



사용 설명서

AWS CloudFormation



API 버전 2010-05-15

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

AWS CloudFormation: 사용 설명서

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

Amazon의 상표 및 브랜드 디자인은 Amazon 외 제품 또는 서비스와 함께, Amazon 브랜드 이미지를 떨어뜨리거나 고객에게 혼동을 일으킬 수 있는 방식으로 사용할 수 없습니다. Amazon이 소유하지 않은 기타 모든 상표는 Amazon과 제휴 관계이거나 관련이 있거나 후원 관계와 관계없이 해당 소유자의 자산입니다.

Table of Contents

CloudFormation란 무엇인가요?	1
인프라 관리 간소화	1
신속하게 인프라 복제	1
인프라 변경 사항을 쉽게 제어 및 추적	2
CloudFormation 시작하기	2
관련 정보	2
시작	3
CloudFormation 작동 방식	3
주요 개념	3
CloudFormation 작동 방식	7
CloudFormation을 시작하는 방법	10
AWS 계정에 가입	10
AWS 계정에 가입	10
관리자 액세스 권한이 있는 사용자 생성	11
첫 번째 스택 생성	12
사전 조건	13
콘솔을 사용하여 CloudFormation 스택 생성	13
스택 생성 모니터링	17
웹 서버 테스트	18
문제 해결	18
정리	19
다음 단계	20
모범 사례	22
피드백 루프를 단축하여 개발 속도 개선	23
수명 주기 및 소유권별로 스택 구성	24
교차 스택 참조를 사용하여 다른 스택에서 내보낸 출력 값 반환	24
다중 계정 및 다중 리전 배포에 CloudFormation StackSets 사용	25
모든 리소스 유형에 대한 할당량 확인	25
템플릿을 재사용하여 다양한 환경에서 스택 복제	25
모듈을 사용하여 리소스 구성 재사용	26
코드형 인프라 관행 채택	26
템플릿에 자격 증명을 포함하지 않음	26
AWS 특정 파라미터 유형 사용	27
파라미터 제약 조건 사용	27

유사 파라미터를 사용하여 이식성 향상	27
AWS::CloudFormation::Init를 사용하여 Amazon EC2 인스턴스에서 소프트웨어 애플리케이션 배포	28
최신 헬퍼 스크립트 사용	29
템플릿을 사용하기 전에 템플릿 확인	29
조직 정책 준수를 위한 템플릿 확인	29
템플릿 작성에 YAML 또는 JSON 사용	30
포괄적인 태그 지정 전략 구현	30
고급 변환에 템플릿 매크로를 활용	31
CloudFormation을 통해 모든 스택 리소스 관리	31
스택을 업데이트하기 전에 변경 세트 생성	31
스택 정책을 사용하여 리소스 보호	32
CloudFormation 호출 기록에 AWS CloudTrail 사용	32
코드 검토 및 버전 관리를 사용하여 사용자 템플릿 관리	32
정기적으로 Amazon EC2 인스턴스 업데이트	32
드리프트 감지를 정기적으로 사용	32
자동 복구를 위한 롤백 트리거 구성	33
효과적인 스택 리팩터링 전략 구현	33
수명 주기 관리에 CloudFormation Hooks 사용	34
IaC 생성기를 사용하여 기존 리소스에서 템플릿 생성	34
시각적 템플릿 설계에 AWS 인프라 컴포저 사용	34
복잡한 인프라에 AWS Cloud Development Kit (AWS CDK) 사용을 고려	35
IAM을 사용하여 액세스 제어	35
최소 권한 원칙을 적용	36
민감한 파라미터를 보호	36
AWS CloudFormation Guard를 사용하여 코드형 정책 구현	36
템플릿 사용	37
템플릿이 저장되는 위치	38
템플릿 유효성 검사	38
템플릿 시작하기	2
CloudFormation 템플릿 참조 사용 계획	39
샘플 템플릿	40
템플릿 형식	40
템플릿 구조	40
설명	42
사양	43

자세히 알아보기	44
정규식 사용	44
템플릿 섹션	45
Resources	46
Parameters	55
Outputs	67
Mappings	73
Metadata	80
Rules	86
Conditions	106
Transform	118
포맷 버전	119
Description	119
Infrastructure Composer	120
CloudFormation 콘솔 모드에서 Infrastructure Composer를 사용하는 이유는 무엇인가요? ...	120
이 모드는 Infrastructure Composer 콘솔과 어떻게 다른가요?	120
AWS CloudFormation 언어 서버	121
AWS CloudFormation 언어 서버와 통합되는 IDE	121
시작하기	122
IDE에서 CloudFormation 프로젝트 초기화	128
오픈 소스	129
도움이 필요하세요?	129
IaC 생성기	129
고려 사항	130
권한	131
자주 사용되는 명령	131
AWS CDK로 템플릿 마이그레이션	132
리소스 스캔 시작	132
스캔 요약 보기	136
스캔한 리소스에서 템플릿 생성	137
스캔한 리소스에서 스택 생성	142
쓰기 전용 속성 해결	143
다른 서비스에 저장된 값 가져오기	148
일반적인 고려 사항	148
Systems Manager 일반 텍스트 값 가져오기	149
Systems Manager 보안 문자열 가져오기	153

Secrets Manager 값 가져오기	156
AWS 값 가져오기	160
구문	160
사용 가능한 의사 파라미터	161
예제	163
스택 출력 가져오기	167
스택 출력값 내보내기과 중첩 스택 사용 비교	167
고려 사항	168
내보낸 출력값 나열	168
출력값을 가져오고 내보내는 스택 나열	169
런타임 시 기존 리소스 지정	170
개요	171
예제	172
고려 사항	175
지원되는 AWS 특정 파라미터 유형	176
지원되는 Systems Manager 파라미터 유형	177
지원되지 않는 Systems Manager 파라미터 유형	178
연습	178
다른 CloudFormation 스택의 리소스 출력 참조	179
Amazon EC2에 애플리케이션 배포	184
스택 업데이트	194
조정 및 로드 밸런싱된 애플리케이션 생성	210
다른 계정의 VPC와 피어링	223
ECS 블루/그린 배포 수행	234
템플릿 코드 조각	257
일반	258
Auto Scaling	268
AWS 빌링 콘솔	300
CloudFormation	303
CloudFront	310
CloudWatch	322
CloudWatch Logs	328
DynamoDB	362
Amazon EC2	366
Amazon ECS	403
Amazon EFS	426

Elastic Beanstalk	451
Elastic Load Balancing	456
IAM	461
AWS Lambda	478
Amazon Redshift	485
Amazon RDS	495
Route 53	503
Amazon S3	511
Amazon SNS	520
Amazon SQS	520
Amazon Timestream	521
Windows 기반 스택	550
Windows 스택 부트스트래핑	552
CloudFormation 제공 리소스 유형 사용	558
사용자 지정 리소스	560
템플릿 매크로	603
중첩 스택	619
대기 조건	626
모듈을 사용하여 재사용 가능한 리소스 구성 생성	632
모듈 사용 시 고려 사항	633
모듈 버전 관리 이해	633
템플릿에서 모듈 사용	635
파라미터를 사용하여 모듈 값 지정	636
모듈 리소스 참조	639
스택 생성 및 관리	641
스택 관리를 위한 인터페이스	642
스택을 만듭니다	643
스택 생성	643
스택 옵션 구성	646
스택 구성 미리 보기	648
스택 정보 보기	649
스택 템플릿 업데이트	652
스택 리소스의 업데이트 동작 이해	654
변경 세트를 사용하여 스택 업데이트	655
변경 세트 생성	657
변경 세트 보기	662

드리프트 인식 변경 세트	671
변경 세트 실행	676
변경 세트 삭제	678
변경 세트의 예	680
중첩 스택에 대한 변경 세트	692
스택 배포 검증	698
사전 배포 검증 작동 방식	699
고려 사항	699
사전 조건	700
스택 배포 검증(콘솔)	700
스택 배포 검증(AWS CLI)	701
검증 유형	702
리소스 제한 사항	703
직접 스택 업데이트	717
스택 업데이트 취소	720
스택 업데이트를 취소하려면 다음을 수행하세요(콘솔).	721
스택 업데이트를 취소하려면 다음을 수행하세요(AWS CLI).	721
스택 삭제	721
관련 리소스	723
삭제된 스택 보기	723
스택 진행 상황 모니터링	724
스택 이벤트 보기	724
작업별 스택 이벤트 보기	732
스택 배포 타임라인 그래프 보기	736
스택 생성 이벤트 이해	739
스택 업데이트 모니터링	741
업데이트 롤백 계속	743
스택 실패 원인 파악	746
스택 실패 옵션	747
경보 위반 시 스택 롤백	757
스택 생성 또는 업데이트 중에 롤백 트리거 추가	758
변경 세트에 롤백 트리거 추가	759
스택에 대한 롤백 트리거 보기	759
변경 세트에 대한 롤백 트리거 보기	759
스택 및 리소스에 대한 비관리형 구성 변경 감지	760
드리프트란 무엇인가요?	760

드리프트 감지 상태 코드	761
드리프트 감지 시 고려 사항	763
전체 CloudFormation 스택의 드리프트 감지	765
개별 스택 리소스의 드리프트 감지	771
가져오기 작업을 통한 드리프트 해결	775
AWS 리소스 가져오기	785
수동으로 AWS 리소스 가져오기	786
자동으로 AWS 리소스 가져오기	822
가져오기 작업 되돌리기	823
스택 리팩터링	829
스택 리팩터링 작동 방식	829
스택 리팩터링 고려 사항	830
사전 조건	830
스택 리팩터링(콘솔)	831
스택 리팩터링(AWS CLI)	832
리소스 제한 사항	836
리소스 유형 지원	841
빠른 생성 링크를 사용하여 CloudFormation 스택 생성	1028
URL 형식	1028
예제	1030
빠른 생성 링크를 사용하여 스택 생성	1030
AWS CLI 및 PowerShell에 대한 예제 명령	1031
CancelUpdateStack	1032
ContinueUpdateRollback	1033
CreateStack	1034
배포	1038
DeleteStack	1038
DescribeStackEvents	1040
DescribeStackResource	1042
DescribeStackResources	1043
DescribeStacks	1046
GetTemplate	1050
ListStackResources	1051
ListStacks	1053
UpdateStack	1055
ValidateTemplate	1059

S3 버킷에 로컬 아티팩트 업로드	1061
StackSets를 사용하여 스택 관리	1065
StackSets 개념	1066
관리자 및 대상 계정	1066
CloudFormation StackSets	1067
StackSet에 대한 권한 모델	1067
스택 인스턴스	1068
StackSet 작업	1068
StackSet 작업 옵션	1069
Tags	1071
StackSets 상태 코드	1071
스택 인스턴스 상태 코드	1072
사전 조건	1074
기본적으로 비활성화된 AWS 리전 활성화	1074
자체 관리형 권한 부여	1075
신뢰할 수 있는 액세스 활성화	1091
샘플 템플릿을 사용하여 시작하기	1095
사전 조건	1096
콘솔에서 샘플 템플릿을 사용하여 StackSet 생성	1097
StackSet 생성 모니터링	1099
StackSet 결과 보기	1099
StackSet 업데이트	1100
StackSet에 스택 추가	1101
정리	1101
다음 단계	1103
StackSets 생성(자체 관리형 권한)	1103
자체 관리형 권한으로 StackSet 생성(콘솔)	1104
자체 관리형 권한으로 StackSet 생성(AWS CLI)	1105
StackSets 생성(서비스 관리형 권한)	1107
고려 사항	1107
서비스 관리형 권한으로 StackSet 생성(콘솔)	1108
서비스 관리형 권한으로 StackSet 생성(AWS CLI)	1110
자동 배포 활성화 또는 비활성화	1114
자동 배포 작동 방식	1114
고려 사항	1114
자동 배포 활성화 또는 비활성화(콘솔)	1115

자동 배포 활성화 또는 비활성화(AWS CLI)	1115
StackSets 업데이트	1116
StackSet 업데이트(콘솔)	1116
StackSet 업데이트(AWS CLI)	1118
StackSets에 스택 추가	1120
StackSet에 스택 추가(콘솔)	1120
StackSet에 스택 추가(AWS CLI)	1122
스택에서 파라미터 재정의	1122
스택에서 파라미터 재정의(콘솔)	1123
스택에서 파라미터 재정의(AWS CLI)	1125
StackSets에서 스택 삭제	1126
StackSet에서 스택 삭제(콘솔)	1126
StackSet에서 스택 삭제(AWS CLI)	1128
StackSets 삭제	1129
StackSet 삭제(콘솔)	1129
StackSet 삭제(AWS CLI)	1130
서비스 역할 삭제(선택 사항)	1130
대상 계정 게이트	1131
요구 사항	1132
Lambda 함수 생성을 위한 CloudFormation 템플릿	1132
동시성 모드 선택	1133
각 동시성 모드의 작동 방식	1133
배포 속도에 따라 엄격한 내결함성과 소프트 내결함성 중에서 선택	1136
동시성 모드 선택(콘솔)	1138
동시성 모드 선택(AWS CLI)	1138
StackSets에서 드리프트 감지	1139
CloudFormation이 StackSet에서 드리프트 감지를 수행하는 방법	1139
StackSet에서 드리프트 감지(콘솔)	1140
StackSet에서 드리프트 감지(AWS CLI)	1141
StackSet에서 드리프트 감지 중지	1148
StackSet로 스택 가져오기	1149
자체 관리형 스택 가져오기	1149
서비스 관리형 스택 가져오기	1153
스택 가져오기 되돌리기	1158
모범 사례	1158
템플릿 정의	1158

스택 생성 또는 StackSet에 추가	1159
StackSet의 스택 업데이트	1159
샘플 템플릿	1160
문제 해결	1161
일반적인 스택 작업 실패 이유	1162
실패한 스택 생성 또는 업데이트 작업 재시도	1162
스택 인스턴스 삭제 실패	1163
스택 가져오기 작업 실패	1163
StackSets 작업에 대한 스택 인스턴스 실패 횟수	1164
Git 소스 코드와 스택 동기화	1170
Git 동기화 작동 방식	1171
Git 동기화 작동 방식	1171
풀 요청에 대한 의견	1172
스택 배포 파일	1172
CloudFormation 템플릿 파일	1173
템플릿 정의 리포지토리	1174
사전 조건	1174
Git 리포지토리	1174
CloudFormation 템플릿	1175
Git 동기화 서비스 역할	1175
콘솔 사용자의 IAM 권한	1178
리포지토리 소스 코드에서 스택 생성	1181
리포지토리 소스 코드에서 스택 생성	1181
Git 리포지토리에서 스택 업데이트	1184
풀 요청에서 주석 활성화	1184
상태 대시보드	1185
Git 동기화 상태	1185
최신 동기화 이벤트	1186
지원되는 스택 상태	1186
CloudFormation 레지스트리를 통해 익스텐션 관리	1190
관련 설명서	1190
개념	1191
확장 프로그램 유형	1191
퍼블릭 익스텐션	1192
활성화된 확장 프로그램	1192
사용 가능하고 활성화된 확장 프로그램 보기	1193

퍼블릭 익스텐션	1194
IAM 역할	1195
자동으로 새로운 버전의 확장 프로그램 사용	1197
별칭을 사용하여 확장 프로그램 참조	1197
공개 확장 프로그램 작업에 일반적으로 사용되는 AWS CLI 명령	1198
공개 확장 프로그램 활성화	1198
공개 확장 프로그램 업데이트	1202
공개 확장 프로그램 비활성화	1203
프라이빗 익스텐션	1205
IAM 권한	1205
비공개 확장 프로그램 작업에 일반적으로 사용되는 AWS CLI 명령	1205
프라이빗 익스텐션 등록	1206
비공개 확장 프로그램 등록 취소	1208
확장 프로그램에 대한 구성 데이터 편집	1210
동적 참조를 사용하는 데 필요한 권한	1210
확장 프로그램에 대한 구성 데이터 편집(콘솔)	1211
확장 프로그램에 대한 구성 데이터 편집(AWS CLI)	1212
AWS Config에서 리소스 유형 기록	1212
구성 항목에 민감한 속성이 기록되지 않도록 방지	1213
지속적 전달	1214
연습: 테스트 및 프로덕션 스택용 파이프라인 빌드	1214
사전 조건	1214
연습 개요	1215
1단계: 아티팩트 편집 및 S3 버킷에 업로드	1216
2단계: 파이프라인 스택 생성	1217
3단계: WordPress 스택 보기	1223
4단계: 리소스 정리	1223
다음 사항도 참조하세요.	1224
구성 속성 참조	1225
구성 속성(콘솔)	1225
구성 속성(JSON 객체)	1228
다음 사항도 참조하세요.	1232
CloudFormation 아티팩트	1233
스택 템플릿 파일	1233
템플릿 구성 파일	1233
다음 사항도 참조하세요.	1235

CodePipeline 파이프라인에서 파라미터 재정의 함수 사용	1235
Fn::GetArtifactAtt	1235
Fn::GetParam	1237
다음 사항도 참조하세요.	1239
템플릿 참조 가이드	1240
보안	1241
스택 삭제 방지	1242
스택에서 종료 방지 기능을 변경할 수 있는 대상 제어	1243
스택 리소스에 대한 업데이트 방지	1244
스택 정책 예제	1245
스택 정책 정의	1246
스택 정책 설정	1250
보호된 리소스 업데이트	1251
스택 정책 수정	1254
더 많은 스택 정책 예제	1255
데이터 보호	1258
저장 시 암호화	1259
전송 중 암호화	1259
인터넷워크 트래픽 개인 정보	1259
IAM을 통한 액세스 제어	1260
CloudFormation에 대한 IAM ID 기반 정책 정의	1260
CloudFormation 템플릿에서 IAM 리소스 승인	1268
Amazon EC2 인스턴스에서 실행되는 애플리케이션의 자격 증명 관리	1269
임시 액세스 권한 부여(페더레이션된 액세스)	1270
ID 기반 정책 예시	1271
AWS 관리형 정책	1279
CloudFormation 서비스 역할	1281
교차 서비스 혼동된 대리자 방지	1282
FAS 요청 및 권한 평가	1284
API 직접 호출 로깅	1286
CloudTrail의 CloudFormation 정보	1287
CloudFormation 로그 파일 항목 이해	1288
인프라 보안	1292
복원성	1292
규정 준수 확인	1293
구성 및 취약성 분석	1293

보안 모범 사례	1293
IAM을 사용하여 액세스 제어	1293
템플릿에 자격 증명을 포함하지 않음	1294
CloudFormation 호출 기록에 AWS CloudTrail 사용	1294
AWS PrivateLink	1295
CloudFormation VPC 엔드포인트 고려 사항	1295
CloudFormation에 대한 인터페이스 VPC 엔드포인트 생성	1296
CloudFormation에 대한 VPC 엔드포인트 정책 생성	1297
EventBridge로 모니터링	1298
CloudFormation과 Git 동기화 이벤트 개요	1299
권한	1300
사용자 지정 이벤트 패턴 생성	1301
이벤트 세부 정보 참조	1302
리소스 상태 변경	1303
스택 상태 변경	1305
드리프트 감지 상태 변경	1307
StackSet 상태 변경	1310
StackSet 스택 인스턴스 상태 변경	1311
StackSet 작업 상태 변경	1314
리포지토리 동기화 상태 변경	1316
리소스 동기화 상태 변경	1318
할당량	1322
기능 가용성	1328
StackSets 및 매크로	1329
문제 해결	1330
문제 해결 가이드	1330
오류 해결	1331
스택 삭제 실패	1331
종속성 오류	1332
AWS Config 및 AWS Systems Manager 충돌	1333
목록 전달 시 발생하는 파라미터 구분 분석 오류	1333
IAM 권한 부족	1333
잘못된 값 또는 지원되지 않는 리소스 속성	1334
할당량 초과	1334
중첩 스택 롤백 실패	1334
수행할 업데이트 없음	1335

스택 생성, 업데이트 또는 삭제 작업 중 리소스 안정화에 실패	1335
VPC에 보안 그룹이 없음	1336
업데이트 롤백에 실패	1336
대기 조건이 Amazon EC2 인스턴스에서 필요한 수의 신호를 수신하지 못함	1337
리소스가 스택에서 제거되었지만 삭제되지 않음	1338
Support에 문의	1339
Amazon Q로 문제 해결	1340
특성	1340
작동 방식	1340
가격 책정	1341
리전별 가용성	1341
제한 사항	1341
사용 설명서 기록	1342
보관된 업데이트	1363

CloudFormation이란?

CloudFormation은 AWS 리소스를 모델링하고 설정하여 리소스 관리 시간을 줄이고 AWS에서 실행되는 애플리케이션에 더 많은 시간을 사용하도록 해 주는 서비스입니다. 필요한 모든 AWS 리소스(예: Amazon EC2 인스턴스 또는 Amazon RDS DB 인스턴스)를 설명하는 템플릿을 생성하면 CloudFormation이 해당 리소스의 프로비저닝과 구성을 담당합니다. AWS 리소스를 개별적으로 생성하고 구성할 필요가 없으며 어떤 것이 무엇에 의존하는지 파악할 필요도 없습니다. CloudFormation에서 모든 것을 처리합니다. 다음 시나리오는 CloudFormation이 얼마나 유용한지를 보여줍니다.

인프라 관리 간소화

확장 가능한 웹 애플리케이션에 백엔드 데이터베이스도 포함되어 있는 경우 Auto Scaling 그룹, Elastic Load Balancing 로드 밸런서 및 Amazon Relational Database Service 데이터베이스 인스턴스를 사용할 수 있습니다. 각 개별 서비스를 사용하여 이러한 리소스를 프로비저닝할 수 있으며 리소스를 생성한 후에는 함께 작동하도록 구성해야 합니다. 모든 작업을 마치고 애플리케이션을 제대로 실행하려면 복잡하고 많은 시간이 소요될 수 있습니다.

이렇게 하는 대신 CloudFormation 템플릿을 생성하거나 기존 템플릿을 수정할 수 있습니다. 템플릿은 모든 리소스와 해당 속성을 설명합니다. 템플릿을 사용하여 CloudFormation 스택을 생성할 경우 CloudFormation에서 Auto Scaling 그룹, 로드 밸런서 및 데이터베이스를 자동으로 프로비저닝합니다. 스택이 생성된 이후에 AWS 리소스가 실행됩니다. 스택을 쉽게 삭제할 수 있습니다. 스택에 있는 모든 리소스를 삭제하면 됩니다. CloudFormation을 사용하여 리소스 모음을 단일 단위로 쉽게 관리할 수 있습니다.

신속하게 인프라 복제

애플리케이션의 가용성을 확대해야 하는 경우 여러 리전에서 애플리케이션을 복제할 수 있습니다. 그러면 한 리전에서 사용할 수 없는 경우에도 다른 리전의 사용자는 애플리케이션을 계속해서 사용할 수 있습니다. 애플리케이션을 복제할 때 리소스도 복제해야 한다는 문제가 있습니다. 애플리케이션에 필요한 모든 리소스를 기록해야 할 뿐 아니라, 각 리전에서 해당 리소스를 프로비저닝하고 구성해야 합니다.

CloudFormation 템플릿을 재사용하여 리소스를 일관되고 반복적으로 생성할 수 있습니다. 템플릿을 재사용하려면 리소스를 한 번만 설명하고 여러 리전에서 동일한 리소스를 반복적으로 프로비저닝할 수 있습니다.

인프라 변경 사항을 쉽게 제어 및 추적

기본 리소스를 증분 방식으로 업그레이드해야 하는 경우도 있습니다. 예를 들어 Auto Scaling 시작 구성에서 더 우수한 성능의 인스턴스 유형으로 변경하여 Auto Scaling 그룹의 최대 인스턴스 수를 줄일 수 있습니다. 업데이트를 완료한 이후에 문제가 발생할 경우 인프라를 원래 설정으로 롤백해야 할 수 있습니다. 이 작업을 수동으로 수행하려면 변경된 리소스를 기억하고 원래 설정을 알고 있어야 합니다.

CloudFormation을 사용하여 인프라를 프로비저닝할 경우 프로비저닝되는 리소스와 해당 설정이 CloudFormation 템플릿에 정확히 설명됩니다. 템플릿은 텍스트 파일이므로 템플릿에서 차이점을 추적하여 인프라 변경 사항을 추적할 수 있습니다. 이는 개발자가 소스 코드에 대한 개정 사항을 제어하는 방식과 비슷합니다. 예를 들어 버전 제어 시스템을 템플릿과 함께 사용하여 변경된 내용, 변경한 사람, 변경 시간 등을 정확히 알 수 있습니다. 언제든지 인프라에 대한 변경 사항을 되돌려야 하는 경우 이전 버전의 템플릿을 사용할 수 있습니다.

CloudFormation 시작하기

CloudFormation은 CloudFormation [콘솔](#), [API](#), [AWS CLI](#), [AWS SDK](#) 및 여러 통합을 통해 사용할 수 있습니다.

CloudFormation에 대한 소개는 [CloudFormation 작동 방식](#) 단원을 참조하세요.

CloudFormation 사용을 시작하려면 [첫 번째 스택 생성](#) 단원을 참조하세요.

관련 정보

이 사용 설명서와 다음 리소스를 사용하여 CloudFormation에 대해 자세히 알아볼 수 있습니다.

- 제품 세부 정보 및 FAQ는 [AWS CloudFormation 제품 페이지](#)를 참조하세요.
- 요금 정보는 [AWS CloudFormation 요금](#)을 참조하세요.

CloudFormation 시작하기

예제 템플릿에서 스택을 생성하여 AWS Management Console을 통해 CloudFormation 사용을 시작할 수 있습니다. 그러면 스택 생성의 기본 사항을 배우는 데 도움이 됩니다. 템플릿은 스택 내 모든 리소스를 정의하는 텍스트 파일입니다. 스택은 CloudFormation 템플릿의 배포입니다. 단일 템플릿에서 여러 스택을 생성할 수 있습니다. 각 스택은 하나의 단위로 관리할 수 있는 AWS 리소스 모음을 포함합니다.

CloudFormation이 무료 서비스이긴 하지만 스택에 포함시키는 AWS 리소스에 대해서는 리소스별 현재 시세로 요금이 청구됩니다. AWS 요금에 대한 자세한 내용을 알아보려면 <http://aws.amazon.com>에서 각 제품에 대한 세부 정보 페이지를 참조하세요.

주제

- [CloudFormation 작동 방식](#)
- [AWS 계정에 가입](#)
- [첫 번째 스택 생성](#)

CloudFormation 작동 방식

이 주제에서는 CloudFormation의 작동 방식을 설명하고 사용할 때 알아야 할 주요 개념을 소개합니다.

주제

- [주요 개념](#)
- [CloudFormation 작동 방식](#)
- [CloudFormation을 시작하는 방법](#)

주요 개념

CloudFormation을 사용하는 경우에는 템플릿 및 스택으로 작업합니다. 템플릿을 생성하여 AWS 리소스와 해당 속성에 대해 설명합니다. 스택을 생성할 때마다 CloudFormation에서 템플릿에 설명된 리소스를 프로비저닝합니다.

주제

- [템플릿](#)
- [스택](#)
- [변경 세트](#)

템플릿

CloudFormation 템플릿은 YAML 또는 JSON 형식의 텍스트 파일입니다. .yaml, .json, .template, .txt 등의 확장명을 사용하여 이러한 파일을 저장할 수 있습니다. CloudFormation은 이러한 템플릿을 AWS 리소스 구축을 위한 청사진으로 사용합니다. 예를 들어, 템플릿에서 인스턴스 유형, AMI ID, 블록 디바이스 매핑, Amazon EC2 키 페어 이름 등과 같은 Amazon EC2 인스턴스를 설명할 수 있습니다. 스택을 생성할 때마다 CloudFormation에서 템플릿에 설명된 항목을 생성하는 데 사용되는 템플릿도 지정합니다.

예를 들어, 다음 템플릿을 사용하여 스택을 생성한 경우 CloudFormation에서는 ami-0ff8a91507f77f867 AMI ID, t2.micro 인스턴스 유형, testkey 키 페어 이름 및 Amazon EBS 볼륨을 사용하여 인스턴스를 프로비저닝합니다.

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: A sample template
Resources:
  MyEC2Instance:
    Type: 'AWS::EC2::Instance'
    Properties:
      ImageId: ami-0ff8a91507f77f867
      InstanceType: t2.micro
      KeyName: testkey
      BlockDeviceMappings:
        - DeviceName: /dev/sdm
          Ebs:
            VolumeType: io1
            Iops: 200
            DeleteOnTermination: false
            VolumeSize: 20
```

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "A sample template",
  "Resources": {
    "MyEC2Instance": {
      "Type": "AWS::EC2::Instance",
      "Properties": {
```



```

        "ImageId": "ami-0ff8a91507f77f867",
        "InstanceType": "t2.micro",
        "KeyName": "testkey",
        "BlockDeviceMappings": [
            {
                "DeviceName": "/dev/sdm",
                "Ebs": {
                    "VolumeType": "io1",
                    "Iops": 200,
                    "DeleteOnTermination": false,
                    "VolumeSize": 20
                }
            }
        ]
    }
}

```

단일 템플릿에서 여러 리소스를 지정하고 해당 리소스를 함께 작동하도록 구성할 수도 있습니다. 예를 들어, 다음 예제와 같이 탄력적 IP 주소(EIP)를 포함하도록 이전 템플릿을 수정한 후 Amazon EC2 인스턴스와 연결할 수 있습니다.

YAML

```

AWSTemplateFormatVersion: 2010-09-09
Description: A sample template
Resources:
  MyEC2Instance:
    Type: 'AWS::EC2::Instance'
    Properties:
      ImageId: ami-0ff8a91507f77f867
      InstanceType: t2.micro
      KeyName: testkey
      BlockDeviceMappings:
        - DeviceName: /dev/sdm
          Ebs:
            VolumeType: io1
            Iops: 200
            DeleteOnTermination: false
            VolumeSize: 20
  MyEIP:
    Type: 'AWS::EC2::EIP'

```

Properties:

InstanceId: !Ref MyEC2Instance

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "A sample template",
  "Resources": {
    "MyEC2Instance": {
      "Type": "AWS::EC2::Instance",
      "Properties": {
        "ImageId": "ami-0ff8a91507f77f867",
        "InstanceType": "t2.micro",
        "KeyName": "testkey",
        "BlockDeviceMappings": [
          {
            "DeviceName": "/dev/sdm",
            "Ebs": {
              "VolumeType": "io1",
              "Iops": 200,
              "DeleteOnTermination": false,
              "VolumeSize": 20
            }
          }
        ]
      }
    },
    "MyEIP": {
      "Type": "AWS::EC2::EIP",
      "Properties": {
        "InstanceId": {
          "Ref": "MyEC2Instance"
        }
      }
    }
  }
}
```

이전 템플릿은 단일 Amazon EC2 인스턴스를 기반으로 하지만 CloudFormation 템플릿은 복잡한 리소스 세트를 구축하는 데 사용 가능한 추가 기능이 있으며 다양한 컨텍스트에서 템플릿을 재사용할 수 있습니다. 예를 들어 CloudFormation 스택을 생성할 때 값이 지정되는 입력 파라미터를 추가할 수 있습니다.

다. 다시 말해 템플릿을 생성할 때가 아니라 스택을 생성할 때 인스턴스 유형과 같은 값을 지정할 수 있으므로 다양한 상황에서 템플릿을 더 쉽게 재사용할 수 있습니다.

스택

CloudFormation을 사용할 경우 스택이라는 하나의 단위로 관련 리소스를 관리합니다. 스택을 생성, 업데이트, 삭제하여 리소스 모음을 생성, 업데이트 및 삭제합니다. 스택의 모든 리소스는 스택의 CloudFormation 템플릿으로 정의합니다. Auto Scaling 그룹, Elastic Load Balancing 로드 밸런서 및 Amazon Relational Database Service(Amazon RDS) 데이터베이스 인스턴스를 포함하는 템플릿을 생성했다고 가정합니다. 이러한 리소스를 생성하려면 생성한 템플릿을 제출하여 스택을 생성합니다. 그러면 CloudFormation에서 모든 리소스를 자동으로 프로비저닝합니다.

변경 세트

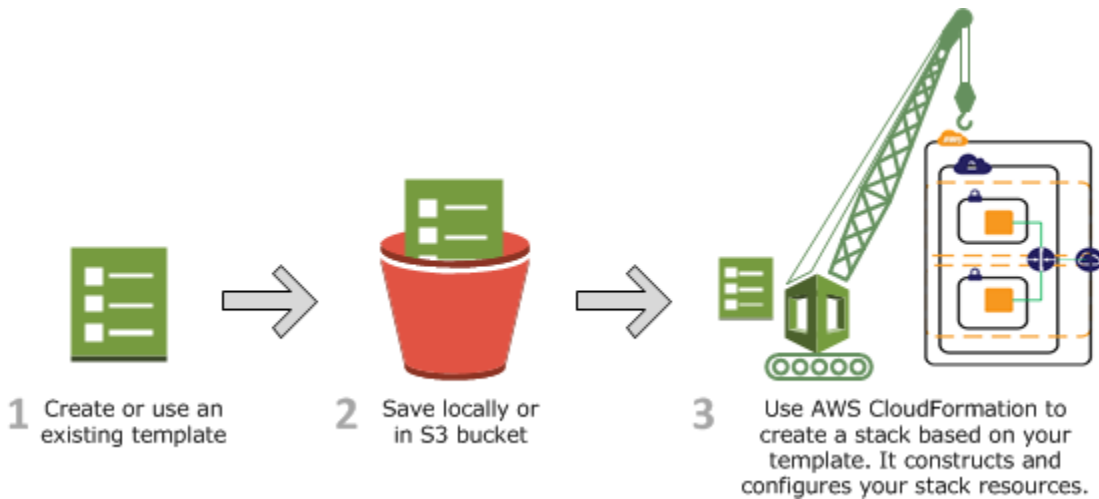
스택에서 실행 중인 리소스를 변경해야 하는 경우 스택을 업데이트합니다. 리소스를 변경하기 전에 제안된 변경 사항이 요약된 변경 세트를 생성할 수 있습니다. 변경 세트를 사용하면 변경 사항을 구현하기 이전에 해당 변경이 실행 중인 리소스 특히, 중요 리소스에 미치는 영향을 확인할 수 있습니다.

예를 들어, Amazon RDS 데이터베이스 인스턴스의 이름을 변경한 경우 CloudFormation에서 새 데이터베이스를 생성하고 이전 데이터베이스를 삭제합니다. 아직 백업하지 않은 경우 이전 데이터베이스의 데이터를 잃게 됩니다. 변경 세트를 생성한 경우 변경 후 데이터베이스가 교체된다는 사실을 알 수 있으므로 스택을 업데이트하기 이전에 적절히 계획할 수 있습니다.

CloudFormation 작동 방식

CloudFormation을 사용하여 스택을 생성하는 경우 CloudFormation에서는 AWS에 대한 기본 서비스 직접 호출을 수행하여 템플릿에서 설명한 리소스를 프로비저닝 및 구성합니다. 이러한 리소스를 생성하려면 권한이 필요합니다. 예를 들어 CloudFormation을 사용하여 EC2 인스턴스를 생성하려면 인스턴스를 생성할 권한이 필요합니다. [AWS Identity and Access Management\(IAM\)](#)을 사용하여 이러한 권한을 관리합니다.

CloudFormation에서 실행하는 호출은 모두 템플릿으로 선언됩니다. 예를 들어 t2.micro 인스턴스 유형으로 EC2 인스턴스를 설명하는 템플릿이 있다고 가정해 보겠습니다. 이 템플릿을 사용해 스택을 생성하는 경우 CloudFormation에서는 Amazon EC2 인스턴스 생성 API를 생성하고 인스턴스 유형을 t2.micro로 지정합니다. 다음 다이어그램은 CloudFormation의 스택 생성 워크플로우를 요약해서 보여줍니다.



스택 생성

1. 텍스트 편집기를 사용하여 JSON 또는 YAML 형식으로 CloudFormation 템플릿을 생성하세요. CloudFormation 템플릿에는 필요한 리소스와 해당하는 설정이 설명됩니다. [Infrastructure Composer](#)를 사용하여 템플릿을 시각화하고 검증하세요. 이렇게 하면 템플릿을 올바르게 구성하고 구문 오류가 없도록 보장할 수 있습니다. 자세한 내용은 [CloudFormation 템플릿 작업](#) 섹션을 참조하세요.
2. 템플릿을 로컬 또는 Amazon S3 버킷에 저장합니다.
3. 템플릿 파일의 위치(예: 로컬 컴퓨터의 경로 또는 Amazon S3 URL)를 지정하여 CloudFormation 스택을 생성합니다. 템플릿에 파라미터가 포함되어 있는 경우 스택을 생성할 때 입력 값을 지정할 수 있습니다. 파라미터를 통해 템플릿을 값을 전달하여 스택을 생성할 때마다 리소스를 사용자 지정할 수 있습니다.

Note

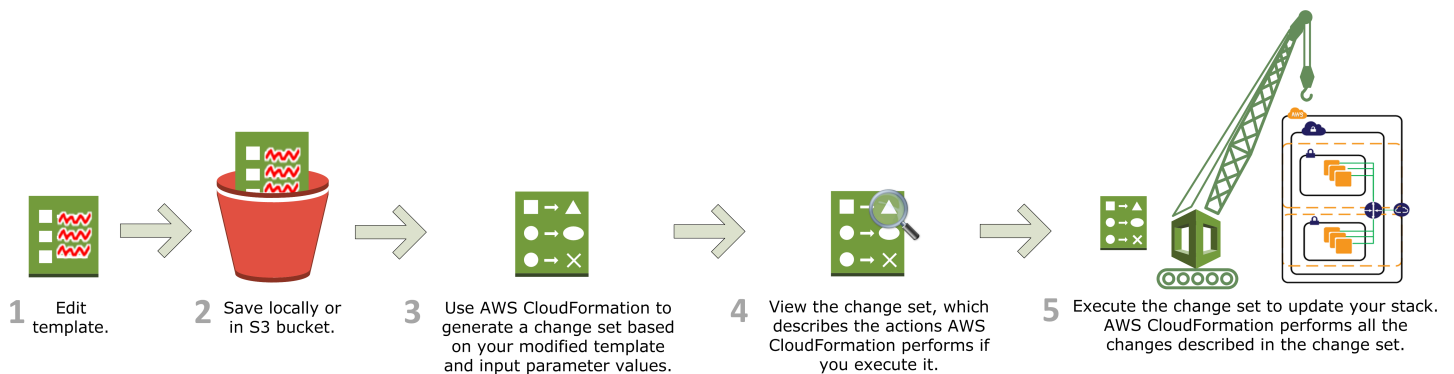
로컬에 저장된 템플릿 파일을 지정하면 CloudFormation이 이를 AWS 계정의 S3 버킷에 업로드합니다. CloudFormation은 템플릿 파일을 업로드하는 각 리전에 대한 버킷을 생성합니다. AWS 계정에서 Amazon Simple Storage Service(Amazon S3) 권한이 있는 모든 사용자가 버킷에 액세스할 수 있습니다. CloudFormation에서 생성한 버킷이 이미 있는 경우, 해당 버킷에 템플릿이 추가됩니다.

템플릿을 Amazon S3에 수동으로 업로드하여 자체 버킷을 사용하고 버킷의 권한을 관리할 수 있습니다. 그런 다음 스택을 생성하거나 업데이트할 때마다 템플릿 파일의 Amazon S3 URL을 지정합니다.

모든 리소스가 생성된 후 CloudFormation에서는 스택이 생성되었음을 보고합니다. 그러면 스택의 리소스를 사용할 수 있습니다. 스택 생성에 실패하면 CloudFormation에서는 생성한 리소스를 삭제하여 변경 사항을 롤백합니다.

변경 세트로 스택 업데이트

스택의 리소스를 업데이트해야 하는 경우 스택의 템플릿을 수정할 수 있습니다. 새 스택을 만들고 이전 스택은 삭제할 필요가 없습니다. 스택을 업데이트하려면 원본 스택 템플릿의 수정된 버전, 다른 입력 파라미터 또는 둘 다를 제출하여 변경 세트를 생성합니다. CloudFormation은 수정된 템플릿을 원본 템플릿과 비교하고 변경 세트를 생성합니다. 변경 세트에는 제안된 변경 사항이 나열되어 있습니다. 변경 사항을 검토한 후 변경 세트를 실행해 스택을 업데이트하거나 새 변경 세트를 시작할 수 있습니다. 다음 다이어그램은 스택 업데이트 워크플로우를 요약해서 보여줍니다.



변경 세트로 스택을 업데이트하는 방법

1. [Infrastructure Composer](#) 또는 텍스트 편집기를 사용하여 CloudFormation 스택 템플릿을 수정할 수 있습니다. 자세한 내용은 [스택 템플릿 업데이트](#) 섹션을 참조하세요.

템플릿을 업데이트할 때 업데이트로 인해 중단이 발생할 수 있다는 점에 유의하세요. 업데이트 중인 리소스 및 속성에 따라 업데이트 시 기존 리소스가 중단되거나 대체될 수 있습니다. 자세한 내용은 [스택 리소스의 업데이트 동작 이해](#) 섹션을 참조하세요.

2. CloudFormation 템플릿을 로컬 또는 S3 버킷에 저장합니다.
3. 업데이트하려는 스택과 수정된 템플릿의 위치(예: 로컬 컴퓨터의 경로 또는 Amazon S3 URL)를 지정하여 변경 세트를 생성합니다. 변경 세트 생성에 대한 자세한 내용은 [변경 세트를 사용하여 CloudFormation 스택 업데이트](#) 단원을 참조하십시오.

Note

로컬 컴퓨터에 저장되는 템플릿을 지정한 경우 CloudFormation에서는 해당 템플릿을 AWS 계정의 S3 버킷에 템플릿을 자동으로 업로드합니다.

4. 변경 세트를 보고 기대하는 변경을 CloudFormation에서 수행할지 확인합니다. 예를 들어 CloudFormation에서 중요한 스택 리소스를 대체할지 확인합니다. 원하는 변경 사항을 포함할 때까지 변경 세트는 필요한 만큼 생성할 수 있습니다.

Important

변경 세트는 스택 업데이트에 성공했는지 여부는 나타내지 않습니다. 예를 들어 변경 세트는 계정 [할당량](#)을 초과하는 경우, 업데이트를 지원하지 않는 리소스를 업데이트하는 경우 또는 리소스를 수정할 [권한](#)이 불충분한 경우를 확인하지 않습니다. 이러한 경우는 스택 업데이트 실패를 일으킬 수 있습니다.

5. 스택에 적용하려는 변경 세트를 시작합니다. CloudFormation에서는 수정한 리소스만 업데이트하여 스택을 업데이트하고 스택 업데이트에 성공했다는 신호를 전송합니다. 스택 업데이트에 실패하면 CloudFormation에서는 변경 사항을 롤백해 스택을 알려진 마지막 작동 상태로 복원합니다.

CloudFormation을 시작하는 방법

콘솔을 사용하여 hello world CloudFormation 스택을 생성하려면 [첫 번째 스택 생성](#) 섹션을 참조하세요.

안내 학습을 위해 템플릿 개발에 대한 실습 경험을 제공하는 [CloudFormation 시작하기](#) 워크숍을 사용해 보세요.

AWS 계정에 가입

AWS에 가입하면 CloudFormation를 포함하여 AWS의 모든 서비스에 AWS 계정 계정이 자동으로 등록됩니다. 이미 AWS 계정이 있다면 다음 항목으로 건너뛰세요. AWS 계정이 없는 경우에는 다음 절차에 따라 계정을 만드세요.

AWS 계정에 가입

AWS 계정이 없는 경우 다음 절차에 따라 계정을 생성합니다.

AWS 계정에 가입

1. <https://portal.aws.amazon.com/billing/signup>을 엽니다.
2. 온라인 지시 사항을 따릅니다.

등록 절차 중 전화 또는 텍스트 메시지를 받고 전화 키패드로 확인 코드를 입력하는 과정이 있습니다.

AWS 계정에 가입하면 AWS 계정 루트 사용자가 생성됩니다. 루트 사용자에게는 계정의 모든 AWS 서비스 및 리소스에 액세스할 권한이 있습니다. 보안 모범 사례는 사용자에게 관리 액세스 권한을 할당하고, 루트 사용자만 사용하여 [루트 사용자 액세스 권한이 필요한 작업](#)을 수행하는 것입니다.

AWS는 가입 절차 완료된 후 사용자에게 확인 이메일을 전송합니다. 언제든지 <https://aws.amazon.com/>으로 이동하고 내 계정을 선택하여 현재 계정 활동을 보고 계정을 관리할 수 있습니다.

관리자 액세스 권한이 있는 사용자 생성

AWS 계정에 가입하고 AWS 계정 루트 사용자에게 보안 조치를 한 다음, AWS IAM Identity Center를 활성화하고 일상적인 작업에 루트 사용자를 사용하지 않도록 관리 사용자를 생성합니다.

귀하의 AWS 계정 루트 사용자 보호

1. 루트 사용자를 선택하고 AWS 계정 이메일 주소를 입력하여 [AWS Management Console](#)에 계정 소유자로 로그인합니다. 다음 페이지에서 비밀번호를 입력합니다.

루트 사용자를 사용하여 로그인하는 데 도움이 필요하면 AWS 로그인 User Guide의 [루트 사용자 로 로그인](#)을 참조하세요.

2. 루트 사용자의 다중 인증(MFA)을 활성화합니다.

지침은 IAM 사용 설명서의 [AWS 계정 루트 사용자용 가상 MFA 디바이스 활성화\(콘솔\)](#)를 참조하세요.

관리자 액세스 권한이 있는 사용자 생성

1. IAM Identity Center를 활성화합니다.

지침은 AWS IAM Identity Center 사용 설명서의 [AWS IAM Identity Center 설정](#)을 참조하세요.

2. IAM Identity Center에서 사용자에게 관리 액세스 권한을 부여합니다.

IAM Identity Center 디렉터리를 ID 소스로 사용하는 방법에 대한 자습서는 AWS IAM Identity Center 사용 설명서의 [기본 IAM Identity Center 디렉터리로 사용자 액세스 구성](#)을 참조하세요.

관리 액세스 권한이 있는 사용자로 로그인

- IAM Identity Center 사용자로 로그인하려면 IAM Identity Center 사용자를 생성할 때 이메일 주소로 전송된 로그인 URL을 사용합니다.

IAM Identity Center 사용자로 로그인하는 데 도움이 필요한 경우 AWS 로그인 사용 설명서의 [AWS 액세스 포털에 로그인](#)을 참조하세요.

추가 사용자에게 액세스 권한 할당

1. IAM Identity Center에서 최소 권한 적용 모범 사례를 따르는 권한 세트를 생성합니다.

지침은 AWS IAM Identity Center 사용 설명서의 [Create a permission set](#)를 참조하세요.

2. 사용자를 그룹에 할당하고, 그룹에 Single Sign-On 액세스 권한을 할당합니다.

지침은 AWS IAM Identity Center 사용 설명서의 [Add groups](#)를 참조하세요.

Note

사용자의 액세스 권한과 액세스 대상을 관리하는 방법에 대한 자세한 내용은 [AWS Identity and Access Management를 사용하여 CloudFormation 액세스 제어](#) 섹션을 참조하세요.

첫 번째 스택 생성

이 주제에서는 AWS Management Console을 사용하여 첫 번째 CloudFormation 스택을 생성하는 방법을 설명합니다. 이 자습서를 따르면 기본 AWS 리소스를 프로비저닝하고 스택 이벤트를 모니터링하고 출력을 생성하는 방법을 배울 수 있습니다.

이 예제에서 CloudFormation 템플릿은 YAML로 작성됩니다. YAML은 인프라를 코드로 정의하는 데 널리 사용되는 사람이 읽을 수 있는 형식입니다. CloudFormation에 대한 학습을 계속하다 보면 JSON 형식의 다른 템플릿도 접할 수 있지만 이 자습서에서는 가독성을 위해 YAML을 선택했습니다.

Note

CloudFormation은 무료이지만 생성하는 Amazon EC2 및 Amazon S3 리소스에 대해서는 요금이 부과됩니다. 그러나 AWS를 처음 사용하는 경우 [프리 티어](#)를 활용하여 이 학습 프로세스 중에 비용을 최소화하거나 제거할 수 있습니다.

주제

- [사전 조건](#)
- [콘솔을 사용하여 CloudFormation 스택 생성](#)
- [스택 생성 모니터링](#)
- [웹 서버 테스트](#)
- [문제 해결](#)
- [정리](#)
- [다음 단계](#)

사전 조건

- Amazon EC2, Amazon S3 및 CloudFormation을 사용할 수 있는 권한을 보유한 IAM 사용자 또는 역할이 있는 AWS 계정에 대한 액세스 권한 또는 관리 사용자 액세스 권한이 있어야 합니다.
- 인터넷에 액세스할 수 있는 가상 프라이빗 클라우드(VPC)가 있어야 합니다. 이 연습 템플릿에는 최신 AWS 계정과 함께 자동으로 제공되는 기본 VPC가 필요합니다. 기본 VPC가 없거나 삭제된 경우에 적용되는 다른 해결 방법은 이 주제의 문제 해결 섹션을 참조하세요.

콘솔을 사용하여 CloudFormation 스택 생성

콘솔을 사용하여 Hello World CloudFormation 스택을 생성하려면

1. [CloudFormation 콘솔](#)을 엽니다.
2. 스택 생성을 선택합니다.
3. 스택 생성 페이지에서 Infrastructure Composer에서 빌드를 선택한 다음 Infrastructure Composer에서 생성을 선택합니다. 그러면 예제 템플릿을 업로드하고 검증할 수 있는 CloudFormation 콘솔 모드의 Infrastructure Composer로 이동합니다.
4. 예제 템플릿을 업로드하고 검증하려면 다음을 수행합니다.

- a. 템플릿을 선택합니다. 다음 CloudFormation 템플릿을 복사하여 템플릿 편집기에 붙여넣습니다.

```
AWSTemplateFormatVersion: 2010-09-09
Description: CloudFormation Template for WebServer with Security Group and EC2 Instance

Parameters:
  LatestAmiId:
    Description: The latest Amazon Linux 2 AMI from the Parameter Store
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: '/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2'

  InstanceType:
    Description: WebServer EC2 instance type
    Type: String
    Default: t2.micro
    AllowedValues:
      - t3.micro
      - t2.micro
    ConstraintDescription: must be a valid EC2 instance type.

  MyIP:
    Description: Your IP address in CIDR format (e.g. 203.0.113.1/32).
    Type: String
    MinLength: '9'
    MaxLength: '18'
    Default: 0.0.0.0/0
    AllowedPattern: '^(\d{1,3}\.){3}\d{1,3}\.(\d{1,2})$'
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.

Resources:
  WebServerSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Allow HTTP access via my IP address
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: !Ref MyIP
```

```

WebServer:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref LatestAmiId
    InstanceType: !Ref InstanceType
    SecurityGroupIds:
      - !Ref WebServerSecurityGroup
    UserData: !Base64 |
      #!/bin/bash
      yum update -y
      yum install -y httpd
      systemctl start httpd
      systemctl enable httpd
      echo "<html><body><h1>Hello World!</h1></body></html>" > /var/www/html/
index.html

Outputs:
  WebsiteURL:
    Value: !Join
      - ''
      - - http://
        - !GetAtt WebServer.PublicDnsName
    Description: Website URL

```

다음 단계로 넘어가기 전에 템플릿을 살펴보고 몇 가지 주요 CloudFormation 개념을 이해하는 시간을 갖도록 하겠습니다.

- **Parameters** 섹션은 스택을 생성할 때 템플릿으로 전달할 수 있는 값을 선언합니다. 템플릿 뒷부분에 지정된 리소스는 이러한 값을 참조하고 데이터를 사용합니다. 파라미터는 템플릿 자체에 저장하고 싶지 않은 정보를 지정하는 효과적인 방법입니다. 또한 배포 중인 특정 애플리케이션 또는 구성에 고유할 수 있는 정보를 지정하는 방법이기도 합니다.
- 템플릿에서 정의하는 파라미터는 다음과 같습니다.
 - **LatestAmiId** - AWS Systems Manager Parameter Store에서 최신 Amazon Linux 2 AMI ID를 검색합니다.
 - **InstanceType** - EC2 인스턴스 유형을 선택할 수 있습니다(기본값: t2.micro, 허용: t3.micro, t2.micro).
 - **MyIP** - HTTP 액세스의 IP 주소 범위를 지정합니다(기본값: 0.0.0.0/0, 모든 IP에서 액세스 허용).

- **Resources** 섹션에는 이 템플릿으로 생성하려는 AWS 리소스에 대한 정의가 포함됩니다. 리소스 선언은 한 번에 모든 구성 설정을 지정할 수 있는 효율적인 방법입니다. 템플릿에 리소스 선언을 포함하면 해당 템플릿을 사용하여 스택을 생성함으로써 선언된 모든 리소스를 생성 및 구성할 수 있습니다. 동일한 템플릿에서 새 스택을 생성하여 동일한 구성의 리소스를 시작할 수도 있습니다.
 - 이 템플릿은 다음 리소스를 만듭니다.
 - **WebServerSecurityGroup** - 지정된 IP 범위에서 포트 80의 인바운드 HTTP 트래픽을 허용하는 EC2 보안 그룹입니다.
 - **WebServer** - 다음 구성의 EC2 인스턴스입니다.
 - 최신 Amazon Linux 2 AMI를 사용합니다.
 - 선택한 인스턴스 유형을 적용합니다.
 - SecurityGroupIds 속성에 WebServerSecurityGroup을 추가합니다.
 - Apache HTTP Server를 설치하기 위한 사용자 데이터 스크립트가 포함되어 있습니다.
 - 논리명은 각 리소스 및 파라미터 선언의 시작 부분에 지정됩니다. 예를 들어 WebServerSecurityGroup은 EC2 보안 그룹 리소스에 할당된 논리명입니다. 그런 다음 Ref 함수를 사용하여 템플릿의 다른 부분에서 논리명으로 리소스 및 파라미터를 참조합니다. 한 리소스가 다른 리소스를 참조하면 리소스 간에 종속성이 생성됩니다.
 - 이 **Outputs** 섹션에서는 스택 생성 후 반환되는 사용자 지정 값을 정의합니다. 출력 값을 사용하여 리소스 식별자 또는 URL과 같은 스택의 리소스 정보를 반환할 수 있습니다.
 - 템플릿은 하나의 출력을 정의합니다.
 - **WebsiteURL** - EC2 인스턴스의 퍼블릭 DNS 이름을 사용하여 구성된 배포된 웹 서버의 URL입니다. Join 함수는 http://와 변수 PublicDnsName을 단일 문자열로 결합하여 웹 서버의 전체 URL을 쉽게 출력할 수 있도록 합니다.
- b. 템플릿을 업로드하기 전에 YAML 코드가 유효한지 확인하려면 검증을 선택합니다.
 - c. 그런 다음 템플릿 생성을 선택하여 템플릿을 생성하고 S3 버킷에 추가합니다.
 - d. 나중에 삭제할 수 있도록 열리는 대화 상자에서 S3 버킷의 이름을 기록해 둡니다. 그런 다음 확인 후 CloudFormation 계속 진행을 선택합니다. 그러면 이제 템플릿에 대한 S3 경로가 지정되어 있는 CloudFormation 콘솔로 이동합니다.
5. 스택 생성 페이지에서 다음을 선택합니다.
 6. 스택 세부 정보 지정 페이지의 스택 이름 필드에 이름을 입력합니다. 스택 이름에는 공백이 있어서는 안 됩니다. **MyTestStack**를 입력하세요.
 7. 파라미터에서 다음과 같이 파라미터 값을 지정합니다.

- LatestAmild: 기본적으로 최신 Amazon Linux 2 AMI로 설정됩니다.
- InstanceType: EC2 인스턴스 유형으로 t2.micro 또는 t3.micro를 선택합니다.

Note

AWS를 처음 사용하는 경우 프리 티어를 사용하여 12개월 동안 무료로 t2.micro 인스턴스를 시작하고 사용할 수 있습니다(t2.micro를 사용할 수 없는 리전에서는 프리 티어에서 t3.micro 인스턴스를 사용할 수 있음).

- MyIP: /32 접미사를 사용하여 실제 퍼블릭 IP 주소를 지정합니다. /32 접미사는 CIDR 표기법에서 단일 IP 주소를 허용하도록 지정하는 데 사용됩니다. 이는 기본적으로 이 특정 IP 주소와의 트래픽을 허용하고 다른 IP 주소는 허용하지 않음을 의미합니다.
8. 다음을 두 번 선택하여 검토 및 생성 페이지로 이동합니다. 이 자습서에서는 스택 옵션 구성 페이지의 기본값을 그대로 둘 수 있습니다.
 9. 스택에 대한 정보를 검토합니다. 설정에 만족하면 제출을 클릭합니다.

스택 생성 모니터링

제출을 선택하면 CloudFormation이 템플릿에 지정된 리소스를 생성하기 시작합니다.

CloudFormation 콘솔의 윗부분에 있는 목록에 새로운 스택 **MyTestStack**이 나타납니다. 상태는 CREATE_IN_PROGRESS과 같아야 합니다. 스택의 이벤트를 보고 스택에 대한 자세한 상태를 확인할 수 있습니다.

스택에 대한 이벤트를 보려면

1. CloudFormation 콘솔에서 목록 내 스택 **MyTestStack**을 선택합니다.
2. 스택 세부 정보 창에서 [이벤트(Events)] 탭을 선택합니다.

콘솔은 60초마다 최신 이벤트로 이벤트 목록을 자동으로 새로 고칩니다.

이벤트 탭에는 스택을 생성하는 각 주요 단계가 각 이벤트의 시간 순으로 정렬되어 표시됩니다. 즉, 최신 이벤트가 맨 위에 표시됩니다.

(이벤트 목록의 맨 아래 있는) 첫 번째 이벤트가 스택 생성 프로세스의 시작입니다.

2024-12-23 18:54 UTC-7 MyTestStack CREATE_IN_PROGRESS User initiated

다음은 각 리소스의 생성 시작 및 완료를 표시하는 이벤트입니다. 예를 들어, EC2 인스턴스가 생성되면 다음 항목이 나타납니다.

