYPT")
        }

    KmsClient.fromEnvironment { region = "us-west-2" }.use { kmsClient ->
        val result = kmsClient.createKey(request)
        println("Created a customer key with id " + result.keyMetadata?.arn)
        return result.keyMetadata?.keyId
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [CreateKey](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
```



```

    * @param string $keySpec
    * @param string $keyUsage
    * @param string $description
    * @return array
    */
    public function createKey(string $keySpec = "", string $keyUsage = "", string
    $description = "Created by the SDK for PHP")
    {
        $parameters = ['Description' => $description];
        if($keySpec && $keyUsage){
            $parameters['KeySpec'] = $keySpec;
            $parameters['KeyUsage'] = $keyUsage;
        }
        try {
            $result = $this->client->createKey($parameters);
            return $result['KeyMetadata'];
        }catch(KmsException $caught){
            // Check for error specific to createKey operations
            if ($caught->getAwsErrorMessage() == "LimitExceededException"){
                echo "The request was rejected because a quota was exceeded. For
                more information, see Quotas in the Key Management Service Developer Guide.";
            }
            throw $caught;
        }
    }
}

```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [CreateKey](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

class KeyManager:
    def __init__(self, kms_client):

```

```
self.kms_client = kms_client
self.created_keys = []

@classmethod
def from_client(cls) -> "KeyManager":
    """
    Creates a KeyManager instance with a default KMS client.

    :return: An instance of KeyManager initialized with the default KMS
client.
    """
    kms_client = boto3.client("kms")
    return cls(kms_client)

def create_key(self, key_description: str) -> dict[str, any]:
    """
    Creates a key with a user-provided description.

    :param key_description: A description for the key.
    :return: The key ID.
    """
    try:
        key = self.kms_client.create_key(Description=key_description)
["KeyMetadata"]
        self.created_keys.append(key)
        return key
    except ClientError as err:
        logging.error(
            "Couldn't create your key. Here's why: %s",
            err.response["Error"]["Message"],
        )
        raise
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [CreateKey](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
require 'aws-sdk-kms' # v2: require 'aws-sdk'

# Create a AWS KMS key.
# As long we are only encrypting small amounts of data (4 KiB or less) directly,
# a KMS key is fine for our purposes.
# For larger amounts of data,
# use the KMS key to encrypt a data encryption key (DEK).

client = Aws::KMS::Client.new

resp = client.create_key({
  tags: [
    {
      tag_key: 'CreatedBy',
      tag_value: 'ExampleUser'
    }
  ]
})

puts resp.key_metadata.key_id
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [CreateKey](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
async fn make_key(client: &Client) -> Result<(), Error> {
    let resp = client.create_key().send().await?;

    let id = resp.key_metadata.as_ref().unwrap().key_id();

    println!("Key: {}", id);

    Ok(())
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Rust API 參考》中的 [CreateKey](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.
    " iv_description = 'Created by the AWS SDK for SAP ABAP'
    oo_result = lo_kms->createkey( iv_description = iv_description ).
    MESSAGE 'KMS key created successfully.' TYPE 'I'.
CATCH /aws1/cx_kmskmsinternalex.
    MESSAGE 'An internal error occurred.' TYPE 'E'.
CATCH /aws1/cx_kmslimitexceedex.
```

```
MESSAGE 'Limit exceeded for KMS resources.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [CreateKey](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Decrypt 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 Decrypt。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

CLI

AWS CLI

範例 1：使用對稱 KMS 金鑰 (Linux 和 macOS) 將加密的訊息解密

下列 decrypt 命令範例示範使用 CLI AWS 解密資料的建議方法。此版本示範如何在對稱 KMS 金鑰下解密資料。

在檔案中提供密文。--ciphertext-blob 參數的值中，使用 fileb:// 字首，其會告知 CLI 從二進位檔案讀取資料。如果檔案不在目前的目錄中，請輸入檔案的完整路徑。如需從檔案讀取 AWS CLI 參數值的詳細資訊，請參閱《AWS 命令列界面使用者指南》中的從檔案 <<https://docs.aws.amazon.com/cli/latest/userguide/cli-usage-parameters-file.html>> 載入 AWS CLI 參數，以及《AWS 命令列工具部落格》中的本機檔案參數最佳實務 <<https://aws.amazon.com/blogs/developer/best-practices-for-local-file-parameters/>>。指定 KMS 金鑰以解密加密文字。使用對稱 KMS 金鑰解密時，不需要 --key-id 參數。AWS KMS 可以取得用於加密加密加密文字中中繼資料之 KMS 金鑰的金鑰 ID。但是指定您正在使用的 KMS 金鑰永遠是最佳實務。此實務可確保您使用想要的 KMS 金鑰，並防止不小心使用您不信任的 KMS 金鑰來解密加密文字。請求純文字輸出做為文字值。--query 參數會告知 CLI 僅從輸出取得 Plaintext 欄位的值。--output 參數會以純文字傳回輸出。Base64 將純文字解碼，並儲存在檔案中。下列範例會將 Plaintext 參數的管道符號 (|) 值輸送至 Base64 公用程式，以將其解碼。然後，將解碼的輸出重新導向 (>) 至 ExamplePlaintext 檔案。

執行此命令之前，請將範例金鑰 ID 取代為來自您 AWS 帳戶的有效金鑰 ID。

```
aws kms decrypt \  
  --ciphertext-blob fileb://ExampleEncryptedFile \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --output text \  
  --query Plaintext | base64 \  
  --decode > ExamplePlaintextFile
```

此命令不會產生輸出。來自 decrypt 命令的輸出經過 base64 解碼，並儲存在檔案中。

如需詳細資訊，請參閱《AWS Key Management Service API 參考》中的[解密](#)。

範例 2：使用對稱 KMS 金鑰解密加密的訊息 (Windows 命令提示)

下列範例與上一個範例相同，唯一不同的是它使用 certutil 公用程式對純文字資料進行 Base64 解碼。此程序需要兩個命令，如下列範例所示。

執行此命令之前，請將範例金鑰 ID 取代為來自您 AWS 帳戶的有效金鑰 ID。

```
aws kms decrypt ^  
  --ciphertext-blob fileb://ExampleEncryptedFile ^  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab ^  
  --output text ^  
  --query Plaintext > ExamplePlaintextFile.base64
```

執行 certutil 命令。

```
certutil -decode ExamplePlaintextFile.base64 ExamplePlaintextFile
```

輸出：

```
Input Length = 18  
Output Length = 12  
CertUtil: -decode command completed successfully.
```

如需詳細資訊，請參閱《AWS Key Management Service API 參考》中的[解密](#)。

範例 3：使用非對稱 KMS 金鑰 (Linux 和 macOS) 解密加密的訊息

下列 decrypt 命令範例示範如何將以 RSA 非對稱 KMS 金鑰加密的資料解密。

使用非對稱 KMS 金鑰時，需要指定用於加密純文字的演算法的 `encryption-algorithm` 參數。

執行此命令之前，請將範例金鑰 ID 取代為來自您 AWS 帳戶的有效金鑰 ID。

```
aws kms decrypt \
  --ciphertext-blob fileb://ExampleEncryptedFile \
  --key-id 0987dcba-09fe-87dc-65ba-ab0987654321 \
  --encryption-algorithm RSAES_OAEP_SHA_256 \
  --output text \
  --query Plaintext | base64 \
  --decode > ExamplePlaintextFile
```

此命令不會產生輸出。來自 `decrypt` 命令的輸出經過 base64 解碼，並儲存在檔案中。

如需詳細資訊，請參閱 [Key Management Service 開發人員指南中的 AWS KMS 中的非對稱金鑰](#)。AWS

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [Decrypt](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Asynchronously decrypts the given encrypted data using the specified key
 * ID.
 *
 * @param encryptedData The encrypted data to be decrypted.
 * @param keyId The ID of the key to be used for decryption.
 * @return A CompletableFuture that, when completed, will contain the
 *         decrypted data as a String.
 *
 *         If an error occurs during the decryption process, the
 *         CompletableFuture will complete
 *
 *         exceptionally with the error, and the method will return an empty
 *         String.
 */
```

```

    */
    public CompletableFuture<String> decryptDataAsync(SdkBytes encryptedData,
String keyId) {
        DecryptRequest decryptRequest = DecryptRequest.builder()
            .ciphertextBlob(encryptedData)
            .keyId(keyId)
            .build();

        CompletableFuture<DecryptResponse> responseFuture =
getAsyncClient().decrypt(decryptRequest);
        responseFuture.whenComplete((decryptResponse, exception) -> {
            if (exception == null) {
                logger.info("Data decrypted successfully for key ID: " + keyId);
            } else {
                if (exception instanceof KmsException kmsEx) {
                    throw new RuntimeException("KMS error occurred while
decrypting data: " + kmsEx.getMessage(), kmsEx);
                } else {
                    throw new RuntimeException("An unexpected error occurred
while decrypting data: " + exception.getMessage(), exception);
                }
            }
        });

        return responseFuture.thenApply(decryptResponse ->
decryptResponse.plaintext().asString(StandardCharsets.UTF_8));
    }

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [Decrypt](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun encryptData(keyIdValue: String): ByteArray? {
```



```
val text = "This is the text to encrypt by using the AWS KMS Service"
val myBytes: ByteArray = text.toByteArray()

val encryptRequest =
    EncryptRequest {
        keyId = keyIdValue
        plaintext = myBytes
    }

KmsClient.fromEnvironment { region = "us-west-2" }.use { kmsClient ->
    val response = kmsClient.encrypt(encryptRequest)
    val algorithm: String = response.encryptionAlgorithm.toString()
    println("The encryption algorithm is $algorithm")

    // Return the encrypted data.
    return response.ciphertextBlob
}

suspend fun decryptData(
    encryptedDataVal: ByteArray?,
    keyIdVal: String?,
) {
    val decryptRequest =
        DecryptRequest {
            ciphertextBlob = encryptedDataVal
            keyId = keyIdVal
        }

    KmsClient { region = "us-west-2" }.use { kmsClient ->
        val decryptResponse = kmsClient.decrypt(decryptRequest)
        val myVal = decryptResponse.plaintext

        // Print the decrypted data.
        print(myVal)
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [Decrypt](#)。

PHP

適用於 PHP 的 SDK

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * @param string $keyId
 * @param string $ciphertext
 * @param string $algorithm
 * @return Result
 */
public function decrypt(string $keyId, string $ciphertext, string $algorithm
= "SYMMETRIC_DEFAULT")
{
    try{
        return $this->client->decrypt([
            'CiphertextBlob' => $ciphertext,
            'EncryptionAlgorithm' => $algorithm,
            'KeyId' => $keyId,
        ]);
    }catch(KmsException $caught){
        echo "There was a problem decrypting the data: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [Decrypt](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class KeyEncrypt:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyEncrypt":
        """
        Creates a KeyEncrypt instance with a default KMS client.

        :return: An instance of KeyEncrypt initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def decrypt(self, key_id: str, cipher_text: bytes) -> str:
        """
        Decrypts text previously encrypted with a key.

        :param key_id: The ARN or ID of the key used to decrypt the data.
        :param cipher_text: The encrypted text to decrypt.
        :return: The decrypted text.
        """
        try:
            return self.kms_client.decrypt(KeyId=key_id,
CiphertextBlob=cipher_text)[
                "Plaintext"
            ].decode()
        except ClientError as err:
            logger.error(
                "Couldn't decrypt your ciphertext. Here's why: %s",
```

```
        err.response["Error"]["Message"],
    )
    raise
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [Decrypt](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
require 'aws-sdk-kms' # v2: require 'aws-sdk'

# Decrypted blob

blob =
  '01020200785d68faeec386af1057904926253051eb2919d3c16078badf65b808b26dd057c101747cadf3593'
blob_packed = [blob].pack('H*')

client = Aws::KMS::Client.new(region: 'us-west-2')

resp = client.decrypt({
  ciphertext_blob: blob_packed
})

puts 'Raw text: '
puts resp.plaintext
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [Decrypt](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
async fn decrypt_key(client: &Client, key: &str, filename: &str) -> Result<(),
Error> {
    // Open input text file and get contents as a string
    // input is a base-64 encoded string, so decode it:
    let data = fs::read_to_string(filename)
        .map(|input| {
            base64::decode(input).expect("Input file does not contain valid base
64 characters.")
        })
        .map(Blob::new);

    let resp = client
        .decrypt()
        .key_id(key)
        .ciphertext_blob(data.unwrap())
        .send()
        .await?;

    let inner = resp.plaintext.unwrap();
    let bytes = inner.as_ref();

    let s = String::from_utf8(bytes.to_vec()).expect("Could not convert to
UTF-8");

    println();
    println!("Decoded string:");
    println!("{}", s);

    Ok(())
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Rust API 參考》中的 [Decrypt](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    " iv_key_id = 'arn:aws:kms:us-  
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'  
    " iv_ciphertext_blob contains the encrypted data  
    oo_result = lo_kms->decrypt(  
        iv_keyid = iv_key_id  
        iv_ciphertextblob = iv_ciphertext_blob  
    ).  
    MESSAGE 'Text decrypted successfully.' TYPE 'I'.  
CATCH /aws1/cx_kmsdisabledexception.  
    MESSAGE 'The key is disabled.' TYPE 'E'.  
CATCH /aws1/cx_kmsincorrectkeyex.  
    MESSAGE 'Incorrect key for decryption.' TYPE 'E'.  
CATCH /aws1/cx_kmsnotfoundexception.  
    MESSAGE 'Key not found.' TYPE 'E'.  
CATCH /aws1/cx_kmskmsinternalex.  
    MESSAGE 'An internal error occurred.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [解密](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteAlias 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteAlias。

CLI

AWS CLI

刪除 AWS KMS 別名

以下 `delete-alias` 範例會刪除別名 `alias/example-alias`。別名名稱必須以別名/ 開頭。

```
aws kms delete-alias \  
  --alias-name alias/example-alias
```

此命令不會產生輸出。要尋找別名，請使用 `list-aliases` 命令。

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的[刪除別名](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DeleteAlias](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**  
 * Deletes a specific KMS alias asynchronously.  
 *  
 * @param aliasName the name of the alias to be deleted  
 * @return a {@link CompletableFuture} representing the asynchronous  
operation of deleting the specified alias  
 */  
public CompletableFuture<Void> deleteSpecificAliasAsync(String aliasName) {  
    DeleteAliasRequest deleteAliasRequest = DeleteAliasRequest.builder()  
        .aliasName(aliasName)  
        .build();  
  
    return getAsyncClient().deleteAlias(deleteAliasRequest)  
        .thenRun(() -> {  
            logger.info("Alias {} has been deleted successfully", aliasName);  
        });  
}
```

```
    })
    .exceptionally(throwable -> {
        throw new RuntimeException("Failed to delete alias: " +
aliasName, throwable);
    });
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DeleteAlias](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * @param string $aliasName
 * @return void
 */
public function deleteAlias(string $aliasName)
{
    try {
        $this->client->deleteAlias([
            'AliasName' => $aliasName,
        ]);
    } catch (KmsException $caught){
        echo "There was a problem deleting the alias: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [DeleteAlias](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class AliasManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_key = None

    @classmethod
    def from_client(cls) -> "AliasManager":
        """
        Creates an AliasManager instance with a default KMS client.

        :return: An instance of AliasManager initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def delete_alias(self, alias: str) -> None:
        """
        Deletes an alias.

        :param alias: The alias to delete.
        """
        try:
            self.kms_client.delete_alias(AliasName=alias)
        except ClientError as err:
            logger.error(
                "Couldn't delete alias %s. Here's why: %s",
                alias,
                err.response["Error"]["Message"],
            )
            raise
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [DeleteAlias](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    " iv_alias_name = 'alias/my-key-alias'  
    lo_kms->deletealias( iv_aliasname = iv_alias_name ).  
    MESSAGE 'Alias deleted successfully.' TYPE 'I'.  
CATCH /aws1/cx_kmsnotfoundexception.  
    MESSAGE 'Alias not found.' TYPE 'E'.  
CATCH /aws1/cx_kmskmsinternalex.  
    MESSAGE 'An internal error occurred.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [DeleteAlias](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DescribeKey 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DescribeKey。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon.KeyManagementService;
using Amazon.KeyManagementService.Model;

/// <summary>
/// Retrieve information about an AWS Key Management Service (AWS KMS) key.
/// You can supply either the key Id or the key Amazon Resource Name (ARN)
/// to the DescribeKeyRequest KeyId property.
/// </summary>
public class DescribeKey
{
    public static async Task Main()
    {
        var keyId = "7c9eccc2-38cb-4c4f-9db3-766ee8dd3ad4";
        var request = new DescribeKeyRequest
        {
            KeyId = keyId,
        };

        var client = new AmazonKeyManagementServiceClient();

        var response = await client.DescribeKeyAsync(request);
        var metadata = response.KeyMetadata;

        Console.WriteLine($"{metadata.KeyId} created on:
{metadata.CreationDate}");
        Console.WriteLine($"State: {metadata.KeyState}");
        Console.WriteLine($"{metadata.Description}");
    }
}
```

```
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [DescribeKey](#)。

CLI

AWS CLI

範例 1：尋找 KMS 金鑰的詳細資訊

下列 `describe-key` 範例會取得範例帳戶和區域中 Amazon S3 AWS 受管金鑰的詳細資訊。您可以使用此命令來尋找 AWS 受管金鑰和客戶受管金鑰的詳細資訊。

若要指定 KMS 金鑰，請使用 `key-id` 參數。此範例使用別名的名稱值，但您可以在此命令中使用金鑰 ID、金鑰 ARN、別名的名稱或別名 ARN。

```
aws kms describe-key \
  --key-id alias/aws/s3
```

輸出：

```
{
  "KeyMetadata": {
    "AWSAccountId": "846764612917",
    "KeyId": "b8a9477d-836c-491f-857e-07937918959b",
    "Arn": "arn:aws:kms:us-west-2:846764612917:key/b8a9477d-836c-491f-857e-07937918959b",
    "CurrentKeyMaterialId":
    "0b7fd7ddb6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",
    "CreationDate": 2017-06-30T21:44:32.140000+00:00,
    "Enabled": true,
    "Description": "Default KMS key that protects my S3 objects when no other
key is defined",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "KeyState": "Enabled",
    "Origin": "AWS_KMS",
    "KeyManager": "AWS",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
```

```
    ]
  }
}
```

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的[檢視金鑰](#)。

範例 2：取得 RSA 非對稱 KMS 金鑰的詳細資訊

下列 describe-key 範例會取得用於簽署和驗證的非對稱 RSA KMS 金鑰的詳細資訊。

```
aws kms describe-key \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

輸出：

```
{
  "KeyMetadata": {
    "AWSAccountId": "111122223333",
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "Arn": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "CreationDate": "2019-12-02T19:47:14.861000+00:00",
    "CustomerMasterKeySpec": "RSA_2048",
    "Enabled": false,
    "Description": "",
    "KeyState": "Disabled",
    "Origin": "AWS_KMS",
    "MultiRegion": false,
    "KeyManager": "CUSTOMER",
    "KeySpec": "RSA_2048",
    "KeyUsage": "SIGN_VERIFY",
    "SigningAlgorithms": [
      "RSASSA_PKCS1_V1_5_SHA_256",
      "RSASSA_PKCS1_V1_5_SHA_384",
      "RSASSA_PKCS1_V1_5_SHA_512",
      "RSASSA_PSS_SHA_256",
      "RSASSA_PSS_SHA_384",
      "RSASSA_PSS_SHA_512"
    ]
  }
}
```

範例 3：取得多區域複本金鑰的詳細資訊

下列 `describe-key` 範例會取得多區域複本金鑰的中繼資料。此多區域金鑰是對稱加密金鑰。任何多區域金鑰的 `describe-key` 命令輸出，都會傳回主金鑰及其所有複本的相關資訊。

```
aws kms describe-key \
  --key-id arn:aws:kms:ap-northeast-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab
```

輸出：

```
{
  "KeyMetadata": {
    "MultiRegion": true,
    "AWSAccountId": "111122223333",
    "Arn": "arn:aws:kms:ap-northeast-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
    "CreationDate": "2021-06-28T21:09:16.114000+00:00",
    "CurrentKeyMaterialId":
"0b7fd7ddb6c6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",
    "Description": "",
    "Enabled": true,
    "KeyId": "mrk-1234abcd12ab34cd56ef1234567890ab",
    "KeyManager": "CUSTOMER",
    "KeyState": "Enabled",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "Origin": "AWS_KMS",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ],
    "MultiRegionConfiguration": {
      "MultiRegionKeyType": "PRIMARY",
      "PrimaryKey": {
        "Arn": "arn:aws:kms:us-west-2:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
        "Region": "us-west-2"
      },
      "ReplicaKeys": [
        {
          "Arn": "arn:aws:kms:eu-west-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
          "Region": "eu-west-1"
        },
        {
```

```

        "Arn": "arn:aws:kms:ap-northeast-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
        "Region": "ap-northeast-1"
    },
    {
        "Arn": "arn:aws:kms:sa-east-1:111122223333:key/
mrk-1234abcd12ab34cd56ef1234567890ab",
        "Region": "sa-east-1"
    }
]
}
}
}
}

```

範例 4：取得 HMAC KMS 金鑰的詳細資訊

下列 `describe-key` 範例會取得 HMAC KMS 金鑰的詳細資訊。

```

aws kms describe-key \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab

```

輸出：

```

{
  "KeyMetadata": {
    "AWSAccountId": "123456789012",
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "Arn": "arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "CreationDate": "2022-04-03T22:23:10.194000+00:00",
    "Enabled": true,
    "Description": "Test key",
    "KeyUsage": "GENERATE_VERIFY_MAC",
    "KeyState": "Enabled",
    "Origin": "AWS_KMS",
    "KeyManager": "CUSTOMER",
    "CustomerMasterKeySpec": "HMAC_256",
    "MacAlgorithms": [
      "HMAC_SHA_256"
    ],
    "MultiRegion": false
  }
}

```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DescribeKey](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Asynchronously checks if a specified key is enabled.
 *
 * @param keyId the ID of the key to check
 * @return a {@link CompletableFuture} that, when completed, indicates
 * whether the key is enabled or not
 *
 * @throws RuntimeException if an exception occurs while checking the key
 * state
 */
public CompletableFuture<Boolean> isKeyEnabledAsync(String keyId) {
    DescribeKeyRequest keyRequest = DescribeKeyRequest.builder()
        .keyId(keyId)
        .build();

    CompletableFuture<DescribeKeyResponse> responseFuture =
getAsyncClient().describeKey(keyRequest);
    return responseFuture.whenComplete((resp, ex) -> {
        if (resp != null) {
            KeyState keyState = resp.keyMetadata().keyState();
            if (keyState == KeyState.ENABLED) {
                logger.info("The key is enabled.");
            } else {
                logger.info("The key is not enabled. Key state: {}",
keyState);
            }
        } else {
            throw new RuntimeException(ex);
        }
    }).thenApply(resp -> resp.keyMetadata().keyState() == KeyState.ENABLED);
}
```



```
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DescribeKey](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun describeSpecifcKey(keyIdVal: String?) {  
    val request =  
        DescribeKeyRequest {  
            keyId = keyIdVal  
        }  
  
    KmsClient.fromEnvironment { region = "us-west-2" }.use { kmsClient ->  
        val response = kmsClient.describeKey(request)  
        println("The key description is ${response.keyMetadata?.description}")  
        println("The key ARN is ${response.keyMetadata?.arn}")  
    }  
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [DescribeKey](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * @param string $keyId
 * @return array
 */
public function describeKey(string $keyId)
{
    try {
        $result = $this->client->describeKey([
            "KeyId" => $keyId,
        ]);
        return $result['KeyMetadata'];
    } catch (KmsException $caught) {
        if ($caught->getAwsErrorMessage() == "NotFoundException") {
            echo "The request was rejected because the specified entity or
resource could not be found.\n";
        }
        throw $caught;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [DescribeKey](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
```

```
def from_client(cls) -> "KeyManager":
    """
    Creates a KeyManager instance with a default KMS client.

    :return: An instance of KeyManager initialized with the default KMS
client.
    """
    kms_client = boto3.client("kms")
    return cls(kms_client)

def describe_key(self, key_id: str) -> dict[str, any]:
    """
    Describes a key.

    :param key_id: The ARN or ID of the key to describe.
    :return: Information about the key.
    """

    try:
        key = self.kms_client.describe_key(KeyId=key_id)["KeyMetadata"]
        return key
    except ClientError as err:
        logging.error(
            "Couldn't get key '%s'. Here's why: %s",
            key_id,
            err.response["Error"]["Message"],
        )
        raise
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [DescribeKey](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    " iv_key_id = 'arn:aws:kms:us-  
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'  
    oo_result = lo_kms->describekey( iv_keyid = iv_key_id ).  
    DATA(lo_key) = oo_result->get_keymetadata( ).  
    MESSAGE 'Retrieved key information successfully.' TYPE 'I'.  
CATCH /aws1/cx_kmsnotfoundexception.  
    MESSAGE 'Key not found.' TYPE 'E'.  
CATCH /aws1/cx_kmskmsinternalex.  
    MESSAGE 'An internal error occurred.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [DescribeKey](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DisableKey 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DisableKey。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
```

```
using System.Threading.Tasks;
using Amazon.KeyManagementService;
using Amazon.KeyManagementService.Model;

/// <summary>
/// Disable an AWS Key Management Service (AWS KMS) key and then retrieve
/// the key's status to show that it has been disabled.
/// </summary>
public class DisableKey
{
    public static async Task Main()
    {
        var client = new AmazonKeyManagementServiceClient();

        // The identifier of the AWS KMS key to disable. You can use the
        // key Id or the Amazon Resource Name (ARN) of the AWS KMS key.
        var keyId = "1234abcd-12ab-34cd-56ef-1234567890ab";

        var request = new DisableKeyRequest
        {
            KeyId = keyId,
        };

        var response = await client.DisableKeyAsync(request);

        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            // Retrieve information about the key to show that it has now
            // been disabled.
            var describeResponse = await client.DescribeKeyAsync(new
DescribeKeyRequest
            {
                KeyId = keyId,
            });
            Console.WriteLine($"{describeResponse.KeyMetadata.KeyId} - state:
{describeResponse.KeyMetadata.KeyState}");
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [DisableKey](#)。

CLI

AWS CLI

暫時停用 KMS 金鑰

下列 `disable-key` 命令會停用客戶自管 KMS 金鑰。若要重新啟用 KMS 金鑰，請使用 `enable-key` 命令。

```
aws kms disable-key \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的[啟用和停用金鑰](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [DisableKey](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**  
 * Asynchronously disables the specified AWS Key Management Service (KMS)  
key.  
 *  
 * @param keyId the ID or Amazon Resource Name (ARN) of the KMS key to be  
disabled  
 * @return a CompletableFuture that, when completed, indicates that the key  
has been disabled successfully  
 */  
public CompletableFuture<Void> disableKeyAsync(String keyId) {  
    DisableKeyRequest keyRequest = DisableKeyRequest.builder()
```

```
        .keyId(keyId)
        .build();

return getAsyncClient().disableKey(keyRequest)
    .thenRun(() -> {
        logger.info("Key {} has been disabled successfully",keyId);
    })
    .exceptionally(throwable -> {
        throw new RuntimeException("Failed to disable key: " + keyId,
throwable);
    });
}
```

- 如需詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DisableKey](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun disableKey(keyIdVal: String?) {
    val request =
        DisableKeyRequest {
            keyId = keyIdVal
        }

    KmsClient.fromEnvironment { region = "us-west-2" }.use { kmsClient ->
        kmsClient.disableKey(request)
        println("$keyIdVal was successfully disabled")
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [DisableKey](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * @param string $keyId
 * @return void
 */
public function disableKey(string $keyId)
{
    try {
        $this->client->disableKey([
            'KeyId' => $keyId,
        ]);
    } catch (KmsException $caught){
        echo "There was a problem disabling the key: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [DisableKey](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。


```

class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
    def from_client(cls) -> "KeyManager":
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
        client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def disable_key(self, key_id: str) -> None:
        try:
            self.kms_client.disable_key(KeyId=key_id)
        except ClientError as err:
            logging.error(
                "Couldn't disable key '%s'. Here's why: %s",
                key_id,
                err.response["Error"]["Message"],
            )
            raise

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [DisableKey](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    " iv_key_id = 'arn:aws:kms:us-  
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'  
    lo_kms->disablekey( iv_keyid = iv_key_id ).  
    MESSAGE 'KMS key disabled successfully.' TYPE 'I'.  
CATCH /aws1/cx_kmsnotfoundexception.  
    MESSAGE 'Key not found.' TYPE 'E'.  
CATCH /aws1/cx_kmskmsinternalex.  
    MESSAGE 'An internal error occurred.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [DisableKey](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

EnableKey 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 EnableKey。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;  
using System.Threading.Tasks;
```

```
using Amazon.KeyManagementService;
using Amazon.KeyManagementService.Model;

/// <summary>
/// Enable an AWS Key Management Service (AWS KMS) key.
/// </summary>
public class EnableKey
{
    public static async Task Main()
    {
        var client = new AmazonKeyManagementServiceClient();

        // The identifier of the AWS KMS key to enable. You can use the
        // key Id or the Amazon Resource Name (ARN) of the AWS KMS key.
        var keyId = "1234abcd-12ab-34cd-56ef-1234567890ab";

        var request = new EnableKeyRequest
        {
            KeyId = keyId,
        };

        var response = await client.EnableKeyAsync(request);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            // Retrieve information about the key to show that it has now
            // been enabled.
            var describeResponse = await client.DescribeKeyAsync(new
DescribeKeyRequest
            {
                KeyId = keyId,
            });
            Console.WriteLine($"{describeResponse.KeyMetadata.KeyId} - state:
{describeResponse.KeyMetadata.KeyState}");
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [EnableKey](#)。

CLI

AWS CLI

啟用 KMS 金鑰

下列 `enable-key` 範例會啟用客戶自管金鑰。您可以使用像這樣的命令，來啟用使用 `disable-key` 命令暫時停用的 KMS 金鑰。您也可以使用它來啟用已停用的 KMS 金鑰，因為其已排定刪除的時程，且已取消刪除。

若要指定 KMS 金鑰，請使用 `key-id` 參數。此範例使用金鑰 ID 值，但您可以在此命令中使用金鑰 ID 或金鑰 ARN 值。

執行此命令之前，請將範例金鑰 ID 取代為有效的代碼。

```
aws kms enable-key \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

此命令不會產生輸出。若要確認 KMS 金鑰已啟用，請使用 `describe-key` 命令。查看 `describe-key` 輸出中 `KeyState` 和 `Enabled` 欄位的值。

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的[啟用和停用金鑰](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [EnableKey](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**  
 * Asynchronously enables the specified key.  
 *  
 * @param keyId the ID of the key to enable
```

```

    * @return a {@link CompletableFuture} that completes when the key has been
    enabled
    */
    public CompletableFuture<Void> enableKeyAsync(String keyId) {
        EnableKeyRequest enableKeyRequest = EnableKeyRequest.builder()
            .keyId(keyId)
            .build();

        CompletableFuture<EnableKeyResponse> responseFuture =
            getAsyncClient().enableKey(enableKeyRequest);
        responseFuture.whenComplete((response, exception) -> {
            if (exception == null) {
                logger.info("Key with ID [{}] has been enabled.", keyId);
            } else {
                if (exception instanceof KmsException kmsEx) {
                    throw new RuntimeException("KMS error occurred while enabling
key: " + kmsEx.getMessage(), kmsEx);
                } else {
                    throw new RuntimeException("An unexpected error occurred
while enabling key: " + exception.getMessage(), exception);
                }
            }
        });

        return responseFuture.thenApply(response -> null);
    }

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [EnableKey](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

suspend fun enableKey(keyIdVal: String?) {
    val request =

```

```

        EnableKeyRequest {
            keyId = keyIdVal
        }

        KmsClient.fromEnvironment { region = "us-west-2" }.use { kmsClient ->
            kmsClient.enableKey(request)
            println("$keyIdVal was successfully enabled.")
        }
    }
}

```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [EnableKey](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

/**
 * @param string $keyId
 * @return void
 */
public function enableKey(string $keyId)
{
    try {
        $this->client->enableKey([
            'KeyId' => $keyId,
        ]);
    } catch (KmsException $caught) {
        if ($caught->getAwsErrorMessage() == "NotFoundException") {
            echo "The request was rejected because the specified entity or
resource could not be found.\n";
        }
        throw $caught;
    }
}

```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [EnableKey](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
    def from_client(cls) -> "KeyManager":
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
        client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def enable_key(self, key_id: str) -> None:
        """
        Enables a key. Gets the key state after each state change.