```
2024-12-23 18:59 UTC-7 WebServer CREATE_COMPLETE
```

```
2024-12-23 18:54 UTC-7 WebServer CREATE_IN_PROGRESS Resource creation initiated
```

CREATE_IN_PROGRESS 이벤트는 CloudFormation에서 리소스 생성이 시작되었다고 보고할 때 로깅됩니다. CREATE_COMPLETE 이벤트는 리소스를 성공적으로 생성한 경우에 로깅됩니다.

CloudFormation에서 스택을 성공적으로 생성하면 [이벤트(Events)] 탭 맨 위에 다음 이벤트가 표시됩니다.

```
2024-12-23 19:17 UTC-7 MyTestStack CREATE_COMPLETE
```

CloudFormation에서 리소스를 생성할 수 없는 경우 CREATE_FAILED 이벤트를 보고하고 기본적으로 스택을 롤백한 다음 생성된 리소스를 모두 삭제합니다. 상태 사유 열에는 실패를 일으키는 문제가 표시됩니다.

스택이 생성된 후 리소스 탭으로 이동하여 생성한 EC2 인스턴스 및 보안 그룹을 확인할 수 있습니다.

웹 서버 테스트

스택이 성공적으로 생성되면 CloudFormation 콘솔의 출력 탭으로 이동합니다. WebsiteURL 필드를 찾습니다. 여기에는 EC2 인스턴스의 퍼블릭 URL이 있습니다.

브라우저를 열고 WebsiteURL에 나열된 URL로 이동합니다. 브라우저에 간단한 "Hello World!" 메시지가 표시됩니다.

이를 통해 EC2 인스턴스가 Apache HTTP Server를 실행하고 기본 웹 페이지를 제공하고 있음을 확인할 수 있습니다.

문제 해결

스택 생성 중에 롤백이 발생하는 경우, VPC 누락 때문일 수 있습니다. 이 문제를 해결하는 방법은 다음과 같습니다.

사용 가능한 기본 VPC 없음

이 연습의 템플릿에는 기본 VPC가 필요합니다. VPC 또는 서브넷 가용성 오류로 인해 스택 생성에 실패하는 경우, 계정에 기본 VPC가 없기 때문일 수 있습니다. 다음과 같은 옵션이 있습니다.

- 새 기본 VPC 생성 - Amazon VPC 콘솔을 통해 새 기본 VPC를 생성할 수 있습니다. 자세한 지침은 Amazon VPC 사용 설명서의 [기본 VPC 생성](#)을 참조하세요.
- 템플릿을 수정하여 서브넷 지정 - 기본 VPC 이외의 VPC가 있는 경우 템플릿을 수정하여 VPC 및 서브넷 ID를 명시적으로 지정할 수 있습니다. 템플릿에 다음 파라미터를 추가합니다.

```
SubnetId:
  Description: The subnet ID to launch the instance into
  Type: AWS::EC2::Subnet::Id
```

그런 다음 서브넷 ID를 포함하도록 WebServer 리소스를 업데이트합니다.

```
WebServer:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref LatestAmiId
    InstanceType: !Ref InstanceType
    SecurityGroupIds:
      - !Ref WebServerSecurityGroup
    SubnetId: !Ref SubnetId
    UserData: !Base64 |
      #!/bin/bash
      yum update -y
      yum install -y httpd
      systemctl start httpd
      systemctl enable httpd
      echo "<html><body><h1>Hello World!</h1></body></html>" > /var/www/html/
index.html
```

스택을 생성할 때, 웹 서버에 연결할 수 있도록 인터넷 액세스가 가능한 서브넷을 지정해야 합니다.

정리

불필요한 서비스에 대해 요금이 청구되지 않도록 하려면 스택 및 리소스를 삭제하려 정리할 수 있습니다. 스택의 템플릿을 저장하는 S3 버킷을 삭제할 수도 있습니다.

스택 및 스택의 리소스를 삭제하려면

1. [CloudFormation 콘솔](#)을 엽니다.

2. 스택 페이지에서 생성한 스택의 이름(**MyTestStack**) 옆에 있는 옵션을 선택한 다음 삭제를 선택합니다.
3. 확인 메시지가 나타나면 삭제를 선택합니다.
4. 이벤트 탭에서 스택 삭제 프로세스의 진행 상황을 모니터링합니다. **MyTestStack**의 상태를 DELETE_IN_PROGRESS로 변경합니다. CloudFormation에서 스택 삭제를 완료하면 목록에서 해당 스택이 제거됩니다.

예제 템플릿 작업을 마쳤는데 더 이상 Amazon S3 버킷이 필요하지 않은 경우 버킷을 삭제합니다. 버킷을 삭제하려면 먼저 버킷을 비워야 합니다. 버킷을 비우면 버킷 내 모든 객체가 삭제됩니다.

Amazon S3 버킷을 비우고 삭제하려면

1. [Amazon S3 콘솔](#)을 엽니다.
2. 콘솔 왼쪽의 탐색 창에서 버킷을 선택합니다.
3. 버킷 목록에서 이 자습서를 위해 생성한 버킷의 이름 옆에 있는 옵션을 선택한 다음 비우기를 선택합니다.
4. 버킷 비우기 페이지에서 텍스트 필드에 **permanently delete**를 입력하여 버킷을 비우려는 의사를 확인한 후 비우기를 선택합니다.
5. [버킷 비우기: 상태(Empty bucket: Status)] 페이지에서 버킷 비우기 프로세스의 진행 상황을 모니터링합니다.
6. 버킷 목록으로 돌아가려면 종료를 선택합니다.
7. 버킷의 이름 옆에 있는 옵션을 선택한 다음 삭제를 선택합니다.
8. 확인 메시지가 나타나면 버킷의 이름을 입력한 다음 버킷 삭제를 선택합니다.
9. 버킷 목록에서 버킷 삭제 프로세스의 진행 상황을 모니터링합니다. Amazon S3가 버킷 삭제를 완료하면 목록에서 해당 버킷을 제거합니다.

다음 단계

축하합니다! 성공적으로 스택을 생성하고, 생성을 모니터링하고, 출력을 사용했습니다.

계속 학습하려면:

- 템플릿을 직접 생성할 수 있도록 템플릿에 대해 자세히 알아봅니다. 자세한 내용은 [CloudFormation 템플릿 작업](#) 섹션을 참조하세요.
- 템플릿 생성에 대한 실습을 더 해보려면 [CloudFormation 시작하기](#) 워크숍을 참조하세요.

- [CloudFormation 시작하기](#)의 단축 버전은 [Amazon EC2에 애플리케이션 배포](#) 섹션을 참조하세요. 이 주제에서는 `cfn-init`라는 CloudFormation 헬퍼 스크립트를 사용하여 Amazon EC2 인스턴스를 부트스트래핑하는, 동일한 시나리오를 설명합니다.

CloudFormation 모범 사례

모범 사례는 전체 워크플로에서 CloudFormation을 보다 효과적으로 사용하고 안전한 관행을 채택할 수 있도록 돕는 권장 사항입니다. 스택을 계획 및 구성하는 방법, 이러한 스택에서 실행되는 리소스 및 소프트웨어 애플리케이션을 형성하는 템플릿을 생성하는 방법 및 스택 및 리소스를 관리하는 방법을 알아봅니다. 다음 모범 사례는 현재 CloudFormation 고객의 실제 경험을 기반으로 합니다.

계획 및 구성

- [피드백 루프를 단축하여 개발 속도 개선](#)
- [수명 주기 및 소유권별로 스택 구성](#)
- [교차 스택 참조를 사용하여 다른 스택에서 내보낸 출력 값 반환](#)
- [다중 계정 및 다중 리전 배포에 CloudFormation StackSets 사용](#)
- [템플릿을 재사용하여 다양한 환경에서 스택 복제](#)
- [모든 리소스 유형에 대한 할당량 확인](#)
- [모듈을 사용하여 리소스 구성 재사용](#)
- [코드형 인프라 관행 채택](#)

템플릿 생성

- [템플릿에 자격 증명을 포함하지 않음](#)
- [AWS 특정 파라미터 유형 사용](#)
- [파라미터 제약 조건 사용](#)
- [유사 파라미터를 사용하여 이식성 향상](#)
- [AWS::CloudFormation::Init를 사용하여 Amazon EC2 인스턴스에서 소프트웨어 애플리케이션 배포](#)
- [최신 헬퍼 스크립트 사용](#)
- [템플릿을 사용하기 전에 템플릿 확인](#)
- [템플릿 작성에 YAML 또는 JSON 사용](#)
- [포괄적인 태그 지정 전략 구현](#)
- [고급 변환에 템플릿 매크로를 활용](#)

스택 관리

- [CloudFormation을 통해 모든 스택 리소스 관리](#)
- [스택을 업데이트하기 전에 변경 세트 생성](#)
- [스택 정책을 사용하여 리소스 보호](#)

- [CloudFormation 호출 기록에 AWS CloudTrail 사용](#)
- [코드 검토 및 버전 관리를 사용하여 사용자 템플릿 관리](#)
- [정기적으로 Amazon EC2 인스턴스 업데이트](#)
- [드리프트 감지를 정기적으로 사용](#)
- [자동 복구를 위한 롤백 트리거 구성](#)
- [효과적인 스택 리팩터링 전략 구현](#)
- [수명 주기 관리에 CloudFormation Hooks 사용](#)

작성 도구

- [IaC 생성기를 사용하여 기존 리소스에서 템플릿 생성](#)
- [시각적 템플릿 설계에 AWS 인프라 컴포저 사용](#)
- [복잡한 인프라에 AWS Cloud Development Kit \(AWS CDK\) 사용을 고려](#)

보안 및 규정 준수

- [IAM을 사용하여 액세스 제어](#)
- [최소 권한 원칙을 적용](#)
- [민감한 파라미터를 보호](#)
- [AWS CloudFormation Guard를 사용하여 코드형 정책 구현](#)

피드백 루프를 단축하여 개발 속도 개선

CloudFormation 템플릿으로 설명하는 인프라에 대한 피드백 루프를 단축하는 데 도움이 되는 방법과 도구를 채택하세요. 여기에는 워크스테이션에서 템플릿의 초기 린팅 및 테스트 수행이 포함됩니다. 이를 통해 소스 코드 리포지토리에 기여를 제출하기 전에도 잠재적인 구문 및 구성 문제를 발견할 수 있습니다. 이러한 문제를 조기에 발견하면 개발, 품질 보증 및 생산과 같은 공식 수명 주기 환경에 도달하는 것을 방지할 수 있습니다. 이러한 조기 테스트 및 빠른 실패 접근 방식은 재작업 대기 시간을 줄이고, 잠재적 영향 영역을 줄이고, 성공적인 프로비저닝 작업에 대한 신뢰도를 높이는 이점을 제공합니다.

빠른 실패 실행을 달성하는 데 도움이 되는 도구 선택에는 [CloudFormation Linter\(cfn-lint\)](#) 및 [TaskCat](#) 명령줄 도구가 포함됩니다. cfn-lint 도구는 [CloudFormation 리소스 사양](#)에 대해 CloudFormation 템플릿을 검증할 수 있는 기능을 제공합니다. 여기에는 리소스 속성의 유효한 값 확인과 모범 사례가 포함됩니다. cfn-lint용 플러그인은 [여러 코드 편집기에서 사용할 수 있습니다](#). 이를 통해 편집기 내에서 문제를 시각화하고 직접적인 린터 피드백을 얻을 수 있습니다. 기여를 커밋할 때 템플릿 검증을 수행할 수 있도록 소스 코드 리포지토리의 구성에 cfn-lint를 통합하도록 선택할 수도 있습니다. 자세한 내용은 [Git pre-commit validation of CloudFormation templates with cfn-](#)

[lint](#)를 참조하세요. 초기 린팅을 수행하고 cfn-lint에서 발생할 수 있는 모든 문제를 수정한 후에는 TaskCat을 사용하여 프로그래밍 방식으로 선택한 AWS 리전에 스택을 생성하여 템플릿을 테스트할 수 있습니다. 또한 TaskCat은 선택한 각 리전의 합격/불합격 성적이 포함된 보고서를 생성합니다.

두 도구를 모두 사용하여 피드백 루프를 단축하는 방법을 단계별로 실습해 보려면 [CloudFormation Workshop](#)의 [Linting and Testing lab](#)을 수행하세요.

수명 주기 및 소유권별로 스택 구성

AWS 리소스의 수명 주기 및 소유권을 사용하면 각 스택에 어떤 리소스가 있어야 하는지 결정하는 데 도움이 됩니다. 처음에는 하나의 스택에 모든 리소스를 넣어 둘 수 있지만 스택의 규모가 커지고 범위가 넓어지면 단일 스택을 관리하는 것이 번거롭고 시간이 오래 걸릴 수 있습니다. 공통 수명 주기 및 소유권으로 리소스를 그룹화하면 소유자는 다른 리소스에 영향을 미치지 않고 고유한 프로세스 및 일정을 사용하여 리소스 세트를 변경할 수 있습니다.

예를 들어 개발자와 엔지니어로 구성된 팀이 로드 밸런서를 지원하는 Amazon EC2 Auto Scaling 인스턴스에서 호스팅되는 웹 사이트를 소유하고 있다고 가정해 보겠습니다. 웹사이트는 자체 수명 주기를 가지고 있고 웹사이트 팀에서 유지하기 때문에 웹사이트 및 해당 리소스에 대한 스택을 생성할 수 있습니다. 이제 웹사이트에서 백엔드 데이터베이스를 사용한다고 가정해 보겠습니다. 여기에서 데이터베이스는 데이터베이스 관리자가 소유하고 유지하는 별도의 스택에 있습니다. 웹사이트 팀 또는 데이터베이스팀은 리소스를 업데이트해야 할 때마다 서로의 스택에 영향을 주지 않고 업데이트할 수 있습니다. 모든 리소스가 단일 스택에 있는 경우 업데이트 조정 및 전달이 어려울 수 있습니다.

스택 구성에 대한 추가 지침을 얻기 위해 다계층 아키텍처 및 서비스 중심 아키텍처(SOA), 이렇게 두 가지 공통 프레임워크를 사용할 수 있습니다.

계층화된 아키텍처는 서로 위에 쌓이는 다중 수평 계층으로 스택을 구성하고, 여기서 각 계층은 바로 아래 계층에 대한 종속성을 갖습니다. 각 계층에는 스택이 하나 이상 있을 수 있지만 각 계층 내에서 스택에는 수명 주기 및 소유권이 유사한 AWS 리소스가 있어야 합니다.

서비스 중심 아키텍처 덕분에 커다란 비즈니스 문제를 관리 가능한 부분으로 구성할 수 있습니다. 이러한 각 부분은 명확하게 정의된 목적을 갖는 서비스로, 독립형 기능 단위를 나타냅니다. 이러한 서비스는 스택에 매핑할 수 있으며 각 스택에는 자체 수명 주기 및 소유권이 있습니다. 이러한 서비스(스택)는 서로 상호 작용할 수 있도록 함께 연결할 수 있습니다.

교차 스택 참조를 사용하여 다른 스택에서 내보낸 출력 값 반환

수명 주기 및 소유권을 기반으로 AWS 리소스를 구성하는 경우 다른 스택에 있는 리소스를 사용하는 스택을 구축하려고 할 수 있습니다. 값을 하드코딩하거나 입력 파라미터를 사용하여 리소스 이름 및 ID

를 전달할 수 있습니다. 그러나 이러한 방법을 사용하면 템플릿을 재사용하기 어렵거나 스택을 실행하기 위해 오버헤드가 증가할 수 있습니다. 대신, 교차 스택 참조를 사용하여 다른 스택에서 내보낸 출력 값을 반환하여 다른 스택에서 사용할 수 있도록 합니다. 스택은 `Fn::ImportValue` 함수를 사용하여 내보낸 리소스를 호출해 사용할 수 있습니다.

예를 들어 VPC, 보안 그룹 및 서브넷을 포함하는 네트워크 스택이 있을 수 있습니다. 모든 퍼블릭 웹 애플리케이션에서 이러한 리소스를 사용하도록 하려고 합니다. 리소스를 내보내면 퍼블릭 웹 애플리케이션이 있는 모든 스택에서 이러한 리소스를 사용하도록 허용할 수 있습니다. 자세한 내용은 [배포된 CloudFormation 스택에서 내보낸 출력 가져오기](#) 섹션을 참조하세요.

다중 계정 및 다중 리전 배포에 CloudFormation StackSets 사용

CloudFormation StackSets에서는 단일 작업으로 여러 계정 및 리전에 대해 스택을 생성, 업데이트 또는 삭제할 수 있도록 하여 스택의 기능을 확장합니다. 조직 전체에 공통 인프라 구성 요소, 규정 준수 제어 또는 공유 서비스를 배포하는 데 StackSet를 사용합니다.

StackSet를 사용하는 경우 간소화된 권한 관리를 위해 AWS Organizations를 사용하여 서비스 관리형 권한을 구현합니다. 이 접근 방식을 사용하면 각 계정에서 IAM 역할을 수동으로 구성할 필요 없이 조직 내 계정에 StackSet를 배포할 수 있습니다.

StackSet에 대한 자세한 내용은 [StackSet 개념](#)을 참조하세요.

모든 리소스 유형에 대한 할당량 확인

스택을 시작하기 전에 AWS 계정 제한에 도달하지 않고 모든 리소스를 생성할 수 있는지 확인합니다. 제한을 초과한 경우 할당량을 늘리거나 추가 리소스를 삭제하기 전까지 CloudFormation에서는 스택을 성공적으로 생성할 수 없습니다. 각 서비스에는 스택을 시작하기 전에 알아두어야 할 다양한 제한이 있을 수 있습니다. 예를 들어 기본적으로 여러분의 AWS 계정에서는 리전당 CloudFormation 스택을 2,000개만 시작할 수 있습니다. 제한 및 기본 제한을 늘리는 방법에 대한 자세한 내용은 AWS 일반 참조의 [AWS Service Quotas](#)를 참조하세요.

템플릿을 재사용하여 다양한 환경에서 스택 복제

스택 및 리소스를 설정한 후에 템플릿을 재사용하여 다양한 환경에서 인프라를 복제할 수 있습니다. 예를 들어 변경 사항을 프로덕션에 구현하기 전에 변경 사항을 테스트할 수 있도록 개발, 테스트 및 프로덕션을 위한 환경을 생성할 수 있습니다. 템플릿을 재사용할 수 있도록 하려면 스택 생성 시 스택을 사용자 지정할 수 있도록 파라미터, 매핑 및 조건 섹션을 사용합니다. 예를 들어 개발 환경의 경우 프로덕션 환경에 비해 저렴한 인스턴스 유형을 지정할 수 있지만 다른 모든 구성 및 설정은 동일하게 유지됩니다.

니다. 파라미터, 매핑 및 조건에 대한 자세한 내용은 [CloudFormation 템플릿 섹션](#) 단원을 참조하십시오.

모듈을 사용하여 리소스 구성 재사용

인프라가 커짐에 따라 각 템플릿에서 동일한 구성 요소를 선언하는 공통 패턴이 나타날 수 있습니다. 모듈을 사용하면 모든 스택 템플릿에 포함하기 위한 리소스 구성을 투명하고 관리가 편리하면서도 반복적으로 패키징할 수 있습니다. 모듈은 공통적 서비스 구성과 모범 사례를 모듈식 맞춤형 구성 요소로 캡슐화하고 스택 템플릿에 포함할 수 있습니다.

Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 정의할 때의 모범 사례와 같이 단일 리소스에 이러한 구성 요소를 사용하거나 여러 리소스에 사용하여 애플리케이션 아키텍처의 공통 패턴을 정의할 수 있습니다. 이러한 구성 요소를 다른 모듈에 중첩하여 상위 구성 요소 안에 모범 사례를 포함할 수 있습니다. CloudFormation 모듈은 [CloudFormation 레지스트리](#)에서 사용할 수 있으므로 기본 리소스처럼 사용할 수 있습니다. CloudFormation 모듈을 사용하면 모듈 템플릿이 소비 템플릿으로 확장되므로 [Ref](#) 또는 [Fn::GetAtt](#)를 사용하여 모듈 안의 리소스에 액세스할 수 있습니다. 자세한 내용은 [CloudFormation 모듈을 사용하여 템플릿 전체에 포함할 수 있는 재사용 가능한 리소스 구성 생성](#) 섹션을 참조하세요.

코드형 인프라 관행 채택

코드형 인프라(IaC) 관행을 구현하여 CloudFormation 템플릿을 코드로 취급합니다. 템플릿을 버전 관리 시스템에 저장하고, 코드 검토를 구현하고, 자동 테스트를 사용하여 변경 사항을 검증합니다. 이 접근 방식은 일관성을 보장하고, 협업을 개선하고, 인프라 변경 사항에 대한 감사 추적을 제공합니다.

인프라 코드용 CI/CD 파이프라인을 구현하여 CloudFormation 템플릿의 테스트 및 배포를 자동화하는 것이 좋습니다. AWS CodePipeline, AWS CodeBuild, AWS CodeDeploy와 같은 도구를 사용하여 인프라 배포를 위한 자동화된 워크플로를 생성할 수 있습니다.

IaC 모범 사례 구현에 대한 자세한 내용은 [IaC 도구로 AWS CloudFormation 사용](#)을 참조하세요.

CloudFormation에서 지속적 전달을 사용하는 방법에 대한 자세한 내용은 [CodePipeline을 통한 지속적 전달](#)을 참조하세요.

템플릿에 자격 증명을 포함하지 않음

CloudFormation 템플릿에 민감한 정보를 포함하는 대신, 스택 템플릿의 동적 참조를 사용하는 것이 좋습니다.

동적 참조는 AWS Systems Manager Parameter Store 또는 AWS Secrets Manager와 같이 다른 서비스에서 저장 및 관리되는 외부 값을 참조하는 간결하면서 강력한 방법을 제공합니다. 동적 참조 사용 시 CloudFormation은 스택 및 변경 세트 작업 중에 필요한 경우 지정된 참조의 값을 검색하고 적절한 리소스에 값을 전달합니다. 하지만 CloudFormation은 실제 참조 값을 저장하지 않습니다. 자세한 내용은 [동적 참조를 사용하여 템플릿 값 지정](#)을 참조하세요.

[AWS Secrets Manager](#)는 데이터베이스와 다른 서비스의 자격 증명을 안전하게 암호화, 저장 및 검색하는 데 효과적입니다. [AWS Systems Manager Parameter Store](#)는 구성 데이터 관리를 위한 안전한 계층적 스토리지를 제공합니다.

템플릿 파라미터 정의에 대한 자세한 내용은 [CloudFormation 템플릿 Parameters 구문](#) 단원을 참조하십시오.

AWS 특정 파라미터 유형 사용

템플릿에 기존 AWS별 값(예: 기존 Amazon Virtual Private Cloud ID 또는 Amazon EC2 키 페어 이름)에 대한 입력값이 필요한 경우 AWS별 파라미터 유형을 사용합니다. 예를 들어 파라미터를 `AWS::EC2::KeyPair::KeyName` 유형으로 지정할 수 있습니다. 이는 스택을 생성한 AWS 계정 및 리전에 있는 기존 키 페어 이름을 가져옵니다. CloudFormation은 스택을 생성하기 전에 AWS 특정 파라미터 유형에 대한 값을 빠르게 검증할 수 있습니다. 또한 CloudFormation 콘솔을 사용할 경우 CloudFormation에 유효한 값의 드롭다운 목록이 표시되므로 올바른 VPC ID 또는 키 페어 이름을 조회하거나 기억할 필요가 없습니다. 자세한 내용은 [CloudFormation에서 제공하는 파라미터 유형을 사용하여 런타임 시 기존 리소스 지정](#) 섹션을 참조하세요.

파라미터 제약 조건 사용

제약 조건을 사용하면 스택을 생성하기 전에 CloudFormation에서 잘못된 값을 포착할 수 있도록 허용되는 입력 값을 설명할 수 있습니다. 최소 길이, 최대 길이 및 허용되는 패턴과 같은 제약 조건을 설정할 수 있습니다. 예를 들어 최소 길이는 8자이고 영숫자만 포함하도록 데이터베이스 사용자 이름값에 대한 제약 조건을 설정할 수 있습니다. 자세한 내용은 [CloudFormation 템플릿 Parameters 구문](#) 섹션을 참조하세요.

유사 파라미터를 사용하여 이식성 향상

템플릿에서 [의사 파라미터](#)를 `Ref` 및 `Fn::Sub`와 같은 [내장 함수](#)에 대한 인수로 사용할 수 있습니다. 가상 파라미터는 CloudFormation에서 사전 정의된 파라미터입니다. 따라서 템플릿에서 가상 파라미터

를 선언할 필요가 없습니다. 내장 함수에서 의사 파라미터를 사용하면 리전 및 계정 간에 스택 템플릿의 이식성이 향상됩니다.

예를 들어, 지정된 리소스 속성에 대해 다른 기존 리소스의 [Amazon 리소스 이름\(ARN\)](#)을 지정해야 하는 템플릿을 생성하려고 한다고 가정해 보겠습니다. 이 경우 기존 리소스는 ARN이 `arn:aws:ssm:us-east-1:123456789012:parameter/MySampleParameter`인 [AWS Systems Manager Parameter Store](#) 리소스입니다. [ARN 형식](#)을 대상 AWS 파티션, 리전 및 계정 ID에 맞게 조정해야 합니다. 이러한 값을 하드 코딩하는 대신 `AWS::Partition`, `AWS::Region` 및 `AWS::AccountId` 유사 파라미터를 사용하여 템플릿의 이식성을 높일 수 있습니다. 이 경우 다음 예제에서는 ARN의 요소를 CloudFormation과 연결하는 방법을 보여줍니다. !Sub 'arn:\${AWS::Partition}:ssm:\${AWS::Region}:\${AWS::AccountId}:parameter/MySampleParameter

다른 예로 여러 스택에서 리소스나 구성을 공유하려고 한다고 가정해 보겠습니다. 이 예에서는 VPC에 대한 [서브넷](#)을 생성한 다음 동일한 AWS 계정 및 리전의 다른 스택에서 사용하기 위해 해당 ID를 내보냈다고 가정합니다. 다른 스택에서는 Amazon EC2 인스턴스를 설명할 때 내보낸 서브넷 ID 값을 참조합니다. Export 출력 필드 및 Fn::ImportValue 내장 함수 사용에 대한 자세한 예제는 [다른 CloudFormation 스택의 리소스 출력 참조](#)를 참조하세요.

스택 내보내기는 계정 및 리전별로 고유해야 합니다. 따라서 이 경우 `AWS::StackName` 의사 파라미터를 사용하여 내보내기에 대한 접두사를 생성할 수 있습니다. 스택 이름은 계정 및 리전별로 고유해야 하므로 이 의사 파라미터를 접두사로 사용하면 고유한 내보내기 이름을 가질 가능성이 높아지는 동시에 값을 내보내는 스택 전체에서 재사용 가능한 접근 방식이 장려됩니다. 또는 원하는 접두사를 사용할 수 있습니다.

AWS::CloudFormation::Init를 사용하여 Amazon EC2 인스턴스에서 소프트웨어 애플리케이션 배포

스택을 시작하면 `cfn-init` 헬퍼 스크립트와 `AWS::CloudFormation::Init` 리소스를 사용하여 Amazon EC2 인스턴스에서 소프트웨어 애플리케이션을 설치하고 구성할 수 있습니다. `AWS::CloudFormation::Init`를 사용하여 절차 단계를 스크립팅하지 않고 원하는 구성을 설명할 수 있습니다. 인스턴스를 다시 생성하지 않고 구성을 업데이트할 수도 있습니다. 구성에 문제가 발생한 경우 CloudFormation에서는 문제를 조사하는 데 사용할 수 있는 로그를 생성합니다.

템플릿의 `AWS::CloudFormation::Init` 리소스에서 설치 및 구성 상태를 지정합니다. `cfn-init`와 `AWS::CloudFormation::Init`를 사용하는 방법을 보여주는 연습은 [Amazon EC2에 애플리케이션 배포](#) 섹션을 참조하세요.

최신 헬퍼 스크립트 사용

CloudFormation 헬퍼 스크립트는 주기적으로 업데이트됩니다. 헬퍼 스크립트를 호출하여 시작된 인스턴스에서 최신 헬퍼 스크립트를 가져왔는지 확인하기 전에 템플릿의 UserData 속성에 다음 명령이 포함되어 있는지 확인합니다.

```
yum install -y aws-cfn-bootstrap
```

최신 헬퍼 스크립트 가져오기에 대한 자세한 내용은 CloudFormation 템플릿 참조 가이드의 [CloudFormation 헬퍼 스크립트 참조](#)를 참조하세요.

템플릿을 사용하기 전에 템플릿 확인

템플릿을 사용하여 스택을 생성하거나 업데이트하기 전에 CloudFormation을 사용하여 템플릿을 확인합니다. CloudFormation에서 리소스를 생성하기 전에 템플릿을 확인하면 구문 및 일부 의미 체계 오류(예: 순환 종속성)를 파악할 수 있습니다. CloudFormation 콘솔을 사용하는 경우 입력 파라미터를 지정하면 콘솔에서 자동으로 템플릿을 확인합니다. AWS CLI 또는 CloudFormation API의 경우 [validate-template](#) CLI 명령 또는 [ValidateTemplate](#) API 작업을 사용합니다.

확인 중에 CloudFormation에서는 먼저 템플릿이 유효한 JSON인지를 확인합니다. 유효한 JSON이 아니면 CloudFormation은 템플릿이 유효한 YAML인지를 확인합니다. 두 확인이 모두 실패한 경우 CloudFormation에서는 템플릿 확인 오류를 반환합니다.

조직 정책 준수를 위한 템플릿 확인

템플릿을 확인하여 조직 정책 지침을 준수하는지 검증할 수도 있습니다. AWS CloudFormation Guard(cfn-guard)는 코드 정책 언어를 제공하여 필수 리소스 구성과 금지된 리소스 구성을 모두 확인할 수 있는 규칙을 정의하는 오픈 소스 명령줄 인터페이스(CLI) 도구입니다. 그런 다음 이러한 규칙에 대해 템플릿을 확인할 수 있습니다. 예를 들어, 관리자는 사용자가 항상 암호화된 Amazon S3 버킷을 생성하도록 보장하는 규칙을 생성할 수 있습니다.

템플릿을 편집하는 동안 cfn-guard를 로컬로 사용하거나 CI/CD 파이프라인의 일부로 자동 사용함으로써 규정 미준수 리소스의 배포를 방지할 수 있습니다.

또한 cfn-guard에는 기존 규정 준수 CloudFormation 템플릿에서 규칙을 추출할 수 있는 기능인 rulegen이 포함되어 있습니다,

자세한 내용은 GitHub에서 [cfn-guard](#) 리포지토리를 참조하세요.

템플릿 작성에 YAML 또는 JSON 사용

CloudFormation은 템플릿에 YAML 및 JSON 형식을 모두 지원합니다. 각 형식에는 장점이 있으며 선택은 특정 요구 사항에 따라 달라집니다.

다음과 같은 경우 YAML을 사용합니다.

- 인적 가독성 및 유지 관리 가능성을 우선시하는 경우
- 템플릿을 문서화하기 위한 설명을 포함하려는 경우
- 중첩 구조가 있는 복잡한 템플릿에서 작업하는 경우
- 반복을 줄이기 위해 앵커 및 별칭과 같은 YAML 관련 기능을 사용하려는 경우

다음과 같은 경우 JSON을 사용합니다.

- JSON을 선호하는 도구 또는 시스템과 통합해야 하는 경우
- 프로그래밍 방식 템플릿 생성 또는 조작을 사용하여 작업하는 경우
- 엄격한 데이터 검증이 필요한 경우

YAML은 일반적으로 가독성 및 설명 지원 때문에 수동 템플릿 작성에 권장됩니다. 들여쓰기 기반 구조가 리소스 계층 구조를 시각화하는 데 도움이 되는 복잡한 템플릿에 특히 유용합니다. JSON은 자동화된 워크플로에서 또는 JSON 입력이 필요한 API를 사용할 때 유리할 수 있습니다. 또한 특정 구조를 엄격하게 준수해야 하는 경우에도 유용합니다. 선택한 형식에 관계없이 잘 구성되고, 문서화되고, 유지 관리 가능한 템플릿을 만드는 데 중점을 둡니다. YAML을 사용하는 경우 앵커 및 별칭과 같은 기능을 활용하여 반복을 줄이고 유지 관리를 개선합니다.

포괄적인 태그 지정 전략 구현

CloudFormation 템플릿에서 생성한 모든 리소스에 대해 일관된 태그 지정 전략을 구현합니다. 태그는 리소스 구성, 비용 할당, 액세스 제어, 자동화에 도움이 됩니다. 환경, 소유자, 비용 센터, 애플리케이션 및 용도에 대한 태그를 포함하는 것이 좋습니다.

AWS::CloudFormation::Stack 리소스의 Tags 속성을 사용하여 스택에서 지원되는 모든 리소스에 태그를 적용합니다. 또한 여러 리소스 유형에 사용할 수 있는 TagSpecifications 속성을 사용하여 리소스 생성 중에 태그를 적용할 수 있습니다.

태그 지정에 대한 자세한 내용은 [리소스 태그](#)를 참조하세요.

고급 변환에 템플릿 매크로를 활용

CloudFormation 매크로를 사용하면 찾기 및 바꾸기 작업과 같은 간단한 작업부터 추가 리소스를 생성하는 복잡한 변환에 이르기까지 템플릿에서 사용자 지정 처리를 수행할 수 있습니다. 매크로를 사용하여 CloudFormation 템플릿의 기능을 확장하고 조직 전체에서 재사용 가능한 패턴을 구현합니다.

AWS Serverless Application Model은 서버리스 애플리케이션의 개발을 간소화하는 매크로의 예입니다. 조직별 패턴 및 요구 사항에 대한 사용자 지정 매크로를 생성하는 것이 좋습니다.

템플릿에서 매크로를 사용하는 방법에 대한 자세한 내용은 [CloudFormation 매크로 개요](#)를 참조하세요.

CloudFormation을 통해 모든 스택 리소스 관리

스택을 시작한 후에는 CloudFormation [콘솔](#), [API](#) 또는 [AWS CLI](#)를 사용하여 스택의 리소스를 업데이트합니다. CloudFormation 외부의 스택 리소스는 변경하지 마세요. 변경하면 스택 템플릿과 스택 리소스의 현재 상태 간에 불일치가 발생할 수 있으며 이로 인해 스택을 업데이트하거나 삭제할 경우 오류가 발생할 수 있습니다. 이를 드리프트라고 합니다. CloudFormation 템플릿 외부의 리소스가 변경되고 스택을 업데이트하면 리소스에 직접 적용된 변경 사항이 폐기되고 리소스 구성이 템플릿의 구성으로 되돌아갑니다.

자세한 내용은 [드리프트란 무엇인가요?](#) 섹션을 참조하세요.

스택 업데이트에 대한 자세한 내용은 [CloudFormation 스택 업데이트](#) 섹션을 참조하십시오.

스택을 업데이트하기 전에 변경 세트 생성

변경 세트를 사용하면 제안된 스택 변경 사항이 구현 전에 실행 중인 리소스에 어떤 영향을 미칠 수 있는지 확인할 수 있습니다. CloudFormation은 변경 세트를 실행할 때까지 스택을 변경하지 않으므로 제안된 변경을 계속 진행할지 아니면 다른 변경 세트를 생성할지 결정할 수 있습니다.

변경 세트를 사용하여 변경 사항이 실행 중인 리소스 특히, 중요 리소스에 미치는 영향을 확인합니다. 예를 들어 Amazon RDS 데이터베이스 인스턴스의 이름을 변경하면 CloudFormation에서는 새 데이터베이스가 생성되고 이전 데이터베이스는 삭제됩니다. 따라서 데이터베이스를 백업하지 않은 경우 이전 데이터베이스의 데이터가 손실됩니다. 변경 세트를 생성하면 변경 사항이 데이터베이스를 대체합니다. 따라서 스택을 업데이트하기 전에 계획을 세우기 좋습니다. 자세한 내용은 [변경 세트를 사용하여 CloudFormation 스택 업데이트](#) 섹션을 참조하세요.

스택 정책을 사용하여 리소스 보호

스택 정책을 사용하면 리소스 중단 또는 리소스 대체를 일으킬 수 있는 중요 스택 리소스의 의도치 않은 업데이트를 방지할 수 있습니다. 스택 정책은 지정된 리소스에 대해 수행할 수 있는 업데이트 작업을 설명하는 JSON 문서입니다. 중요 리소스가 있는 스택을 생성할 때마다 스택 정책을 지정합니다.

스택을 업데이트하는 동안 업데이트하려는 보호된 리소스를 명시적으로 지정해야 합니다. 그렇지 않으면 보호된 리소스가 변경되지 않습니다. 자세한 내용은 [스택 리소스에 대한 업데이트 방지](#) 섹션을 참조하세요.

CloudFormation 호출 기록에 AWS CloudTrail 사용

AWS CloudTrail은 AWS 계정에서 CloudFormation API를 직접 호출하는 모든 사용자를 추적합니다. CloudFormation API, CloudFormation 콘솔, 백엔드 콘솔 또는 CloudFormation AWS CLI 명령을 사용할 때마다 API 직접 호출이 로깅됩니다. 로깅을 활성화하고 로그를 저장할 S3 버킷을 지정합니다. 이렇게 하면 필요한 경우 계정에서 누가 어떤 CloudFormation 호출을 실행했는지 감사할 수 있습니다.

자세한 내용은 [AWS CloudTrail를 사용하여 CloudFormation API 호출 로깅](#) 섹션을 참조하세요.

코드 검토 및 버전 관리를 사용하여 사용자 템플릿 관리

스택 템플릿에서는 속성값과 같은 AWS 리소스의 구성을 설명합니다. 변경 사항을 검토하고 정확한 리소스 기록을 유지하려면 코드 검토 및 버전 관리를 사용합니다. 이러한 방법을 사용하면 템플릿의 여러 버전 간에 변경 사항을 추적할 수 있어 스택 리소스의 변경 사항을 추적할 수 있습니다. 또한 기록을 유지하면 언제든지 스택을 특정 템플릿 버전으로 되돌릴 수 있습니다.

정기적으로 Amazon EC2 인스턴스 업데이트

CloudFormation에서 생성된 모든 Amazon EC2 Windows 인스턴스 및 Amazon EC2 Linux 인스턴스에서 정기적으로 `yum update` 명령을 실행하여 RPM 패키지를 업데이트합니다. 이렇게 하면 최신 수정 사항 및 보안 업데이트를 받을 수 있습니다.

드리프트 감지를 정기적으로 사용

CloudFormation 드리프트 감지 기능을 정기적으로 사용하여 CloudFormation 관리 외부에서 수정된 리소스를 식별합니다. 드리프트를 감지하고 해결하면 코드형 인프라 접근 방식의 무결성을 유지하고 템플릿이 배포된 리소스의 상태를 정확하게 반영할 수 있습니다.

운영 절차의 일부로 자동 드리프트 감지를 구현하는 것이 좋습니다. Amazon EventBridge 규칙으로 트리거되는 AWS Lambda 함수를 사용하여 드리프트를 주기적으로 확인하고 불일치가 감지되면 팀에 알릴 수 있습니다.

드리프트에 대한 자세한 내용은 [드리프트 감지를 사용하여 스택 및 리소스에 대한 관리되지 않는 구성 변경 감지](#)를 참조하세요.

자동 복구를 위한 롤백 트리거 구성

롤백 트리거를 사용하여 스택 생성 및 업데이트 작업 중에 CloudFormation이 모니터링해야 하는 Amazon CloudWatch 경보를 지정합니다. 지정된 경보 중 하나라도 ALARM 상태가 되면 CloudFormation은 자동으로 전체 스택 작업을 롤백하여 인프라가 안정적인 상태를 유지하도록 합니다.

애플리케이션 오류율, 시스템 리소스 사용률, 애플리케이션 및 인프라의 상태를 나타내는 사용자 지정 비즈니스 지표와 같은 중요한 지표에 대한 롤백 트리거를 구성합니다.

롤백 트리거에 대한 자세한 내용은 [경보 위반 시 스택 롤백](#)을 참조하세요.

효과적인 스택 리팩터링 전략 구현

인프라가 변화함에 따라 유지 관리 가능성을 개선하거나, 복잡성을 줄이거나, 변화하는 요구 사항에 맞게 조정하기 위해 CloudFormation 스택을 리팩터링해야 할 수 있습니다. 스택 리팩터링에는 외부 동작과 기능을 유지하면서 템플릿 및 리소스를 재구성하는 작업이 포함됩니다. 스택 리팩터링은 CloudFormation에서 다음과 같은 방식으로 사용하면 유용합니다.

- 모놀리식 스택 분할: 크고 복잡한 스택을 수명 주기 또는 소유권을 기준으로 구성된 더 작고 관리 가능한 스택으로 분할
- 관련 리소스 통합: 여러 스택의 관련 리소스를 하나의 응집력 있는 스택으로 결합하여 관리를 간소화
- 재사용 가능한 구성 요소 추출: 재사용 및 일관성을 높이기 위해 일반적인 패턴을 모듈 또는 중첩 스택으로 이동
- 리소스 구성 개선: 관계 및 종속성을 더 잘 반영하도록 스택 내에서 리소스를 재구성

CloudFormation 스택 리팩터링에 대한 자세한 내용은 [스택 리팩터링](#)을 참조하세요.

수명 주기 관리에 CloudFormation Hooks 사용

CloudFormation Hooks는 프로비저닝 전에 AWS 리소스의 구성을 사전에 검사하고 복잡한 유효성 검사를 수행하는 코드를 제공합니다. Hooks는 리소스, 스택, 변경 세트가 조직의 보안, 운영 및 비용 최적화 요구 사항을 준수하는지 확인합니다. 리소스 프로비저닝 전에 경고를 제공하거나 구성 방식에 따라 작업을 실패시키고 완전히 중지합니다. 위반 및 경고는 Amazon CloudWatch에 로깅되어 규정 미준수 배포에 대한 가시성을 제공합니다.

이러한 Hooks 모범 사례에 대한 자세한 내용은 [AWS CloudFormation Hooks 개념](#)을 참조하세요.

Hooks가 CloudFormation 리소스에 수행할 수 있는 작업에 대한 자세한 내용은 [AWS CloudFormation Hooks란 무엇인가요?](#)를 참조하세요.

IaC 생성기를 사용하여 기존 리소스에서 템플릿 생성

CloudFormation IaC(코드형 인프라) 생성기를 사용하면 기존 AWS 리소스에서 CloudFormation 템플릿을 생성할 수 있습니다. 이 기능은 기존 인프라를 복제하거나, 수동으로 생성한 리소스를 문서화하거나, 비관리형 리소스를 CloudFormation 관리로 전환해야 할 때 특히 유용합니다. IaC 생성기는 다음과 같은 방법으로 CloudFormation 템플릿을 생성하는 데 유용합니다.

- 템플릿 생성 가속화: 템플릿을 처음부터 작성하는 대신 기존 리소스에서 생성
- 일관된 인프라: 생성된 템플릿을 시작점으로 사용하여 새 환경이 기존 환경과 일치하도록 보장
- 코드형 인프라로 마이그레이션: 수동으로 생성된 리소스를 점진적으로 CloudFormation 관리로 전환
- 문서화: 템플릿 형식으로 기존 인프라의 레코드 생성

IaC 생성기에 대한 자세한 내용은 [IaC 생성기를 사용하여 기존 리소스에서 템플릿 생성](#)을 참조하세요.

시각적 템플릿 설계에 AWS 인프라 컴포저 사용

AWS 인프라 컴포저는 끌어서 놓기 인터페이스를 사용하여 CloudFormation 템플릿을 생성, 시각화, 수정하는 데 도움이 되는 시각적 디자인 도구입니다. 이 도구는 다음과 같은 방법으로 CloudFormation을 사용할 때 특히 유용할 수 있습니다.

- 아키텍처 계획: 구현 전에 인프라 아키텍처를 설계 및 검증
- 템플릿 현대화: 기존 템플릿을 시각화하여 구조를 이해하고 개선 기회를 식별
- 교육 및 온보딩: 새로운 팀원이 시각적 학습을 통해 CloudFormation 개념과 AWS 서비스 관계를 이해하도록 지원

- 이해관계자 의사 소통: 명확한 시각적 표현을 사용하여 비기술 이해관계자에게 인프라 설계를 제시
- 규정 준수 검토: 시각적 다이어그램을 사용하여 인프라 설계의 보안 및 규정 준수 검토를 촉진
- 규정 준수 검토: 시각적 다이어그램을 사용하여 인프라 설계의 보안 및 규정 준수 검토를 촉진

Infrastructure Composer에 대한 자세한 내용은 [AWS 인프라 컴포저란 무엇인가요?](#)를 참조하세요.

복잡한 인프라에 AWS Cloud Development Kit (AWS CDK) 사용을 고려

복잡한 인프라 요구 사항의 경우 CDK를 사용하여 TypeScript, Python, Java, .NET 같은 친숙한 프로그래밍 언어를 사용하여 클라우드 리소스를 정의하는 것이 좋습니다. AWS CDK는 코드에서 CloudFormation 템플릿을 생성하므로 원하는 언어의 추상화 및 프로그래밍 구문을 사용하는 동시에 CloudFormation의 전체 기능을 활용할 수 있습니다.

AWS CDK는 모범 사례를 캡슐화하고 일반적인 인프라 패턴의 정의를 간소화하는 상위 수준 구문을 제공합니다. 이렇게 하면 인프라를 정의하는 데 필요한 코드를 크게 줄이는 동시에 모범 사례를 준수할 수 있습니다.

CDK에 대한 자세한 내용은 [AWS Cloud Development Kit \(AWS CDK\)](#) 섹션을 참조하세요.

IAM을 사용하여 액세스 제어

IAM은 AWS에서 사용자 및 사용자 권한을 관리하는 데 사용할 수 있는 AWS 서비스입니다.

CloudFormation에 IAM을 사용하여 스택 템플릿 보기, 스택 생성 또는 스택 삭제와 같은 사용자가 수행할 수 있는 CloudFormation 작업을 지정할 수 있습니다. 또한 CloudFormation 스택을 관리하는 사람에게서는 누구나 이러한 스택 내의 리소스에 대한 권한이 필요합니다. 예를 들어 CloudFormation을 사용하여 Amazon EC2 인스턴스를 시작, 업데이트 또는 종료하려는 경우 관련 Amazon EC2 작업을 호출하기 위한 권한이 있어야 합니다.

대부분의 경우 템플릿에서 모든 리소스를 관리하기 위해서는 전체 액세스 권한이 필요합니다.

CloudFormation은 사용자 대신 이러한 리소스를 생성, 수정 및 삭제하도록 호출합니다. 사용자와 CloudFormation 서비스 간에 권한을 분리하려면 서비스 역할을 사용합니다. CloudFormation에서는 사용자의 정책 대신 서비스 역할의 정책을 사용하여 호출합니다. 자세한 내용은 [CloudFormation 서비스 역할](#) 섹션을 참조하세요.

최소 권한 원칙을 적용

CloudFormation 서비스 역할 또는 템플릿이 생성한 리소스의 IAM 역할을 구성할 때는 항상 최소 권한 원칙을 적용합니다. 의도한 기능에 필요한 권한만 부여하고 가능한 한 와일드카드 권한을 사용하지 마세요.

IAM Access Analyzer를 사용하여 CloudFormation 서비스 역할에 부여된 권한을 검토하고 제거할 수 있는 미사용 권한을 식별합니다. IAM 정책을 정기적으로 검토 및 업데이트하여 보안 요구 사항에 부합하는지 확인합니다.

민감한 파라미터를 보호

암호, API 키 및 기타 비밀과 같은 민감한 정보의 경우 템플릿에 직접 임베딩하는 대신 AWS Systems Manager Parameter Store 또는 AWS Secrets Manager를 사용합니다. 템플릿에서 동적 참조를 사용하여 스택 작업 중에 이러한 값을 안전하게 검색합니다.

템플릿에서 파라미터를 사용하는 경우 민감한 파라미터의 값이 콘솔, API 응답 또는 CLI 출력에 표시되지 않도록 NoEcho 속성을 true로 설정합니다. NoEcho는 값이 값을 로깅할 수 있는 다른 서비스 또는 리소스로 전달되는 경우 값 로깅을 방지하지 않습니다.

CloudFormation에서 AWS Systems Manager Parameter Store를 사용하는 방법에 대한 자세한 내용은 [AWS Systems Manager Parameter Store에서 일반 텍스트 값 가져오기](#)를 참조하세요.

NoEcho 속성 사용에 대한 자세한 내용은 [CloudFormation 템플릿 파라미터 구문](#)을 참조하세요.

CloudFormation에서 AWS Secrets Manager를 사용하는 방법에 대한 자세한 내용은 [AWS Secrets Manager에서 AWS CloudFormation 비밀 생성](#)을 참조하세요.

AWS CloudFormation Guard를 사용하여 코드형 정책 구현

AWS CloudFormation Guard(cfn-guard)는 CloudFormation 템플릿에 대한 규칙을 정의하고 적용할 수 있는 오픈 소스 코드형 정책 도구입니다. cfn-guard를 사용하여 템플릿이 조직 정책, 보안 모범 사례, 거버넌스 요구 사항을 준수하는지 확인합니다.

cfn-guard를 CI/CD 파이프라인에 통합하여 템플릿을 배포 전에 정책 규칙에 대해 자동으로 검증합니다. 이를 통해 규정 미준수 리소스가 환경에 배포되는 것을 방지하고 개발자에게 정책 위반 피드백을 조기에 제공할 수 있습니다.

Guard에 대한 자세한 내용은 [AWS CloudFormation Guard란 무엇인가요?](#)를 참조하세요.

CloudFormation 템플릿 작업

AWS CloudFormation 템플릿은 스택의 일부로 생성, 업데이트 또는 삭제하려는 AWS 리소스를 정의합니다. 템플릿에는 여러 섹션이 있지만 [Resources](#) 섹션은 하나 이상의 리소스를 선언해야 하는 유일한 필수 섹션입니다.

다음 방법을 사용하여 템플릿을 생성할 수 있습니다.

- AWS 인프라 컴포저 - 템플릿 디자인을 위한 시각적 인터페이스입니다.
- 텍스트 편집기 - JSON 또는 YAML 구문으로 직접 템플릿을 작성합니다.
- IaC 생성기 - 계정에 프로비저닝된 리소스 중 현재 CloudFormation에서 관리하지 않는 리소스에서 템플릿을 생성합니다. IaC 생성기는 사용자 리전의 Cloud Control API에서 지원하는 다양한 리소스 유형과 함께 작동합니다.

이 섹션에서는 CloudFormation 템플릿의 다양한 섹션을 사용하는 방법과 스택 템플릿 생성을 시작하는 방법에 대한 포괄적인 가이드를 제공합니다. 다음과 같은 주제를 다룹니다.

주제

- [템플릿이 저장되는 위치](#)
- [템플릿 유효성 검사](#)
- [템플릿 시작하기](#)
- [샘플 템플릿](#)
- [CloudFormation 템플릿 형식](#)
- [CloudFormation 템플릿 섹션](#)
- [Infrastructure Composer를 사용하여 시각적으로 템플릿 생성](#)
- [AWS CloudFormation 언어 서버](#)
- [IaC 생성기를 사용하여 기존 리소스에서 템플릿 생성](#)
- [동적 참조를 사용하여 다른 서비스에 저장된 값 가져오기](#)
- [가상 파라미터를 사용하여 AWS 값 가져오기](#)
- [배포된 CloudFormation 스택에서 내보낸 출력 가져오기](#)
- [CloudFormation에서 제공하는 파라미터 유형을 사용하여 런타임 시 기존 리소스 지정](#)
- [CloudFormation 연습](#)
- [CloudFormation 템플릿 스니펫](#)

- [CloudFormation을 사용하여 Windows 기반 스택 배포](#)
- [CloudFormation에서 제공하는 리소스 유형을 사용하여 템플릿의 기능 확장](#)
- [CloudFormation 모듈을 사용하여 템플릿 전체에 포함할 수 있는 재사용 가능한 리소스 구성 생성](#)

템플릿이 저장되는 위치

Amazon S3 버킷

CloudFormation 템플릿을 Amazon S3 버킷에 저장할 수 있습니다. 스택을 생성하거나 업데이트할 때 직접 업로드하는 대신 템플릿의 S3 URL을 지정할 수 있습니다.

AWS Management Console 또는 AWS CLI를 통해 템플릿을 직접 업로드하는 경우 S3 버킷이 자동으로 생성됩니다. 자세한 내용은 [CloudFormation 콘솔에서 스택 생성](#) 섹션을 참조하세요.

Git 리포지토리

[Git 동기화](#)를 사용하면 Git 리포지토리에 템플릿을 저장할 수 있습니다. 스택을 생성하거나 업데이트할 때 직접 업로드하거나 S3 URL을 참조하는 대신 템플릿이 포함된 Git 리포지토리 위치와 브랜치를 지정할 수 있습니다. CloudFormation은 지정된 리포지토리 및 브랜치를 자동으로 모니터링하여 템플릿 변경을 확인합니다. 자세한 내용은 [Git 동기화를 사용하여 리포지토리 소스 코드에서 스택 생성](#) 섹션을 참조하세요.

템플릿 유효성 검사

구문 유효성 검사

[validate-template](#) CLI 명령을 사용하거나 콘솔에서 템플릿을 지정하여 템플릿의 JSON 또는 YAML 구문을 확인할 수 있습니다. 콘솔은 유효성 검사를 자동으로 수행합니다. 자세한 내용은 [CloudFormation 콘솔에서 스택 생성](#) 섹션을 참조하세요.

하지만 이러한 방법은 템플릿의 구문만 확인하고 리소스에 지정한 속성 값은 확인하지 않습니다.

추가 유효성 검사 도구

더 복잡한 유효성 검사 및 모범 사례 검사의 경우 다음과 같은 추가 도구를 사용할 수 있습니다.

- [CloudFormation Linter\(cfn-lint\)](#) - [CloudFormation 리소스 공급자 스키마](#)를 기준으로 템플릿 유효성을 검사합니다. 리소스 속성의 유효한 값과 모범 사례를 검사하는 것이 포함됩니다.

- [CloudFormation Rain\(rain fmt\)](#) - CloudFormation 템플릿을 일관된 표준으로 형식을 지정하거나 템플릿을 JSON에서 YAML로(또는 YAML에서 JSON으로) 형식을 재지정합니다. YAML을 사용할 때 주석을 보존하고 가능한 경우 내장 함수 사용을 짧은 구문으로 전환합니다.

템플릿 시작하기

CloudFormation 템플릿 생성을 시작하려면 다음 단계를 따릅니다.

1. 리소스 선택 - EC2 인스턴스, VPC, 보안 그룹 등과 같이 스택에 포함하려는 AWS 리소스를 식별합니다.
2. 템플릿 작성 - 리소스와 속성을 정의하여 JSON 또는 YAML 형식으로 템플릿을 작성합니다.
3. 템플릿 저장 - .json, .yaml, .txt 등의 파일 확장자를 사용하여 템플릿을 로컬에 저장합니다.
4. 템플릿 유효성 검사 - [템플릿 유효성 검사](#) 섹션에서 설명하는 방법을 사용하여 템플릿의 유효성을 검사합니다.
5. 스택 생성 - 유효성 검사가 완료된 템플릿을 사용하여 스택을 생성합니다.

CloudFormation 템플릿 참조 사용 계획

템플릿을 작성할 때 [AWS 리소스 및 속성 유형 참조](#)에서 다양한 리소스 유형에 대한 자세한 구문에 대한 설명서를 찾을 수 있습니다.

스택 템플릿에는 런타임까지 사용할 수 없는 속성 값을 할당하는 내재 함수와 리소스의 동작을 제어하는 특수 속성이 필요한 경우가 많습니다. 템플릿을 작성할 때 다음 리소스를 참조하여 지침을 확인하세요.

- [내장 함수 참조](#) - 일반적으로 사용되는 일부 내장 함수는 다음과 같습니다.
 - Ref - 리소스의 파라미터 또는 물리적 ID 값을 가져옵니다.
 - Sub - 문자열의 자리 표시자를 실제 값으로 대체합니다.
 - GetAtt - 템플릿의 리소스에서 속성 값을 반환합니다.
 - Join - 값 세트를 단일 문자열로 결합합니다.
- [리소스 속성 참조](#) - 일반적으로 사용되는 일부 특수 속성은 다음과 같습니다.
 - DependsOn - 이 속성을 사용하면 한 리소스 다음에 다른 리소스 하나를 생성하도록 지정할 수 있습니다.
 - DeletionPolicy - 이 속성을 사용하면 CloudFormation에서 리소스 삭제를 처리하는 방법을 지정할 수 있습니다.

샘플 템플릿

CloudFormation은 시작하는 데 사용할 수 있는 오픈 소스 스택 템플릿을 제공합니다. 자세한 내용은 GitHub 웹 사이트의 [CloudFormation Sample Templates](#)를 참조하세요.

이러한 템플릿은 프로덕션에 바로 사용할 수 있는 것이 아니라는 점에 유의하세요. 시간을 내어 이러한 프레임워크의 작동 방식을 알아보고, 필요에 맞게 조정하고, 회사의 규정 준수 기준을 충족하는지 확인해야 합니다.

이 리포지토리의 각 템플릿은 [CloudFormation Linter](#)(cfn-lint) 검사를 통과했으며, Center for Internet Security (CIS) Top 20를 기반으로 하는 기본 AWS CloudFormation Guard 규칙 세트(샘플을 단일 사용 사례에 집중하는 것이 합당한 일부 규칙 제외)도 통과했습니다.

CloudFormation 템플릿 형식

JSON 또는 YAML 형식으로 CloudFormation 템플릿을 작성할 수 있습니다. 두 형식 모두 용도는 같지만 가독성과 복잡성 측면에서 분명한 이점을 제공합니다.

- JSON - JSON은 시스템이 쉽게 구문 분석하고 생성할 수 있는 가벼운 데이터 교환 형식입니다. 하지만 특히 복잡한 구성의 경우 사람이 읽고 쓰는 데 번거로울 수 있습니다. JSON에서 템플릿은 중첩된 중괄호 {}와 대괄호 []를 사용하여 리소스, 파라미터 및 기타 구성 요소를 정의하는 구조입니다. 구문에서 모든 요소를 명시적으로 선언해야 하므로 템플릿이 장황해질 수 있지만 구조화된 형식을 엄격하게 준수할 수 있습니다.
- YAML - YAML은 JSON보다 사람이 읽기 쉬우면서 장황하지 않도록 설계되었습니다. 중첩을 나타내는 데 중괄호와 대괄호 대신 들여쓰기를 사용하므로 리소스 및 파라미터의 계층 구조를 보다 쉽게 시각화할 수 있습니다. YAML은 명확성과 사용 편의성 때문에 선호되는 경우가 많으며, 특히 복잡한 템플릿을 다룰 때 선호됩니다. 하지만 YAML이 들여쓰기에 의존하기에 띄어쓰기가 일정하지 않을 경우 오류가 발생할 수 있으므로 정확성을 유지하기 위해 세심한 주의를 기울여야 합니다.

템플릿 구조

CloudFormation 템플릿은 여러 섹션으로 나뉘며 각 섹션은 특정 유형의 정보를 유지하도록 설계되어 있습니다. 일부 섹션은 특정 순서로 선언해야 하지만 나머지 섹션은 순서가 중요하지 않습니다. 하지만 템플릿을 작성할 때 한 섹션의 값이 이전 섹션의 값을 참조할 수도 있으므로 다음 예제의 논리적 순서를 사용하는 것이 유용할 수 있습니다.

템플릿을 작성할 때 Resources 섹션과 같이 중복된 주요 섹션을 사용해서는 안 됩니다. CloudFormation은 이러한 템플릿을 수락할 수 있지만 템플릿을 처리할 때 정의되지 않은 동작을 하고 리소스를 잘못 프로비저닝하거나 설명할 수 없는 오류를 반환할 수 있습니다.

JSON

다음 예제에서는 사용 가능한 모든 섹션이 있는 JSON 형식의 템플릿 구조를 보여줍니다.

```
{
  "AWSTemplateFormatVersion" : "version date",

  "Description" : "JSON string",

  "Metadata" : {
    template metadata
  },

  "Parameters" : {
    set of parameters
  },

  "Rules" : {
    set of rules
  },

  "Mappings" : {
    set of mappings
  },

  "Conditions" : {
    set of conditions
  },

  "Transform" : {
    set of transforms
  },

  "Resources" : {
    set of resources
  },

  "Outputs" : {
    set of outputs
  }
```

```
}  
}
```

YAML

다음 예제에서는 사용 가능한 모든 섹션이 있는 YAML 형식의 템플릿 구조를 보여줍니다.

```
---  
AWSTemplateFormatVersion: version date  
  
Description:  
  String  
  
Metadata:  
  template metadata  
  
Parameters:  
  set of parameters  
  
Rules:  
  set of rules  
  
Mappings:  
  set of mappings  
  
Conditions:  
  set of conditions  
  
Transform:  
  set of transforms  
  
Resources:  
  set of resources  
  
Outputs:  
  set of outputs
```

설명

JSON 형식의 템플릿에서는 주석이 지원되지 않습니다. JSON은 설계상 주석 구문을 포함하지 않으므로 JSON 구조 내에서 직접 주석을 추가할 수 없습니다. 하지만 설명 노트나 문서를 포함해야 하는 경우 메타데이터 추가를 고려할 수 있습니다. 자세한 내용은 [Metadata 속성](#)을 참조하세요.

YAML 형식의 템플릿에서는 # 기호를 사용하여 인라인 주석을 포함할 수 있습니다.

다음 예제에서는 인라인 설명이 있는 YAML 템플릿을 보여 줍니다.

```
AWSTemplateFormatVersion: 2010-09-09
Description: A sample CloudFormation template with YAML comments.
# Resources section
Resources:
  MyEC2Instance:
    Type: AWS::EC2::Instance
    Properties:
      # Linux AMI
      ImageId: ami-1234567890abcdef0
      InstanceType: t2.micro
      KeyName: MyKey
      BlockDeviceMappings:
        - DeviceName: /dev/sdm
          Ebs:
            VolumeType: io1
            Iops: 200
            DeleteOnTermination: false
            VolumeSize: 20
```

사양

CloudFormation은 다음과 같은 JSON 및 YAML 사양을 지원합니다.

JSON

CloudFormation은 ECMA-404 JSON 표준을 따릅니다. JSON 형식에 대한 자세한 내용은 <http://www.json.org>를 참조하십시오.

YAML

CloudFormation은 몇 가지 예외가 있지만 YAML 버전 1.1 사양을 지원합니다. CloudFormation은 다음과 같은 기능을 지원하지 않습니다.

- binary, omap, pairs, set 및 timestamp 태그
- 예일리어스
- 해시 병합

YAML에 대한 자세한 내용은 <http://yaml.org/>를 참조하세요.

자세히 알아보기

템플릿에서 지정하는 각 리소스에 대해 JSON 또는 YAML의 특정 구문 규칙을 사용하여 해당 속성과 값을 정의합니다. 각 형식의 템플릿 구문에 대한 자세한 내용은 [CloudFormation 템플릿 섹션](#) 단원을 참조하십시오.

CloudFormation 템플릿에서 정규식 사용

템플릿 [파라미터](#)를 생성할 때 AllowedPattern 속성의 경우와 같이 CloudFormation 템플릿 내의 수많은 장소에서 정규식(일반적으로 regex라고 함)을 사용할 수 있습니다.

CloudFormation의 정규식은 Java regex 구문을 따릅니다. Java regex 구문과 그 구조에 대한 포괄적인 설명은 [java.util.regex.Pattern](#)을 참조하세요.

CloudFormation 템플릿을 JSON 구문으로 작성하는 경우 백슬래시를 추가하여 정규 표현식에서 백슬래시 문자(\)를 이스케이프해야 합니다. 이는 JSON이 백슬래시를 이스케이프 문자로 해석하기 때문이며, 정규식에서 백슬래시를 리터럴 백슬래시로 처리하려면 이스케이프해야 합니다.

예를 들어, 숫자 문자를 일치시키기 위해 정규식에 \d를 포함시키는 경우 JSON 템플릿에서는 \\d로 작성해야 합니다.

다음 예제에서 AllowedPattern 속성은 4자리 연속 문자(\d{4})와 일치하는 정규식을 지정합니다. 하지만 정규식은 JSON 템플릿에 정의되어 있으므로 추가 백슬래시(\\d)로 백슬래시 문자를 이스케이프해야 합니다.

```
{
  "Parameters": {
    "MyParameter": {
      "Type": "String",
      "AllowedPattern": "\\d{4}"
    }
  }
}
```

CloudFormation 템플릿을 YAML 구문으로 작성하는 경우 정규식을 작은따옴표(")로 묶어야 합니다. 추가 이스케이프는 필요하지 않습니다.

```
Parameters:
  MyParameter:
    Type: String
```



```
AllowedPattern: '\\d{4}'
```

Note

CloudFormation의 정규식은 AllowedPattern과 같은 특정 컨텍스트에서 검증 목적으로만 지원됩니다. 패턴 일치에 아닌 정확한 문자열 비교만 수행하는 Fn::Equals와 같은 CloudFormation 내장 함수에서는 패턴 일치 작업으로 지원되지 않습니다.

CloudFormation 템플릿 섹션

모든 CloudFormation 템플릿은 하나 이상의 섹션으로 구성되며, 각 섹션에는 특정한 목적이 있습니다.

Resources 섹션은 모든 CloudFormation 템플릿에 필요하고 템플릿의 핵심을 구성합니다. 이 섹션은 Amazon EC2 인스턴스 또는 Amazon S3 버킷 같은 스택 리소스 및 해당 속성을 지정합니다. 각 리소스는 고유한 논리적 ID, 유형 및 특정 구성 세부 정보로 정의됩니다.

Parameters 섹션은 선택 사항이지만 템플릿을 보다 유연하게 만드는 데 중요한 역할을 합니다. 이를 통해 사용자는 스택을 생성하거나 업데이트할 때 런타임에서 값을 전달할 수 있습니다. Resources 및 Outputs 섹션에서 이러한 파라미터를 참조할 수 있으므로 템플릿 자체를 변경하지 않고도 사용자 지정할 수 있습니다. 예를 들어, 파라미터를 사용하여 배포마다 달라지는 인스턴스 유형이나 환경 설정을 지정할 수 있습니다.

선택 사항이기도 한 Outputs 섹션에서는 스택의 속성을 볼 때 반환되는 값을 정의합니다. Outputs는 리소스 식별자 또는 URL과 같은 유용한 정보를 제공하며, 이러한 정보는 운영 목적이나 다른 스택과의 통합에 활용할 수 있습니다. 이 섹션을 통해 사용자는 템플릿에 의해 생성된 리소스에 대한 중요한 세부 정보를 검색하고 사용할 수 있습니다.

기타 선택 사항인 섹션에는 조건부 값을 관리하는 조회 테이블과 같은 기능을 하는 Mappings이 있습니다. 매핑을 사용하면 키-값 페어를 정의하고 Resources 및 Outputs 섹션의 Fn::FindInMap 내장 함수와 함께 이를 사용할 수 있습니다. 이는 AWS 리전 또는 환경 등의 조건에 따라 구성을 조정해야 하는 시나리오에 유용합니다.

Metadata 및 Rules 섹션은 자주 사용되지 않지만 추가 기능을 제공합니다. Metadata는 템플릿에 대한 추가 정보를 포함할 수 있고, Rules는 스택 생성 또는 업데이트 중에 파라미터 또는 파라미터 조합의 유효성을 검사하여 특정 기준을 충족하는지 확인합니다. Conditions 섹션에서는 환경 유형과 같은 조건에 따라 특정 리소스가 생성되는지 또는 속성에 값이 할당되는지 제어하여 유연성을 더욱 높입니다.

마지막으로 Transform 섹션은 템플릿 처리 중 매크로를 적용하는 데 사용됩니다. 서버리스 애플리케이션(Lambda 애플리케이션이라고도 함)의 경우 사용할 [AWS Serverless Application Model\(AWS SAM\)](#)의 버전을 지정합니다. 변환을 지정할 경우 AWS SAM 구문을 사용하여 템플릿에 리소스를 선언할 수 있습니다. 이 모델은 사용할 수 있는 구문과 처리 방식을 정의합니다. 또한 AWS::Include 변환을 사용하여 기본 CloudFormation 템플릿과는 별도로 저장되는 템플릿 스니펫을 포함시킬 수 있습니다.

다음 주제에서는 각 섹션 사용에 대한 자세한 정보와 예제를 제공합니다.

주제

- [CloudFormation 템플릿 Resources 구문](#)
- [CloudFormation 템플릿 Parameters 구문](#)
- [CloudFormation 템플릿 Outputs 구문](#)
- [CloudFormation 템플릿 Mappings 구문](#)
- [CloudFormation 템플릿 Metadata 구문](#)
- [CloudFormation 템플릿 Rules 구문](#)
- [CloudFormation 템플릿 Conditions 구문](#)
- [CloudFormation 템플릿 Transform 섹션](#)
- [CloudFormation 템플릿 형식 버전 구문](#)
- [CloudFormation 템플릿 Description 구문](#)

CloudFormation 템플릿 Resources 구문

Resources 섹션은 CloudFormation 템플릿의 필수 최상위 섹션입니다. CloudFormation에서 스택의 일부로 프로비저닝 및 구성하려는 AWS 리소스를 선언합니다.

구문

Resources 섹션에서는 다음 구문을 사용합니다.

JSON

```
"Resources" : {  
  "LogicalResourceName1" : {  
    "Type" : "AWS::ServiceName::ResourceType",  
    "Properties" : {
```

```

        "PropertyName1" : "PropertyValue1",
        ...
    }
},

"LogicalResourceName2" : {
    "Type" : "AWS::ServiceName::ResourceType",
    "Properties" : {
        "PropertyName1" : "PropertyValue1",
        ...
    }
}
}

```

YAML

```

Resources:
  LogicalResourceName1:
    Type: AWS::ServiceName::ResourceType
    Properties:
      PropertyName1: PropertyValue1
      ...

  LogicalResourceName2:
    Type: AWS::ServiceName::ResourceType
    Properties:
      PropertyName1: PropertyValue1
      ...

```

논리적 ID(논리명이라고도 함)

CloudFormation 템플릿 내에서 리소스는 논리적 이름으로 식별됩니다. 이 이름은 영숫자(A-Z, a-z, 0-9)여야 하며 템플릿 내에서 고유해야 합니다. 논리명은 템플릿의 다른 섹션에 있는 리소스를 참조하는 데 사용됩니다.

리소스 유형

각 리소스에는 Type 속성이 있어야 하며, 이 속성은 AWS 리소스의 종류를 정의합니다. Type 속성은 AWS::*ServiceName*::ResourceType 형식을 갖습니다. 예를 들어, Amazon S3 버킷의 Type 속성은 AWS::S3::Bucket입니다.

지원되는 리소스 유형의 전체 목록은 [AWS 리소스 및 속성 유형 참조](#)를 참조하세요.

리소스 속성

리소스 속성은 특정 리소스 유형에 대한 구성 세부 정보를 정의하기 위해 지정할 수 있는 추가 옵션입니다. 일부 속성은 필수이고, 선택 사항인 속성도 있습니다. 일부 속성에는 기본값이 있으므로 해당 속성을 지정하는 것은 선택 사항입니다.

각 리소스 유형에 지원되는 속성에 대한 자세한 내용은 [AWS 리소스 및 속성 유형 참조](#)의 주제를 참조하세요.

속성값은 리터럴 문자열, 문자열 목록, 부울, 동적 참조, 파라미터 참조, 가상 참조 또는 함수가 반환하는 값 중 하나일 수 있습니다. 다음 예제에서는 다양한 속성 값 유형을 선언하는 방법을 보여줍니다.

JSON

```
"Properties" : {  
  "String" : "A string value",  
  "Number" : 123,  
  "LiteralList" : [ "first-value", "second-value" ],  
  "Boolean" : true  
}
```

YAML

```
Properties:  
  String: A string value  
  Number: 123  
  LiteralList:  
    - first-value  
    - second-value  
  Boolean: true
```

물리적 ID

논리적 ID 외에도, 특정 리소스에는 EC2 인스턴스 ID 또는 S3 버킷 이름 같은 해당 리소스에 대해 실제 할당된 이름인 물리적 ID도 지정됩니다. 물리적 ID를 사용하여 CloudFormation 템플릿 외부에 있는 리소스를 식별할 수 있지만 이는 리소스가 생성된 후에만 가능합니다. 예를 들어 EC2 인스턴스에 논리적 ID MyEC2Instance의 리소스를 제공한다고 가정하겠습니다. CloudFormation에서 인스턴스를 생성할 때 CloudFormation은 자동적으로 물리적 ID(예: i-1234567890abcdef0)를 생성하여 인스턴스에 할당합니다. 이 물리적 ID를 사용하면 Amazon EC2 콘솔에서 인스턴스를 식별하고 인스턴스의 속성(예: DNS 이름)을 볼 수 있습니다.

Amazon S3 버킷과 기타 여러 리소스의 경우 사용자가 명시적으로 지정하지 않으면 CloudFormation 이 리소스에 대한 고유한 물리명을 자동으로 생성합니다. 이 물리명은 CloudFormation 스택의 이름, CloudFormation 템플릿에 지정된 리소스의 논리명, 고유 ID의 조합을 기반으로 합니다. 예를 들어, MyStack이라는 스택에 논리명이 MyBucket인 Amazon S3 버킷이 있는 경우 CloudFormation은 MyStack-MyBucket-abcdefghijkl1이라는 물리적 이름으로 버킷의 이름을 지정할 수 있습니다.

사용자 지정 이름을 지원하는 리소스의 경우, 물리적 이름을 할당하면 리소스를 보다 신속하게 식별할 수 있습니다. 예를 들면 로그를 저장하는 S3 버킷에 MyPerformanceLogs 이름을 지정할 수 있습니다. 자세한 내용은 [이름 유형](#)을 참조하세요.

참조 리소스

다른 리소스의 이름이나 속성을 기반으로 한 리소스의 속성을 설정해야 하는 경우가 있습니다. 예를 들어, S3 버킷이 지원하는 CloudFront 배포 또는 EC2 보안 그룹을 사용하는 EC2 인스턴스를 생성할 수 있습니다. 이러한 모든 리소스는 동일한 CloudFormation 템플릿에서 생성할 수 있습니다.

CloudFormation은 다른 리소스 및 해당 속성을 참조하는 데 사용할 수 있는 내장 함수를 제공합니다. 이러한 함수를 사용하면 리소스 간에 종속성을 생성하고 한 리소스에서 다른 리소스로 값을 전달할 수 있습니다.

Ref 함수

Ref 함수는 대체로 동일한 CloudFormation 템플릿 내에 정의된 리소스의 식별 속성을 검색하는 데 사용됩니다. 반환되는 내용은 리소스 유형에 따라 달라집니다. 대부분의 리소스는 리소스의 물리명을 반환합니다. 하지만 일부 리소스 유형의 경우 AWS::EC2::EIP 리소스의 IP 주소나 Amazon SNS 주제의 Amazon 리소스 이름(ARN) 등의 다른 값을 반환할 수 있습니다.

다음 예제는 속성에서 Ref 함수의 사용 방법을 설명합니다. 각 예제에서 Ref 함수는 템플릿의 다른 곳에서 선언된 LogicalResourceName 리소스의 실제 이름을 반환합니다. YAML 예제의 !Ref 구문 예제는 Ref 함수를 보다 간단하게 작성하는 방법입니다.

JSON

```
"Properties" : {  
  "PropertyName" : { "Ref" : "LogicalResourceName" }  
}
```

YAML

```
Properties:
```

```

PropertyName1:
  Ref: LogicalResourceName
PropertyName2: !Ref LogicalResourceName

```

Ref 함수에 대한 더 자세한 내용은 [Ref](#)를 참조하세요.

Fn::GetAtt 함수

Ref 함수는 리소스에 대해 반환되는 파라미터나 값이 정확히 원하는 값인 경우에 유용합니다. 그러나 리소스의 다른 속성이 필요할 수 있습니다. 예를 들어 S3 오리진을 사용하여 CloudFront 배포를 생성하려는 경우 DNS 스타일 주소를 사용하여 버킷 위치를 지정해야 합니다. 일부 리소스에는 템플릿에서 사용할 수 있는 값이 지정되는 추가 속성이 있습니다. 이러한 속성을 가져오려면 Fn::GetAtt 함수를 사용합니다.

다음 예제는 속성에서 GetAtt 함수의 사용 방법을 설명합니다. Fn::GetAtt 함수는 두 개 파라미터, 리소스의 논리적 이름, 그리고 검색할 속성의 이름을 사용합니다. YAML 예제의 !GetAtt 구문 예제는 GetAtt 함수를 보다 간단하게 작성하는 방법입니다.

JSON

```

"Properties" : {
  "PropertyName" : {
    "Fn::GetAtt" : [ "LogicalResourceName", "AttributeName" ]
  }
}

```

YAML

```

Properties:
  PropertyName1:
    Fn::GetAtt:
      - LogicalResourceName
      - AttributeName
  PropertyName2: !GetAtt LogicalResourceName.AttributeName

```

GetAtt 함수에 대한 더 자세한 내용은 [Fn::GetAtt](#)를 참조하세요.

예제

다음 예제는 리소스를 선언하는 방법과 CloudFormation 템플릿이 동일한 템플릿 내에 정의된 다른 리소스와 기존 AWS 리소스를 참조하는 방법을 보여줍니다.

주제

- [사용자 지정 이름을 사용하여 단일 리소스 선언](#)
- [Ref 함수로 다른 리소스 참조](#)
- [Fn::GetAtt 함수를 사용하여 리소스 속성 참조](#)

사용자 지정 이름을 사용하여 단일 리소스 선언

다음 예제에서는 논리명이 MyBucket인 AWS::S3::Bucket 유형의 단일 리소스를 선언합니다. BucketName 속성이 *amzn-s3-demo-bucket*으로 설정되는데, 이를 원하는 S3 버킷 이름으로 바꿔야 합니다.

이 리소스 선언을 사용하여 스택을 생성할 경우 CloudFormation에서는 기본 설정으로 Amazon S3 버킷을 생성합니다. Amazon EC2 인스턴스나 Auto Scaling 그룹 같은 기타 리소스의 경우 CloudFormation은 추가 정보를 요구합니다.

JSON

```
{
  "Resources": {
    "MyBucket": {
      "Type": "AWS::S3::Bucket",
      "Properties": {
        "BucketName": "amzn-s3-demo-bucket"
      }
    }
  }
}
```

YAML

```
Resources:
  MyBucket:
    Type: AWS::S3::Bucket
    Properties:
      BucketName: amzn-s3-demo-bucket
```

Ref 함수로 다른 리소스 참조

다음 예제는 EC2 인스턴스와 보안 그룹을 정의하는 리소스 선언을 보여줍니다. Ec2Instance 리소스는 Ref 함수를 사용하여 해당 SecurityGroupIds 속성의 일부로 InstanceSecurityGroup 리소

스를 참조합니다. 템플릿에 선언되지 않은 기존 보안 그룹(sg-12a4c434)도 포함됩니다. 리터럴 문자열을 사용하여 기존 AWS 리소스를 참조합니다.

JSON

```
{
  "Resources": {
    "Ec2Instance": {
      "Type": "AWS::EC2::Instance",
      "Properties": {
        "SecurityGroupIds": [
          {
            "Ref": "InstanceSecurityGroup"
          },
          "sg-12a4c434"
        ],
        "KeyName": "MyKey",
        "ImageId": "ami-1234567890abcdef0"
      }
    },
    "InstanceSecurityGroup": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "GroupDescription": "Enable SSH access via port 22",
        "SecurityGroupIngress": [
          {
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIp": "0.0.0.0/0"
          }
        ]
      }
    }
  }
}
```

YAML

```
Resources:
  Ec2Instance:
    Type: AWS::EC2::Instance
    Properties:
```



```

    SecurityGroupIds:
      - !Ref InstanceSecurityGroup
      - sg-12a4c434
    KeyName: MyKey
    ImageId: ami-1234567890abcdef0
  InstanceSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Enable SSH access via port 22
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          CidrIp: 0.0.0.0/0

```

Fn::GetAtt 함수를 사용하여 리소스 속성 참조

다음 예제에서는 CloudFront 배포 리소스와 S3 버킷을 정의하는 리소스 선언을 보여줍니다.

MyDistribution 리소스는 버킷의 DomainName 속성을 가져오는 Fn::GetAtt 함수를 사용하여 MyBucket 리소스의 DNS 이름을 지정합니다. Fn::GetAtt 함수는 두 파라미터를 어레이 형태로 나열합니다. 여러 파라미터를 사용하는 함수의 경우 어레이를 사용하여 파라미터를 지정합니다.

JSON

```

{
  "Resources": {
    "MyBucket": {
      "Type": "AWS::S3::Bucket"
    },
    "MyDistribution": {
      "Type": "AWS::CloudFront::Distribution",
      "Properties": {
        "DistributionConfig": {
          "Origins": [
            {
              "DomainName": {
                "Fn::GetAtt": [
                  "MyBucket",
                  "DomainName"
                ]
              },
              "Id": "MyS3origin",
              "S3originConfig": {}
            }
          ]
        }
      }
    }
  }
}

```


CloudFormation 템플릿 Parameters 구문

선택 사항인 Parameters 단원을 사용하여 템플릿을 사용자 지정합니다. 파라미터를 통해 스택을 생성하거나 업데이트 할 때마다 템플릿에 사용자 지정 값을 입력할 수 있습니다. 템플릿에 파라미터를 사용하여 특정 시나리오에 맞게 조정할 수 있는 재사용 가능하고 유연한 템플릿을 만들 수 있습니다.

적절한 유형의 파라미터를 정의하여 콘솔로 스택을 생성할 때 기존 리소스의 식별자 목록에서 선택할 수 있습니다. 자세한 내용은 [CloudFormation에서 제공하는 파라미터 유형을 사용하여 런타임 시 기존 리소스 지정](#) 섹션을 참조하세요.

파라미터는 스택 리소스의 속성 값을 지정하는 데 주로 사용되는 방법입니다. 그러나 리전에 따라 달라지는 설정이 있거나 다른 조건이나 종속성으로 인해 사용자가 파악하기 어려운 설정이 있을 수 있습니다. 이러한 경우 원하는 결과를 얻기 위해 보다 간단한 값을 지정할 수 있도록(또는 전혀 지정할 필요가 없도록) 템플릿에 몇 가지 논리를 추가하는 것이 좋습니다(예: 매핑 사용). 자세한 내용은 [CloudFormation 템플릿 Mappings 구문](#) 섹션을 참조하세요.

구문

다음 일반 구문을 사용하는 템플릿의 Parameters 섹션에서 파라미터를 선언합니다.

JSON

```
"Parameters" : {  
  "ParameterLogicalID" : {  
    "Description": "Information about the parameter",  
    "Type" : "DataType",  
    "Default" : "value",  
    "AllowedValues" : ["value1", "value2"]  
  }  
}
```

YAML

```
Parameters:  
  ParameterLogicalID:  
    Description: Information about the parameter  
    Type: DataType  
    Default: value  
    AllowedValues:  
      - value1  
      - value2
```

파라미터에는 해당 값과 값에 대한 구속 조건을 정의하는 속성 목록이 포함됩니다. 유일한 필수 속성은 Type(String, Number 또는 CloudFormation에서 제공하는 파라미터 유형일 수 있음)입니다. 사용자가 지정해야 하는 값의 종류에 대해 설명하는 Description 속성을 추가할 수도 있습니다. 파라미터의 이름과 설명은 스택 생성 마법사에서 템플릿을 사용할 때 파라미터 지정 페이지에 나타납니다.

Note

기본적으로 CloudFormation 콘솔에는 입력 파라미터가 논리적 ID에 따라 영문자순으로 나열됩니다. 이 기본 순서를 재정의하고 관련 파라미터를 함께 그룹화하려면 템플릿의 `AWS::CloudFormation::Interface` 메타데이터 키를 사용할 수 있습니다. 자세한 내용은 [AWS::CloudFormation::Interface 메타데이터를 사용하여 CloudFormation 파라미터 구성](#) 섹션을 참조하세요.

기본값이 있는 파라미터의 경우 CloudFormation에서는 사용자가 다른 값을 지정하지 않을 경우 기본값이 사용됩니다. 기본 속성을 생략하는 경우 사용자가 해당 파라미터에 대한 값을 지정해야 합니다. 그러나 사용자에게 값을 입력하도록 요구한다고 해서 해당 값의 유효성이 보장되는 것은 아닙니다. 파라미터 값을 검증하려는 경우 제약을 선언하거나 AWS 특정 유형을 지정할 수 있습니다.

기본값이 없는 파라미터의 경우 사용자는 스택 생성 시 키 이름 값을 지정해야 합니다. 지정하지 않을 경우 CloudFormation에서 스택을 생성할 수 없으며 다음 예외가 발생합니다.

```
Parameters: [KeyName] must have values
```

속성

AllowedPattern

String 또는 CommaDelimitedList 유형에 허용할 패턴을 나타내는 정규식입니다.

String 파라미터 유형에 적용할 경우 패턴이 제공된 전체 파라미터값과 일치해야 합니다.

CommaDelimitedList 파라미터 유형에 적용할 경우 패턴이 목록의 각 값과 일치해야 합니다.

필수 항목 여부: 아니요

AllowedValues

파라미터에 허용되는 값 목록을 포함하는 어레이입니다. String 파라미터 유형에 적용할 경우 파라미터값은 허용된 값 중 하나여야 합니다. CommaDelimitedList 파라미터 유형에 적용할 경우 목록의 각 값은 지정된 허용된 값 중 하나여야 합니다.

필수 항목 여부: 아니요

 Note

YAML을 사용하면서 AllowedValues에 Yes 및 No 문자열을 사용하려는 경우 작은따옴표를 사용하여 YAML 구문 분석기가 이러한 부울 값을 고려하지 않도록 합니다.

ConstraintDescription

제약 위반 시 해당 제약을 설명하는 문자열입니다. 예를 들어 제약 설명이 없으면 [A-Za-z0-9]+ 패턴이 허용된 파라미터에 사용자가 유효하지 않은 값을 지정할 때 다음과 같은 오류 메시지가 표시됩니다.

```
Malformed input-Parameter MyParameter must match pattern [A-Za-z0-9]+
```

'문자(대문자 및 소문자)와 숫자만 포함시킬 수 있음' 같은 제약을 추가, 다음과 같은 사용자 지정 오류 메시지를 표시할 수 있습니다.

```
Malformed input-Parameter MyParameter must only contain uppercase and lowercase letters and numbers
```

필수 항목 여부: 아니요

Default

스택 생성 시 지정된 값이 없는 경우에 사용할 템플릿에 적합한 유형의 값입니다. 파라미터에 대한 제약을 정의하는 경우 이러한 제약을 준수하는 값을 지정해야 합니다.

필수 항목 여부: 아니요

Description

파라미터를 설명하는 최대 4000자 문자열입니다.

필수 항목 여부: 아니요

MaxLength

String 유형에 허용할 최대 문자 수를 결정하는 정수 값입니다.

필수 항목 여부: 아니요

MaxValue

Number 유형에 허용할 최대 숫자 값을 결정하는 숫자 값입니다.

필수 항목 여부: 아니요

MinLength

String 유형에 허용할 최소 문자 수를 결정하는 정수 값입니다.

필수 항목 여부: 아니요

MinValue

Number 유형에 허용할 최소 숫자 값을 결정하는 숫자 값입니다.

필수 항목 여부: 아니요

NoEcho

콘솔, 명령줄 도구 또는 API에 표시되지 않도록 파라미터값을 마스킹 처리할지 여부입니다.

NoEcho 속성을 true로 설정한 경우, 아래 지정된 위치에 저장된 정보를 제외하고 CloudFormation은 스택 또는 스택 이벤트를 설명하는 모든 호출에 대해 별표(****)로 마스킹 처리된 파라미터값을 반환합니다.

필수 항목 여부: 아니요

Important

NoEcho 속성을 사용해도 다음에 저장된 정보는 마스킹되지 않습니다.

- Metadata 템플릿 섹션. CloudFormation은 Metadata 섹션에 포함된 정보를 변환, 수정 또는 삭제하지 않습니다. 자세한 내용은 [메타데이터](#)를 참조하십시오.
- Outputs 템플릿 섹션. 자세한 내용은 [출력](#)을 참조하십시오.
- 리소스 정의의 Metadata 속성입니다. 자세한 내용은 [Metadata 속성](#)을 참조하세요. 이러한 메커니즘을 사용하여 암호나 보안 정보와 같은 중요한 정보를 포함하지 않는 것이 좋습니다.

Important

AWS Systems Manager Parameter Store 또는 AWS Secrets Manager와 같이 CloudFormation 외부에서 저장 및 관리되는 중요한 정보를 참조하려면 CloudFormation 템

플릿에 직접 중요한 정보를 포함하는 대신 스택 템플릿에 있는 동적 파라미터를 사용하는 것이 좋습니다.

자세한 내용은 [템플릿에 자격 증명을 포함하지 않음](#)을 참조하십시오.

Important

리소스의 기본 식별자의 일부인 리소스 속성에 NoEcho 파라미터 또는 민감한 데이터를 포함하지 않는 것이 좋습니다.

기본 리소스 식별자를 구성하는 속성에 NoEcho 파라미터가 포함된 경우 CloudFormation은 기본 리소스 식별자에서 실제 일반 텍스트값을 사용할 수 있습니다. 이 리소스 ID는 파생된 출력 또는 대상에 나타날 수 있습니다.

리소스 유형의 기본 식별자를 구성하는 리소스 속성을 확인하려면 [AWS 리소스 및 속성 유형 참조](#)에서 해당 리소스에 대한 리소스 참조 설명서를 참조하세요. Return values(반환 값) 섹션에서 Ref 함수 반환 값은 리소스 유형의 기본 식별자를 구성하는 리소스 속성을 나타냅니다.

Type

파라미터(DataType)의 데이터 유형입니다.

필수 항목 여부: 예

CloudFormation은 다음과 같은 파라미터 유형을 지원합니다.

String

리터럴 문자열입니다. MinLength, MaxLength, Default, AllowedValues 및 AllowedPattern 속성을 사용하여 제약 조건을 선언할 수 있습니다.

예를 들면 사용자는 "MyUserName"을 지정할 수 있습니다.

Number

정수 또는 부동 소수점. CloudFormation에서 파라미터값이 숫자로 확인되지만, 템플릿 내 다른 곳에서 이 파라미터를 사용하는 경우(예를 들어, Ref 내장 함수를 사용하여) 파라미터값이 문자열이 됩니다.

MinValue, MaxValue, Default 및 AllowedValues 속성을 사용하여 제약 조건을 선언할 수 있습니다.

예를 들면 사용자는 "8888"을 지정할 수 있습니다.

List<Number>

쉼표로 구분된 정수 또는 부동 소수점의 배열. CloudFormation에서 파라미터값이 숫자로 확인되지만, 템플릿 내 다른 곳에서 이 파라미터를 사용하는 경우(예를 들어, Ref 내장 함수를 사용하여) 파라미터값이 문자열 목록이 됩니다.

예를 들어 사용자가 "80,20"을 지정하면 Ref는 ["80","20"]이 됩니다.

CommaDelimitedList

쉼표로 구분된 리터럴 문자열 어레이입니다. 총 문자열 수는 총 쉼표 수보다 하나 더 많아야 합니다. 또한 각 멤버 문자열에는 공백이 삽입됩니다.

예를 들어 사용자가 "test,dev,prod"을 지정하면 Ref는 ["test","dev","prod"]이 됩니다.

AWS 특정 파라미터 유형

Amazon EC2 키 페어 이름 및 VPC ID와 같은 AWS 값입니다. 자세한 내용은 [런타임 시 기존 리소스 지정](#) 섹션을 참조하세요.

Systems Manager 파라미터 유형

시스템 관리자 Parameter Store의 기존 파라미터에 해당되는 파라미터. Systems Manager 파라미터 키를 Systems Manager 파라미터 유형 값으로 지정합니다. 그러면 CloudFormation이 Parameter Store에서 스택에 사용할 최신 값을 검색합니다. 자세한 내용은 [런타임 시 기존 리소스 지정](#) 섹션을 참조하세요.

파라미터의 일반 요구 사항

파라미터를 사용할 때는 다음 요구 사항이 적용됩니다.

- CloudFormation 템플릿에는 최대 200개의 파라미터를 지정할 수 있습니다.
- 파라미터마다 논리명(논리적 ID)을 지정해야 합니다. 이 이름은 템플릿 내 모든 논리명 간에 고유해야 하며 영숫자로 지정해야 합니다.
- 파라미터마다 CloudFormation에서 지원하는 파라미터 유형을 할당해야 합니다. 자세한 내용은 [유형을 참조하십시오](#).
- 파라미터마다 런타임에 값을 할당해야 CloudFormation이 스택을 성공적으로 프로비저닝할 수 있습니다. 다른 값이 제공되는 경우를 제외하고, CloudFormation에 대한 기본값을 선택적으로 지정할 수 있습니다.

- 파라미터는 동일한 템플릿에서 선언하고 참조해야 합니다. 템플릿의 Resources 및 Outputs의 파라미터를 참조할 수 있습니다.

예제

주제

- [단순 문자열 파라미터](#)
- [암호 파라미터](#)
- [파라미터 참조](#)
- [선택으로 구분된 목록 파라미터](#)
- [선택으로 구분된 목록 파라미터에서 값 반환](#)

단순 문자열 파라미터

다음 예제는 String 유형의 InstanceTypeParameter라는 파라미터를 선언합니다. 스택의 Amazon EC2 인스턴스 유형을 지정할 수 있는 파라미터입니다. 스택 생성 또는 업데이트 중 값이 제공되지 않으면 CloudFormation은 기본값인 t2.micro를 사용합니다.

JSON

```
"Parameters" : {
  "InstanceTypeParameter" : {
    "Description" : "Enter t2.micro, m1.small, or m1.large. Default is t2.micro.",
    "Type" : "String",
    "Default" : "t2.micro",
    "AllowedValues" : ["t2.micro", "m1.small", "m1.large"]
  }
}
```

YAML

```
Parameters:
  InstanceTypeParameter:
    Description: Enter t2.micro, m1.small, or m1.large. Default is t2.micro.
    Type: String
    Default: t2.micro
    AllowedValues:
      - t2.micro
```

- m1.small
- m1.large

암호 파라미터

다음 예제는 기본값 없이 String 유형의 DBPwd라는 파라미터를 선언합니다. NoEcho 속성을 true로 설정하면 파라미터값이 스택 설명에 표시되지 않습니다. 지정할 수 있는 최소 길이는 1이고, 지정할 수 있는 최대 길이는 41입니다. 이 패턴에는 영문자 소문자/대문자와 숫자가 허용됩니다. 이 예제에서는 AllowedPattern 속성에 정규식을 사용하는 방법도 보여줍니다.

JSON

```
"Parameters" : {
  "DBPwd" : {
    "NoEcho" : "true",
    "Description" : "The database admin account password",
    "Type" : "String",
    "MinLength" : "1",
    "MaxLength" : "41",
    "AllowedPattern" : "^[a-zA-Z0-9]*$"
  }
}
```

YAML

```
Parameters:
  DBPwd:
    NoEcho: true
    Description: The database admin account password
    Type: String
    MinLength: 1
    MaxLength: 41
    AllowedPattern: ^[a-zA-Z0-9]*$
```

파라미터 참조

Ref 내장 함수를 사용하여 파라미터를 참조할 수 있고, CloudFormation은 스택 프로비저닝에 이 파라미터값을 사용합니다. 동일한 템플릿의 Resources 및 Outputs의 파라미터를 참조할 수 있습니다.

다음은 EC2 인스턴스의 InstanceType 속성이 InstanceTypeParameter 파라미터값을 참조하는 예제입니다.

JSON

```
"Ec2Instance" : {
  "Type" : "AWS::EC2::Instance",
  "Properties" : {
    "InstanceType" : { "Ref" : "InstanceTypeParameter" },
    "ImageId" : "ami-0ff8a91507f77f867"
  }
}
```

YAML

```
Ec2Instance:
  Type: AWS::EC2::Instance
  Properties:
    InstanceType:
      Ref: InstanceTypeParameter
    ImageId: ami-0ff8a91507f77f867
```

쉼표로 구분된 목록 파라미터

CommaDelimitedList 파라미터 유형은 단일 속성에 여러 값을 제공해야 할 때 유용할 수 있습니다. 다음 예제는 쉼표로 구분된 3개의 CIDR 블록을 기본값으로 사용하여 DbSubnetIpBlocks라는 파라미터를 선언합니다.

JSON

```
"Parameters" : {
  "DbSubnetIpBlocks": {
    "Description": "Comma-delimited list of three CIDR blocks",
    "Type": "CommaDelimitedList",
    "Default": "10.0.48.0/24, 10.0.112.0/24, 10.0.176.0/24"
  }
}
```

YAML

```
Parameters:
  DbSubnetIpBlocks:
    Description: "Comma-delimited list of three CIDR blocks"
    Type: CommaDelimitedList
```

Default: "10.0.48.0/24, 10.0.112.0/24, 10.0.176.0/24"

쉼표로 구분된 목록 파라미터에서 값 반환

파라미터의 쉼표로 구분된 목록에서 특정 값을 참조하려면 템플릿의 Resources 섹션에서 Fn::Select 내장 함수를 사용합니다. 다음 예제와 같이 원하는 객체의 인덱스 값과 객체 목록을 전달합니다.

JSON

```
{
  "Parameters": {
    "VPC": {
      "Type": "String",
      "Default": "vpc-123456"
    },
    "VpcAzs": {
      "Type": "CommaDelimitedList",
      "Default": "us-west-2a, us-west-2b, us-west-2c"
    },
    "DbSubnetIpBlocks": {
      "Type": "CommaDelimitedList",
      "Default": "172.16.0.0/26, 172.16.0.64/26, 172.16.0.128/26"
    }
  },
  "Resources": {
    "DbSubnet1": {
      "Type": "AWS::EC2::Subnet",
      "Properties": {
        "AvailabilityZone": {
          "Fn::Select": [
            0,
            {
              "Ref": "VpcAzs"
            }
          ]
        },
        "VpcId": {
          "Ref": "VPC"
        },
        "CidrBlock": {
          "Fn::Select": [
            0,
```

```

        { "Ref": "DbSubnetIpBlocks" }
      ]
    }
  },
  "DbSubnet2": {
    "Type": "AWS::EC2::Subnet",
    "Properties": {
      "AvailabilityZone": {
        "Fn::Sub": [
          "${AWS::Region}${AZ}",
          {
            "AZ": {
              "Fn::Select": [
                1,
                { "Ref": "VpcAzs" }
              ]
            }
          }
        ]
      },
      "VpcId": {
        "Ref": "VPC"
      },
      "CidrBlock": {
        "Fn::Select": [
          1,
          { "Ref": "DbSubnetIpBlocks" }
        ]
      }
    }
  },
  "DbSubnet3": {
    "Type": "AWS::EC2::Subnet",
    "Properties": {
      "AvailabilityZone": {
        "Fn::Sub": [
          "${AWS::Region}${AZ}",
          {
            "AZ": {
              "Fn::Select": [
                2,
                { "Ref": "VpcAzs" }
              ]
            }
          }
        ]
      }
    }
  }
}

```

```
    }  
  ]  
},  
"VpcId": {  
  "Ref": "VPC"  
},  
"CidrBlock": {  
  "Fn::Select": [  
    2,  
    { "Ref": "DbSubnetIpBlocks" }  
  ]  
}  
}  
}  
}
```

YAML

```
Parameters:
  VPC:
    Type: String
    Default: vpc-123456
  VpcAzs:
    Type: CommaDelimitedList
    Default: us-west-2a, us-west-2b, us-west-2c
  DbSubnetIpBlocks:
    Type: CommaDelimitedList
    Default: 172.16.0.0/26, 172.16.0.64/26, 172.16.0.128/26

Resources:
  DbSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      AvailabilityZone: !Select
        - 0
        - !Ref VpcAzs
      VpcId: !Ref VPC
      CidrBlock: !Select
        - 0
        - !Ref DbSubnetIpBlocks
  DbSubnet2:
    Type: AWS::EC2::Subnet
```

```

Properties:
  AvailabilityZone: !Sub
    - ${AWS::Region}${AZ}
    - AZ: !Select
      - 1
      - !Ref VpcAzs
  VpcId: !Ref VPC
  CidrBlock: !Select
    - 1
    - !Ref DbSubnetIpBlocks
DbSubnet3:
  Type: AWS::EC2::Subnet
  Properties:
    AvailabilityZone: !Sub
      - ${AWS::Region}${AZ}
      - AZ: !Select
        - 2
        - !Ref VpcAzs
    VpcId: !Ref VPC
    CidrBlock: !Select
      - 2
      - !Ref DbSubnetIpBlocks

```

관련 리소스

또한 CloudFormation은 동적 참조를 사용하여 속성값을 동적으로 지정할 수 있도록 지원합니다. 예를 들어, Systems Manager Parameter Store에 저장되는 보안 문자열을 참조해야 할 수 있습니다. 자세한 내용은 [동적 참조를 사용하여 다른 서비스에 저장된 값 가져오기](#) 섹션을 참조하세요.

Ref 또는 Sub 함수 내에서 가상 파라미터를 사용하여 값을 동적으로 채울 수도 있습니다. 자세한 내용은 [가상 파라미터를 사용하여 AWS 값 가져오기](#) 섹션을 참조하세요.

CloudFormation 템플릿 Outputs 구문

선택 사항인 Outputs 섹션은 스택의 출력 값을 선언합니다. 이러한 출력 값은 다양한 방식으로 사용할 수 있습니다.

- 리소스 관련 중요한 세부 정보 수집 - 출력은 리소스 관련 중요 정보를 편리하게 수집할 수 있는 방법입니다. 예를 들면 스택의 S3 버킷 이름을 출력하면 버킷을 보다 쉽게 찾을 수 있습니다. CloudFormation 콘솔의 출력 탭이나 [describe-stacks](#) CLI 명령을 사용하여 출력 값을 볼 수 있습니다.

- 교차 스택 참조 - 출력 값을 다른 스택으로 가져와서 [교차 스택 참조를 생성](#)할 수 있습니다. 이는 여러 스택에서 리소스 또는 구성을 공유해야 할 때 유용합니다.

⚠ Important

CloudFormation은 Outputs 섹션에 포함된 정보를 삭제하거나 난독화하지 않습니다. 이 섹션을 사용하여 암호나 보안 암호와 같은 민감한 정보를 출력하지 않는 것이 좋습니다.

출력값은 스택 작업이 완료된 후에 사용할 수 있습니다. 스택 상태가 IN_PROGRESS [상태](#) 중 하나일 때는 스택 출력 값을 사용할 수 없습니다. 출력값을 항상 사용할 수 있는 것은 아니므로 서비스 런타임과 스택 출력값 간에 종속성을 설정하지 않는 것이 좋습니다.

구문

Outputs 섹션은 키 이름 Outputs로 이루어집니다. 템플릿 안에 최대 60개 출력을 선언할 수 있습니다.

다음 예제에서는 Outputs 섹션의 구조를 보여줍니다.

JSON

모든 출력 선언을 괄호 안에 포함합니다. 출력이 여러 개인 경우 쉼표로 구분합니다.

```
"Outputs" : {
  "OutputLogicalID" : {
    "Description" : "Information about the value",
    "Value" : "Value to return",
    "Export" : {
      "Name" : "Name of resource to export"
    }
  }
}
```

YAML

```
Outputs:
  OutputLogicalID:
    Description: Information about the value
    Value: Value to return
    Export:
```


Name: *Name of resource to export*

출력 필드

Outputs 섹션은 다음 필드를 포함할 수 있습니다.

논리적 ID(논리명이라고도 함)

현재 출력의 식별자입니다. 논리적 ID는 영숫자(a-z, A-Z, 0-9)여야 하며 템플릿 내에서 고유해야 합니다.

Description (선택 사항)

출력값을 설명하는 String 유형입니다. 설명 선언 값은 0~1023바이트 길이의 리터럴 문자열이어야 합니다. 파라미터나 함수를 사용하여 설명을 지정할 수 없습니다.

Value(필수)

[describe-stacks](#) 명령에 의해 반환되는 속성의 값입니다. 출력의 값에는 리터럴, 파라미터 참조, 가상 파라미터, 매핑 값 또는 내장 함수가 포함될 수 있습니다.

Export (선택 사항)

교차 스택 참조를 위해 내보낼 리소스 출력의 이름입니다.

내장 함수를 사용하여 내보내기의 Name 값을 사용자 지정할 수 있습니다.

자세한 내용은 [배포된 CloudFormation 스택에서 내보낸 출력 가져오기](#) 섹션을 참조하세요.

출력과 조건을 연결하려면 템플릿의 [Conditions](#) 섹션에 조건을 정의합니다.

예제

다음 예제에서는 스택 출력의 작동 방식을 보여줍니다.

주제

- [스택 출력](#)
- [Fn::Sub를 사용하여 내보내기 이름 사용자 지정](#)
- [Fn::Join를 사용하여 내보내기 이름 사용자 지정](#)
- [Fn::Join를 사용하여 구성된 URL 반환](#)

스택 출력

다음 예제에서는 BackupLoadBalancerDNSName 이름의 출력이 BackupLoadBalancer 조건이 true일 때만 논리적 ID가 CreateProdResources인 리소스의 DNS 이름을 반환합니다. 이름이 InstanceID인 출력은 논리적 ID EC2Instance와 함께 EC2 인스턴스의 ID를 반환합니다.

JSON

```
"Outputs" : {
  "BackupLoadBalancerDNSName" : {
    "Description": "The DNSName of the backup load balancer",
    "Value" : { "Fn::GetAtt" : [ "BackupLoadBalancer", "DNSName" ]},
    "Condition" : "CreateProdResources"
  },
  "InstanceID" : {
    "Description": "The Instance ID",
    "Value" : { "Ref" : "EC2Instance" }
  }
}
```

YAML

```
Outputs:
  BackupLoadBalancerDNSName:
    Description: The DNSName of the backup load balancer
    Value: !GetAtt BackupLoadBalancer.DNSName
    Condition: CreateProdResources
  InstanceID:
    Description: The Instance ID
    Value: !Ref EC2Instance
```

Fn::Sub를 사용하여 내보내기 이름 사용자 지정

다음 예제에서는 StackVPC는 VPC의 ID를 반환하고 나서, 스택 이름에 추가된 VPCID 이름을 사용하여 교차 스택 참조 값을 내보냅니다.

JSON

```
"Outputs" : {
  "StackVPC" : {
    "Description" : "The ID of the VPC",
```

```

    "Value" : { "Ref" : "MyVPC" },
    "Export" : {
      "Name" : { "Fn::Sub": "${AWS::StackName}-VPCID" }
    }
  }
}

```

YAML

```

Outputs:
  StackVPC:
    Description: The ID of the VPC
    Value: !Ref MyVPC
    Export:
      Name: !Sub "${AWS::StackName}-VPCID"

```

`Fn::Sub` 함수에 대한 자세한 내용은 [Fn::Sub](#) 단원을 참조하십시오.

Fn::Join를 사용하여 내보내기 이름 사용자 지정

또한 `Fn::Join` 함수를 사용하여 파라미터, 리소스 속성 및 기타 문자열을 기반으로 값을 구성할 수 있습니다.

다음 예제에서는 `Fn::Sub` 함수 대신 `Fn::Join` 함수를 사용하여 내보내기 이름을 사용자 지정합니다. 예제 `Fn::Join` 함수는 콜론을 구분자로 사용하여 스택 이름과 이름 `VPCID`를 연결합니다.

JSON

```

"Outputs" : {
  "StackVPC" : {
    "Description" : "The ID of the VPC",
    "Value" : { "Ref" : "MyVPC" },
    "Export" : {
      "Name" : { "Fn::Join" : [ ":", [ { "Ref" : "AWS::StackName" }, "VPCID" ] ] }
    }
  }
}

```

YAML

```

Outputs:

```

```
StackVPC:
  Description: The ID of the VPC
  Value: !Ref MyVPC
  Export:
    Name: !Join [ ":", [ !Ref "AWS::StackName", VPCID ] ]
```

`Fn::Join` 함수에 대한 자세한 내용은 [Fn::Join](#) 단원을 참조하십시오.

Fn::Join을 사용하여 구성된 URL 반환

WordPress 사이트를 생성하는 템플릿에 대한 이전 예제에서 `InstallURL`은 `Fn::Join` 함수 호출에서 반환된 문자열로, `http://`, 리소스 `ElasticLoadBalancer`의 DNS 이름, `/wp-admin/install.php`를 연결합니다. 출력 값은 다음과 비슷합니다.