        :param key_id: The ARN or ID of the key to enable.
        """
        try:
            self.kms_client.enable_key(KeyId=key_id)
        except ClientError as err:
            logging.error(
```

```

        "Couldn't enable key '%s'. Here's why: %s",
        key_id,
        err.response["Error"]["Message"],
    )
    raise

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [EnableKey](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    " iv_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'
    lo_kms->enablekey( iv_keyid = iv_key_id ).
    MESSAGE 'KMS key enabled successfully.' TYPE 'I'.
CATCH /aws1/cx_kmsnotfoundexception.
    MESSAGE 'Key not found.' TYPE 'E'.
CATCH /aws1/cx_kmskmsinternalex.
    MESSAGE 'An internal error occurred.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [EnableKey](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

EnableKeyRotation 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 EnableKeyRotation。

CLI

AWS CLI

啟用 KMS 金鑰的自動輪換

下列 `enable-key-rotation` 範例會啟用客戶自管 KMS 金鑰的自動輪換，輪換期間為 180 天。KMS 金鑰將從此命令完成之日起一年 (大約 365 天) 輪換，之後每年輪換一次。

`--key-id` 參數可識別 KMS 金鑰。此範例使用金鑰 ARN 值，但您可以使用金鑰 ID 或 KMS 金鑰的 ARN。`--rotation-period-in-days` 參數會指定每個輪換日期之間的天數。指定介於 90 到 2560 之間的值。如未指定任何值，則預設值為 365 天。

```
aws kms enable-key-rotation \
  --key-id arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab \
  --rotation-period-in-days 180
```

此命令不會產生輸出。若要確認 KMS 金鑰已啟用，請使用 `get-key-rotation-status` 命令。

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的[輪換金鑰](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的[EnableKeyRotation](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[AWS 程式碼範例儲存庫](#)中設定和執行。

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []
```

```
@classmethod
def from_client(cls) -> "KeyManager":
    """
    Creates a KeyManager instance with a default KMS client.

    :return: An instance of KeyManager initialized with the default KMS
    client.
    """
    kms_client = boto3.client("kms")
    return cls(kms_client)

def enable_key_rotation(self, key_id: str) -> None:
    """
    Enables rotation for a key.

    :param key_id: The ARN or ID of the key to enable rotation for.
    """
    try:
        self.kms_client.enable_key_rotation(KeyId=key_id)
    except ClientError as err:
        logging.error(
            "Couldn't enable rotation for key '%s'. Here's why: %s",
            key_id,
            err.response["Error"]["Message"],
        )
        raise
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [EnableKeyRotation](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    " iv_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'
    lo_kms->enablekeyrotation( iv_keyid = iv_key_id ).
    MESSAGE 'Key rotation enabled successfully.' TYPE 'I'.
CATCH /aws1/cx_kmsdisabledexception.
    MESSAGE 'The key is disabled.' TYPE 'E'.
CATCH /aws1/cx_kmsnotfoundexception.
    MESSAGE 'Key not found.' TYPE 'E'.
CATCH /aws1/cx_kmsunsupportedopex.
    MESSAGE 'Operation not supported for this key.' TYPE 'E'.
CATCH /aws1/cx_kmskmsinternalex.
    MESSAGE 'An internal error occurred.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [EnableKeyRotation](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Encrypt 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 Encrypt。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

CLI

AWS CLI

範例 1：在 Linux 或 MacOS 上加密檔案的內容

下列encrypt命令示範使用 CLI AWS 加密資料的建議方式。

```

aws kms encrypt \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \

```

```
--plaintext fileb://ExamplePlaintextFile \
--output text \
--query CiphertextBlob | base64 \
--decode > ExampleEncryptedFile
```

命令會執行數個動作：

使用 `--plaintext` 參數來指示要加密的資料。此參數值必須是 base64 編碼。plaintext 參數的值必須是 base64 編碼，或者您必須使用 `fileb://` 字首，指示 AWS CLI 從檔案讀取二進位資料。如果檔案不在目前的目錄中，請輸入檔案的完整路徑。例如：`fileb:///var/tmp/ExamplePlaintextFile` 或 `fileb://C:\Temp\ExamplePlaintextFile`。如需從檔案讀取 AWS CLI 參數值的詳細資訊，請參閱《AWS 命令列界面使用者指南》中的[從檔案載入參數](#)，以及《AWS 命令列工具部落格》上的[本機檔案參數的最佳實務](#)。使用 `--output` 和 `--query` 參數來控制命令的輸出。這些參數會擷取加密的資料，稱為加密文字 從命令的輸出。如需控制輸出的詳細資訊，請參閱《[命令列界面使用者指南](#)》中的[控制命令輸出](#)。使用 base64 公用程式將擷取的輸出解碼為二進位資料。成功 `encrypt` 命令傳回的加密文字是 base64 編碼的文字。AWS 您必須先解碼此文字，才能使用 AWS CLI 將其解密。將二進位加密文字儲存到 檔案。命令 (`> ExampleEncryptedFile`) 的最終部分會將二進位加密文字儲存到 檔案，讓解密更容易。如需使用 AWS CLI 解密資料的範例命令，請參閱解密範例。

範例 2：使用 AWS CLI 加密 Windows 上的資料

此範例與上一個範例相同，但它使用 `certutil` 工具而非 base64。此程序需要兩個命令，如下列範例所示。

```
aws kms encrypt \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --plaintext fileb://ExamplePlaintextFile \
  --output text \
  --query CiphertextBlob > C:\Temp\ExampleEncryptedFile.base64

certutil -decode C:\Temp\ExampleEncryptedFile.base64 C:\Temp\ExampleEncryptedFile
```

範例 3：使用非對稱 KMS 金鑰加密

下列 `encrypt` 命令顯示如何使用非對稱 KMS 金鑰加密純文字。`--encryption-algorithm` 參數是必要參數。如同所有 `encrypt` CLI 命令，plaintext 參數必須是 base64 編碼，或者您必須使用 `fileb://` 字首，告知 AWS CLI 從 檔案讀取二進位資料。

```
aws kms encrypt \
```

```
--key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
--encryption-algorithm RSAES_OAEP_SHA_256 \
--plaintext fileb://ExamplePlaintextFile \
--output text \
--query CiphertextBlob | base64 \
--decode > ExampleEncryptedFile
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [Encrypt](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Encrypts the given text asynchronously using the specified KMS client and
 * key ID.
 *
 * @param keyId the ID of the KMS key to use for encryption
 * @param text the text to encrypt
 * @return a CompletableFuture that completes with the encrypted data as an
 * SdkBytes object
 */
public CompletableFuture<SdkBytes> encryptDataAsync(String keyId, String
text) {
    SdkBytes myBytes = SdkBytes.fromUtf8String(text);
    EncryptRequest encryptRequest = EncryptRequest.builder()
        .keyId(keyId)
        .plaintext(myBytes)
        .build();

    CompletableFuture<EncryptResponse> responseFuture =
getAsyncClient().encrypt(encryptRequest).toCompletableFuture();
    return responseFuture.whenComplete((response, ex) -> {
        if (response != null) {
```

```

        String algorithm = response.encryptionAlgorithm().toString();
        logger.info("The string was encrypted with algorithm {}.",
algorithm);
    } else {
        throw new RuntimeException(ex);
    }
}).thenApply(EncryptResponse::ciphertextBlob);
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [Encrypt](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

suspend fun encryptData(keyIdValue: String): ByteArray? {
    val text = "This is the text to encrypt by using the AWS KMS Service"
    val myBytes: ByteArray = text.toByteArray()

    val encryptRequest =
        EncryptRequest {
            keyId = keyIdValue
            plaintext = myBytes
        }

    KmsClient.fromEnvironment { region = "us-west-2" }.use { kmsClient ->
        val response = kmsClient.encrypt(encryptRequest)
        val algorithm: String = response.encryptionAlgorithm.toString()
        println("The encryption algorithm is $algorithm")

        // Return the encrypted data.
        return response.ciphertextBlob
    }
}

```

```
suspend fun decryptData(
    encryptedDataVal: ByteArray?,
    keyIdVal: String?,
) {
    val decryptRequest =
        DecryptRequest {
            ciphertextBlob = encryptedDataVal
            keyId = keyIdVal
        }
    KmsClient { region = "us-west-2" }.use { kmsClient ->
        val decryptResponse = kmsClient.decrypt(decryptRequest)
        val myVal = decryptResponse.plaintext

        // Print the decrypted data.
        print(myVal)
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [Encrypt](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * @param string $keyId
 * @param string $text
 * @return Result
 */
public function encrypt(string $keyId, string $text)
{
    try {
        return $this->client->encrypt([
            'KeyId' => $keyId,
```

```

        'Plaintext' => $text,
    });
} catch (KmsException $caught){
    if($caught->getAwsErrorMessage() == "DisabledException"){
        echo "The request was rejected because the specified KMS key is
not enabled.\n";
    }
    throw $caught;
}
}

```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [Encrypt](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

class KeyEncrypt:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyEncrypt":
        """
        Creates a KeyEncrypt instance with a default KMS client.

        :return: An instance of KeyEncrypt initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def encrypt(self, key_id: str, text: str) -> bytes:

```



```

"""
Encrypts text by using the specified key.

:param key_id: The ARN or ID of the key to use for encryption.
:param text: The text to encrypt.
:return: The encrypted version of the text.
"""
try:
    response = self.kms_client.encrypt(KeyId=key_id,
Plaintext=text.encode())
    print(
        f"The string was encrypted with algorithm
{response['EncryptionAlgorithm']}"
    )
    return response["CiphertextBlob"]
except ClientError as err:
    if err.response["Error"]["Code"] == "DisabledException":
        logger.error(
            "Could not encrypt because the key %s is disabled.", key_id
        )
    else:
        logger.error(
            "Couldn't encrypt text. Here's why: %s",
            err.response["Error"]["Message"],
        )
    raise

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [Encrypt](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
require 'aws-sdk-kms' # v2: require 'aws-sdk'
```

```
# ARN of the AWS KMS key.
#
# Replace the fictitious key ARN with a valid key ID

keyId = 'arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab'

text = '1234567890'

client = Aws::KMS::Client.new(region: 'us-west-2')

resp = client.encrypt({
    key_id: keyId,
    plaintext: text
})

# Display a readable version of the resulting encrypted blob.
puts 'Blob:'
puts resp.ciphertext_blob.unpack('H*')
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [Encrypt](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
async fn encrypt_string(
    verbose: bool,
    client: &Client,
    text: &str,
    key: &str,
    out_file: &str,
) -> Result<(), Error> {
    let blob = Blob::new(text.as_bytes());
```

```

let resp = client.encrypt().key_id(key).plaintext(blob).send().await?;

// Did we get an encrypted blob?
let blob = resp.ciphertext_blob.expect("Could not get encrypted text");
let bytes = blob.as_ref();

let s = base64::encode(bytes);

let mut ofile = File::create(out_file).expect("unable to create file");
ofile.write_all(s.as_bytes()).expect("unable to write");

if verbose {
    println!("Wrote the following to {:?}", out_file);
    println!("{}", s);
}

Ok(())
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Rust API 參考》中的 [Encrypt](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
  " iv_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'
  " iv_plaintext contains the data to encrypt
  oo_result = lo_kms->encrypt(
    iv_keyid = iv_key_id
    iv_plaintext = iv_plaintext
  ).
  MESSAGE 'Text encrypted successfully.' TYPE 'I'.

```

```
CATCH /aws1/cx_kmsdisabledexception.  
    MESSAGE 'The key is disabled.' TYPE 'E'.  
CATCH /aws1/cx_kmsnotfoundexception.  
    MESSAGE 'Key not found.' TYPE 'E'.  
CATCH /aws1/cx_kmskmsinternalex.  
    MESSAGE 'An internal error occurred.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的[加密](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GenerateDataKey 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GenerateDataKey。

CLI

AWS CLI

範例 1：產生 256 位元對稱資料金鑰

下列 generate-data-key 範例會請求在外部使用的 256 位元對稱資料金鑰 AWS。命令會傳回純文字資料金鑰，以供立即使用和刪除，以及在指定的 KMS 金鑰下加密的資料金鑰複本。加密的資料金鑰可以安全地跟加密資料一起存放。

若要請求 256 位元資料金鑰，請使用值為 AES_256 的 key-spec 參數。若要請求 128 位元資料金鑰，請使用值為 AES_128 的 key-spec 參數。對於所有其他資料金鑰長度，請使用 number-of-bytes 參數。

您指定的 KMS 金鑰必須是對稱加密 KMS 金鑰，也就是金鑰規格值為 SYMMETRIC_DEFAULT 的 KMS 金鑰。

```
aws kms generate-data-key \  
    --key-id alias/ExampleAlias \  
    --key-spec AES_256
```

輸出：

```
{
  "Plaintext": "VdzKNHGzUAzJeRBVY+uUmofUGGiDzyB3+i9fVkh3piw=",
  "KeyId": "arn:aws:kms:us-east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "KeyMaterialId":
    "0b7fd7ddb6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6",
  "CiphertextBlob":
    "AQEDAHjRYf5WytIc0C857tFSnBaPn2F8DgfmThbJlGfR8P3WlwAAAH4wfAYJKoZIhvcNAQcGoG8wbQIBADBoBgk
    +YdhV8MrkBQPeac0ReRVNDt9qlAt+SHgIRF8P0H+7U="
}
```

Plaintext (純文字資料金鑰) 和 CiphertextBlob(加密的資料金鑰) 會以 base64 編碼格式傳回。

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的[資料金鑰](#)。範例 2：產生 512 位元對稱資料金鑰

下列 generate-data-key 範例會請求 512 位元對稱資料金鑰，以進行加密和解密。命令會傳回純文字資料金鑰，以供立即使用和刪除，以及在指定的 KMS 金鑰下加密的資料金鑰複本。加密的資料金鑰可以安全地跟加密資料一起存放。

若要請求 128 或 256 位元以外的金鑰長度，請使用 number-of-bytes 參數。若要請求 512 位元資料金鑰，下列範例會使用值為 64 (位元組) 的 number-of-bytes 參數。

您指定的 KMS 金鑰必須是對稱加密 KMS 金鑰，也就是金鑰規格值為 SYMMETRIC_DEFAULT 的 KMS 金鑰。

注意：此範例輸出中的值會截斷以方便顯示。

```
aws kms generate-data-key \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --number-of-bytes 64
```

輸出：

```
{
  "CiphertextBlob": "AQIBAHi6LtupRpdKl2aJTzkk6Fbh0tQkMlQJJH3PdtHvS/y+hAEnX/
  QQNmMwDfg2korNMEc8AAACaDCCAmQGCSqGSIB3DQEHbQCCA1UwggJRAgEAMIICSgYJKoZ...",
  "Plaintext": "ty8Lr0Bk60F07M2Bwt6qbFdNB
  +G00ZLtf5MSEb4a13R2UKWGOp06njAwy2n72VRm2m7z/
  Pm9Wpbvttz6a4lSo9hgPvKhZ5y6RTm40ovEXiVfBveyX3DQxDzRSwbKDPk/...",
}
```

```
"KeyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "KeyMaterialId":
    "0b7fd7ddb6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6"
}
```

Plaintext (純文字資料金鑰) 和 CiphertextBlob(加密的資料金鑰) 會以 base64 編碼格式傳回。

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的[資料金鑰](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GenerateDataKey](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
    def from_client(cls) -> "KeyManager":
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
        client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def generate_data_key(self, key_id):
        """
```

```

        Generates a symmetric data key that can be used for client-side
        encryption.
        """
        answer = input(
            f"Do you want to generate a symmetric data key from key {key_id} (y/
n)? "
        )
        if answer.lower() == "y":
            try:
                data_key = self.kms_client.generate_data_key(
                    KeyId=key_id, KeySpec="AES_256"
                )
            except ClientError as err:
                logger.error(
                    "Couldn't generate a data key for key %s. Here's why: %s",
                    key_id,
                    err.response["Error"]["Message"],
                )
            else:
                pprint(data_key)

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [GenerateDataKey](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

async fn make_key(client: &Client, key: &str) -> Result<(), Error> {
    let resp = client
        .generate_data_key()
        .key_id(key)
        .key_spec(DataKeySpec::Aes256)

```

```

        .send()
        .await?;

// Did we get an encrypted blob?
let blob = resp.ciphertext_blob.expect("Could not get encrypted text");
let bytes = blob.as_ref();

let s = base64::encode(bytes);

println!();
println!("Data key:");
println!("{}", s);

Ok(())
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Rust API 參考》中的 [GenerateDataKey](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    " iv_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'
    " iv_keyspec = 'AES_256'
    oo_result = lo_kms->generatedatakey(
        iv_keyid = iv_key_id
        iv_keyspec = 'AES_256'
    ).
    MESSAGE 'Data key generated successfully.' TYPE 'I'.
CATCH /aws1/cx_kmsdisabledexception.
    MESSAGE 'The key is disabled.' TYPE 'E'.
CATCH /aws1/cx_kmsnotfoundexception.
    MESSAGE 'Key not found.' TYPE 'E'.

```



```
CATCH /aws1/cx_kmskmsinternalex.  
MESSAGE 'An internal error occurred.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [GenerateDataKey](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GenerateDataKeyWithoutPlaintext 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GenerateDataKeyWithoutPlaintext。

CLI

AWS CLI

產生不含純文字金鑰的 256 位元對稱資料金鑰

下列 generate-data-key-without-plaintext 範例會請求 256 位元對稱資料金鑰的加密複本，以供在 AWS 之外使用。當您準備好要使用資料金鑰時，您可以呼叫 AWS KMS 來解密資料金鑰。

若要請求 256 位元資料金鑰，請使用值為 AES_256 的 key-spec 參數。若要請求 128 位元資料金鑰，請使用值為 AES_128 的 key-spec 參數。對於所有其他資料金鑰長度，請使用 number-of-bytes 參數。

您指定的 KMS 金鑰必須是對稱加密 KMS 金鑰，也就是金鑰規格值為 SYMMETRIC_DEFAULT 的 KMS 金鑰。

```
aws kms generate-data-key-without-plaintext \  
  --key-id "arn:aws:kms:us-  
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab" \  
  --key-spec AES_256
```

輸出：

```
{
```

```

    "CiphertextBlob":
    "AQEDAHjRYf5WytIc0C857tFSnBaPn2F8DgfmThbJlGfR8P3WlwAAAH4wfAYJKoZIhvcNAQcGoG8wbQIBADBoBgk
    "KeyId": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
    "KeyMaterialId":
    "0b7fd7ddb6eef27907413567cad8c810e2883dc8a7534067a82ee1142fc1e6"
  }

```

CiphertextBlob (加密的資料金鑰) 會以 base64 編碼格式傳回。

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的[資料金鑰](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的[GenerateDataKeyWithoutPlaintext](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[AWS 程式碼範例儲存庫](#)中設定和執行。

```

async fn make_key(client: &Client, key: &str) -> Result<(), Error> {
    let resp = client
        .generate_data_key_without_plaintext()
        .key_id(key)
        .key_spec(DataKeySpec::Aes256)
        .send()
        .await?;

    // Did we get an encrypted blob?
    let blob = resp.ciphertext_blob.expect("Could not get encrypted text");
    let bytes = blob.as_ref();

    let s = base64::encode(bytes);

    println!();
    println!("Data key:");
    println!("{}", s);
}

```

```
Ok(() )  
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Rust API 參考》中的 [GenerateDataKeyWithoutPlaintext](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GenerateRandom 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GenerateRandom。

CLI

AWS CLI

範例 1：產生 256 位元隨機位元組字串 (Linux 或 macOS)

下列 generate-random 範例會產生 256 位元 (32 位元組) 的 base64 編碼隨機位元組字串。此範例會解碼位元組字串，並將其儲存在隨機檔案中。

執行此命令時，您必須使用 number-of-bytes 參數來指定以位元組為單位的隨機值長度。

當您執行此命令時，不會指定 KMS 金鑰。隨機位元組字串與任何 KMS 金鑰無關。

根據預設，AWS KMS 會產生隨機數字。不過，如果您指定 [自訂金鑰存放區](#)，隨機位元組字串會在與自訂金鑰存放區相關聯的 AWS CloudHSM 叢集中產生。

此範例使用下列參數和值：

它使用值為的必要 `--number-of-bytes` 參數 32 來請求 32 位元組 (256 位元) string。It 使用值為的 `--output` 參數 `text` 來指示 AWS CLI 以文字形式傳回輸出。而不是 JSON。It 使用從 `response`。It 管道 (|) 將命令的輸出 `--query parameter` 擷取到 base64 公用程式的 Plaintext 屬性值，會解碼擷取的 `output`。It 使用重新導向運算子 (>) 將解碼的位元組字串儲存到 `ExampleRandom file`。It 使用重新導向運算子 (>) 將二進位加密文字儲存到檔案。

```
aws kms generate-random \
```

```
--number-of-bytes 32 \  
--output text \  
--query Plaintext | base64 --decode > ExampleRandom
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS Key Management Service API 參考》中的 [GenerateRandom](#)。

範例 2：產生 256 位元隨機數字 (Windows 命令提示)

以下範例使用 `generate-random` 命令來產生 256 位元 (32 位元組) 的 base64 編碼隨機位元組字串。此範例會解碼位元組字串，並將其儲存在隨機檔案中。此範例與先前的範例相同，不過它會在 Windows 中使用 `certutil` 公用程式將隨機位元組字串進行 base64 解碼，然後再將其儲存在檔案中。

首先，產生 base64 編碼的隨機位元組字串，並將其儲存在暫存檔案 `ExampleRandom.base64` 中。

```
aws kms generate-random \  
  --number-of-bytes 32 \  
  --output text \  
  --query Plaintext > ExampleRandom.base64
```

由於 `generate-random` 命令的輸出會儲存在檔案中，此範例不會產生輸出。

現在，請使用 `certutil -decode` 命令來解碼 `ExampleRandom.base64` 檔案中的 base64 編碼位元組字串。然後，它會將解碼後的位元組字串儲存在 `ExampleRandom` 檔案中。

```
certutil -decode ExampleRandom.base64 ExampleRandom
```

輸出：

```
Input Length = 18  
Output Length = 12  
CertUtil: -decode command completed successfully.
```

如需詳細資訊，請參閱《AWS Key Management Service API 參考》中的 [GenerateRandom](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GenerateRandom](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
async fn make_string(client: &Client, length: i32) -> Result<(), Error> {
    let resp = client
        .generate_random()
        .number_of_bytes(length)
        .send()
        .await?;

    // Did we get an encrypted blob?
    let blob = resp.plaintext.expect("Could not get encrypted text");
    let bytes = blob.as_ref();

    let s = base64::encode(bytes);

    println!();
    println!("Data key:");
    println!("{}", s);

    Ok(())
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Rust API 參考》中的 [GenerateRandom](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetKeyPolicy 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetKeyPolicy。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

CLI

AWS CLI

將金鑰政策從一個 KMS 金鑰複製到另一個 KMS 金鑰

下列 `get-key-policy` 範例會從一個 KMS 金鑰取得金鑰政策，並將其儲存在文字檔中。然後，它會使用文字檔取代不同 KMS 金鑰的政策，做為政策輸入。

由於 `put-key-policy` 的 `--policy` 參數需要字串，您必須使用 `--output text` 選項將輸出以文字字串傳回，而非 JSON。

```
aws kms get-key-policy \
  --policy-name default \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --query Policy \
  --output text > policy.txt

aws kms put-key-policy \
  --policy-name default \
  --key-id 0987dcba-09fe-87dc-65ba-ab0987654321 \
  --policy file://policy.txt
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS KMS API 參考》中的《[PutKeyPolicy](#)》。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [GetKeyPolicy](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class KeyPolicy:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyPolicy":
        """
        Creates a KeyPolicy instance with a default KMS client.

        :return: An instance of KeyPolicy initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def get_policy(self, key_id: str) -> dict[str, str]:
        """
        Gets the policy of a key.

        :param key_id: The ARN or ID of the key to query.
        :return: The key policy as a dict.
        """
        if key_id != "":
            try:
                response = self.kms_client.get_key_policy(
                    KeyId=key_id,
                )
                policy = json.loads(response["Policy"])
            except ClientError as err:
                logger.error(
                    "Couldn't get policy for key %. Here's why: %s",
```

```

        key_id,
        err.response["Error"]["Message"],
    )
    raise
else:
    pprint(policy)
    return policy
else:
    print("Skipping get policy demo.")

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [GetKeyPolicy](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    " iv_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'
    oo_result = lo_kms->getkeypolicy(
        iv_keyid = iv_key_id
        iv_policyname = 'default'
    ).
    MESSAGE 'Retrieved key policy successfully.' TYPE 'I'.
CATCH /aws1/cx_kmsnotfoundexception.
    MESSAGE 'Key not found.' TYPE 'E'.
CATCH /aws1/cx_kmskmsinternalex.
    MESSAGE 'An internal error occurred.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [GetKeyPolicy](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListAliases 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ListAliases。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon.KeyManagementService;
using Amazon.KeyManagementService.Model;

/// <summary>
/// List the AWS Key Management Service (AWS KMS) aliases that have been
defined for
/// the keys in the same AWS Region as the default user. If you want to list
/// the aliases in a different Region, pass the Region to the client
/// constructor.
/// </summary>
public class ListAliases
{
    public static async Task Main()
    {
        var client = new AmazonKeyManagementServiceClient();
        var request = new ListAliasesRequest();
        var response = new ListAliasesResponse();
```

```
        do
        {
            response = await client.ListAliasesAsync(request);

            response.Aliases.ForEach(alias =>
            {
                Console.WriteLine($"Created: {alias.CreationDate} Last
Update: {alias.LastUpdatedDate} Name: {alias.AliasName}");
            });

            request.Marker = response.NextMarker;
        }
        while (response.Truncated);
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [ListAliases](#)。

CLI

AWS CLI

範例 1：列出 AWS 帳戶和區域中的所有別名

下列範例使用 `list-aliases` 命令列出 AWS 帳戶預設區域中的所有別名。輸出包含與 AWS 受管 KMS 金鑰和客戶受管 KMS 金鑰相關聯的別名。

```
aws kms list-aliases
```

輸出：

```
{
  "Aliases": [
    {
      "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/testKey",
      "AliasName": "alias/testKey",
      "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    {
```

```

        "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/FinanceDept",
        "AliasName": "alias/FinanceDept",
        "TargetKeyId": "0987dcba-09fe-87dc-65ba-ab0987654321"
    },
    {
        "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/aws/dynamodb",
        "AliasName": "alias/aws/dynamodb",
        "TargetKeyId": "1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d"
    },
    {
        "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/aws/ebs",
        "AliasName": "alias/aws/ebs",
        "TargetKeyId": "0987ab65-43cd-21ef-09ab-87654321cdef"
    },
    ...
]
}

```

範例 2：列出特定 KMS 金鑰的所有別名

以下範例使用 `list-aliases` 命令及其 `key-id` 參數來列出與特定 KMS 金鑰相關聯的所有別名。

每個別名只會與一個 KMS 金鑰相關聯，但 KMS 金鑰可以有多個別名。此命令非常有用，因為 AWS KMS 主控台只會為每個 KMS 金鑰列出一個別名。若要尋找 KMS 金鑰的所有別名，您必須使用 `list-aliases` 命令。

此範例使用 `--key-id` 參數之 KMS 金鑰的金鑰 ID，但您可以在此命令中使用金鑰 ID、金鑰 ARN、別名的名稱，或別名 ARN。

```
aws kms list-aliases --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

輸出：

```

{
  "Aliases": [
    {
      "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
      "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/oregon-test-key",
      "AliasName": "alias/oregon-test-key"
    },
  ],
}

```

```
{
  "TargetKeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
  "AliasArn": "arn:aws:kms:us-west-2:111122223333:alias/project121-
test",
  "AliasName": "alias/project121-test"
}
]
```

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的[使用別名](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ListAliases](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Asynchronously lists all the aliases in the current AWS account.
 *
 * @return a {@link CompletableFuture} that completes when the list of
 * aliases has been processed
 */
public CompletableFuture<Object> listAllAliasesAsync() {
    ListAliasesRequest aliasesRequest = ListAliasesRequest.builder()
        .limit(15)
        .build();

    ListAliasesPublisher paginator =
getAsyncClient().listAliasesPaginator(aliasesRequest);
    return paginator.subscribe(response -> {
        response.aliases().forEach(alias ->
            logger.info("The alias name is: " + alias.aliasName())
        );
    })
        .thenApply(v -> null)
```

```
        .exceptionally(ex -> {
            if (ex.getCause() instanceof KmsException) {
                KmsException e = (KmsException) ex.getCause();
                throw new RuntimeException("A KMS exception occurred: " +
e.getMessage());
            } else {
                throw new RuntimeException("An unexpected error occurred: " +
ex.getMessage());
            }
        });
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ListAliases](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun listAllAliases() {
    val request =
        ListAliasesRequest {
            limit = 15
        }

    KmsClient.fromEnvironment { region = "us-west-2" }.use { kmsClient ->
        val response = kmsClient.listAliases(request)
        response.aliases?.forEach { alias ->
            println("The alias name is ${alias.aliasName}")
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [ListAliases](#)。

PHP

適用於 PHP 的 SDK

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * @param string $keyId
 * @param int $limit
 * @return ResultPaginator
 */
public function listAliases(string $keyId = "", int $limit = 0)
{
    $args = [];
    if($keyId){
        $args['KeyId'] = $keyId;
    }
    if($limit){
        $args['Limit'] = $limit;
    }
    try{
        return $this->client->getPaginator("ListAliases", $args);
    }catch(KmsException $caught){
        if($caught->getAwsErrorMessage() == "InvalidMarkerException"){
            echo "The request was rejected because the marker that specifies
where pagination should next begin is not valid.\n";
        }
        throw $caught;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [ListAliases](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class AliasManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_key = None

    @classmethod
    def from_client(cls) -> "AliasManager":
        """
        Creates an AliasManager instance with a default KMS client.

        :return: An instance of AliasManager initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def list_aliases(self, page_size: int) -> None:
        """
        Lists aliases for the current account.
        :param page_size: The number of aliases to list per page.
        """
        try:
            alias_paginator = self.kms_client.get_paginator("list_aliases")
            for alias_page in alias_paginator.paginate(
                PaginationConfig={"PageSize": page_size}
            ):
                print(f"Here are {page_size} aliases:")
                pprint(alias_page["Aliases"])
                if alias_page["Truncated"]:
                    answer = input(
```

```
        f"Do you want to see the next {page_size} aliases (y/n)?\n\n        )\n        if answer.lower() != "y":\n            break\n        else:\n            print("That's all your aliases!")\n    except ClientError as err:\n        logging.error(\n            "Couldn't list your aliases. Here's why: %s",\n            err.response["Error"]["Message"],\n        )\n        raise
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [ListAliases](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.\n    oo_result = lo_kms->listaliases( ).\n    MESSAGE 'Retrieved KMS aliases list.' TYPE 'I'. \n    CATCH /aws1/cx_kmskmsinternalex.\n        MESSAGE 'An internal error occurred.' TYPE 'E'. \n    ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [ListAliases](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListGrants 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ListGrants。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon.KeyManagementService;
using Amazon.KeyManagementService.Model;

/// <summary>
/// List the AWS Key Management Service (AWS KMS) grants that are associated
with
/// a specific key.
/// </summary>
public class ListGrants
{
    public static async Task Main()
    {
        // The identifier of the AWS KMS key to disable. You can use the
        // key Id or the Amazon Resource Name (ARN) of the AWS KMS key.
        var keyId = "1234abcd-12ab-34cd-56ef-1234567890ab";
        var client = new AmazonKeyManagementServiceClient();
        var request = new ListGrantsRequest
        {
            KeyId = keyId,
        };
    }
}
```

```
var response = new ListGrantsResponse();

do
{
    response = await client.ListGrantsAsync(request);

    response.Grants.ForEach(grant =>
    {
        Console.WriteLine($"{grant.GrantId}");
    });

    request.Marker = response.NextMarker;
}
while (response.Truncated);
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [ListGrants](#)。

CLI

AWS CLI

檢視 AWS KMS 金鑰上的授予

下列 `list-grants` 範例顯示您帳戶中 Amazon DynamoDB 指定之 AWS 受管 KMS 金鑰上的所有授予。此授權可讓 DynamoDB 代表您使用 KMS 金鑰加密 DynamoDB 資料表，然後再寫入磁碟。您可以使用像這樣的命令來檢視 AWS 帳戶和區域中 AWS 受管 KMS 金鑰和客戶受管 KMS 金鑰的授予。

此命令搭配使用 `key-id` 參數和金鑰 ID 來識別 KMS 金鑰。您可以使用金鑰 ID 或金鑰 ARN 來識別 KMS 金鑰。若要取得 AWS 受管 KMS 金鑰的金鑰 ID 或金鑰 ARN，請使用 `list-keys` 或 `list-aliases` 命令。

```
aws kms list-grants \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

輸出顯示，授權包括為 Amazon DynamoDB 提供使用 KMS 金鑰進行密碼編譯操作的許可權，並提供讓其檢視 KMS 金鑰 (`DescribeKey`) 詳細資訊，和淘汰授予 (`RetireGrant`) 的權

限。EncryptionContextSubset 限制條件會將這些許可權限制為包含指定加密內容對的請求。因此，授予的許可權僅在指定的帳戶和 DynamoDB 資料表上有效。

```
{
  "Grants": [
    {
      "Constraints": {
        "EncryptionContextSubset": {
          "aws:dynamodb:subscriberId": "123456789012",
          "aws:dynamodb:tableName": "Services"
        }
      },
      "IssuingAccount": "arn:aws:iam::123456789012:root",
      "Name": "8276b9a6-6cf0-46f1-b2f0-7993a7f8c89a",
      "Operations": [
        "Decrypt",
        "Encrypt",
        "GenerateDataKey",
        "ReEncryptFrom",
        "ReEncryptTo",
        "RetireGrant",
        "DescribeKey"
      ],
      "GrantId":
        "1667b97d27cf748cf05b487217dd4179526c949d14fb3903858e25193253fe59",
      "KeyId": "arn:aws:kms:us-west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "RetiringPrincipal": "dynamodb.us-west-2.amazonaws.com",
      "GranteePrincipal": "dynamodb.us-west-2.amazonaws.com",
      "CreationDate": "2021-05-13T18:32:45.144000+00:00"
    }
  ]
}
```

如需詳細資訊，請參閱 AWS Key Management Service 開發人員指南中的 [AWS KMS 中的授權](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ListGrants](#)。

Java

SDK for Java 2.x

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Asynchronously displays the grant IDs for the specified key ID.
 *
 * @param keyId the ID of the AWS KMS key for which to list the grants
 * @return a {@link CompletableFuture} that, when completed, will be null
 * if the operation succeeded, or will throw a {@link RuntimeException} if the
 * operation failed
 * @throws RuntimeException if there was an error listing the grants, either
 * due to an {@link KmsException} or an unexpected error
 */
public CompletableFuture<Object> displayGrantIdsAsync(String keyId) {
    ListGrantsRequest grantsRequest = ListGrantsRequest.builder()
        .keyId(keyId)
        .limit(15)
        .build();

    ListGrantsPublisher paginator =
        getAsyncClient().listGrantsPaginator(grantsRequest);
    return paginator.subscribe(response -> {
        response.grants().forEach(grant -> {
            logger.info("The grant Id is: " + grant.grantId());
        });
    })
        .thenApply(v -> null)
        .exceptionally(ex -> {
            Throwable cause = ex.getCause();
            if (cause instanceof KmsException) {
                throw new RuntimeException("Failed to list grants: " +
                    cause.getMessage(), cause);
            } else {
                throw new RuntimeException("An unexpected error occurred: " +
                    cause.getMessage(), cause);
            }
        });
}
```

```
        }  
    });  
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ListGrants](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun displayGrantIds(keyIdVal: String?) {  
    val request =  
        ListGrantsRequest {  
            keyId = keyIdVal  
            limit = 15  
        }  
  
    KmsClient.fromEnvironment { region = "us-west-2" }.use { kmsClient ->  
        val response = kmsClient.listGrants(request)  
        response.grants?.forEach { grant ->  
            println("The grant Id is ${grant.grantId}")  
        }  
    }  
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [ListGrants](#)。

PHP

適用於 PHP 的 SDK

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * @param string $keyId
 * @return Result
 */
public function listGrants(string $keyId)
{
    try{
        return $this->client->listGrants([
            'KeyId' => $keyId,
        ]);
    }catch(KmsException $caught){
        if($caught->getAwsErrorMessage() == "NotFoundException"){
            echo "    The request was rejected because the specified entity
or resource could not be found.\n";
        }
        throw $caught;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [ListGrants](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class GrantManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "GrantManager":
        """
        Creates a GrantManager instance with a default KMS client.

        :return: An instance of GrantManager initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def list_grants(self, key_id):
        """
        Lists grants for a key.

        :param key_id: The ARN or ID of the key to query.
        :return: The grants for the key.
        """
        try:
            paginator = self.kms_client.get_paginator("list_grants")
            grants = []
            page_iterator = paginator.paginate(KeyId=key_id)
            for page in page_iterator:
                grants.extend(page["Grants"])

            print(f"Grants for key {key_id}:")
            pprint(grants)
```

```
        return grants
    except ClientError as err:
        logger.error(
            "Couldn't list grants for key %s. Here's why: %s",
            key_id,
            err.response["Error"]["Message"],
        )
        raise
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [ListGrants](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.
    " iv_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'
    oo_result = lo_kms->listgrants( iv_keyid = iv_key_id ).
    MESSAGE 'Retrieved grants list.' TYPE 'I'.
CATCH /aws1/cx_kmsnotfoundexception.
    MESSAGE 'Key not found.' TYPE 'E'.
CATCH /aws1/cx_kmskmsinternalex.
    MESSAGE 'An internal error occurred.' TYPE 'E'.
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [ListGrants](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListKeyPolicies 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ListKeyPolicies。

CLI

AWS CLI

取得 KMS 金鑰的金鑰政策名稱

下列 list-key-policies 範例會取得範例帳戶和區域中客戶自管金鑰的金鑰政策名稱。您可以使用此命令來尋找 AWS 受管金鑰和客戶受管金鑰的金鑰政策名稱。

由於唯一有效的金鑰政策名稱是 default，因此這個命令沒有作用。

若要指定 KMS 金鑰，請使用 key-id 參數。此範例使用金鑰 ID 值，但您可以在此命令中使用金鑰 ID 或金鑰 ARN。

```
aws kms list-key-policies \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

輸出：

```
{
  "PolicyNames": [
    "default"
  ]
}
```

如需 AWS KMS 金鑰政策的詳細資訊，請參閱 [Key Management Service 開發人員指南中的在 AWS KMS 中使用金鑰政策](#)。AWS

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ListKeyPolicies](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Asynchronously retrieves the key policy for the specified key ID and
 * policy name.
 *
 * @param keyId      the ID of the AWS KMS key for which to retrieve the
 * policy
 * @param policyName the name of the key policy to retrieve
 * @return a {@link CompletableFuture} that, when completed, contains the key
 * policy as a {@link String}
 */
public CompletableFuture<String> getKeyPolicyAsync(String keyId, String
policyName) {
    GetKeyPolicyRequest policyRequest = GetKeyPolicyRequest.builder()
        .keyId(keyId)
        .policyName(policyName)
        .build();

    return getAsyncClient().getKeyPolicy(policyRequest)
        .thenApply(response -> {
            String policy = response.policy();
            logger.info("The response is: " + policy);
            return policy;
        })
        .exceptionally(ex -> {
            throw new RuntimeException("Failed to get key policy", ex);
        });
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ListKeyPolicies](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class KeyPolicy:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyPolicy":
        """
        Creates a KeyPolicy instance with a default KMS client.

        :return: An instance of KeyPolicy initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def list_policies(self, key_id):
        """
        Lists the names of the policies for a key.

        :param key_id: The ARN or ID of the key to query.
        """
        try:
            policy_names = self.kms_client.list_key_policies(KeyId=key_id)[
                "PolicyNames"
            ]
        except ClientError as err:
            logging.error(
                "Couldn't list your policies. Here's why: %s",
                err.response["Error"]["Message"],
            )
            raise
```

```
else:
    print(f"The policies for key {key_id} are:")
    pprint(policy_names)
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [ListKeyPolicies](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.
    " iv_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'
    oo_result = lo_kms->listkeypolicies( iv_keyid = iv_key_id ).
    MESSAGE 'Retrieved key policies list.' TYPE 'I'.
CATCH /aws1/cx_kmsnotfoundexception.
    MESSAGE 'Key not found.' TYPE 'E'.
CATCH /aws1/cx_kmskmsinternalex.
    MESSAGE 'An internal error occurred.' TYPE 'E'.
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [ListKeyPolicies](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListKeys 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ListKeys。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon.KeyManagementService;
using Amazon.KeyManagementService.Model;

/// <summary>
/// List the AWS Key Managements Service (AWS KMS) keys for the AWS Region
/// of the default user. To list keys in another AWS Region, supply the
Region
/// as a parameter to the client constructor.
/// </summary>
public class ListKeys
{
    public static async Task Main()
    {
        var client = new AmazonKeyManagementServiceClient();
        var request = new ListKeysRequest();
        var response = new ListKeysResponse();

        do
        {
            response = await client.ListKeysAsync(request);

            response.Keys.ForEach(key =>
            {
                Console.WriteLine($"ID: {key.KeyId}, {key.KeyArn}");
            });
        }
    }
}
```

```
});

// Set the Marker property when response.Truncated is true
// in order to get the next keys.
request.Marker = response.NextMarker;
}
while (response.Truncated);
}
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [ListKeys](#)。

CLI

AWS CLI

取得帳戶和區域中的 KMS 金鑰

下列 `list-keys` 範例取得帳戶和區域中的 KMS 金鑰。此命令會同時傳回 AWS 受管金鑰和客戶受管金鑰。

```
aws kms list-keys
```

輸出：

```
{
  "Keys": [
    {
      "KeyArn": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    {
      "KeyArn": "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321",
      "KeyId": "0987dcba-09fe-87dc-65ba-ab0987654321"
    },
    {
      "KeyArn": "arn:aws:kms:us-east-2:111122223333:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d",

```

```
        "KeyId": "1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d"
    }
]
}
```

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的[檢視金鑰](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ListKeys](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.services.kms.KmsAsyncClient;
import software.amazon.awssdk.services.kms.model.ListKeysRequest;
import software.amazon.awssdk.services.kms.paginators.ListKeysPublisher;
import java.util.concurrent.CompletableFuture;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class HelloKMS {
    public static void main(String[] args) {
        listAllKeys();
    }

    public static void listAllKeys() {
        KmsAsyncClient kmsAsyncClient = KmsAsyncClient.builder()
            .build();
        ListKeysRequest listKeysRequest = ListKeysRequest.builder()
```

```

        .limit(15)
        .build();

    /*
     * The `subscribe` method is required when using paginator methods in the
    AWS SDK
     * because paginator methods return an instance of a `ListKeysPublisher`,
    which is
     * based on a reactive stream. This allows asynchronous retrieval of
    paginated
     * results as they become available. By subscribing to the stream, we can
    process
     * each page of results as they are emitted.
     */
    ListKeysPublisher keysPublisher =
kmsAsyncClient.listKeysPaginator(listKeysRequest);
    CompletableFuture<Void> future = keysPublisher
        .subscribe(r -> r.keys().forEach(key ->
            System.out.println("The key ARN is: " + key.keyArn() + ". The key
    Id is: " + key.keyId()))
        .whenComplete((result, exception) -> {
            if (exception != null) {
                System.err.println("Error occurred: " +
exception.getMessage());
            } else {
                System.out.println("Successfully listed all keys.");
            }
        });

    try {
        future.join();
    } catch (Exception e) {
        System.err.println("Failed to list keys: " + e.getMessage());
    }
}
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ListKeys](#)。

Kotlin

適用於 Kotlin 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
suspend fun listAllKeys() {
    val request =
        ListKeysRequest {
            limit = 15
        }

    KmsClient.fromEnvironment { region = "us-west-2" }.use { kmsClient ->
        val response = kmsClient.listKeys(request)
        response.keys?.forEach { key ->
            println("The key ARN is ${key.keyArn}")
            println("The key Id is ${key.keyId}")
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 Kotlin 的 AWS SDK API 參考》中的 [ListKeys](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
```

```

    * @return array
    */
    public function listKeys()
    {
        try {
            $contents = [];
            $paginator = $this->client->getPaginator("ListKeys");
            foreach($paginator as $result){
                foreach ($result['Content'] as $object) {
                    $contents[] = $object;
                }
            }
            return $contents;
        }catch(KmsException $caught){
            echo "There was a problem listing the keys: {$caught-
>getAwsErrorMessage()}\n";
            throw $caught;
        }
    }
}

```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [ListKeys](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
    def from_client(cls) -> "KeyManager":
        """

```

```

    Creates a KeyManager instance with a default KMS client.

    :return: An instance of KeyManager initialized with the default KMS
client.
    """
    kms_client = boto3.client("kms")
    return cls(kms_client)

def list_keys(self):
    """
    Lists the keys for the current account by using a paginator.
    """
    try:
        page_size = 10
        print("\nLet's list your keys.")
        key_paginator = self.kms_client.get_paginator("list_keys")
        for key_page in key_paginator.paginate(PaginationConfig={"PageSize":
10}):
            print(f"Here are {len(key_page['Keys'])} keys:")
            pprint(key_page["Keys"])
            if key_page["Truncated"]:
                answer = input(
                    f"Do you want to see the next {page_size} keys (y/n)? "
                )
                if answer.lower() != "y":
                    break
            else:
                print("That's all your keys!")
    except ClientError as err:
        logging.error(
            "Couldn't list your keys. Here's why: %s",
            err.response["Error"]["Message"],
        )

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [ListKeys](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
async fn show_keys(client: &Client) -> Result<(), Error> {
    let resp = client.list_keys().send().await?;

    let keys = resp.keys.unwrap_or_default();

    let len = keys.len();

    for key in keys {
        println!("Key ARN: {}", key.key_arn.as_deref().unwrap_or_default());
    }

    println();
    println!("Found {} keys", len);

    Ok(())
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Rust API 參考》中的 [ListKeys](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    oo_result = lo_kms->listkeys( ).  
    MESSAGE 'Retrieved KMS keys list.' TYPE 'I'.  
CATCH /aws1/cx_kmskmsinternalex.  
    MESSAGE 'An internal error occurred.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [ListKeys](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutKeyPolicy 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutKeyPolicy。

CLI

AWS CLI

變更 KMS 金鑰的金鑰政策

下列 put-key-policy 範例會變更客戶自管金鑰的金鑰政策。

一開始，請建立金鑰政策，並將其儲存在本機 JSON 檔案中。在此範例中，檔案是 key_policy.json。您也可以將金鑰政策指定為 policy 參數的字串值。

此金鑰政策中的第一個陳述式提供 AWS 帳戶使用 IAM 政策控制 KMS 金鑰存取的許可。第二個陳述式提供 test-user 使用者在 KMS 金鑰上執行 describe-key 和 list-keys 命令的許可權。

key_policy.json 的內容：

```
{  
  "Version": "2012-10-17",  
  "Id" : "key-default-1",  
  "Statement" : [  
    {  
      "Sid" : "Enable IAM User Permissions",  
      "Effect" : "Allow",
```

```

        "Principal" : {
            "AWS" : "arn:aws:iam::111122223333:root"
        },
        "Action" : "kms:*",
        "Resource" : "*"
    },
    {
        "Sid" : "Allow Use of Key",
        "Effect" : "Allow",
        "Principal" : {
            "AWS" : "arn:aws:iam::111122223333:user/test-user"
        },
        "Action" : [
            "kms:DescribeKey",
            "kms:ListKeys"
        ],
        "Resource" : "*"
    }
]
}

```

若要識別 KMS 金鑰，此範例會使用金鑰 ID，但您也可以使用金鑰 ARN。為指定金鑰政策，命令使用 `policy` 參數。為了指出政策位於檔案中，它會使用必要的 `file://` 字首。要識別所有支援的作業系統上的檔案，需要此字首。最後，命令會使用值為 `default` 的 `policy-name` 參數。若未指定政策名稱，預設值為 `default`。唯一有效的值為 `default`。

```

aws kms put-key-policy \
  --policy-name default \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --policy file://key_policy.json

```

此命令不會產生任何輸出。若要驗證命令是否有效，請使用 `get-key-policy` 命令。下列範例命令會取得相同 KMS 金鑰的金鑰政策。值為 `text` 的 `output` 參數會傳回易於讀取的文字格式。

```

aws kms get-key-policy \
  --policy-name default \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --output text

```

輸出：

```
{
  "Version": "2012-10-17",
  "Id": "key-default-1",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "Allow Use of Key",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:user/test-user"
      },
      "Action": [ "kms:Describe", "kms:List" ],
      "Resource": "*"
    }
  ]
}
```

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的[變更金鑰政策](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [PutKeyPolicy](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/*...
```

```
* @param string $keyId
* @param string $policy
* @return void
*/
public function putKeyPolicy(string $keyId, string $policy)
{
    try {
        $this->client->putKeyPolicy([
            'KeyId' => $keyId,
            'Policy' => $policy,
        ]);
    } catch (KmsException $caught) {
        echo "There was a problem replacing the key policy: {$caught->getAwsErrorMessage()}\n";
        throw $caught;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [PutKeyPolicy](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class KeyPolicy:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyPolicy":
        """
        Creates a KeyPolicy instance with a default KMS client.
```



```

        :return: An instance of KeyPolicy initialized with the default KMS
client.
    """
    kms_client = boto3.client("kms")
    return cls(kms_client)

    def set_policy(self, key_id: str, policy: dict[str, any]) -> None:
        """
        Sets the policy of a key. Setting a policy entirely overwrites the
existing
        policy, so care is taken to add a statement to the existing list of
statements
        rather than simply writing a new policy.

        :param key_id: The ARN or ID of the key to set the policy to.
        :param policy: The existing policy of the key.
        :return: None
        """
        principal = input(
            "Enter the ARN of an IAM role to set as the principal on the policy:
"
        )
        if key_id != "" and principal != "":
            # The updated policy replaces the existing policy. Add a new
statement to
            # the list along with the original policy statements.
            policy["Statement"].append(
                {
                    "Sid": "Allow access for ExampleRole",
                    "Effect": "Allow",
                    "Principal": {"AWS": principal},
                    "Action": [
                        "kms:Encrypt",
                        "kms:GenerateDataKey*",
                        "kms:Decrypt",
                        "kms:DescribeKey",
                        "kms:ReEncrypt*",
                    ],
                    "Resource": "*",
                }
            )
        try:

```

```

        self.kms_client.put_key_policy(KeyId=key_id,
Policy=json.dumps(policy))
    except ClientError as err:
        logger.error(
            "Couldn't set policy for key %s. Here's why %s",
            key_id,
            err.response["Error"]["Message"],
        )
        raise
    else:
        print(f"Set policy for key {key_id}.")
else:
    print("Skipping set policy demo.")

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [PutKeyPolicy](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    " iv_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'
    " iv_policy = '{"Version":"2012-10-17",          "Statement": [...]}'
    lo_kms->putkeypolicy(
        iv_keyid = iv_key_id
        iv_policyname = 'default'
        iv_policy = iv_policy
    ).
    MESSAGE 'Key policy updated successfully.' TYPE 'I'.
CATCH /aws1/cx_kmsnotfoundexception.
    MESSAGE 'Key not found.' TYPE 'E'.
CATCH /aws1/cx_kmsmalformedplydocex.
    MESSAGE 'Malformed policy document.' TYPE 'E'.

```

```
CATCH /aws1/cx_kmskmsinternalex.  
    MESSAGE 'An internal error occurred.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [PutKeyPolicy](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ReEncrypt 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ReEncrypt。

CLI

AWS CLI

範例 1：在不同的對稱 KMS 金鑰 (Linux 和 macOS) 下，重新加密訊息。

下列re-encrypt命令範例示範使用 AWS CLI 重新加密資料的建議方法。

在檔案中提供密文。--ciphertext-blob 參數的值中，使用 fileb:// 字首，其會告知 CLI 從二進位檔案讀取資料。如果檔案不在目前的目錄中，請輸入檔案的完整路徑。如需從檔案讀取 AWS CLI 參數值的詳細資訊，請參閱《AWS 命令列界面使用者指南》中的[從檔案載入 AWS CLI 參數](#)，以及《AWS 命令列工具部落格》中的[本機檔案參數最佳實務](#)。指定解密加密文字的來源 KMS 金鑰。使用對稱加密 KMS 金鑰解密時不需要 --source-key-id 參數。AWS KMS 可以取得用於加密加密加密加密文字 Blob 中中繼資料之資料的 KMS 金鑰。但是指定您正在使用的 KMS 金鑰永遠是最佳實務。此做法可確保使用您想要的 KMS 金鑰，並可防止不小心使用不信任的 KMS 金鑰解密加密文字。指定目的地 KMS 金鑰，這會重新加密資料。--destination-key-id 參數一律為必要項。此範例使用金鑰 ARN，但您可以使用任何有效的金鑰識別碼。請求純文字輸出做為文字值。--query 參數會告知 CLI 僅從輸出取得 Plaintext 欄位的值。--output 參數會以純文字傳回輸出。Base64 將純文字解碼，並儲存在檔案中。下列範例會將 Plaintext 參數的管道符號 (|) 值輸送至 Base64 公用程式，以將其解碼。然後，將解碼的輸出重新導向 (>) 至 ExamplePlaintext 檔案。

執行此命令之前，請將範例金鑰 IDs 取代為 AWS 帳戶中有效的金鑰識別符。

```
aws kms re-encrypt \
```

```
--ciphertext-blob fileb://ExampleEncryptedFile \  
--source-key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
--destination-key-id 0987dcba-09fe-87dc-65ba-ab0987654321 \  
--query CiphertextBlob \  
--output text | base64 --decode > ExampleReEncryptedFile
```

此命令不會產生輸出。來自 re-encrypt 命令的輸出經過 base64 解碼，並儲存在檔案中。

如需詳細資訊，請參閱《AWS Key Management Service API 參考》中的 [ReEncrypt](#)。

範例 2：在不同的對稱 KMS 金鑰 (Windows 命令提示) 下，重新加密已加密的訊息。

下列 re-encrypt 命令範例與上一個範例相同，唯一不同的是它使用 certutil 公用程式對純文字資料進行 Base64 解碼。此程序需要兩個命令，如下列範例所示。

執行此命令之前，請將範例金鑰 ID 取代為來自您 AWS 帳戶的有效金鑰 ID。

```
aws kms re-encrypt ^  
--ciphertext-blob fileb://ExampleEncryptedFile ^  
--source-key-id 1234abcd-12ab-34cd-56ef-1234567890ab ^  
--destination-key-id 0987dcba-09fe-87dc-65ba-ab0987654321 ^  
--query CiphertextBlob ^  
--output text > ExampleReEncryptedFile.base64
```

然後使用 certutil 公用程式

```
certutil -decode ExamplePlaintextFile.base64 ExamplePlaintextFile
```

輸出：

```
Input Length = 18  
Output Length = 12  
CertUtil: -decode command completed successfully.
```

如需詳細資訊，請參閱《AWS Key Management Service API 參考》中的 [ReEncrypt](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ReEncrypt](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class KeyEncrypt:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyEncrypt":
        """
        Creates a KeyEncrypt instance with a default KMS client.

        :return: An instance of KeyEncrypt initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def re_encrypt(self, source_key_id, cipher_text):
        """
        Takes ciphertext previously encrypted with one key and reencrypt it by
using
        another key.

        :param source_key_id: The ARN or ID of the original key used to encrypt
the
                           ciphertext.
        :param cipher_text: The encrypted ciphertext.
        :return: The ciphertext encrypted by the second key.
        """
        destination_key_id = input(
            f"Your ciphertext is currently encrypted with key {source_key_id}. "
            f"Enter another key ID or ARN to reencrypt it: "
        )
```

```

if destination_key_id != "":
    try:
        cipher_text = self.kms_client.re_encrypt(
            SourceKeyId=source_key_id,
            DestinationKeyId=destination_key_id,
            CiphertextBlob=cipher_text,
        )["CiphertextBlob"]
    except ClientError as err:
        logger.error(
            "Couldn't reencrypt your ciphertext. Here's why: %s",
            err.response["Error"]["Message"],
        )
    else:
        print(f"Reencrypted your ciphertext as: {cipher_text}")
        return cipher_text
else:
    print("Skipping reencryption demo.")

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [ReEncrypt](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

require 'aws-sdk-kms' # v2: require 'aws-sdk'

# Human-readable version of the ciphertext of the data to reencrypt.

blob =
  '01020200785d68faeec386af1057904926253051eb2919d3c16078badf65b808b26dd057c101747cadf3593'
sourceCiphertextBlob = [blob].pack('H*')

# Replace the fictitious key ARN with a valid key ID

```

```

destinationKeyId = 'arn:aws:kms:us-
west-2:111122223333:key/0987dcba-09fe-87dc-65ba-ab0987654321'

client = Aws::KMS::Client.new(region: 'us-west-2')

resp = client.re_encrypt({
    ciphertext_blob: sourceCiphertextBlob,
    destination_key_id: destinationKeyId
})

# Display a readable version of the resulting re-encrypted blob.
puts 'Blob:'
puts resp.ciphertext_blob.unpack('H*')

```

- 如需詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [ReEncrypt](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

async fn reencrypt_string(
    verbose: bool,
    client: &Client,
    input_file: &str,
    output_file: &str,
    first_key: &str,
    new_key: &str,
) -> Result<(), Error> {
    // Get blob from input file
    // Open input text file and get contents as a string
    // input is a base-64 encoded string, so decode it:
    let data = fs::read_to_string(input_file)
        .map(|input_file| base64::decode(input_file).expect("invalid base 64"))
        .map(Blob::new);

```

```
let resp = client
    .re_encrypt()
    .ciphertext_blob(data.unwrap())
    .source_key_id(first_key)
    .destination_key_id(new_key)
    .send()
    .await?;

// Did we get an encrypted blob?
let blob = resp.ciphertext_blob.expect("Could not get encrypted text");
let bytes = blob.as_ref();

let s = base64::encode(bytes);
let o = &output_file;

let mut ofile = File::create(o).expect("unable to create file");
ofile.write_all(s.as_bytes()).expect("unable to write");

if verbose {
    println!("Wrote the following to {}: ", output_file);
    println!("{}", s);
} else {
    println!("Wrote base64-encoded output to {}", output_file);
}

Ok(())
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Rust API 參考》中的 [ReEncrypt](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。


```

TRY.
    " iv_source_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'
    " iv_destination_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/5678dcba-56cd-78ef-90ab-5678901234cd'
    " iv_ciphertext_blob contains the encrypted data
    oo_result = lo_kms->reencrypt(
        iv_sourcekeyid = iv_source_key_id
        iv_destinationkeyid = iv_destination_key_id
        iv_ciphertextblob = iv_ciphertext_blob
    ).
    MESSAGE 'Ciphertext reencrypted successfully.' TYPE 'I'.
CATCH /aws1/cx_kmsdisabledexception.
    MESSAGE 'The key is disabled.' TYPE 'E'.
CATCH /aws1/cx_kmsincorrectkeyex.
    MESSAGE 'Incorrect source key for decryption.' TYPE 'E'.
CATCH /aws1/cx_kmsnotfoundexception.
    MESSAGE 'Key not found.' TYPE 'E'.
CATCH /aws1/cx_kmskmsinternalex.
    MESSAGE 'An internal error occurred.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [ReEncrypt](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

RetireGrant 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 RetireGrant。

CLI

AWS CLI

淘汰客戶主金鑰的授予

下列 retire-grant 範例會從 KMS 金鑰刪除授予。

下列範例命令會指定 grant-id 和 key-id 參數。key-id 參數的值必須是 KMS 金鑰的金鑰 ARN。

```
aws kms retire-grant \
  --grant-id 1234a2345b8a4e350500d432bccf8ecd6506710e1391880c4f7f7140160c9af3 \
  --key-id arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab
```

此命令不會產生輸出。若要確認授予已淘汰，請使用 `list-grants` 命令。

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的[淘汰和撤銷授予](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [RetireGrant](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class GrantManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "GrantManager":
        """
        Creates a GrantManager instance with a default KMS client.

        :return: An instance of GrantManager initialized with the default KMS
        client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def retire_grant(self, grant):
        """
        Retires a grant so that it can no longer be used.
```

```

:param grant: The grant to retire.
"""
try:
    self.kms_client.retire_grant(GrantToken=grant["GrantToken"])
except ClientError as err:
    logger.error(
        "Couldn't retire grant %s. Here's why: %s",
        grant["GrantId"],
        err.response["Error"]["Message"],
    )
else:
    print(f"Grant {grant['GrantId']} retired.")

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [RetireGrant](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
  " iv_grant_token = 'AQpAM2RhZ...'
  lo_kms->retiregrant( iv_granttoken = iv_grant_token ).
  MESSAGE 'Grant retired successfully.' TYPE 'I'.
CATCH /aws1/cx_kmsnotfoundexception.
  MESSAGE 'Grant not found.' TYPE 'E'.
CATCH /aws1/cx_kmsinvgranttokenex.
  MESSAGE 'Invalid grant token.' TYPE 'E'.
CATCH /aws1/cx_kmskmsinternalex.
  MESSAGE 'An internal error occurred.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [RetireGrant](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

RevokeGrant 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 RevokeGrant。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

CLI

AWS CLI

撤銷客戶主金鑰的授予

下列 revoke-grant 範例會從 KMS 金鑰刪除授予。下列範例命令會指定 grant-id 和 key-id 參數。key-id 參數的值可以是 KMS 金鑰的金鑰 ID 或金鑰 ARN。

```
aws kms revoke-grant \  
  --grant-id 1234a2345b8a4e350500d432bccf8ecd6506710e1391880c4f7f7140160c9af3 \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

此命令不會產生輸出。若要確認授予已撤銷，請使用 list-grants 命令。

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的 [淘汰和撤銷授予](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [RevokeGrant](#)。

Java

SDK for Java 2.x

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Revokes a grant for the specified AWS KMS key asynchronously.
 *
 * @param keyId The ID or key ARN of the AWS KMS key.
 * @param grantId The identifier of the grant to be revoked.
 * @return A {@link CompletableFuture} representing the asynchronous
 * operation of revoking the grant.
 *
 * The {@link CompletableFuture} will complete with a {@link
 * RevokeGrantResponse} object
 *
 * if the operation is successful, or with a {@code null} value if an
 * error occurs.
 */
public CompletableFuture<RevokeGrantResponse> revokeKeyGrantAsync(String
keyId, String grantId) {
    RevokeGrantRequest grantRequest = RevokeGrantRequest.builder()
        .keyId(keyId)
        .grantId(grantId)
        .build();

    CompletableFuture<RevokeGrantResponse> responseFuture =
getAsyncClient().revokeGrant(grantRequest);
    responseFuture.whenComplete((response, exception) -> {
        if (exception == null) {
            logger.info("Grant ID: [" + grantId + "] was successfully
revoked!");
        } else {
            if (exception instanceof KmsException kmsEx) {
                if (kmsEx.getMessage().contains("Grant does not exist")) {
                    logger.info("The grant ID '" + grantId + "' does not
exist. Moving on...");
                } else {

```

```

        throw new RuntimeException("KMS error occurred: " +
kmsEx.getMessage(), kmsEx);
    }
    } else {
        throw new RuntimeException("An unexpected error occurred: " +
exception.getMessage(), exception);
    }
    }
});

return responseFuture;
}

```

- 如需詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [RevokeGrant](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

/**
 * @param string $grantId
 * @param string $keyId
 * @return void
 */
public function revokeGrant(string $grantId, string $keyId)
{
    try{
        $this->client->revokeGrant([
            'GrantId' => $grantId,
            'KeyId' => $keyId,
        ]);
    }catch(KmsException $caught){
        echo "There was a problem with revoking the grant: {$caught-
>getAwsErrorMessage()}.\\n";
    }
}

```

```

        throw $caught;
    }
}

```

- 如需詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [RevokeGrant](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

class GrantManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "GrantManager":
        """
        Creates a GrantManager instance with a default KMS client.