```
http://mywptests-elasticl-1gb51l6sl8y5v-206169572.aws-region.elb.amazonaws.com/wp-admin/install.php
```

JSON

```
{
  "Outputs": {
    "InstallURL": {
      "Value": {
        "Fn::Join": [
          "",
          [
            "http://",
            {
              "Fn::GetAtt": [
                "ElasticLoadBalancer",
                "DNSName"
              ]
            },
            "/wp-admin/install.php"
          ]
        ]
      },
      "Description": "Installation URL of the WordPress website"
    }
  }
}
```

YAML

```
Outputs:
  InstallURL:
    Value: !Join
      - ''
      - - 'http://'
        - !GetAtt
          - ElasticLoadBalancer
          - DNSName
        - /wp-admin/install.php
    Description: Installation URL of the WordPress website
```

Fn::Join 함수에 대한 자세한 내용은 [Fn::Join](#) 단원을 참조하십시오.

CloudFormation 템플릿 Mappings 구문

선택 사항인 Mappings 섹션을 사용하면 특정 조건이나 종속성을 기반으로 값을 지정하는 데 사용할 수 있는 키-값 페어를 만들 수 있습니다.

Mappings 섹션의 일반적인 사용 사례 중 하나는 스택이 배포된 AWS 리전을 기반으로 값을 설정하는 것입니다. AWS::Region 가상 파라미터를 사용하여 설정할 수 있습니다. AWS::Region 가상 파라미터는 CloudFormation에서 스택이 생성되는 리전으로 확인되는 값입니다. 가상 파라미터는 스택 생성 시 CloudFormation에서 확인됩니다.

맵에서 값을 검색하려면 템플릿의 Resources 섹션 내에서 Fn::FindInMap 내장 함수를 사용하면 됩니다.

구문

Mappings 섹션에서는 다음 구문을 사용합니다.

JSON

```
"Mappings" : {
  "MappingLogicalName" : {
    "Key1" : {
      "Name" : "Value1"
    },
    "Key2" : {
      "Name" : "Value2"
    },
  },
}
```

```

    "Key3" : {
      "Name" : "Value3"
    }
  }
}

```

YAML

```

Mappings:
  MappingLogicalName:
    Key1:
      Name: Value1
    Key2:
      Name: Value2
    Key3:
      Name: Value3

```

- MappingLogicalName은 매핑의 논리명입니다.
- 매핑 내에서 각 맵은 키이며, 그 뒤에는 다른 매핑이 이어집니다.
- 키는 이름-값 페어의 맵이어야 하며 매핑 내에서 고유해야 합니다.
- 이름-값 페어는 레이블과, 매핑할 값입니다. 값에 이름을 지정하여 두 개 이상의 값 세트를 키로 매핑할 수 있습니다.
- 매핑의 키는 문자열이어야 합니다.
- 값은 String 또는 List 유형이 될 수 있습니다.

Note

Mappings 섹션에는 파라미터, 가상 파라미터 또는 내장 함수를 포함할 수 없습니다. 매핑의 값이 현재 스택에서 사용되지 않는 경우, 매핑만 업데이트할 수는 없습니다. 리소스를 추가, 수정 또는 삭제하는 변경 내용을 포함해야 합니다.

예제

주제

- [기본 매핑](#)
- [여러 값을 사용하여 매핑](#)

- [매핑에서 값 반환](#)
- [입력 파라미터 및 Fn::FindInMap](#)

기본 매핑

다음 예제에서는 Mappings 맵을 사용하는 RegionToInstanceType 매핑을 보여줍니다. 이 매핑에는 단일 문자열 값을 포함하는 이름-값 페어로 매핑된 다섯 개 키가 들어 있습니다. 키는 리전 이름입니다. 각 이름-값 페어는 키로 표현되는 리전에서 사용할 수 있는 T 패밀리의 인스턴스 유형입니다. 이름-값 페어에는 이름(이 예제에서는 InstanceType)과 값이 지정됩니다.

JSON

```
"Mappings" : {
  "RegionToInstanceType" : {
    "us-east-1"      : { "InstanceType" : "t2.micro" },
    "us-west-1"      : { "InstanceType" : "t2.micro" },
    "eu-west-1"      : { "InstanceType" : "t2.micro" },
    "eu-north-1"     : { "InstanceType" : "t3.micro" },
    "me-south-1"     : { "InstanceType" : "t3.micro" }
  }
}
```

YAML

```
Mappings:
  RegionToInstanceType:
    us-east-1:
      InstanceType: t2.micro
    us-west-1:
      InstanceType: t2.micro
    eu-west-1:
      InstanceType: t2.micro
    eu-north-1:
      InstanceType: t3.micro
    me-south-1:
      InstanceType: t3.micro
```

여러 값을 사용하여 매핑

다음 예제에는 MyAMI1 및 MyAMI2, 이렇게 두 가지 값 세트로 매핑되는 리전 키가 있습니다.

Note

이 예제에 표시된 AMI ID는 데모용 자리 표시자입니다. 가능하면 Mappings 섹션의 대안으로 AWS Systems Manager 파라미터에 대한 동적 참조를 사용하는 것이 좋습니다. 사용하려는 AMI가 변경될 때마다 모든 템플릿을 새 ID로 업데이트하지 않으려면 스택이 생성되거나 업데이트될 때 AWS Systems Manager 파라미터를 사용하여 최신 AMI ID를 검색하세요. 일반적으로 사용되는 AMI의 최신 버전도 Systems Manager의 퍼블릭 파라미터로 사용할 수 있습니다. 자세한 내용은 [동적 참조를 사용하여 다른 서비스에 저장된 값 가져오기](#)를 참조하세요.

JSON

```
"Mappings" : {
  "RegionToAMI" : {
    "us-east-1"      : { "MyAMI1" : "ami-12345678901234567", "MyAMI2" :
"ami-23456789012345678" },
    "us-west-1"     : { "MyAMI1" : "ami-34567890123456789", "MyAMI2" :
"ami-45678901234567890" },
    "eu-west-1"     : { "MyAMI1" : "ami-56789012345678901", "MyAMI2" :
"ami-67890123456789012" },
    "ap-southeast-1" : { "MyAMI1" : "ami-78901234567890123", "MyAMI2" :
"ami-89012345678901234" },
    "ap-northeast-1" : { "MyAMI1" : "ami-90123456789012345", "MyAMI2" :
"ami-01234567890123456" }
  }
}
```

YAML

```
Mappings:
  RegionToAMI:
    us-east-1:
      MyAMI1: ami-12345678901234567
      MyAMI2: ami-23456789012345678
    us-west-1:
      MyAMI1: ami-34567890123456789
      MyAMI2: ami-45678901234567890
    eu-west-1:
      MyAMI1: ami-56789012345678901
      MyAMI2: ami-67890123456789012
    ap-southeast-1:
```



```

MyAMI1: ami-78901234567890123
MyAMI2: ami-89012345678901234
ap-northeast-1:
MyAMI1: ami-90123456789012345
MyAMI2: ami-01234567890123456

```

매핑에서 값 반환

`Fn::FindInMap` 함수를 사용하여 지정한 키에 따라 명명된 값을 반환할 수 있습니다. 다음 예제 템플릿에는 `FindInMap` 함수를 통해 `InstanceType` 속성이 할당된 Amazon EC2 리소스가 포함되어 있습니다. `FindInMap` 함수는 키를 스택이 생성되는 AWS 리전(`AWS::Region` 가상 파라미터 사용)으로 지정하고 `InstanceType`을 매핑할 값 이름으로 지정합니다. `ImageId`는 Systems Manager 파라미터를 사용하여 최신 Amazon Linux 2 AMI를 동적으로 검색합니다. 가상 parameters에 대한 자세한 내용은 [가상 파라미터를 사용하여 AWS 값 가져오기](#) 단원을 참조하십시오.

JSON

```

{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Mappings" : {
    "RegionToInstanceType" : {
      "us-east-1"      : { "InstanceType" : "t2.micro" },
      "us-west-1"      : { "InstanceType" : "t2.micro" },
      "eu-west-1"      : { "InstanceType" : "t2.micro" },
      "eu-north-1"     : { "InstanceType" : "t3.micro" },
      "me-south-1"     : { "InstanceType" : "t3.micro" }
    }
  },
  "Resources" : {
    "myEC2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "ImageId" : "{{resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2}}",
        "InstanceType" : { "Fn::FindInMap" : [ "RegionToInstanceType", { "Ref" :
"AWS::Region" }, "InstanceType" ] }
      }
    }
  }
}

```

YAML

```

AWSTemplateFormatVersion: 2010-09-09
Mappings:
  RegionToInstanceType:
    us-east-1:
      InstanceType: t2.micro
    us-west-1:
      InstanceType: t2.micro
    eu-west-1:
      InstanceType: t2.micro
    eu-north-1:
      InstanceType: t3.micro
    me-south-1:
      InstanceType: t3.micro
Resources:
  myEC2Instance:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: '{{resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2}}'
      InstanceType: !FindInMap [RegionToInstanceType, !Ref 'AWS::Region', InstanceType]

```

입력 파라미터 및 **Fn::FindInMap**

다음 예제 템플릿은 여러 매핑을 사용하여 EC2 인스턴스를 생성하는 방법을 보여줍니다. 템플릿은 중첩 매핑을 사용하여 대상 AWS 리전 및 환경 유형(Dev 또는 Prod)에 따라 적절한 인스턴스 유형과 보안 그룹을 자동으로 선택합니다. 또한 Systems Manager 파라미터를 사용하여 최신 Amazon Linux 2 AMI를 동적으로 검색합니다.

JSON

```

{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Parameters" : {
    "EnvironmentType" : {
      "Description" : "The environment type (Dev or Prod)",
      "Type" : "String",
      "Default" : "Dev",
      "AllowedValues" : [ "Dev", "Prod" ]
    }
  },
  "Mappings" : {

```

```

    "RegionAndEnvironmentToInstanceType" : {
      "us-east-1"      : { "Dev" : "t3.micro", "Prod" : "c5.large" },
      "us-west-1"      : { "Dev" : "t2.micro", "Prod" : "m5.large" }
    },
    "RegionAndEnvironmentToSecurityGroup" : {
      "us-east-1"      : { "Dev" : "sg-12345678", "Prod" : "sg-abcdef01" },
      "us-west-1"      : { "Dev" : "sg-ghijkl23", "Prod" : "sg-45678abc" }
    }
  },
  "Resources" : {
    "Ec2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "ImageId" : "{{resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2}}",
        "InstanceType" : { "Fn::FindInMap": [ "RegionAndEnvironmentToInstanceType",
        { "Ref": "AWS::Region" }, { "Ref": "EnvironmentType" } ] },
        "SecurityGroupIds" : [{ "Fn::FindInMap" :
        [ "RegionAndEnvironmentToSecurityGroup", { "Ref" : "AWS::Region" }, { "Ref" :
        "EnvironmentType" } ] } ]
      }
    }
  }
}

```

YAML

```

AWSTemplateFormatVersion: 2010-09-09
Parameters:
  EnvironmentType:
    Description: The environment type (Dev or Prod)
    Type: String
    Default: Dev
    AllowedValues:
      - Dev
      - Prod
Mappings:
  RegionAndEnvironmentToInstanceType:
    us-east-1:
      Dev: t3.micro
      Prod: c5.large
    us-west-1:
      Dev: t2.micro

```

```

    Prod: m5.large
RegionAndEnvironmentToSecurityGroup:
  us-east-1:
    Dev: sg-12345678
    Prod: sg-abcdef01
  us-west-1:
    Dev: sg-ghijkl23
    Prod: sg-45678abc
Resources:
  Ec2Instance:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: '{{resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-
x86_64-gp2}}'
      InstanceType: !FindInMap [RegionAndEnvironmentToInstanceType, !Ref
'AWS::Region', !Ref EnvironmentType]
      SecurityGroupIds:
        - !FindInMap [RegionAndEnvironmentToSecurityGroup, !Ref 'AWS::Region', !Ref
EnvironmentType]

```

관련 리소스

이러한 관련 주제는 `Fn::FindInMap` 함수를 사용하는 템플릿을 개발할 때 도움이 될 수 있습니다.

- [Fn::FindInMap](#)
- [Fn::FindInMap 기능 향상](#)
- [Fn::Sub](#)

CloudFormation 템플릿 Metadata 구문

Metadata는 JSON 또는 YAML 객체를 사용하여 추가 정보를 저장합니다. 템플릿에서 사용할 수 있는 템플릿 수준 메타데이터 유형은 다음과 같습니다.

사용자 지정 메타데이터

사용자 정의 키-값 페어를 저장합니다. 예를 들어 리소스 생성에는 영향을 미치지 않지만 인프라, 팀 또는 배포 세부 사항에 대한 추가 컨텍스트를 제시하는 추가 정보를 제공할 수 있습니다.

AWS::CloudFormation::Interface

CloudFormation 콘솔에 표시되는 입력 파라미터의 그룹화 및 순서를 정의합니다. 기본적으로 CloudFormation 콘솔에는 파라미터가 논리적 ID에 따라 영문자순으로 정렬됩니다.

AWS::CloudFormation::Designer

CloudFormation Designer(Designer)는 2025년 2월 5일 수명이 종료되었습니다.

Important

스택 업데이트 중에는 Metadata 섹션을 자체적으로 업데이트할 수 없습니다. 리소스를 추가, 수정 또는 삭제하는 변경 내용을 포함할 때만 업데이트할 수 있습니다.

CloudFormation은 Metadata 섹션에 포함된 정보를 변환, 수정 또는 삭제하지 않습니다. 이러한 이유 때문에 이 섹션을 사용하여 암호나 보안 암호와 같은 민감한 정보를 저장하지 않는 것이 좋습니다.

구문

CloudFormation 템플릿에서 사용자 지정 메타데이터를 선언하려면 다음 구문을 사용합니다.

JSON

```
"Metadata" : {  
  "Instances" : {"Description" : "Information about the instances"},  
  "Databases" : {"Description" : "Information about the databases"}  
}
```

YAML

```
Metadata:  
  Instances:  
    Description: "Information about the instances"  
  Databases:  
    Description: "Information about the databases"
```

AWS::CloudFormation::Interface에 대한 구문은 [AWS::CloudFormation::Interface 메타데이터를 사용하여 CloudFormation 파라미터 구성](#) 섹션을 참조하세요.

AWS::CloudFormation::Interface 메타데이터를 사용하여 CloudFormation 파라미터 구성

AWS::CloudFormation::Interface는 CloudFormation 콘솔에서 파라미터가 그룹화되고 정렬되는 방식을 정의하는 메타데이터 키입니다. 기본적으로 콘솔에서 스택을 생성하거나 업데이트하면 콘솔에는 입력 파라미터가 논리적 ID를 기준으로 사전순으로 나열됩니다. 이 키를 사용하면 사용자가 파라미터 값을 효율적으로 지정할 수 있도록 고유의 파라미터 그룹화 및 정렬을 정의할 수 있습니다. 예를 들어, 모든 EC2 관련 파라미터를 한 그룹으로 그룹화하고 모든 VPC 관련 파라미터를 다른 한 그룹으로 그룹화할 수 있습니다.

메타데이터 키에서는 생성할 그룹, 각 그룹에 포함할 파라미터, 그리고 콘솔에서 그룹 내 각 파라미터를 표시하는 순서를 지정할 수 있습니다.

파라미터의 레이블을 정의할 수도 있습니다. 레이블은 파라미터의 논리적 ID 대신 콘솔에 표시되는 표시 이름 또는 설명입니다. 레이블은 사용자가 각 파라미터에 지정할 값을 이해하도록 돕는 데 유용합니다. 예를 들어, KeyPair 파라미터에 Select an EC2 key pair라는 레이블을 지정할 수 있습니다.

메타데이터 키에서 참조하는 모든 파라미터는 템플릿의 Parameters 섹션에 선언해야 합니다.

Note

CloudFormation 콘솔만 AWS::CloudFormation::Interface 메타데이터 키를 사용합니다. AWS CLI 및 API 호출은 이 키를 사용하지 않습니다.

구문

CloudFormation 템플릿에서 이 객체를 선언하려면 다음 구문을 사용합니다.

JSON

```
"Metadata" : {
  "AWS::CloudFormation::Interface" : {
    "ParameterGroups": [
      {
        "Label": {
          "default": "Group Label"
        },
        "Parameters": [
          "Parameter1",
```

```

        "Parameter2"
    ]
}
],
"ParameterLabels": {
    "Parameter1": {
        "default": "Friendly Name for Parameter1"
    }
}
}
}

```

YAML

```

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      - Label:
          default: Group Label
        Parameters:
          - Parameter1
          - Parameter2
    ParameterLabels:
      Parameter1:
        default: Friendly Name for Parameter1

```

속성

ParameterGroups

그룹 이름, 각 그룹의 파라미터 및 파라미터가 표시되는 순서를 지정하는 파라미터 그룹 유형의 목록입니다.

필수 항목 여부: 아니요

Label

파라미터 그룹의 이름입니다.

필수 항목 여부: 아니요

default

CloudFormation 콘솔이 파라미터 그룹의 이름을 지정하는 데 사용하는 기본 레이블입니다.

필수 항목 여부: 아니요

유형: 문자열

Parameters

그룹에 포함할 파라미터 논리적 ID(대/소문자 구분)의 목록입니다. 파라미터가 템플릿의 Parameters 섹션에 이미 정의되어 있어야 합니다. 파라미터는 한 파라미터 그룹에만 포함될 수 있습니다.

Other parameters 그룹에서 파라미터 그룹과 연결되지 않은 파라미터가 사전 순으로 콘솔에 나열됩니다.

필수 항목 여부: 아니요

유형: 문자열 값의 목록

ParameterLabels

스택을 생성하거나 업데이트할 때 CloudFormation 콘솔에 표시되는 파라미터 및 해당 표시 이름의 매핑입니다.

필수 항목 여부: 아니요

파라미터 레이블

파라미터에 대한 레이블입니다. 이 레이블은 스택 생성 또는 업데이트 시 CloudFormation 콘솔이 [파라미터 지정(Specify Parameters)] 페이지에 표시하는 기억하기 쉬운 이름 또는 설명을 정의합니다. 파라미터 라벨은 템플릿의 Parameters 섹션에서 선언된 유효한 파라미터의 논리적 ID(대/소문자 구분)여야 합니다.

필수 항목 여부: 아니요

default

CloudFormation 콘솔이 파라미터의 이름을 지정하는 데 사용하는 기본 레이블입니다.

필수 항목 여부: 아니요

유형: 문자열

예제

다음 예제에서는 Network Configuration 및 Amazon EC2 Configuration이라는 두 개의 파라미터 그룹을 정의합니다. Network Configuration 그룹에는 템플릿(표시되지 않음)의 VPCID 섹션

에서 정의되는 SubnetId, SecurityGroupId 및 Parameters 파라미터가 포함됩니다. 이러한 파라미터가 콘솔에 표시되는 순서는 VPCID 파라미터로 시작하여 파라미터가 나열되는 순서에 따라 정의됩니다. 예제에서는 비슷한 방식으로 Amazon EC2 Configuration 파라미터를 그룹화하고 정렬합니다.

예제에서는 VPCID 파라미터의 레이블도 정의합니다. 콘솔에서 파라미터의 논리적 ID(VPCID) 대신 Which VPC should this be deployed to?(어떤 VPC에 배포합니까?)가 표시됩니다.

JSON

```
"Metadata" : {
  "AWS::CloudFormation::Interface" : {
    "ParameterGroups" : [
      {
        "Label" : { "default" : "Network Configuration" },
        "Parameters" : [ "VPCID", "SubnetId", "SecurityGroupId" ]
      },
      {
        "Label" : { "default": "Amazon EC2 Configuration" },
        "Parameters" : [ "InstanceType", "KeyName" ]
      }
    ],
    "ParameterLabels" : {
      "VPCID" : { "default" : "Which VPC should this be deployed to?" }
    }
  }
}
```

YAML

```
Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      - Label:
          default: "Network Configuration"
        Parameters:
          - VPCID
          - SubnetId
          - SecurityGroupId
      - Label:
          default: "Amazon EC2 Configuration"
        Parameters:
```

```

- InstanceType
- KeyName
ParameterLabels:
  VPCID:
    default: "Which VPC should this be deployed to?"

```

콘솔의 파라미터 그룹

다음 그림에서는 이 예제의 메타데이터 키를 사용하여 스택을 생성하거나 업데이트할 때 파라미터 그룹이 콘솔에 표시되는 방식을 보여 줍니다. 콘솔의 파라미터 그룹

Parameters

Network Configuration

Which VPC should this be deployed to? Search by ID, or Name tag value
VpcId of your existing Virtual Private Cloud (VPC)

SubnetId Search by ID, or Name tag value
The list of SubnetIds in your Virtual Private Cloud (VPC)

SecurityGroupId Search by ID, name or Name tag value
VpcId of your existing Virtual Private Cloud (VPC)

Amazon EC2 Configuration

InstanceType m1.small Web

KeyName Search
Name of an existing EC2 KeyPair to enable SSH access to the instance

CloudFormation 템플릿 Rules 구문

Rules 섹션은 사용자 지정 검증 로직을 활성화하는 CloudFormation 템플릿의 선택 사항 부분입니다. 포함된 경우 이 섹션에는 CloudFormation에서 리소스를 생성하거나 업데이트하기 전에 파라미터 값을 검증하는 규칙 함수가 포함되어 있습니다.

규칙은 표준 파라미터 제약 조건이 충분하지 않은 경우에 유용합니다. 예를 들어 SSL이 활성화된 경우 인증서와 도메인 이름을 모두 제공해야 합니다. 규칙을 통해 이러한 종속성이 충족되도록 보장할 수 있습니다.

구문

Rules 섹션에서는 다음 구문을 사용합니다.

JSON

템플릿의 Rules 섹션은 키 이름 Rules와 그 뒤에 이어지는 콜론 하나로 구성됩니다. 모든 규칙 선언을 괄호로 묶어야 합니다. 규칙을 여러 개 선언할 경우 쉼표로 구분됩니다. 각 규칙에 대해 인용 부호 안의 논리적 이름과 그 뒤에 오는 콜론, 그리고 규칙 조건과 어설션을 묶는 중괄호를 선언합니다.

```
{
  "Rules": {
    "LogicalRuleName1": {
      "RuleCondition": {
        "rule-specific intrinsic function": "Value"
      },
      "Assertions": [
        {
          "Assert": {
            "rule-specific intrinsic function": "Value"
          },
          "AssertDescription": "Information about this assert"
        },
        {
          "Assert": {
            "rule-specific intrinsic function": "Value"
          },
          "AssertDescription": "Information about this assert"
        }
      ]
    },
    "LogicalRuleName2": {
      "Assertions": [
        {
          "Assert": {
            "rule-specific intrinsic function": "Value"
          },
          "AssertDescription": "Information about this assert"
        }
      ]
    }
  }
}
```

YAML

```
Rules:
```

```

LogicalRuleName1:
  RuleCondition:
    rule-specific intrinsic function: Value
  Assertions:
    - Assert:
        rule-specific intrinsic function: Value
      AssertDescription: Information about this assert
    - Assert:
        rule-specific intrinsic function: Value
      AssertDescription: Information about this assert
LogicalRuleName2:
  Assertions:
    - Assert:
        rule-specific intrinsic function: Value
      AssertDescription: Information about this assert

```

규칙 필드

Rules 섹션은 다음 필드를 포함할 수 있습니다.

논리적 ID(논리명이라고도 함)

각 규칙의 고유 식별자입니다.

RuleCondition (선택 사항)

규칙이 언제 적용될지 결정하는 속성입니다. 규칙 조건을 정의하지 않은 경우, 규칙의 어설션이 항상 적용됩니다. 각 규칙은 하나의 규칙 조건만 정의할 수 있습니다.

Assertions(필수)

특정 파라미터에 대해 허용되는 값을 지정하는 하나 이상의 문입니다.

Assert

반드시 true로 평가되는 조건입니다.

AssertDescription

어설션 실패 시 표시되는 메시지입니다.

규칙별 내장 함수

규칙을 정의하려면 템플릿의 Rules 섹션에서만 사용할 수 있는 함수인 규칙 관련 함수를 사용해야 합니다. 이러한 함수는 중첩될 수 있지만, 규칙 조건 또는 어설션의 최종 결과가 true이거나 false이어야 합니다.

다음과 같은 규칙 함수를 사용할 수 있습니다.

- [Fn::And](#)
- [Fn::Contains](#)
- [Fn::EachMemberEquals](#)
- [Fn::EachMemberIn](#)
- [Fn::Equals](#)
- [Fn::If](#)
- [Fn::Not](#)
- [Fn::Or](#)
- [Fn::RefAll](#)
- [Fn::ValueOf](#)
- [Fn::ValueOfAll](#)

이러한 함수는 규칙의 조건 또는 어설션에서 사용합니다. 이 조건 속성은 CloudFormation이 어설션을 적용할지 여부를 결정합니다. 조건이 true로 평가되면 CloudFormation은 프로비저닝된 제품이 생성되거나 업데이트될 때 파라미터 값이 유효한지 여부를 확인하기 위해 어설션을 평가합니다. 파라미터 값이 유효하지 않으면 CloudFormation은 스택을 생성하거나 업데이트하지 않습니다. 조건이 false로 평가되면 CloudFormation은 파라미터 값을 확인하지 않고 스택 작업을 진행합니다.

예제

주제

- [조건부로 파라미터 값 확인](#)
- [파라미터 간 검증](#)

조건부로 파라미터 값 확인

다음 예제에서는 규칙 두 개가 InstanceType 파라미터의 값을 검사합니다. 환경 파라미터(test 또는 prod)의 값에 따라 사용자는 InstanceType 파라미터에 대해 t3.medium 또는 t3.large를 지

정해야 합니다. InstanceType 및 Environment 파라미터는 동일한 템플릿의 Parameters 섹션에서 선언해야 합니다.

JSON

```
{
  "Rules": {
    "testInstanceType": {
      "RuleCondition": {
        "Fn::Equals": [
          {"Ref": "Environment"},
          "test"
        ]
      },
      "Assertions": [
        {
          "Assert": {
            "Fn::Contains": [
              ["t3.medium"],
              {"Ref": "InstanceType"}
            ]
          },
          "AssertDescription": "For a test environment, the instance type must be t3.medium"
        }
      ]
    },
    "prodInstanceType": {
      "RuleCondition": {
        "Fn::Equals": [
          {"Ref": "Environment"},
          "prod"
        ]
      },
      "Assertions": [
        {
          "Assert": {
            "Fn::Contains": [
              ["t3.large"],
              {"Ref": "InstanceType"}
            ]
          },
          "AssertDescription": "For a production environment, the instance type must be t3.large"
        }
      ]
    }
  }
}
```

```

    }
  ]
}
}
}

```

YAML

```

Rules:
  testInstanceType:
    RuleCondition: !Equals
      - !Ref Environment
      - test
    Assertions:
      - Assert:
          'Fn::Contains':
            - - t3.medium
              - !Ref InstanceType
          AssertDescription: 'For a test environment, the instance type must be
t3.medium'
      prodInstanceType:
        RuleCondition: !Equals
          - !Ref Environment
          - prod
        Assertions:
          - Assert:
              'Fn::Contains':
                - - t3.large
                  - !Ref InstanceType
              AssertDescription: 'For a production environment, the instance type must be
t3.large'

```

파라미터 간 검증

다음 샘플 템플릿은 파라미터 간 검증을 위한 규칙의 사용을 보여줍니다. 이는 로드 밸런서 뒤의 Auto Scaling 그룹에서 실행되는 샘플 웹 사이트를 생성합니다. 웹 사이트는 입력 파라미터에 따라 포트 80 또는 443에서 사용할 수 있습니다. Auto Scaling 그룹의 인스턴스가 모든 포트(기본값 8888)에서 수신 되도록 구성할 수 있습니다.

이 템플릿의 규칙은 스택 생성 전에 입력 파라미터를 검증합니다. 모든 서브넷이 지정된 VPC에 속하는 지 검증하고 UseSSL 파라미터가 Yes로 설정된 경우, SSL 인증서 ARN과 호스팅 영역 이름이 모두 제공되는지 확인합니다.

 Note

이 템플릿에서 스택을 생성할 경우 사용한 AWS 리소스에 대한 요금이 청구됩니다.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "VpcId": {
      "Type": "AWS::EC2::VPC::Id",
      "Description": "VpcId of your existing Virtual Private Cloud (VPC)",
      "ConstraintDescription": "must be the VPC Id of an existing Virtual Private Cloud."
    },
    "Subnets": {
      "Type": "List<AWS::EC2::Subnet::Id>",
      "Description": "The list of SubnetIds in your Virtual Private Cloud (VPC)",
      "ConstraintDescription": "must be a list of at least two existing subnets associated with at least two different availability zones."
    },
    "InstanceType": {
      "Description": "WebServer EC2 instance type",
      "Type": "String",
      "Default": "t2.micro",
      "AllowedValues": ["t2.micro", "t3.micro"],
      "ConstraintDescription": "must be a valid EC2 instance type."
    },
    "KeyName": {
      "Description": "Name of an existing EC2 KeyPair to enable SSH access to the instances",
      "Type": "AWS::EC2::KeyPair::KeyName",
      "ConstraintDescription": "must be the name of an existing EC2 KeyPair."
    },
    "SSHLocation": {
      "Description": "The IP address range that can be used to SSH to the EC2 instances",
      "Type": "String",
      "MinLength": "9",
      "MaxLength": "18",
      "Default": "0.0.0.0/0",
      "AllowedPattern": "(\\d{1,3})\\. (\\d{1,3})\\. (\\d{1,3})\\. (\\d{1,3})/(\\d{1,2})",

```



```

    "ConstraintDescription": "must be a valid IP CIDR range of the form x.x.x.x/x."
  },
  "UseSSL": {
    "AllowedValues": ["Yes", "No"],
    "Default": "No",
    "Description": "Select \"Yes\" to implement SSL, \"No\" to skip (default).",
    "Type": "String"
  },
  "ALBSSLCertificateARN": {
    "Default": "",
    "Description": "[Optional] The ARN of the SSL certificate to be used for the Application Load Balancer",
    "Type": "String"
  },
  "HostedZoneName": {
    "AllowedPattern": "^$|(([a-zA-Z0-9]|[a-zA-Z0-9][a-zA-Z0-9\\-\\-]*[a-zA-Z0-9])\\
\\.)*([A-Za-z0-9]|[A-Za-z0-9][A-Za-z0-9\\-\\-]*[A-Za-z0-9])$",
    "Default": "",
    "Description": "[Optional] The domain name of a valid Hosted Zone on AWS.",
    "Type": "String"
  }
},
"Conditions": {
  "UseALBSSL": {"Fn::Equals": [{"Ref": "UseSSL"}, "Yes"]}
},
"Rules": {
  "SubnetsInVPC": {
    "Assertions": [
      {
        "Assert": {"Fn::EachMemberEquals": [{"Fn::ValueOf": ["Subnets", "VpcId"]}, {"Ref": "VpcId"}]},
        "AssertDescription": "All subnets must be in the VPC"
      }
    ]
  },
  "ValidateHostedZone": {
    "RuleCondition": {"Fn::Equals": [{"Ref": "UseSSL"}, "Yes"]},
    "Assertions": [
      {
        "Assert": {"Fn::Not": [{"Fn::Equals": [{"Ref": "ALBSSLCertificateARN"}, ""]}]},
        "AssertDescription": "ACM Certificate value cannot be empty if SSL is required"
      }
    ]
  }
}

```

```

    {
      "Assert": {"Fn::Not": [{"Fn::Equals": [{"Ref": "HostedZoneName"}, ""]}]},
      "AssertDescription": "Route53 Hosted Zone Name is mandatory when SSL is
required"
    }
  ]
},
"Resources": {
  "WebServerGroup": {
    "Type": "AWS::AutoScaling::AutoScalingGroup",
    "Properties": {
      "VPCZoneIdentifier": {"Ref": "Subnets"},
      "LaunchTemplate": {
        "LaunchTemplateId": {"Ref": "LaunchTemplate"},
        "Version": {"Fn::GetAtt": ["LaunchTemplate", "LatestVersionNumber"]}
      },
      "MinSize": "2",
      "MaxSize": "2",
      "TargetGroupARNs": [{"Ref": "ALBTargetGroup"}]
    },
    "CreationPolicy": {
      "ResourceSignal": {"Timeout": "PT15M"}
    },
    "UpdatePolicy": {
      "AutoScalingRollingUpdate": {
        "MinInstancesInService": "1",
        "MaxBatchSize": "1",
        "PauseTime": "PT15M",
        "WaitOnResourceSignals": true
      }
    }
  },
  "LaunchTemplate": {
    "Type": "AWS::EC2::LaunchTemplate",
    "Metadata": {
      "Comment": "Install a simple application",
      "AWS::CloudFormation::Init": {
        "config": {
          "packages": {"yum": {"httpd": []}},
          "files": {
            "/var/www/html/index.html": {
              "content": {"Fn::Join": ["\n", ["<h1>Congratulations, you have
successfully launched the AWS CloudFormation sample.</h1>"]]}},

```

```

        "mode": "000644",
        "owner": "root",
        "group": "root"
    },
    "/etc/cfn/cfn-hup.conf": {
        "content": {"Fn::Join": ["", [
            "[main]\n",
            "stack=", {"Ref": "AWS::StackId"}, "\n",
            "region=", {"Ref": "AWS::Region"}, "\n"
        ]]},
        "mode": "000400",
        "owner": "root",
        "group": "root"
    },
    "/etc/cfn/hooks.d/cfn-auto-reloader.conf": {
        "content": {"Fn::Join": ["", [
            "[cfn-auto-reloader-hook]\n",
            "triggers=post.update\n",
            "path=Resources.LaunchTemplate.Metadata.AWS::CloudFormation::Init\n",
            "action=/opt/aws/bin/cfn-init -v ",
            "    --stack ", {"Ref": "AWS::StackName"},
            "    --resource LaunchTemplate ",
            "    --region ", {"Ref": "AWS::Region"}, "\n",
            "runas=root\n"
        ]]},
        "mode": "000400",
        "owner": "root",
        "group": "root"
    }
},
"services": {
    "sysvinit": {
        "httpd": {
            "enabled": "true",
            "ensureRunning": "true"
        },
        "cfn-hup": {
            "enabled": "true",
            "ensureRunning": "true",
            "files": [
                "/etc/cfn/cfn-hup.conf",
                "/etc/cfn/hooks.d/cfn-auto-reloader.conf"
            ]
        }
    }
}

```

```

        }
    }
}
},
"Properties": {
    "LaunchTemplateName": {"Fn::Sub": "${AWS::StackName}-launch-template"},
    "LaunchTemplateData": {
        "ImageId": "{{resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2}}",
        "SecurityGroupIds": [{"Ref": "InstanceSecurityGroup"}],
        "InstanceType": {"Ref": "InstanceType"},
        "KeyName": {"Ref": "KeyName"},
        "UserData": {
            "Fn::Base64": {"Fn::Join": ["", [
                "#!/bin/bash\n",
                "yum install -y aws-cfn-bootstrap\n",
                "/opt/aws/bin/cfn-init -v ",
                "    --stack ", {"Ref": "AWS::StackName"},
                "    --resource LaunchTemplate ",
                "    --region ", {"Ref": "AWS::Region"}, "\n",
                "/opt/aws/bin/cfn-signal -e $? ",
                "    --stack ", {"Ref": "AWS::StackName"},
                "    --resource WebServerGroup ",
                "    --region ", {"Ref": "AWS::Region"}, "\n"
            ]]}
        }
    }
},
"ELBSecurityGroup": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
        "GroupDescription": "Allow access to the ELB",
        "VpcId": {"Ref": "VpcId"},
        "SecurityGroupIngress": [{
            "Fn::If": [
                "UseALBSSL",
                {
                    "IpProtocol": "tcp",
                    "FromPort": 443,
                    "ToPort": 443,
                    "CidrIp": "0.0.0.0/0"
                }
            ]
        }],
    }
},

```

```

        {
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIp": "0.0.0.0/0"
        }
    ]
}
}],
"ApplicationLoadBalancer": {
    "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
    "Properties": {
        "Subnets": {"Ref": "Subnets"},
        "SecurityGroups": [{"Ref": "ELBSecurityGroup"}]
    }
},
"ALBListener": {
    "Type": "AWS::ElasticLoadBalancingV2::Listener",
    "Properties": {
        "DefaultActions": [{
            "Type": "forward",
            "TargetGroupArn": {"Ref": "ALBTargetGroup"}
        }],
        "LoadBalancerArn": {"Ref": "ApplicationLoadBalancer"},
        "Port": {"Fn::If": ["UseALBSSL", 443, 80]},
        "Protocol": {"Fn::If": ["UseALBSSL", "HTTPS", "HTTP"]},
        "Certificates": [{
            "Fn::If": [
                "UseALBSSL",
                {"CertificateArn": {"Ref": "ALBSSLCertificateARN"}},
                {"Ref": "AWS::NoValue"}
            ]
        }
    ]
}],
}
},
"ALBTargetGroup": {
    "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
    "Properties": {
        "HealthCheckIntervalSeconds": 30,
        "HealthCheckTimeoutSeconds": 5,
        "HealthyThresholdCount": 3,
        "Port": 80,
        "Protocol": "HTTP",

```

```

        "UnhealthyThresholdCount": 5,
        "VpcId": {"Ref": "VpcId"}
    },
    "InstanceSecurityGroup": {
        "Type": "AWS::EC2::SecurityGroup",
        "Properties": {
            "GroupDescription": "Enable SSH access and HTTP access on the inbound port",
            "SecurityGroupIngress": [
                {
                    "IpProtocol": "tcp",
                    "FromPort": 80,
                    "ToPort": 80,
                    "SourceSecurityGroupId": {"Fn::Select": [0, {"Fn::GetAtt":
["ApplicationLoadBalancer", "SecurityGroups"]}]}
                },
                {
                    "IpProtocol": "tcp",
                    "FromPort": 22,
                    "ToPort": 22,
                    "CidrIp": {"Ref": "SSHLocation"}
                }
            ],
            "VpcId": {"Ref": "VpcId"}
        }
    },
    "RecordSet": {
        "Type": "AWS::Route53::RecordSetGroup",
        "Condition": "UseALBSSL",
        "Properties": {
            "HostedZoneName": {"Fn::Join": [ "", [{"Ref": "HostedZoneName"}, "." ] ] },
            "RecordSets": [ {
                "Name": {"Fn::Join": [ "", [
                    {"Fn::Select": [ "0", {"Fn::Split": [ ".", {"Fn::GetAtt":
["ApplicationLoadBalancer", "DNSName"]} ] ] } ],
                    ".",
                    {"Ref": "HostedZoneName"},
                    "."
                ] ] },
                "Type": "A",
                "AliasTarget": {
                    "DNSName": {"Fn::GetAtt": ["ApplicationLoadBalancer", "DNSName"]},
                    "EvaluateTargetHealth": true,

```

```

        "HostedZoneId": {"Fn::GetAtt": ["ApplicationLoadBalancer",
"CanonicalHostedZoneID"]}
    }
  ]]
}
},
"Outputs": {
  "URL": {
    "Description": "URL of the website",
    "Value": {"Fn::Join": ["", [
      {"Fn::If": [
        "UseALBSSL",
        {"Fn::Join": ["", [
          "https://",
          {"Fn::Join": ["", [
            {"Fn::Select": ["0", {"Fn::Split": [".", {"Fn::GetAtt":
["ApplicationLoadBalancer", "DNSName"]}]}}],
            ".",
            {"Ref": "HostedZoneName"},
            "."
          ]}]
        ]}],
        {"Fn::Join": ["", [
          "http://",
          {"Fn::GetAtt": ["ApplicationLoadBalancer", "DNSName"]}
        ]}]
      ]}]
    ]}]
  }
}
}
}

```

YAML

```

AWSTemplateFormatVersion: 2010-09-09
Parameters:
  VpcId:
    Type: AWS::EC2::VPC::Id
    Description: VpcId of your existing Virtual Private Cloud (VPC)
    ConstraintDescription: must be the VPC Id of an existing Virtual Private Cloud.
  Subnets:
    Type: List<AWS::EC2::Subnet::Id>

```

Description: The list of SubnetIds in your Virtual Private Cloud (VPC)

ConstraintDescription: >-

must be a list of at least two existing subnets associated with at least two different availability zones. They should be residing in the selected Virtual Private Cloud.

InstanceType:

Description: WebServer EC2 instance type

Type: String

Default: t2.micro

AllowedValues:

- t2.micro
- t3.micro

ConstraintDescription: must be a valid EC2 instance type.

KeyName:

Description: Name of an existing EC2 KeyPair to enable SSH access to the instances

Type: AWS::EC2::KeyPair::KeyName

ConstraintDescription: must be the name of an existing EC2 KeyPair.

SSHLocation:

Description: The IP address range that can be used to SSH to the EC2 instances

Type: String

MinLength: '9'

MaxLength: '18'

Default: 0.0.0.0/0

AllowedPattern: '(\d{1,3})\.\(\d{1,3})\.\(\d{1,3})\.\(\d{1,3})/(\d{1,2})'

ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.

UseSSL:

AllowedValues:

- 'Yes'
- 'No'

ConstraintDescription: Select Yes to create a HTTPS Listener

Default: 'No'

Description: 'Select "Yes" to implement SSL, "No" to skip (default).'

Type: String

ALBSSLCertificateARN:

Default: ''

Description: >-

[Optional] The ARN of the SSL certificate to be used for the Application Load Balancer

Type: String

HostedZoneName:

AllowedPattern: >-

^\$|((([a-zA-Z0-9]|[a-zA-Z0-9][a-zA-Z0-9\-\-]*[a-zA-Z0-9])\.)*([A-Za-z0-9]|[A-Za-z0-9][A-Za-z0-9\-\-]*[A-Za-z0-9]))\$

Default: ''


```

    Description: '[Optional] The domain name of a valid Hosted Zone on AWS.'
    Type: String
Conditions:
  UseALBSSL: !Equals
    - !Ref UseSSL
    - 'Yes'
Rules:
  SubnetsInVPC:
    Assertions:
      - Assert:
          'Fn::EachMemberEquals':
            - 'Fn::ValueOf':
                - Subnets
                - VpcId
            - Ref: VpcId
          AssertDescription: All subnets must be in the VPC
  ValidateHostedZone:
    RuleCondition: !Equals
      - !Ref UseSSL
      - 'Yes'
    Assertions:
      - Assert: !Not
          - !Equals
            - !Ref ALBSSLCertificateARN
            - ''
          AssertDescription: ACM Certificate value cannot be empty if SSL is required
      - Assert: !Not
          - !Equals
            - !Ref HostedZoneName
            - ''
          AssertDescription: Route53 Hosted Zone Name is mandatory when SSL is required
Resources:
  WebServerGroup:
    Type: AWS::AutoScaling::AutoScalingGroup
    Properties:
      VPCZoneIdentifier: !Ref Subnets
      LaunchTemplate:
        LaunchTemplateId: !Ref LaunchTemplate
        Version: !GetAtt LaunchTemplate.LatestVersionNumber
      MinSize: '2'
      MaxSize: '2'
      TargetGroupARNs:
        - !Ref ALBTargetGroup
    CreationPolicy:

```

```

    ResourceSignal:
      Timeout: PT15M
  UpdatePolicy:
    AutoScalingRollingUpdate:
      MinInstancesInService: '1'
      MaxBatchSize: '1'
      PauseTime: PT15M
      WaitOnResourceSignals: 'true'
  LaunchTemplate:
    Type: AWS::EC2::LaunchTemplate
    Metadata:
      Comment: Install a simple application
      AWS::CloudFormation::Init:
        config:
          packages:
            yum:
              httpd: []
          files:
            /var/www/html/index.html:
              content: !Join
                - |+
                - - >-
                  <h1>Congratulations, you have successfully launched the AWS
                    CloudFormation sample.</h1>
              mode: '000644'
              owner: root
              group: root
            /etc/cfn/cfn-hup.conf:
              content: !Sub |
                [main]
                stack=${AWS::StackId}
                region=${AWS::Region}
              mode: '000400'
              owner: root
              group: root
            /etc/cfn/hooks.d/cfn-auto-reloader.conf:
              content: !Sub |-
                [cfn-auto-reloader-hook]
                triggers=post.update
                path=Resources.LaunchTemplate.Metadata.AWS::CloudFormation::Init
                action=/opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource
LaunchTemplate --region ${AWS::Region}
                runas=root
              mode: '000400'

```

```

        owner: root
        group: root
    services:
        sysvinit:
            httpd:
                enabled: 'true'
                ensureRunning: 'true'
            cfn-hup:
                enabled: 'true'
                ensureRunning: 'true'
            files:
                - /etc/cfn/cfn-hup.conf
                - /etc/cfn/hooks.d/cfn-auto-reloader.conf
    Properties:
        LaunchTemplateName: !Sub ${AWS::StackName}-launch-template
        LaunchTemplateData:
            ImageId: '{{resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-
x86_64-gp2}}'
            SecurityGroupIds:
                - !Ref InstanceSecurityGroup
            InstanceType: !Ref InstanceType
            KeyName: !Ref KeyName
            UserData: !Base64
                Fn::Sub: |
                    #!/bin/bash
                    yum install -y aws-cfn-bootstrap
                    /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource
LaunchTemplate --region ${AWS::Region}
                    /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource
WebServerGroup --region ${AWS::Region}
            ELBSecurityGroup:
                Type: AWS::EC2::SecurityGroup
            Properties:
                GroupDescription: Allow access to the ELB
                VpcId: !Ref VpcId
                SecurityGroupIngress:
                    - !If
                        - UseALBSSL
                        - IpProtocol: tcp
                          FromPort: 443
                          ToPort: 443
                          CidrIp: 0.0.0.0/0
                    - IpProtocol: tcp
                      FromPort: 80

```

```
        ToPort: 80
        CidrIp: 0.0.0.0/0
ApplicationLoadBalancer:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
    Subnets: !Ref Subnets
    SecurityGroups:
      - !Ref ELBSecurityGroup
ALBListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
      - Type: forward
        TargetGroupArn: !Ref ALBTargetGroup
    LoadBalancerArn: !Ref ApplicationLoadBalancer
    Port: !If
      - UseALBSSL
      - 443
      - 80
    Protocol: !If
      - UseALBSSL
      - HTTPS
      - HTTP
    Certificates:
      - !If
        - UseALBSSL
        - CertificateArn: !Ref ALBSSLCertificateARN
        - !Ref 'AWS::NoValue'
ALBTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    HealthCheckIntervalSeconds: 30
    HealthCheckTimeoutSeconds: 5
    HealthyThresholdCount: 3
    Port: 80
    Protocol: HTTP
    UnhealthyThresholdCount: 5
    VpcId: !Ref VpcId
InstanceSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Enable SSH access and HTTP access on the inbound port
    SecurityGroupIngress:
      - IpProtocol: tcp
```

```

    FromPort: 80
    ToPort: 80
    SourceSecurityGroupId: !Select
      - 0
      - !GetAtt
        - ApplicationLoadBalancer
        - SecurityGroups
    - IpProtocol: tcp
      FromPort: 22
      ToPort: 22
      CidrIp: !Ref SSHLocation
    VpcId: !Ref VpcId
RecordSet:
  Type: AWS::Route53::RecordSetGroup
  Condition: UseALBSSL
  Properties:
    HostedZoneName: !Join
      - ''
      - - !Ref HostedZoneName
        - .
    RecordSets:
      - Name: !Join
        - ''
        - - !Select
          - '0'
          - !Split
            - .
            - !GetAtt
              - ApplicationLoadBalancer
              - DNSName
        - .
        - !Ref HostedZoneName
        - .
      Type: A
    AliasTarget:
      DNSName: !GetAtt
        - ApplicationLoadBalancer
        - DNSName
      EvaluateTargetHealth: true
      HostedZoneId: !GetAtt
        - ApplicationLoadBalancer
        - CanonicalHostedZoneID
Outputs:
  URL:

```

```

Description: URL of the website
Value: !Join
- ''
- - !If
  - UseALBSSL
  - !Join
    - ''
    - - 'https://'
    - !Join
      - ''
      - - !Select
        - '0'
        - !Split
          - .
          - !GetAtt
            - ApplicationLoadBalancer
            - DNSName
        - .
      - !Ref HostedZoneName
    - .
- !Join
  - ''
  - - 'http://'
  - !GetAtt
    - ApplicationLoadBalancer
    - DNSName

```

CloudFormation 템플릿 Conditions 구문

Conditions 섹션(선택 사항)에는 엔터티가 생성 또는 구성된 환경을 정의하는 문이 포함되어 있습니다. 예를 들어 조건을 생성하고 이를 리소스 또는 출력에 연결하여 조건이 true인 경우에만 CloudFormation에서 리소스 또는 출력을 생성하도록 할 수 있습니다. 이와 유사하게 조건을 속성에 연결하여 조건이 true인 경우에만 CloudFormation에서 속성을 특정 값으로 설정하도록 할 수 있습니다. 조건이 false인 경우 CloudFormation에서 사용자가 지정한 대체 값으로 속성을 설정합니다.

다양한 컨텍스트(예: 테스트 환경과 프로덕션 환경)에서 리소스를 생성하는 데 템플릿을 재사용하려는 경우에 조건을 사용할 수 있습니다. 예를 들어 템플릿에서 prod 또는 test를 입력으로 수락하는 EnvironmentType 입력 파라미터를 추가할 수 있습니다. prod 환경에서는 특정 기능을 가진 EC2 인스턴스를 포함할 수 있지만, test 환경에서는 비용 절감을 위해 더 적은 기능을 사용하려고 합니다. 이 조건 정의를 통해, 생성되는 리소스와 각 환경 유형에 대한 리소스 구성 방법을 정의할 수 있습니다.

구문

Conditions 섹션은 키 이름 Conditions로 이루어집니다. 각 조건 선언에는 논리적 ID와 하나 이상의 내장 함수가 포함됩니다.

JSON

```
"Conditions": {
  "LogicalConditionName1": {
    "Intrinsic function": ...[
  },

  "LogicalConditionName2": {
    "Intrinsic function": ...
  }
}
```

YAML

```
Conditions:
  LogicalConditionName1:
    Intrinsic function:
      ...

  LogicalConditionName2:
    Intrinsic function:
      ...
```

조건 작동 방식

조건을 사용하려면 다음 단계를 따릅니다.

1. 파라미터 정의 추가 - 템플릿의 Parameters 섹션에서 조건을 사용하여 평가할 입력을 정의합니다. 조건은 이러한 입력 파라미터의 값에 따라 true 또는 false로 평가합니다. 가상 파라미터는 자동으로 제공되며 Parameters 섹션에 명시적으로 정의할 필요가 없습니다. 가상 parameters에 대한 자세한 내용은 [가상 파라미터를 사용하여 AWS 값 가져오기](#) 단원을 참조하십시오.
2. 조건 정의 추가 - Fn::If, Fn::Equals와 같은 내장 함수를 사용하여 Conditions 섹션에 조건을 정의합니다. 이러한 조건은 CloudFormation에서 연결된 리소스가 생성되는 경우를 결정합니다. 조건은 다음을 기준으로 할 수 있습니다.
 - 입력 또는 가상 파라미터 값

- 기타 조건
- 매핑 값

그러나 조건에서 리소스 논리적 ID 또는 해당 속성을 참조할 수 없습니다.

3. 조건을 리소스 또는 출력과 연결 - Condition 키와 조건의 논리적 ID를 사용하여 리소스 또는 출력에서 조건을 참조합니다. 필요한 경우, 템플릿의 다른 부분(예: 속성 값)에 Fn::If을 사용하여 조건을 기준으로 값을 설정할 수 있습니다. 자세한 내용은 [Condition 키 사용](#) 섹션을 참조하세요.

CloudFormation은 스택을 생성하거나 업데이트할 때 조건을 평가합니다. CloudFormation에서는 true 조건과 연결된 엔터티를 생성하고 false 조건과 연결된 엔터티를 무시합니다. 또한 CloudFormation은 리소스를 수정하기 전에 각 스택 업데이트 시에 이러한 조건을 다시 평가합니다. true 조건과 연결된 상태로 남아 있는 엔터티는 업데이트되고, false 조건과 연결되는 엔터티는 삭제됩니다.

Important

스택 업데이트 중에는 조건을 자체적으로 업데이트할 수 없습니다. 조건은 리소스를 추가, 수정 또는 삭제하는 변경 내용을 포함할 때만 업데이트할 수 있습니다.

조건 내장 함수

다음 내장 함수를 사용하여 조건을 정의할 수 있습니다.

- [Fn::And](#)
- [Fn::Equals](#)
- [Fn::ForEach](#)
- [Fn::If](#)
- [Fn::Not](#)
- [Fn::Or](#)

Note

템플릿의 Resources 섹션과 Outputs 섹션에 있는 메타데이터 속성, 업데이트 정책 속성 및 속성값에서만 Fn::If가 지원됩니다.

Condition 키 사용

조건이 정의되면 Condition 키를 사용하여 Resources, Outputs 등 템플릿의 여러 위치에 조건을 적용할 수 있습니다. Condition 키는 조건의 논리명을 참조하고 지정된 조건의 평가된 결과를 반환합니다.

주제

- [조건을 리소스와 연결](#)
- [조건을 출력과 연결](#)
- [다른 조건의 조건 참조](#)
- [Fn::If를 사용하여 조건부로 속성 값 반환](#)

조건을 리소스와 연결

리소스를 조건부로 생성하려면 조건의 Condition 키와 논리적 ID를 리소스에 속성으로 추가합니다. CloudFormation은 조건이 true로 평가될 때만 리소스를 생성합니다.

JSON

```
"NewVolume" : {
  "Type" : "AWS::EC2::Volume",
  "Condition" : "IsProduction",
  "Properties" : {
    "Size" : "100",
    "AvailabilityZone" : { "Fn::GetAtt" : [ "EC2Instance", "AvailabilityZone" ] }
  }
}
```

YAML

```
NewVolume:
  Type: AWS::EC2::Volume
  Condition: IsProduction
  Properties:
    Size: 100
    AvailabilityZone: !GetAtt EC2Instance.AvailabilityZone
```

조건을 출력과 연결

조건을 출력과 연결할 수도 있습니다. CloudFormation은 연결된 조건이 true로 평가될 때만 출력을 생성합니다.

JSON

```
"Outputs" : {
  "VolumeId" : {
    "Condition" : "IsProduction",
    "Value" : { "Ref" : "NewVolume" }
  }
}
```

YAML

```
Outputs:
  VolumeId:
    Condition: IsProduction
    Value: !Ref NewVolume
```

다른 조건의 조건 참조

Conditions 섹션에서 조건을 정의할 때 Condition 키를 사용하여 다른 조건을 참조할 수 있습니다. 이를 통해 여러 조건을 결합하여 더 복잡한 조건부 로직을 생성할 수 있습니다.

다음 예제에서 IsProdAndFeatureEnabled 조건은 IsProduction 및 IsFeatureEnabled 조건이 true로 평가되는 경우에만 true로 평가됩니다.

JSON

```
"Conditions": {
  "IsProduction" : {"Fn::Equals" : [{"Ref" : "Environment"}, "prod"]},
  "IsFeatureEnabled" : { "Fn::Equals" : [{"Ref" : "FeatureFlag"}, "enabled"]},
  "IsProdAndFeatureEnabled" : {
    "Fn::And" : [
      {"Condition" : "IsProduction"},
      {"Condition" : "IsFeatureEnabled"}
    ]
  }
}
```

YAML

```
Conditions:
  IsProduction:
    !Equals [!Ref Environment, "prod"]
  IsFeatureEnabled:
    !Equals [!Ref FeatureFlag, "enabled"]
  IsProdAndFeatureEnabled: !And
    - !Condition IsProduction
    - !Condition IsFeatureEnabled
```

Fn::If를 사용하여 조건부로 속성 값 반환

더 세분화된 제어를 위해 **Fn::If** 내장 함수를 사용하여 리소스 또는 출력 내에서 두 속성 값 중 하나를 조건부로 반환할 수 있습니다. 이 함수는 조건을 평가하고 조건이 true인 경우 하나의 값을 반환하고, 조건이 false인 경우 다른 값을 반환합니다.

조건부 속성 값

다음 예제에서는 환경 조건을 기반으로 EC2 인스턴스 유형을 설정하는 방법을 보여줍니다.

IsProduction 조건이 true로 평가되면 인스턴스 유형이 **c5.xlarge**로 설정됩니다. 그렇지 않으면 **t3.small**로 설정됩니다.

JSON

```
"Properties" : {
  "InstanceType" : {
    "Fn::If" : [
      "IsProduction",
      "c5.xlarge",
      "t3.small"
    ]
  }
}
```

YAML

```
Properties:
  InstanceType: !If
    - IsProduction
    - c5.xlarge
    - t3.small
```

조건부 속성 제거

또한 `AWS::NoValue` 가상 파라미터를 반환 값으로 사용하여 조건이 `false`인 경우 해당 속성을 제거할 수 있습니다.

JSON

```
"DBSnapshotIdentifier" : {
  "Fn::If" : [
    "UseDBSnapshot",
    {"Ref" : "DBSnapshotName"},
    {"Ref" : "AWS::NoValue"}
  ]
}
```

YAML

```
DBSnapshotIdentifier: !If
- UseDBSnapshot
- !Ref DBSnapshotName
- !Ref "AWS::NoValue"
```

예제

주제

- [환경 기반 리소스 생성](#)
- [다중 조건 리소스 프로비저닝](#)

환경 기반 리소스 생성

다음 예에서는 EC2 인스턴스를 프로비저닝하고, 환경 유형이 `prod`인 경우에만 새 EBS 볼륨을 조건부로 생성 및 연결합니다. 환경이 `test`인 경우에는 추가 볼륨 없이 EC2 인스턴스만 생성합니다.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "EnvType": {
      "Description": "Environment type",
      "Default": "test",
```

```
        "Type": "String",
        "AllowedValues": [
            "prod",
            "test"
        ],
        "ConstraintDescription": "must specify prod or test"
    }
},
"Conditions": {
    "IsProduction": {
        "Fn::Equals": [
            {
                "Ref": "EnvType"
            },
            "prod"
        ]
    }
},
"Resources": {
    "EC2Instance": {
        "Type": "AWS::EC2::Instance",
        "Properties": {
            "ImageId": "ami-1234567890abcdef0",
            "InstanceType": "c5.xlarge"
        }
    },
    "MountPoint": {
        "Type": "AWS::EC2::VolumeAttachment",
        "Condition": "IsProduction",
        "Properties": {
            "InstanceId": {
                "Ref": "EC2Instance"
            },
            "VolumeId": {
                "Ref": "NewVolume"
            },
            "Device": "/dev/sdh"
        }
    },
    "NewVolume": {
        "Type": "AWS::EC2::Volume",
        "Condition": "IsProduction",
        "Properties": {
            "Size": 100,
```

```

        "AvailabilityZone": {
            "Fn::GetAtt": [
                "EC2Instance",
                "AvailabilityZone"
            ]
        }
    }
}

```

YAML

```

AWSTemplateFormatVersion: 2010-09-09
Parameters:
  EnvType:
    Description: Environment type
    Default: test
    Type: String
    AllowedValues:
      - prod
      - test
    ConstraintDescription: must specify prod or test
Conditions:
  IsProduction: !Equals
    - !Ref EnvType
    - prod
Resources:
  EC2Instance:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: ami-1234567890abcdef0
      InstanceType: c5.xlarge
  MountPoint:
    Type: AWS::EC2::VolumeAttachment
    Condition: IsProduction
    Properties:
      InstanceId: !Ref EC2Instance
      VolumeId: !Ref NewVolume
      Device: /dev/sdh
  NewVolume:
    Type: AWS::EC2::Volume
    Condition: IsProduction

```

```

Properties:
  Size: 100
  AvailabilityZone: !GetAtt
    - EC2Instance
    - AvailabilityZone

```

다중 조건 리소스 프로비저닝

다음 예에서는 버킷 이름이 제공된 경우 조건부로 S3 버킷을 생성하고, 환경이 prod로 설정된 경우에만 버킷 정책을 연결합니다. 버킷 이름이 지정되지 않았거나 환경이 test인 경우에는 리소스가 생성되지 않습니다.

JSON

```

{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "EnvType": {
      "Type": "String",
      "AllowedValues": [
        "prod",
        "test"
      ]
    },
    "BucketName": {
      "Default": "",
      "Type": "String"
    }
  },
  "Conditions": {
    "IsProduction": {
      "Fn::Equals": [
        {
          "Ref": "EnvType"
        },
        "prod"
      ]
    }
  },
  "CreateBucket": {
    "Fn::Not": [
      {
        "Fn::Equals": [

```

```

                "Ref": "BucketName"
            },
            ""
        ]
    }
]
},
"CreateBucketPolicy": {
    "Fn::And": [
        {
            "Condition": "IsProduction"
        },
        {
            "Condition": "CreateBucket"
        }
    ]
}
},
"Resources": {
    "Bucket": {
        "Type": "AWS::S3::Bucket",
        "Condition": "CreateBucket",
        "Properties": {
            "BucketName": {
                "Ref": "BucketName"
            }
        }
    },
    "Policy": {
        "Type": "AWS::S3::BucketPolicy",
        "Condition": "CreateBucketPolicy",
        "Properties": {
            "Bucket": {
                "Ref": "Bucket"
            },
            "PolicyDocument": { ... }
        }
    }
}
}
}

```


YAML

```
AWSTemplateFormatVersion: 2010-09-09
Parameters:
  EnvType:
    Type: String
    AllowedValues:
      - prod
      - test
  BucketName:
    Default: ''
    Type: String
Conditions:
  IsProduction: !Equals
    - !Ref EnvType
    - prod
  CreateBucket: !Not
    - !Equals
      - !Ref BucketName
      - ''
  CreateBucketPolicy: !And
    - !Condition IsProduction
    - !Condition CreateBucket
Resources:
  Bucket:
    Type: AWS::S3::Bucket
    Condition: CreateBucket
    Properties:
      BucketName: !Ref BucketName
  Policy:
    Type: AWS::S3::BucketPolicy
    Condition: CreateBucketPolicy
    Properties:
      Bucket: !Ref Bucket
      PolicyDocument: ...
```

이 예제에서 CreateBucketPolicy 조건은 Condition 키를 사용하여 다른 조건을 참조하는 방법을 보여줍니다. 정책은 IsProduction 및 CreateBucket 조건이 모두 true로 평가되는 경우에만 생성됩니다.

Note

더 복잡한 조건 사용 예제는 CloudFormation 템플릿 참조 가이드의 [Condition 속성](#) 주제를 참조하세요.

CloudFormation 템플릿 Transform 섹션

선택 사항인 Transform 섹션은 CloudFormation에서 템플릿을 어떤 방식으로 처리하는 데 사용하는 하나 이상의 매크로를 지정합니다.

매크로는 텍스트 찾기 및 바꾸기와 같은 간단한 작업을 수행하거나 전체 템플릿을 보다 광범위하게 변환할 수 있습니다. CloudFormation은 지정된 순서에 따라 매크로를 실행합니다. 변경 세트를 생성할 때 CloudFormation에서 처리된 템플릿 콘텐츠를 포함하는 변경 세트를 생성합니다. 변경 사항을 검토하고 변경 세트를 실행할 수 있습니다. 매크로 작동 방식에 대한 자세한 내용은 [템플릿 매크로를 사용하여 CloudFormation 템플릿에서 사용자 지정 처리 수행](#) 단원을 참조하세요.

CloudFormation은 CloudFormation에서 호스팅되는 매크로인 변환도 지원합니다. CloudFormation은 이러한 변환을 실행 순서 및 범위 측면에서 사용자가 생성한 모든 매크로와 동일하게 처리합니다. 자세한 내용은 [변환 참조](#)를 참조하세요.

여러 매크로를 선언하려면 목록 형식을 사용하고 하나 이상의 매크로를 지정합니다.

예를 들어, 아래 템플릿 샘플에서 CloudFormation은 MyMacro를 평가하고 AWS::Serverless를 평가합니다. 두 가지는 모두 전체 템플릿의 콘텐츠를 처리할 수 있는데, 이는 Transform 섹션이 포함되어 있기 때문입니다.