        :return: An instance of GrantManager initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def revoke_grant(self, key_id: str, grant_id: str) -> None:
        """
        Revokes a grant so that it can no longer be used.

        :param key_id: The ARN or ID of the key associated with the grant.
        :param grant_id: The ID of the grant to revoke.
        """
        try:

```

```

        self.kms_client.revoke_grant(KeyId=key_id, GrantId=grant_id)
    except ClientError as err:
        logger.error(
            "Couldn't revoke grant %s. Here's why: %s",
            grant_id,
            err.response["Error"]["Message"],
        )
        raise

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [RevokeGrant](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    " iv_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'
    " iv_grant_id = '1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p'
    lo_kms->revokegrant(
        iv_keyid = iv_key_id
        iv_grantid = iv_grant_id
    ).
    MESSAGE 'Grant revoked successfully.' TYPE 'I'.
CATCH /aws1/cx_kmsnotfoundexception.
    MESSAGE 'Grant or key not found.' TYPE 'E'.
CATCH /aws1/cx_kmsinvalidgrantidex.
    MESSAGE 'Invalid grant ID.' TYPE 'E'.
CATCH /aws1/cx_kmskmsinternalex.
    MESSAGE 'An internal error occurred.' TYPE 'E'.
ENDTRY.

```


- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [RevokeGrant](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ScheduleKeyDeletion 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ScheduleKeyDeletion。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

CLI

AWS CLI

為客戶自管 KMS 金鑰刪除排程。

下列 schedule-key-deletion 範例為需要在 15 天內刪除的指定客戶自管 KMS 金鑰排程。

--key-id 參數可識別 KMS 金鑰。此範例使用金鑰 ARN 值，但您可以使用金鑰 ID 或 KMS 金鑰的 ARN。--pending-window-in-days 參數會指定 7-30 天等待期的時間長度。預設等待期為 30 天。此範例會指定值 15，指示 AWS 命令完成後 15 天永久刪除 KMS 金鑰。

```
aws kms schedule-key-deletion \
  --key-id arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab \
  --pending-window-in-days 15
```

回應包含金鑰 ARN、金鑰狀態、等待期 (PendingWindowInDays)，以及以 Unix 時間表示的刪除日期。若要以當地時間檢視刪除日期，請使用 AWS KMS 主控台。PendingDeletion 金鑰狀態中的 KMS 金鑰，無法在密碼編譯操作中使用。

```
{
  "KeyId": "arn:aws:kms:us-
west-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "DeletionDate": "2022-06-18T23:43:51.272000+00:00",
  "KeyState": "PendingDeletion",
```

```
"PendingWindowInDays": 15
}
```

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的[刪除金鑰](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ScheduleKeyDeletion](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Deletes a KMS key asynchronously.
 *
 * <p><strong>Warning:</strong> Deleting a KMS key is a destructive and
 * potentially dangerous operation.
 * When a KMS key is deleted, all data that was encrypted under the KMS key
 * becomes unrecoverable.
 * This means that any files, databases, or other data that were encrypted
 * using the deleted KMS key
 * will become permanently inaccessible. Exercise extreme caution when
 * deleting KMS keys.</p>
 *
 * @param keyId the ID of the KMS key to delete
 * @return a {@link CompletableFuture} that completes when the key deletion
 * is scheduled
 */
public CompletableFuture<Void> deleteKeyAsync(String keyId) {
    ScheduleKeyDeletionRequest deletionRequest =
ScheduleKeyDeletionRequest.builder()
        .keyId(keyId)
        .pendingWindowInDays(7)
        .build();

    return getAsyncClient().scheduleKeyDeletion(deletionRequest)
        .thenRun(() -> {
```

```
        logger.info("Key {} will be deleted in 7 days", keyId);
    })
    .exceptionally(throwable -> {
        throw new RuntimeException("Failed to schedule key deletion for
key ID: " + keyId, throwable);
    });
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ScheduleKeyDeletion](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * @param string $keyId
 * @param int $pendingWindowInDays
 * @return void
 */
public function scheduleKeyDeletion(string $keyId, int $pendingWindowInDays =
7)
{
    try {
        $this->client->scheduleKeyDeletion([
            'KeyId' => $keyId,
            'PendingWindowInDays' => $pendingWindowInDays,
        ]);
    } catch (KmsException $caught){
        echo "There was a problem scheduling the key deletion: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
}
```

```
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [ScheduleKeyDeletion](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
    def from_client(cls) -> "KeyManager":
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
        client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def delete_key(self, key_id: str, window: int) -> None:
        """
        Deletes a list of keys.

        Warning:
        Deleting a KMS key is a destructive and potentially dangerous operation.
        When a KMS key is deleted,
```

```

    all data that was encrypted under the KMS key is unrecoverable.

    :param key_id: The ARN or ID of the key to delete.
    :param window: The waiting period, in days, before the KMS key is
deleted.
    """

    try:
        self.kms_client.schedule_key_deletion(
            KeyId=key_id, PendingWindowInDays=window
        )
    except ClientError as err:
        logging.error(
            "Couldn't delete key %s. Here's why: %s",
            key_id,
            err.response["Error"]["Message"],
        )
        raise

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [ScheduleKeyDeletion](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    " iv_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'
    " iv_pending_window_days = 7
    oo_result = lo_kms->schedulekeydeletion(
        iv_keyid = iv_key_id
        iv_pendingwindowdays = iv_pending_window_days

```

```
    ).  
    MESSAGE 'Key scheduled for deletion.' TYPE 'I'.  
  CATCH /aws1/cx_kmsnotfoundexception.  
    MESSAGE 'Key not found.' TYPE 'E'.  
  CATCH /aws1/cx_kmskmsinternalex.  
    MESSAGE 'An internal error occurred.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [ScheduleKeyDeletion](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Sign 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 Sign。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

CLI

AWS CLI

範例 1：產生訊息的數位簽章

下列 sign 範例會產生簡訊的密碼編譯簽章。命令的輸出包含 base-64 編碼 Signature 欄位，您可以使用 verify 命令進行驗證。

您必須指定要簽署的訊息，以及非對稱 KMS 金鑰支援的簽署演算法。若要取得 KMS 金鑰的簽署演算法，請使用 describe-key 命令。

在 AWS CLI v2 中，message 參數的值必須是 Base64-encoded。或者，您可以將訊息儲存在檔案中，並使用 fileb://字首，告知 AWS CLI 從檔案讀取二進位資料。

執行此命令之前，請將範例金鑰 ID 取代為來自您 AWS 帳戶的有效金鑰 ID。金鑰 ID 必須代表金鑰用量為 SIGN_VERIFY 的非對稱 KMS 金鑰。

```
msg=(echo 'Hello World' | base64)

aws kms sign \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --message fileb://UnsignedMessage \
  --message-type RAW \
  --signing-algorithm RSASSA_PKCS1_V1_5_SHA_256
```

輸出：

```
{
  "KeyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "Signature": "ABCDEFhpyVYyTxbaFE74ccSvEJLJr3zuoV1Hfymz4qv+/
fxmxNLA7SE1SiF8lHw80fKZZ3bJ...",
  "SigningAlgorithm": "RSASSA_PKCS1_V1_5_SHA_256"
}
```

如需在 AWS KMS 中使用非對稱 KMS 金鑰的詳細資訊，請參閱 Key Management Service 開發人員指南中的 [AWS KMS 中的非對稱金鑰](#)。AWS

範例 2：在檔案 (Linux 和 macOS) 中儲存數位簽章

下列 sign 範例會為儲存在本機檔案中的簡訊產生密碼編譯簽章。命令也會從回應取得 Signature 屬性，以 Base64 進行解碼，然後將其儲存在 ExampleSignature 檔案中。您可以在驗證簽章的 verify 命令中使用簽章檔案。

sign 命令需要以 Base64 編碼的訊息，以及非對稱 KMS 金鑰支援的簽署演算法。若要取得 KMS 金鑰支援的簽署演算法，請使用 describe-key 命令。

執行此命令之前，請將範例金鑰 ID 取代為來自您 AWS 帳戶的有效金鑰 ID。金鑰 ID 必須代表金鑰用量為 SIGN_VERIFY 的非對稱 KMS 金鑰。

```
echo 'hello world' | base64 > EncodedMessage

aws kms sign \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --message fileb://EncodedMessage \
  --message-type RAW \
  --signing-algorithm RSASSA_PKCS1_V1_5_SHA_256 \
  --output text \
```

```
--query Signature | base64 --decode > ExampleSignature
```

此命令不會產生輸出。此範例會擷取輸出的 Signature 屬性，並將其儲存在檔案中。

如需在 AWS KMS 中使用非對稱 KMS 金鑰的詳細資訊，請參閱 Key Management Service 開發人員指南中的 [AWS KMS 中的非對稱金鑰](#)。AWS

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [Sign](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Asynchronously signs and verifies data using AWS KMS.
 *
 * <p>The method performs the following steps:
 * <ol>
 *     <li>Creates an AWS KMS key with the specified key spec, key usage, and
origin.</li>
 *     <li>Signs the provided message using the created KMS key and the
RSASSA-PSS-SHA-256 algorithm.</li>
 *     <li>Verifies the signature of the message using the created KMS key
and the RSASSA-PSS-SHA-256 algorithm.</li>
 * </ol>
 *
 * @return a {@link CompletableFuture} that completes with the result of the
signature verification,
 *         {@code true} if the signature is valid, {@code false} otherwise.
 * @throws KmsException if any error occurs during the KMS operations.
 * @throws RuntimeException if an unexpected error occurs.
 */
public CompletableFuture<Boolean> signVerifyDataAsync() {
    String signMessage = "Here is the message that will be digitally signed";

    // Create an AWS KMS key used to digitally sign data.
```



```

        CreateKeyRequest createKeyRequest = CreateKeyRequest.builder()
            .keySpec(KeySpec.RSA_2048)
            .keyUsage(KeyUsageType.SIGN_VERIFY)
            .origin(OriginType.AWS_KMS)
            .build();

        return getAsyncClient().createKey(createKeyRequest)
            .thenCompose(createKeyResponse -> {
                String keyId = createKeyResponse.keyMetadata().keyId();

                SdkBytes messageBytes = SdkBytes.fromString(signMessage,
Charset.defaultCharset());
                SignRequest signRequest = SignRequest.builder()
                    .keyId(keyId)
                    .message(messageBytes)
                    .signingAlgorithm(SigningAlgorithmSpec.RSASSA_PSS_SHA_256)
                    .build();

                return getAsyncClient().sign(signRequest)
                    .thenCompose(signResponse -> {
                        byte[] signedBytes =
signResponse.signature().asByteArray();

                        VerifyRequest verifyRequest = VerifyRequest.builder()
                            .keyId(keyId)

                            .message(SdkBytes.fromByteArray(signMessage.getBytes(Charset.defaultCharset())))

                            .signature(SdkBytes.fromByteBuffer(ByteBuffer.wrap(signedBytes)))

                            .signingAlgorithm(SigningAlgorithmSpec.RSASSA_PSS_SHA_256)
                                .build();

                        return getAsyncClient().verify(verifyRequest)
                            .thenApply(verifyResponse -> {
                                return (boolean) verifyResponse.signatureValid();
                            });
                    });
            })
            .exceptionally(throwable -> {
                throw new RuntimeException("Failed to sign or verify data",
throwable);
            });
    }

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [Sign](#)。

PHP

適用於 PHP 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * @param string $keyId
 * @param string $message
 * @param string $algorithm
 * @return Result
 */
public function sign(string $keyId, string $message, string $algorithm)
{
    try {
        return $this->client->sign([
            'KeyId' => $keyId,
            'Message' => $message,
            'SigningAlgorithm' => $algorithm,
        ]);
    } catch (KmsException $caught){
        echo "There was a problem signing the data: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 PHP 的 AWS SDK API 參考》中的 [Sign](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class KeyEncrypt:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyEncrypt":
        """
        Creates a KeyEncrypt instance with a default KMS client.

        :return: An instance of KeyEncrypt initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def sign(self, key_id: str, message: str) -> str:
        """
        Signs a message with a key.

        :param key_id: The ARN or ID of the key to use for signing.
        :param message: The message to sign.
        :return: The signature of the message.
        """
        try:
            return self.kms_client.sign(
                KeyId=key_id,
                Message=message.encode(),
                SigningAlgorithm="RSASSA_PSS_SHA_256",
            )["Signature"]
        except ClientError as err:
            logger.error(
```

```

        "Couldn't sign your message. Here's why: %s",
        err.response["Error"]["Message"],
    )
    raise

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [Sign](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    " iv_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab' (asymmetric key)
    " iv_message contains the message to sign
    " iv_signing_algorithm = 'RSASSA_PSS_SHA_256'
    oo_result = lo_kms->sign(
        iv_keyid = iv_key_id
        iv_message = iv_message
        iv_signingalgorithm = iv_signing_algorithm
    ).
    MESSAGE 'Message signed successfully.' TYPE 'I'.
CATCH /aws1/cx_kmsdisabledexception.
    MESSAGE 'The key is disabled.' TYPE 'E'.
CATCH /aws1/cx_kmsnotfoundexception.
    MESSAGE 'Key not found.' TYPE 'E'.
CATCH /aws1/cx_kmsinvalidkeyusageex.
    MESSAGE 'Key cannot be used for signing.' TYPE 'E'.
CATCH /aws1/cx_kmskmsinternalex.
    MESSAGE 'An internal error occurred.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱[登入](#)適用於 AWS SAP ABAP 的 SDK API 參考。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

TagResource 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 TagResource。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [了解基本概念](#)

CLI

AWS CLI

將標籤新增至 KMS 金鑰

下列 tag-resource 範例會將 "Purpose":"Test" 和 "Dept":"IT" 標籤新增至客戶自管 KMS 金鑰。您可以使用這類標籤來標記 KMS 金鑰，並建立 KMS 金鑰類別以供許可和稽核之用。

若要指定 KMS 金鑰，請使用 key-id 參數。此範例使用金鑰 ID 值，但您可以在此命令中使用金鑰 ID 或金鑰 ARN。

```
aws kms tag-resource \  
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \  
  --tags TagKey='Purpose',TagValue='Test' TagKey='Dept',TagValue='IT'
```

此命令不會產生輸出。若要檢視 AWS KMS KMS 金鑰上的標籤，請使用 list-resource-tags 命令。

如需在 AWS KMS 中使用標籤的詳細資訊，請參閱 AWS Key Management Service 開發人員指南中的[標記金鑰](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [TagResource](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * Asynchronously tags a KMS key with a specific tag.
 *
 * @param keyId the ID of the KMS key to be tagged
 * @return a {@link CompletableFuture} that completes when the tagging
 * operation is finished
 */
public CompletableFuture<Void> tagKMSKeyAsync(String keyId) {
    Tag tag = Tag.builder()
        .tagKey("Environment")
        .tagValue("Production")
        .build();

    TagResourceRequest tagResourceRequest = TagResourceRequest.builder()
        .keyId(keyId)
        .tags(tag)
        .build();

    return getAsyncClient().tagResource(tagResourceRequest)
        .thenRun(() -> {
            logger.info("{} key was tagged", keyId);
        })
        .exceptionally(throwable -> {
            throw new RuntimeException("Failed to tag the KMS key",
throwable);
        });
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API 參考中的 [TagResource](#)。

PHP

適用於 PHP 的 SDK

 Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
/**
 * @param string $keyId
 * @param array $tags
 * @return void
 */
public function tagResource(string $keyId, array $tags)
{
    try {
        $this->client->tagResource([
            'KeyId' => $keyId,
            'Tags' => $tags,
        ]);
    } catch (KmsException $caught) {
        echo "There was a problem applying the tag(s): {"$caught->getAwsErrorMessage()}\n";
        throw $caught;
    }
}
```

- 如需 API 詳細資訊，請參閱 適用於 PHP 的 AWS SDK API 參考中的 [TagResource](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class KeyManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_keys = []

    @classmethod
    def from_client(cls) -> "KeyManager":
        """
        Creates a KeyManager instance with a default KMS client.

        :return: An instance of KeyManager initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def tag_resource(self, key_id: str, tag_key: str, tag_value: str) -> None:
        """
        Add or edit tags on a customer managed key.

        :param key_id: The ARN or ID of the key to enable rotation for.
        :param tag_key: Key for the tag.
        :param tag_value: Value for the tag.
        """
        try:
            self.kms_client.tag_resource(
                KeyId=key_id, Tags=[{"TagKey": tag_key, "TagValue": tag_value}]
            )
        except ClientError as err:
            logging.error(
                "Couldn't add a tag for the key '%s'. Here's why: %s",

```



```

        key_id,
        err.response["Error"]["Message"],
    )
    raise

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [TagResource](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

DATA lt_tags TYPE /aws1/cl_kmstag=>tt_taglist.

TRY.
    " iv_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab'
    " iv_tag_key = 'Environment'
    " iv_tag_value = 'Production'
    APPEND NEW /aws1/cl_kmstag(
        iv_tagkey = iv_tag_key
        iv_tagvalue = iv_tag_value
    ) TO lt_tags.

    lo_kms->tagresource(
        iv_keyid = iv_key_id
        it_tags = lt_tags
    ).
    MESSAGE 'Tag added to KMS key successfully.' TYPE 'I'.
CATCH /aws1/cx_kmsnotfoundexception.
    MESSAGE 'Key not found.' TYPE 'E'.
CATCH /aws1/cx_kmstagexception.
    MESSAGE 'Invalid tag format.' TYPE 'E'.
CATCH /aws1/cx_kmskmsinternalex.
    MESSAGE 'An internal error occurred.' TYPE 'E'.

```

```
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [TagResource](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

UpdateAlias 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 UpdateAlias。

CLI

AWS CLI

將別名與不同的 KMS 金鑰相關聯

下列 update-alias 範例會將別名 alias/test-key 與不同的 KMS 金鑰相關聯。

請注意，--alias-name 參數會指定別名。別名的名稱值必須以 alias/ 開頭。--target-key-id 參數會指定要與別名建立關聯的 KMS 金鑰。您不需要為別名指定目前的 KMS 金鑰。

```
aws kms update-alias \  
  --alias-name alias/test-key \  
  --target-key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

此命令不會產生輸出。要尋找別名，請使用 list-aliases 命令。

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的 [更新別名](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [UpdateAlias](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class AliasManager:
    def __init__(self, kms_client):
        self.kms_client = kms_client
        self.created_key = None

    @classmethod
    def from_client(cls) -> "AliasManager":
        """
        Creates an AliasManager instance with a default KMS client.

        :return: An instance of AliasManager initialized with the default KMS
client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def update_alias(self, alias, current_key_id):
        """
        Updates an alias by assigning it to another key.

        :param alias: The alias to reassign.
        :param current_key_id: The ARN or ID of the key currently associated with
the alias.
        """
        new_key_id = input(
            f"Alias {alias} is currently associated with {current_key_id}. "
            f"Enter another key ID or ARN that you want to associate with
{alias}: "
        )
        if new_key_id != "":
            try:
```

```

        self.kms_client.update_alias(AliasName=alias,
TargetKeyId=new_key_id)
    except ClientError as err:
        logger.error(
            "Couldn't associate alias %s with key %s. Here's why: %s",
            alias,
            new_key_id,
            err.response["Error"]["Message"],
        )
    else:
        print(f"Alias {alias} is now associated with key {new_key_id}.")
else:
    print("Skipping alias update.")

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [UpdateAlias](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    " iv_alias_name = 'alias/my-key-alias'
    " iv_target_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/5678dcba-56cd-78ef-90ab-5678901234cd'
    lo_kms->updatealias(
        iv_aliasname = iv_alias_name
        iv_targetkeyid = iv_target_key_id
    ).
    MESSAGE 'Alias updated successfully.' TYPE 'I'.
CATCH /aws1/cx_kmsnotfoundexception.
    MESSAGE 'Alias or key not found.' TYPE 'E'.
CATCH /aws1/cx_kmskmsinternalex.
    MESSAGE 'An internal error occurred.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [UpdateAlias](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Verify 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 Verify。

CLI

AWS CLI

驗證數位簽章

下列 `verify` 命令會驗證以 Base64 編碼的簡短訊息的加密簽章。金鑰 ID、訊息、訊息類型和簽署演算法，必須與用來簽署訊息的演算法相同。

在 AWS CLI v2 中，`message` 參數的值必須是 Base64-encoded。或者，您可以將訊息儲存在檔案中，並使用 `fileb://` 字首，告知 AWS CLI 從檔案讀取二進位資料。

您指定的簽章不能以 base64 編碼。如需將 `sign` 命令傳回的簽章解碼的說明，請參閱 `sign` 命令範例。

命令的輸出包含布林值 `SignatureValid` 欄位，指出簽章已驗證。如果簽章驗證失敗，`verify` 命令也會失敗。

執行此命令之前，請將範例金鑰 ID 取代為來自您 AWS 帳戶的有效金鑰 ID。

```
aws kms verify \
  --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
  --message fileb://EncodedMessage \
  --message-type RAW \
  --signing-algorithm RSASSA_PKCS1_V1_5_SHA_256 \
  --signature fileb://ExampleSignature
```

輸出：

```
{
```

```
"KeyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
"SignatureValid": true,
"SigningAlgorithm": "RSASSA_PKCS1_V1_5_SHA_256"
}
```

如需在 KMS 中使用非對稱 AWS KMS 金鑰的詳細資訊，請參閱 AWS Key Management Service 開發人員指南中的[使用非對稱金鑰](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的[Verify](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[AWS 程式碼範例儲存庫](#)中設定和執行。

```
class KeyEncrypt:
    def __init__(self, kms_client):
        self.kms_client = kms_client

    @classmethod
    def from_client(cls) -> "KeyEncrypt":
        """
        Creates a KeyEncrypt instance with a default KMS client.

        :return: An instance of KeyEncrypt initialized with the default KMS
        client.
        """
        kms_client = boto3.client("kms")
        return cls(kms_client)

    def verify(self, key_id: str, message: str, signature: str) -> bool:
        """
        Verifies a signature against a message.

        :param key_id: The ARN or ID of the key used to sign the message.
```

```

:param message: The message to verify.
:param signature: The signature to verify.
:return: True when the signature matches the message, otherwise False.
"""
try:
    response = self.kms_client.verify(
        KeyId=key_id,
        Message=message.encode(),
        Signature=signature,
        SigningAlgorithm="RSASSA_PSS_SHA_256",
    )
    valid = response["SignatureValid"]
    print(f"The signature is {'valid' if valid else 'invalid'}.")
    return valid
except ClientError as err:
    if err.response["Error"]["Code"] == "SignatureDoesNotMatchException":
        print("The signature is not valid.")
    else:
        logger.error(
            "Couldn't verify your signature. Here's why: %s",
            err.response["Error"]["Message"],
        )
    raise

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [Verify](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

TRY.

```

" iv_key_id = 'arn:aws:kms:us-
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab' (asymmetric key)
" iv_message contains the original message

```

```

" iv_signature contains the signature to verify
" iv_signing_algorithm = 'RSASSA_PSS_SHA_256'
oo_result = lo_kms->verify(
    iv_keyid = iv_key_id
    iv_message = iv_message
    iv_signature = iv_signature
    iv_signingalgorithm = iv_signing_algorithm
).
DATA(lv_valid) = oo_result->get_signaturevalid( ).
IF lv_valid = abap_true.
    MESSAGE 'Signature is valid.' TYPE 'I'.
ELSE.
    MESSAGE 'Signature is invalid.' TYPE 'I'.
ENDIF.
CATCH /aws1/cx_kmsdisabledexception.
    MESSAGE 'The key is disabled.' TYPE 'E'.
CATCH /aws1/cx_kmsnotfoundexception.
    MESSAGE 'Key not found.' TYPE 'E'.
CATCH /aws1/cx_kmskmsinvalidsigex.
    MESSAGE 'Invalid signature.' TYPE 'E'.
CATCH /aws1/cx_kmskmsinternalex.
    MESSAGE 'An internal error occurred.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的[驗證](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

AWS KMS 使用 AWS SDKs 案例

下列程式碼範例示範如何在 AWS SDKs 中 AWS KMS 實作常見案例。這些案例說明如何透過在 中呼叫多個函數 AWS KMS 或與其他函數結合，來完成特定任務 AWS 服務。每個案例均包含完整原始碼的連結，您可在連結中找到如何設定和執行程式碼的相關指示。

案例的目標是獲得中等水平的經驗，協助您了解內容中的服務動作。

範例

- [使用 AWS Command Line Interface v2 使用 DynamoDB 資料表加密](#)

使用 AWS Command Line Interface v2 使用 DynamoDB 資料表加密

下列程式碼範例示範如何管理 DynamoDB 資料表的加密選項。

- 建立具預設加密的儲存貯體。
- 利用客戶管理的 CMK 建立資料表。
- 更新資料表加密設定。
- 描述資料表加密。

Bash

AWS CLI 使用 Bash 指令碼

建立具預設加密的儲存貯體。

```
# Create a table with default encryption (AWS owned key)
aws dynamodb create-table \
  --table-name CustomerData \
  --attribute-definitions \
    AttributeName=CustomerID,AttributeType=S \
  --key-schema \
    AttributeName=CustomerID,KeyType=HASH \
  --billing-mode PAY_PER_REQUEST \
  --sse-specification Enabled=true,SSEType=KMS
```

利用客戶管理的 CMK 建立資料表。

```
# Step 1: Create a customer managed key in KMS
aws kms create-key \
  --description "Key for DynamoDB table encryption" \
  --key-usage ENCRYPT_DECRYPT \
  --customer-master-key-spec SYMMETRIC_DEFAULT

# Store the key ID for later use
KEY_ID=$(aws kms list-keys --query "Keys[?contains(KeyArn, 'Key for
DynamoDB')].KeyId" --output text)

# Step 2: Create a table with the customer managed key
aws dynamodb create-table \
  --table-name SensitiveData \
```

```
--attribute-definitions \  
    AttributeName=RecordID,AttributeType=S \  
--key-schema \  
    AttributeName=RecordID,KeyType=HASH \  
--billing-mode PAY_PER_REQUEST \  
--sse-specification Enabled=true,SSEType=KMS,KMSMasterKeyId=$KEY_ID
```

更新資料表加密。

```
# Update a table to use a different KMS key  
aws dynamodb update-table \  
    --table-name CustomerData \  
    --sse-specification Enabled=true,SSEType=KMS,KMSMasterKeyId=$KEY_ID
```

描述資料表加密。

```
# Describe the table to see encryption settings  
aws dynamodb describe-table \  
    --table-name CustomerData \  
    --query "Table.SSEDescription"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。
 - [CreateKey](#)
 - [CreateTable](#)
 - [DescribeTable](#)
 - [UpdateTable](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 AWS SDK 使用此服務](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

中的密碼編譯認證支援 AWS KMS

AWS KMS 支援 [AWS Nitro Enclaves](#) 和 [AWS NitroTPM](#) 的密碼編譯認證。支援這些認證方法的應用程式會使用已簽署的認證文件呼叫下列 AWS KMS 密碼編譯操作。會 AWS KMS 驗證認證文件來自有效的來源 (Nitro enclave 或 NitroTPM)。然後，這些 APIs 不會在回應中傳回純文字資料，而是使用證明文件中的公有金鑰來加密純文字，並傳回只能由 enclave 或 EC2 執行個體中對應的私有金鑰解密的加密文字。

- [解密](#)
- [DeriveSharedSecret](#)
- [GenerateDataKey](#)
- [GenerateDataKeyPair](#)
- [GenerateRandom](#)

下表顯示對已證明請求的回應與每個 API 操作的標準回應有何不同。

AWS KMS 操作	標準回應	對已證明請求的回應
Decrypt	傳回純文字資料	傳回純文字資料 (經來自證明文件的公有金鑰加密)
DeriveSharedSecret	傳回原始共用秘密	從證明文件傳回由公有金鑰加密的原始共用秘密
GenerateDataKey	傳回資料金鑰的純文字複本 (也會傳回 KMS 金鑰加密的資料金鑰複本)	傳回資料金鑰複本 (經來自證明文件的公有金鑰加密) (也會傳回 KMS 金鑰加密的資料金鑰複本)
GenerateDataKeyPair	傳回私有金鑰的純文字副本 (也會傳回公有金鑰以及由 KMS 金鑰加密的私有金鑰副本)	傳回私有金鑰複本 (經來自證明文件的公有金鑰加密) (也會傳回公有金鑰以及由 KMS 金鑰加密的私有金鑰副本)

AWS KMS 操作	標準回應	對已證明請求的回應
GenerateRandom	傳回隨機位元組字串	傳回隨機位元組字串 (經來自證明文件的公有金鑰加密)

AWS KMS 支援[政策條件金鑰](#)，您可以根據證明文件的內容，使用 AWS KMS 金鑰來允許或拒絕已驗證的操作。您也可以在 AWS CloudTrail 日誌中[監控對的已驗證請求 AWS KMS](#)。

進一步了解

- [密碼編譯認證](#)
- [AWS KMS 已驗證平台的條件索引鍵](#)
- [如何對 進行經驗證的呼叫 AWS KMS](#)
- [監控已驗證的請求](#)

如何對 進行經驗證的呼叫 AWS KMS

若要對 進行經證明的呼叫 AWS KMS，請在 請求中使用 Recipient 參數來提供已簽署的證明文件，以及與證明文件中的公有金鑰搭配使用的加密演算法。當請求包含已簽署證明文件的 Recipient 參數時，回應會包含 CiphertextForRecipient 欄位，以及由公有金鑰加密的密文。純文字欄位為空值或空白。

Recipient 參數必須指定來自 AWS Nitro Enclaves 或 AWS NitroTPM 的已簽署證明文件。AWS KMS 依賴證明文件的數位簽章，以證明請求中的公有金鑰來自有效的來源。您無法提供自己的憑證來數位簽署證明文件。

AWS Nitro Enclaves SDK 僅支援 Nitro enclave，會自動將 Recipient 參數及其值新增至每個 AWS KMS 請求。

若要在 AWS SDKs 中提出經證明的請求，您必須指定 Recipient 參數及其值。您可以使用 [nitro-tpm-attest 公用程式](#) 從 NitroTPM 擷取證明文件，或使用 NSM [API 從 Nitro 安全模組 \(NSM\)](#) 擷取證明文件。

AWS KMS 支援[政策條件金鑰](#)，您可以根據證明文件的內容，使用 AWS KMS 金鑰來允許或拒絕已驗證的操作。您也可以在 AWS CloudTrail 日誌中[監控對的已驗證請求 AWS KMS](#)。

如需 Recipient 參數和 AWS CiphertextForRecipient 回應欄位的詳細資訊，請參閱 AWS Key Management Service API 參考中的

[Decrypt](#)、[DeriveSharedSecret](#)、[GenerateDataKey](#)、[GenerateDataKeyPair](#) 和 [GenerateRandom](#) 主題、[AWS Nitro Enclaves SDK](#) 或任何 AWS SDK。如需設定資料和資料金鑰以進行加密的詳細資訊，請參閱[搭配使用密碼編譯認證 AWS KMS](#)。