```
# Start of processable content for MyMacro and AWS::Serverless
Transform:
  - MyMacro
  - 'AWS::Serverless'
Resources:
  WaitCondition:
    Type: AWS::CloudFormation::WaitCondition
  MyBucket:
    Type: AWS::S3::Bucket
    Properties:
      BucketName: amzn-s3-demo-bucket
      Tags: [{"key": "value"}]
      CorsConfiguration: []
```

```
MyEc2Instance:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: ami-1234567890abcdef0
# End of processable content for MyMacro and AWS::Serverless
```

CloudFormation 템플릿 형식 버전 구문

AWSTemplateFormatVersion 섹션(선택 사항)은 템플릿이 따르는 템플릿 형식 버전을 식별합니다. 최신 템플릿 포맷 버전은 2010-09-09이며 현재 유일한 유효 값입니다.

템플릿 포맷 버전은 API 버전과 같지 않습니다. 템플릿 포맷 버전은 API 버전과 상관없이 변경될 수 있습니다.

템플릿 포맷 버전 선언 값은 리터럴 문자열이어야 합니다. 파라미터나 함수를 사용하여 템플릿 포맷 버전을 지정할 수 없습니다. 값을 지정하지 않을 경우 CloudFormation에서는 최신 템플릿 포맷 버전이라고 가정합니다. 다음 코드 조각은 템플릿 포맷 버전 선언 예입니다.

JSON

```
"AWSTemplateFormatVersion" : "2010-09-09"
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
```

CloudFormation 템플릿 Description 구문

템플릿의 Description 섹션(선택 사항)을 사용하면 템플릿을 설명하는 텍스트 문자열을 포함시킬 수 있습니다. 이 섹션은 항상 템플릿 포맷 버전 섹션 다음에 이어져야 합니다.

설명 선언 값은 0 ~ 1023바이트 길이의 리터럴 문자열이어야 합니다. 파라미터나 함수를 사용하여 설명을 지정할 수 없습니다. 다음 코드 조각은 설명 선언을 보여주는 예입니다.

Important

스택 업데이트 중에는 Description 섹션을 자체적으로 업데이트할 수 없습니다. 리소스를 추가, 수정 또는 삭제하는 변경 내용을 포함할 때만 업데이트할 수 있습니다.

JSON

```
"Description" : "Here are some details about the template."
```

YAML

```
Description: > Here are some details about the template.
```

Infrastructure Composer를 사용하여 시각적으로 템플릿 생성

AWS 인프라 컴포저(구 Application Composer)를 사용하면 AWS에서 최신 애플리케이션을 시각적으로 작성하고 구성할 수 있습니다. 코드를 작성할 필요없이 다양한 리소스를 끌어서 놓아 애플리케이션을 시각적으로 작성할 수 있습니다.

CloudFormation 콘솔 모드의 Infrastructure Composer는 시각적으로 CloudFormation 템플릿 작업을 하는 권장 도구입니다. CloudFormation 콘솔에서 액세스할 수 있는 이 Infrastructure Composer 버전은 CloudFormation Designer라는 이전 도구를 개선한 것입니다.

CloudFormation 콘솔 모드의 Infrastructure Composer를 사용하면 카드는 다양한 리소스를 시각적 캔버스로 끌어서 놓고 구성하고 연결할 수 있습니다. 이 시각적 접근 방식을 사용하면 직접 템플릿으로 작업하지 않고도 애플리케이션 아키텍처를 쉽게 설계하고 편집할 수 있습니다. [CloudFormation 콘솔](#)에서 이 모드에 액세스하려면 왼쪽 탐색 메뉴에서 Infrastructure Composer를 선택합니다.

자세한 내용은 AWS 인프라 컴포저 개발자 안내서의 [AWS 인프라 컴포저에서 구성하는 방법](#)을 참조하세요.

CloudFormation 콘솔 모드에서 Infrastructure Composer를 사용하는 이유는 무엇인가요?

Infrastructure Composer에서 템플릿을 시각화하면 CloudFormation 템플릿과 애플리케이션 아키텍처의 격차와 개선 영역을 식별하는 데 도움이 됩니다. Infrastructure Composer를 사용하면 CloudFormation 스택을 쉽고 효율적이고 시각적으로 구축하고 수정할 수 있어 개발 경험이 향상됩니다. 초기 초안부터 시작하여 배포 가능한 코드를 생성하고 개발자 워크플로를 Infrastructure Composer의 비주얼 디자이너와 통합할 수 있습니다.

이 모드는 Infrastructure Composer 콘솔과 어떻게 다른가요?

CloudFormation 콘솔 버전의 Infrastructure Composer는 표준 Infrastructure Composer 콘솔과 기능이 유사하지만 몇 가지 차이점이 있습니다. Lambda 관련 카드(Lambda 함수 및 Lambda 계층)에는

CloudFormation 콘솔 모드의 Infrastructure Composer에서 사용할 수 없는 코드 빌드 및 패키징 솔루션이 필요합니다. 이 모드에서는 로컬 동기화도 사용할 수 없습니다.

하지만 [Infrastructure Composer 콘솔](#) 또는 AWS Toolkit for Visual Studio Code에서 이러한 Lambda 관련 카드와 로컬 동기화 기능을 사용할 수 있습니다. 자세한 내용은 [AWS 인프라 컴포저 개발자 안내서](#)와 AWS Toolkit for Visual Studio Code 사용 설명서의 [Infrastructure Composer](#)를 참조하세요.

AWS CloudFormation 언어 서버

AWS CloudFormation 언어 서버는 코드형 인프라(IaC) 작성을 가속화하고 AWS 리소스를 안전하고 자신 있게 배포하는 기능을 제공합니다. [언어 서버 프로토콜](#)(LSP)에 따라 가리키기, 자동 완성, 정적 검증을 통한 진단에 대한 설명서를 제공하고 정의 및 코드 작업으로 이동합니다. 이러한 기존 언어 서버 기능 외에도 서버는 CloudFormation을 통해 AWS 리소스를 탐색 및 배포하는 온라인 기능을 추가합니다. 여기에는 변경 세트를 사용하여 템플릿을 검증 및 배포하고, 스택 차이, 이벤트, 리소스 및 출력을 보며, 스택을 나열하고 유형별로 리소스를 찾아보고, 라이브 리소스 상태를 CloudFormation 템플릿에 직접 삽입하는 기능이 포함됩니다. 개발자 도구 및 통합 개발 환경(IDE)은 언어 서버 기능을 활용하여 CloudFormation에서 IaC 개발자의 경험을 개선할 수 있습니다.

주제

- [AWS CloudFormation 언어 서버와 통합되는 IDE](#)
- [시작하기](#)
- [IDE에서 CloudFormation 프로젝트 초기화](#)
- [오픈 소스](#)
- [도움이 필요하세요?](#)

AWS CloudFormation 언어 서버와 통합되는 IDE

[AWS Toolkit](#) 확장 프로그램은 CloudFormation 언어 서버를 통합하여 작성 환경을 개선합니다. 이 통합은 현재 다음을 포함하여 VS Code 기반 IDE에서 지원됩니다.

- [Visual Studio Code](#)
- [Kiro](#)
- [Cursor](#)

시작하기

주제

- [사전 조건](#)
- [1단계: AWS Toolkit 설치 또는 업그레이드](#)
- [2단계: AWS Toolkit 패널에서 CloudFormation에 액세스](#)
- [3단계: 템플릿 검증, 테스트 및 세부 조정](#)
- [4단계: 템플릿 탐색](#)
- [5단계: 검증 및 배포](#)

사전 조건

시작하기 전에 다음을 확인하세요.

- 지원되는 운영 체제(macOS, Windows 또는 Linux)에서 VS Code 기반 IDE를 사용하고 있습니다.
- IDE에 대한 [AWS Toolkit](#)의 최신 버전을 설치했거나 해당 버전으로 업그레이드했습니다.

AWS CloudFormation 언어 서버의 일부 기능에는 활성 AWS 계정 및 구성된 자격 증명이 필요합니다. 유효한 자격 증명을 사용하여 AWS Toolkit를 통해 AWS 계정에 로그인해야 합니다.

1단계: AWS Toolkit 설치 또는 업그레이드

- IDE의 확장 프로그램 또는 플러그인 관리자를 엽니다.
- AWS Toolkit을 검색합니다.
- IDE에 대한 AWS Toolkit 버전 3.85.0 이상을 설치하거나 업데이트합니다.
- 설치 후 IDE를 다시 시작합니다.

참고: 설치 후 AWS Toolkit는 CloudFormation IDE 지원을 자동으로 활성화합니다. AWS CloudFormation 언어 서버 확장 프로그램을 사용하여 AWS Toolkit를 처음 설치하거나 업그레이드할 때 익명 사용 데이터를 수집할 수 있는 AWS에 대한 권한을 부여하라는 메시지가 표시됩니다. 이 데이터는 AWS에서 CloudFormation 언어 서버 및 작성 환경을 개선하는 데 도움이 됩니다. 민감한 정보는 수집되지 않으며 AWS에서는 템플릿 콘텐츠, 리소스 구성 또는 식별 가능한 고객 데이터를 기록하거나 저장하지 않습니다. IDE 설정에서 언제든지 원격 측정 기본 설정을 변경할 수 있습니다. 수집된 사용 데이터는 기능 상호 작용 및 성능 지표에만 중점을 둡니다. 이러한 인사이트를 통해 AWS에서 더 빠

른 검증, 개선된 자동 완성, 더 나은 오류 진단과 같은 개선 사항을 식별하고 우선순위를 지정할 수 있습니다.

2단계: AWS Toolkit 패널에서 CloudFormation에 액세스

IDE의 활동 표시줄에서 AWS Toolkit 패널을 엽니다. AWS Toolkit에서 CLOUDFORMATION을 선택합니다.

패널 섹션:

- 리전: 현재 AWS 리전을 표시합니다. 리전 이름을 선택하거나 명령 팔레트에서 AWS CloudFormation: 리전 선택 명령을 사용하여 변경할 수 있습니다.
- 환경: CFN init 도중 선택한 환경(예: 개발, 테스트 또는 프로덕션)을 나타냅니다. 이 값은 환경 선택 단계를 완료한 후에만 나타납니다.
- 스택: 계정에서 CloudFormation 스택의 페이지가 매겨진 목록을 표시합니다.
 - 새로 고침 아이콘을 클릭하여 스택 목록을 업데이트합니다.
 - + 아이콘을 사용하여 새 템플릿을 스택으로 배포합니다.
 - 각 스택 항목에는 다음이 포함됩니다.
 - 개요: 스택 요약 및 상태 표시
 - 스택 ID
 - 설명
 - 생성 시간 및 업데이트 시간
 - 상태 및 상태 이유
 - 이벤트, 출력, 리소스
- 변경 세트
- 리소스: 리소스 유형을 선택한 후 패널에는 계정에서 해당 유형의 AWS 리소스가 표시됩니다. 보거나 새로 고치거나 복사하거나 템플릿으로 가져올 수 있습니다.

3단계: 템플릿 검증, 테스트 및 세부 조정

CloudFormation 템플릿을 작성할 때 IDE는 정확하고 규정을 준수하는 인프라를 더 빠르게 생성하는데 도움이 되는 지능형 작성 지원 기능을 제공합니다. CloudFormation 언어 서버는 백그라운드에서 실행되며 다음과 같은 작성 기능을 제공합니다.

- 코드 완성: CloudFormation 스키마를 기반으로 리소스 유형, 파라미터, 속성을 제안합니다.

- 기존 AWS 리소스 추가: AWS 계정에서 템플릿으로 기존 리소스를 가져올 수 있습니다. IDE는 [AWS Cloud Control API\(CCAPI\)](#)를 사용하여 리소스의 라이브 구성 및 속성을 검색하며, 이를 통해 템플릿 내 기존 인프라를 복제하거나 재사용할 수 있습니다.

템플릿에 리소스를 추가하는 방법

- CloudFormation 리소스 패널 확장: AWS Toolkit 사이드 패널에서 CloudFormation을 확장한 다음 리소스에서 추가 + 아이콘을 클릭합니다.
- 리소스 유형 검색: 특정 AWS 리소스 유형을 찾으려면 검색 표시줄에 추가하려는 특정 AWS 리소스 유형을 입력합니다. 예제:
 - `AWS::S3::Bucket`
 - `AWS::Lambda::Function`

명령 팔레트에서 AWS CloudFormation: 리소스 유형 추가를 사용하여 리소스 유형을 선택할 수 있습니다.

- 템플릿에 리소스 추가: 리소스 패널 아래에 계정에서 감지된 AWS 리소스의 페이지가 매겨진 목록이 표시됩니다. 리소스가 많은 경우 첫 번째 페이지만 표시됩니다. 패널 하단의 탐색 제어를 사용하여 추가 페이지를 이동하고 사용 가능한 모든 리소스를 봅니다.
- 템플릿에 포함하려는 리소스를 선택합니다.
- 다음과 같이 목표에 따라 두 가지 방법으로 템플릿에 리소스를 삽입할 수 있습니다.
 - 기존 리소스 복제: 기존 AWS 리소스의 라이브 구성 및 속성을 사용하여 템플릿에서 새 리소스를 생성합니다.
 - 기존 리소스 가져오기: 라이브 상태를 사용하여 템플릿에 추가해 스택에 실제 리소스를 삽입합니다.

팁

- 언제든지 리소스 패널을 새로 고쳐 계정 또는 리전에서 사용 가능한 최신 리소스 목록을 볼 수 있습니다.
- 리소스를 가져오는 경우 동일한 계정의 기존 CloudFormation 스택에 이미 속한 리소스를 추가하지 마세요.
- 리소스가 CloudFormation에서 이미 관리되고 있는지 확인하려면 리소스 이름 옆에 있는 i(정보) 아이콘을 클릭합니다.
- 또는 명령 AWS CloudFormation: 스택 관리 정보 가져오기를 사용하여 리소스가 스택에 속하는지 확인할 수 있습니다.

관련 리소스 추가

명령 AWS CloudFormation: 유형별 관련 리소스 추가를 사용하여 선택한 리소스에 관련 리소스를 추가할 수 있습니다. 템플릿에 이미 정의된 리소스 유형에서 리소스 유형을 선택하면 IDE는 일반적으로 해당 유형에 연결되거나 종속된 리소스 목록을 표시합니다. 예를 들어 `AWS::EC2::Instance`를 선택하면 IDE에서 `AWS::EC2::SecurityGroup` 또는 `AWS::EC2::Subnet`과 같은 관련 리소스 추가를 제안할 수 있습니다. 이 기능을 사용하면 호환되는 리소스 유형을 수동으로 검색하지 않고도 연결된 인프라 구성 요소를 빠르게 빌드할 수 있습니다.

정적 검증

CloudFormation IDE는 [AWS CloudFormation Linter\(cfn-lint\)](#) 및 [AWS CloudFormation Guard](#)로 구동되는 기본 제공 정적 검증을 제공합니다. 이러한 검증은 템플릿을 작성할 때 백그라운드에서 실행되므로 배포 전에 구문 오류, 규정 준수 격차, 모범 사례 문제를 식별하는 데 도움이 됩니다.

정적 검증 개요

IDE에는 다음과 같은 두 가지 유형의 실시간 정적 검증이 표시됩니다.

- CloudFormation Linter(cfn-lint): CloudFormation 리소스 사양 및 스키마 규칙을 기준으로 템플릿을 검증합니다.
- Guard(cfn-guard): 규정 준수 규칙 및 조직 정책 팩을 기준으로 템플릿을 검증합니다.

CloudFormation Linter(cfn-lint)

CloudFormation Linter는 IDE에 통합되어 입력 시 템플릿 구문 및 구조를 자동으로 확인합니다.

- 스키마 검증: 구문 및 스키마 오류를 감지하여 템플릿이 CloudFormation 리소스 스키마를 준수하는지 확인합니다.
- 오류 강조 표시: 문제 아래에 빨간색 또는 노란색의 물결선을 표시하여 배포 블로커 또는 경고를 나타냅니다.
- 가리킬 때 도움말 표시: 오류 위로 마우스를 가져가면 IDE에서 해당 문제와 관련된 진단 메시지를 표시합니다. 빠른 수정이 가능한 경우 마우스 포인터 패널에서도 제공됩니다.

Guard 통합

Guard는 규정 준수 및 보안 정책을 정의하는 규칙 세트를 기준으로 템플릿을 검증합니다. IDE는 CloudFormation 언어 서버를 통해 실시간으로 Guard 검증을 실행하여 템플릿을 작성하는 동안 즉각적인 피드백을 제공합니다.

- 기본 규칙 팩: IDE에는 리소스 보안 및 구성 위생을 위한 기본 모범 사례에 초점을 맞춘 사전 등록된 Guard 규칙 세트가 있습니다. [Guard 규칙 레지스트리](#)에 대한 추가 링크를 알아보는 방법.
- 규칙 팩 추가: 규칙 세트를 추가하거나 수정하려면 설정, Guard를 차례로 열고 추가 Guard 규칙 팩을 선택하거나 업로드합니다.

팁: 물결선 표시 이해

- 파란색의 물결선: 모범 사례 힌트 또는 최적화 권장 사항.
- 노란색의 물결선: 차단 외 문제(예: 태그 또는 파라미터 누락)에 대한 경고.
- 빨간색의 물결선: 유효하지 않은 속성 이름, 필수 필드 누락 또는 스키마 불일치와 같은 배포 블록.

4단계: 템플릿 탐색

탐색기에서 템플릿 파일을 클릭하면 개요 패널에 CloudFormation 템플릿의 구조화된 계층 보기가 자동으로 표시됩니다. 템플릿을 Parameters, Resources, Outputs, Mappings와 같은 섹션으로 구성하고 각 리소스 유형과 논리적 ID를 표시합니다. 그러면 대규모 템플릿 내에서 특정 리소스 또는 파라미터를 빠르게 찾고 탐색할 수 있습니다.

GetAtt 및 Ref와 같은 내장 함수에 대한 정의로 이동을 사용하면 템플릿에서 참조된 리소스 또는 파라미터로 직접 이동할 수 있습니다. 이를 통해 종속성을 추적하고, 리소스 관계를 이해하며, 보다 효율적으로 편집할 수 있습니다.

5단계: 검증 및 배포

CloudFormation 템플릿을 배포할 준비가 되면 명령 팔레트를 열고 AWS CloudFormation: 검증 및 배포를 실행합니다. 명령은 템플릿을 검증하고 차단 오류가 발견되지 않으면 배포 단계로 진행합니다. 확인 전에 IDE는 [드리프트 인식 변경 세트](#) 요약과 차이 보기를 표시하므로 제안된 모든 변경 사항을 검토할 수 있습니다.

검증 작동 방식

IDE는 [배포 전에 검증 검사](#)를 자동으로 수행하고 다음과 같은 일반적인 장애 원인에 대해 템플릿을 검증합니다.

- 유효하지 않은 속성 구문 또는 스키마 불일치: 이러한 문제는 일반적으로 작성 중에 cfn-lint에서 발견하지만 사용자가 문제를 해결하지 않고 배포를 진행하는 경우 CloudFormation의 배포 시 검증은 스택이 생성되거나 업데이트되기 전에 동일한 오류를 표시합니다.

- 리소스 이름이 계정의 기존 리소스와 충돌합니다.
- S3 버킷 이름 충돌 또는 암호화 누락과 같은 서비스별 제약 조건.

검증에서 오류가 감지되면 IDE는 빨간색 또는 노란색의 물결선을 사용하여 템플릿에서 직접 문제를 강조 표시하고 문제 패널에 오류를 나열합니다. 각 문제에는 실패의 원인이 되는 특정 속성 또는 리소스와 제안된 수정 사항이 포함되어 있습니다. 차단 오류가 없는 경우 배포 단계로 진행할 수 있습니다.

경고(차단 외 문제)가 발견되면 배포를 진행하거나 취소하고 수정할 수 있는 대화 상자가 나타납니다.

IDE는 현재 템플릿과 배포된 스택 구성 간의 차이를 표시하는 [드리프트 인식 변경 세트](#)를 엽니다. 이를 통해 실행 전에 변경 세트를 검토, 확인 또는 취소할 수 있습니다.

참고: 드리프트 인식 변경 세트는 스택 드리프트를 안전하게 처리할 수 있도록 함으로써 CloudFormation 배포 프로세스를 개선합니다. 스택 드리프트는 리소스의 실제 상태가 CloudFormation 템플릿에 정의된 상태와 다를 때 발생합니다. 종종 AWS Management Console, CLI 또는 SDK를 통해 수행된 수동 변경으로 인해 발생합니다. CloudFormation의 [드리프트 인식 변경 세트](#)가 처리된 스택 구성을 라이브 리소스 상태와 비교하고, IDE에서 이러한 차이를 표시하므로 배포 전에 리소스를 다시 규정 준수 상태로 되돌릴 수 있습니다.

스택 이벤트 보기

배포가 시작되면 패널의 CloudFormation 탭으로 이동하여 진행 상황을 실시간으로 모니터링할 수 있습니다. 스택 이벤트 아래에서 배포 중에 수행된 작업 목록이 표시됩니다. 각 이벤트에는 다음과 같은 세부 정보가 포함됩니다.

- 타임스탬프: 이벤트가 발생한 시간
- 리소스: 생성, 업데이트 또는 삭제되는 특정 AWS 리소스
- 상태: 작업의 현재 상태(예: CREATE_IN_PROGRESS, UPDATE_COMPLETE 또는 ROLLBACK_IN_PROGRESS)
- 이유: 해당하는 경우 추가 컨텍스트 또는 오류 메시지

이 패널에서 스택의 리소스 및 출력을 볼 수도 있습니다. AWS Management Console에서 스택을 열려면 스택 이름 옆에 있는 외부 링크 아이콘을 사용합니다. 스택 이벤트 보기를 통해 배포 진행 상황을 추적하고, 잠재적 문제를 식별하며, 스택이 성공적으로 완료된 시점을 확인할 수 있습니다.

IDE에서 CloudFormation 프로젝트 초기화

IDE에서 CloudFormation 프로젝트를 초기화하면 올바른 폴더, 환경 구성 및 AWS 자격 증명으로 구조화된 작업 영역을 설정하여 신뢰할 수 있는 방식으로 템플릿을 검증하고 배포할 수 있습니다. IDE에서 직접 새 CloudFormation 프로젝트를 초기화하여 이 권장 설정을 생성할 수 있습니다.

CloudFormation 프로젝트를 초기화하는 방법:

- 명령 팔레트 열기
 - IDE에서 명령 팔레트(Ctrl+Shift+P 또는 macOS의 Cmd+Shift+P)를 엽니다.
 - AWS CloudFormation: CFN Init: 프로젝트 초기화를 선택합니다.
- 프로젝트 디렉터리 선택
 - 기본적으로 IDE는 현재 작업 디렉터리를 사용합니다.
 - 이 경로를 CloudFormation 템플릿을 저장하려는 폴더로 변경할 수 있습니다.
- AWS 자격 증명 프로파일 선택
 - AWS 자격 증명 프로파일을 선택하라는 메시지가 표시됩니다. 선택한 프로파일은 환경 감지, 검증 및 배포에 사용됩니다.
- 환경 설정
 - 환경을 생성하거나 선택하라는 메시지가 표시됩니다.
 - 환경에서는 템플릿을 배포하거나 검증할 위치와 방법을 정의합니다(예: 개발, 베타 또는 프로덕션). AWS CloudFormation: CFN Init: 환경 추가를 사용하여 환경을 선택하거나 변경할 수 있습니다.
 - AWS CloudFormation: CFN Init: 환경 제거를 사용하여 선택한 환경을 제거할 수 있습니다.
- (선택 사항) 파라미터 파일 가져오기
 - 기존 파라미터 파일이 이미 있는 경우 IDE를 사용하여 초기화 중에 해당 파일을 가져올 수 있습니다.
 - IDE는 호환되는 파일을 자동으로 감지하고 템플릿 검증 및 배포에 사용할 수 있도록 프로젝트에 연결합니다.
- 프로젝트 이름 지정 및 마무리
 - beta-environment와 같은 프로젝트 이름을 제공하고 설정을 완료합니다.
 - IDE는 사용자를 대신해 초기 프로젝트 구조 및 구성 파일을 생성합니다.

IDE에서 직접 검증을 실행하거나 배포를 미리 보거나 여러 환경 사이를 전환할 수 있습니다.

오픈 소스

AWS CloudFormation 언어 서버는 Apache-2.0 라이선스에 따라 오픈 소스로 제공되므로 고객에게 템플릿 진단, 스키마 검증 및 정적 분석을 수행하는 방법에 대한 완전한 투명성을 제공합니다. 이를 통해 도구를 채택하기 전에 소스 수준의 가시성이 필요한 고객 측에서 보안 및 규정 준수 마찰이 줄어듭니다.

코드 베이스는 GitHub(<https://github.com/aws-cloudformation/cloudformation-languageserver/>)에서 공개적으로 사용할 수 있습니다.

도움이 필요하세요?

AWS re:Post에서 [CloudFormation 커뮤니티](#)를 이용해 보세요.

laC 생성기를 사용하여 기존 리소스에서 템플릿 생성

CloudFormation 인프라를 코드 생성기(laC 생성기)로 사용하면 계정에 프로비저닝된 리소스 중에 CloudFormation에서 아직 관리하지 않는 AWS 리소스를 사용하여 템플릿을 생성할 수 있습니다.

laC 생성기의 장점은 다음과 같습니다.

- 전체 애플리케이션을 CloudFormation 관리에 포함하거나 AWS CDK 앱으로 마이그레이션합니다.
- 리소스 속성을 속성별로 설명할 필요 없이 템플릿을 생성한 다음 이를 JSON 또는 YAML 구문으로 변환합니다.
- 템플릿을 사용하여 새 계정 또는 리전에서 리소스를 복제합니다.

laC 생성 프로세스는 다음 단계로 구성됩니다.

1. 리소스 스캔 - 첫 번째 단계는 리소스 스캔을 시작하는 것입니다. 이 스캔은 리전 전체에 적용되며 30일 후에 만료됩니다. 이 시간 동안 동일한 스캔에서 템플릿을 여러 개 생성할 수 있습니다.
2. 템플릿 생성 - 템플릿을 생성하려면 다음 두 가지 옵션이 있습니다.
 - 처음부터 새 템플릿을 생성하고 스캔한 리소스와 관련 리소스를 추가합니다.
 - 기존 CloudFormation 스택을 시작점으로 사용하고 템플릿에 스캔한 리소스와 관련 리소스를 추가합니다.
3. 리소스 가져오기 - 템플릿을 사용하여 리소스를 CloudFormation 스택으로 가져오거나 AWS CDK 앱으로 마이그레이션합니다.

laC 생성기 특성은 모든 상업 리전에서 사용할 수 있으며 많은 일반적인 AWS 리소스 유형을 지원합니다. 지원되는 전체 리소스 목록은 [리소스 유형 지원](#) 섹션을 참조하세요.

주제

- [고려 사항](#)
- [리소스 스캔에 필요한 IAM 권한](#)
- [템플릿 생성, 관리 및 삭제에 일반적으로 사용되는 명령](#)
- [AWS CDK로 템플릿 마이그레이션](#)
- [CloudFormation laC 생성기로 리소스 스캔 시작](#)
- [CloudFormation 콘솔의 스캔 요약 보기](#)
- [laC 생성기로 스캔한 리소스로 CloudFormation 템플릿 생성](#)
- [스캔한 리소스에서 CloudFormation 스택 생성](#)
- [쓰기 전용 속성 해결](#)

고려 사항

읽기 액세스 권한이 있는 AWS 리소스에 대해 JSON 또는 YAML 템플릿을 생성할 수 있습니다. laC 생성기 기능의 템플릿은 리소스 속성을 속성별로 설명할 필요 없이 클라우드 리소스를 안정적이고 빠르게 모델링합니다.

다음 표에는 laC 생성 특성에 사용할 수 있는 할당량이 나와 있습니다.

이름	전체 스캔	부분 스캔
스캔에서 처리할 수 있는 최대 리소스 수	100,000건	100,000건
일일 스캔 횟수(리소스가 10,000개 미만인 스캔의 경우)	10	10
일일 스캔 횟수(리소스가 10,000개를 초과하는 스캔의 경우)	1	1
계정당 생성되는 템플릿의 동시 개수	5	5
한 번의 템플릿 생성을 위해 모델링된 동시 리소스 수	5	5
하나의 템플릿에서 모델링할 수 있는 총 리소스 수	500	500

이름	전체 스캔	부분 스캔
계정당 생성되는 최대 템플릿 수	1,000	1,000

Important

IaC 생성기는 리전의 Cloud Control API에서 지원하는 AWS 리소스만 지원합니다. 자세한 내용은 [리소스 유형 지원](#) 섹션을 참조하세요.

리소스 스캔에 필요한 IAM 권한

IaC 생성기로 리소스를 스캔하려면 IAM 위탁자(사용자, 역할 또는 그룹)에 다음이 있어야 합니다.

- CloudFormation 스캔 권한
- 대상 AWS 서비스에 대한 읽기 권한

스캔 범위는 읽기 액세스 권한이 있는 리소스로 제한됩니다. 권한이 누락되어도 스캔에 실패하지 않지만 해당 리소스는 제외됩니다.

스캔 및 템플릿 관리 권한을 부여하는 IAM 정책의 예는 [모든 IaC 생성기 작업 허용](#) 섹션을 참조하세요.

템플릿 생성, 관리 및 삭제에 일반적으로 사용되는 명령

IaC 생성기 작업에 일반적으로 사용되는 명령은 다음과 같습니다.

- [start-resource-scan](#) - AWS 리전의 계정에서 리소스 스캔을 시작합니다.
- [describe-resource-scan](#) - 리소스 스캔의 진행 상황을 모니터링합니다.
- [list-resource-scans](#)에 AWS 리전의 리소스 스캔이 나열됩니다.
- [list-resource-scan-resources](#) - 리소스 스캔 중 찾은 리소스를 나열합니다.
- [list-resource-scan-related-resources](#) - 스캔한 리소스와 관련된 리소스를 나열합니다.
- [create-generated-template](#) - 스캔한 리소스 세트에서 CloudFormation 템플릿을 생성합니다.
- [update-generated-template](#) - 생성된 템플릿을 업데이트합니다.
- [describe-generated-template](#) - 생성된 템플릿에 대한 정보를 반환합니다.
- [list-generated-templates](#) - 계정과 현재 리전에서 생성된 모든 템플릿을 나열합니다.

- [delete-generated-template](#) - 생성된 템플릿을 삭제합니다.

AWS CDK로 템플릿 마이그레이션

AWS Cloud Development Kit (AWS CDK)는 널리 사용되는 프로그래밍 언어를 사용하여 CloudFormation 리소스를 개발, 관리, 배포하는 데 사용할 수 있는 오픈 소스 소프트웨어 개발 프레임 워크입니다.

AWS CDK CLI는 IaC 생성기와의 통합 기능을 제공합니다. AWS CDK CLI `cdk migrate` 명령을 사용하여 CloudFormation 템플릿을 변환하고 리소스를 포함하는 새 CDK 앱을 생성합니다. 그런 다음 AWS CDK를 사용하여 리소스를 관리하고 CloudFormation에 배포할 수 있습니다.

자세한 내용은 AWS Cloud Development Kit (AWS CDK) 개발자 안내서의 [AWS CDK로 마이그레이션](#)을 참조하세요.

CloudFormation IaC 생성기로 리소스 스캔 시작

기존 리소스에서 템플릿을 생성하기 전에 먼저 리소스 스캔을 시작하여 현재 리소스 및 해당 관계를 검색해야 합니다.

다음 옵션 중 하나를 사용하여 리소스 스캔을 시작할 수 있습니다. IaC 생성기를 처음 사용하는 경우 첫 번째 옵션을 권장합니다.

- 모든 리소스 스캔(전체 스캔) - 현재 계정 및 리전의 모든 기존 리소스를 스캔합니다. 이 스캔 프로세스는 1,000개의 리소스를 스캔하는 데 최대 10분이 걸릴 수 있습니다.
- 특정 리소스 스캔(부분 스캔) - 현재 계정 및 리전에서 스캔할 리소스 유형을 수동으로 선택합니다. 이 옵션은 더 빠르고 집중적인 스캔 프로세스를 제공하므로 반복 템플릿 개발에 적합합니다.

스캔이 완료되면 템플릿을 생성할 때 포함할 리소스 및 관련 리소스를 선택할 수 있습니다. 부분 스캔을 사용하는 경우 다음 중 하나에 해당할 때에만 템플릿 생성 중에 관련 리소스를 사용할 수 있습니다.

- 스캔을 시작하기 전에 특별히 선택한 경우 또는
- 선택한 리소스 유형을 검색하는 데 필요한 경우.

예를 들어 `AWS::EKS::Cluster`를 선택하지 않고 `AWS::EKS::Nodegroup`을 선택하는 경우 노드 그룹을 검색하려면 먼저 클러스터를 검색해야 하므로 IaC 생성기는 스캔에 `AWS::EKS::Cluster` 리소스를 자동으로 포함합니다. 다른 모든 경우 스캔에는 특별히 사용자가 선택한 리소스만 포함됩니다.

Note

계속 진행하기 전에 IaC 생성기를 사용하는 데 필요한 권한이 있는지 확인합니다. 자세한 내용은 [리소스 스캔에 필요한 IAM 권한](#) 섹션을 참조하세요.

주제

- [리소스 스캔 시작\(콘솔\)](#)
- [리소스 스캔 시작\(AWS CLI\)](#)

리소스 스캔 시작(콘솔)

모든 리소스 유형의 리소스 스캔을 시작하는 방법(전체 스캔)

1. CloudFormation 콘솔의 [IaC 생성기 페이지](#)를 엽니다.
2. 화면 상단의 탐색 모음에서 스캔할 리소스가 포함된 AWS 리전을 선택합니다.
3. 스캔 패널에서 새 스캔 시작을 선택한 다음 모든 리소스 스캔을 선택하세요.

특정 리소스 유형의 리소스 스캔을 시작하는 방법(부분 스캔)

1. CloudFormation 콘솔의 [IaC 생성기 페이지](#)를 엽니다.
2. 화면 상단의 탐색 모음에서 스캔할 리소스가 포함된 AWS 리전을 선택합니다.
3. 스캔 패널에서 새 스캔 시작을 선택한 다음 특정 리소스 스캔을 선택하세요.
4. 부분 스캔 시작 대화 상자에서 최대 100개의 리소스 유형을 선택한 다음 스캔 시작을 선택하세요.

리소스 스캔 시작(AWS CLI)

모든 리소스 유형의 리소스 스캔을 시작하는 방법(전체 스캔)

다음 [start-resource-scan](#) 명령을 사용합니다. *us-east-1*을 스캔할 리소스가 포함된 AWS 리전으로 바꿉니다.

```
aws cloudformation start-resource-scan --region us-east-1
```

이 명령이 성공하면 스캔의 ARN을 반환합니다. ResourceScanId 속성의 ARN을 기록해 둡니다. 템플릿을 생성하는 데 필요합니다.

```
{
  "ResourceScanId":
    "arn:aws:cloudformation:region:account-id:resourceScan/0a699f15-489c-43ca-a3ef-3e6ecfa5da60"
}
```

특정 리소스 유형의 리소스 스캔을 시작하는 방법(부분 스캔)

1. [cat](#) 명령을 사용하여 홈 디렉터리에서 이름이 config.json인 JSON 파일에 스캔하려는 리소스 유형 및 식별자를 저장하세요. 다음은 Amazon EC2 인스턴스, 보안 그룹 및 모든 Amazon S3 리소스를 스캔하는 스캔 구성 예제입니다.

```
$ cat > config.json
[
  {
    "Types":[
      "AWS::EC2::Instance",
      "AWS::EC2::SecurityGroup",
      "AWS::S3::*"
    ]
  }
]
```

2. 생성한 config.json 파일과 함께 --scan-filters 옵션을 포함해 [start-resource-scan](#) 명령을 사용하여 부분 스캔을 시작하세요. *us-east-1*을 스캔할 리소스가 포함된 AWS 리전으로 바꿉니다.

```
aws cloudformation start-resource-scan --scan-filters file://config.json --
region us-east-1
```

이 명령이 성공하면 스캔의 ARN을 반환합니다. ResourceScanId 속성의 ARN을 기록해 둡니다. 템플릿을 생성하는 데 필요합니다.

```
{
  "ResourceScanId":
    "arn:aws:cloudformation:region:account-id:resourceScan/0a699f15-489c-43ca-a3ef-3e6ecfa5da60"
}
```

리소스 스캔 진행 상황을 모니터링하는 방법

[describe-resource-scan](#) 명령을 사용합니다. --resource-scan-id 옵션의 경우 샘플 ARN을 실제 ARN으로 바꿉니다.

```
aws cloudformation describe-resource-scan --region us-east-1 \
  --resource-scan-id arn:aws:cloudformation:us-east-1:123456789012:resourceScan/0a699f15-489c-43ca-a3ef-3e6ecfa5da60
```

이 명령이 제대로 실행되면 다음과 비슷한 출력이 반환됩니다.

```
{
  "ResourceScanId": "arn:aws:cloudformation:region:account-id:resourceScan/0a699f15-489c-43ca-a3ef-3e6ecfa5da60",
  "Status": "COMPLETE",
  "StartTime": "2023-08-21T03:10:38.485000+00:00",
  "EndTime": "2023-08-21T03:20:28.485000+00:00",
  "PercentageCompleted": 100.0,
  "ResourceTypes": [
    "AWS::CloudFront::CachePolicy",
    "AWS::CloudFront::OriginRequestPolicy",
    "AWS::EC2::DHCPOptions",
    "AWS::EC2::InternetGateway",
    "AWS::EC2::KeyPair",
    "AWS::EC2::NetworkAcl",
    "AWS::EC2::NetworkInsightsPath",
    "AWS::EC2::NetworkInterface",
    "AWS::EC2::PlacementGroup",
    "AWS::EC2::Route",
    "AWS::EC2::RouteTable",
    "AWS::EC2::SecurityGroup",
    "AWS::EC2::Subnet",
    "AWS::EC2::SubnetCidrBlock",
    "AWS::EC2::SubnetNetworkAclAssociation",
    "AWS::EC2::SubnetRouteTableAssociation",
    ...
  ],
  "ResourcesRead": 676
}
```

부분 스캔의 경우 다음과 비슷한 출력이 표시됩니다.

```
{
```

```

    "ResourceScanId": "arn:aws:cloudformation:region:account-id:resourceScan/0a699f15-489c-43ca-a3ef-3e6ecfa5da60",
    "Status": "COMPLETE",
    "StartTime": "2025-03-06T18:24:19.542000+00:00",
    "EndTime": "2025-03-06T18:25:23.142000+00:00",
    "PercentageCompleted": 100.0,
    "ResourceTypes": [
        "AWS::EC2::Instance",
        "AWS::EC2::SecurityGroup",
        "AWS::S3::Bucket",
        "AWS::S3::BucketPolicy"
    ],
    "ResourcesRead": 65,
    "ScanFilters": [
        {
            "Types": [
                "AWS::EC2::Instance",
                "AWS::EC2::SecurityGroup",
                "AWS::S3::*"
            ]
        }
    ]
}

```

출력의 필드에 대한 설명은 AWS CloudFormation API 참조의 [DescribeResourceScan](#)을 참조하세요.

CloudFormation 콘솔의 스캔 요약 보기

스캔 완료 후 스캔 중에 찾은 리소스를 시각화하여 볼 수 있어 다양한 제품 유형에 걸쳐 리소스가 얼마나 집중되어 있는지 식별하는 데 도움이 됩니다.

스캔 중에 찾은 리소스에 대한 정보를 보는 방법

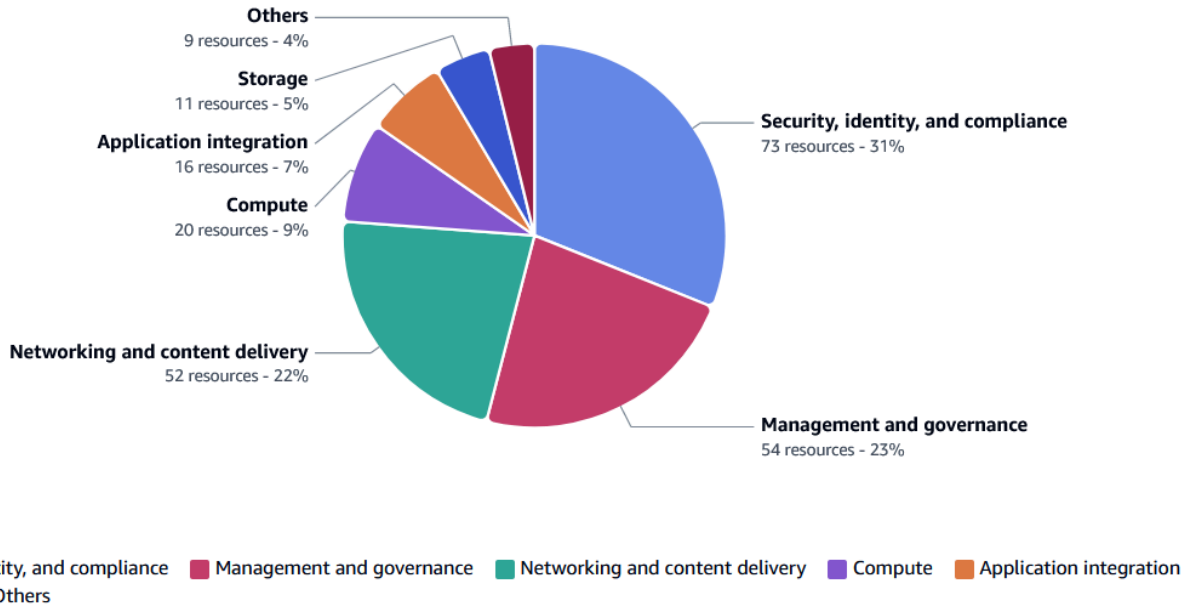
1. CloudFormation 콘솔의 [IaC 생성기 페이지](#)를 엽니다.
2. 화면 상단의 탐색 모음에서 보려는 리소스 스캔이 포함된 AWS 리전을 선택합니다.
3. 탐색 창에서 IaC 생성기를 선택합니다.
4. 스캔한 리소스의 시각적 분석을 스캔한 리소스 분석에서 찾을 수 있습니다(예: 컴퓨팅, 스토리지).
5. 표시되는 제품 유형 수를 사용자 지정하려면 표시된 데이터 필터링을 선택합니다. 그러면 가장 관심 있는 제품 유형에 초점을 맞춰 시각화를 조정하는 데 도움이 됩니다.
6. 페이지 오른쪽에 스캔 요약 세부 정보 패널이 있습니다. 패널을 열려면 패널 열기 아이콘을 선택합니다.

Scanned resources breakdown 235

View a visual breakdown of your scanned resources by product types. Select a product type for a further breakdown.

Filter displayed data

Filter by product type ▼



IaC 생성기로 스캔한 리소스로 CloudFormation 템플릿 생성

이 주제에서는 IaC 생성기 기능을 사용하여 스캔한 리소스에서 템플릿을 생성하는 방법을 설명합니다.

스캔한 리소스에서 템플릿 생성(콘솔)

스캔한 리소스에서 새 스택 템플릿 생성

1. CloudFormation 콘솔의 [IaC 생성기 페이지](#)를 엽니다.
2. 화면 상단의 탐색 모음에서 스캔한 리소스가 포함된 AWS 리전을 선택합니다.
3. 템플릿 섹션에서 템플릿 생성을 선택합니다.
4. 새 템플릿에서 시작을 선택합니다.
 - a. 템플릿 이름에 템플릿의 이름을 입력합니다.
 - b. (선택 사항) 삭제 정책 및 업데이트 교체 정책을 구성합니다.
 - c. 다음을 선택하여 스캔한 리소스를 템플릿에 추가합니다.

- 스캔한 리소스 추가에서 스캔한 리소스 목록을 탐색하고 템플릿에 추가할 리소스를 선택합니다. 리소스 식별자, 리소스 유형 또는 태그를 기준으로 리소스를 필터링할 수 있습니다. 필터는 상호 포괄적입니다.
- 템플릿에 필요한 리소스를 모두 추가했으면 다음을 선택하여 스캔한 리소스 추가 페이지를 종료하고 관련 리소스 추가 페이지로 진행합니다.
- 관련 리소스 권장 목록을 검토하세요. Amazon EC2 인스턴스와 보안 그룹 등의 관련 리소스는 상호 의존적이며 일반적으로 동일한 워크로드에 속합니다. 생성된 템플릿에 포함하려는 관련 리소스를 선택합니다.

Note

이 템플릿에 모든 관련 리소스를 추가하는 것이 좋습니다.

- 템플릿 세부 정보, 스캔한 리소스, 관련 리소스를 검토하세요.
- 템플릿 생성을 선택하여 검토 및 생성 페이지를 종료하고 템플릿을 생성합니다.

스캔한 리소스에서 템플릿 생성(AWS CLI)

스캔한 리소스에서 새 스택 템플릿 생성

- 스캔 중 찾은 리소스를 나열하려면 [list-resource-scan-resources](#) 명령을 사용하고 필요에 따라 출력을 제한하는 `--resource-identifier` 옵션을 지정합니다. `--resource-scan-id` 옵션의 경우 샘플 ARN을 실제 ARN으로 바꿉니다.

```
aws cloudformation list-resource-scan-resources \
  --resource-scan-id arn:aws:cloudformation:us-east-1:123456789012:resourceScan/0a699f15-489c-43ca-a3ef-3e6ecfa5da60 \
  --resource-identifier MyApp
```

다음은 응답 예제이며, 여기서 `ManagedByStack`은 CloudFormation이 리소스를 이미 관리하고 있는지 여부를 나타냅니다. 출력을 복사합니다. 이 정보는 다음 단계에 필요합니다.

```
{
  "Resources": [
    {
      "ResourceType": "AWS::EKS::Cluster",
      "ResourceIdentifier": {
        "ClusterName": "MyAppClusterName"
      }
    }
  ]
}
```

```

    },
    "ManagedByStack": false
  },
  {
    "ResourceType": "AWS::AutoScaling::AutoScalingGroup",
    "ResourceIdentifier": {
      "AutoScalingGroupName": "MyAppASGName"
    },
    "ManagedByStack": false
  }
]
}

```

출력의 필드에 대한 설명은 AWS CloudFormation API 참조의 [ScannedResource](#)를 참조하세요.

2. cat 명령을 사용하여 홈 디렉터리의 이름이 resources.json인 JSON 파일에 리소스 유형과 식별자를 저장합니다. 다음은 이전 단계의 예제 출력을 기반으로 하는 JSON 예제입니다.

```

$ cat > resources.json
[
  {
    "ResourceType": "AWS::EKS::Cluster",
    "ResourceIdentifier": {
      "ClusterName": "MyAppClusterName"
    }
  },
  {
    "ResourceType": "AWS::AutoScaling::AutoScalingGroup",
    "ResourceIdentifier": {
      "AutoScalingGroupName": "MyAppASGName"
    }
  }
]

```

3. 스캔한 리소스와 관련된 리소스를 나열하려면 [list-resource-scan-related-resources](#) 명령과 함께 생성한 resources.json 파일을 사용합니다.

```

aws cloudformation list-resource-scan-related-resources \
  --resource-scan-id arn:aws:cloudformation:us-
east-1:123456789012:resourceScan/0a699f15-489c-43ca-a3ef-3e6ecfa5da60 \
  --resources file://resources.json

```

다음은 응답 예제이며, 여기서 `ManagedByStack`은 CloudFormation이 리소스를 이미 관리하고 있는지 여부를 나타냅니다. 이전 단계에서 생성한 JSON 파일에 리소스를 추가합니다. 템플릿을 생성할 때 필요합니다.

```
{
  "RelatedResources": [
    {
      "ResourceType": "AWS::EKS::Nodegroup",
      "ResourceIdentifier": {
        "NodegroupName": "MyAppNodegroupName"
      },
      "ManagedByStack": false
    },
    {
      "ResourceType": "AWS::IAM::Role",
      "ResourceIdentifier": {
        "RoleId": "arn:aws::iam::account-id:role/MyAppIAMRole"
      },
      "ManagedByStack": false
    }
  ]
}
```

출력의 필드에 대한 설명은 AWS CloudFormation API 참조의 [ScannedResource](#)를 참조하세요.

Note

리소스 입력 목록의 길이는 100을 초과할 수 없습니다. 100개 이상의 리소스에 대한 관련 리소스를 나열하려면 100개씩 배치로 `list-resource-scan-related-resources` 명령을 실행하고 결과를 통합합니다.

목록의 중복된 리소스가 출력에 포함될 수 있다는 점에 유의하세요.

4. 새 스택 템플릿을 생성하려면 [create-generated-template](#) 명령을 다음과 같이 수정하여 사용합니다.

- 스캔한 리소스가 포함된 AWS 리전으로 `us-east-1`을 바꿉니다.
- 생성할 템플릿의 이름으로 `MyTemplate`을 바꿉니다.

```
aws cloudformation create-generated-template --region us-east-1 \
```



```
--generated-template-name MyTemplate \
--resources file://resources.json
```

다음은 예 resources.json 파일입니다.

```
[
  {
    "ResourceType": "AWS::EKS::Cluster",
    "LogicalResourceId": "MyCluster",
    "ResourceIdentifier": {
      "ClusterName": "MyAppClusterName"
    }
  },
  {
    "ResourceType": "AWS::AutoScaling::AutoScalingGroup",
    "LogicalResourceId": "MyASG",
    "ResourceIdentifier": {
      "AutoScalingGroupName": "MyAppASGName"
    }
  },
  {
    "ResourceType": "AWS::EKS::Nodegroup",
    "LogicalResourceId": "MyNodegroup",
    "ResourceIdentifier": {
      "NodegroupName": "MyAppNodegroupName"
    }
  },
  {
    "ResourceType": "AWS::IAM::Role",
    "LogicalResourceId": "MyRole",
    "ResourceIdentifier": {
      "RoleId": "arn:aws::iam::account-id:role/MyAppIAMRole"
    }
  }
]
```

이 명령이 제대로 실행되면 다음을 반환합니다.