監控已驗證的請求

您可以使用 AWS CloudTrail 日誌來監控

[Decrypt](#)、[DeriveSharedSecret](#)、[GenerateDataKey](#)、[GenerateDataKeyPair](#) 和 [GenerateRandom](#) 使用證明的操作。在這些日誌項目中，additionalEventData 欄位有一個recipient欄位，其中包含來自請求中證明文件的資訊。只有當請求中的 Recipient 參數指定簽署的證明文件時，才會包含這些欄位。

CloudTrail 日誌中包含的特定資訊取決於使用的認證方法。

對 Nitro Enclaves 的監空請求

對於 Nitro enclave 認證，CloudTrail 日誌包含認證文件中的模組 ID (attestationDocumentModuleId)、映像摘要 (attestationDocumentEnclaveImageDigest) 和平台組態註冊 (PCRs)。

模組 ID 是 Nitro enclave 的 [enclave ID](#)。影像摘要是 enclave 影像的 SHA384 雜湊。您可在[金鑰政策與 IAM 政策的條件](#)運用影像摘要及 PCR 值。如需 PCR 的相關資訊，請參閱《AWS Nitro Enclaves 使用者指南》的[Where to get an enclave's measurements](#)。

本節針對每個支援的 Nitro enclave AWS KMS請求顯示 CloudTrail 日誌項目範例。

Decrypt (針對 enclave)

下列範例顯示 AWS Nitro enclave [解密](#)操作的 AWS CloudTrail 日誌項目。

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2020-07-27T22:58:24Z",
```

```

    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
      "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
      "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": null,
    "additionalEventData": {
      "recipient": {
        "attestationDocumentModuleId": "i-123456789abcde123-enc123456789abcde12",
        "attestationDocumentEnclaveImageDigest": "<AttestationDocument.PCR0>",
        "attestationDocumentEnclavePCR1": "<AttestationDocument.PCR1>",
        "attestationDocumentEnclavePCR2": "<AttestationDocument.PCR2>",
        "attestationDocumentEnclavePCR3": "<AttestationDocument.PCR3>",
        "attestationDocumentEnclavePCR4": "<AttestationDocument.PCR4>",
        "attestationDocumentEnclavePCR8": "<AttestationDocument.PCR8>"
      }
    },
    "requestID": "b4a65126-30d5-4b28-98b9-9153da559963",
    "eventID": "e5a2f202-ba1a-467c-b4ba-f729d45ae521",
    "readOnly": true,
    "resources": [
      {
        "accountId": "111122223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
  }
}

```

GenerateDataKey (針對 enclave)

下列範例顯示 AWS Nitro enclave 的 [GenerateDataKey](#) 操作的 AWS CloudTrail 日誌項目。

```

{
  "eventVersion": "1.02",

```

```

"userIdentity": {
  "type": "IAMUser",
  "principalId": "EX_PRINCIPAL_ID",
  "arn": "arn:aws:iam::111122223333:user/Alice",
  "accountId": "111122223333",
  "accessKeyId": "EXAMPLE_KEY_ID",
  "userName": "Alice"
},
"eventTime": "2014-11-04T00:52:40Z",
"eventSource": "kms.amazonaws.com",
"eventName": "GenerateDataKey",
"awsRegion": "us-east-1",
"sourceIPAddress": "192.0.2.0",
"userAgent": "AWS Internal",
"requestParameters": {
  "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
  "numberOfBytes": 32
},
"responseElements": null,
"additionalEventData": {
  "recipient": {
    "attestationDocumentModuleId": "i-123456789abcde123-enc123456789abcde12",
    "attestationDocumentEnclaveImageDigest": "<AttestationDocument.PCR0>",
    "attestationDocumentEnclavePCR1": "<AttestationDocument.PCR1>",
    "attestationDocumentEnclavePCR2": "<AttestationDocument.PCR2>",
    "attestationDocumentEnclavePCR3": "<AttestationDocument.PCR3>",
    "attestationDocumentEnclavePCR4": "<AttestationDocument.PCR4>",
    "attestationDocumentEnclavePCR8": "<AttestationDocument.PCR8>"
  }
},
"requestID": "e0eb83e3-63bc-11e4-bc2b-4198b6150d5c",
"eventID": "a9dea4f9-8395-46c0-942c-f509c02c2b71",
"readOnly": true,
"resources": [{
  "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
  "accountId": "111122223333"
}],
"eventType": "AwsApiCall",
"recipientAccountId": "111122223333"
}

```

GenerateDataKeyPair (針對 enclave)

下列範例顯示 AWS Nitro enclave 的 [GenerateDataKeyPair](#) 操作的 AWS CloudTrail 日誌項目。

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2020-07-27T18:57:57Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GenerateDataKeyPair",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyPairSpec": "RSA_3072",
    "encryptionContext": {
      "Project": "Alpha"
    },
    "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
  },
  "responseElements": null,
  "additionalEventData": {
    "recipient": {
      "attestationDocumentModuleId": "i-123456789abcde123-enc123456789abcde12",
      "attestationDocumentEnclaveImageDigest": "<AttestationDocument.PCR0>",
      "attestationDocumentEnclavePCR1": "<AttestationDocument.PCR1>",
      "attestationDocumentEnclavePCR2": "<AttestationDocument.PCR2>",
      "attestationDocumentEnclavePCR3": "<AttestationDocument.PCR3>",
      "attestationDocumentEnclavePCR4": "<AttestationDocument.PCR4>",
      "attestationDocumentEnclavePCR8": "<AttestationDocument.PCR8>"
    }
  },
  "requestID": "52fb127b-0fe5-42bb-8e5e-f560febde6b0",
  "eventID": "9b6bd6d2-529d-4890-a949-593b13800ad7",
  "readOnly": true,
  "resources": [
    {
```



```

        "accountId": "111122223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
],
"eventType": "AwsApiCall",
"recipientAccountId": "111122223333"
}

```

GenerateRandom (針對 enclave)

下列範例顯示 AWS Nitro enclave 的 [GenerateRandom](#) 操作的 AWS CloudTrail 日誌項目。

```

{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2014-11-04T00:52:37Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GenerateRandom",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": null,
  "responseElements": null,
  "additionalEventData": {
    "recipient": {
      "attestationDocumentModuleId": "i-123456789abcde123-enc123456789abcde12",
      "attestationDocumentEnclaveImageDigest": "<AttestationDocument.PCR0>",
      "attestationDocumentEnclavePCR1": "<AttestationDocument.PCR1>",
      "attestationDocumentEnclavePCR2": "<AttestationDocument.PCR2>",
      "attestationDocumentEnclavePCR3": "<AttestationDocument.PCR3>",
      "attestationDocumentEnclavePCR4": "<AttestationDocument.PCR4>",
      "attestationDocumentEnclavePCR8": "<AttestationDocument.PCR8>"
    }
  },
}

```

```
{
  "requestID": "df1e3de6-63bc-11e4-bc2b-4198b6150d5c",
  "eventID": "239cb9f7-ae05-4c94-9221-6ea30eef0442",
  "readOnly": true,
  "resources": [],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

監控 NitroTPM 的請求

對於 NitroTPM 認證，CloudTrail 日誌包含認證文件中的模組 ID (attestationDocumentModuleId) 和平台組態註冊 (PCRs)。

模組 ID 是具有 TPM 識別符的 NitroTPM 的 EC2 執行個體 ID。您可以在[金鑰政策和 IAM 政策的條件中使用 PCR](#) 值。

本節顯示每個支援的 NitroTPM 請求的範例 CloudTrail 日誌項目 AWS KMS。

Decrypt (適用於 NitroTPM)

下列範例顯示 NitroTPM [解密](#)操作的 AWS CloudTrail 日誌項目。

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2020-07-27T22:58:24Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "Decrypt",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
}
```

```

"responseElements": null,
"additionalEventData": {
  "recipient": {
    "attestationDocumentModuleId": "i-123456789abcde123-tpm000000000000000000",
    "attestationDocumentNitroTPMPCR4": "<AttestationDocument.PCR4>",
    "attestationDocumentNitroTPMPCR7": "<AttestationDocument.PCR7>",
    "attestationDocumentNitroTPMPCR8": "<AttestationDocument.PCR8>",
    "attestationDocumentNitroTPMPCR9": "<AttestationDocument.PCR9>",
    "attestationDocumentNitroTPMPCR16": "<AttestationDocument.PCR16>",
    "attestationDocumentNitroTPMPCR23": "<AttestationDocument.PCR23>"
  }
},
"requestID": "b4a65126-30d5-4b28-98b9-9153da559963",
"eventID": "e5a2f202-ba1a-467c-b4ba-f729d45ae521",
"readOnly": true,
"resources": [
  {
    "accountId": "111122223333",
    "type": "AWS::KMS::Key",
    "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  }
],
"eventType": "AwsApiCall",
"recipientAccountId": "111122223333"
}

```

GenerateDataKey (適用於 NitroTPM)

下列範例顯示 NitroTPM 的 [GenerateDataKey](#) 操作的 AWS CloudTrail 日誌項目。

```

{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2014-11-04T00:52:40Z",
  "eventSource": "kms.amazonaws.com",

```

```

    "eventName": "GenerateDataKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
      "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
      "numberOfBytes": 32
    },
    "responseElements": null,
    "additionalEventData": {
      "recipient": {
        "attestationDocumentModuleId": "i-123456789abcde123-tpm0000000000000000",
        "attestationDocumentNitroTPMPCR4": "<AttestationDocument.PCR4>",
        "attestationDocumentNitroTPMPCR7": "<AttestationDocument.PCR7>",
        "attestationDocumentNitroTPMPCR8": "<AttestationDocument.PCR8>",
        "attestationDocumentNitroTPMPCR9": "<AttestationDocument.PCR9>",
        "attestationDocumentNitroTPMPCR16": "<AttestationDocument.PCR16>",
        "attestationDocumentNitroTPMPCR23": "<AttestationDocument.PCR23>"
      }
    },
    "requestID": "e0eb83e3-63bc-11e4-bc2b-4198b6150d5c",
    "eventID": "a9dea4f9-8395-46c0-942c-f509c02c2b71",
    "readOnly": true,
    "resources": [{
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "accountId": "111122223333"
    }],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
  }
}

```

GenerateDataKeyPair (適用於 NitroTPM)

下列範例顯示 NitroTPM 的 [GenerateDataKeyPair](#) 操作的 AWS CloudTrail 日誌項目。

```

{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",

```

```

        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
    },
    "eventTime": "2020-07-27T18:57:57Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyPair",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "AWS Internal",
    "requestParameters": {
        "keyPairSpec": "RSA_3072",
        "encryptionContext": {
            "Project": "Alpha"
        },
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "responseElements": null,
    "additionalEventData": {
        "recipient": {
            "attestationDocumentModuleId": "i-123456789abcde123-tpm000000000000000000",
            "attestationDocumentNitroTPMPCR4": "<AttestationDocument.PCR4>",
            "attestationDocumentNitroTPMPCR7": "<AttestationDocument.PCR7>",
            "attestationDocumentNitroTPMPCR8": "<AttestationDocument.PCR8>",
            "attestationDocumentNitroTPMPCR9": "<AttestationDocument.PCR9>",
            "attestationDocumentNitroTPMPCR16": "<AttestationDocument.PCR16>",
            "attestationDocumentNitroTPMPCR23": "<AttestationDocument.PCR23>"
        }
    },
    "requestID": "52fb127b-0fe5-42bb-8e5e-f560febde6b0",
    "eventID": "9b6bd6d2-529d-4890-a949-593b13800ad7",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}

```

GenerateRandom (適用於 NitroTPM)

下列範例顯示 NitroTPM 的 [GenerateRandom](#) 操作的 AWS CloudTrail 日誌項目。

```
{
  "eventVersion": "1.02",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::111122223333:user/Alice",
    "accountId": "111122223333",
    "accessKeyId": "EXAMPLE_KEY_ID",
    "userName": "Alice"
  },
  "eventTime": "2014-11-04T00:52:37Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "GenerateRandom",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "AWS Internal",
  "requestParameters": null,
  "responseElements": null,
  "additionalEventData": {
    "recipient": {
      "attestationDocumentModuleId": "i-123456789abcde123-tpm0000000000000000",
      "attestationDocumentNitroTPMPCR4": "<AttestationDocument.PCR4>",
      "attestationDocumentNitroTPMPCR7": "<AttestationDocument.PCR7>",
      "attestationDocumentNitroTPMPCR8": "<AttestationDocument.PCR8>",
      "attestationDocumentNitroTPMPCR9": "<AttestationDocument.PCR9>",
      "attestationDocumentNitroTPMPCR16": "<AttestationDocument.PCR16>",
      "attestationDocumentNitroTPMPCR23": "<AttestationDocument.PCR23>"
    }
  },
  "requestID": "df1e3de6-63bc-11e4-bc2b-4198b6150d5c",
  "eventID": "239cb9f7-ae05-4c94-9221-6ea30eef0442",
  "readOnly": true,
  "resources": [],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

搭配 AWS 服務使用 AWS KMS 加密

使用 AWS Key Management Service，您可以提供加密金鑰來保護其他服務中的資料 AWS。將 AWS KMS 加密與 AWS 服務搭配使用是指 AWS KMS 與其他 AWS 服務整合的程序，以加密和解密靜態或傳輸中的資料。開發人員、系統管理員和安全專業人員可能對本主題感興趣，以保護透過 AWS 服務存放或傳輸的敏感資料、符合法規合規要求，或實作加密最佳實務。常見的使用案例包括加密 Amazon EBS 磁碟區、Amazon S3 儲存貯體和 Amazon RDS 資料庫。以下各節將涵蓋設定和管理特定 AWS 服務的 AWS KMS 加密金鑰的步驟，以確保整個 AWS 環境的資料機密性和完整性。如需整合 AWS 的服務完整清單 AWS KMS，請參閱 [AWS 服務整合](#)。

下列主題詳細討論特定服務如何使用 AWS KMS，包括他們支援的 KMS 金鑰、他們如何管理資料金鑰、他們需要的許可，以及如何追蹤每個服務對您帳戶中 KMS 金鑰的使用。

Important

[AWS 與 整合的服務 AWS KMS](#) 僅使用對稱加密 KMS 金鑰來加密您的資料。這些服務不支援使用非對稱 KMS 金鑰進行加密。如需有關如何判斷 KMS 金鑰是對稱還是不對稱的說明，請參閱 [識別不同的金鑰類型](#)。

主題

- [Amazon Elastic Block Store \(Amazon EBS\) 如何使用 AWS KMS](#)
- [Amazon EMR 如何使用 AWS KMS](#)
- [Amazon Redshift 如何使用 AWS KMS](#)

Amazon Elastic Block Store (Amazon EBS) 如何使用 AWS KMS

本主題詳細討論 [Amazon Elastic Block Store \(Amazon EBS\)](#) 如何使用 AWS KMS 來加密磁碟區和快照。對於加密 Amazon EBS 磁碟區的基本說明，請參閱 [Amazon EBS 加密](#)。

主題

- [Amazon EBS 加密](#)
- [使用 KMS 金鑰和資料金鑰](#)
- [Amazon EBS 加密內容](#)

- [偵測 Amazon EBS 失敗](#)
- [使用 AWS CloudFormation 建立加密的 Amazon EBS 磁碟區](#)

Amazon EBS 加密

當您連接加密 Amazon EBS 磁碟區到[支援的 Amazon Elastic Compute Cloud \(Amazon EC2\) 執行個體類型](#)，存放在磁碟區的靜態資料、磁碟輸入/輸出，以及從磁碟區建立的快照全部都會加密。加密在託管 Amazon EC2 執行個體的伺服器上進行。

所有 [Amazon EBS 磁碟區類型](#) 都支援此功能。您存取加密磁碟區方式與存取其他磁碟區的方式相同；加密和解密的處理過程皆相當透明，不需要您、您的 EC2 執行個體或您的應用程式進行任何額外動作。加密磁碟區的快照會自動加密，而從加密快照建立的磁碟區也會自動加密。

EBS 磁碟區的加密狀態取決於您建立磁碟區時。您不能改變現有磁碟區的加密狀態。不過，您可以在加密和未加密的磁碟區之間[遷移資料](#)，並在複製快照時套用新的加密狀態。

Amazon EBS 預設支援可選的加密。您可以在 AWS 帳戶和區域中的所有新 EBS 磁碟區和快照複本上自動啟用加密。此組態設定不會影響現有的磁碟區或快照。如需詳細資訊，請參閱《[Amazon EBS 使用者指南](#)》中的 [Amazon EBS 加密](#)。

使用 KMS 金鑰和資料金鑰

[建立加密的 Amazon EBS 磁碟區](#)時，您可以指定 AWS KMS key。根據預設，Amazon EBS 會使用您帳戶 (aws/efs) 中的 Amazon EBS [AWS 受管金鑰](#)。但是，您可以指定您建立和管理的[客戶受管金鑰](#)。

若要使用客戶受管金鑰，您必須授予 Amazon EBS 許可，才能代表您使用 KMS 金鑰。如需詳細資訊，請參閱《[Amazon EBS 使用者指南](#)》中的 [Amazon EBS 加密如何運作](#)。

Important

Amazon EBS 只支援[對稱 KMS 金鑰](#)。您無法使用[非對稱 KMS 金鑰](#)來加密 Amazon EBS 磁碟區。如需判斷 KMS 金鑰是對稱還是不對稱的說明，請參閱 [識別不同的金鑰類型](#)。

對於每個磁碟區，Amazon EBS AWS KMS 會要求產生以您指定的 KMS 金鑰加密的唯一資料金鑰。Amazon EBS 會隨著磁碟區存放加密的資料金鑰。然後，當您將磁碟區連接到 Amazon EC2 執行個體時，Amazon EBS 會呼叫 AWS KMS 來解密資料金鑰。Amazon EBS 使用存放在 Hypervisor 記

憶體的純文字資料金鑰來加密所有磁碟區的磁碟輸入/輸出。如需詳細資訊，請參閱《[Amazon EC2 使用者指南](#)》或《[Amazon EC2 使用者指南](#)》中的 EBS 加密如何運作。

Amazon EBS 加密內容

在其 [GenerateDataKeyWithoutPlaintext](#) 和 [Decrypt](#) 請求中 AWS KMS，Amazon EBS 會使用加密內容搭配識別請求中磁碟區或快照的名稱值對。加密內容中的名稱會維持不變。

[加密內容](#)是一組金鑰/值對，其中包含任意非私密資料。當您在加密資料的請求中包含加密內容時，會以 AWS KMS 加密方式將加密內容繫結至加密的資料。若要解密資料，您必須傳遞相同的加密內容。

針對使用 Amazon EBS [CreateSnapshot](#) 操作建立的所有磁碟區和加密的快照，Amazon EBS 會使用磁碟區 ID 作為加密內容值。在 CloudTrail 日誌項目的 requestParameters 欄位中，加密內容看起來如下：

```
"encryptionContext": {
  "aws:eks:id": "vol-0cfb133e847d28be9"
}
```

針對使用 Amazon EC2 [CopySnapshot](#) 操作建立的加密快照，Amazon EBS 會使用快照 ID 作為加密內容值。在 CloudTrail 日誌項目的 requestParameters 欄位中，加密內容看起來如下：

```
"encryptionContext": {
  "aws:eks:id": "snap-069a655b568de654f"
}
```

偵測 Amazon EBS 失敗

為了建立加密的 EBS 磁碟區或將磁碟區連接到 EC2 執行個體，Amazon EBS 和 Amazon EC2 基礎設施必須能夠使用您為 EBS 磁碟區加密指定的 KMS 金鑰。當 KMS 金鑰無法使用，例如，當其[金鑰狀態](#)不是 Enabled 時，磁碟區建立或磁碟區連接會失敗。

在這種情況下，Amazon EBS 會傳送事件給 Amazon EventBridge (原稱為 CloudWatch Events)，通知您作業失敗。您可以使用 EventBridge 建立規則，觸發可自動執行的動作來回應這些事件。如需詳細資訊，請參閱《[Amazon EBS 使用者指南](#)》中的 [Amazon CloudWatch Events for Amazon EBS](#)，特別是下列章節：

- [磁碟區連接或重新連接時的無效加密金鑰](#)
- [建立磁碟區時的無效加密金鑰](#)

若要修正這些問題，請確保您為 EBS 磁碟區加密指定的 KMS 金鑰已啟用。若要執行此作業，請先[檢視 KMS 金鑰](#)以判斷其目前的金鑰狀態 (中的狀態欄 AWS 管理主控台)。接著，查看下列其中一個連結的相關資訊：

- 如果 KMS 金鑰的金鑰狀態為已停用，則請[啟用](#)。
- 如果 KMS 金鑰的金鑰狀態為待匯入，則請[匯入金鑰材料](#)。
- 如果 KMS 金鑰的金鑰狀態為待刪除，則請[取消金鑰刪除](#)。

使用 AWS CloudFormation 建立加密的 Amazon EBS 磁碟區

您可以使用 [AWS CloudFormation](#) 來建立加密的 Amazon EBS 磁碟區。如需詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的 [AWS::EC2::Volume](#)。

Amazon EMR 如何使用 AWS KMS

當使用 [Amazon EMR](#) 叢集時，您可以在將資料儲存至持久性儲存位置之前，設定叢集加密靜態資料。您可以加密 EMR 檔案系統 (EMRFS)、叢集節點之儲存磁碟區，或兩者上的靜態資料。若要靜態加密資料，您可以使用 AWS KMS key。下列主題說明 Amazon EMR 叢集如何使用 KMS 金鑰來加密靜態資料。

Important

Amazon EMR 僅支援[對稱 KMS 金鑰](#)。您無法使用[非對稱 KMS 金鑰](#)來加密 Amazon EMR 叢集中的靜態資料。如需有關如何判斷 KMS 金鑰是對稱還是不對稱的說明，請參閱[識別不同的金鑰類型](#)。

Amazon EMR 叢集也會加密傳輸中的資料，這表示叢集會在透過網路傳送資料之前進行加密。您不能使用 KMS 金鑰加密傳輸中的資料。如需詳細資訊，請參閱《Amazon EMR 管理指南》中的[傳輸資料加密](#)。

如需 Amazon EMR 中所有可用加密選項的詳細資訊，請參閱《Amazon EMR 管理指南》中的[加密選項](#)。

主題

- [加密 EMR 檔案系統 \(EMRFS\) 上的資料](#)
- [加密叢集節點之儲存磁碟區上的資料](#)

- [加密內容](#)

加密 EMR 檔案系統 (EMRFS) 上的資料

Amazon EMR 叢集使用兩個分散式檔案系統：

- Hadoop 分散式檔案系統 (HDFS)。HDFS 加密不使用 AWS KMS 中的 KMS 金鑰。
- EMR 檔案系統 (EMRFS)。EMRFS 是 HDFS 實作，可讓 Amazon EMR 叢集將資料存放在 Amazon Simple Storage Service (Amazon S3) 中。EMRFS 支援四種加密選項，其中兩種使用 AWS KMS 中的 KMS 金鑰。如需所有四個 EMRFS 加密選項的詳細資訊，請參閱《Amazon EMR 管理指南》中的[加密選項](#)。

使用 KMS 金鑰的兩個 EMRFS 加密選項使用 Amazon S3 提供的以下加密功能：

- [使用伺服器端加密搭配 AWS Key Management Service \(SSE-KMS\) 保護資料](#)。Amazon EMR 叢集會將資料傳送至 Simple Storage Service (Amazon S3)。Simple Storage Service (Amazon S3) 使用 KMS 金鑰來加密資料，然後將其儲存至 S3 儲存貯體。如需此操作如何進行的詳細資訊，請參閱[使用 SSE-KMS 加密 EMRFS 上資料的程序](#)。
- [使用用戶端加密保護資料 \(CSE-KMS\)](#)。Amazon EMR 中的資料在 AWS KMS key 下加密，然後將其傳送至 Simple Storage Service (Amazon S3) 進行儲存。如需此操作如何進行的詳細資訊，請參閱[使用 CSE-KMS 加密 EMRFS 上資料的程序](#)。

當您設定 Amazon EMR 叢集在 EMRFS 上使用 KMS 金鑰來加密資料時，您可以選擇您想要 Simple Storage Service (Amazon S3) 或 Amazon EMR 叢集使用的 KMS 金鑰。使用 SSE-KMS，您可以選擇具有別名 aws/s3 的 Amazon S3 AWS 受管金鑰，或您建立的對稱客戶受管金鑰。使用用戶端加密時，您必須選擇您建立的對稱客戶受管金鑰。選擇客戶受管金鑰時，您必須確保 Amazon EMR 叢集具有使用 KMS 金鑰的許可。如需詳細資訊，請參閱《Amazon EMR 管理指南》中的[使用 AWS KMS keys 進行加密](#)。

對於伺服器端和用戶端加密，您選擇的 KMS 金鑰是[信封加密](#)工作流程中的根金鑰。資料會使用在 KMS 金鑰下加密的唯一[資料](#)金鑰進行加密 AWS KMS。加密的資料及其資料金鑰的加密副本會以單一加密物件一起存放在 S3 儲存貯體中。如需其如何運作的詳細資訊，請參閱下列主題。

主題

- [使用 SSE-KMS 加密 EMRFS 上資料的程序](#)
- [使用 CSE-KMS 加密 EMRFS 上資料的程序](#)

使用 SSE-KMS 加密 EMRFS 上資料的程序

設定 Amazon EMR 叢集使用 SSE-KMS 時，加密程序的運作方式如下：

1. 叢集將資料傳送至 Amazon S3 以便儲存在 S3 儲存貯體中。
2. Amazon S3 將 [GenerateDataKey](#) 請求傳送至 AWS KMS，指定您在將叢集設定為使用 SSE-KMS 時所選 KMS 金鑰的金鑰 ID。請求包含加密內容；如需詳細資訊，請參閱[加密內容](#)。
3. AWS KMS 會產生唯一的資料加密金鑰（資料金鑰），然後將此資料金鑰的兩個副本傳送至 Amazon S3。一個複本是未加密的（純文字），另一個複本則透過 KMS 金鑰加密。
4. Amazon S3 使用純文字資料金鑰來加密它在步驟 1 收到的資料，然後在使用後盡快從記憶體移除純文字資料金鑰。
5. Amazon S3 將加密的資料及資料金鑰的加密複本以單一加密物件一起存放在 S3 儲存貯體中。

解密程序的運作方式如下：

1. 叢集向 S3 儲存貯體請求加密的資料物件。
2. Amazon S3 從 S3 物件擷取加密的資料金鑰，然後使用 AWS KMS [解密](#) 請求將加密的資料金鑰傳送至。此請求包含[加密內容](#)。
3. AWS KMS 使用用來加密的相同 KMS 金鑰來解密加密的資料金鑰，然後將解密的（純文字）資料金鑰傳送至 Amazon S3。
4. Amazon S3 使用純文字資料金鑰來解密加密的資料，然後在使用後盡快從記憶體移除純文字資料金鑰。
5. Amazon S3 將解密的資料傳送給叢集。

使用 CSE-KMS 加密 EMRFS 上資料的程序

設定 Amazon EMR 叢集使用 CSE-KMS 時，加密程序的運作方式如下：

1. 準備好將資料存放在 Amazon S3 時，叢集會傳送 [GenerateDataKey](#) 請求至 AWS KMS，指定您在將叢集設定為使用 CSE-KMS 時所選 KMS 金鑰的金鑰 ID。請求包含加密內容；如需詳細資訊，請參閱[加密內容](#)。
2. AWS KMS 會產生唯一的資料加密金鑰（資料金鑰），然後將此資料金鑰的兩個副本傳送至叢集。一個複本是未加密的（純文字），另一個複本則透過 KMS 金鑰加密。
3. 叢集使用純文字資料金鑰來加密資料，然後在使用後盡快從記憶體移除純文字資料金鑰。
4. 叢集將加密的資料及資料金鑰的加密副本合併為單一加密物件。

5. 叢集將加密的物件傳送到 Amazon S3 以便儲存。

解密程序的運作方式如下：

1. 叢集向 S3 儲存貯體請求加密的資料物件。
2. Amazon S3 傳送加密的物件給叢集。
3. 叢集會從加密的物件擷取加密的資料金鑰，然後使用 AWS KMS [解密](#) 請求將加密的資料金鑰傳送至。此請求包含[加密內容](#)。
4. AWS KMS 使用用來加密的相同 KMS 金鑰來解密加密的資料金鑰，然後將解密的（純文字）資料金鑰傳送至叢集。
5. 叢集使用純文字資料金鑰來解密加密的資料，然後在使用後盡快從記憶體移除純文字資料金鑰。

加密叢集節點之儲存磁碟區上的資料

Amazon EMR 叢集是 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的集合。叢集中的每個執行個體稱為叢集節點或節點。每個節點可以有兩種類型的儲存磁碟區：執行個體存放磁碟區和 Amazon Elastic Block Store (Amazon EBS) 磁碟區。您可以設定叢集為使用 [Linux 統一金鑰設定 \(LUKS\)](#) 來加密節點上的這兩種儲存磁碟區類型 (但不包含每個節點的開機磁碟區)。這稱為本機磁碟加密。

當您啟用叢集的本機磁碟加密時，可以選擇使用 AWS KMS 中的 KMS 金鑰來加密 LUKS 金鑰。您必須選擇您建立的[客戶受管金鑰](#)，您無法使用 [AWS 受管金鑰](#)。如果選擇客戶受管金鑰，則您必須確保 Amazon EMR 叢集具有使用 KMS 金鑰的許可。如需詳細資訊，請參閱《Amazon EMR 管理指南》中的[使用 AWS KMS keys 進行加密](#)。

當您啟用使用 KMS 金鑰的本機磁碟加密，加密程序的運作方式如下：

1. 當每個叢集節點啟動時，它會傳送 [GenerateDataKey](#) 請求給 AWS KMS，指定您在為叢集啟用本機磁碟加密時選擇的 KMS 金鑰的金鑰 ID。
2. AWS KMS 會產生唯一的資料加密金鑰（資料金鑰），然後將此資料金鑰的兩個副本傳送至節點。一個複本是未加密的（純文字），另一個複本則透過 KMS 金鑰加密。
3. 節點使用純文字資料金鑰的 base64 編碼版本做為保護 LUKS 金鑰的密碼。節點將資料金鑰的加密副本儲存在開機磁碟區。
4. 如果節點重新啟動，重新啟動的節點會使用 AWS KMS [解密](#) 請求將加密的資料金鑰傳送至。
5. AWS KMS 使用用來加密的相同 KMS 金鑰來解密加密的資料金鑰，然後將解密的（純文字）資料金鑰傳送至節點。

6. 節點使用純文字資料金鑰的 base64 編碼版本做為解鎖 LUKS 金鑰的密碼。

加密內容

與整合的每個 AWS 服務 AWS KMS 都可以在服務 AWS KMS 用來產生資料金鑰或加密或解密資料時指定加密內容。加密內容是額外的驗證資訊，AWS KMS 用於檢查資料完整性。當服務指定加密操作的加密內容，它必須為相對應的解密操作指定相同的加密內容，否則解密將失敗。加密內容也會寫入 AWS CloudTrail 日誌檔案，這可協助您了解使用特定 KMS 金鑰的原因。

以下章節說明使用 KMS 金鑰之每個 Amazon EMR 加密案例所用的加密內容。

使用 SSE-KMS 進行 EMRFS 加密的加密內容

透過 SSE-KMS，Amazon EMR 叢集會傳送資料至 Amazon S3，接著 Amazon S3 使用 KMS 金鑰加密資料，然後將其儲存到 S3 儲存貯體。在此情況下，Amazon S3 會使用 S3 物件的 Amazon Resource Name (ARN) 作為加密內容，其中包含其傳送的每個 [GenerateDataKey](#) 和 [Decrypt](#) 請求 AWS KMS。以下範例顯示 Amazon S3 所用加密內容的 JSON 顯示方式。

```
{ "aws:s3:arn" : "arn:aws:s3:::S3_bucket_name/S3_object_key" }
```

使用 CSE-KMS 進行 EMRFS 加密的加密內容

透過 CSE-KMS，Amazon EMR 叢集使用 KMS 金鑰加密資料，然後將其傳送到 Amazon S3 儲存貯體儲存。在此情況下，叢集會使用 KMS 金鑰的 Amazon Resource Name (ARN) 做為加密內容，其中包含其傳送的每個 [GenerateDataKey](#) 和 [Decrypt](#) 請求 AWS KMS。以下範例顯示叢集所用加密內容的 JSON 顯示方式。

```
{ "kms_cmek_id" : "arn:aws:kms:us-east-2:111122223333:key/0987ab65-43cd-21ef-09ab-87654321cdef" }
```

使用 LUKS 進行本機磁碟加密的加密內容

當 Amazon EMR 叢集使用搭配 LUKS 的本機磁碟加密時，叢集節點不會指定加密內容，其中包含其傳送至 AWS KMS 的 [GenerateDataKey](#) 和 [Decrypt](#) 請求。

Amazon Redshift 如何使用 AWS KMS

本主題討論 Amazon Redshift 如何使用 AWS KMS 來加密資料。

主題

- [Amazon Redshift 加密](#)
- [加密內容](#)

Amazon Redshift 加密

Amazon Redshift 資料倉儲是稱為節點的運算資源的集合，組織成稱為叢集的群組。每個叢集皆執行 Amazon Redshift 引擎並包含一或多個資料庫。

Amazon Redshift 使用四個階層的金鑰架構來加密。此架構包含資料加密金鑰、資料庫金鑰、叢集金鑰和根金鑰。您可以使用 AWS KMS key 做為根金鑰。

資料加密金鑰會加密叢集中的資料區塊。每個資料區塊都會獲指派一個隨機產生的 AES-256 金鑰。這些金鑰使用叢集的資料庫金鑰來加密。

資料庫金鑰會加密叢集中的資料加密金鑰。資料庫金鑰是隨機產生的 AES-256 金鑰。它會存放在與 Amazon Redshift 叢集不同之網路的磁碟上，並透過安全通道傳送給叢集。

叢集金鑰會加密 Amazon Redshift 叢集的資料庫金鑰。您可以使用 AWS KMS AWS CloudHSM、或外部硬體安全模組 (HSM) 來管理叢集金鑰。請參閱 [Amazon Redshift Database 加密](#) 文件以取得更多詳細資訊。

您可以在 Amazon Redshift 主控台中勾選適當的方塊來請求加密。您可以從加密方塊下方的清單中選擇一個項目，指定要[客戶受管金鑰](#)。如果您未指定客戶受管金鑰，Amazon Redshift 會在您的帳戶下使用 Amazon Redshift 的 [AWS 受管金鑰](#)。

Important

Amazon Redshift 只支援對稱加密 KMS 金鑰。您無法在 Amazon Redshift 加密工作流程中使用非對稱 KMS 金鑰。如需判斷 KMS 金鑰是對稱還是不對稱的說明，請參閱 [識別不同的金鑰類型](#)。

加密內容

與整合的每個服務會在請求資料金鑰、加密和解密時 AWS KMS 指定[加密內容](#)。加密內容是額外的驗證資料 (AAD)，AWS KMS 用於檢查資料完整性。也就是說，為加密操作指定加密內容時，服務也必須為解密操作指定相同的加密內容，否則解密將無法成功。Amazon Redshift 會使用叢集 ID 和加密內容的建立時間。在 CloudTrail 日誌檔案的 requestParameters 欄位中，加密內容看起來如下。