```
{
  "Arn":
    "arn:aws:cloudformation:region:account-id:generatedtemplate/7fc8512c-d8cb-4e02-
    b266-d39c48344e48",
```

```
"Name": "MyTemplate"
}
```

스캔한 리소스에서 CloudFormation 스택 생성

템플릿을 생성한 후 스택을 생성하고 스캔한 리소스를 가져오기 전에 Infrastructure Composer를 사용하여 생성된 템플릿을 미리 볼 수 있습니다. 이를 통해 리소스가 포함된 전체 애플리케이션 아키텍처와 그 관계를 시각화할 수 있습니다. Infrastructure Composer에 대한 자세한 내용은 [Infrastructure Composer를 사용하여 시각적으로 템플릿 생성](#) 섹션을 참조하세요.

스택을 생성하고 스캔한 리소스를 가져오는 방법

1. CloudFormation 콘솔의 [IaC 생성기 페이지](#)를 엽니다.
2. 화면 상단의 탐색 모음에서 템플릿에 대한 AWS 리전을 선택합니다.
3. 템플릿 탭을 선택한 다음에 템플릿 이름을 선택하여 자세한 내용을 봅니다.
4. 템플릿 섹션 상단의 템플릿 정의 탭에서 기본 설정을 기반으로 템플릿을 YAML에서 JSON 구문으로 전환할 수 있습니다.
5. 템플릿의 세부 정보를 검토하여 모든 항목이 올바르게 설정되었는지 확인합니다. 템플릿을 더 쉽게 검토하고 이해할 수 있도록 Infrastructure Composer를 사용하여 기본 코드 보기에서 템플릿에 설명된 인프라의 그래픽 형태 보기로 전환할 수 있습니다. 이렇게 하려면 템플릿에서 템플릿 대신에 캔버스를 선택합니다.

캔버스 작업

- 템플릿 내 특정 리소스의 세부 정보에 집중하려면 카드를 두 번 클릭하여 리소스 속성 패널을 불러옵니다.
 - 캔버스에서 카드를 시각적으로 정렬하고 구성하려면 캔버스 왼쪽 상단에서 정렬을 선택합니다.
 - 캔버스를 확대 및 축소하려면 캔버스 오른쪽 아래에 있는 확대/축소 컨트롤을 사용합니다.
6. 콘솔의 특정 리소스를 보려면 템플릿 리소스 탭을 선택한 다음에 살펴보려는 리소스의 물리적 ID를 선택합니다. 그러면 해당 특정 리소스의 콘솔로 이동합니다. 템플릿 리소스 탭에서 템플릿 정의의 리소스를 추가, 제거 및 재동기화할 수도 있습니다.
 7. 템플릿 정의 탭에서 IaC 생성기는 쓰기 전용 속성을 포함하는 리소스에 대해 경고를 표시할 수 있습니다. 경고를 검토한 후 생성된 템플릿을 다운로드하고 필요에 따라 변경할 수 있습니다. 자세한 내용은 [쓰기 전용 속성 해결](#) 섹션을 참조하세요.
 8. 템플릿 정의에 만족하면 템플릿 정의 탭에서 스택으로 가져오기와 다음을 차례로 선택합니다.

9. 스택 세부 정보 지정 페이지의 스택 지정 패널에서 스택 이름을 입력하고 다음을 선택합니다.
10. 스택에 대한 파라미터를 검토하고 입력합니다. 다음을 선택합니다.
11. 변경 사항 검토 페이지에서 옵션을 검토하고 다음을 선택합니다.
12. 검토 및 가져오기 페이지에서 세부 정보를 검토하고 리소스 가져오기를 선택합니다.

쓰기 전용 속성 해결

CloudFormation IaC 생성기를 사용하면 계정에 프로비저닝된 리소스 중 CloudFormation에서 아직 관리하지 않는 리소스를 사용하여 템플릿을 생성할 수 있습니다. 하지만 특정 리소스 속성은 쓰기 전용으로 지정되므로 데이터베이스 암호와 같이 CloudFormation에서 쓸 수 있지만 읽을 수는 없습니다.

기존 리소스에서 CloudFormation 템플릿을 생성할 때 쓰기 전용 속성은 문제를 야기합니다. 대부분의 경우 CloudFormation은 이러한 속성을 생성된 템플릿의 파라미터로 변환합니다. 이렇게 하면 가져오기 작업 중에 속성을 파라미터값으로 입력할 수 있습니다. 하지만 이러한 변환이 불가능한 시나리오도 있으며, CloudFormation은 이러한 경우를 다르게 처리합니다.

상호 배타적 속성

몇몇 리소스에는 상호 배타적인 속성 세트가 여러 개 있으며, 그중 일부는 쓰기 전용입니다. 이러한 경우 IaC 생성기는 생성 도중 리소스에 어떤 배타적인 속성 세트가 적용되었는지 확인할 수 없습니다. 예를 들어 [AWS::Lambda::Function](#)에 대한 코드를

- Code/S3Bucket, Code/S3Key, 선택적으로 Code/S3ObjectVersion 속성 세트를 사용하여 제공할 수 있습니다.
- Code/ImageUri
- Code/ZipFile

해당 속성은 모두 쓰기 전용입니다. IaC 생성기는 독점적인 속성 세트 중에서 하나를 선택하여 생성된 템플릿에 추가합니다. 각 쓰기 전용 속성에 대해 파라미터가 추가됩니다. 파라미터 이름에는 OneOf가 포함되며 파라미터 설명에 따르면 해당 속성을 다른 독점적인 속성으로 바꿀 수 있습니다. IaC 생성기는 포함된 속성에 대해 MUTUALLY_EXCLUSIVE_PROPERTIES의 경고 유형을 설정합니다.

상호 배타적 유형

경우에 따라 쓰기 전용 속성이 여러 데이터 유형일 수 있습니다. 예를 들어 [AWS::ApiGateway::RestApi](#)의 Body 속성은 object 또는 string가 될 수 있습니다.

이러한 경우 IaC 생성기는 string 유형을 사용하여 생성된 템플릿에 속성을 포함하고 MUTUALLY_EXCLUSIVE_TYPES의 경고 유형을 설정합니다.

Array 속성

쓰기 전용 속성의 유형이 array인 경우 파라미터는 스칼라 값만 사용할 수 있으므로 IaC 생성기는 생성된 템플릿에 해당 속성을 포함할 수 없습니다. 이 경우 템플릿에서 해당 속성이 생략되고 경고 유형이 UNSUPPORTED_PROPERTIES로 설정됩니다.

선택적 속성

선택적 쓰기 전용 속성의 경우 IaC 생성기는 리소스를 설정할 때 속성이 사용되었는지 감지할 수 없습니다. 이 경우 생성된 템플릿에서 해당 속성이 생략되고 경고 유형이 UNSUPPORTED_PROPERTIES로 설정됩니다.

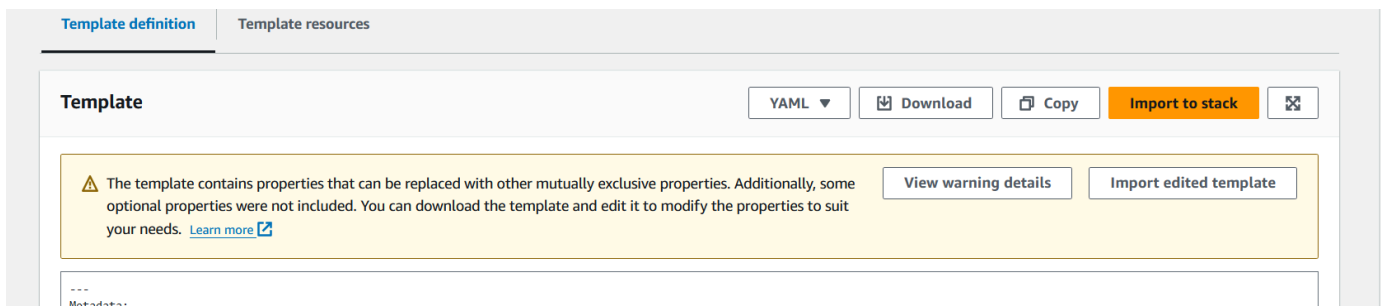
경고 및 다음 단계

어떤 속성이 쓰기 전용인지 판단하려면 IaC 생성기 콘솔에서 반환되는 경고를 살펴보아야 합니다. [AWS 리소스 및 속성 유형 참조](#)는 속성이 쓰기 전용인지 또는 여러 유형을 지원하는지를 나타내지 않습니다.

또는 리소스 공급자 스키마에서 어떤 속성이 쓰기 전용인지 확인할 수 있습니다. 리소스 공급자 스키마를 다운로드하려면 [CloudFormation 리소스 공급자 스키마](#)를 참조하세요.

쓰기 전용 속성 문제를 해결하는 방법

1. CloudFormation 콘솔의 [IaC 생성기 페이지](#)를 엽니다.
2. 화면 상단의 탐색 모음에서 템플릿에 대한 AWS 리전을 선택합니다.
3. 템플릿 탭을 선택한 다음에 생성한 템플릿의 이름을 선택합니다.
4. 템플릿 정의 탭에서는 생성된 템플릿에 쓰기 전용 속성이 있는 리소스가 포함되어 있는 경우 IaC 생성기 콘솔에 문제 유형에 대한 요약과 함께 경고가 표시됩니다. 예:



5. 자세히 알아보려면 경고 세부 정보 보기를 선택합니다. 쓰기 전용 속성이 있는 리소스는 생성된 템플릿과 리소스 유형에 사용된 논리적 ID에 따라 식별됩니다.

경고 목록을 사용하여 쓰기 전용 속성이 있는 리소스를 식별하고 생성된 템플릿에 변경이 필요한 경우 어떤 변경이 필요한지 판단하기 위해 각 리소스를 살펴봅니다.

Resource warnings (1) ×

▼ LambdaFunction001aClambda002fD6K (AWS::Lambda::Function) [Learn more](#)

Unsupported properties	Mutually exclusive properties
1. SnapStart/ApplyOn Applying SnapStart setting on function resource type. Possible values: [PublishedVersions, None] This property can be replaced with other types	1. Code/S3Bucket An Amazon S3 bucket in the same AWS Region as your function. The bucket can be in a different AWS account. This property can be replaced with other exclusive properties
2. Code/S3ObjectVersion For versioned objects, the version of the deployment package object to use. This property can be replaced with other exclusive properties	2. Code/S3Key The Amazon S3 key of the deployment package. This property can be replaced with other exclusive properties

6. 템플릿을 업데이트하여 쓰기 전용 속성 문제를 해결해야 한다면 다음과 같은 단계를 수행합니다.
- 다운로드를 선택하여 템플릿 사본을 다운로드합니다.
 - 템플릿을 편집합니다.
 - 변경이 완료되면 편집한 템플릿 가져오기 버튼을 선택하여 가져오기 프로세스를 계속할 수 있습니다.

AWS::ApiGateway::RestAPI 리소스의 쓰기 전용 속성 문제를 해결하는 방법

이 주제에서는 IaC 생성기를 사용할 때 [AWS::ApiGateway::RestApi](#) 리소스의 쓰기 전용 속성 문제를 해결하는 방법을 설명합니다.

문제

생성된 템플릿에 `AWS::ApiGateway::RestApi` 리소스가 포함되어 있으면 `Body`, `BodyS3Location`, `CloneFrom` 속성이 `UNSUPPORTED_PROPERTIES`로 식별된다는 경고가 생성됩니다. 이는 해당 속성이 선택적인 쓰기 전용 속성이기 때문입니다. IaC 생성기는 해당 속성이 리소스에 적용되었는지 여부를 알지 못합니다. 따라서 생성된 템플릿에서 이러한 속성을 생략합니다.

해결 방법

REST API의 `Body` 속성을 설정하려면 생성된 템플릿을 업데이트하세요.

1. Amazon API Gateway [GetExport](#) API 작업을 사용하여 API를 다운로드합니다. 예를 들어 [aws apigateway get-export](#) AWS CLI 명령을 사용합니다. 자세한 내용을 알아보려면 API Gateway 개발자 안내서의 [API Gateway에서 REST API 내보내기](#)를 참조하세요.
2. GetExport API 작업의 응답에서 Body 속성을 검색합니다. Amazon S3 버킷에 업로드합니다.
3. 생성된 템플릿을 다운로드합니다.
4. 템플릿에 BodyS3Location/Bucket 및 BodyS3Location/Key 속성을 추가하고 Body가 저장되는 버킷 이름과 키를 지정합니다.
5. IaC 생성기 콘솔에서 생성된 템플릿을 열고 편집한 템플릿 가져오기를 선택합니다.

AWS::Lambda::Function 리소스의 쓰기 전용 속성 문제를 해결하는 방법

이 주제에서는 IaC 생성기를 사용할 때 [AWS::Lambda::Function](#) 리소스의 쓰기 전용 속성 문제를 해결하는 방법을 설명합니다.

문제

AWS::Lambda::Function 리소스에는 Lambda 코드를 지정하기 위한 상호 배타적인 속성 세트가 3개 있습니다.

- Code/S3Bucket 및 Code/S3Key 속성과 선택적으로 Code/S3ObjectVersion 속성 중 하나만 있을 수 있습니다.
- Code/ImageUri 속성
- Code/ZipFile 속성

지정된 AWS::Lambda::Function 리소스에는 이러한 세트 중 하나만 사용할 수 있습니다.

IaC 생성기는 리소스를 생성하거나 업데이트하는 데 사용된 독점적인 쓰기 전용 속성 세트를 확인할 수 없습니다. 결과적으로 생성된 템플릿에 첫 번째 속성 세트만 포함합니다. Code/ImageUri 및 Code/ZipFile 속성은 생략됩니다.

또한 IaC 생성기는 다음과 같은 경고를 표시합니다.

- **MUTUALLY_EXCLUSIVE_PROPERTIES** - Code/S3Bucket과 Code/S3Key가 상호 배타적 속성으로 식별된다고 경고합니다.
- **UNSUPPORTED_PROPERTIES** - Code/S3ObjectVersion 속성이 지원되지 않는다고 경고합니다.

생성된 템플릿에 `AWS::Lambda::Function` 리소스를 포함하려면 템플릿을 다운로드하고 올바른 코드 속성으로 업데이트해야 합니다.

해결 방법

Amazon S3 버킷에 Lambda 코드를 저장하고 **S3ObjectVersion** 속성을 사용하지 않는 경우에는 생성된 템플릿을 수정하지 않고 가져올 수 있습니다. IaC 생성기는 가져오기 작업 중에 템플릿 파라미터로 Amazon S3 버킷과 키를 요청합니다.

Lambda 코드를 Amazon ECR 리포지토리로 저장하는 경우에는 다음 지침에 따라 템플릿을 업데이트할 수 있습니다.

1. 생성된 템플릿을 다운로드합니다.
2. 생성된 템플릿에서 Code/S3Bucket 및 Code/S3Key 속성에 대한 속성 및 해당 파라미터를 제거합니다.
3. 생성된 템플릿에서 제거된 속성을 Code/ImageUri 속성으로 바꾸고 Amazon ECR 리포지토리의 URL을 지정합니다.
4. IaC 생성기 콘솔에서 생성된 템플릿을 열고 편집한 템플릿 가져오기 버튼을 선택합니다.

Lambda 코드를 zip 파일로 저장하는 경우에는 다음 지침에 따라 템플릿을 업데이트할 수 있습니다.

1. 생성된 템플릿을 다운로드합니다.
2. 생성된 템플릿에서 Code/S3Bucket 및 Code/S3Key 속성에 대한 속성 및 해당 파라미터를 제거합니다.
3. 생성된 템플릿에서 제거된 속성을 Code/ZipFile 속성으로 바꿉니다.
4. IaC 생성기 콘솔에서 생성된 템플릿을 열고 편집한 템플릿 가져오기 버튼을 선택합니다.

Lambda 코드 사본이 없는 경우에는 다음 지침에 따라 템플릿을 업데이트할 수 있습니다.

1. AWS Lambda [GetFunction](#) API 작업을 사용하세요(예: [aws lambda get-function](#) AWS CLI 명령 사용).
2. 응답에서 RepositoryType 파라미터는 코드가 Amazon S3 버킷에 있는 경우 S3, 코드가 Amazon ECR 리포지토리에 있는 경우에는 ECR입니다.
3. 응답에서 Location 파라미터에는 배포 패키지를 10분 동안 다운로드하는 데 사용할 수 있는 미리 서명된 URL이 포함되어 있습니다. 코드를 다운로드하십시오.
4. Amazon S3 버킷에 코드를 업로드합니다.

5. 생성된 템플릿으로 가져오기 작업을 실행하고 버킷 이름과 키를 파라미터 값으로 제공합니다.

동적 참조를 사용하여 다른 서비스에 저장된 값 가져오기

동적 참조는 다른 서비스에서 저장되고 관리되는 외부 값을 지정하고 코드형 인프라 템플릿에서 민감한 정보를 분리하는 편리한 방법을 제공합니다. CloudFormation에서 스택 및 변경 세트 작업 도중 필요한 지정된 참조의 값을 검색합니다.

동적 참조를 사용하여 다음을 수행할 수 있습니다.

- 보안 문자열 사용 - 민감한 데이터의 경우 데이터가 저장 시 암호화되도록 항상 AWS Systems Manager Parameter Store의 보안 문자열 파라미터 또는 AWS Secrets Manager의 보안 암호를 사용하세요.
- 액세스 제한 - Parameter Store 파라미터 또는 Secrets Manager 보안 암호에 대한 액세스를 승인된 보안 주체 및 역할로만 제한합니다.
- 자격 증명 교체 - Parameter Store 또는 Secrets Manager에 저장된 민감한 데이터를 정기적으로 교체하여 높은 수준의 보안을 유지하세요.
- 자동 교체 - Secrets Manager의 자동 교체 기능을 활용하여 민감한 데이터를 정기적으로 업데이트하고 애플리케이션 및 환경 전체에 배포하세요.

일반적인 고려 사항

다음은 CloudFormation 템플릿에서 동적 참조를 지정하기 전에 고려할 일반적인 사항들입니다.

- 리소스의 기본 식별자의 일부인 리소스 속성에 동적 참조 또는 민감한 데이터를 포함하지 마세요. CloudFormation은 기본 리소스 식별자의 실제 일반 텍스트 값을 사용할 수 있으며, 이는 보안 위험이 될 수 있습니다. 이 리소스 ID는 파생된 출력 또는 대상에 나타날 수 있습니다.

리소스 유형의 기본 식별자를 구성하는 리소스 속성을 확인하려면 [AWS 리소스 및 속성 유형 참조](#)에서 해당 리소스에 대한 리소스 참조 설명서를 참조하세요. Return values(반환 값) 섹션에서 Ref 함수 반환 값은 리소스 유형의 기본 식별자를 구성하는 리소스 속성을 나타냅니다.

- 스택 템플릿에 최대 60개의 동적 참조를 포함할 수 있습니다.
- 변환(예: AWS::Include 또는 AWS::Serverless)을 사용하는 경우 CloudFormation은 변환을 적용하기 전에 동적 참조를 확인하지 않습니다. 대신 동적 참조의 리터럴 문자열을 변형으로 전달하고, 템플릿을 사용하여 변경 세트를 실행할 때 참조를 확인합니다.

- 사용자 지정 리소스의 보안 값(예: Parameter Store 또는 Secrets Manager에 저장된 값)에서는 동적 참조를 사용할 수 없습니다.
- 동적 참조는 `AWS::CloudFormation::Init` 메타데이터 및 Amazon EC2 UserData 속성에서도 지원되지 않습니다.
- 백슬래시(\)로 끝나는 동적 참조를 생성하지 마세요. CloudFormation은 이러한 참조를 확인할 수 없으므로 스택 작업이 실패합니다.

다음 주제에서는 동적 참조 사용에 대한 정보 및 기타 고려 사항을 제공합니다.

주제

- [Systems Manager Parameter Store에서 일반 텍스트 값 가져오기](#)
- [Systems Manager Parameter Store에서 보안 문자열 값 가져오기](#)
- [Secrets Manager에서 보안 암호 또는 보안 암호 값 가져오기](#)

Systems Manager Parameter Store에서 일반 텍스트 값 가져오기

CloudFormation 템플릿을 생성할 때 Parameter Store에 저장된 일반 텍스트 값을 사용하는 것이 좋습니다. Parameter Store는 AWS Systems Manager의 기능입니다. Parameter Store 소개는 AWS Systems Manager 사용 설명서의 [AWS Systems Manager Parameter Store](#)를 참조하세요.

템플릿 내에서 Parameter Store의 일반 텍스트 값을 사용하려면 ssm 동적 참조를 사용합니다. 이 참조를 사용하면 Parameter Store의 String 또는 StringList 유형의 파라미터에서 값을 액세스할 수 있습니다.

스택 작업에 사용할 ssm 동적 참조의 버전을 확인하려면 스택 작업에 대한 변경 세트를 생성합니다. 그런 다음 템플릿 탭에서 처리된 템플릿을 검토합니다. 자세한 내용은 [CloudFormation 스택에 대한 변경 세트 생성](#) 섹션을 참조하세요.

ssm 동적 참조를 사용할 때는 다음과 같은 몇 가지를 염두에 두어야 합니다.

- CloudFormation은 동적 참조에 대한 드리프트 감지를 지원하지 않습니다. 파라미터 버전을 지정하지 않은 ssm 동적 참조에 대해 Systems Manager에서 파라미터 버전을 업데이트하는 경우 ssm 동적 참조가 포함된 모든 스택에서 스택 업데이트 작업을 수행하여 최신 파라미터 버전을 가져오는 것이 좋습니다.
- CloudFormation 템플릿의 Parameters 섹션에서 ssm 동적 참조를 사용하려면 버전 번호를 포함해야 합니다. CloudFormation에서는 이 섹션에 버전 번호가 없으면 Parameter Store 값을 참조할

수 없습니다. 템플릿에서 파라미터를 Systems Manager 파라미터 유형으로 정의할 수도 있습니다. 이렇게 하면 Systems Manager 파라미터 키를 파라미터의 기본값으로 지정할 수 있습니다. 그러면 버전 번호를 지정하지 않아도 CloudFormation이 Parameter Store에서 최신 버전의 파라미터 값을 검색합니다. 이렇게 하면 템플릿을 더 간단하고 쉽게 유지할 수 있습니다. 자세한 내용은 [CloudFormation에서 제공하는 파라미터 유형을 사용하여 런타임 시 기존 리소스 지정](#) 섹션을 참조하세요.

- 사용자 지정 리소스의 경우 CloudFormation에서 사용자 지정 리소스로 요청을 전송하기 전에 ssm 동적 참조를 확인합니다.
- CloudFormation은 동적 참조를 사용하여 다른 AWS 계정에서 공유된 파라미터를 참조하는 것을 지원하지 않습니다.
- CloudFormation에서 동적 참조 내에서 Systems Manager 파라미터 레이블 사용을 지원하지 않습니다.

권한

시스템 관리자 Parameter Store에 저장된 파라미터를 지정하려면 지정된 파라미터에 대한 [GetParameters](#)를 호출할 수 있는 권한이 있어야 합니다. 특정 Systems Manager 파라미터에 대한 액세스를 제공하는 IAM 정책을 생성하는 방법을 알아보려면 AWS Systems Manager 사용 설명서의 [IAM 정책을 사용하여 Systems Manager 파라미터에 대한 액세스 제한](#)을 참조하세요.

참조 패턴

CloudFormation 템플릿의 Systems Manager Parameter Store에 저장된 일반 텍스트 값을 참조하려면 다음 ssm 참조 패턴을 사용합니다.

```
{{resolve:ssm:parameter-name:version}}
```

참조는 parameter-name 및 version에 대해 다음 정규 표현식 패턴을 준수해야 합니다.

```
{{resolve:ssm:[a-zA-Z0-9_.\-/+](?:\d+)??}}
```

parameter-name

Parameter Store의 파라미터 이름입니다. 파라미터 이름은 대/소문자를 구분합니다.

필수 사항입니다.

version

사용할 파라미터의 버전을 지정하는 정수입니다. 정확한 버전을 지정하지 않으면 CloudFormation 은 스택을 생성하거나 업데이트할 때마다 최신 버전의 파라미터를 사용합니다. 자세한 내용을 알아보려면 AWS Systems Manager 사용 설명서의 [파라미터 버전 사용](#)을 참조하세요.

선택 사항.

예제

주제

- [퍼블릭 AMI ID 파라미터](#)
- [사용자 지정 AMI ID 파라미터](#)

퍼블릭 AMI ID 파라미터

다음 예제에서는 퍼블릭 AMI 파라미터를 참조하는 EC2 인스턴스를 생성합니다. 동적 참조는 퍼블릭 파라미터에서 최신 Amazon Linux 2023 AMI ID를 검색합니다. 퍼블릭 파라미터에 대한 자세한 내용은 AWS Systems Manager 사용 설명서의 [Parameter Store에서 퍼블릭 파라미터 검색](#)을 참조하세요.

JSON

```
{
  "Resources": {
    "MyInstance": {
      "Type": "AWS::EC2::Instance",
      "Properties": {
        "ImageId": "{{resolve:ssm:/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-6.1-x86_64}}",
        "InstanceType": "t2.micro"
      }
    }
  }
}
```

YAML

```
Resources:
  MyInstance:
```

```
Type: AWS::EC2::Instance
Properties:
  ImageId: '{{resolve:ssm:/aws/service/ami-amazon-linux-latest/al2023-ami-
kernel-6.1-x86_64}}'
  InstanceType: t2.micro
```

사용자 지정 AMI ID 파라미터

다음 예제는 Parameter Store에 저장된 사용자 지정 AMI ID를 참조하는 EC2 시작 템플릿을 생성합니다. 동적 참조는 인스턴스가 시작 템플릿에서 시작될 때마다 2 버전의 *golden-ami* 파라미터에서 AMI 검색합니다.

JSON

```
{
  "Resources": {
    "MyLaunchTemplate": {
      "Type": "AWS::EC2::LaunchTemplate",
      "Properties": {
        "LaunchTemplateName": {
          "Fn::Sub": "${AWS::StackName}-launch-template"
        },
        "LaunchTemplateData": {
          "ImageId": "{{resolve:ssm:golden-ami:2}}",
          "InstanceType": "t2.micro"
        }
      }
    }
  }
}
```

YAML

```
Resources:
  MyLaunchTemplate:
    Type: AWS::EC2::LaunchTemplate
    Properties:
      LaunchTemplateName: !Sub ${AWS::StackName}-launch-template
      LaunchTemplateData:
        ImageId: '{{resolve:ssm:golden-ami:2}}'
        InstanceType: t2.micro
```

Systems Manager Parameter Store에서 보안 문자열 값 가져오기

CloudFormation에서 민감한 데이터를 AWS Systems Manager Parameter Store에 “보안 문자열”로 저장하여 템플릿에 직접 노출하지 않고도 암호 또는 라이선스 키와 같은 민감한 데이터를 사용할 수 있습니다. Parameter Store 소개는 AWS Systems Manager 사용 설명서의 [AWS Systems Manager Parameter Store](#)를 참조하세요.

템플릿 내에서 Parameter Store 보안 문자열을 참조하려면 `ssm-secure` 동적 참조를 사용합니다. CloudFormation은 실제 보안 문자열 값을 저장하지 않습니다. 대신 보안 문자열의 일반 텍스트 파라미터가 포함된 리터럴 동적 참조를 저장합니다.

스택 생성 또는 업데이트 도중 CloudFormation은 실제 값을 노출하지 않고 필요에 따라 보안 문자열 값에 액세스합니다. 보안 문자열은 `ssm-secure` 동적 참조 패턴을 지원하는 리소스 속성에 대해서만 사용할 수 있습니다. 자세한 내용은 [보안 문자열에 대한 동적 파라미터 패턴을 지원하는 리소스](#) 섹션을 참조하세요.

CloudFormation에서는 API 호출의 보안 문자열에 대한 실제 파라미터 값을 반환하지 않습니다. 리터럴 동적 참조만 반환합니다. 변경 세트를 사용하여 변경을 비교할 때 CloudFormation에서는 리터럴 동적 참조 문자열만 비교합니다. 실제 보안 문자열 값을 확인하고 비교하지는 않습니다.

`ssm-secure` 동적 참조를 사용할 때는 다음과 같은 몇 가지를 염두에 두어야 합니다.

- CloudFormation은 다른 AWS 계정의 Parameter Store 값에 액세스할 수 없습니다.
- CloudFormation에서 동적 참조 내에서 Systems Manager 파라미터 레이블 또는 퍼블릭 파라미터 사용을 지원하지 않습니다.
- `cn-north-1` 및 `cn-northwest-1` 리전에서 보안 문자열은 Systems Manager에서 지원되지 않습니다.
- `ssm-secure`와 같은 보안 암호 값에 대한 동적 참조는 현재 사용자 지정 리소스에서 지원되지 않습니다.
- CloudFormation에서 스택 업데이트를 롤백해야 하고, 이전에 지정된 버전의 보안 문자열 파라미터를 더 이상 사용할 수 없는 경우 롤백 작업은 실패합니다. 이 경우 다음 두 가지 선택지가 있습니다.
 - `CONTINUE_UPDATE_ROLLBACK`을 사용하여 리소스를 건너뛰니다.
 - 시스템 관리자 Parameter Store의 보안 문자열 파라미터를 재생성하고, 파라미터 버전이 템플릿에서 사용되는 버전에 도달할 때까지 업데이트합니다. 그런 다음 리소스를 건너뛰지 않고 `CONTINUE_UPDATE_ROLLBACK`을 사용합니다.

보안 문자열에 대한 동적 파라미터 패턴을 지원하는 리소스

ssm-secure 동적 참조 패턴을 지원하는 리소스는 다음과 같습니다.

리소스	속성 유형	속성
AWS::DirectoryService::MicrosoftAD		Password
AWS::DirectoryService::SimpleAD		Password
AWS::ElastiCache::ReplicationGroup		AuthToken
AWS::IAM::User	LoginProfile	Password
AWS::KinesisFirehose::DeliveryStream	RedshiftDestinationConfiguration	Password
AWS::OpsWorks::App	소스	Password
AWS::OpsWorks::Stack	CustomCookbooksSource	Password
AWS::OpsWorks::Stack	RdsDbInstances	DbPassword
AWS::RDS::DBCluster		MasterUserPassword
AWS::RDS::DBInstance		MasterUserPassword
AWS::Redshift::Cluster		MasterUserPassword

참조 패턴

CloudFormation 템플릿에서 Systems Manager Parameter Store에서 보안 문자열 값을 참조하려면 다음 ssm-secure 동적 패턴을 사용합니다.

```
{{resolve:ssm-secure:parameter-name:version}}
```

참조는 parameter-name 및 version에 대해 다음 정규 표현식 패턴을 준수해야 합니다.

```
{{resolve:ssm-secure:[a-zA-Z0-9_.\-/]+(:\d+)?}}
```

parameter-name

Parameter Store의 파라미터 이름입니다. 파라미터 이름은 대/소문자를 구분합니다.

필수 사항입니다.

version

사용할 파라미터의 버전을 지정하는 정수입니다. 정확한 버전을 지정하지 않으면 CloudFormation은 스택을 생성하거나 업데이트할 때마다 최신 버전의 파라미터를 사용합니다. 자세한 내용을 알아보려면 AWS Systems Manager 사용 설명서의 [파라미터 버전 사용](#)을 참조하세요.

선택 사항.

예제

다음 예제에서는 ssm-secure 동적 참조를 사용하여 Parameter Store에 저장된 보안 문자열에 대한 IAM 사용자의 암호를 설정합니다. 지정된 것과 같이 CloudFormation은 스택 및 변경 세트 작업에 대해 **10** 버전의 *IAMUserPassword* 파라미터를 사용합니다.

JSON

```
{
  "MyIAMUser": {
    "Type": "AWS::IAM::User",
    "Properties": {
      "UserName": "MyUserName",
      "LoginProfile": {
        "Password": "{{resolve:ssm-secure:IAMUserPassword:10}}"
      }
    }
  }
}
```

YAML

```
MyIAMUser:
  Type: AWS::IAM::User
  Properties:
    UserName: 'MyUserName'
```

```
LoginProfile:
  Password: '{{resolve:ssm-secure:IAMUserPassword:10}}'
```

Secrets Manager에서 보안 암호 또는 보안 암호 값 가져오기

Secrets Manager는 데이터베이스 자격 증명, 암호, 타사 API 키와 같은 보안 암호를 안전하게 저장 및 관리하는 서비스입니다. Secrets Manager를 사용하면 이러한 보안 암호에 대한 액세스를 중앙에서 저장하고 제어할 수 있으므로 코드의 하드코딩된 자격 증명(암호 포함)을 Secrets Manager에 대한 API 호출로 대체하여 프로그래밍 방식으로 보안 암호를 검색할 수 있습니다. 자세한 내용은 AWS Secrets Manager 사용 설명서의 [AWS Secrets Manager이란?](#)을 참조하세요.

CloudFormation 템플릿 내에서 Secrets Manager에 저장된 전체 보안 암호 또는 보안 암호 값을 사용하려면 `secretsmanager` 동적 참조를 사용합니다.

모범 사례

CloudFormation 템플릿에서 Secrets Manager 동적 참조를 사용할 때는 다음 모범 사례를 따르세요.

- CloudFormation 템플릿에 버전 없는 참조 사용 - 자격 증명을 Secrets Manager에 저장하고 적절한 보안 암호 교체 워크플로를 지원하기 위해 `version-stage` 또는 `version-id` 파라미터를 지정하지 않고 동적 참조를 사용합니다.
- 자동 교체 활용 - 자격 증명 관리를 위해 버전 없는 동적 참조와 함께 Secrets Manager의 자동 교체 기능을 사용합니다. 이렇게 하면 템플릿 변경 없이 자격 증명을 정기적으로 업데이트할 수 있습니다. 자세한 내용을 알아보려면 [AWS Secrets Manager 보안 암호 교체](#)를 참조하세요.
- 버전이 지정된 참조를 거의 사용 - 테스트 또는 롤백 상황과 같은 특정 시나리오에 대해 명시적 `version-stage` 또는 `version-id` 파라미터만 지정합니다.

고려 사항

`secretsmanager` 동적 참조를 사용할 때는 다음과 같은 중요한 보안 고려 사항을 염두에 두어야 합니다.

- CloudFormation은 이전 배포에서 사용된 보안 암호 버전을 추적하지 않습니다. 동적 참조를 구현하기 전에 보안 암호 관리 전략을 신중하게 계획합니다. 가능하면 버전 없는 참조를 사용하여 자동 보안 암호 교체를 활용합니다. 버전 관리되지 않은 동적 참조에서 버전 관리된 동적 참조로 전환하거나 그 반대로 전환하는 경우와 같이 동적 참조 구성을 변경할 때 리소스 업데이트를 모니터링하고 검증합니다.

- Secrets Manager에서 보안 암호 값만 업데이트해도 CloudFormation에서 새 값을 자동으로 검색하지는 않습니다. CloudFormation은 리소스 생성 또는 동적 참조가 포함된 리소스를 수정하는 업데이트 중에만 보안 암호 값을 검색합니다.

예를 들어 템플릿에 MasterPassword 속성이 Secrets Manager 동적 참조로 설정된 [AWS::RDS::DBInstance](#) 리소스가 포함되어 있다고 가정합니다. 이 템플릿에서 스택을 생성한 후 Secrets Manager에서 보안 암호 값을 업데이트합니다. 그러나 MasterPassword 속성은 이전 암호 값을 유지합니다.

새 보안 암호 값을 적용하려면 CloudFormation 템플릿에서 `AWS::RDS::DBInstance` 리소스를 수정하고 스택 업데이트를 수행해야 합니다.

향후 이 수동 프로세스를 거치지 않으려면 Secrets Manager를 사용하여 보안 암호를 자동으로 교체하는 것이 좋습니다.

- `secretsmanager`와 같은 보안 암호 값에 대한 동적 참조는 현재 사용자 지정 리소스에서 지원되지 않습니다.
- `secretsmanager` 동적 참조는 모든 리소스 속성에서 사용될 수 있습니다. `secretsmanager` 동적 참조 사용은 Secrets Manager 또는 CloudFormation 로그에서 확인된 보안 암호 값을 보존하지 않아야 함을 나타냅니다. 하지만 보안 암호 값이 리소스가 사용 중인 서비스에 표시될 수 있습니다. 사용을 검토하여 보안 암호 데이터 유출을 피합니다.

권한

Secrets Manager에 저장된 보안 암호를 지정하려면 보안 암호에 대한 [GetSecretValue](#)를 호출할 수 있는 권한이 있어야 합니다.

참조 패턴

CloudFormation 템플릿에서 Secrets Manager 보안 암호를 참조하려면 다음과 같은 `secretsmanager` 동적 참조 패턴을 사용합니다.

```
{{resolve:secretsmanager:secret-id:secret-string:json-key:version-stage:version-id}}
```

secret-id

보안 암호의 이름 또는 ARN입니다.

AWS 계정의 보안 암호에 액세스하려면 보안 암호 이름만 지정하면 됩니다. 다른 AWS 계정의 보안 암호에 액세스하려면 보안 암호의 전체 ARN을 지정해야 합니다.

필수 사항입니다.

secret-string

지원되는 유일한 값은 SecretString입니다. 기본값은 SecretString입니다.

json-key

검색하고자 하는 값을 보유한 키-값 페어의 키 이름입니다. json-key를 지정하지 않는 경우 CloudFormation에서 전체 암호 텍스트를 검색합니다.

이 세그먼트에는 콜론 문자(:)가 포함되지 않을 수 있습니다.

version-stage

사용할 보안 암호 버전의 스테이징 레이블입니다. Secrets Manager는 교체 프로세스 도중 다른 버전을 추적하는 데 스테이징 레이블을 사용합니다. version-stage를 사용하는 경우 version-id를 지정하지 마세요. version-stage 또는 version-id를 지정하지 않은 경우 기본값은 AWSCURRENT 버전입니다.

이 세그먼트에는 콜론 문자(:)가 포함되지 않을 수 있습니다.

version-id

사용하고자 하는 보안 암호의 버전에 대한 고유 식별자입니다. version-id를 지정할 경우 version-stage를 지정하지 마세요. version-stage 또는 version-id를 지정하지 않은 경우 기본값은 AWSCURRENT 버전입니다.

이 세그먼트에는 콜론 문자(:)가 포함되지 않을 수 있습니다.

예제

주제

- [보안 암호에서 사용자 이름 및 암호 값 검색](#)
- [전체 SecretString 검색](#)
- [특정 버전의 보안 암호에서 값 검색](#)
- [다른 AWS 계정에서 보안 암호 검색](#)

보안 암호에서 사용자 이름 및 암호 값 검색

다음 [AWS::RDS::DBInstance](#) 예제에서는 *MySecret* 보안 암호에 저장된 사용자 이름 및 암호 값을 검색합니다. 이 예제는 버전 없는 동적 참조에 권장되는 패턴을 보여줍니다. 이 패턴은 템플릿 변경 없이 자동으로 AWSCURRENT 버전을 사용하고 Secrets Manager 회전 워크플로를 지원합니다.

JSON

```
{
  "MyRDSInstance": {
    "Type": "AWS::RDS::DBInstance",
    "Properties": {
      "DBName": "MyRDSInstance",
      "AllocatedStorage": "20",
      "DBInstanceClass": "db.t2.micro",
      "Engine": "mysql",
      "MasterUsername":
        "{{resolve:secretsmanager:MySecret:SecretString:username}}",
      "MasterUserPassword":
        "{{resolve:secretsmanager:MySecret:SecretString:password}}"
    }
  }
}
```

YAML

```
MyRDSInstance:
  Type: AWS::RDS::DBInstance
  Properties:
    DBName: MyRDSInstance
    AllocatedStorage: '20'
    DBInstanceClass: db.t2.micro
    Engine: mysql
    MasterUsername: '{{resolve:secretsmanager:MySecret:SecretString:username}}'
    MasterUserPassword: '{{resolve:secretsmanager:MySecret:SecretString:password}}'
```

전체 SecretString 검색

다음 동적 참조는 SecretString의 *MySecret*을 검색합니다.

```
{{resolve:secretsmanager:MySecret}}
```

대안:

```
{{resolve:secretsmanager:MySecret:::}}
```

특정 버전의 보안 암호에서 값 검색

다음 동적 참조는 *AWSPREVIOUS* 버전의 *password*에 대한 *MySecret* 값을 검색합니다.

```
{{resolve:secretsmanager:MySecret:SecretString:password:AWSPREVIOUS}}
```

다른 AWS 계정에서 보안 암호 검색

다음 동적 참조는 다른 AWS 계정에 있는 *MySecret*에 대한 *SecretString*을 검색합니다. 다른 AWS 계정의 보안 암호에 액세스하려면 전체 보안 암호 ARN을 지정해야 합니다.

```
{{resolve:secretsmanager:arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret}}
```

다음 동적 참조는 다른 AWS 계정에 있는 *MySecret*에 대한 *password* 값을 검색합니다. 다른 AWS 계정의 보안 암호에 액세스하려면 전체 보안 암호 ARN을 지정해야 합니다.

```
{{resolve:secretsmanager:arn:aws:secretsmanager:us-west-2:123456789012:secret:MySecret:SecretString:password}}
```

가상 파라미터를 사용하여 AWS 값 가져오기

가상 파라미터는 배포 또는 환경 간에 변경될 수 있는 계정 ID, 리전 이름, 스택 세부 정보 등의 중요한 AWS 환경 정보에 대한 액세스를 제공하는 기본 제공 변수입니다.

하드 코딩된 값 대신 가상 파라미터를 사용하여 템플릿을 더 쉽게 이동하고 여러 AWS 계정 및 리전에 재사용할 수 있습니다.

구문

가상 파라미터는 `Ref` 내장 함수 또는 `Fn::Sub` 내장 함수를 사용하여 참조할 수 있습니다.

Ref

`Ref` 내장 함수는 다음과 같은 일반 구문을 사용합니다. 자세한 내용은 [Ref](#)를 참조하세요.

JSON

```
{ "Ref" : "AWS::PseudoParameter" }
```

YAML

```
!Ref AWS::PseudoParameter
```

Fn::Sub

Fn::Sub 내장 함수는 가상 파라미터 주위에 \${} 구문을 포함하는 다른 형식을 사용합니다. 자세한 내용은 [Fn::Sub](#) 단원을 참조하십시오.

JSON

```
{ "Fn::Sub" : "${AWS::PseudoParameter}" }
```

YAML

```
!Sub '${AWS::PseudoParameter}
```

사용 가능한 의사 파라미터

AWS::AccountId

스택이 생성되는 계정의 AWS 계정 ID를 반환합니다(예: 123456789012).

이 가상 파라미터는 계정별 ARN과 관련한 IAM 역할, 정책 및 기타 리소스 정책을 정의하는 데 많이 사용됩니다.

AWS::NotificationARNs

스택 이벤트 알림이 발생한 Amazon SNS 주제의 Amazon 리소스 이름(ARN) 목록을 반환합니다. 스택을 생성하거나 업데이트할 때 AWS CLI의 --notification-arns 옵션을 통해, 또는 콘솔을 통해 이러한 ARN을 지정할 수 있습니다.

단일 값을 반환하는 다른 가상 파라미터와 달리, AWS::NotificationARNs는 ARN 목록을 반환합니다. 목록의 특정 ARN에 액세스하려면 Fn::Select 내장 함수를 사용합니다. 자세한 내용은 [Fn::Select](#) 섹션을 참조하세요.

AWS::NoValue

`Fn::If` 내장 함수에 반환 값으로 지정된 경우 해당 리소스 속성을 제거합니다. 자세한 내용은 [Fn::If](#) 섹션을 참조하세요.

이 가상 파라미터는 특정 조건에서만 포함되어야 하는 조건부 리소스 속성을 생성하는 데 특히 유용합니다.

AWS::Partition

리소스가 있는 파티션을 반환합니다. 표준 AWS 리전에서 파티션은 `aws`입니다. 다른 파티션에 있는 리소스의 경우 파티션은 `aws-partitionname`입니다. 예를 들어 중국(베이징 및 Ningxia) 리전에 있는 리소스에 대한 파티션은 `aws-cn`이고, AWS GovCloud(미국 서부) 리전에 있는 리소스에 대한 파티션은 `aws-us-gov`입니다.

이 파티션은 리소스의 ARN에서 일부분을 구성합니다. `AWS::Partition`을 사용하면 여러 AWS 파티션에서 템플릿이 올바르게 작동합니다.

AWS::Region

포괄 리소스를 생성하는 리전을 나타내는 문자열을 반환합니다(예: `us-west-2`).

수정 없이 여러 AWS 리전에 템플릿이 적용될 수 있기 때문에, 가장 일반적으로 사용되는 가상 파라미터 중 하나입니다.

AWS::StackId

`arn:aws:cloudformation:us-west-2:123456789012:stack/teststack/51af3dc0-da77-11e4-872e-1234567db123`과 같은 스택의 ID(ARN)를 반환합니다.

AWS::StackName

`teststack`과 같은 스택의 이름을 반환합니다.

스택 이름은 고유한 리소스 이름을 생성하는 데 많이 사용되며, 해당 리소스가 특정 스택에 속해 있음을 쉽게 식별하는 데 도움이 됩니다.

AWS::URLSuffix

스택이 배포된 AWS 리전에서 AWS 도메인의 접미사를 반환합니다. 이 접미사는 일반적으로 `amazonaws.com`이지만 중국(베이징) 리전의 경우 이 접미사는 `amazonaws.com.cn`입니다.

이 파라미터는 AWS 서비스 엔드포인트의 URL을 구성할 때 특히 유용합니다.

예제

주제

- [기본 사용법](#)
- [AWS::NotificationARNs 사용하기](#)
- [AWS::NoValue을 사용한 조건부 속성](#)

기본 사용법

다음 예에서는 두 가지 리소스, 즉 Amazon SNS 주제와 해당 주제에 알림을 전송하는 CloudWatch 경보를 생성합니다. AWS::StackName, AWS::Region 및 AWS::AccountId를 사용하여 스택 이름, 현재 AWS 리전 및 계정 ID를 리소스 이름, 설명 및 ARN에 동적으로 삽입합니다.

JSON

```
{
  "Resources": {
    "MyNotificationTopic": {
      "Type": "AWS::SNS::Topic",
      "Properties": {
        "DisplayName": { "Fn::Sub": "Notifications for ${AWS::StackName}" }
      }
    },
    "CPUALarm": {
      "Type": "AWS::CloudWatch::Alarm",
      "Properties": {
        "AlarmDescription": { "Fn::Sub": "Alarm for high CPU in ${AWS::Region}" },
        "AlarmName": { "Fn::Sub": "${AWS::StackName}-HighCPUALarm" },
        "MetricName": "CPUUtilization",
        "Namespace": "AWS/EC2",
        "Statistic": "Average",
        "Period": 300,
        "EvaluationPeriods": 1,
        "Threshold": 80,
        "ComparisonOperator": "GreaterThanThreshold",
        "AlarmActions": [{ "Fn::Sub": "arn:aws:sns:${AWS::Region}:${AWS::AccountId}:${MyNotificationTopic}" }]
      }
    }
  }
}
```

```

    }
  }
}

```

YAML

```

Resources:
  MyNotificationTopic:
    Type: AWS::SNS::Topic
    Properties:
      DisplayName: !Sub Notifications for ${AWS::StackName}
  CPUAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmDescription: !Sub Alarm for high CPU in ${AWS::Region}
      AlarmName: !Sub ${AWS::StackName}-HighCPUAlarm
      MetricName: CPUUtilization
      Namespace: AWS/EC2
      Statistic: Average
      Period: 300
      EvaluationPeriods: 1
      Threshold: 80
      ComparisonOperator: GreaterThanThreshold
      AlarmActions:
        - !Sub arn:aws:sns:${AWS::Region}:${AWS::AccountId}:${MyNotificationTopic}

```

AWS::NotificationARNs 사용하기

다음 예에서는 인스턴스 시작 이벤트 및 시작 오류에 대한 알림을 전송하도록 Auto Scaling 그룹을 구성합니다. 이 구성에는 스택 생성 중에 지정된 Amazon SNS 주제 ARN 목록을 제공하는 `AWS::NotificationARNs` 가상 파라미터를 사용합니다. `Fn::Select` 함수는 해당 목록에서 첫 번째 ARN을 선택합니다.

JSON

```

"myASG": {
  "Type": "AWS::AutoScaling::AutoScalingGroup",
  "Properties": {
    "LaunchTemplate": {
      "LaunchTemplateId": { "Ref": "myLaunchTemplate" },
      "Version": { "Fn::GetAtt": [ "myLaunchTemplate", "LatestVersionNumber" ] }
    }
  }
}

```



```

    },
    "MaxSize": "1",
    "MinSize": "1",
    "VPCZoneIdentifier": [
        "subnetIdAz1",
        "subnetIdAz2",
        "subnetIdAz3"
    ],
    "NotificationConfigurations" : [{
        "TopicARN" : { "Fn::Select" : [ "0", { "Ref" : "AWS::NotificationARNs" } ] },
        "NotificationTypes" : [ "autoscaling:EC2_INSTANCE_LAUNCH",
"autoscaling:EC2_INSTANCE_LAUNCH_ERROR" ]
    }]
}
}

```

YAML

```

myASG:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    LaunchTemplate:
      LaunchTemplateId: !Ref myLaunchTemplate
      Version: !GetAtt myLaunchTemplate.LatestVersionNumber
    MinSize: '1'
    MaxSize: '1'
    VPCZoneIdentifier:
      - subnetIdAz1
      - subnetIdAz2
      - subnetIdAz3
    NotificationConfigurations:
      - TopicARN:
          Fn::Select:
            - '0'
            - Ref: AWS::NotificationARNs
        NotificationTypes:
          - autoscaling:EC2_INSTANCE_LAUNCH
          - autoscaling:EC2_INSTANCE_LAUNCH_ERROR

```

AWS::NoValue을 사용한 조건부 속성

다음 예는 스냅샷 ID가 제공된 경우에만 스냅샷을 사용하는 Amazon RDS DB 인스턴스를 생성합니다. UseDBSnapshot 조건이 true로 평가되면 CloudFormation에서는 DBSnapshotIdentifier 속

성에 대한 DBSnapshotName 파라미터 값을 사용합니다. 조건이 false로 평가되면 CloudFormation이 DBSnapshotIdentifier 속성을 제거합니다.

JSON

```
"MyDB" : {
  "Type" : "AWS::RDS::DBInstance",
  "Properties" : {
    "AllocatedStorage" : "5",
    "DBInstanceClass" : "db.t2.small",
    "Engine" : "MySQL",
    "EngineVersion" : "5.5",
    "MasterUsername" : { "Ref" : "DBUser" },
    "MasterUserPassword" : { "Ref" : "DBPassword" },
    "DBParameterGroupName" : { "Ref" : "MyRDSParamGroup" },
    "DBSnapshotIdentifier" : {
      "Fn::If" : [
        "UseDBSnapshot",
        { "Ref" : "DBSnapshotName" },
        { "Ref" : "AWS::NoValue" }
      ]
    }
  }
}
```

YAML

```
MyDB:
  Type: AWS::RDS::DBInstance
  Properties:
    AllocatedStorage: '5'
    DBInstanceClass: db.t2.small
    Engine: MySQL
    EngineVersion: '5.5'
    MasterUsername:
      Ref: DBUser
    MasterUserPassword:
      Ref: DBPassword
    DBParameterGroupName:
      Ref: MyRDSParamGroup
    DBSnapshotIdentifier:
      Fn::If:
        - UseDBSnapshot
```

- Ref: DBSnapshotName
- Ref: AWS::NoValue

배포된 CloudFormation 스택에서 내보낸 출력 가져오기

동일한 AWS 계정 및 리전에 여러 스택이 있으면 스택 간에 정보를 공유하는 것이 좋습니다. 이는 하나의 스택에서 생성한 리소스를 다른 스택에서 사용해야 할 때 유용합니다.

예를 들어, 서브넷 및 보안 그룹과 같은 네트워크 리소스를 생성하는 웹 서버용 스택이 하나 있을 수 있습니다. 그러면 실제 웹 서버를 생성하는 다른 스택이 첫 번째 스택에서 생성된 네트워크 리소스를 사용할 수 있습니다. 스택의 템플릿 내에서 리소스 ID를 하드 코딩하거나 리소스 ID를 입력 파라미터로 전달할 필요가 없습니다.

스택 간에 정보를 공유하려면 한 스택에서 출력값을 내보내기하고 다른 스택으로 가져오기를 합니다. 작동 방식은 다음과 같습니다.

1. 첫 번째 스택의 템플릿(예: 네트워킹 스택)에서 Outputs 섹션의 Export 필드를 사용하여 내보낼 특정 값을 정의합니다. 자세한 내용은 [CloudFormation 템플릿 Outputs 구문](#) 섹션을 참조하세요.
2. 해당 스택을 생성하거나 업데이트하면 CloudFormation에서는 동일 AWS 계정 및 리전의 다른 스택에서 사용할 수 있도록 출력값을 내보냅니다.
3. 다른 스택의 템플릿에서는 [Fn::ImportValue](#) 함수를 사용하여 첫 번째 스택에서 내보낸 값을 가져옵니다.
4. 두 번째 스택(예: 웹 서버 스택)을 생성하거나 업데이트하면 CloudFormation에서는 첫 번째 스택에서 내보낸 값을 자동으로 검색하여 사용합니다.

연습 및 샘플 템플릿은 [다른 CloudFormation 스택의 리소스 출력 참조](#) 단원을 참조하십시오.

스택 출력값 내보내기과 중첩 스택 사용 비교

중첩 스택은 `AWS::CloudFormation::Stack` 리소스를 사용하여 다른 스택 내에 생성하는 스택입니다. 중첩 스택을 사용하여 단일 스택의 모든 리소스를 배포 및 관리합니다. 중첩 스택 그룹 내 스택 하나의 출력은 그룹 내 다른 스택에 대한 입력으로 사용할 수 있습니다. 이는 값을 내보내는 것과는 다릅니다.

중첩 스택 그룹 내로 정보 공유를 제한하려면 중첩 스택을 사용하는 것이 좋습니다. (중첩 스택 그룹 내에서뿐만 아니라 해당 그룹 외부에서도) 다른 스택과 정보를 공유하려면 값을 내보냅니다. 예를 들어 서브넷을 사용해 단일 스택을 생성한 다음 이 서브넷의 ID를 내보냅니다. 다른 스택은 ID를 가져와

서 해당 서브넷을 사용할 수 있습니다. 각 스택은 자체 서브넷을 생성할 필요가 없습니다. 스택이 서브넷 ID를 가져오는 한 이 ID를 변경하거나 삭제할 수 없습니다.

중첩 스택에 대한 자세한 내용은 [중첩 스택을 사용하여 템플릿을 재사용 가능한 조각으로 분할](#) 섹션을 참조하세요.

고려 사항

교차 스택 참조에 적용되는 제한은 다음과 같습니다.

- 각 AWS 계정의 Export 이름은 한 리전 내에서 고유해야 합니다.
- 여러 리전 간에 교차 스택 참조를 생성할 수 없습니다. Fn::ImportValue 내장 함수는 동일 리전 안에서 내보냈던 값을 가져오는 데만 사용할 수 있습니다.
- 출력의 경우, 리소스에 의존하는 Name 또는 Export 함수를 Ref의 GetAtt 속성값으로 사용할 수 없습니다.

마찬가지로 ImportValue 함수는 리소스에 의존하는 Ref 또는 GetAtt 함수를 포함할 수 없습니다.

- 다른 스택이 출력 값을 가져오면 출력값을 내보내는 스택을 삭제하거나 내보낸 출력 값을 수정할 수 없습니다. 따라서 내보내는 스택을 삭제하거나 출력값을 수정하려면 가져오기를 모두 제거해야 합니다.

내보낸 출력값 나열

스택에서 내보낸 출력 값을 확인해야 하는 경우 다음 방법 중 하나를 사용하세요.

내보낸 출력값을 나열하려면(콘솔)

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 왼쪽 탐색 창에서 내보내기를 선택합니다.

내보낸 출력값을 나열하려면(AWS CLI)

다음 [list-exports](#) 명령을 사용합니다. *us-east-1*을 해당 AWS 리전으로 대체합니다.

```
aws cloudformation list-exports --region us-east-1
```

다음은 예제 출력입니다.

```
{
  "Exports": [
    {
      "ExportingStackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/private-vpc/99764070-b56c-xmpl-bee8-062a88d1d800",
      "Name": "private-vpc-subnet-a",
      "Value": "subnet-07b410xmplddcfa03"
    },
    {
      "ExportingStackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/private-vpc/99764070-b56c-xmpl-bee8-062a88d1d800",
      "Name": "private-vpc-subnet-b",
      "Value": "subnet-075ed3xmplebd2fb1"
    },
    {
      "ExportingStackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/private-vpc/99764070-b56c-xmpl-bee8-062a88d1d800",
      "Name": "private-vpc-vpcid",
      "Value": "vpc-011d7xmpl100e9841"
    }
  ]
}
```

CloudFormation에서는 출력을 내보낸 현재 리전 및 스택에 대해 내보낸 출력의 이름 및 값을 표시합니다. 다른 스택의 템플릿에서 내보낸 출력 값을 사용하려면 내보내기 이름 및 Fn::ImportValue 함수를 사용하여 참조할 수 있습니다.

출력값을 가져오고 내보내는 스택 나열

내보낸 출력 값을 삭제하거나 변경하려면 먼저 어떤 스택이 해당 출력 값을 가져오는지 확인해야 합니다.

내보낸 출력 값을 가져오는 스택을 보려면 다음 방법 중 하나를 사용하세요.

내보낸 출력값을 가져오는 스택을 나열하려면(콘솔)

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 왼쪽 탐색 창에서 내보내기를 선택합니다.

3. 지정된 내보내기 값을 가져오는 스택을 보려면 해당 내보내기 값에 대해 Export Name(내보내기 이름)을 선택합니다. CloudFormation은 값을 가져오는 모든 스택을 나열하는 내보내기 세부 정보 페이지를 표시합니다.

내보낸 출력값을 가져오는 스택을 나열하려면(AWS CLI)

[list-exports](#) 명령을 사용합니다. *us-east-1*을 AWS 리전으로 바꾸고 *private-vpc-vpcid*를 내보낸 출력 값의 이름으로 바꿉니다.

```
aws cloudformation list-exports --region us-east-1 \
  --export-name private-vpc-vpcid
```

CloudFormation에서는 값을 가져오는 스택 목록을 반환합니다.

```
{
  "Imports": [
    "my-app-stack"
  ]
}
```

어떤 스택에서 내보낸 특정 값을 가져오는지 확인한 후에는 해당 출력 값을 참조하는 Fn::ImportValue 함수를 제거하도록 스택을 수정합니다. 내보낸 출력값을 삭제 또는 수정하려면 내보낸 출력 값을 참조하는 가져오기를 모두 제거해야 합니다.

CloudFormation에서 제공하는 파라미터 유형을 사용하여 런타임 시 기존 리소스 지정

템플릿을 생성할 때 CloudFormation에서 제공하는 특수 파라미터 유형을 사용하여 사용자가 기존 AWS 리소스의 식별자 또는 Systems Manager 파라미터의 식별자를 입력해야 하는 파라미터를 생성할 수 있습니다.

주제

- [개요](#)
- [예제](#)
- [고려 사항](#)
- [지원되는 AWS 특정 파라미터 유형](#)
- [지원되는 Systems Manager 파라미터 유형](#)

- [지원되지 않는 Systems Manager 파라미터 유형](#)

개요

CloudFormation에서 파라미터를 사용하여 스택 생성 또는 업데이트 중에 입력 값을 제공하여 스택을 사용자 지정할 수 있습니다. 이 특성을 통해 템플릿을 다양한 시나리오에서 유연하게 재사용할 수 있습니다.

파라미터는 CloudFormation 템플릿의 Parameters 섹션에 정의되어 있습니다. 파라미터마다 이름과 유형이 있으며, 기본값과 허용되는 값 등의 추가 설정이 있을 수 있습니다. 자세한 내용은 [CloudFormation 템플릿 Parameters 구문](#) 섹션을 참조하세요.

파라미터 유형에 따라 파라미터가 허용할 수 있는 입력 값의 종류가 결정됩니다. 예를 들어, Number는 숫자 값만 허용하고 String은 텍스트 입력을 허용합니다.

CloudFormation은 템플릿에서 기존 AWS 리소스 및 Systems Manager 파라미터를 참조하는 데 사용할 수 있는 몇 가지 추가 파라미터 유형을 제공합니다.

이러한 파라미터 유형은 두 가지 범주로 분류됩니다.

- AWS 특정 파라미터 유형 - CloudFormation은 스택을 생성하거나 업데이트할 때 잘못된 값을 포착하는 데 도움이 되는 파라미터 유형 세트를 제공합니다. 이러한 파라미터 유형을 사용할 때 템플릿을 사용하는 모든 사용자는 스택을 생성하는 AWS 계정과 리전에서 유효한 값을 지정해야 합니다.

AWS Management Console을 사용하는 경우 CloudFormation은 계정과 리전의 미리 채워진 기존 값 목록을 제공합니다. 이렇게 하면 사용자가 특정 이름이나 ID를 기억하고 정확하게 입력할 필요가 없습니다. 대신 드롭다운 목록에서 값을 선택하기만 하면 됩니다. 경우에 따라 ID, 이름 또는 Name 태그 값으로 값을 검색할 수도 있습니다.

- Systems Manager 파라미터 유형 - CloudFormation은 또한 Systems Manager Parameter Store의 기존 파라미터에 해당하는 파라미터 유형을 제공합니다. 이러한 파라미터 유형을 사용하는 경우 템플릿을 사용하는 모든 사용자는 Parameter Store 파라미터 키를 Systems Manager 파라미터 유형 값으로 지정해야 합니다. 그러면 CloudFormation이 Parameter Store에서 스택에 사용할 최신 값을 검색합니다. 이는 새로운 Amazon Machine Image(AMI) ID와 같은 새 속성 값으로 애플리케이션을 자주 업데이트해야 할 때 유용할 수 있습니다. Parameter Store에 대한 자세한 내용은 [Systems Manager Parameter Store](#)를 참조하세요.

Parameters 섹션에 파라미터가 정의되면 Ref 함수를 사용하여 CloudFormation 템플릿 전체에서 파라미터 값을 참조할 수 있습니다.

예제

다음 예에서는 다음 파라미터 유형을 사용하는 템플릿을 보여줍니다.

- AWS::EC2::VPC::Id
- AWS::EC2::Subnet::Id
- AWS::EC2::KeyPair::KeyName
- AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>

이 템플릿에서 스택을 생성하려면 계정에서 기존 VPC ID, 서브넷 ID 및 키 페어 이름을 지정해야 합니다. 원하는 AMI ID를 참조하는 기존 Parameter Store 키를 지정하거나 기본값인 `/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2`를 유지할 수도 있습니다. 이 공용 파라미터는 최신 Amazon Linux 2 AMI의 리전 AMI ID에 대한 별칭입니다. 공용 파라미터에 대한 자세한 내용은 AWS Systems Manager 사용 설명서의 [Parameter Store에서 퍼블릭 파라미터 검색](#)을 참조하세요.

JSON

```
{
  "Parameters": {
    "VpcId": {
      "Description": "ID of an existing Virtual Private Cloud (VPC).",
      "Type": "AWS::EC2::VPC::Id"
    },
    "PublicSubnetId": {
      "Description": "ID of an existing public subnet within the specified VPC.",
      "Type": "AWS::EC2::Subnet::Id"
    },
    "KeyName": {
      "Description": "Name of an existing EC2 key pair to enable SSH access to the instance.",
      "Type": "AWS::EC2::KeyPair::KeyName"
    },
    "AMIID": {
      "Description": "Name of a Parameter Store parameter that stores the ID of the Amazon Machine Image (AMI).",
      "Type": "AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>",
      "Default": "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2"
    }
  }
},
```



```

"Resources": {
  "InstanceSecurityGroup": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
      "GroupDescription": "Enable SSH access via port 22",
      "VpcId": { "Ref": "VpcId" },
      "SecurityGroupIngress": [
        {
          "IpProtocol": "tcp",
          "FromPort": 22,
          "ToPort": 22,
          "CidrIp": "0.0.0.0/0"
        }
      ]
    }
  },
  "Ec2Instance": {
    "Type": "AWS::EC2::Instance",
    "Properties": {
      "KeyName": { "Ref": "KeyName" },
      "ImageId": { "Ref": "AMIID" },
      "NetworkInterfaces": [
        {
          "AssociatePublicIpAddress": "true",
          "DeviceIndex": "0",
          "SubnetId": { "Ref": "PublicSubnetId" },
          "GroupSet": [{ "Ref": "InstanceSecurityGroup" }]
        }
      ]
    }
  }
},
"Outputs": {
  "InstanceId": {
    "Value": { "Ref": "Ec2Instance" }
  }
}
}

```

YAML

```

Parameters:
  VpcId:

```

```

    Description: ID of an existing Virtual Private Cloud (VPC).
    Type: AWS::EC2::VPC::Id
PublicSubnetId:
    Description: ID of an existing public subnet within the specified VPC.
    Type: AWS::EC2::Subnet::Id
KeyName:
    Description: Name of an existing EC2 KeyPair to enable SSH access to the instance.
    Type: AWS::EC2::KeyPair::KeyName
AMIId:
    Description: Name of a Parameter Store parameter that stores the ID of the Amazon
Machine Image (AMI).
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2
Resources:
    InstanceSecurityGroup:
        Type: AWS::EC2::SecurityGroup
        Properties:
            GroupDescription: Enable SSH access via port 22
            VpcId: !Ref VpcId
            SecurityGroupIngress:
                - IpProtocol: tcp
                  FromPort: 22
                  ToPort: 22
                  CidrIp: 0.0.0.0/0
    Ec2Instance:
        Type: AWS::EC2::Instance
        Properties:
            KeyName: !Ref KeyName
            ImageId: !Ref AMIID
            NetworkInterfaces:
                - AssociatePublicIpAddress: "true"
                  DeviceIndex: "0"
                  SubnetId: !Ref PublicSubnetId
                  GroupSet:
                      - !Ref InstanceSecurityGroup
Outputs:
    InstanceId:
        Value: !Ref Ec2Instance

```

스택을 생성하는 AWS CLI 명령

다음 [create-stack](#) 명령은 예제 템플릿을 기초로 스택을 생성합니다.

```
aws cloudformation create-stack --stack-name MyStack \
  --template-body file://sampltemplate.json \
  --parameters \
    ParameterKey="VpcId",ParameterValue="vpc-a123baa3" \
    ParameterKey="PublicSubnetId",ParameterValue="subnet-123a351e" \
    ParameterKey="KeyName",ParameterValue="MyKeyName" \
    ParameterKey="AMIId",ParameterValue="MyParameterKey"
```

List<AWS::EC2::Subnet::Id>와 같이 문자열 목록을 허용하는 파라미터 유형을 사용하려면 다음 예와 같이 ParameterValue 내부의 쉼표를 이중 백슬래시로 이스케이프해야 합니다.

```
--parameters ParameterKey="SubnetIDs",ParameterValue="subnet-5ea0c127\\, subnet-6194ea3b\\, subnet-c87f2be0"
```

고려 사항

동적 참조를 사용하여 타사 자격 증명과 같은 민감한 구성 정의에 대한 액세스를 제한하는 것이 좋습니다. 자세한 내용은 [동적 참조를 사용하여 다른 서비스에 저장된 값 가져오기](#) 섹션을 참조하세요.

템플릿 사용자가 다른 AWS 계정의 값을 지정할 수 있도록 하려면 AWS 특정 파라미터 유형을 사용하지 마세요. 대신 String 또는 CommaDelimitedList 유형의 파라미터를 정의하세요.

Systems Manager 파라미터 유형과 관련하여 몇 가지 유의해야 할 사항이 있습니다.

- 확인된 파라미터 값은 콘솔의 스택 파라미터 탭에서 확인하거나 [describe-stacks](#) 또는 [describe-change-set](#)를 실행하여 확인할 수 있습니다. 이러한 값은 스택이 생성되거나 업데이트될 때 설정되므로 Parameter Store의 최신 값과 다를 수 있습니다.
- 스택 업데이트의 경우 기존 값 사용 옵션을 사용하거나 UsePreviousValue를 true로 설정하면 해당 값이 아닌 동일한 Parameter Store 키를 계속 사용한다는 의미입니다. CloudFormation은 항상 최신 값을 검색합니다.
- 허용되는 값이나 기타 제약 조건을 지정하는 경우 CloudFormation은 지정한 파라미터 키에 대해 이를 검증하지만 해당 값을 검증하지 않습니다. Parameter Store 자체에서 값을 검증해야 합니다.
- 스택을 생성하거나 업데이트하고 변경 세트를 생성할 때 CloudFormation은 해당 시점에 Parameter Store에 있는 임의의 값을 사용합니다. 지정한 파라미터가 호출자의 AWS 계정과 연결된 Parameter Store에 없는 경우 CloudFormation은 검증 오류를 반환합니다.
- 변경 세트를 실행하면, CloudFormation은 변경 세트에서 지정한 값을 사용합니다. 변경 세트를 실행하기 전에 값을 확인해야 합니다. 변경 세트를 생성해 실행하는 동안 Parameter Store에서 변경될 수 있기 때문입니다.

- 동일한 AWS 계정에 저장된 Parameter Store 파라미터의 경우 파라미터 이름을 제공해야 합니다. 다른 AWS 계정에서 공유하는 Parameter Store 파라미터의 경우 전체 파라미터 ARN을 제공해야 합니다.

지원되는 AWS 특정 파라미터 유형

CloudFormation에서는 다음과 같은 AWS 관련 유형을 지원합니다.

`AWS::EC2::AvailabilityZone::Name`

가용 영역(예: us-west-2a).

`AWS::EC2::Image::Id`

Amazon EC2 이미지 ID(예: ami-0ff8a91507f77f867). CloudFormation 콘솔은 이 파라미터 유형에 대해 드롭다운 값 목록을 표시하지 않습니다.

`AWS::EC2::Instance::Id`

Amazon EC2 인스턴스 ID(예: i-1e731a32).

`AWS::EC2::KeyPair::KeyName`

Amazon EC2 키 페어 이름.

`AWS::EC2::SecurityGroup::GroupName`

기본 VPC 보안 그룹 이름(예: my-sg-abc).

`AWS::EC2::SecurityGroup::Id`

보안 그룹 ID(예: sg-a123fd85).

`AWS::EC2::Subnet::Id`

서브넷 ID(예: subnet-123a351e).

`AWS::EC2::Volume::Id`

Amazon EBS 볼륨 ID(예: vol-3cdd3f56).

`AWS::EC2::VPC::Id`

VPC ID(예: vpc-a123baa3).

`AWS::Route53::HostedZone::Id`

Amazon Route 53 호스팅 영역 ID(예: Z23YXV40VPL04A).

List<AWS::EC2::AvailabilityZone::Name>

리전 가용 영역 어레이(예: us-west-2a, us-west-2b).

List<AWS::EC2::Image::Id>

Amazon EC2 이미지 ID 어레이(예: ami-0ff8a91507f77f867, ami-0a584ac55a7631c0c).
CloudFormation 콘솔은 이 파라미터 유형에 대해 드롭다운 값 목록을 표시하지 않습니다.

List<AWS::EC2::Instance::Id>

Amazon EC2 인스턴스 ID 어레이(예: i-1e731a32, i-1e731a34).

List<AWS::EC2::SecurityGroup::GroupName>

기본 VPC 보안 그룹 이름 어레이(예: my-sg-abc, my-sg-def).

List<AWS::EC2::SecurityGroup::Id>

보안 그룹 ID 어레이(예: sg-a123fd85, sg-b456fd85).

List<AWS::EC2::Subnet::Id>

서브넷 ID 어레이(예: subnet-123a351e, subnet-456b351e).

List<AWS::EC2::Volume::Id>

Amazon EBS 볼륨 ID 어레이(예: vol-3cdd3f56, vol-4cdd3f56).

List<AWS::EC2::VPC::Id>

VPC ID 어레이(예: vpc-a123baa3, vpc-b456baa3).

List<AWS::Route53::HostedZone::Id>

Amazon Route 53 호스팅 영역 ID 어레이(예: Z23YXV40VPL04A, Z23YXV40VPL04B).

지원되는 Systems Manager 파라미터 유형

CloudFormation은 다음과 같은 Systems Manager 파라미터 유형을 지원합니다.

AWS::SSM::Parameter::Name

시스템 관리자 파라미터 키의 이름입니다. 이 파라미터 유형은 필수 파라미터가 존재하는지 확인하는 데만 사용됩니다. CloudFormation은 파라미터와 연결된 실제 값을 검색하지 않습니다.

`AWS::SSM::Parameter::Value<String>`

값이 문자열인 시스템 관리자 파라미터입니다. 이는 Parameter Store의 String 파라미터 유형과 일치합니다.

`AWS::SSM::Parameter::Value<List<String>>` 또는

`AWS::SSM::Parameter::Value<CommaDelimitedList>`

값이 문자열 목록인 시스템 관리자 파라미터입니다. 이는 Parameter Store의 StringList 파라미터 유형과 일치합니다.

`AWS::SSM::Parameter::Value<AWS-specific parameter type>`

값이 AWS 특정 파라미터 유형인 Systems Manager 파라미터입니다.

다음은 `AWS::EC2::KeyPair::KeyName` 유형을 지정하는 예제입니다.

- `AWS::SSM::Parameter::Value<AWS::EC2::KeyPair::KeyName>`

`AWS::SSM::Parameter::Value<List<AWS-specific parameter type>>`

값이 AWS 특정 파라미터 유형 목록인 Systems Manager 파라미터입니다.

다음은 `AWS::EC2::KeyPair::KeyName` 유형 목록을 지정하는 예제입니다.

- `AWS::SSM::Parameter::Value<List<AWS::EC2::KeyPair::KeyName>>`

지원되지 않는 Systems Manager 파라미터 유형

CloudFormation은 다음 Systems Manager 파라미터 유형을 지원하지 않습니다.

- Systems Manager 파라미터 유형 목록(예: `List<AWS::SSM::Parameter::Value<String>>`)

추가로 CloudFormation에서는 SecureString Systems Manager 파라미터 유형으로 템플릿 파라미터를 정의하는 것을 지원하지 않습니다. 하지만 보안 문자열을 특정 리소스에 대한 파라미터 값으로 지정할 수 있습니다. 자세한 내용은 [동적 참조를 사용하여 다른 서비스에 저장된 값 가져오기](#) 섹션을 참조하세요.

CloudFormation 연습

이 설명서에서는 스택 배포를 직접 실습할 수 있도록 설계된 연습 모음을 제공합니다.

- [다른 CloudFormation 스택의 리소스 출력 참조](#) - 이 연습에서는 다른 스택 내에서 한 CloudFormation 스택의 출력을 참조하는 방법을 보여줍니다. 단일 스택의 모든 리소스를 포함하는 대신 별도의 스택에서 관련 AWS 리소스를 생성하여 보다 모듈화되고 재사용 가능한 템플릿을 생성할 수 있습니다.
- [Amazon EC2에 애플리케이션 배포](#) - CloudFormation을 사용하여 Amazon EC2 인스턴스에 애플리케이션을 자동으로 설치, 구성 및 시작하는 방법을 알아보세요. 이렇게 하면 인스턴스에 직접 연결하지 않고도 배포를 쉽게 복제하고 기존 설치를 업데이트할 수 있습니다.
- [CloudFormation 스택 업데이트](#) - CloudFormation에서 실행 중인 스택에 대한 간단한 업데이트 진행 과정을 살펴보겠습니다.
- [조정 및 로드 밸런싱된 애플리케이션 생성](#) - CloudFormation을 사용하여 확장 가능하고 로드 밸런싱된 애플리케이션을 생성하는 방법을 알아보세요. 이 연습에서는 애플리케이션이 다양한 트래픽 부하를 처리하고고가용성을 유지할 수 있도록 Auto Scaling 그룹, 로드 밸런서 및 기타 관련 리소스를 생성하는 방법을 다룹니다.
- [다른 AWS 계정의 VPC와 피어링](#) - 이 연습에서는 서로 다른 AWS 계정에 있는 두 VPC 간에 가상 프라이빗 클라우드(VPC) 피어링 연결을 생성하는 프로세스를 안내합니다. VPC 피어링을 사용하면 VPC 간에 트래픽을 라우팅하고 마치 동일한 네트워크에 속한 것처럼 리소스에 액세스할 수 있습니다.
- [CloudFormation을 사용하여 CodeDeploy를 통한 ECS 블루/그린 배포 수행](#) - CloudFormation을 사용하여 Amazon ECS에서 AWS CodeDeploy 블루/그린 배포를 수행하는 방법을 알아봅니다. 블루/그린 배포는 최소한의 가동 중지 시간으로 애플리케이션이나 서비스를 업데이트하는 방법입니다.

다른 CloudFormation 스택의 리소스 출력 참조

이 연습에서는 다른 스택 내에서 한 CloudFormation 스택의 출력을 참조하여 보다 모듈화되고 재사용 가능한 템플릿을 생성하는 방법을 보여줍니다.

단일 스택의 모든 리소스를 포함하는 대신 별도의 스택에서 관련 AWS 리소스를 생성합니다. 그런 다음 다른 스택의 필수 리소스 출력을 참조할 수 있습니다. 교차 스택 참조를 출력으로 제한하여 다른 스택에서 참조되는 스택 부분을 제어합니다.

예를 들어, VPC, 보안 그룹 및 퍼블릭 웹 애플리케이션의 서브넷을 포함하는 네트워크 스택과 별도의 퍼블릭 웹 애플리케이션 스택이 있습니다. 웹 애플리케이션에서 네트워크 스택의 보안 그룹과 서브넷을 사용하도록 하려면 웹 애플리케이션 스택에서 네트워크 스택의 리소스 출력을 참조하도록 허용하는 교차 스택 참조를 생성합니다. 교차 스택 참조를 사용하면 웹 애플리케이션 스택 소유자가 네트워크 규칙 또는 자산을 생성하거나 유지할 필요가 없습니다.

교차 스택 참조를 생성하려면 Export 출력 필드를 사용하여 내보낸 리소스 출력 값에 플래그를 표시합니다. 그런 다음 Fn::ImportValue 내장 함수를 사용하여 값을 가져옵니다. 자세한 내용은 [배포된 CloudFormation 스택에서 내보낸 출력 가져오기](#) 섹션을 참조하세요.

Note

CloudFormation은 무료 서비스입니다. 하지만 스택에 포함시키는 AWS 리소스에 대해서는 리소스별 현재 시세로 요금이 청구됩니다. AWS 요금에 대한 자세한 내용은 [각 제품의 세부 정보 페이지](#)를 참조하십시오.

주제

- [샘플 템플릿을 사용하여 네트워크 스택 생성](#)
- [샘플 템플릿을 사용하여 웹 애플리케이션 스택 생성](#)
- [스택이 설계대로 작동하는지 확인](#)
- [AMI 매핑 오류 해결](#)
- [리소스 정리](#)

샘플 템플릿을 사용하여 네트워크 스택 생성

이 연습을 시작하기 전에 Amazon VPC, Amazon EC2 및 CloudFormation 서비스를 모두 사용할 수 있는 IAM 권한이 있는지 확인합니다.

네트워크 스택에는 웹 애플리케이션 스택에서 사용할 VPC, 보안 그룹 및 서브넷이 포함되어 있습니다. 이러한 리소스 외에 네트워크 스택에서는 퍼블릭 액세스를 활성화하기 위해 인터넷 게이트웨이 및 라우팅 테이블을 생성합니다.

웹 애플리케이션 스택을 생성하기 전에 이 스택을 생성해야 합니다. 웹 애플리케이션 스택을 먼저 생성한 경우에는 보안 그룹 또는 서브넷이 없습니다.

스택 템플릿은 <https://s3.amazonaws.com/cloudformation-examples/user-guide/cross-stack/SampleNetworkCrossStack.template>에서 제공됩니다. 스택에서 생성되는 리소스를 보려면 링크를 선택하여 템플릿을 엽니다. Outputs 섹션에서 샘플 템플릿에서 내보내는 네트워킹 리소스를 확인할 수 있습니다. 다른 스택의 네트워킹 리소스를 내보낼 경우 내보낸 리소스의 이름에 스택 이름이 접두사로 붙습니다. 사용자가 네트워킹 리소스를 가져올 때 리소스를 가져올 스택을 지정할 수 있습니다.

네트워크 스택을 생성하려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 스택 페이지에서 오른쪽 상단의 스택 생성을 선택하고 새 리소스 사용(표준)을 선택합니다.
3. 기존 템플릿 선택을 선택하고 템플릿 지정 섹션에서 Amazon S3 URL을 선택합니다.
4. Amazon S3 URL에 다음 URL을 붙여넣습니다. **`https://s3.amazonaws.com/cloudformation-examples/user-guide/cross-stack/SampleNetworkCrossStack.template`**.
5. 다음을 선택합니다.
6. 스택 이름에 **SampleNetworkCrossStack**을 입력하고 다음을 선택합니다.

Note

이 스택의 이름을 기록해 둡니다. 웹 애플리케이션 스택을 시작할 때 스택 이름이 필요합니다.

7. 다음을 선택합니다. 이 연습에서는 태그를 추가하거나 고급 설정을 지정할 필요가 없습니다.
8. 스택 이름과 템플릿 URL이 올바른지 확인한 다음 스택 생성을 선택합니다.

CloudFormation에서 스택을 생성하는 데 몇 분 정도 걸릴 수 있습니다. 모든 리소스가 생성될 때까지 기다렸다가 웹 애플리케이션 스택 생성을 계속합니다.

9. 진행 상황을 모니터링하려면 스택 이벤트를 확인합니다. 자세한 내용은 [스택 진행 상황 모니터링](#) 섹션을 참조하세요.

샘플 템플릿을 사용하여 웹 애플리케이션 스택 생성

웹 애플리케이션 스택에서는 네트워크 스택에서 보안 그룹 및 서브넷을 사용하는 EC2 인스턴스를 생성합니다.

네트워크 스택과 동일한 AWS 리전에서 이 스택을 생성해야 합니다.

스택 템플릿은 <https://s3.amazonaws.com/cloudformation-examples/user-guide/cross-stack/SampleWebAppCrossStack.template>에서 제공됩니다. 스택에서 생성되는 리소스를 보려면 링크를 선택하여 템플릿을 엽니다. Resources 섹션에서 EC2 인스턴스의 속성을 봅니다. Fn::ImportValue 함수를 사용하여 다른 스택에서 네트워킹 리소스를 가져오는 방법을 확인할 수 있습니다.

웹 애플리케이션 스택을 생성하려면

1. 스택 페이지의 오른쪽 상단에서 스택 생성을 선택하고 새 리소스 사용(표준)을 선택합니다.
2. 기존 템플릿 선택을 선택하고 템플릿 지정 섹션에서 Amazon S3 URL을 선택합니다.
3. Amazon S3 URL에 다음 URL을 붙여넣습니다. **`https://s3.amazonaws.com/cloudformation-examples/user-guide/cross-stack/SampleWebAppCrossStack.template`**.
4. 다음을 선택합니다.
5. 스택 이름에는 **SampleWebAppCrossStack**을 입력합니다. 파라미터 섹션에서 NetworkStackName 파라미터에 대한 기본값을 사용한 다음 다음을 선택합니다.

샘플 템플릿에서는 파라미터 값을 사용하여 값을 가져올 스택을 지정합니다.

6. 다음을 선택합니다. 이 연습에서는 태그를 추가하거나 고급 설정을 지정할 필요가 없습니다.
7. 스택 이름과 템플릿 URL이 올바른지 확인한 다음 스택 생성을 선택합니다.

CloudFormation에서 스택을 생성하는 데 몇 분 정도 걸릴 수 있습니다.

스택이 설계대로 작동하는지 확인

스택이 생성된 이후에 리소스를 보고 인스턴스 ID를 기록해 둡니다. 스택 리소스 보기에 대한 자세한 내용은 [CloudFormation 콘솔에서 스택 정보 보기](#) 단원을 참조하십시오.

인스턴스의 보안 그룹과 서브넷을 확인하려면 [Amazon EC2 콘솔](#)에서 인스턴스의 속성을 봅니다. 인스턴스에서 SampleNetworkCrossStack 스택의 보안 그룹 및 서브넷을 사용하는 경우 교차 스택 참조를 성공적으로 생성한 것입니다.

콘솔을 사용하여 스택 출력과 예제 웹 사이트 URL을 보고 웹 애플리케이션이 실행 중인지 확인합니다. 자세한 내용은 [CloudFormation 콘솔에서 스택 정보 보기](#) 섹션을 참조하세요.

AMI 매핑 오류 해결

Template error: Unable to get mapping for AWSRegionArch2AMI::

[region]::HVM64 오류가 발생하면 AWS 리전에 대한 AMI 매핑이 템플릿에 포함되지 않습니다. 매핑을 업데이트하는 대신 Systems Manager 퍼블릭 파라미터를 사용하여 최신 AMI를 동적으로 참조하는 것이 좋습니다.

1. <https://s3.amazonaws.com/cloudformation-examples/user-guide/cross-stack/SampleWebAppCrossStack.template>에서 로컬 시스템에 SampleWebAppCrossStack 템플릿을 다운로드합니다.
2. 전체 AWSRegionArch2AMI 매핑 섹션을 삭제합니다.
3. 다음 Systems Manager 파라미터를 추가합니다.

```
"LatestAmiId": {
  "Description": "The latest Amazon Linux 2 AMI from the Parameter Store",
  "Type": "AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>",
  "Default": "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2"
}
```

4. 기존 ImageId 참조를 바꿉니다.

```
"ImageId": { "Fn::FindInMap": [ "AWSRegionArch2AMI", { "Ref": "AWS::Region" } ,
  "HVM64" ] },
```

변경 후:

```
"ImageId": { "Ref": "LatestAmiId" },
```

이 파라미터는 스택을 배포하는 리전의 최신 Amazon Linux 2 AMI로 자동 확인됩니다.

다른 Linux 배포판의 경우 적절한 파라미터 경로를 사용합니다. 자세한 내용은 AWS Systems Manager 사용 설명서의 [Parameter Store에서 퍼블릭 파라미터 검색](#)을 참조하세요.

5. 계정의 S3 버킷에 수정된 템플릿을 업로드합니다.

```
aws s3 cp SampleWebAppCrossStack.template s3://amzn-s3-demo-bucket/
```

6. 스택을 생성할 때 예시 URL 대신 S3 템플릿 URL을 지정합니다.

리소스 정리

원치 않는 서비스에 대해 요금이 청구되지 않도록 하려면 스택을 삭제합니다.

스택을 삭제하려면

1. CloudFormation 콘솔에서 SampleWebAppCrossStack 스택을 선택합니다.

2. 작업을 선택한 다음 스택 삭제를 선택합니다.
3. 확인 메시지에서 삭제를 선택합니다.
4. 스택이 삭제된 후 SampleNetworkCrossStack 스택에 대해 동일한 단계를 반복합니다.

Note

CloudFormation에서 SampleWebAppCrossStack 스택을 완전히 삭제할 때까지 기다립니다. EC2 인스턴스가 VPC에서 계속 실행 중인 경우 CloudFormation은 SampleNetworkCrossStack 스택에서 VPC를 삭제하지 않습니다.

Amazon EC2에 애플리케이션 배포

CloudFormation을 사용하여 Amazon EC2 인스턴스에서 애플리케이션을 자동으로 설치, 구성 및 시작할 수 있습니다. 그러면 인스턴스에 연결하지 않고 배포를 쉽게 복제하고 기존 설치를 업데이트할 수 있으므로, 많은 시간과 노력을 절약할 수 있습니다.

CloudFormation에는 cloud-init를 기반으로 하는 헬퍼 스크립트(cfn-init, cfn-signal, cfn-get-metadata, cfn-hup) 세트가 포함되어 있습니다. CloudFormation 템플릿에서 이러한 헬퍼 스크립트를 호출하여 동일한 템플릿에 있는 Amazon EC2 인스턴스에서 애플리케이션을 설치, 구성 및 업데이트합니다. 자세한 내용은 CloudFormation 템플릿 참조 가이드의 [CloudFormation 헬퍼 스크립트 참조](#)를 참조하세요.

[시작하기 자습서](#)에서는 기본 bash 스크립트와 UserData를 함께 사용하여 간단한 웹 서버를 생성했습니다. 이는 간단한 'Hello World' 페이지에는 효과적이지만, 실제 애플리케이션에는 다음과 같은 보다 복잡한 구성이 필요한 경우가 많습니다.

- 여러 소프트웨어 패키지가 올바른 순서로 설치되어야 합니다.
- 특정 콘텐츠를 사용하여 복잡한 구성 파일을 생성해야 합니다.
- 서비스가 시작되고 자동으로 실행되도록 구성된 상태여야 합니다.
- 설정 프로세스의 오류 처리 및 검증이 구성되어 있어야 합니다.

CloudFormation의 헬퍼 스크립트는 UserData의 기본 bash 스크립트보다 더 강력하고 유지 관리가 용이한 EC2 인스턴스 구성 방법을 제공합니다. cfn-init 헬퍼 스크립트는 템플릿의 메타데이터에서 구성 데이터를 읽고 이를 인스턴스에 체계적으로 적용합니다.

이 자습서에서는 cfn-init 헬퍼 스크립트를 사용하고 부트스트래핑 프로세스를 모니터링하는 방법을 알아봅니다.

Note

CloudFormation은 무료이지만 생성하는 Amazon EC2 리소스에 대해서는 요금이 부과됩니다. 그러나 AWS를 처음 사용하는 경우 [프리 티어](#)를 활용하여 이 학습 프로세스 중에 비용을 최소화하거나 제거할 수 있습니다.

주제

- [사전 조건](#)
- [부트스트랩 개념 이해](#)
- [간단한 부트스트랩 예로 시작하기](#)
- [파일 및 명령 추가](#)
- [네트워크 보안 추가](#)
- [완전한 부트스트랩 템플릿](#)
- [콘솔을 사용하여 스택 생성](#)
- [부트스트랩 프로세스 모니터링](#)
- [부트스트래핑된 웹 서버 테스트](#)
- [부트스트랩 문제 해결](#)
- [리소스 정리](#)
- [다음 단계](#)

사전 조건

- [첫 번째 스택 생성](#) 자습서를 완료했거나, 그에 준하는 CloudFormation 관련 기본 지식이 있어야 합니다.
- Amazon EC2 및 CloudFormation을 사용할 수 있는 권한을 보유한 IAM 사용자 또는 역할이 있는 AWS 계정에 대한 액세스 권한 또는 관리 사용자 액세스 권한이 있어야 합니다.
- 인터넷에 액세스할 수 있는 가상 프라이빗 클라우드(VPC)가 있어야 합니다. 이 자습서 템플릿에는 최신 AWS 계정과 함께 자동으로 제공되는 기본 VPC가 필요합니다. 기본 VPC가 없거나 삭제된 경우에 적용되는 다른 해결 방법은 [첫 번째 스택 생성](#) 자습서의 문제 해결 섹션을 참조하세요.

부트스트랩 개념 이해

템플릿을 생성하기 전에 부트스트래핑의 작동과 관련한 주요 개념을 알아보겠습니다.

cfn-init 헬퍼 스크립트

CloudFormation은 Amazon EC2 인스턴스에서 소프트웨어를 설치하고 서비스를 시작하는 데 사용할 수 있는 Python 헬퍼 스크립트 세트를 제공합니다. cfn-init 스크립트는 템플릿에서 리소스 메타데이터를 읽고, 해당 구성을 인스턴스에 적용합니다.

이 프로세스는 다음과 같이 작동합니다.

1. 사용자가 EC2 리소스의 Metadata 섹션에서 구성을 정의합니다.
2. UserData 스크립트에서 cfn-init를 직접적으로 호출합니다.
3. cfn-init가 메타데이터를 읽고 구성을 적용합니다.
4. 인스턴스는 사용자가 지정한 설정에 따라 구성됩니다.

메타데이터 구조

구성은 EC2 인스턴스 내에 특정한 구조로 정의됩니다.

```
Resources:
  EC2Instance:
    Type: AWS::EC2::Instance
    Metadata:
      # Metadata section for the resource
      AWS::CloudFormation::Init:
        # Required key that cfn-init looks for
        config:
          # Configuration name (you can have multiple)
          packages:
            # Install packages
          files:
            # Create files
          commands:
            # Run commands
          services:
            # Start/stop services
```

cfn-init 스크립트는 패키지, 그룹, 사용자, 소스, 파일, 명령 및 서비스 순으로 이러한 섹션을 처리합니다.

간단한 부트스트랩 예로 시작하기

Apache를 설치하고 시작하는 간단한 부트스트랩 예로 시작해 보겠습니다.

```
Resources:
  EC2Instance:
```

```

Type: AWS::EC2::Instance
Metadata:
  AWS::CloudFormation::Init:
    config:
      packages:                # Install Apache web server
        yum:
          httpd: []
      services:                # Start Apache and enable it to start on boot
        systemd:
          httpd:
            enabled: true
            ensureRunning: true
Properties:
  ImageId: !Ref LatestAmiId
  InstanceType: !Ref InstanceType
  UserData: !Base64          # Script that runs when instance starts
  Fn::Sub: |
    #!/bin/bash
    yum install -y aws-cfn-bootstrap
    /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource EC2Instance --
    region ${AWS::Region}

```

이 간단한 예는 핵심 개념을 보여줍니다.

- `packages` 섹션에서는 `yum`을 사용하여 `httpd` 패키지를 설치합니다. 이는 Amazon Linux 및 `yum`을 사용하는 기타 Linux 배포판에서 작동합니다.
- `services` 섹션은 `httpd`가 자동으로 시작되고 실행되도록 합니다.
- `UserData`는 최신 부트스트랩 도구를 설치하고 `cfn-init`를 직접적으로 호출합니다.

파일 및 명령 추가

이제 EC2 인스턴스의 `/var/log` 디렉터리에 사용자 지정 웹 페이지와 로그 파일을 추가하여 이 예를 개선해 보겠습니다.

파일 생성

`files` 섹션에서는 특정 콘텐츠가 포함된 파일을 인스턴스에 생성할 수 있습니다. 세로 파이프(`|`)를 사용하면 텍스트의 리터럴 블록(HTML 코드)을 파일(`/var/www/html/index.html`)의 콘텐츠로 전달할 수 있습니다.

```
files:
```

```

/var/www/html/index.html:
  content: |
    <body>
      <h1>Congratulations, you have successfully launched the AWS CloudFormation
sample.</h1>
    </body>

```

명령 실행

commands 섹션은 부트스트랩 프로세스 중에 셸 명령을 실행하는 데 사용됩니다. 이 명령은 EC2 인스턴스의 /var/log/welcome.txt에 로그 파일을 생성합니다. 이 로그 파일을 보려면 SSH 액세스에 사용할 Amazon EC2 키 페어와 인스턴스에 대한 SSH에 사용 가능한 IP 주소 범위가 필요합니다(여기에서는 다루지 않음).

```

commands:
  createWelcomeLog:
    command: "echo 'cfn-init ran successfully!' > /var/log/welcome.txt"

```

네트워크 보안 추가

여기서는 웹 서버를 설정하는 것이므로 웹 트래픽(HTTP)이 EC2 인스턴스에 도달하도록 허용해야 합니다. 이를 위해 IP 주소에서 포트 80으로 들어오는 트래픽을 허용하는 보안 그룹을 생성합니다. 또한 EC2 인스턴스는 패키지 업데이트를 설치하는 등의 경우에 인터넷으로 트래픽을 전송해야 합니다. 기본적으로 보안 그룹은 나가는 트래픽을 모두 허용합니다. 다음으로 SecurityGroupIds 속성을 사용하여 이 보안 그룹을 EC2 인스턴스와 연결합니다.

```

WebServerSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Allow HTTP access from my IP address
    SecurityGroupIngress:
      - IpProtocol: tcp
        Description: HTTP
        FromPort: 80
        ToPort: 80
        CidrIp: !Ref MyIP

```

완전한 부트스트랩 템플릿

이제 이 모든 요소를 하나로 결합해 보겠습니다. 다음은 앞서 설명한 모든 개념을 결합한 완전한 템플릿입니다.

AWS::TemplateFormatVersion: 2010-09-09

Description: Bootstrap an EC2 instance with Apache web server using cfn-init

Parameters:

LatestAmiId:

Description: The latest Amazon Linux 2 AMI from the Parameter Store

Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>

Default: '/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2'

InstanceType:

Description: EC2 instance type

Type: String

Default: t2.micro

AllowedValues:

- t3.micro
- t2.micro

ConstraintDescription: must be a valid EC2 instance type.

MyIP:

Description: Your IP address in CIDR format (e.g. 203.0.113.1/32)

Type: String

MinLength: 9

MaxLength: 18

Default: 0.0.0.0/0

AllowedPattern: '^(\d{1,3}\.){3}\d{1,3}\.\/\d{1,2}\$'

ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.

Resources:

WebServerSecurityGroup:

Type: AWS::EC2::SecurityGroup

Properties:

GroupDescription: Allow HTTP access from my IP address

SecurityGroupIngress:

- IpProtocol: tcp
- Description: HTTP
- FromPort: 80
- ToPort: 80
- CidrIp: !Ref MyIP

WebServer:

Type: AWS::EC2::Instance

Metadata:

AWS::CloudFormation::Init:

```
config:
  packages:
    yum:
      httpd: []
  files:
    /var/www/html/index.html:
      content: |
        <body>
          <h1>Congratulations, you have successfully launched the AWS
CloudFormation sample.</h1>
        </body>
  commands:
    createWelcomeLog:
      command: "echo 'cfn-init ran successfully!' > /var/log/welcome.txt"
  services:
    systemd:
      httpd:
        enabled: true
        ensureRunning: true
Properties:
  ImageId: !Ref LatestAmiId
  InstanceType: !Ref InstanceType
  SecurityGroupIds:
    - !Ref WebServerSecurityGroup
  UserData: !Base64
    Fn::Sub: |
      #!/bin/bash
      yum install -y aws-cfn-bootstrap
      /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource WebServer --
region ${AWS::Region}
  Tags:
    - Key: Name
      Value: Bootstrap Tutorial Web Server

Outputs:
  WebsiteURL:
    Value: !Sub 'http://${WebServer.PublicDnsName}'
    Description: EC2 instance public DNS name
```

콘솔을 사용하여 스택 생성

다음 절차에는 파일에서 샘플 스택 템플릿을 업로드하는 절차가 포함됩니다. 로컬 시스템에서 텍스트 편집기를 열고 템플릿을 추가합니다. `samplelinux2stack.template` 이름으로 파일을 저장합니다.

스택 템플릿을 시작하려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 스택 생성, 새 리소스 사용(표준)를 선택합니다.
3. 템플릿 지정에서 템플릿 파일 업로드, 파일 선택을 차례로 선택하여 `samplelinux2stack.template` 파일을 업로드하세요.
4. 다음을 선택합니다.
5. 스택 세부 정보 지정 페이지에서 스택 이름으로 **BootstrapTutorialStack**을 입력합니다.
6. 파라미터에서 다음을 수행합니다.
 - LatestAmild: 기본값을 그대로 둡니다.
 - InstanceType: EC2 인스턴스 유형으로 `t2.micro` 또는 `t3.micro`를 선택합니다.
 - MyIP: /32 접미사를 사용하여 퍼블릭 IP 주소를 입력합니다.
7. 다음을 두 번 선택한 다음 제출을 선택하여 스택을 생성합니다.

부트스트랩 프로세스 모니터링

추가 소프트웨어를 설치 및 구성하므로 부트스트랩 프로세스는 단순히 EC2를 실행하는 경우보다 오래 걸립니다.

부트스트랩 진행 상황을 모니터링하려면

1. CloudFormation 콘솔에서 스택을 선택하고 이벤트 탭을 엽니다.
2. WebServer CREATE_IN_PROGRESS 이벤트를 살펴봅니다. 부트스트랩 프로세스는 인스턴스가 시작된 후에 시작됩니다.
3. 부트스트랩 프로세스는 일반적으로 몇 분 정도 걸립니다. 완료되면 WebServer CREATE_COMPLETE가 표시됩니다.

부트스트랩 프로세스 중에 발생하는 상황을 확인하려면 인스턴스 로그를 참조합니다.

부트스트랩 로그를 보려면(선택 사항)

1. [EC2 콘솔](#)을 열고 인스턴스를 찾습니다.
2. 인스턴스를 선택하고 작업, 모니터링 및 문제 해결, 시스템 로그 가져오기를 차례로 선택하여 부트스트랩 진행 상태를 확인합니다.
3. 로그가 즉시 표시되지 않으면 기다렸다가 페이지를 새로 고칩니다.

부트스트래핑된 웹 서버 테스트

스택이 CREATE_COMPLETE로 표시되면 웹 서버를 테스트합니다.

웹 서버를 테스트하려면

1. CloudFormation 콘솔에서 스택의 출력 탭으로 이동합니다.
2. WebsiteURL 값을 클릭하여 새 탭에서 웹 서버를 엽니다.
3. Congratulations, you have successfully launched the AWS CloudFormation sample이라는 메시지와 함께 사용자 지정 웹 페이지가 표시됩니다.

Note

페이지가 즉시 로드되지 않으면 1분 정도 기다렸다가 다시 시도합니다. 스택이 CREATE_COMPLETE로 표시된 후에도 부트스트랩 프로세스가 완료되는 중일 수 있습니다.

부트스트랩 문제 해결

부트스트랩 프로세스가 실패하거나 웹 서버가 작동하지 않는 경우, 원인이 될 수 있는 일반적인 문제와 해결 방법은 다음과 같습니다.

일반적인 문제

- 스택 생성 실패 - 이벤트 탭에서 구체적인 오류 메시지를 확인합니다.
- 웹 서버에 액세스할 수 없음 - MyIP 파라미터에서 IP 주소가 올바른지 확인합니다. 끝에 /32를 포함해야 합니다.
- 부트스트랩 프로세스 실패 - 인스턴스는 시작되지만 cfn-init가 실패합니다. 모니터링 섹션의 설명에 따라 시스템 로그를 확인합니다.

리소스 정리

요금이 지속적으로 부과되지 않도록 스택과 해당 리소스를 삭제하여 정리할 수 있습니다.

스택 및 스택의 리소스를 삭제하려면

1. [CloudFormation 콘솔](#)을 엽니다.
2. 스택 페이지에서 생성한 스택의 이름(**BootstrapTutorialStack**) 옆에 있는 옵션을 선택한 다음 삭제를 선택합니다.
3. 확인 메시지가 나타나면 삭제를 선택합니다.
4. 이벤트 탭에서 스택 삭제 프로세스의 진행 상황을 모니터링합니다.
BootstrapTutorialStack의 상태를 DELETE_IN_PROGRESS로 변경합니다. CloudFormation에서 스택 삭제를 완료하면 목록에서 해당 스택이 제거됩니다.

다음 단계

축하합니다! 지금까지 CloudFormation을 사용하여 EC2 인스턴스를 부트스트래핑하는 방법을 배웠습니다. 이제 다음을 이해했습니다.

- cfn-init 헬퍼 스크립트 사용 방법
- 부트스트래핑을 위한 메타데이터를 구성하는 방법
- 패키지를 설치하고, 파일을 생성하고, 명령을 실행하고, 서비스를 관리하는 방법
- 부트스트랩 문제를 모니터링하는 방법

계속 학습하려면:

- 실행 중인 스택을 업데이트하고 cfn-hup 헬퍼 스크립트를 사용하는 방법에 대해 알아보십시오. 자세한 내용은 [CloudFormation 스택 업데이트](#) 섹션을 참조하세요.
- Windows 스택을 부트스트래핑하는 방법을 알아보십시오. 자세한 내용은 [Windows 기반 CloudFormation 스택 부트스트래핑](#) 섹션을 참조하세요.
- 구성 세트가 여러 개인 더 복잡한 부트스트랩 시나리오를 살펴봅니다. 자세한 내용은 CloudFormation 템플릿 참조 안내서의 [cfn-init](#) 및 [AWS::CloudFormation::Init](#)를 참조하세요.
- 부트스트랩 완료 상태 보고를 위한 cfn-signal에 대해 알아보십시오. 자세한 내용은 CloudFormation 템플릿 참조 안내서의 [cfn-signal](#)을 참조하세요.

CloudFormation 스택 업데이트

Note

이 자습서는 [Amazon EC2에 애플리케이션 배포](#) 자습서에 제공된 개념을 기반으로 합니다. 이 자습서를 완료하지 않은 경우 먼저 CloudFormation에서 EC2 부트스트래핑을 이해하는 것이 좋습니다.

이 주제에서는 실행 중인 스택에 대한 간단한 업데이트 진행 상황을 보여줍니다. 다음 단계를 살펴보겠습니다.

1. 초기 스택 생성 - 기본 Amazon Linux 2 AMI를 사용하여 스택을 생성하고, CloudFormation 헬퍼 스크립트를 사용하여 Apache Web Server 및 간단한 PHP 애플리케이션을 설치합니다.
2. 애플리케이션 업데이트 - CloudFormation을 사용하여 애플리케이션의 파일 중 하나를 업데이트하고 소프트웨어를 배포합니다.
3. 키 페어 추가 - Amazon EC2 키 페어를 인스턴스에 추가한 다음, 인스턴스에 대한 SSH 액세스를 허용하도록 보안 그룹을 업데이트합니다.
4. 인스턴스 유형 업데이트 - 기존 Amazon EC2 인스턴스의 인스턴스 유형을 변경합니다.
5. AMI 업데이트 - 스택에서 Amazon EC2 인스턴스용 Amazon Machine Image(AMI)를 변경합니다.

Note

CloudFormation은 무료이지만 생성하는 Amazon EC2 리소스에 대해서는 요금이 부과됩니다. 그러나 AWS를 처음 사용하는 경우 [프리 티어](#)를 활용하여 이 학습 프로세스 중에 비용을 최소화하거나 제거할 수 있습니다.

주제

- [1단계: 초기 스택 생성](#)
- [2단계: 애플리케이션 업데이트](#)
- [3단계: 키 페어를 사용하여 SSH 액세스 추가](#)
- [4단계: 인스턴스 유형 업데이트](#)
- [5단계: AMI 업데이트](#)
- [가용성 및 영향 고려 사항](#)

- [관련 리소스](#)

1단계: 초기 스택 생성

이 주제의 나머지 부분에서 사용할 수 있는 스택을 생성하는 것부터 시작합니다. 간단한 템플릿 하나를 제공했습니다. 이 템플릿은 Apache Web Server에 호스팅되고 Amazon Linux 2 AMI에서 실행 중인 단일 인스턴스 PHP 웹 애플리케이션을 시작합니다.

Apache Web Server, PHP 및 간단한 PHP 애플리케이션은 기본적으로 Amazon Linux 2 AMI에 설치되는 CloudFormation 헬퍼 스크립트를 통해 모두 설치됩니다. 다음 템플릿 코드 조각에서는 설치할 패키지와 파일(여기서는 Amazon Linux 2 AMI용 Yum 리포지토리의 Apache Web Server 및 PHP 인프라)을 설명하는 메타데이터를 표시합니다. 이 코드 조각에서는 Apache Web Server가 실행 중임을 보장하는 Services 섹션도 보여줍니다.

```
WebServerInstance:
  Type: AWS::EC2::Instance
  Metadata:
    AWS::CloudFormation::Init:
      config:
        packages:
          yum:
            httpd: []
            php: []
        files:
          /var/www/html/index.php:
            content: |
              <?php
              echo '<h1>Hello World!</h1>';
              ?>
            mode: '000644'
            owner: apache
            group: apache
        services:
          systemd:
            httpd:
              enabled: true
              ensureRunning: true
```

애플리케이션 자체는 전체가 템플릿 안에 정의되어 있는 'Hello World' 예제입니다. 실제 애플리케이션의 경우, 파일이 Amazon S3, GitHub 또는 다른 리포지토리에 저장되어 템플릿에서 참조될 수도 있습니다. CloudFormation은 패키지(예: RPM 또는 RubyGems)를 다운로드하고 개별 파일을 참조하며

.zip 및 .tar 파일의 압축을 풀어서 Amazon EC2 인스턴스에 애플리케이션 아티팩트를 생성할 수 있습니다.

이 템플릿은 cfn-hup 대몬이 Amazon EC2 인스턴스에 대한 메타데이터에 정의된 구성의 변경 내용을 수신하도록 구성하고 활성화합니다. cfn-hup 대몬을 사용하여 Apache 또는 PHP 버전 등 애플리케이션 소프트웨어를 업데이트하거나 CloudFormation에서 PHP 애플리케이션 파일 자체를 업데이트할 수 있습니다. 템플릿의 동일한 Amazon EC2 리소스에서 가져온 다음 코드 조각에서는 메타데이터에 대한 업데이트를 확인하고 적용하기 위해 2분마다 cfn-init를 직접적으로 호출하도록 cfn-hup을 구성하는 데 필요한 부분을 보여줍니다. 그렇지 않으면 시작 시 cfn-init가 한 번만 실행됩니다.

```
files:
  /etc/cfn/cfn-hup.conf:
    content: !Sub |
      [main]
      stack=${AWS::StackId}
      region=${AWS::Region}
      # The interval used to check for changes to the resource metadata in minutes.
      Default is 15
      interval=2
      mode: '000400'
      owner: root
      group: root
  /etc/cfn/hooks.d/cfn-auto-reloader.conf:
    content: !Sub |
      [cfn-auto-reloader-hook]
      triggers=post.update
      path=Resources.WebServerInstance.Metadata.AWS::CloudFormation::Init
      action=/opt/aws/bin/cfn-init -s ${AWS::StackId} -r WebServerInstance --region
      ${AWS::Region}
      runas=root
services:
  systemd:
    cfn-hup:
      enabled: true
      ensureRunning: true
      files:
        - /etc/cfn/cfn-hup.conf
        - /etc/cfn/hooks.d/cfn-auto-reloader.conf
```

스택을 완료하기 위해 Amazon EC2 인스턴스 정의의 Properties 섹션에서 UserData 속성에는 패키지 및 파일을 설치하기 위해 cfn-init를 직접적으로 호출하는 cloud-init 스크립트가 포함됩니다.

다. 자세한 내용은 CloudFormation 템플릿 참조 가이드의 [CloudFormation 헬퍼 스크립트 참조](#)를 참조하세요. 템플릿은 Amazon EC2 보안 그룹도 생성합니다.

```
AWSTemplateFormatVersion: 2010-09-09
```

```
Parameters:
```

```
  LatestAmiId:
```

```
    Description: The latest Amazon Linux 2 AMI from the Parameter Store
```

```
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
```

```
    Default: '/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2'
```

```
  InstanceType:
```

```
    Description: WebServer EC2 instance type
```

```
    Type: String
```

```
    Default: t3.micro
```

```
    AllowedValues:
```

- t3.nano
- t3.micro
- t3.small
- t3.medium
- t3a.nano
- t3a.micro
- t3a.small
- t3a.medium
- m5.large
- m5.xlarge
- m5.2xlarge
- m5a.large
- m5a.xlarge
- m5a.2xlarge
- c5.large
- c5.xlarge
- c5.2xlarge
- r5.large
- r5.xlarge
- r5.2xlarge
- r5a.large
- r5a.xlarge
- r5a.2xlarge

```
    ConstraintDescription: must be a valid EC2 instance type.
```

```
Resources:
```

```
  WebServerInstance:
```

```

Type: AWS::EC2::Instance
Properties:
  ImageId: !Ref LatestAmiId
  InstanceType: !Ref InstanceType
  SecurityGroupIds:
    - !Ref WebServerSecurityGroup
  UserData:
    Fn::Base64: !Sub |
      #!/bin/bash -xe
      # Get the latest CloudFormation package
      yum update -y aws-cfn-bootstrap
      # Run cfn-init
      /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource
WebServerInstance --region ${AWS::Region} || error_exit 'Failed to run cfn-init'

      # Start up the cfn-hup daemon to listen for changes to the EC2 instance
metadata
      /opt/aws/bin/cfn-hup || error_exit 'Failed to start cfn-hup'
      # Signal success or failure
      /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource
WebServerInstance --region ${AWS::Region}
Metadata:
  AWS::CloudFormation::Init:
    config:
      packages:
        yum:
          httpd: []
          php: []
      files:
        /var/www/html/index.php:
          content: |
            <?php
            echo "<h1>Hello World!</h1>";
            ?>
          mode: '000644'
          owner: apache
          group: apache
        /etc/cfn/cfn-hup.conf:
          content: !Sub |
            [main]
            stack=${AWS::StackId}
            region=${AWS::Region}
            # The interval used to check for changes to the resource metadata in
minutes. Default is 15

```

```

        interval=2
        mode: '000400'
        owner: root
        group: root
    /etc/cfn/hooks.d/cfn-auto-reloader.conf:
        content: !Sub |
            [cfn-auto-reloader-hook]
            triggers=post.update
            path=Resources.WebServerInstance.Metadata.AWS::CloudFormation::Init
            action=/opt/aws/bin/cfn-init -s ${AWS::StackId} -r WebServerInstance --
region ${AWS::Region}
        runas=root
    services:
        systemd:
            httpd:
                enabled: true
                ensureRunning: true
            cfn-hup:
                enabled: true
                ensureRunning: true
            files:
                - /etc/cfn/cfn-hup.conf
                - /etc/cfn/hooks.d/cfn-auto-reloader.conf
    CreationPolicy:
        ResourceSignal:
        Timeout: PT5M

WebServerSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
        GroupDescription: Enable HTTP access via port 80
        SecurityGroupIngress:
            - IpProtocol: tcp
              FromPort: 80
              ToPort: 80
              CidrIp: 0.0.0.0/0

Outputs:
    WebsiteURL:
        Value: !Sub 'http://${WebServerInstance.PublicDnsName}'
        Description: URL of the web application

```

이 템플릿에서 스택을 시작하는 방법

1. 템플릿을 복사하여 텍스트 파일로 시스템에 로컬로 저장하세요. 이후 단계에서는 이 파일을 사용해야 하므로 이 위치를 적어둡니다.
2. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
3. 스택 생성, 새 리소스 사용(표준)을 선택하세요.
4. 기존 템플릿 선택을 선택합니다.
5. 템플릿 지정에서 템플릿 파일 업로드를 선택하고 첫 번째 단계에서 생성한 파일로 이동한 후 다음을 선택하세요.
6. 스택 세부 정보 지정 페이지에서 스택 이름으로 **UpdateTutorial**을 입력하세요.
7. 파라미터에서 모든 파라미터를 동일하게 유지하고 다음을 두 번 선택하세요.
8. 검토 및 생성 화면에서 제출을 선택하세요.

스택 상태가 CREATE_COMPLETE가 된 후에는 출력 탭에 웹 사이트의 URL이 표시됩니다. WebsiteURL 출력 값을 선택하면 새 PHP 애플리케이션의 작동이 표시됩니다.

2단계: 애플리케이션 업데이트

이제 스택을 배포했으며, 애플리케이션을 업데이트하겠습니다. 애플리케이션에서 출력되는 텍스트를 조금 변경해 보겠습니다. 이렇게 하기 위해 이 템플릿 코드 조각에 표시된 대로 index.php 파일에 echo 명령을 추가하겠습니다.

```
files:
  /var/www/html/index.php:
    content: |
      <?php
      echo "<h1>Hello World!</h1>";
      echo "<p>This is an updated version of our application.</p>";
      ?>
    mode: '000644'
    owner: apache
    group: apache
```

텍스트 편집기를 사용하여 로컬로 저장한 템플릿 파일을 수동으로 편집합니다.

이제 스택을 업데이트하세요.

업데이트된 템플릿으로 스택을 업데이트하는 방법

1. CloudFormation 콘솔에서 **UpdateTutorial** 스택을 선택하세요.
2. 업데이트, 직접 업데이트를 선택하세요.
3. 기존 템플릿 교체를 선택하세요.
4. 템플릿 지정에서 템플릿 파일 업로드를 선택하고 템플릿을 업로드한 후 다음을 선택하세요.
5. 스택 세부 정보 지정 페이지에서 모든 파라미터를 동일하게 유지하고 다음을 두 번 선택하세요.
6. 검토 페이지에서 변경 내용을 검토하세요. 변경 내용에서 CloudFormation이 WebServerInstance 리소스를 업데이트함을 확인할 수 있습니다.
7. 제출을 선택합니다.

스택이 UPDATE_COMPLETE 상태이면 WebsiteURL 출력 값을 다시 선택하여 애플리케이션에서 적용한 변경 내용을 확인합니다. cfn-hup 대문은 2분마다 실행되므로 스택이 업데이트되고 나서 애플리케이션이 변경하는 데 최대 2분이 걸릴 수 있습니다.

업데이트된 리소스 세트를 확인하려면 CloudFormation 콘솔로 이동합니다. 이벤트 탭에서 스택 이벤트를 살펴봅니다. 이 특수한 경우에 Amazon EC2 인스턴스 WebServerInstance의 메타데이터가 업데이트되고, 이때 변경 내용이 없는지 확인하기 위해 CloudFormation에서 다른 리소스(WebServerSecurityGroup)를 다시 평가합니다. 다른 스택 리소스는 수정되지 않았습니다. CloudFormation은 스택 내에서 스택 변경에 영향을 받는 리소스만 업데이트합니다. 이러한 변경에는 속성이나 메타데이터 변경과 같은 직접적인 변경도 있고, Ref, GetAtt 또는 기타 내장 템플릿 함수를 통한 데이터 흐름이나 종속성으로 인한 간접적인 변경도 있을 수 있습니다. 자세한 내용은 [내장 함수 참조](#)를 참조하세요.

이 간단한 업데이트에서 프로세스를 보여줍니다. 그러나 Amazon EC2 인스턴스로 배포되는 파일 및 패키지에 대한 보다 복잡한 변경도 가능합니다. 예를 들면 MySQL에 대한 PHP 지원과 함께 인스턴스에 MySQL을 추가해야 할 수도 있습니다. 이렇게 하려면 추가 서비스와 함께 추가적인 패키지 및 파일을 구성에 추가하고 나서 변경 내용을 배포하도록 스택을 업데이트하기만 하면 됩니다.

```
packages:
  yum:
    httpd: []
    php: []
    mysql: []
    php-mysql: []
    mysql-server: []
    mysql-lib: []
```

```

...

services:
  systemd:
    httpd:
      enabled: true
      ensureRunning: true
    cfn-hup:
      enabled: true
      ensureRunning: true
      files:
        - /etc/cfn/cfn-hup.conf
        - /etc/cfn/hooks.d/cfn-auto-reloader.conf
  mysqld:
    enabled: true
    ensureRunning: true

```

CloudFormation 메타데이터를 업데이트하여 애플리케이션에 사용되는 패키지의 새 버전으로 업데이트합니다. 이전 예제에서는 각 패키지의 버전 속성이 비어 있어서 cfn-init가 패키지의 최신 버전을 설치해야 했습니다.

```

packages:
  yum:
    httpd: []
    php: []

```

원할 경우 패키지의 버전 문자열을 지정할 수 있습니다. 이후 업데이트 스택 직접 호출에서 버전 문자열을 변경하는 경우 패키지의 새 버전이 배포됩니다. 다음은 RubyGems 패키지의 버전 번호를 사용하는 예입니다. 버전 관리를 지원하는 패키지마다 특정 버전이 있을 수 있습니다.

```

packages:
  rubygems:
    mysql: []
    rubygems-update:
      - "1.6.2"
    rake:
      - "0.8.7"
    rails:
      - "2.3.11"

```

3단계: 키 페어를 사용하여 SSH 액세스 추가

템플릿에 원래 지정되지 않았던 속성을 추가하도록 템플릿에서 리소스를 업데이트할 수도 있습니다. 이를 보여주기 위해 Amazon EC2 키 페어를 기존 EC2 인스턴스에 추가하고 나서 Amazon EC2 보안 그룹에서 포트 22를 열어 Secure Shell(SSH)을 사용해 인스턴스에 액세스할 수 있도록 하겠습니다.

Amazon EC2 인스턴스에 SSH 액세스를 추가하려면

1. 템플릿에 기존 Amazon EC2 키 페어 및 SSH 위치의 이름을 전달할 추가 파라미터 두 개를 추가합니다.

Parameters:

KeyName:

Description: Name of an existing EC2 KeyPair to enable SSH access to the instance

Type: AWS::EC2::KeyPair::KeyName

ConstraintDescription: must be the name of an existing EC2 KeyPair.

SSHLocation:

Description: The IP address that can be used to SSH to the EC2 instances in CIDR format (e.g. 203.0.113.1/32)

Type: String

MinLength: 9

MaxLength: 18

Default: 0.0.0.0/0

AllowedPattern: '^(\d{1,3}\.){3}\d{1,3}\.\d{1,2}\$'

ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.

2. KeyName 속성을 Amazon EC2 인스턴스에 추가합니다.

WebServerInstance:

Type: AWS::EC2::Instance

Properties:

ImageId: !Ref LatestAmiId

InstanceType: !Ref InstanceType

KeyName: !Ref KeyName

SecurityGroupIds:

- !Ref WebServerSecurityGroup

3. 포트 22 및 SSH 위치를 Amazon EC2 보안 그룹의 수신 규칙에 추가합니다.

WebServerSecurityGroup:

```
Type: AWS::EC2::SecurityGroup
Properties:
  GroupDescription: Enable HTTP access via port 80 and SSH access via port 22
  SecurityGroupIngress:
    - IpProtocol: tcp
      FromPort: 80
      ToPort: 80
      CidrIp: 0.0.0.0/0
    - IpProtocol: tcp
      FromPort: 22
      ToPort: 22
      CidrIp: !Ref SSHLocation
```

4. [2단계: 애플리케이션 업데이트](#)에 설명된 동일한 단계를 사용하여 스택을 업데이트합니다.

4단계: 인스턴스 유형 업데이트

이제 인스턴스 유형을 변경하여 기본 인프라를 업데이트하는 방법을 살펴보겠습니다.

지금까지 생성한 스택에는 t3.micro Amazon EC2 인스턴스가 사용됩니다. 새로 생성한 웹 사이트의 트래픽 양이 t3.micro 인스턴스에서 처리할 수 있는 것보다 점점 더 많아지고 있어서 m5.large Amazon EC2 인스턴스 유형으로 이동하려 한다고 가정합니다. 인스턴스 유형의 아키텍처가 변경되는 경우 다른 AMI로 인스턴스가 생성되어야 합니다. 그러나 t3.micro 및 m5.large 모두 동일한 CPU 아키텍처를 사용하고 Amazon Linux 2(x86_64) AMI를 실행합니다. 자세한 내용은 Amazon EC2 사용 설명서의 [인스턴스 유형 변경을 위한 호환성](#)을 참조하세요.

이전 단계에서 수정한 템플릿을 사용하여 인스턴스 유형을 변경해 보겠습니다. InstanceType이 템플릿에 대한 입력 파라미터이므로 템플릿을 수정할 필요가 없습니다. 스택 세부 정보 지정 페이지에서 파라미터 값을 변경할 수 있습니다.

스택을 새 파라미터 값으로 업데이트하는 방법

1. CloudFormation 콘솔에서 **UpdateTutorial** 스택을 선택하세요.
2. 업데이트, 직접 업데이트를 선택하세요.
3. 현재 템플릿 사용을 선택하고 다음을 선택하세요.
4. 스택 세부 정보 지정 페이지에서 InstanceType 텍스트 상자의 값을 t3.micro에서 m5.large로 변경하세요. 그리고 다음을 두 번 선택하세요.
5. 검토 페이지에서 변경 내용을 검토하세요. 변경 내용에서 CloudFormation이 WebServerInstance 리소스를 업데이트함을 확인할 수 있습니다.

6. 제출을 선택합니다.

인스턴스를 시작했다가 중지하여 EBS 지원 Amazon EC2 인스턴스의 인스턴스 유형을 동적으로 변경할 수 있습니다. CloudFormation에서는 인스턴스 유형을 업데이트하고 인스턴스를 다시 시작하여 변경을 최적화하려고 하므로 인스턴스 ID가 변경되지 않습니다. 하지만 인스턴스가 다시 시작되는 경우 인스턴스의 퍼블릭 IP 주소가 변경됩니다. 변경 후 탄력적 IP 주소가 올바르게 바인딩되도록 하기 위해 CloudFormation에서 탄력적 IP 주소도 업데이트합니다. CloudFormation 콘솔의 이벤트 탭에서 변경 내용을 확인할 수 있습니다.

AWS Management Console에서 인스턴스 유형을 확인하려면 Amazon EC2 콘솔을 열고 해당 인스턴스를 찾습니다.

5단계: AMI 업데이트

이제 차세대 Amazon Linux인 Amazon Linux 2023을 사용하도록 스택을 업데이트해 보겠습니다.

AMI 업데이트는 인스턴스를 교체해야 하는 주요 변경 사항입니다. AMI를 수정하기 위해 단순히 인스턴스를 시작했다가 중지할 수 없으므로 CloudFormation은 이 작업을 리소스의 변경 불가능 속성에 대한 변경으로 간주합니다. 변경 불가능한 속성을 변경하려면 CloudFormation에서 대체 리소스(이 경우에는 새 AMI를 실행 중인 Amazon EC2 인스턴스)를 시작해야 합니다.

Amazon Linux 2023을 사용하도록 스택 템플릿을 업데이트하는 방법을 살펴보겠습니다. 주요 변경 사항으로, AMI 파라미터 업데이트 및 yum에서 dnf 패키지 관리자로의 변경이 포함됩니다.