```
"encryptionContext": {  
  "aws:redshift:arn": "arn:aws:redshift:region:account_ID:cluster:cluster_name",  
  "aws:redshift:createtime": "20150206T1832Z"  
},
```

您可以在 CloudTrail 日誌中搜尋叢集名稱，以了解使用 AWS KMS key (KMS 金鑰) 執行了哪些操作。操作包括叢集加密、叢集解密和產生資料金鑰。

AWS KMS 參考

下列參考資料提供有關使用和管理 KMS 金鑰的實用資訊。

- [金鑰類型參考](#)。列出支援每個 AWS KMS API 操作的 KMS 金鑰類型。

若要尋找：我可以啟用和停用 RSA 簽署 KMS 金鑰嗎？

- [金鑰狀態資料表](#)。顯示 KMS 金鑰的金鑰狀態會如何影響其在 AWS KMS API 操作的使用。

若要尋找：我可以變更等待刪除的 KMS 金鑰的別名嗎？

- [AWS KMS API 許可參考](#)。提供每個 AWS KMS API 操作所需許可的相關資訊。

若要尋找：我可以在不同 AWS 帳戶中的金鑰上執行 [GetKeyPolicy](#) 嗎？我可以允許 IAM 政策中的 `kms:Decrypt` 許可嗎？

- [ViaService 參考](#)。支援 `kms:ViaService` 條件金鑰的 AWS 服務清單。

若要尋找：我可以使用 `kms:ViaService` 條件金鑰只在當許可來自 Amazon ElastiCache 時允許它嗎？Amazon Neptune 呢？

- [AWS KMS 定價](#)。列出並說明 KMS 金鑰的價格。

若要尋找：使用我的非對稱金鑰的費用多少？

- [AWS KMS 請求配額](#)。列出每個帳戶和區域中 AWS KMS API 請求的每秒配額。

若要尋找：我每秒可以執行多少 [Decrypt](#) 請求？我可以在自訂金鑰存放區的 KMS 金鑰上執行多少 [Decrypt](#) 請求？

- [AWS KMS 資源配額](#)。列出 AWS KMS 資源的配額。

若要尋找：我的帳戶每個區域可以有多少 KMS 金鑰？每個 KMS 金鑰可以有多少個別名？

- [AWS 與 整合的 服務 AWS KMS](#)。列出使用 KMS 金鑰來保護其建立、存放和管理之資源 AWS 的服務。

若要尋找：Amazon Connect 是否使用 KMS 金鑰來保護我的 Connect 資源？

金鑰的 AWS KMS 金鑰狀態

AWS KMS key 一律具有金鑰狀態。KMS 金鑰及其環境的操作可以變更金鑰狀態。金鑰狀態可以暫時變更，或直到另一個操作變更其金鑰狀態為止。這些操作會以非同步方式或 API 呼叫來完成。

本節中的表格顯示金鑰狀態如何影響對 AWS KMS API 操作的呼叫。其金鑰狀態的結果，KMS 金鑰上的操作預期會成功 (#)，失敗 (X)，或者只在某些條件下成功 (?)。結果對於含匯入金鑰材料的 KMS 金鑰來說，通常是不同的。

此資料表僅包含使用現有 KMS 金鑰的 API 操作。其他操作，例如 [CreateKey](#) 和 [ListKeys](#)，則會省略。

主題

- [金鑰狀態和 KMS 金鑰類型](#)
- [金鑰狀態資料表](#)

金鑰狀態和 KMS 金鑰類型

KMS 金鑰的類型決定了其可以具有的金鑰狀態。

- 所有 KMS 金鑰都可以處於 Enabled、Disabled 和 PendingDeletion 狀態。
- 大部分的 KMS 金鑰都是在 Enabled 狀態中建立。具有匯入金鑰材料的金鑰會在 PendingImport 狀態中建立。
- PendingImport 狀態僅適用於具有[匯入金鑰材料](#)的 KMS 金鑰。當匯入金鑰的任何金鑰材料遭到刪除或過期時，狀態會從變更為 Enabled PendingImport。
- Unavailable 狀態僅適用於[自訂金鑰存放區](#)中的 KMS 金鑰。[AWS CloudHSM 金鑰存放區](#)中的 KMS 金鑰是當自訂金鑰存放區刻意與其 AWS CloudHSM 叢集中斷連線Unavailable時。當自訂金鑰存放區刻意中斷與其[外部金鑰存放區代理](#)的連接時，[外部金鑰存放區](#)中的 KMS 金鑰為 Unavailable。您可以檢視和管理無法使用的 KMS 金鑰，但無法在密碼編譯操作中使用它們。

自訂金鑰存放區中 KMS 金鑰的金鑰狀態不會受其備份金鑰的變更影響。AWS CloudHSM 金鑰存放區中的 KMS 金鑰不會受到 AWS CloudHSM 叢集中其[相關聯金鑰材料](#)的變更影響。外部金鑰存放區中的 KMS 金鑰不會受外部金鑰管理器中[外部金鑰](#)的變更影響。如果停用或刪除備份金鑰，KMS 金鑰狀態不會變更，但使用 KMS 金鑰進行的密碼編譯操作會失敗。

- Creating、Updating 和 PendingReplicaDeletion 金鑰狀態僅適用於[多區域金鑰](#)。
 - 多區域複本金鑰在建立時處於暫時 Creating 金鑰狀態。[ReplicateKey](#) 操作完成時，此程序可能仍在進行中。複寫程序完成時，複本金鑰處於 Enabled 或 PendingImport 狀態。
 - 多區域金鑰在主要區域更新時處於暫時 Updating 金鑰狀態。[UpdatePrimaryRegion](#) 操作完成時，此程序可能仍在進行中。當更新程序完成時，主要和複本金鑰會繼續保持 Enabled 金鑰狀態。

- 當您排程刪除具有複本金鑰的多區域主要金鑰時，主要金鑰處於 PendingReplicaDeletion 狀態，直到刪除其所有複本金鑰為止。其金鑰狀態會變更為 PendingDeletion。如需詳細資訊，請參閱 [Deleting multi-Region keys](#)。

金鑰狀態資料表

下表顯示 KMS 金鑰的金鑰狀態如何影響 AWS KMS 操作。

編號註腳的描述 ([n]) 在本主題的結尾處。

Note

您可能需要水平或垂直捲動，才能查看此資料表中的所有資料。

API	已啟用	Disabled	待刪除 待刪除複本	待匯入	Unavailable	正在建立	更新中
CancelKey Deletion	 [4]	 [4]		 [4]	 [4], [13]	 [4]	 [4]
CreateAlias			 [3]				
CreateGrant		 [1]	 [2] 或 [3]	 [5]		 [14]	
解密		 [1]	 [2] 或 [3]	 [5]	 [11]	 [14]	

API	已啟用	Disabled	待刪除 待刪除複本	待匯入	Unavailable	正在建立	更新中
DeleteAlias	✓	✓	✓	✓	✓	✓	✓
DeleteImportedKeyMaterial	✓ [9]	✓ [9]	✓ [9]	✓	N/A	✗ [14]	✗ [15]
DeriveSharedSecret	✓	✗ [1]	✗ [2] 或 [3]	✗ [5]	N/A	✗ [14]	✓
DescribeKey	✓	✓	✓	✓	✓	✓	✓
DisableKey	✓	✓	✗ [3]	✗ [5]	✓ [12]	✗ [14]	✗ [15]
DisableKeyRotation	⚙ [7]	✗ [1] 或 [7]	✗ [3] 或 [7]	✗ [6]	✗ [7]	✗ [14]	⚙ [7]
EnableKey	✓	✓	✗ [3]	✗ [5]	✓ [12]	✗ [14]	✗ [15]
EnableKeyRotation	⚙ [7]	✗ [1] 或 [7]	✗ [3] 或 [7]	✗ [6]	✗ [7]	✗ [14]	⚙ [7]

API	已啟用	Disabled	待刪除 待刪除複本	待匯入	Unavailable	正在建立	更新中
加密	✓	 [1]	 [2] 或 [3]	 [5]	 [11]	 [14]	✓
GenerateDataKey	✓	 [1]	 [2] 或 [3]	 [5]	 [11]	 [14]	✓
GenerateDataKeyPair	✓	 [1]	 [2] 或 [3]	 [5]	 [7]	 [14]	✓
GenerateDataKeyPairWithoutPlaintext	✓	 [1]	 [2] 或 [3]	 [5]	 [7]	 [14]	✓
GenerateDataKeyWithoutPlaintext	✓	 [1]	 [2] 或 [3]	 [5]	 [11]	 [14]	✓
GenerateMac	✓	 [1]	 [2] 或 [3]	 [5]	N/A	 [14]	✓
GetKeyPolicy	✓	✓	✓	✓	✓	✓	✓

API	已啟用	Disabled	待刪除 待刪除複本	待匯入	Unavailable	正在建立	更新中
GetKeyRotationStatus	 [7]	 [7]	 [7]	 [6]	 [7]	 [7]	 [7]
GetParametersForImport	 [9]	 [9]	 [8] 或 [9]		 [9]	 [14]	 [15]
GetPublicKey			 [2] 或 [3]		N/A	 [14]	
ImportKeyMaterial	 [9]	 [9]	 [9]		 [9]	 [14]	
ListAliases							
ListGrants							
ListKeyPolicies							
ListKeyRotations	 [7]	 [7]	 [7]		 [7]	 [7]	 [7]
ListResourceTags							

API	已啟用	Disabled	待刪除 待刪除複本	待匯入	Unavailable	正在建立	更新中
PutKeyPolicy	✓	✓	✓	✓	✓	✓	✓
ReEncrypt	✓	 [1]	 [2] 或 [3]	 [5]	 [11]	 [14]	✓
Replicate Key	✓	 [1]	 [2] 或 [3]	 [5]	N/A	 [14]	 [15]
RetireGrant	✓	✓	✓	✓	✓	✓	✓
RevokeGrant	✓	✓	✓	✓	✓	✓	✓
RotateKey OnDemand	 [7]	 [1] 或 [7]	 [3] 或 [7]	 [5]	 [7]	 [14]	 [7]
ScheduleKeyDeletion	✓	✓	 [3]	✓	✓	✓	 [15]
符號	✓	 [1]	 [2] 或 [3]	 [5]	N/A	 [14]	✓

API	已啟用	Disabled	待刪除 待刪除複 本	待匯入	Unavailab le	正在建立	更新中
TagResource	✓	✓	✗ [3]	✓	✓	✓	✓
UntagResource	✓	✓	✗ [3]	✓	✓	✓	✓
UpdateAlias	✓	✓	⓪ [10]	✓	✓	✓	✓
UpdateKeyDescription	✓	✓	✗ [3]	✓	✓	✓	✓
UpdatePrimaryRegion	✓	✗ [1]	✗ [2] 或 [3]	✗ [5]	N/A	✗ [14]	✓
確認	✓	✗ [1]	✗ [2] 或 [3]	✗ [5]	N/A	✗ [14]	✓
VerifyMac	✓	✗ [1]	✗ [2] 或 [3]	✗ [5]	N/A	✗ [14]	✓

資料表詳細資訊

- [1] DisabledException: `<key ARN>` is disabled.
- [2] DisabledException: `<key ARN>` is pending deletion (or pending replica deletion).
- [3] KMSInvalidStateException: `<key ARN>` is pending deletion (or pending replica deletion).
- [4] KMSInvalidStateException: `<key ARN>` is not pending deletion (or pending replica deletion).
- [5] KMSInvalidStateException: `<key ARN>` is pending import because no key material has ever been imported or one of the imported key materials is deleted or expired.
- [6] UnsupportedOperationException: `<key ARN>` origin is EXTERNAL which is not valid for this operation.
- 【7】 如果 KMS 金鑰位於自訂金鑰存放區中：UnsupportedOperationException。
- [8] 如果 KMS 金鑰已匯入金鑰材料：KMSInvalidStateException
- 【9】 如果 KMS 金鑰無法匯入金鑰材料：UnsupportedOperationException。
- [10] 如果來源 KMS 金鑰正在等待刪除，則命令成功。如果目的地 KMS 金鑰正在等待刪除，則命令失敗，並出現錯誤：KMSInvalidStateException： `<key ARN>` is pending deletion.
- [11] KMSInvalidStateException: `<key ARN>` is unavailable. 對於無法使用的 KMS 金鑰，您無法執行此操作。
- [12] 操作成功，但 KMS 金鑰在變成可用之前，狀態不會變更。
- [13] 當自訂金鑰存放區中的 KMS 金鑰等待刪除時，即使 KMS 金鑰變成無法使用，其金鑰狀態仍為 PendingDeletion。這可讓您在等待期間隨時取消刪除 KMS 金鑰。
- 【14】 在複寫多區域金鑰 () 時KMSInvalidStateException: `<key ARN>` is creating. AWS KMS 擲回此例外狀況ReplicateKey。
- 【15】 在更新多區域金鑰的主要區域 () 時KMSInvalidStateException: `<key ARN>` is updating. AWS KMS 擲回此例外狀況UpdatePrimaryRegion。

金鑰類型參考

AWS KMS 支援不同類型的 KMS 金鑰的不同功能。例如，您只能使用[對稱加密 KMS 金鑰](#)產生[對稱資料金鑰](#)和[非對稱資料金鑰對](#)。此外，僅對稱加密 KMS 金鑰支援[金鑰資料匯入](#)和[金鑰自動輪換](#)，而且只能在[自訂金鑰存放區](#)中建立對稱加密 KMS 金鑰。

此參考包括兩個資料表。

- [金鑰類型資料表](#)列出對對稱加密 KMS 金鑰、非對稱 KMS 金鑰和 HMAC KMS 金鑰有效的 AWS KMS 操作。
- [特殊功能資料表](#)列出多區域 KMS 金鑰、含有匯入金鑰材料的 KMS 金鑰以及自訂金鑰存放區中的 KMS 金鑰的有效 AWS KMS 操作。

金鑰類型資料表

您可能需要水平或垂直捲動，才能查看此資料表中的所有資料。

AWS KMS API 操作	對稱加密 KMS 金鑰	HMAC KMS 金鑰	非對稱 KMS 金鑰 (ENCRYPT/DECRYPT)	非對稱 KMS 金鑰 (SIGN_VERIFY)	非對稱 KMS 金鑰 (KEY_AGREEMENT)
CancelKeyDeletion	是	是	是	是	是
CreateAlias	是	是	是	是	是
CreateGrant	是	是	是	是	是
CreateKey	是	是	是	是	是
解密	是	否	是	否	否
DeleteAlias	是	是	是	是	是
DeleteImportedKeyMaterial	是	是	是	是	是
僅對含有匯入金鑰材料的 KMS 金鑰有效 (Origin 為 EXTERNAL)。					
DeriveSharedSecret	否	否	否	否	是
DescribeKey	是	是	是	是	是

AWS KMS API 操作	對稱加密 KMS 金鑰	HMAC KMS 金鑰	非對稱 KMS 金鑰 (ENCRYPT/DECRYPT)	非對稱 KMS 金鑰 (SIGN_VERIFY)	非對稱 KMS 金鑰 (KEY_AGREEMENT)
DisableKey	是	是	是	是	是
DisableKeyRotation	是 僅適用於具有金鑰材料 (Origin 為 AWS_KMS) 的 KMS AWS KMS 金鑰。	否	否	否	否
EnableKey	是	是	是	是	是
EnableKeyRotation	是 僅適用於具有金鑰材料 (Origin 為 AWS_KMS) 的 KMS AWS KMS 金鑰。	否	否	否	否

AWS KMS API 操作	對稱加密 KMS 金鑰	HMAC KMS 金鑰	非對稱 KMS 金鑰 (ENCRYPT/DECRYPT)	非對稱 KMS 金鑰 (SIGN_VERIFY)	非對稱 KMS 金鑰 (KEY_AGREEMENT)
加密	是	否	是	否	否
GenerateDataKey	是	否	否	否	否
GenerateDataKeyPair 產生受對稱加密 KMS 金鑰保護的非對稱資料金鑰對。	是 對於自訂金鑰存放區中的 KMS 金鑰無效。	否	否	否	否
GenerateDataKeyPairWithoutPlaintext 產生受對稱加密 KMS 金鑰保護的非對稱資料金鑰對。	是 對於自訂金鑰存放區中的 KMS 金鑰無效。	否	否	否	否
GenerateDataKeyWithoutPlaintext	是	否	否	否	否
GenerateMac	否	是	否	否	否
GetKeyPolicy	是	是	是	是	是

AWS KMS API 操作	對稱加密 KMS 金鑰	HMAC KMS 金鑰	非對稱 KMS 金鑰 (ENCRYPT/DECRYPT)	非對稱 KMS 金鑰 (SIGN_VERIFY)	非對稱 KMS 金鑰 (KEY_AGREEMENT)
GetKeyRotationStatus	是	是 (KeyRotationEnabled 一律為 false.)	是 (KeyRotationEnabled 一律為 false.)	是 (KeyRotationEnabled 一律為 false.)	是 (KeyRotationEnabled 一律為 false.)
GetParametersForImport 僅對含有匯入金鑰材料的 KMS 金鑰有效 (Origin 為 EXTERNAL)。	是	是	是	是	是
GetPublicKey	否	否	是	是	是
ImportKeyMaterial 僅對含有匯入金鑰材料的 KMS 金鑰有效 (Origin 為 EXTERNAL)。	是	是	是	是	是
ListAliases	是	是	是	是	是
ListGrants	是	是	是	是	是
ListKeyPolicies	是	是	是	是	是

AWS KMS API 操作	對稱加密 KMS 金鑰	HMAC KMS 金鑰	非對稱 KMS 金鑰 (ENCRYPT/DECRYPT)	非對稱 KMS 金鑰 (SIGN_VERIFY)	非對稱 KMS 金鑰 (KEY_AGREEMENT)
ListKeyRotations	是	是 (Rotations 欄位一律為 null 或空白。)	是 (Rotations 欄位一律為 null 或空白。)	是 (Rotations 欄位一律為 null 或空白。)	是 (Rotations 欄位一律為 null 或空白。)
ListResourceTags	是	是	是	是	是
ListRetirableGrants	是	是	是	是	是
PutKeyPolicy	是	是	是	是	是
ReEncrypt	是	否	是	否	否
ReplicateKey - 僅適用於多區域金鑰	是	是	是	是	是
RetireGrant	是	是	是	是	是
RevokeGrant	是	是	是	是	是

AWS KMS API 操作	對稱加密 KMS 金鑰	HMAC KMS 金鑰	非對稱 KMS 金鑰 (ENCRYPT/DECRYPT)	非對稱 KMS 金鑰 (SIGN_VERIFY)	非對稱 KMS 金鑰 (KEY_AGREEMENT)
RotateKeyOnDemand	是 僅適用於使用 AWS_KMS:EXTERNAL 始伺服器客戶受管對稱加密 KMS 金鑰。	否	否	否	否
ScheduleKeyDeletion	是	是	是	是	是
符號	否	否	否	是	否
TagResource	是	是	是	是	是
UntagResource	是	是	是	是	是
UpdateAlias	是	是	是	是	是
目前的 KMS 金鑰和新的 KMS 金鑰必須是相同類型 (兩者皆為對稱或皆為非對稱或皆為 HMAC) , 且必須具有相同的 金鑰用途 。					

AWS KMS API 操作	對稱加密 KMS 金鑰	HMAC KMS 金鑰	非對稱 KMS 金鑰 (ENCRYPT/DECRYPT)	非對稱 KMS 金鑰 (SIGN_VERIFY)	非對稱 KMS 金鑰 (KEY_AGREEMENT)
UpdateKeyDescription	是	是	是	是	是
UpdateReplicaRegion - 僅適用於多區域金鑰	是	是	是	是	是
VerifyMac	否	否	否	是	否
VerifyMac	否	是	否	否	否

特殊功能資料表

此資料表顯示每種特殊用途金鑰類型支援的 AWS KMS API 操作。

閱讀此資料表時，請注意下列互動：

- [多區域金鑰](#)：
 - 多區域金鑰可以是對稱加密 KMS 金鑰、非對稱 KMS 金鑰、HMAC KMS 金鑰、含有匯入金鑰材料的 KMS 金鑰。
 - 您無法在自訂金鑰存放區建立多區域金鑰。
- [匯入的金鑰資料](#)
 - 您可針對對稱加密 KMS 金鑰、非對稱 KMS 金鑰，以及 HMAC KMS 金鑰匯入金鑰資料。
 - 您可[利用匯入金鑰資料來建立多區域金鑰](#)。
 - 您無法在自訂金鑰存放區中建立含有匯入金鑰材料的金鑰。
 - 具有匯入金鑰資料的 KMS 金鑰不支援自動金鑰輪換 (EnableKeyRotation、DisableKeyRotation)。
 - 具有匯入金鑰材料的對稱加密 KMS 金鑰支援隨需金鑰輪換 (RotateKeyOnDemand)。

- [自訂金鑰存放區](#)

- 自訂金鑰存放區僅支援對稱加密 KMS 金鑰。
- 自訂金鑰存放區中的 KMS 金鑰不支援非對稱金鑰對 (GenerateDataKeyPair、GenerateDataKeyPairWithoutPlaintext) 的對稱操作。
- 自訂金鑰存放區中的 KMS 金鑰不支援自動金鑰輪換 (EnableKeyRotation、DisableKeyRotation)。
- 您無法在自訂金鑰存放區建立多區域金鑰。

您可能需要水平或垂直捲動，才能查看此資料表中的所有資料。

AWS KMS API 操作	多區域金鑰	匯入的金鑰資料	在自訂金鑰存放區中的 KMS 金鑰
CancelKeyDeletion	✓	✓	✓
CreateAlias	✓	✓	✓
CreateGrant	✓	✓	✓
CreateKey 您可以使用 CreateKey 來建立多區域主要金鑰、含有匯入金鑰材料的 KMS 金鑰，或者自訂金鑰存放區中的 KMS 金鑰。若要建立多區域複本金鑰，請使用 ReplicateKey 。	✓	✓	✓
解密 只有在 KeyUsage 為 ENCRYPT_D ECRYPT 時有效	✓	✓	✓

AWS KMS API 操作	多區域金鑰	匯入的金鑰資料	在自訂金鑰存放區中的 KMS 金鑰
DeleteAlias	✓	✓	✓
DeleteImportedKeyMaterial	✓ 僅對含有匯入金鑰材料的金鑰有效 (Origin 為 EXTERNAL)	✓	✗
DeriveSharedSecret	✓ 僅在 KeyUsage 為 時有效 KEY_AGREEMENT)	✓ 僅在 KeyUsage 為 時有效 KEY_AGREEMENT)	✗
DescribeKey	✓	✓	✓
DisableKey	✓	✓	✓
DisableKeyRotation	✓ 僅適用於具有金鑰材料 (Origin 為 AWS_KMS) 的對稱加密 AWS KMS 金鑰。	✗	✗

AWS KMS API 操作	多區域金鑰	匯入的金鑰資料	在自訂金鑰存放區中的 KMS 金鑰
EnableKey	 僅對對稱加密 KMS 金鑰有效		
EnableKeyRotation	 僅適用於具有金鑰材料 (Origin 為 AWS_KMS) 的對稱加密 AWS KMS 金鑰。		
加密	 只有在 KeyUsage 為 ENCRYPT_DECRYPT 時有效		
GenerateDataKey	 僅對對稱加密 KMS 金鑰有效		
GenerateDataKeyPair	 僅對對稱加密 KMS 金鑰有效		

AWS KMS API 操作	多區域金鑰	匯入的金鑰資料	在自訂金鑰存放區中的 KMS 金鑰
GenerateDataKeyPairWithoutPlaintext	✓ 僅對對稱加密 KMS 金鑰有效	✓	✗
GenerateDataKeyWithoutPlaintext	✓ 僅對對稱加密 KMS 金鑰有效	✓	✓
GenerateMac 僅對 HMAC KMS 金鑰有效	✓	✓	✗
GetKeyPolicy	✓	✓	✓
GetKeyRotationStatus	✓	✓ (KeyRotationEnabled 一律為 false.)	✗
GetParametersForImport	✓ 僅對含有匯入金鑰材料的金鑰有效 (Origin 為 EXTERNAL)。	✓	✗
GetPublicKey 僅對 <u>非對稱 KMS 金鑰</u> 有效。	✓	✓	✗

AWS KMS API 操作	多區域金鑰	匯入的金鑰資料	在自訂金鑰存放區中的 KMS 金鑰
ImportKeyMaterial	 僅對含有匯入金鑰材料的金鑰有效 (Origin 為 EXTERNAL)。		
ListAliases			
ListGrants			
ListKeyPolicies			
ListKeyRotations	 僅適用於具有 AWS_KMS 或 EXTERNAL 原始伺服器的對稱加密金鑰。	 僅在對稱加密金鑰上有效。	
ListResourceTags			
ListRetirableGrants			
PutKeyPolicy			

AWS KMS API 操作	多區域金鑰	匯入的金鑰資料	在自訂金鑰存放區中的 KMS 金鑰
ReEncrypt	 只有在 KeyUsage 為 ENCRYPT_DECRYPT 時有效		
ReplicateKey	 僅對多區域主金鑰有效。	 僅對多區域主金鑰有效。	
RetireGrant			
RevokeGrant			
RotateKeyOnDemand	 僅適用於具有 AWS_KMS 或 EXTERNAL 原始伺服器的對稱加密金鑰。	 僅在對稱加密金鑰上有效。	
ScheduleKeyDeletion			
符號 只有在 KeyUsage 為 SIGN_VERIFY 時有效。			

AWS KMS API 操作	多區域金鑰	匯入的金鑰資料	在自訂金鑰存放區中的 KMS 金鑰
TagResource	✓	✓	✓
UntagResource	✓	✓	✓
UpdateAlias - 目前的 KMS 金鑰和新的 KMS 金鑰必須是相同類型 (兩者皆為對稱或皆為非對稱或皆為 HMAC)，且必須具有相同的 金鑰用途 。	✓	✓	✓
UpdateKeyDescription	✓	✓	✓
UpdateReplicaRegion	✓	✓ 僅對多區域金鑰有效。	✗
確認 只有在 KeyUsage 為 SIGN_VERIFY 時有效。	✓	✓	✗
VerifyMac 僅對 HMAC KMS 金鑰有效	✓	✓	✗

金鑰規格參考

在建立非對稱 KMS 金鑰或 HMAC KMS 金鑰時，需選取其 [key spec](#) (金鑰規格)。金鑰規格是每個的屬性 AWS KMS key，代表 KMS 金鑰的密碼編譯組態。您可以在建立 KMS 金鑰時選擇金鑰規格，之後便無法進行變更。如果選取了錯誤的金鑰規格，請[刪除 KMS 金鑰](#)，再建立新的金鑰規格。

Note

KMS 金鑰的金鑰規格稱為「客戶主要金鑰規格」。已取代 [CreateKey](#) 操作的 `CustomerMasterKeySpec` 參數。請改用 `KeySpec` 參數。`CreateKey` 和 [DescribeKey](#) 操作的回應包括具有相同值的 `KeySpec` 和 `CustomerMasterKeySpec`。

金鑰規格會決定 KMS 金鑰是對稱還是非對稱、KMS 金鑰中的金鑰材料類型，以及 KMS 金鑰 AWS KMS 支援的加密演算法、簽署演算法或訊息身分驗證碼 (MAC) 演算法。您選擇的金鑰規格通常會依您的使用案例和法規要求決定。不過，具有不同金鑰規格的 KMS 金鑰的密碼編譯操作定價也不同，而且有不同的配額。如需定價詳細資訊，請參閱 [AWS Key Management Service 定價](#)。如需請求配額的詳細資訊，請參閱 [請求配額](#)。

如要限制委託人在建立 KMS 金鑰時可使用的金鑰規格，請使用 [kms:KeySpec](#) 條件索引鍵。您也可以使用 `kms:KeySpec` 條件金鑰，允許委託人僅在具有特定金鑰規格的 KMS 金鑰上呼叫 AWS KMS 操作。例如，您可以拒絕排程刪除任何具有 `RSA_4096` 金鑰規格之 KMS 金鑰的許可。

AWS KMS 支援下列 KMS 金鑰的金鑰規格：

[對稱加密金鑰規格](#) (預設)

- `SYMMETRIC_DEFAULT`

[RSA 金鑰規格](#) (加密和解密或簽署和驗證)

- `RSA_2048`
- `RSA_3072`
- `RSA_4096`

[橢圓曲線金鑰規格](#)

- 非對稱 NIST 標準 [橢圓曲線金鑰對](#) (簽署和驗證 - 或 - 衍生共用秘密)
 - `ECC_NIST_P256` (`secp256r1`)
 - `ECC_NIST_P384` (`secp384r1`)
 - `ECC_NIST_P521` (`secp521r1`)
 - `ECC_NIST_EDWARDS25519` (`ed25519`) - 僅限簽署和驗證
 - 注意：對於 `ECC_NIST_EDWARDS25519` KMS 金鑰，`ED25519_SHA_512` 簽署演算法需要 [MessageType:RAW](#)，而 `ED25519_PH_SHA_512` 需要 [MessageType:DIGEST](#)。這些訊息類型無法互換使用。
- 其他非對稱橢圓曲線金鑰對 (簽署和驗證)

- ECC_SECG_P256K1 ([secp256k1](#))，常用於加密貨幣。

[SM2 金鑰規格](#) (加密和解密 – 或 – 簽署和驗證 – 或 – 衍生共用秘密)

- SM2 (僅限中國區域)

[HMAC 金鑰規格](#)

- HMAC_224
- HMAC_256
- HMAC_384
- HMAC_512

[ML-DSA 金鑰規格](#)

- ML_DSA_44
- ML_DSA_65
- ML_DSA_87

SYMMETRIC_DEFAULT 金鑰規格

預設的金鑰規格 SYMMETRIC_DEFAULT 是對稱加密 KMS 金鑰的金鑰規格。當您在 AWS KMS 主控台中選取對稱金鑰類型和加密和解密金鑰用量時，它會選取 SYMMETRIC_DEFAULT 金鑰規格。在 [CreateKey](#) 操作中，如果您不指定 KeySpec 值，即會選取 SYMMETRIC_DEFAULT。如果沒有使用不同金鑰規格的理由，SYMMETRIC_DEFAULT 是很好的選擇。

SYMMETRIC_DEFAULT 代表 AES-256-GCM，這是一種以 [Galois 計數器模式](#) (GCM) 中的 [進階加密標準](#) (AES) 為基礎的對稱演算法，具有 256 位元金鑰，是安全加密的業界標準。此演算法產生的加密文字支援額外的驗證資料 (AAD)，如 [加密內容](#)，而 GCM 則提供對加密文字的額外完整性檢查。

使用 AES-256-GCM 加密的資料現在和未來都受到保護。密碼學家認為此演算法具有「量子抗性」。在理論上，未來針對以 256 位元 AES-GCM 金鑰建立的加密文字所做的大規模量子運算攻擊，會 [將金鑰的有效安全性降低到 128 位元](#)。但是，此安全層級足以對 AWS KMS 加密文字進行暴力破解攻擊。

中國區域是唯一的例外，在該處 SYMMETRIC_DEFAULT 代表使用 SM4 加密的 128 位元對稱金鑰。您只能在中國區域內建立 128 位元 SM4 金鑰。您無法在中國區域內建立 256 位元 AES-GCM KMS 金鑰。

您可以在 中使用對稱加密 KMS 金鑰 AWS KMS 來加密、解密和重新加密資料，以及保護產生的資料金鑰和資料金鑰對。與 整合 AWS 的服務 AWS KMS 會使用對稱加密 KMS 金鑰來加密靜態資料。您可以 [將自己的金鑰資料匯入](#) 對稱加密 KMS 金鑰，並在 [自訂金鑰存放區](#) 中建立對稱加密 KMS 金鑰。如需對稱和非對稱 KMS 金鑰執行之操作的比較表，請參閱 [比較對稱和非對稱 KMS 金鑰](#)。

您可以在 [中](#) 使用對稱加密 KMS 金鑰 AWS KMS 來加密、解密和重新加密資料，並產生資料金鑰和資料金鑰對。您可以建立 [多區域](#) 對稱加密 KMS 金鑰、[將自己的金鑰資料匯入](#) 對稱加密 KMS 金鑰，並在 [自訂金鑰存放區](#) 中建立對稱加密 KMS 金鑰。如需不同類型的 KMS 金鑰執行之操作的比較表，請參閱 [金鑰類型參考](#)。

RSA 金鑰規格

當您使用 RSA 金鑰規格時，會 AWS KMS 建立具有 RSA 金鑰對的非對稱 KMS 金鑰。私有金鑰永遠不會讓 AWS KMS 處於未加密狀態。您可以在 [中](#) 使用公有金鑰 AWS KMS，或下載公有金鑰以在 [外部](#) 使用 AWS KMS。

Warning

當您在 [外部](#) 加密資料時 AWS KMS，請確定您可以解密加密文字。如果您使用已從 AWS KMS 刪除之 KMS 金鑰的公有金鑰、設定用於簽署和驗證之 KMS 金鑰的公有金鑰，或不受 KMS 金鑰支援的加密演算法，該資料無法復原。

在 [中](#) AWS KMS，您可以將非對稱 KMS 金鑰與 RSA 金鑰對搭配使用，以進行加密和解密，或簽署和驗證，但不能同時使用兩者。此屬性稱為 [Key usage](#)，與金鑰規格分開決定，但您應該在選取金鑰規格之前做出該決定。

AWS KMS 支援下列 RSA 金鑰規格進行加密和解密，或簽署和驗證：

- RSA_2048
- RSA_3072
- RSA_4096

RSA 金鑰規格因 RSA 金鑰的位元長度而不同。您選擇的 RSA 金鑰規格可能隨安全標準或任務需求而定。任務一般會使用實用且可負擔的最長密鑰。具有不同 RSA 金鑰規格之 KMS 金鑰的密碼編譯操作定價也不同。如需 AWS KMS 定價的資訊，請參閱 [AWS Key Management Service 定價](#)。如需請求配額的詳細資訊，請參閱 [請求配額](#)。

用於加密和解密的 RSA 金鑰規格

使用 RSA 非對稱 KMS 金鑰加密和解密時，您會使用公有金鑰加密，並使用私有金鑰解密。當您在 [Encrypt](#) 中呼叫 RSA KMS 金鑰 AWS KMS 的操作時，AWS KMS 會使用 RSA 金鑰對中的公有金

鑰，以及您指定的加密演算法來加密資料。若要解密加密文字，請呼叫 `Decrypt` 操作並指定相同的 KMS 金鑰和加密演算法。AWS KMS 然後，會使用 RSA 金鑰對中的私有金鑰來解密您的資料。

您也可以下載公有金鑰，並使用它來加密外部的資料 AWS KMS。請務必使用 AWS KMS 支援 RSA KMS 金鑰的加密演算法。若要解密加密文字，請使用相同的 KMS 金鑰和加密演算法呼叫 `Decrypt` 函數。

AWS KMS 支援兩種具有 RSA 金鑰規格的 KMS 金鑰加密演算法。這些演算法同在 [PKCS #1 v2.2](#) 中定義，但其內部使用的雜湊函數不同。在 AWS KMS 中，RSAES_OAEP 演算法對雜湊目的和遮罩產生函數 (MGF1) 一律使用相同的雜湊函數。當您呼叫 [Encrypt](#) 和 [Decrypt](#) 操作時，必須指定加密演算法。您可以為每個請求選擇不同的演算法。

支援 RSA 金鑰規格的加密演算法

加密演算法	演算法說明
RSAES_OAEP_SHA_1	PKCS #1 v2.2 第 7.1 節。RSA 加密與 OAEP 填補在雜湊和 MGF1 遮罩產生函數中都使用 SHA-1 加一個空標籤。
RSAES_OAEP_SHA_256	PKCS #1 第 7.1 節。RSA 加密與 OAEP 填補在雜湊和 MGF1 遮罩產生函數中都使用 SHA-256 加一個空標籤。

您無法將 KMS 金鑰設定為使用特定的加密演算法。但可以使用 [kms:EncryptionAlgorithm](#) 政策條件指定允許主體搭配 KMS 金鑰使用的加密演算法。

若要取得 KMS 金鑰的加密演算法，請在 AWS KMS 主控台中[檢視 KMS 金鑰的密碼編譯組態](#)，或使用 [DescribeKey](#) 操作。AWS KMS 當您在 AWS KMS 主控台中下載公有金鑰或使用 [GetPublicKey](#) 操作時，也會提供金鑰規格和加密演算法。

您可根據能在每個請求中加密的純文字資料長度，選擇 RSA 金鑰規格。下表顯示您在單次呼叫 [Encrypt](#) 操作中可以加密的純文字大小上限 (以位元組為單位)。這些值與金鑰規格和加密演算法不同。若要進行比較，您可以使用對稱加密 KMS 金鑰一次加密多達 4096 個位元組。

若要計算這些演算法的純文字長度上限 (以位元組為單位)，請使用以下公式： $(key\_size\_in\_bits / 8) - (2 * hash\_length\_in\_bits / 8) - 2$ 。例如，具有 SHA-256 之 RSA_2048 的純文字大小上限 (位元組) 為 $(2048/8) - (2 * 256/8) - 2 = 190$ 。

Encrypt 操作中的純文字大小上限 (位元組)

	加密演算法	
金鑰規格	RSAES_OAEP_SHA_1	RSAES_OAEP_SHA_256
RSA_2048	214	190
RSA_3072	342	318
RSA_4096	470	446

用於簽署和驗證的 RSA 金鑰規格

使用 RSA 非對稱 KMS 金鑰執行簽署和驗證作業時，您會使用私有金鑰產生訊息的簽章，並使用公有金鑰驗證該簽章。

當您 AWS KMS 在中呼叫非對稱 KMS 金鑰 Sign 的操作時，AWS KMS 會使用 RSA 金鑰對中的私有金鑰、訊息和您指定的簽署演算法來產生簽章。若要驗證簽章，請呼叫 [Verify](#) 操作。指定簽章，加上相同的 KMS 金鑰、訊息和簽署演算法。AWS KMS 然後，會使用 RSA 金鑰對中的公有金鑰來驗證簽章。您也可以下載公有金鑰，並使用它來驗證外部的簽章 AWS KMS。

AWS KMS 針對具有 RSA 金鑰規格的所有 KMS 金鑰，支援下列簽署演算法。當您呼叫 [Sign](#) 和 [Verify](#) 操作時，必須指定簽署演算法。您可以為每個請求選擇不同的演算法。使用 RSA 金鑰對進行簽署時，偏好使用 RSASSA-PSS 演算法。我們包含 RSASSA-PKCS1-v1_5 演算法，以便與現有應用程式相容。

支援 RSA 金鑰規格的簽署演算法

簽署演算法	演算法說明
RSASSA_PSS_SHA_256	PKCS #1 v2.2 第 8.1 節，具有 PSS 填補和使用 SHA-256 的 RSA 簽章，適用於訊息摘要和 MGF1 遮罩產生函數以及 256 位元的 salt
RSASSA_PSS_SHA_384	PKCS #1 v2.2 第 8.1 節，具有 PSS 填補和使用 SHA-384 的 RSA 簽章，適用於訊息摘要和 MGF1 遮罩產生函數以及 384 位元的 salt

簽署演算法	演算法說明
RSASSA_PSS_SHA_512	PKCS #1 v2.2 第 8.1 節，具有 PSS 填補和使用 SHA-512 的 RSA 簽章，適用於訊息摘要和 MGF1 遮罩產生函數以及 512 位元的 salt
RSASSA_PKCS1_V1_5_SHA_256	PKCS #1 v2.2 第 8.2 節，具有 PKCS #1v1.5 填補和 SHA-256 的 RSA 簽章
RSASSA_PKCS1_V1_5_SHA_384	PKCS #1 v2.2 第 8.2 節，具有 PKCS #1v1.5 填補和 SHA-384 的 RSA 簽章
RSASSA_PKCS1_V1_5_SHA_512	PKCS #1 v2.2 第 8.2 節，具有 PKCS #1v1.5 填補和 SHA-512 的 RSA 簽章

您無法將 KMS 金鑰設定為使用特定的簽署演算法。不過，您可以使用 [kms:SigningAlgorithm](#) 政策條件指定允許主體搭配 KMS 金鑰使用的簽署演算法。

若要取得 KMS 金鑰的簽署演算法，請在 AWS KMS 主控台或使用 [DescribeKey](#) 操作檢視 KMS 金鑰的 [密碼編譯組態](#)。AWS KMS 當您在 AWS KMS 主控台下載公有金鑰或使用 [GetPublicKey](#) 操作時，也會提供金鑰規格和簽署演算法。

橢圓曲線金鑰規格

當您使用橢圓曲線 (ECC) 金鑰規格時，AWS KMS 會建立具有 ECC 金鑰對的非對稱 KMS 金鑰，用於簽署和驗證或衍生共用秘密（但不能同時建立兩者）。產生簽章或衍生共用秘密的私有金鑰絕不會保持 AWS KMS 未加密狀態。您可以使用公有金鑰來[驗證內部的簽章](#) AWS KMS，或[下載公有金鑰](#)以供外部使用 AWS KMS。

當您使用 Edwards Curve 金鑰規格時，AWS KMS 會建立具有 Ed25519 金鑰對的非對稱 KMS 金鑰，以進行簽署和驗證。產生簽章的私有金鑰永遠不會讓 AWS KMS 處於未加密狀態。您可以使用公有金鑰來[驗證內的簽章](#) AWS KMS，或[下載公有金鑰](#)以供外部使用 AWS KMS。

AWS KMS 支援下列非對稱 KMS 金鑰的 ECC 金鑰規格。

- 非對稱 NIST 標準橢圓曲線金鑰對（簽署和驗證 - 或 - 衍生共用秘密）
 - ECC_NIST_P256 (secp256r1)
 - ECC_NIST_P384 (secp384r1)

- ECC_NIST_P521 (secp521r1)
- ECC_NIST_EDWARDS25519 (ed25519) - 僅限簽署和驗證
 - 注意：對於 ECC_NIST_EDWARDS25519 KMS 金鑰，ED25519_SHA_512 簽署演算法需要 [MessageType:RAW](#)，而 ED25519_PH_SHA_512 需要 [MessageType:DIGEST](#)。這些訊息類型無法互換使用。
- 其他非對稱橢圓曲線金鑰對 (簽署和驗證)
 - ECC_SECG_P256K1 ([secp256k1](#))，常用於加密貨幣。

您選擇的 ECC 金鑰規格可能隨安全標準或任務需求而定。任務一般會使用實用且可負擔的最多點曲線。

如果您要建立非對稱 KMS 金鑰來[衍生共用秘密](#)，請使用其中一個 NIST 標準橢圓曲線金鑰規格 (ECC_SECG_P256K1 和 ECC_NIST_EDWARDS25519 除外)。衍生共用秘密唯一支援的金鑰協議演算法是[橢圓曲線密碼編譯 Cofactor Diffie-Hellman 基本](#) (ECDH)。如需如何離線衍生共用秘密的範例，請參閱 [the section called “離線衍生共用秘密”](#)。

如果您要建立非對稱 KMS 金鑰以搭配使用加密貨幣，請使用 ECC_SECG_P256K1 金鑰規格。此金鑰規格也可供其他用途使用，但對 Bitcoin 和其他加密貨幣而言是必要項目。

下表顯示每個 ECC 金鑰規格 AWS KMS 支援的簽署演算法。您無法將 KMS 金鑰設定為使用特定的簽署演算法。不過，您可以使用 [kms:SigningAlgorithm](#) 政策條件指定允許委託人搭配 KMS 金鑰使用的簽署演算法。

支援 ECC 金鑰規格的簽署演算法

金鑰規格	簽署演算法	演算法說明
ECC_NIST_P256	ECDSA_SHA_256	NIST FIPS 186-4 第 6.4 節，適用於訊息摘要之金鑰和 SHA-256 所指定的使用曲線的 ECDSA 簽章。
ECC_NIST_P384	ECDSA_SHA_384	NIST FIPS 186-4 第 6.4 節，適用於訊息摘要之金鑰和 SHA-384 所指定的使用曲線的 ECDSA 簽章。

金鑰規格	簽署演算法	演算法說明
ECC_NIST_P521	ECDSA_SHA_512	NIST FIPS 186-4 第 6.4 節，適用於訊息摘要之金鑰和 SHA-512 所指定的使用曲線的 ECDSA 簽章。
ECC_SECG_P256K1	ECDSA_SHA_256	NIST FIPS 186-4 第 6.4 節，適用於訊息摘要之金鑰和 SHA-256 所指定的使用曲線的 ECDSA 簽章。
ECC_NIST_EDWARDS25519	ED25519_SHA_512	NIST FIPS 186-5，第 7 節，使用金鑰和 SHA-512 為訊息摘要指定的曲線的 EdDSA 簽章。KMS MessageType:RAW 需要使用此演算法。
ECC_NIST_EDWARDS25519	ED25519_PH_SHA_512	NIST FIPS 186-5，第 7 節，使用金鑰和 SHA-512 為訊息摘要指定的曲線的 EdDSA 簽章。KMS MessageType:DIGEST 需要使用此演算法。

HMAC KMS 金鑰的金鑰規格

AWS KMS 支援不同長度的對稱 HMAC 金鑰。金鑰規格可根據您的安全、法規或業務需求進行選取。金鑰的長度決定了 [GenerateMac](#) 和 [VerifyMac](#) 操作中使用的 MAC 演算法。通常，較長的金鑰更安全。使用適用於您的使用案例的最長金鑰。

HMAC 金鑰規格	MAC 演算法
HMAC_224	HMAC_SHA_224
HMAC_256	HMAC_SHA_256

HMAC 金鑰規格	MAC 演算法
HMAC_384	HMAC_SHA_384
HMAC_512	HMAC_SHA_512

ML-DSA 金鑰規格

ML-DSA 金鑰是 Module-Lattice-Based Digital Signature Algorithm (ML-DSA) 中使用的密碼編譯金鑰，專為量子後密碼編譯而設計。此演算法是 NIST（國家標準技術研究所）標準化工作的一部分，在[聯邦資訊處理標準 \(FIPS\) 204](#) 中特別概述。

ML-DSA 金鑰用於公有-私有金鑰對系統。私有金鑰用於簽署資料，而公有金鑰用於驗證簽章。即使面臨潛在的量子電腦威脅，此系統也能確保數位訊息或文件的真實性、完整性和不可否認性。

當您使用 ML-DSA 金鑰規格建立金鑰時，會使用 ML-DSA 金鑰對 AWS KMS 建立非對稱 KMS 金鑰以進行簽署和驗證。產生簽章的私有金鑰永遠不會讓 AWS KMS 處於未加密狀態。您可以使用公有金鑰來[驗證內部的簽章](#) AWS KMS，或[下載公有金鑰](#)以供外部使用 AWS KMS。

AWS KMS 支援下列非對稱 KMS 金鑰的 ML-DSA 金鑰規格：

- ML_DSA_44
- ML_DSA_65
- ML_DSA_87

AWS KMS 支援所有 ML-DSA 金鑰規格的 ML_DSA_SHAKE_256 簽署演算法。

SM2 金鑰規格 (僅限中國區域)

SM2 金鑰規格是在[中國商用密碼管理辦公室 \(OSCCA\)](#) 公佈的 GM/T 系列規格中定義的橢圓曲線金鑰規格。SM2 金鑰規格僅在中國區域提供。當您使用 SM2 金鑰規格時，會 AWS KMS 建立具有 SM2 金鑰對的非對稱 KMS 金鑰。您可以在 中使用 SM2 金鑰對 AWS KMS，或下載公有金鑰以在 外部使用 AWS KMS。如需詳細資訊，請參閱[the section called “使用 SM2 金鑰對進行離線驗證 \(僅限中國區域\)”](#)。

每個 KMS 金鑰只能有一個 [Key usage](#)。您可以使用 SM2 KMS 金鑰來簽署和驗證、加密和解密，或衍生共用秘密。在建立 KMS 金鑰時，您必須指定金鑰使用方式，且在金鑰建立之後即無法變更。

如果您要建立非對稱 KMS 金鑰來[衍生共用秘密](#)，請使用 SM2 金鑰規格。產生共用秘密唯一支援的金鑰協議演算法是[橢圓曲線密碼編譯 Cofactor Diffie-Hellman 基本](#) (ECDH)。

AWS KMS 支援下列 SM2 加密和簽署演算法：

- SM2PKE 加密演算法

SM2PKE 是 OSCCA 在 GM/T 0003.4-2012 中定義的橢圓曲線型加密演算法。

- SM2DSA 簽署演算法

SM2DSA 是 OSCCA 在 GM/T 0003.2-2012 中定義的橢圓曲線型簽署演算法。SM2DSA 需要使用 SM3 雜湊演算法雜湊的辨別 ID，然後與您傳遞的訊息或訊息摘要結合 AWS KMS。然後，此串連值會經過雜湊處理並由簽署 AWS KMS。

AWS KMS 許可

此資料表旨在協助您了解 AWS KMS 許可，以便控制對 AWS KMS 資源的存取。欄標題的定義顯示在表格下方。

您也可以[在服務授權參考主題的動作、資源和條件索引鍵](#) [AWS Key Management Service](#) 中了解 AWS KMS 許可。然而，該主題不會列出您可以用於細化每個許可的所有條件金鑰。

如需哪些 AWS KMS 操作對對稱加密 KMS 金鑰、非對稱 KMS 金鑰和 HMAC KMS 金鑰有效的詳細資訊，請參閱 [金鑰類型參考](#)。

Note

您可能需要水平或垂直捲動，才能查看資料表中的所有資料。

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
CancelKeyDeletion	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
				kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService
ConnectCustomKeyStore kms:ConnectCustomKeyStore	IAM 政策	否	*	kms:CallerAccount

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
CreateAlias kms:CreateAlias 若要使用此操作時，呼叫者需要兩個資源上的 kms:CreateAlias 許可： - 別名 (在 IAM 政策中) - KMS 金鑰 (在金鑰政策中) 如需詳細資訊，請參閱 控制對別名的存取 。	IAM 政策 (適用於別名)	否	Alias (別名)	無 (控制對別名的存取時)
	KMS 政策 (適用於 KMS 金鑰)	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService
CreateCustomKeyStore kms:CreateCustomKeyStore	IAM 政策	否	*	kms:CallerAccount

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
CreateGrant kms:CreateGrant	金鑰政策	是	KMS 金鑰	加密內容條件： kms:EncryptionContext:context-key kms:EncryptionContext:Keys 授予條件： kms:GrantConstraintType kms:GranteePrincipal kms:GrantIsForAWSResource kms:GrantOperations kms:RetiringPrincipal KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
				aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService
CreateKey kms:CreateKey	IAM 政策	否	*	kms:BypassPolicyLookupSafetyCheck kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ViaService aws : RequestTag/tag-key (AWS 全域條件金鑰) aws : ResourceTag/tag-key (AWS 全域條件金鑰) aws : TagKeys (AWS 全域條件金鑰)

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
解密 kms:Decrypt	金鑰政策	是	KMS 金鑰	密碼編譯操作的條件 kms:EncryptionAlgorithm kms:RequestAlias 加密內容條件： kms:EncryptionContext:context-key kms:EncryptionContext:Keys KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
DeleteAlias kms:DeleteAlias 若要使用此操作時，呼叫者需要兩個資源上的 kms:DeleteAlias 許可： - 別名 (在 IAM 政策中) - KMS 金鑰 (在金鑰政策中) 如需詳細資訊，請參閱 控制對別名的存取 。	IAM 政策 (適用於別名)	否	Alias (別名)	無 (控制對別名的存取時)
	KMS 政策 (適用於 KMS 金鑰)	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService
DeleteCustomKeyStore kms:DeleteCustomKeyStore	IAM 政策	否	*	kms:CallerAccount

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
DeleteImportedKeyMaterial kms:DeleteImportedKeyMaterial	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
DeriveSharedSecret kms:DeriveSharedSecret	金鑰政策	是	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService 密碼編譯操作的條件： kms:KeyAgreementAlgorithm
DescribeCustomKeyStores kms:DescribeCustomKeyStores	IAM 政策	否	*	kms:CallerAccount

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
DescribeKey kms:DescribeKey	金鑰政策	是	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService 其他條件： kms:RequestAlias

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
DisableKey kms:DisableKey	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
DisableKeyRotation kms:DisableKeyRotation	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService
DisconnectCustomKeyStore kms:DisconnectCustomKeyStore	IAM 政策	否	*	kms:CallerAccount

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
EnableKey kms:EnableKey	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
EnableKeyRotation kms:EnableKeyRotation	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService 自動金鑰輪換條件： kms:RotationPeriodInDays

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
加密 kms:Encrypt	金鑰政策	是	KMS 金鑰	密碼編譯操作的條件 kms:EncryptionAlgorithm kms:RequestAlias 加密內容條件： kms:EncryptionContext:context-key kms:EncryptionContext:Keys KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
GenerateDataKey kms:GenerateDataKey	金鑰政策	是	KMS 金鑰	密碼編譯操作的條件 kms:EncryptionAlgorithm kms:RequestAlias 加密內容條件： kms:EncryptionContext:context-key kms:EncryptionContext:Keys KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
GenerateDataKeyPair kms:GenerateDataKeyPair	金鑰政策	是	KMS 金鑰 產生受對稱加密 KMS 金鑰保護的非對稱資料金鑰對。	資料金鑰對的條件： kms:DataKeyPairSpec 密碼編譯操作的條件 kms:EncryptionAlgorithm kms:RequestAlias 加密內容條件： kms:EncryptionContext:context-key kms:EncryptionContextKeys KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰)

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
GenerateDataKeyPairWithoutPlaintext kms:GenerateDataKeyPairWithoutPlaintext	金鑰政策	是	KMS 金鑰 產生受對稱加密 KMS 金鑰保護的非對稱資料金鑰對。	資料金鑰對的條件： kms:DataKeyPairSpec 密碼編譯操作的條件 kms:EncryptionAlgorithm kms:RequestAlias 加密內容條件： kms:EncryptionContext:context-key kms:EncryptionContextKeys KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰)

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
				kms:ViaService
GenerateDataKeyWithoutPlaintext kms:GenerateDataKeyWithoutPlaintext	金鑰政策	是	KMS 金鑰	密碼編譯操作的條件 kms:EncryptionAlgorithm kms:RequestAlias 加密內容條件： kms:EncryptionContext:context-key kms:EncryptionContextKeys KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
GenerateMac kms:GenerateMac	金鑰政策	是	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService 密碼編譯操作的條件： kms:MacAlgorithm kms:RequestAlias
GenerateRandom kms:GenerateRandom	IAM 政策	N/A	*	無

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
GetKeyPolicy kms:GetKeyPolicy	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
GetKeyRotationStatus kms:GetKeyRotationStatus	金鑰政策	是	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
GetParametersForImport kms:GetParametersForImport	金鑰政策	否	KMS 金鑰	kms:WrappingAlgorithm kms:WrappingKeySpec KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
GetPublicKey kms:GetPublicKey	金鑰政策	是	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService 其他條件： kms:RequestAlias

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
ImportKeyMaterial kms:ImportKeyMaterial	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService 其他條件： kms:ExpirationModel kms:ValidTo
ListAliases kms:ListAliases	IAM 政策	否	*	無

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
ListGrants kms:ListGrants	金鑰政策	是	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService 其他條件： kms:GrantIsForAWSResource

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
ListKeyPolicies kms:ListKeyPolicies	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
ListKeyRotations kms:ListKeyRotations	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService
ListKeys kms:ListKeys	IAM 政策	否	*	無

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
ListResourceTags kms:ListResourceTags	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService
ListRetirableGrants kms:ListRetirableGrants	IAM 政策	指定的主體 必須位於本 機帳戶中， 但操作會在 所有帳戶中 傳回授予。	*	無

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
PutKeyPolicy kms:PutKeyPolicy	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService 其他條件： kms:BypassPolicyLockoutSafetyCheck

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
ReEncrypt kms:ReEncryptFrom kms:ReEncryptTo 若要使用此操作時，呼叫者需要兩個 KMS 金鑰上的許可： - KMS 金鑰上的 kms:ReEncryptFrom，用於解密 - KMS 金鑰上的 kms:ReEncryptTo，用於加密	金鑰政策	是	KMS 金鑰	密碼編譯操作的條件 kms:EncryptionAlgorithm kms:RequestAlias 加密內容條件： kms:EncryptionContext:context-key kms:EncryptionContextKeys KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws:ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService 其他條件：

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
				kms:ReEncryptOnSameKey
ReplicateKey kms:ReplicateKey 若要使用此操作時，呼叫者需要以下許可： - 多區域主要金鑰上的 kms:ReplicateKey - 複本區域中 IAM 政策的 kms:CreateKey	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService 其他條件： kms:ReplicaRegion

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
RetireGrant <code>kms:RetireGrant</code> 淘汰授予的許可主要取決於授予。單獨的政策無法允許存取此操作。如需詳細資訊，請參閱淘汰和撤銷授予。	IAM 政策 (此許可可在主要政策中無效。)	是	KMS 金鑰	加密內容條件： kms:EncryptionContext:context-key kms:EncryptionContext:Keys 授予條件： kms:GrantConstraintType KMS 金鑰操作的條件： kms:CallerAccount kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
RevokeGrant kms:RevokeGrant	金鑰政策	是	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService 其他條件： kms:GrantIsForAWSResource

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
RotateKeyOnDemand kms:RotateKeyOnDemand and	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
ScheduleKeyDeletion kms:ScheduleKeyDeletion	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
符號 kms:Sign	金鑰政策	是	KMS 金鑰	簽署和驗證的條件： kms:MessageType kms:RequestAlias kms:SigningAlgorithm KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
TagResource kms:TagResource	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService 標記的條件： aws : RequestTag/tag-key (AWS 全域條件金鑰) aws : TagKeys (AWS 全域條件金鑰)

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
UntagResource kms:UntagResource	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService 標記的條件： aws : RequestTag/tag-key (AWS 全域條件金鑰) aws : TagKeys (AWS 全域條件金鑰)

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
UpdateAlias kms:UpdateAlias 若要使用此操作時，呼叫者需要三個資源上的 kms:UpdateAlias 許可： - 別名 - 目前關聯的 KMS 金鑰 - 新關聯的 KMS 金鑰 如需詳細資訊，請參閱 控制對別名的存取 。	IAM 政策 (適用於別名) 金鑰政策 (適用於 KMS 金鑰)	否 否	Alias (別名) KMS 金鑰	無 (控制對別名的存取時) KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService
UpdateCustomKeyStore kms:UpdateCustomKeyStore	IAM 政策	否	*	kms:CallerAccount

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
UpdateKeyDescription kms:UpdateKeyDescription	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
UpdatePrimaryRegion kms:UpdatePrimaryRegion 若要使用此操作時，呼叫者需要針對會成為複本金鑰之 多區域主要金鑰 和會成為主要金鑰之 多區域複本金鑰 的 kms:UpdatePrimaryRegion 許可。	金鑰政策	否	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService 其他條件 kms:PrimaryRegion

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
確認 kms:Verify	金鑰政策	是	KMS 金鑰	簽署和驗證的條件： kms:MessageType kms:RequestAlias kms:SigningAlgorithm KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService

動作與許可	Policy type (政策類型)	跨帳戶使用	資源 (適用於 IAM 政策)	AWS KMS 條件索引鍵
VerifyMac kms:VerifyMac	金鑰政策	是	KMS 金鑰	KMS 金鑰操作的條件： kms:CallerAccount kms:KeySpec kms:KeyUsage kms:KeyOrigin kms:MultiRegion kms:MultiRegionKeyType kms:ResourceAliases aws : ResourceTag/tag-key (AWS 全域條件金鑰) kms:ViaService 密碼編譯操作的條件： kms:MacAlgorithm kms:RequestAlias

資料欄描述

此資料表中的資料欄提供下列資訊：

- 動作和許可會列出每個 AWS KMS API 操作，以及允許操作的許可。您可以在政策陳述式的 Action 元素中指定操作。
- 政策類型指出許可可用於金鑰政策還是 IAM 政策。

金鑰政策表示您可以在金鑰政策中指定許可。當金鑰政策包含[啟用 IAM 政策的政策陳述式](#)時，您可以在 IAM 政策中指定許可。

IAM 政策表示您只能在 IAM 政策中指定許可。

- 跨帳戶使用顯示授權使用者可以對不同 AWS 帳戶中資源執行的操作。

值為是表示主體可以對不同 AWS 帳戶中的資源執行操作。

值為否表示主體僅可對其自己 AWS 帳戶中的資源執行操作。

如果您為不同帳戶中的主體授予無法在跨帳戶資源上使用的許可，則許可無效。例如，如果您為不同帳戶的主體提供您帳戶中 KMS 金鑰的 [kms:TagResource](#) 許可，則其在您帳戶中標記 KMS 金鑰的嘗試將會失敗。

- 資源列出許可適用的 AWS KMS 資源。AWS KMS 支援兩種資源類型：KMS 金鑰和別名。在金鑰政策中，Resource 元素的值永遠是 *，這指的是 KMS 金鑰所連接的金鑰政策。

使用下列值來代表 IAM 政策中的 AWS KMS 資源。

KMS 金鑰

當資源為 KMS 金鑰時，請使用其[金鑰 ARN](#)。如需協助，請參閱 [the section called “尋找金鑰 ID 和金鑰 ARN”](#)。

`arn:AWS_partition_name:kms:AWS_Region:AWS_account_ID:key/key_ID`

例如：

`arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab`

Alias (別名)

當資源為別名時，請使用其[別名 ARN](#)。如需協助，請參閱 [the section called “尋找別名名稱和別名 ARN”](#)。

`arn:AWS_partition_name:kms:AWS_region:AWS_account_ID:alias/alias_name`

例如：

`arn:aws:kms:us-west-2:111122223333:alias/ExampleAlias`

* (星號)

當許可不適用於特定資源 (KMS 金鑰或別名) 時，請使用星號 (*)。

在 AWS KMS 許可的 IAM 政策中，Resource 元素中的星號表示所有 AWS KMS 資源 (KMS 金鑰和別名)。當 AWS KMS 許可不適用於任何特定 KMS 金鑰或別名時，您也可以 Resource 元素中使用星號。例如，允許或拒絕 kms:CreateKey 或 kms:ListKeys 許可時，您必須將 Resource 元素設定為 *。

- AWS KMS 條件索引鍵列出您可以用來控制操作存取 AWS KMS 的條件索引鍵。您可以在政策的 Condition 元素中指定條件。如需詳細資訊，請參閱[AWS KMS 條件索引鍵](#)。此欄也包含支援的[AWS 全域條件索引鍵](#) AWS KMS，但並非所有服務都 AWS 支援。

AWS KMS 內部操作

AWS Key Management Service (AWS KMS) 提供由 [FIPS 140-3 安全層級 3 驗證的雲端硬體安全模組 \(HSM\)](#) 所保護的密碼編譯金鑰和操作。多個 AWS 雲端服務會使用 AWS KMS 金鑰和功能，而且您可以使用它們來保護應用程式中的資料。此技術指南提供在您使用 AWS 時於內執行的密碼編譯操作的詳細資訊 AWS KMS。

AWS KMS 內部需要為全域分散式金鑰管理服務擴展和保護 HSMs。

主題

- [網域和網域狀態](#)
- [內部通訊安全](#)
- [多區域金鑰的複寫程序](#)
- [耐久性保護](#)

網域和網域狀態

內的信任內部 AWS KMS 實體合作集合 AWS 區域 稱為網域。網域包含一組可信任的實體、一組規則，以及一組稱為網域金鑰的機密金鑰。網域金鑰會在屬於網域成員的 HSM 之間共用。網域狀態由以下欄位組成。

名稱

用來識別此網域的網域名稱。

成員

屬於網域成員的 HSM 清單，包括其公有簽署金鑰和公有協定金鑰。

電信業者

代表此服務運算子的實體、公有簽署金鑰和角色 (AWS KMS 運算子或服務主機) 清單。

Rules

必須滿足每個命令才能在 HSM 上執行命令的仲裁規則清單。

網域金鑰

目前在網域中使用的網域金鑰 (對稱金鑰) 清單。

完整網域狀態僅適用於 HSM。HSM 網域成員之間的網域狀態會同步為匯出的網域字符。

網域金鑰

網域中的所有 HSM 共用一組網域金鑰， $\{DK_r\}$ 。這些金鑰會透過網域狀態匯出常式共用。匯出的網域狀態可以匯入至屬於網域成員的任何 HSM。

一組網域金鑰 $\{DK_r\}$ 一律包含一個作用中的網域金鑰，以及數個已停用的網域金鑰。網域金鑰會每天輪換，以確保 AWS 符合[金鑰管理的建議 - 第 1 部分](#)。在網域金鑰輪換期間，所有在外寄網域金鑰下加密的現有 KMS 金鑰都會在新的作用中網域金鑰下重新加密。作用中的網域金鑰可用來加密任何新的 EKT。在等同於最近輪換網域金鑰次數的天數內，過期的網域金鑰只能用來解密先前加密的 EKT。

已匯出網域字符

需要定期同步處理網域參與者之間的狀態。這是透過每當變更網域時匯出網域狀態來完成的。網域狀態會匯出為匯出的網域字符。

名稱

用來識別此網域的網域名稱。

成員

屬於網域成員的 HSM 清單，包括其簽署和協定公有金鑰。

運算子

實體、公有簽署金鑰以及代表此服務電信業者的角色清單。

Rules

必須滿足每個命令才能在 HSM 網域成員上執行命令的仲裁規則清單。

已加密的網域金鑰

信封加密的網域金鑰。網域金鑰會由上述每個成員的簽署成員進行加密，包裹在其公有協定金鑰中。

簽章

HSM 產生之網域狀態的簽章，必定是匯出網域狀態的網域成員。

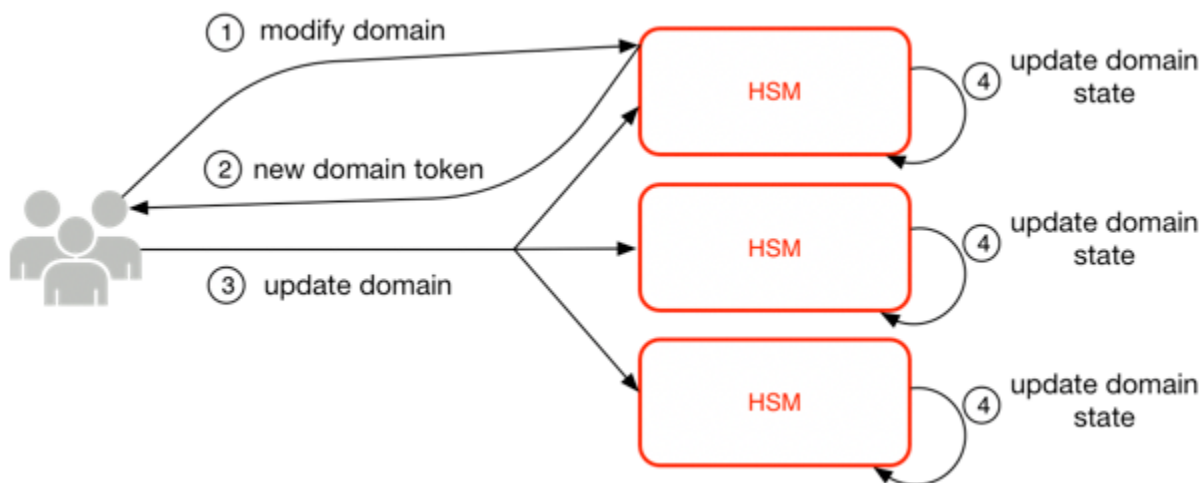
匯出的網域字符形成信任的基礎來源，用於網域內的實體操作。

管理網域狀態

網域狀態是透過仲裁驗證的命令進行管理。這些變更包括修改網域中可信任參與者的清單、修改執行 HSM 命令的仲裁規則，以及定期輪換網域金鑰。這些命令會以每個命令為基礎進行驗證，與已驗證工作階段操作完全不同 (如下圖所示)。

處於初始化和操作狀態的 HSM 包含一組自行產生的非對稱身分金鑰、簽署金鑰對，以及金鑰建立的金鑰對。透過手動程序，AWS KMS 運算子可以建立初始網域，以在區域中的第一個 HSM 上建立。此初始網域包含本主題先前定義的完整網域狀態。它是透過加入命令安裝至網域中每個已定義的 HSM 成員。

HSM 加入初始網域後，它會繫結至該網域中定義的規則。這些規則管控使用客戶密碼編譯金鑰或變更主機或網域狀態的命令。先前已定義之使用密碼編譯金鑰的已驗證工作階段 API 操作。



上述圖片描述了如何修改網域狀態。此程序包含四個步驟：

1. 以仲裁為基礎的命令會傳送至 HSM 以修改網域。
2. 系統會產生新的網域狀態，並匯出為新的匯出網域字符。系統不會修改 HSM 上的狀態，這表示該變更不會實際用於 HSM。
3. 第二個命令會傳送至新匯出的網域字符中的每個 HSM，以使用新的網域字符來更新其網域狀態。
4. 列在新匯出的網域字符中的 HSM 可以驗證命令和網域字符。其也可以解壓縮網域金鑰，以更新網域中所有 HSM 的網域狀態。

HSM 不會直接彼此通訊。相反地，電信業者的仲裁會請求變更網域狀態，從而產生新的匯出網域字符。網域的服務主機成員可用來將新網域狀態分配至網域中的每個 HSM。

離開和加入網域是透過 HSM 管理功能完成的。網域狀態的修改是透過網域管理功能完成的。

離開網域

讓 HSM 離開網域，從記憶體中刪除該網域的所有剩餘部分和金鑰。

加入網域

讓 HSM 加入新網域，或將其目前的網域狀態更新為新網域狀態。現有的網域會用作初始規則集的來源，以對此訊息進行驗證。

建立網域

在 HSM 上建立新網域。傳回可分配給網域成員 HSM 的第一個網域字符。

修改電信業者

在網域中新增或移除授權電信業者及其角色之清單中的電信業者。

修改成員

在網域中新增或移除已授權 HSM 清單中的 HSM。

修改規則

修改在 HSM 上執行命令所需的仲裁規則集。

輪換網域金鑰

建立新的網域金鑰，並將其標示為作用中的網域金鑰。這會將現有的作用中金鑰轉為已停用金鑰，並從網域狀態中移除最舊的已停用金鑰。

內部通訊安全

服務主機或 AWS KMS 運算子與 HSMs 之間的命令會透過 [中描述的兩種機制進行保護](#)[已驗證的工作階段](#)：規定人數簽署的請求方法，以及使用 HSM 服務主機通訊協定的已驗證工作階段。

仲裁簽署命令的設計是為了讓任何單一電信業者都無法修改 HSM 提供的重要安全保護。在已驗證工作階段上執行的命令有助於確保只有授權的服務電信業者可以執行涉及 KMS 金鑰的操作。所有客戶繫結的秘密資訊都會在 AWS 基礎設施中受到保護。

金鑰建立

為了保護內部通訊的安全，AWS KMS 使用兩種不同的金鑰建立方法。第一個在[使用離散對數密碼編譯的成對金鑰建立結構描述建議](#)中定義為 C(1, 2, ECC DH)。此配置擁有搭配靜態簽署金鑰的啟動器。啟動器會產生並簽署暫時橢圓曲線 Diffie-Hellman (ECDH) 金鑰，用於具有靜態 ECDH 協定金鑰的收件人。此方法使用一個暫時金鑰和兩個的靜態金鑰 (使用 ECDH)。這是標籤 C(1, 2, ECC DH) 的衍生。此方法有時稱為單通 ECDH。

第二種金鑰建立方法是 [C\(2, 2, ECC, DH\)](#)。在這個配置中，雙方都有一個靜態簽署金鑰，其產生、簽署和交換暫時的 ECDH 金鑰。這個方法使用兩個靜態金鑰和兩個暫時金鑰，每個金鑰都使用 ECDH。這是標籤 C(2, 2, ECC, DH) 的衍生。這種方法有時被稱為 ECDH 暫時或 ECDHE。所有 ECDH 金鑰都會在曲線 secp384r1 (NIST-P384) 上產生。

HSM 安全邊界

的內部安全界限 AWS KMS 是 HSM。HSM 擁有專有界面，而且沒有其他處於操作狀態的作用中實體界面。在初始化期間，會使用必要的密碼編譯金鑰來佈建操作 HSM，以便在網域中建立其角色。HSM 的敏感密碼編譯材料只會存放在揮發性記憶體中，並在 HSM 移出操作狀態時清除，包括預期或非預期的關機或重設。

HSM API 操作會透過個別命令或透過服務主機建立的互相驗證機密工作階段進行驗證。

仲裁簽署的命令

配額簽署的命令是由運算子向 HSMs 發出。本節描述了如何建立、簽署和驗證以仲裁為基礎的命令。這些規則相當簡單。例如，命令 Foo 需要角色 Bar 的兩個成員，以進行驗證。建立和驗證以仲裁為基礎的命令有三個步驟。第一步是初始命令建立；第二步是提交給其他電信業者進行簽署；第三步是驗證和執行。

為了介紹概念，假設有一組真實的運算子公有金鑰和角色 $\{QOS_s\}$ ，以及一組規定人數規則 $QR = \{Command_i, Rule_{\{i, t\}}\}$ ，其中每個規則都是一組角色，且最小數量為 $N \{Role_t, N_t\}$ 。對於滿足仲裁規

則的命令，命令資料集必須由列於 $\{QOS_s\}$ 中的一組電信業者簽署，使其符合為該命令列出的規則之一。如前所述，仲裁規則和電信業者集會以網域狀態和匯出的網域字符存放。

實際上，初始簽署者會簽署命令 $Sig_1 = \text{Sign}(dO_{p1}, \text{Command})$ 。第二名電信業者也會簽署命令 $Sig_2 = \text{Sign}(dO_{p2}, \text{Command})$ 。雙重簽署的訊息會傳送至 HSM 執行。HSM 將執行以下內容：

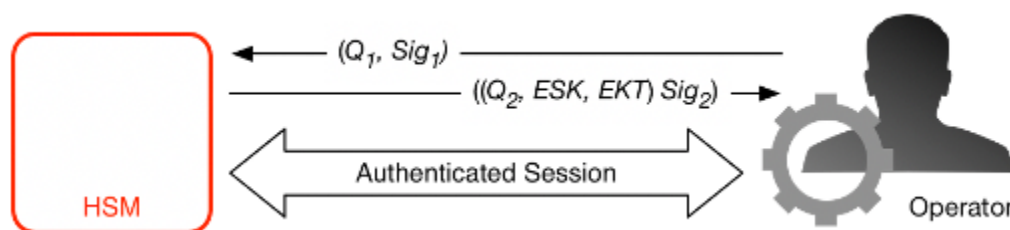
1. 對於每個簽章，它會從網域狀態擷取簽署者的公有金鑰，並驗證命令上的簽章。
2. 它會驗證該組簽署者是否滿足命令的規則。

已驗證的工作階段

您的金鑰操作會在面向外部的 AWS KMS 主機和 HSMs 之間執行。這些命令與密碼編譯金鑰的建立和使用以及安全的隨機數字產生有關。命令會在服務主機和 HSM 之間的工作階段驗證通道上執行。除了需要真實性之外，這些工作階段還需要機密性。在這些工作階段上執行的命令包括傳回純文字資料金鑰，以及為您提供的解密訊息。若要確保不會透過中間人攻擊推翻這些工作階段，則要驗證工作階段。

此通訊協定會在 HSM 與服務主機之間執行相互驗證的 ECDHE 金鑰協定。交換由服務主機啟動，並由 HSM 完成。HSM 也會傳回由交涉金鑰加密的工作階段金鑰 (SK)，以及包含工作階段金鑰的匯出金鑰字符。匯出的金鑰字符包含有效期間，之後服務主機必須重新交涉工作階段金鑰。

服務主機是網域的成員，並具有身分簽署金鑰對 ($dHOS_i$ 、 $QHOS_i$) 和 HSMs 身分公有金鑰的真實副本。它會使用其一組身分簽署金鑰來安全地交涉可在服務主機與網域中任何 HSM 之間使用的工作階段金鑰。匯出的金鑰字符具有與其相關聯的有效期間，之後必須交涉新的金鑰。



程序從服務主機辨識開始，它需要工作階段金鑰來傳送和接收本身與網域的 HSM 成員之間的敏感通訊流程。

1. 服務主機會產生 ECDH 暫時金鑰對 (d_1 , Q_1) 並使用其身分金鑰 $Sig1 = \text{Sign}(dOS, Q_1)$ 進行簽署。
2. HSM 會使用目前的網域字符來驗證已接收公有金鑰上的簽章，並建立 ECDH 暫時金鑰對 (d_2 , Q_2)。接著，它會根據使用離散對數密碼編譯對等金鑰建立結構的建議來完成 ECDH-key-exchange，以形成交涉的 256 位元 AES-GCM 金鑰。<https://csrc.nist.gov/pubs/sp/800/56/a/r3/final> HSM 會產生全新的 256 位元 AES-GCM 工作階段金鑰。它會使用交涉的金鑰加密工作階段金鑰，以形成加密的工

作階段金鑰 (ESK)。它還會將網域金鑰下的工作階段金鑰加密為匯出的金鑰字符 EKT。最後，它會用其身分金鑰對 $\text{Sig}_2 = \text{Sign}(\text{dHSK}, (Q_2, \text{ESK}, \text{EKT}))$ 簽署傳回值。

3. 服務主機會使用其目前的網域字符來驗證已接收金鑰上的簽章。然後，服務主機會根據[使用離散對數加密的成對金鑰建立結構描述的建議來完成 ECDH 金鑰](#)交換。接下來會解密 ESK 以取得工作階段金鑰 SK。

在 EKT 有效期間，服務主機可以使用交涉的工作階段金鑰 SK，將信封加密的命名傳送至 HSM。透過此驗證的工作階段，每個服務主機啟動的命令都會包含 EKT。HSM 會使用相同交涉的工作階段金鑰 SK 進行回應。

多區域金鑰的複寫程序

AWS KMS 使用跨區域複寫機制，將 KMS 金鑰中的金鑰材料從一個 HSM 複製到另一個 AWS 區域 HSM AWS 區域。若要讓此機制正常運作，正在複寫的 KMS 金鑰必須是具有 AWS_KMS 原始伺服器的多區域金鑰。對於具有匯入金鑰材料 (EXTERNAL 原始伺服器) 的多區域金鑰，您必須負責透過將相同的金鑰材料個別匯入每個複本來複寫金鑰材料。

將 KMS 金鑰從一個區域複寫到另一個區域時，區域中 HSMs 無法直接通訊，因為它們位於隔離的網路中。相反地，跨區域複寫期間交換的訊息是由 Proxy 服務負責傳遞。在跨區域複寫期間，AWS KMS HSM 產生的每則訊息都會使用複寫簽署金鑰進行密碼編譯簽署。複寫簽署金鑰 (RSK) 是 NIST P-384 曲線上的 ECDSA 金鑰。每個區域至少擁有一個 RSK，且每個 RSK 的公有元件會與相同 AWS 分割區中的所有其他區域共用。

將金鑰材料從區域 A 複製到區域 B 的跨區域複寫處理的運作方式如下：

1. 區域 B 中的 HSM 會在 NIST P-384 曲線上產生一個短暫的 ECDH 金鑰，複寫協議金鑰 B (RAKB)。RAKB 的公有元件由代理服務傳送至區域 A 中的 HSM。
2. 區域 A 中的 HSM 接收 RAKB 的公有元件，然後在 NIST P-384 曲線上產生另一個暫時的 ECDH 金鑰，複寫協議金鑰 A (RAKA)。HSM 會在 RAKA 和 RAKB 的公有元件上執行 ECDH 金鑰建立配置，並從輸出衍生對稱金鑰複寫包裝金鑰 (RWK)。RWK 是用來加密正在複寫的多區域 KMS 金鑰的金鑰材料。
3. RAKA 的公有元件和使用 RWK 加密的金鑰材料會透過代理服務傳送至區域 B 的 HSM。
4. 區域 B 中的 HSM 會接收 RAKA 的公有元件，以及使用 RWK 加密的金鑰材料。HSM 由 RWK 在 RAKB 及 RAKA 的公有元件上執行 ECDH 金鑰建立配置而衍生。
5. 區域 B 中的 HSM 會使用 RWK 來解密區域 A 中的金鑰材料。

耐久性保護

服務所產生之金鑰的其他服務耐久性由離線 HSM 的使用、匯出網域字符的多個非揮發性儲存體以及加密 KMS 金鑰的備援儲存體提供。離線 HSM 是現有網域的成員。除了沒有上線和參與一般網域操作之外，離線 HSM 在網域狀態中顯示與現有 HSM 成員相同。

耐用性設計旨在保護區域中的所有 KMS 金鑰，應該 AWS 會遇到線上 HSMs 或存放在主要儲存系統中的一組 KMS 金鑰的大規模遺失。AWS KMS keys 具有匯入金鑰材料的 不會包含在提供其他 KMS 金鑰的耐用性保護下。如果 發生全區域故障 AWS KMS，匯入的金鑰材料可能需要重新匯入 KMS 金鑰。

離線 HSM 以及對其進行存取的憑證，都會存放在多個獨立地理位置上受監控之安全室內的保險箱內。每個保險箱至少需要一個 AWS 安全主管和一個 AWS KMS 操作員，來自兩個獨立的團隊 AWS，才能取得這些資料。這些資料的使用由內部政策管理，要求存在規定人數的 AWS KMS 運算子。

文件歷史紀錄

此主題描述《AWS Key Management Service 開發人員指南》的重大更新。

主題

- [最近更新](#)
- [舊版更新](#)

最近更新

下表說明此文件自 2018 年 1 月後的重大變更。除了這裡所列的主要變更外，我們也會經常更新文件以改進說明內容和範例，並且反映您傳送給我們的意見回饋。如要接收重大變更的通知，請訂閱 RSS 摘要。

您可能需要水平或垂直捲動，才能查看此資料表中的所有資料。

變更	描述	日期
功能更新	AWS KMS 支援隨需輪換具有匯入金鑰材料的對稱加密多區域金鑰。	2025 年 11 月 26 日
功能更新	AWS KMS 支援 CloudTrail 事件 <code>tlsDetails</code> 區段中的新 <code>keyExchange</code> 欄位。	2025 年 11 月 24 日
功能更新	AWS KMS 支援新的 KeySpec <code>ECC_NIST_EDWARDS25519</code> 。	2025 年 11 月 7 日
雙堆疊支援	AWS KMS 支援雙堆疊。	2025 年 6 月 18 日
功能更新	新增對 Module-Lattice 數位簽章演算法 (ML-DSA) 量子後密碼編譯簽章的支援。	2025 年 6 月 13 日
匯入的金鑰輪換	您可以使用匯入的金鑰材料 (EXTERNAL 原始伺服器) 執行	2025 年 6 月 5 日

對稱加密 KMS 金鑰的隨需輪換。

[功能更新](#)

新增對中國區域中多區域 KMS 金鑰的支援。

2024 年 11 月 21 日

[AWS 受管政策更新](#)

更新 AWSKeyManagementServiceMultiRegionKeysServiceRolePolicy 服務連結角色，方法是將陳述式 ID (Sid) 新增至政策版本 v2 的受管政策。

2024 年 11 月 21 日

[配額變更](#)

提高 [ImportKeyMaterial](#) 和 [DeleteImportedKeyMaterial](#) 請求的預設請求率。

2024 年 7 月 23 日

[配額變更](#)

提高對稱加密 KMS 金鑰、RSA KMS 金鑰和 ECC 和 SM2 KMS 金鑰的預設密碼編譯操作請求率。

2024 年 7 月 8 日

[新功能](#)

新增KEY_AGREEMENT 了 NIST 建議的橢圓曲線 (ECC) 和 SM2 (僅限中國區域) KMS 金鑰的新KeyUsage類型，並新增[了衍生共用秘密](#)的支援。

2024 年 6 月 13 日

[金鑰輪換的更新](#)

新增對自動金鑰輪換、隨需金鑰輪換的自訂輪換期間支援，以及對金鑰材料輪換的可見性。

2024 年 4 月 12 日

受管政策的更新項目	新增了許可AWSKeyManagementServiceCustomKeyStoresServiceRolePolicy，AWS KMS 允許 監控包含 AWS CloudHSM 叢集的 VPC 中的變更，以便在發生故障時 AWS KMS 提供明確的錯誤訊息。	2023 年 11 月 10 日
功能更新	已新增對 DryRun API 參數的支援。	2023 年 7 月 5 日
功能更新	新增支援匯入所有類型金鑰的 AWS KMS 金鑰材料，但自訂金鑰存放區除外。	2023 年 6 月 5 日
功能更新	Nitro Enclaves AWS KMS APIs 更新	2023 年 3 月 10 日
功能更新	RSAES_PKCS1_V1_5 包裝演算法已棄用。AWS KMS 將根據國家標準技術研究所 (NIST) 的密碼編譯金鑰管理指引 ，在 2023 年 10 月 1 RSAES_PKCS1_V1_5 日之前結束對的所有支援。建議您立即開始採用不同包裝演算法。	2023 年 2 月 28 日
功能更新	新增對外部金鑰存放區的支援，這項功能可讓您在 外部使用密碼編譯金鑰來保護 資源 AWS AWS。	2022 年 11 月 29 日
配額變更	將 AWS KMS keys 資源配額提高到每個帳戶和區域中 100,000 個 KMS 金鑰。	2022 年 7 月 8 日

功能更新	在更多 中新增對 HMAC KMS 金鑰的支援 AWS 區域	2022 年 7 月 8 日
新主題	將 AWS Key Management Service 主題 中的 彈性 新增至 AWS KMS 開發人員指南的安全章節。	2022 年 6 月 14 日
新功能	新增對產生和驗證 HMAC 程式碼之 AWS KMS 金鑰和 API 操作的支援。	2022 年 4 月 19 日
文件變更	使用 AWS KMS key 和 KMS 金鑰 取代術語客戶主要金鑰 (CMK)。	2021 年 8 月 30 日
新功能	新增對 多區域金鑰 的支援，這是一組在不同區域中具有相同金鑰 ID 和金鑰材料的可互通 KMS 金鑰。您可以使用多區域金鑰在一個區域中加密資料，並在其他區域中解密資料。	2021 年 6 月 8 日
新功能	已新增對屬性型存取控制 (ABAC) 的支援。您可以使用標籤和別名來控制對 的存取 AWS KMS keys。	2020 年 12 月 17 日
新功能	已新增對 VPC 端點政策的支援。	2020 年 7 月 9 日
新內容	說明 的安全屬性 AWS KMS。	2020 年 6 月 18 日
新功能	新增對非對稱 AWS KMS keys 和非對稱資料金鑰的支援。	2019 年 11 月 25 日

更新的功能	您可以在 AWS KMS 主控台 AWS 受管金鑰 中檢視 的金鑰政策。此功能過去僅限於客戶受管金鑰。	2019 年 11 月 15 日
新功能	說明如何將 TLS 中的 混合式後量子金鑰交換 演算法用於對 AWS KMS 的呼叫。	2019 年 11 月 4 日
配額變更	提高管理 KMS 金鑰之部分 API 的資源配額。	2019 年 9 月 18 日
配額變更	變更每個 KMS 金鑰之 KMS 金鑰、別名和授予的資源配額。	2019 年 3 月 27 日
配額變更	對於在自訂金鑰存放區中使用 AWS KMS keys 的加密操作，已變更共用的每秒請求配額。	2019 年 3 月 7 日
新功能	說明如何建立和管理 AWS KMS 自訂金鑰存放區 。每個金鑰存放區都由您擁有和控制的 AWS CloudHSM 叢集提供支援。	2018 年 11 月 26 日
新主控台	說明如何使用獨立於 IAM AWS KMS 主控台的新主控台。短期間內仍會保留原始主控台及其使用指示，以利於您熟悉新的主控台。	2018 年 11 月 7 日
配額變更	已變更共用 請求配額 以供使用 AWS KMS keys。	2018 年 8 月 21 日
新內容	說明如何 AWS Secrets Manager 使用 AWS KMS 金鑰來加密秘密中的秘密值。	2018 年 7 月 13 日

[新內容](#)

說明 DynamoDB 如何使用 AWS KMS AWS KMS keys 支援其伺服器端加密選項。

2018 年 5 月 23 日

[新功能](#)

說明如何[在 VPC 中使用私有端點](#)直接連線 AWS KMS，而不是透過網際網路連線。

2018 年 1 月 22 日

舊版更新

下表說明 2018 年之前 AWS Key Management Service 開發人員指南的重要變更。

您可能需要水平或垂直捲動，才能查看此資料表中的所有資料。

變更	描述	Date
新內容	已新增 中的標籤 AWS KMS 的相關文件。	2017 年 2 月 15 日
新內容	已新增 監控 AWS KMS keys 和 使用 Amazon CloudWatch 監控 KMS 金鑰 的相關文件。	2016 年 8 月 31 日
新內容	已新增 匯入的金鑰資料 的相關文件。	2016 年 8 月 11 日
新內容	已新增下列文件： IAM 政策 、 許可參考 和 條件索引鍵 。	2016 年 7 月 5 日
更新	更新 KMS 金鑰存取和許可 章節的文件部分。	2016 年 7 月 5 日
更新	更新 配額 頁面，以反映新的預設配額。	2016 年 5 月 31 日
更新	已更新 配額 頁面，以反映新的預設配額，並且已更新 授予字	2016 年 4 月 11 日

變更	描述	Date
	符 文件，以改善清晰度和準確性。	
新內容	已新增 允許多個 IAM 主體存取 KMS 金鑰 和 使用 IP 地址條件 的相關文件。	2016 年 2 月 17 日
更新	更新 中的金鑰政策 AWS KMS 和 變更金鑰政策 頁面，以改善清晰度和準確性。	2016 年 2 月 17 日
更新	更新管理 KMS 金鑰主題頁面以改善清晰度。	2016 年 1 月 5 日
新內容	新增有關 CloudTrail 的文件。	2015 年 11 月 18 日
新內容	新增 變更金鑰政策 的說明。	2015 年 11 月 18 日
更新	更新 Amazon Relational Database Service 如何使用的文件 AWS KMS。	2015 年 11 月 18 日
新內容	新增有關 Amazon WorkSpaces 的文件。	2015 年 11 月 6 日
更新	更新 中的金鑰政策 AWS KMS 頁面，以改善清晰度。	2015 年 10 月 22 日
新內容	新增 刪除 AWS KMS key 的相關文件，包含 建立警示 和 判斷 KMS 金鑰的過去用量 的相關支援文件。	2015 年 10 月 15 日
新內容	已新增 判斷對 的存取 AWS KMS keys 的相關文件。	2015 年 10 月 15 日
新內容	已新增 金鑰的 AWS KMS 金鑰狀態 的相關文件。	2015 年 10 月 15 日

變更	描述	Date
新內容	新增有關 Amazon Simple Email Service 的文件。	2015 年 10 月 1 日
更新	已更新 配額 頁面以說明新的請求配額。	2015 年 8 月 31 日
新內容	新增使用 之費用的相關資訊 AWS KMS。請參閱 AWS KMS 定價 。	2015 年 8 月 14 日
新內容	已將請求配額新增至 AWS KMS 配額 。	2015 年 6 月 11 日
新內容	已新增示範使用 UpdateAlias 操作的新 Java 程式碼範例。	2015 年 6 月 1 日
更新	將 AWS Key Management Service 區域表格 移動到 AWS 一般參考。	2015 年 5 月 29 日
新內容	已新增 Amazon EMR 如何使用 AWS KMS 的相關文件。	2015 年 1 月 28 日
新內容	新增有關 Amazon WorkMail 的文件。	2015 年 1 月 28 日
新內容	新增 Amazon Relational Database Service 如何使用的文件 AWS KMS。	2015 年 1 月 6 日
新內容	新增有關 Amazon Elastic Transcoder 的文件。	2014 年 11 月 24 日
新指南	已介紹《AWS Key Management Service 開發人員指南》。	2014 年 11 月 12 日

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。