```
AWSTemplateFormatVersion: 2010-09-09
```

```
Parameters:
```

```
  LatestAmiId:
```

```
    Description: The latest Amazon Linux 2023 AMI from the Parameter Store
```

```
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
```

```
    Default: '/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-6.1-x86_64'
```

```
  InstanceType:
```

```
    Description: WebServer EC2 instance type
```

```
    Type: String
```

```
    Default: t3.micro
```

```
    AllowedValues:
```

```
      - t3.nano
```

```
      - t3.micro
```

```
      - t3.small
```

```
      - t3.medium
```

- t3a.nano
- t3a.micro
- t3a.small
- t3a.medium
- m5.large
- m5.xlarge
- m5.2xlarge
- m5a.large
- m5a.xlarge
- m5a.2xlarge
- c5.large
- c5.xlarge
- c5.2xlarge
- r5.large
- r5.xlarge
- r5.2xlarge
- r5a.large
- r5a.xlarge
- r5a.2xlarge

ConstraintDescription: must be a valid EC2 instance type.

KeyName:

Description: Name of an existing EC2 KeyPair to enable SSH access to the instance

Type: AWS::EC2::KeyPair::KeyName

ConstraintDescription: must be the name of an existing EC2 KeyPair.

SSHLocation:

Description: The IP address that can be used to SSH to the EC2 instances in CIDR format (e.g. 203.0.113.1/32)

Type: String

MinLength: 9

MaxLength: 18

Default: 0.0.0.0/0

AllowedPattern: '^(\d{1,3}\.){3}\d{1,3}\.\d{1,2}\$'

ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.

Resources:

WebServerInstance:

Type: AWS::EC2::Instance

Properties:

ImageId: !Ref LatestAmiId

InstanceType: !Ref InstanceType

KeyName: !Ref KeyName

SecurityGroupIds:

```

- !Ref WebServerSecurityGroup
UserData:
  Fn::Base64: !Sub |
    #!/bin/bash -xe
    # Get the latest CloudFormation package
    dnf update -y aws-cfn-bootstrap
    # Run cfn-init
    /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource
WebServerInstance --region ${AWS::Region} || error_exit 'Failed to run cfn-init'

    # Start up the cfn-hup daemon to listen for changes to the EC2 instance
metadata
    /opt/aws/bin/cfn-hup || error_exit 'Failed to start cfn-hup'
    # Signal success or failure
    /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource
WebServerInstance --region ${AWS::Region}
Metadata:
  AWS::CloudFormation::Init:
    config:
      packages:
        dnf:
          httpd: []
          php: []
      files:
        /var/www/html/index.php:
          content: |
            <?php
            echo "<h1>Hello World!</h1>";
            echo "<p>This is an updated version of our application.</p>";
            echo "<p>Running on Amazon Linux 2023!</p>";
            ?>
          mode: '000644'
          owner: apache
          group: apache
        /etc/cfn/cfn-hup.conf:
          content: !Sub |
            [main]
            stack=${AWS::StackId}
            region=${AWS::Region}
            # The interval used to check for changes to the resource metadata in
minutes. Default is 15
            interval=2
            mode: '000400'
            owner: root

```

```
    group: root
    /etc/cfn/hooks.d/cfn-auto-reloader.conf:
      content: !Sub |
        [cfn-auto-reloader-hook]
        triggers=post.update
        path=Resources.WebServerInstance.Metadata.AWS::CloudFormation::Init
        action=/opt/aws/bin/cfn-init -s ${AWS::StackId} -r WebServerInstance --
region ${AWS::Region}
      runas=root
  services:
    systemd:
      httpd:
        enabled: true
        ensureRunning: true
      cfn-hup:
        enabled: true
        ensureRunning: true
        files:
          - /etc/cfn/cfn-hup.conf
          - /etc/cfn/hooks.d/cfn-auto-reloader.conf
  CreationPolicy:
    ResourceSignal:
      Timeout: PT5M

WebServerSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Enable HTTP access via port 80 and SSH access via port 22
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 80
        ToPort: 80
        CidrIp: 0.0.0.0/0
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: !Ref SSHLocation

Outputs:
  WebsiteURL:
    Value: !Sub 'http://${WebServerInstance.PublicDnsName}'
    Description: URL of the web application
```

[2단계: 애플리케이션 업데이트](#)에 설명된 동일한 단계를 사용하여 스택을 업데이트합니다.

새 인스턴스를 실행한 후에는 CloudFormation에서 새 리소스를 가리키도록 스택의 다른 리소스를 업데이트합니다. 새로운 모든 리소스가 생성되면 이전 리소스가 삭제되며, 이를 UPDATE_CLEANUP 프로세스라고 합니다. 현재는 업데이트의 결과로서 스택 내 인스턴스의 인스턴스 ID와 애플리케이션 URL이 변경되었습니다. 이벤트 테이블의 이벤트에는 리소스가 대체되었음을 나타내는 '요청된 업데이트에서 변경 불가능한 속성에 대한 변경을 포함하므로 새 물리적 리소스가 생성됨'이라는 설명이 포함됩니다.

또는 업데이트할 애플리케이션 코드를 AMI에 작성한 경우 새 애플리케이션을 로드하도록 동일한 스택 업데이트 메커니즘을 사용하여 AMI를 업데이트할 수 있습니다.

사용자 지정 애플리케이션 코드로 AMI를 업데이트하는 방법

1. 애플리케이션이나 운영 체제 변경 내용을 포함하는 새 AMI를 생성하세요. 자세한 내용은 Amazon EC2 사용 설명서의 [Amazon EBS 지원 AMI 생성](#)을 참조하세요.
2. 새 AMI ID를 통합하도록 템플릿을 업데이트합니다.
3. [2단계: 애플리케이션 업데이트](#)에 설명된 동일한 단계를 사용하여 스택을 업데이트합니다.

스택 업데이트 시, CloudFormation은 AMI ID가 변경되었음을 감지하고 나서 위에서 트리거했을 때와 동일한 방식으로 스택 업데이트를 시작합니다.

가용성 및 영향 고려 사항

속성마다 스택의 리소스에 미치는 영향이 다릅니다. CloudFormation을 사용하여 속성을 업데이트할 수 있지만, 변경하기 전에 다음 질문을 고려해야 합니다.

1. 업데이트가 리소스 자체에 어떤 영향을 미치는가? 예를 들어 경보 임계값 업데이트는 업데이트 중에 경보를 비활성화합니다. 아시다시피, 인스턴스 유형을 변경하려면 인스턴스를 중지했다가 다시 시작해야 합니다. CloudFormation은 기본 리소스에 대해 업데이트 또는 수정 작업을 사용하여 리소스를 변경합니다. 업데이트 영향을 이해하려면 특정 리소스에 대한 설명서를 확인해야 합니다.
2. 변경이 가능한가? 불가능한가? Amazon EC2 인스턴스에서 AMI 변경과 같이 리소스 속성에 대한 일부 변경은 기본 서비스에서 지원되지 않습니다. 가능한 변경의 경우, CloudFormation은 기본 리소스에 대해 업데이트 또는 수정 유형의 API를 사용합니다. 불가능한 속성 변경의 경우, CloudFormation은 업데이트된 속성을 사용하여 새 리소스를 생성하고 나서 이전 리소스를 삭제하기 전에 새 리소스를 스택에 연결합니다. CloudFormation이 스택 리소스의 중단 시간을 줄이려고 노력하더라도, 리소스 대체는 다단계 프로세스이며 시간이 걸리는 작업입니다. 스택 구성 중에는 애플

리케이션이 부분적으로만 작동합니다. 예를 들면 요청을 처리하거나 데이터베이스에 액세스하지 못할 수도 있습니다.

관련 리소스

CloudFormation을 사용하여 애플리케이션을 시작하고 다른 구성 및 배포 서비스(예: Puppet 및 Opscode Chef)와 통합하는 방법에 대한 자세한 내용은 다음 백서를 참조하세요.

- [CloudFormation을 통해 애플리케이션 부트스트래핑](#)
- [CloudFormation과 Opscode Chef의 통합](#)
- [CloudFormation과 Puppet의 통합](#)

조정 및 로드 밸런싱된 애플리케이션 생성

이 연습에서는 조정 및 로드 밸런싱된 애플리케이션을 설정하는 데 도움이 되는 스택을 생성합니다. 이 연습에서는 스택을 생성하는 데 사용할 샘플 템플릿을 제공합니다. 예제 템플릿은 auto scaling, Application Load Balancer, 로드 밸런서 및 Auto Scaling 그룹에 대한 트래픽을 제어하는 보안 그룹, 조정 활동에 대한 알림을 게시하기 위한 Amazon SNS 알림 구성을 제공합니다.

이 템플릿은 하나 이상의 Amazon EC2 인스턴스와 Application Load Balancer를 생성합니다. 이 템플릿에서 스택을 생성할 경우 사용한 AWS 리소스에 대한 요금이 청구됩니다.

전체 스택 템플릿

템플릿으로 시작하겠습니다.

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Parameters:
  InstanceType:
    Description: The EC2 instance type
    Type: String
    Default: t3.micro
    AllowedValues:
      - t3.micro
      - t3.small
      - t3.medium
  KeyName:
    Description: Name of an existing EC2 key pair to allow SSH access to the instances
```

```

    Type: AWS::EC2::KeyPair::KeyName
LatestAmiId:
    Description: The latest Amazon Linux 2 AMI from the Parameter Store
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: '/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2'
OperatorEmail:
    Description: The email address to notify when there are any scaling activities
    Type: String
SSHLocation:
    Description: The IP address range that can be used to SSH to the EC2 instances
    Type: String
    MinLength: 9
    MaxLength: 18
    Default: 0.0.0.0/0
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.
Subnets:
    Type: 'List<AWS::EC2::Subnet::Id>'
    Description: At least two public subnets in different Availability Zones in the
selected VPC
VPC:
    Type: AWS::EC2::VPC::Id
    Description: A virtual private cloud (VPC) that enables resources in public subnets
to connect to the internet
Resources:
    ELBSecurityGroup:
        Type: AWS::EC2::SecurityGroup
        Properties:
            GroupDescription: ELB Security Group
            VpcId: !Ref VPC
            SecurityGroupIngress:
                - IpProtocol: tcp
                  FromPort: 80
                  ToPort: 80
                  CidrIp: 0.0.0.0/0
    EC2SecurityGroup:
        Type: AWS::EC2::SecurityGroup
        Properties:
            GroupDescription: EC2 Security Group
            VpcId: !Ref VPC
            SecurityGroupIngress:
                - IpProtocol: tcp
                  FromPort: 80
                  ToPort: 80
            SourceSecurityGroupId:

```

```

        Fn::GetAtt:
        - ELBSecurityGroup
        - GroupId
    - IpProtocol: tcp
      FromPort: 22
      ToPort: 22
      CidrIp: !Ref SSHLocation
EC2TargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    HealthCheckIntervalSeconds: 30
    HealthCheckProtocol: HTTP
    HealthCheckTimeoutSeconds: 15
    HealthyThresholdCount: 5
    Matcher:
      HttpCode: '200'
    Name: EC2TargetGroup
    Port: 80
    Protocol: HTTP
    TargetGroupAttributes:
      - Key: deregistration_delay.timeout_seconds
        Value: '20'
    UnhealthyThresholdCount: 3
    VpcId: !Ref VPC
ALBListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
      - Type: forward
        TargetGroupArn: !Ref EC2TargetGroup
    LoadBalancerArn: !Ref ApplicationLoadBalancer
    Port: 80
    Protocol: HTTP
ApplicationLoadBalancer:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
    Scheme: internet-facing
    Subnets: !Ref Subnets
    SecurityGroups:
      - !GetAtt ELBSecurityGroup.GroupId
LaunchTemplate:
  Type: AWS::EC2::LaunchTemplate
  Properties:
    LaunchTemplateName: !Sub ${AWS::StackName}-launch-template

```



```

LaunchTemplateData:
  ImageId: !Ref LatestAmiId
  InstanceType: !Ref InstanceType
  KeyName: !Ref KeyName
  SecurityGroupIds:
    - !Ref EC2SecurityGroup
  UserData:
    Fn::Base64: !Sub |
      #!/bin/bash
      yum update -y
      yum install -y httpd
      systemctl start httpd
      systemctl enable httpd
      echo "<h1>Hello World!</h1>" > /var/www/html/index.html
NotificationTopic:
  Type: AWS::SNS::Topic
  Properties:
    Subscription:
      - Endpoint: !Ref OperatorEmail
        Protocol: email
WebServerGroup:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    LaunchTemplate:
      LaunchTemplateId: !Ref LaunchTemplate
      Version: !GetAtt LaunchTemplate.LatestVersionNumber
    MaxSize: '3'
    MinSize: '1'
    NotificationConfigurations:
      - TopicARN: !Ref NotificationTopic
        NotificationTypes: ['autoscaling:EC2_INSTANCE_LAUNCH',
'autoscaling:EC2_INSTANCE_LAUNCH_ERROR', 'autoscaling:EC2_INSTANCE_TERMINATE',
'autoscaling:EC2_INSTANCE_TERMINATE_ERROR']
    TargetGroupARNs:
      - !Ref EC2TargetGroup
    VPCZoneIdentifier: !Ref Subnets

```

JSON

```

{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "InstanceType": {

```

```

    "Description": "The EC2 instance type",
    "Type": "String",
    "Default": "t3.micro",
    "AllowedValues": [
        "t3.micro",
        "t3.small",
        "t3.medium"
    ]
},
"KeyName": {
    "Description": "Name of an existing EC2 key pair to allow SSH access to the
instances",
    "Type": "AWS::EC2::KeyPair::KeyName"
},
"LatestAmiId": {
    "Description": "The latest Amazon Linux 2 AMI from the Parameter Store",
    "Type": "AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>",
    "Default": "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2"
},
"OperatorEmail": {
    "Description": "The email address to notify when there are any scaling
activities",
    "Type": "String"
},
"SSHLocation": {
    "Description": "The IP address range that can be used to SSH to the EC2
instances",
    "Type": "String",
    "MinLength": 9,
    "MaxLength": 18,
    "Default": "0.0.0.0/0",
    "ConstraintDescription": "Must be a valid IP CIDR range of the form x.x.x.x/x."
},
"Subnets": {
    "Type": "List<AWS::EC2::Subnet::Id>",
    "Description": "At least two public subnets in different Availability Zones in the
selected VPC"
},
"VPC": {
    "Type": "AWS::EC2::VPC::Id",
    "Description": "A virtual private cloud (VPC) that enables resources in public
subnets to connect to the internet"
}
},

```

```
"Resources":{
  "ELBSecurityGroup":{
    "Type":"AWS::EC2::SecurityGroup",
    "Properties":{
      "GroupDescription":"ELB Security Group",
      "VpcId":{
        "Ref":"VPC"
      },
      "SecurityGroupIngress":[
        {
          "IpProtocol":"tcp",
          "FromPort":80,
          "ToPort":80,
          "CidrIp":"0.0.0.0/0"
        }
      ]
    },
  },
  "EC2SecurityGroup":{
    "Type":"AWS::EC2::SecurityGroup",
    "Properties":{
      "GroupDescription":"EC2 Security Group",
      "VpcId":{
        "Ref":"VPC"
      },
      "SecurityGroupIngress":[
        {
          "IpProtocol":"tcp",
          "FromPort":80,
          "ToPort":80,
          "SourceSecurityGroupId":{
            "Fn::GetAtt":[
              "ELBSecurityGroup",
              "GroupId"
            ]
          }
        },
        {
          "IpProtocol":"tcp",
          "FromPort":22,
          "ToPort":22,
          "CidrIp":{
            "Ref":"SSHLocation"
          }
        }
      ]
    }
  }
}
```

```

    }
  ]
}
},
"EC2TargetGroup":{
  "Type":"AWS::ElasticLoadBalancingV2::TargetGroup",
  "Properties":{
    "HealthCheckIntervalSeconds":30,
    "HealthCheckProtocol":"HTTP",
    "HealthCheckTimeoutSeconds":15,
    "HealthyThresholdCount":5,
    "Matcher":{
      "HttpCode":"200"
    },
    "Name":"EC2TargetGroup",
    "Port":80,
    "Protocol":"HTTP",
    "TargetGroupAttributes":[
      {
        "Key":"deregistration_delay.timeout_seconds",
        "Value":"20"
      }
    ],
    "UnhealthyThresholdCount":3,
    "VpcId":{
      "Ref":"VPC"
    }
  }
},
"ALBListener":{
  "Type":"AWS::ElasticLoadBalancingV2::Listener",
  "Properties":{
    "DefaultActions":[
      {
        "Type":"forward",
        "TargetGroupArn":{
          "Ref":"EC2TargetGroup"
        }
      }
    ],
    "LoadBalancerArn":{
      "Ref":"ApplicationLoadBalancer"
    },
    "Port":80,

```

```
        "Protocol": "HTTP"
      }
    },
    "ApplicationLoadBalancer": {
      "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
      "Properties": {
        "Scheme": "internet-facing",
        "Subnets": {
          "Ref": "Subnets"
        },
        "SecurityGroups": [
          {
            "Fn::GetAtt": [
              "ELBSecurityGroup",
              "GroupId"
            ]
          }
        ]
      }
    },
    "LaunchTemplate": {
      "Type": "AWS::EC2::LaunchTemplate",
      "Properties": {
        "LaunchTemplateName": {
          "Fn::Sub": "${AWS::StackName}-launch-template"
        },
        "LaunchTemplateData": {
          "ImageId": {
            "Ref": "LatestAmiId"
          },
          "InstanceType": {
            "Ref": "InstanceType"
          },
          "KeyName": {
            "Ref": "KeyName"
          },
          "SecurityGroupIds": [
            {
              "Ref": "EC2SecurityGroup"
            }
          ],
          "UserData": {
            "Fn::Base64": {
              "Fn::Join": [
```

```

        "",
        [
            "#!/bin/bash\n",
            "yum update -y\n",
            "yum install -y httpd\n",
            "systemctl start httpd\n",
            "systemctl enable httpd\n",
            "echo \"<h1>Hello World!</h1>\" > /var/www/html/index.html"
        ]
    ]
}
}
}
},
"NotificationTopic":{
    "Type":"AWS::SNS::Topic",
    "Properties":{
        "Subscription":[
            {
                "Endpoint":{
                    "Ref":"OperatorEmail"
                },
                "Protocol":"email"
            }
        ]
    }
},
"WebServerGroup":{
    "Type":"AWS::AutoScaling::AutoScalingGroup",
    "Properties":{
        "LaunchTemplate":{
            "LaunchTemplateId":{
                "Ref":"LaunchTemplate"
            },
            "Version":{
                "Fn::GetAtt":[
                    "LaunchTemplate",
                    "LatestVersionNumber"
                ]
            }
        },
        "MaxSize":"3",
        "MinSize":"1",

```

```

    "NotificationConfigurations":[
      {
        "TopicARN":{
          "Ref":"NotificationTopic"
        },
        "NotificationTypes":[
          "autoscaling:EC2_INSTANCE_LAUNCH",
          "autoscaling:EC2_INSTANCE_LAUNCH_ERROR",
          "autoscaling:EC2_INSTANCE_TERMINATE",
          "autoscaling:EC2_INSTANCE_TERMINATE_ERROR"
        ]
      }
    ],
    "TargetGroupARNs":[
      {
        "Ref":"EC2TargetGroup"
      }
    ],
    "VPCZoneIdentifier":{
      "Ref":"Subnets"
    }
  }
}
}
}

```

템플릿 연습

이 템플릿의 첫 번째 부분에서는 Parameters를 지정합니다. 각 파라미터마다 CloudFormation이 스택을 프로비저닝 할 실행 시간에 대한 값을 할당해야 합니다. 템플릿 뒷부분에 지정된 리소스는 이러한 값을 참조하고 데이터를 사용합니다.

- **InstanceType:** Amazon EC2 Auto Scaling에서 프로비저닝하는 EC2 인스턴스의 유형입니다. 지정하지 않으면 기본적으로 t3.micro가 사용됩니다.
- **KeyName:** 인스턴스에 대한 SSH 액세스를 허용하는 기존 EC2 키 페어입니다.
- **LatestAmiId:** 인스턴스의 Amazon Machine Image(AMI)입니다. 지정하지 않으면 AWS에서 유지 관리하는 AWS Systems Manager 퍼블릭 파라미터를 사용하여 Amazon Linux 2 AMI로 인스턴스가 시작됩니다. 자세한 내용은 AWS Systems Manager 사용 설명서의 [Finding public parameters](#)를 참조하세요.
- **OperatorEmail:** 조정 활동 알림을 보낼 이메일 주소입니다.

- **SSHLocation**: SSH를 인스턴스에 연결하는 데 사용할 수 있는 IP 주소 범위입니다.
- **Subnets**: 서로 다른 가용 영역에 두 개 이상의 퍼블릭 서브넷을 보유해야 합니다.
- **VPC**: 퍼블릭 서브넷의 리소스를 인터넷에 연결하는 계정의 Virtual Private Cloud(VPC)입니다.

Note

기본 VPC와 기본 서브넷을 사용하여 인스턴스가 인터넷에 액세스하도록 허용할 수 있습니다. 자체 VPC를 사용 중인 경우, 작업 중인 지역의 각 가용 영역에 매핑된 서브넷이 VPC에 있는지 확인합니다. 로드 밸런서를 생성하려면 사용 가능한 퍼블릭 서브넷이 최소한 두 개 있어야 합니다.

이 템플릿의 다음 부분에서는 **Resources**를 지정합니다. 이 섹션에서는 스택 리소스와 해당 속성을 지정합니다.

[AWS::EC2::SecurityGroup](#) 리소스 ELBSecurityGroup

- **SecurityGroupIngress**에는 모든 IP 주소("CidrIp": "0.0.0.0/0")에서 포트 80에 액세스할 수 있도록 하는 TCP 수신 규칙이 포함되어 있습니다.

[AWS::EC2::SecurityGroup](#) 리소스 EC2SecurityGroup

- **SecurityGroupIngress**에는 두 가지 수신 규칙, 즉 1) **SSHLocation** 입력 파라미터에 제공하는 IP 주소 범위에서 SSH(포트 22)에 액세스하도록 허용하는 TCP 수신 규칙과 2) 로드 밸런서의 보안 그룹을 지정하여 로드 밸런서에서 액세스하도록 허용하는 TCP 수신 규칙이 포함되어 있습니다. [GetAtt](#) 함수는 논리명이 **ELBSecurityGroup**인 보안 그룹의 ID를 가져오는 데 사용됩니다.

[AWS::ElasticLoadBalancingV2::TargetGroup](#) 리소스 EC2TargetGroup

- **Port**, **Protocol** 및 **HealthCheckProtocol**은 **ApplicationLoadBalancer**에서 트래픽을 라우팅하고 **Elastic Load Balancing**에서 EC2 인스턴스의 상태를 확인하는 데 사용되는 EC2 인스턴스 포트(80) 및 프로토콜(HTTP)을 지정합니다.
- **HealthCheckIntervalSeconds**는 EC2 인스턴스에서 상태 확인 간에 30초의 간격을 두도록 합니다. **HealthCheckTimeoutSeconds**는 **Elastic Load Balancing**이 상태를 확인할 대상의 응답을 기다리는 시간의 길이로 정의됩니다(이 예에서는 15초). 제한 시간이 지난 후에는 **Elastic Load Balancing**이 EC2 인스턴스의 상태 확인을 비정상 상태로 표시합니다. EC2 인스턴스가 세 번 연속 상태 확인(**UnhealthyThresholdCount**)에 실패할 경우 **Elastic Load Balancing**은 해당 인스턴스에서 5

회 연속 정상 상태 확인(HealthyThresholdCount)이 있을 때까지 해당 EC2 인스턴스에 대한 트래픽 라우팅을 중지합니다. 5회 연속 정상 상태 확인 시점에 Elastic Load Balancing은 인스턴스를 정상으로 간주하고 인스턴스에 대한 트래픽 라우팅을 다시 시작합니다.

- TargetGroupAttributes는 대상 그룹의 등록 취소 지연 값을 20초로 업데이트합니다. 기본적으로 Elastic Load Balancing은 등록 취소 프로세스를 완료하기 전에 300초 동안 기다립니다.

[AWS::ElasticLoadBalancingV2::Listener](#) 리소스 ALBListener

- DefaultActions는 로드 밸런서가 수신하는 포트, 로드 밸런서가 요청을 전달하는 대상 그룹 및 요청을 라우팅하는 데 사용되는 프로토콜을 지정합니다.

[AWS::ElasticLoadBalancingV2::LoadBalancer](#) 리소스 ApplicationLoadBalancer

- Subnets은 Subnets 입력 파라미터 값을 로드 밸런서 노드가 생성될 퍼블릭 서브넷 목록으로 가져옵니다.
- SecurityGroup은 로드 밸런서 노드에서 수신 트래픽을 제어하기 위해 가상 방화벽 역할을 하는 보안 그룹의 ID를 가져옵니다. [GetAtt](#) 함수는 논리명이 ELBSecurityGroup인 보안 그룹의 ID를 가져오는 데 사용됩니다.

[AWS::EC2::LaunchTemplate](#) 리소스 LaunchTemplate

- ImageId는 사용할 AMI로 LatestAmiId 입력 파라미터 값을 가져옵니다.
- KeyName은 사용할 EC2 키 페어로 KeyName 입력 파라미터 값을 가져옵니다.
- SecurityGroupIds는 EC2 인스턴스에서 수신 트래픽을 제어하기 위해 가상 방화벽 역할을 하는 논리명이 EC2SecurityGroup인 보안 그룹의 ID를 가져옵니다.
- UserData는 인스턴스가 가동되어 실행된 후에 실행되는 구성 스크립트입니다. 이 예제에서 스크립트는 Apache를 설치하고 index.html 파일을 생성합니다.

[AWS::SNS::Topic](#) 리소스 NotificationTopic

- Subscription은 조정 활동이 있을 때 알림 수신자의 이메일 주소로 OperatorEmail 입력 파라미터 값을 가져옵니다.

[AWS::AutoScaling::AutoScalingGroup](#) 리소스 WebServerGroup

- MinSize 및 MaxSize는 auto scaling의 최소 및 최대 EC2 인스턴스 수를 설정합니다.
- TargetGroupARNs는 논리명이 EC2TargetGroup인 대상 그룹의 ARN을 가져옵니다. 이 auto scaling은 규모를 조정하면서 이 대상 그룹에 인스턴스를 자동으로 등록 및 등록 취소합니다.
- VPCZoneIdentifier는 Subnets 입력 파라미터 값을 EC2 인스턴스가 생성될 수 있는 퍼블릭 서브넷 목록으로 가져옵니다.

1단계: 스택 시작

스택을 시작하기 전에 Amazon EC2, Amazon EC2 Auto Scaling, AWS Systems Manager, Elastic Load Balancing, Amazon SNS, CloudFormation 등의 서비스를 모두 사용할 수 있는 AWS Identity and Access Management(IAM) 권한이 있는지 확인합니다.

다음 절차에는 파일에서 샘플 스택 템플릿을 업로드하는 절차가 포함됩니다. 로컬 시스템에서 텍스트 편집기를 열고 템플릿 중 하나를 추가합니다. `sampleloadbalancedappstack.template` 이름으로 파일을 저장합니다.

스택 템플릿을 시작하려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 스택 생성, 새 리소스 사용(표준)를 선택합니다.
3. 템플릿 지정에서 템플릿 파일 업로드, 파일 선택을 차례로 선택하여 `sampleloadbalancedappstack.template` 파일을 업로드하세요.
4. 다음을 선택합니다.
5. 스택 세부 정보 지정 페이지에 스택 이름(예: **SampleLoadBalancedAppStack**)을 입력합니다.
6. 파라미터에서 스택의 매개 변수를 검토하고 OperatorEmail, SSHLocation, KeyName, VPC, 서브넷을 포함하여 기본값이 없는 모든 파라미터의 값을 제공합니다.
7. 다음을 두 번 선택합니다
8. 검토 페이지에서 설정을 검토하고 확인합니다.
9. 제출을 선택합니다.

CloudFormation 콘솔의 상태 열에서 스택의 상태를 볼 수 있습니다. CloudFormation에서 스택이 생성되면 CREATE_COMPLETE 상태가 표시됩니다.

Note

스택을 생성한 후 구독을 확인해야 이메일 주소로 알림 수신을 시작할 수 있습니다. 자세한 정보는 Amazon EC2 Auto Scaling 사용 설명서의 [auto scaling 조정 시 Amazon SNS 알림 수신](#)을 참조하세요.

2단계: 샘플 리소스 정리

사용하지 않은 샘플 리소스에 요금이 부과되지 않도록 하려면 스택을 삭제하세요.

스택을 삭제하려면

1. CloudFormation 콘솔에서 SampleLoadBalancedAppStack 스택을 선택합니다.
2. 삭제를 선택합니다.
3. 확인 메시지에서 스택 삭제를 선택합니다.

SampleLoadBalancedAppStack의 상태가 DELETE_IN_PROGRESS로 변경됩니다.

CloudFormation에서 스택 삭제를 완료하면 목록에서 해당 스택이 제거됩니다.

이 연습의 샘플 템플릿을 사용하여 자체 스택 템플릿을 구축합니다. 자세한 내용은 Amazon EC2 Auto Scaling 사용 설명서의 [Tutorial: Set up a scaled and load-balanced application](#)을 참조하세요.

다른 AWS 계정의 VPC와 피어링

[AWS::EC2::VPCPeeringConnection](#)을 사용하여 다른 AWS 계정의 Virtual Private Cloud(VPC)와 피어링할 수 있습니다. 이렇게 하면 두 VPC 간에 네트워킹 연결이 생성되어 두 VPC 간에 트래픽을 라우팅할 수 있습니다. 따라서 두 VPC가 마치 동일한 네트워크 내에 있는 것처럼 통신할 수 있습니다. VPC 피어링 연결을 사용하면 데이터 액세스 및 데이터 전송을 원활하게 실행할 수 있습니다.

VPC 피어링 연결을 설정하려면 단일 CloudFormation 스택 내에서 개별 AWS 계정 2개를 승인해야 합니다.

VPC 피어링 및 제한 사항에 대한 자세한 내용은 [Amazon VPC 피어링 설명서](#)를 참조하세요.

사전 조건

1. 피어링 연결에는 피어 VPC ID, 피어 AWS 계정 ID 및 [크로스 계정 액세스 역할](#)이 필요합니다.

Note

이 연습에서는 두 가지 계정을 참조합니다. 첫 번째는 교차 계정 피어링을 허용하는 계정이고(수락자 계정), 두 번째는 피어링 연결을 요청하는 계정입니다(요청자 계정).

2. VPC 피어링 연결을 수락하려면 교차 계정 액세스 역할을 맡을 수 있어야 합니다. 이 리소스는 동일한 계정의 VPC 피어링 연결 리소스와 동일한 방식으로 동작합니다. IAM 관리자가 교차 계정 역할을 수임할 수 있는 권한을 부여하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 [사용자에게 역할을 전환할 권한 부여](#)를 참조하세요.

1단계: VPC 및 교차 계정 역할 생성

이 단계에서는 수락자 계정에서 VPC 및 역할을 생성합니다.

VPC 및 교차 계정 액세스 역할을 생성하려면 다음을 수행하세요.

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 스택 페이지의 오른쪽 상단에서 스택 생성을 선택하고 새 리소스 사용(표준)을 선택합니다.
3. 사전 조건 - 템플릿 준비에 기존 템플릿 선택을 선택한 다음 템플릿 파일 업로드, 파일 선택을 차례로 선택합니다.
4. 로컬 시스템에서 텍스트 편집기를 열고 다음 템플릿 중 하나를 추가합니다. 파일을 저장하고 콘솔로 돌아가 템플릿 파일로 선택합니다.

Example JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Create a VPC and an assumable role for cross account VPC peering.",
  "Parameters": {
    "PeerRequesterAccountId": {
      "Type": "String"
    }
  },
  "Resources": {
    "vpc": {
      "Type": "AWS::EC2::VPC",
```



```

"Outputs": {
  "VPCId": {
    "Value": {
      "Ref": "vpc"
    }
  },
  "RoleARN": {
    "Value": {
      "Fn::GetAtt": [
        "peerRole",
        "Arn"
      ]
    }
  }
}
}
}

```

Example YAML

```

AWSTemplateFormatVersion: 2010-09-09
Description: Create a VPC and an assumable role for cross account VPC peering.
Parameters:
  PeerRequesterAccountId:
    Type: String
Resources:
  vpc:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: 10.1.0.0/16
      EnableDnsSupport: false
      EnableDnsHostnames: false
      InstanceTenancy: default
  peerRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Principal:
              AWS: !Ref PeerRequesterAccountId
            Action:
              - 'sts:AssumeRole'
            Effect: Allow
      Path: /

```

```

Policies:
  - PolicyName: root
    PolicyDocument:
      Version: 2012-10-17
      Statement:
        - Effect: Allow
          Action: 'ec2:AcceptVpcPeeringConnection'
          Resource: '*'

Outputs:
  VPCId:
    Value: !Ref vpc
  RoleARN:
    Value: !GetAtt
      - peerRole
      - Arn

```

5. 다음을 선택합니다.
6. 스택에 이름(예: **VPC-owner**)을 지정한 다음 PeerRequesterAccountId 필드에 요청자 계정의 AWS 계정 ID를 입력합니다.
7. 기본값을 수락하고 다음을 선택합니다.
8. CloudFormation에서 IAM 리소스를 생성할 수 있음을 승인합니다를 선택하고 스택 생성을 선택하세요.

2단계: **AWS::EC2::VPCPeeringConnection**을 포함하는 템플릿 생성

이제 VPC 및 크로스 계정 역할을 생성했으므로 다른 AWS 계정(요청자 계정)을 사용하여 VPC와 피어링할 수 있습니다.

[AWS::EC2::VPCPeeringConnection](#) 리소스가 포함된 템플릿을 생성하려면 다음을 수행하세요.

1. CloudFormation 콘솔 홈페이지로 되돌아 갑니다.
2. 스택 페이지의 오른쪽 상단에서 스택 생성을 선택하고 새 리소스 사용(표준)을 선택합니다.
3. 사전 조건 - 템플릿 준비에 기존 템플릿 선택을 선택한 다음 템플릿 파일 업로드, 파일 선택을 차례로 선택합니다.
4. 로컬 시스템에서 텍스트 편집기를 열고 다음 템플릿 중 하나를 추가합니다. 파일을 저장하고 콘솔로 돌아가 템플릿 파일로 선택합니다.

Example JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Create a VPC and a VPC Peering connection using the PeerRole to accept.",
  "Parameters": {
    "PeerVPCAccountId": {
      "Type": "String"
    },
    "PeerVPCId": {
      "Type": "String"
    },
    "PeerRoleArn": {
      "Type": "String"
    }
  },
  "Resources": {
    "vpc": {
      "Type": "AWS::EC2::VPC",
      "Properties": {
        "CidrBlock": "10.2.0.0/16",
        "EnableDnsSupport": false,
        "EnableDnsHostnames": false,
        "InstanceTenancy": "default"
      }
    },
    "vpcPeeringConnection": {
      "Type": "AWS::EC2::VPCPeeringConnection",
      "Properties": {
        "VpcId": {
          "Ref": "vpc"
        },
        "PeerVpcId": {
          "Ref": "PeerVPCId"
        },
        "PeerOwnerId": {
          "Ref": "PeerVPCAccountId"
        },
        "PeerRoleArn": {
          "Ref": "PeerRoleArn"
        }
      }
    }
  }
}
```



```

    }
  },
  "Outputs": {
    "VPCId": {
      "Value": {
        "Ref": "vpc"
      }
    },
    "VPCPeeringConnectionId": {
      "Value": {
        "Ref": "vpcPeeringConnection"
      }
    }
  }
}

```

Example YAML

```

AWSTemplateFormatVersion: 2010-09-09
Description: Create a VPC and a VPC Peering connection using the PeerRole to
  accept.
Parameters:
  PeerVPCAccountId:
    Type: String
  PeerVPCId:
    Type: String
  PeerRoleArn:
    Type: String
Resources:
  vpc:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: 10.2.0.0/16
      EnableDnsSupport: false
      EnableDnsHostnames: false
      InstanceTenancy: default
  vpcPeeringConnection:
    Type: AWS::EC2::VPCPeeringConnection
    Properties:
      VpcId: !Ref vpc
      PeerVpcId: !Ref PeerVPCId
      PeerOwnerId: !Ref PeerVPCAccountId
      PeerRoleArn: !Ref PeerRoleArn

```

```

Outputs:
  VPCId:
    Value: !Ref vpc
  VPCPeeringConnectionId:
    Value: !Ref vpcPeeringConnection

```

5. 다음을 선택합니다.
6. 스택에 이름을 제공합니다(예: **VPC-peering-connection**).
7. 기본값을 수락하고 다음을 선택합니다.
8. CloudFormation에서 IAM 리소스를 생성할 수 있음을 승인합니다를 선택하고 스택 생성을 선택하세요.

아주 제한적인 정책으로 템플릿 생성

VPC를 다른 AWS 계정 계정에 피어링할 때 매우 제한적인 정책을 생성하고자 할 수 있습니다.

다음 예제 템플릿은 VPC 피어 소유자 템플릿(위 1단계에서 생성한 수락자 계정)을 더욱 제한적으로 변경하는 방법을 보여줍니다.

Example JSON

```

{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Create a VPC and an assumable role for cross account VPC peering.",
  "Parameters": {
    "PeerRequesterAccountId": {
      "Type": "String"
    }
  },
  "Resources": {
    "peerRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Statement": [
            {
              "Action": [
                "sts:AssumeRole"
              ],
              "Effect": "Allow",
              "Principal": {
                "AWS": {

```

```

        "Ref": "PeerRequesterAccountId"
      }
    }
  ],
  },
  "Path": "/",
  "Policies": [
    {
      "PolicyDocument": {
        "Statement": [
          {
            "Action": "ec2:acceptVpcPeeringConnection",
            "Effect": "Allow",
            "Resource": {
              "Fn::Sub": "arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:vpc/${vpc}"
            }
          },
          {
            "Action": "ec2:acceptVpcPeeringConnection",
            "Condition": {
              "StringEquals": {
                "ec2:AcceptorVpc": {
                  "Fn::Sub": "arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:vpc/${vpc}"
                }
              }
            },
            "Effect": "Allow",
            "Resource": {
              "Fn::Sub": "arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:vpc-peering-connection/*"
            }
          }
        ],
        "Version": "2012-10-17"
      },
      "PolicyName": "root"
    }
  ]
},
{
  "vpc": {
    "Type": "AWS::EC2::VPC",

```

```

    "Properties":{
      "CidrBlock":"10.1.0.0/16",
      "EnableDnsHostnames":false,
      "EnableDnsSupport":false,
      "InstanceTenancy":"default"
    }
  },
  "Outputs":{
    "RoleARN":{
      "Value":{
        "Fn::GetAtt":[
          "peerRole",
          "Arn"
        ]
      }
    },
    "VPCId":{
      "Value":{
        "Ref":"vpc"
      }
    }
  }
}

```

Example YAML

```

AWSTemplateFormatVersion: 2010-09-09
Description: Create a VPC and an assumable role for cross account VPC peering.
Parameters:
  PeerRequesterAccountId:
    Type: String
Resources:
  peerRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Action:
              - 'sts:AssumeRole'
            Effect: Allow
            Principal:
              AWS:

```

```

    Ref: PeerRequesterAccountId
  Path: /
  Policies:
    - PolicyDocument:
        Statement:
          - Action: 'ec2:acceptVpcPeeringConnection'
            Effect: Allow
            Resource:
              'Fn::Sub': 'arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:vpc/${vpc}'
          - Action: 'ec2:acceptVpcPeeringConnection'
            Condition:
              StringEquals:
                'ec2:AcceptorVpc':
                  'Fn::Sub': 'arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:vpc/
${vpc}'
            Effect: Allow
            Resource:
              'Fn::Sub': >-
                arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:vpc-peering-
connection/*
        Version: 2012-10-17
        PolicyName: root
  vpc:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: 10.1.0.0/16
      EnableDnsHostnames: false
      EnableDnsSupport: false
      InstanceTenancy: default
  Outputs:
    RoleARN:
      Value:
        'Fn::GetAtt':
          - peerRole
          - Arn
    VPCId:
      Value:
        Ref: vpc

```

VPC에 액세스하려면 위의 2단계와 동일한 요청자 템플릿을 사용할 수 있습니다.

자세한 내용은 Amazon VPC 피어링 가이드의 [VPC 피어링에 대한 자격 증명 및 액세스 관리](#)를 참조하세요.

CloudFormation을 사용하여 CodeDeploy를 통한 ECS 블루/그린 배포 수행

Amazon Elastic Container Service(Amazon ECS)에서 실행되는 애플리케이션을 업데이트하려면 CodeDeploy 블루/그린 배포 전략을 사용할 수 있습니다. 이 전략은 애플리케이션 버전 변경으로 인한 중단을 최소화하는 데 도움이 됩니다.

블루/그린 배포에서는 현재의 라이브 환경(블루라고 함)과 함께 새 애플리케이션 환경(그린이라고 함)을 생성합니다. 블루 환경에서 그린 환경으로 라이브 트래픽을 라우팅하기 전에 그린 환경을 모니터링하고 테스트할 수 있습니다. 그린 환경에서 실시간 트래픽을 제공하고 나면 블루 환경을 안전하게 종료할 수 있습니다.

CloudFormation을 사용하여 ECS에서 CodeDeploy 블루/그린 배포를 수행하려면 다음 정보를 스택 템플릿에 포함하세요.

- `AWS::CodeDeploy::BlueGreen` 후크를 설명하는 Hooks 섹션.
- `AWS::CodeDeployBlueGreen` 변환을 지정하는 Transform 섹션.

다음 주제에서는 ECS에서의 블루/그린 배포를 위한 CloudFormation 템플릿을 설정하는 방법을 안내합니다.

주제

- [블루/그린 배포 관련 정보](#)
- [CloudFormation을 사용하여 ECS 블루/그린 배포를 관리할 때 고려 사항](#)
- [AWS::CodeDeploy::BlueGreen 후크 구문](#)
- [블루/그린 배포 템플릿 예제](#)

블루/그린 배포 관련 정보

이 주제에서는 CloudFormation을 사용한 블루/그린 배포의 작동 방식에 대한 개요를 제공합니다. 또한 블루/그린 배포를 위해 CloudFormation 템플릿을 준비하는 방법도 설명합니다.

주제

- [작동 방식](#)
- [그린 배포를 개시하는 리소스 업데이트](#)
- [ECS 블루/그린 배포를 수행하기 위한 템플릿 준비](#)

- [CloudFormation 리소스를 사용하여 블루/그린 배포 모델링](#)
- [변경 세트](#)
- [스택 이벤트 모니터링](#)
- [블루/그린 배포를 위한 IAM 권한](#)

작동 방식

CloudFormation을 사용하여 CodeDeploy를 통해 ECS 블루/그린 배포를 수행하는 경우 사용할 트래픽 라우팅 및 안정화 설정 지정을 포함해 블루 및 그린 애플리케이션 환경 모두에 리소스를 정의하는 스택 템플릿을 먼저 생성합니다. 다음으로 해당 템플릿에서 스택을 생성합니다. 그러면 블루(현재) 애플리케이션이 생성됩니다. CloudFormation은 스택 생성 중에만 블루 리소스를 생성합니다. 그린 배포에 대한 리소스는 필요할 때까지 생성되지 않습니다.

따라서 향후 스택 업데이트에서 블루 애플리케이션의 작업 정의 또는 작업 세트 리소스를 업데이트하는 경우 CloudFormation은 다음을 수행합니다.

- 필요한 모든 그린 애플리케이션 환경 리소스 생성
- 지정된 트래픽 라우팅 파라미터를 기반으로 트래픽 이동
- 블루 리소스 삭제

그린 배포가 성공하고 완료되기 전에 오류가 발생하면 CloudFormation은 전체 그린 배포가 시작되기 전의 상태로 스택을 롤백합니다.

그린 배포를 개시하는 리소스 업데이트

특정 ECS 리소스의 특정 속성을 업데이트하는 스택 업데이트를 수행하면 CloudFormation에서 그린 배포 프로세스를 개시합니다. 이 프로세스를 개시하는 리소스는 다음과 같습니다.

- [AWS::ECS::TaskDefinition](#)
- [AWS::ECS::TaskSet](#)

하지만 이러한 리소스에 대한 업데이트에 교체가 필요한 속성 변경이 포함되지 않는 경우 그린 배포가 개시되지 않습니다. 자세한 내용은 [스택 리소스의 업데이트 동작 이해](#) 섹션을 참조하세요.

동일한 스택 업데이트에서 다른 리소스에 대한 업데이트와 함께 위의 리소스에 대한 업데이트를 포함시킬 수 없다는 점이 중요합니다. 나열된 리소스와 동일한 스택 내의 다른 리소스를 모두 업데이트해야 하는 경우 다음 두 가지 옵션을 사용할 수 있습니다.

- 두 개의 별도 스택 업데이트 작업을 수행합니다. 즉, 위의 리소스에 대한 업데이트만 포함시키는 작업과 기타 리소스에 대한 변경 사항을 포함시키는 별도의 스택 업데이트 작업을 수행합니다.
- 템플릿에서 Transform 및 Hooks 섹션을 제거한 다음 스택 업데이트를 수행합니다. 이 경우 CloudFormation이 그린 배포를 수행하지 않습니다.

ECS 블루/그린 배포를 수행하기 위한 템플릿 준비

스택에서 블루/그린 배포를 활성화하려면, 스택 업데이트를 수행하기 전에 다음 섹션을 스택 템플릿에 포함시키십시오.

- `AWS::CodeDeployBlueGreen` 변환에 대한 참조를 템플릿에 추가합니다.

```
"Transform": [
  "AWS::CodeDeployBlueGreen"
],
```

- `AWS::CodeDeploy::BlueGreen` 후크를 호출하고 배포에 대한 속성을 지정하는 Hooks 섹션을 추가합니다. 자세한 내용은 [AWS::CodeDeploy::BlueGreen 후크 구문](#) 섹션을 참조하세요.
- Resources 섹션에서 배포에 대한 블루 및 그린 리소스를 정의합니다.

템플릿을 처음으로 생성할 때(즉, 스택 자체를 만들기 전에) 이러한 섹션을 추가하거나 스택 업데이트를 수행하기 전에 기존 템플릿에 추가할 수 있습니다. 새 스택에 블루/그린 배포를 지정하는 경우 CloudFormation은 스택 생성 중에 블루 리소스만 생성하며 그린 배포에 대한 리소스는 스택 업데이트 중에 필요할 때까지 생성되지 않습니다.

CloudFormation 리소스를 사용하여 블루/그린 배포 모델링

ECS에서 CodeDeploy 블루/그린 배포를 수행하려면 CloudFormation 템플릿에 Amazon ECS 서비스 및 로드 밸런서와 같이 배포를 모델링하는 리소스를 포함해야 합니다. 이러한 리소스 의미에 대한 자세한 내용을 알아보려면 AWS CodeDeploy 사용 설명서의 [Amazon ECS 배포를 시작하기 전에](#)를 참조하세요.

요구 사항	리소스	필수/선택	바뀌는 경우 블루/그린 배포 개시?
Amazon ECS 클러스터	AWS::ECS::Cluster	선택 사항. 기본 클러스터를 사용할 수 있습니다.	아니요

요구 사항	리소스	필수/선택	바뀌는 경우 블루/그린 배포 개시?
Amazon ECS 서비스	AWS::ECS::Service	필수 사항입니다.	아니요
애플리케이션 또는 Network Load Balancer	AWS::ECS::Service LoadBalancer	필수 사항입니다.	아니요
프로덕션 리스너	AWS::ElasticLoadBalancingV2::Listener	필수 사항입니다.	아니요
테스트 리스너	AWS::ElasticLoadBalancingV2::Listener	선택 사항.	아니요
대상 그룹 두 개	AWS::ElasticLoadBalancingV2::TargetGroup	필수 사항입니다.	아니요
Amazon ECS 작업 정의	AWS::ECS::TaskDefinition	필수 사항입니다.	예
Amazon ECS 애플리케이션의 컨테이너	AWS::ECS::TaskDefinition Container Definition Name	필수 사항입니다.	아니요
대체 작업 세트의 포트	AWS::ECS::TaskDefinition PortMapping ContainerPort	필수 사항입니다.	아니요

변경 세트

그린 배포를 시작할 스택 업데이트를 수행하기 전에 변경 세트를 생성하는 것이 좋습니다. 이렇게 하면 스택 업데이트를 수행하기 전에 스택에 이루어질 실제 변경 사항을 확인할 수 있습니다. 리소스 변경 사항은 스택 업데이트 중에 수행되는 순서대로 나열되지 않을 수 있습니다. 자세한 내용은 [변경 세트를 사용하여 CloudFormation 스택 업데이트](#) 섹션을 참조하세요.

스택 이벤트 모니터링

Stack(스택) 페이지의 Events(이벤트) 탭에서 AWS CLI를 사용하여 ECS 배포의 각 단계에서 생성된 스택 이벤트를 볼 수 있습니다. 자세한 내용은 [스택 진행 상황 모니터링](#) 섹션을 참조하세요.

블루/그린 배포를 위한 IAM 권한

CloudFormation에서 블루-그린 배포를 성공적으로 수행하려면 다음 CodeDeploy 권한이 있어야 합니다.

- `codedeploy:Get*`
- `codedeploy:CreateCloudFormationDeployment`

자세한 내용은 서비스 권한 부여 참조에서 [CodeDeploy에 사용되는 작업, 리소스 및 조건 키](#)를 참조하세요.

CloudFormation을 사용하여 ECS 블루/그린 배포를 관리할 때 고려 사항

CloudFormation을 사용하여 CodeDeploy를 통해 ECS 블루/그린 배포를 수행하는 프로세스는 CodeDeploy만 사용하는 표준 ECS 배포와는 다릅니다. 차이점을 자세히 알아보려면 AWS CodeDeploy 사용 설명서의 [CodeDeploy와 CloudFormation을 통한 Amazon ECS 블루/그린 배포 간 차이점](#)을 참조하세요.

CloudFormation을 사용하여 블루/그린 배포를 관리할 때는 다음과 같은 특정 제한 사항 및 고려 사항을 염두에 두어야 합니다.

- 특정 리소스에 대한 업데이트만 그린 배포를 트리거합니다. 자세한 내용은 [그린 배포를 개시하는 리소스 업데이트](#) 섹션을 참조하세요.
- 그린 배포를 트리거하는 리소스에 대한 업데이트와 다른 리소스에 대한 업데이트를 동일한 스택 업데이트에 포함할 수 없습니다. 자세한 내용은 [그린 배포를 개시하는 리소스 업데이트](#) 섹션을 참조하세요.
- 하나의 ECS 서비스만 배포 대상으로 지정할 수 있습니다.
- CloudFormation에 의해 난독화된 값이 있는 파라미터는 그린 배포 중에 CodeDeploy에서 업데이트할 수 없으며 오류 및 스택 업데이트 실패로 이어질 수 있습니다. 다음이 포함됩니다.
 - NoEcho 속성으로 정의된 파라미터
 - 동적 참조를 사용하여 외부 서비스에서 해당 값을 검색하는 파라미터 동적 참조에 대한 자세한 내용은 [동적 참조를 사용하여 다른 서비스에 저장된 값 가져오기](#)의 내용을 참조하세요.

- 아직 진행 중인 그린 배포를 취소하려면 CodeDeploy 또는 ECS가 아니라 CloudFormation에서 스택 업데이트를 취소하세요. 자세한 내용은 [스택 업데이트 취소](#) 섹션을 참조하세요. 업데이트가 완료된 후에는 업데이트를 취소할 수 없습니다. 하지만 이전 설정을 사용하여 스택을 다시 업데이트할 수 있습니다.
- 다음 CloudFormation 기능은 현재 ECS 블루/그린 배포를 정의하는 템플릿에서 지원되지 않습니다.
 - [출력](#) 선언 또는 [Fn::ImportValue](#) 사용으로 다른 스택에서 값 가져오기.
 - 리소스 가져오기. 리소스 가져오기에 대한 자세한 내용은 [CloudFormation 스택에 AWS 리소스 가져오기](#)의 내용을 참조하세요.
 - 중첩 스택 리소스가 포함된 템플릿에 AWS::CodeDeploy::BlueGreen 후크 사용하기. 중첩 스택에 대한 자세한 내용은 [중첩 스택을 사용하여 템플릿을 재사용 가능한 조각으로 분할](#) 섹션을 참조하세요.
 - 중첩 스택에서 AWS::CodeDeploy::BlueGreen 후크 사용하기.

AWS::CodeDeploy::BlueGreen 후크 구문

다음 구문은 ECS 블루/그린 배포를 위한 AWS::CodeDeploy::BlueGreen 후크 구조를 설명합니다.

구문

```
"Hooks": {
  "Logical ID": {
    "Type": "AWS::CodeDeploy::BlueGreen",
    "Properties": {
      "TrafficRoutingConfig": {
        "Type": "Traffic routing type",
        "TimeBasedCanary": {
          "StepPercentage": Integer,
          "BakeTimeMins": Integer
        },
        "TimeBasedLinear": {
          "StepPercentage": Integer,
          "BakeTimeMins": Integer
        }
      },
      "AdditionalOptions": {"TerminationWaitTimeInMinutes": Integer},
      "LifecycleEventHooks": {
        "BeforeInstall": "FunctionName",
        "AfterInstall": "FunctionName",
        "AfterAllowTestTraffic": "FunctionName",
```

```

    "BeforeAllowTraffic": "FunctionName",
    "AfterAllowTraffic": "FunctionName"
  },
  "ServiceRole": "CodeDeployServiceRoleName",
  "Applications": [
    {
      "Target": {
        "Type": "AWS::ECS::Service",
        "LogicalID": "Logical ID of AWS::ECS::Service"
      },
      "ECSAttributes": {
        "TaskDefinitions": [
          "Logical ID of AWS::ECS::TaskDefinition (Blue)",
          "Logical ID of AWS::ECS::TaskDefinition (Green)"
        ],
        "TaskSets": [
          "Logical ID of AWS::ECS::TaskSet (Blue)",
          "Logical ID of AWS::ECS::TaskSet (Green)"
        ],
        "TrafficRouting": {
          "ProdTrafficRoute": {
            "Type": "AWS::ElasticLoadBalancingV2::Listener",
            "LogicalID": "Logical ID of AWS::ElasticLoadBalancingV2::Listener  
(Production)"
          },
          "TestTrafficRoute": {
            "Type": "AWS::ElasticLoadBalancingV2::Listener",
            "LogicalID": "Logical ID of AWS::ElasticLoadBalancingV2::Listener  
(Test)"
          },
          "TargetGroups": [
            "Logical ID of AWS::ElasticLoadBalancingV2::TargetGroup (Blue)",
            "Logical ID of AWS::ElasticLoadBalancingV2::TargetGroup (Green)"
          ]
        }
      }
    }
  ]
}

```

속성

논리적 ID(논리명이라고도 함)

템플릿의 Hooks 섹션에서 선언된 후크의 논리적 ID입니다. 논리적 ID는 영숫자(A-Za-z0-9)여야 하며 템플릿 내에서 고유해야 합니다.

필수 항목 여부: 예

Type

후크의 유형입니다. `AWS::CodeDeploy::BlueGreen`

필수 항목 여부: 예

Properties

후크의 속성

필수 항목 여부: 예

TrafficRoutingConfig

트래픽 라우팅 구성 설정

필수 항목 여부: 아니요

기본 구성은 시간 기반 카나리 트래픽 이동으로 15% 단계 비율 및 5분 베이크 시간입니다.

Type

배포 구성에 사용되는 트래픽 이동 유형

유효한 값: `AllAtOnce` | `TimeBasedCanary` | `TimeBasedLinear`

필수 항목 여부: 예

TimeBasedCanary

배포의 한 버전에서 다른 버전으로 트래픽을 2씩 증분하여 이동하는 구성을 지정합니다.

필수: 조건부: 트래픽 라우팅 유형으로 `TimeBasedCanary`를 지정하는 경우 `TimeBasedCanary` 파라미터를 포함시켜야 합니다.

StepPercentage

`TimeBasedCanary` 배포의 첫 번째 증분에서 이동할 트래픽의 비율입니다. 단계 비율은 14% 이상이어야 합니다.

필수 항목 여부: 아니요

BakeTimeMins

TimeBasedCanary 배포의 첫 번째와 두 번째 트래픽 이동 사이의 시간(분)입니다.

필수 항목 여부: 아니요

TimeBasedLinear

동일한 증분(각 증분 사이에 동일한 시간(분) 지정)을 사용하여 배포의 한 버전에서 다른 버전으로 트래픽을 이동하는 구성을 지정합니다.

필수: 조건부: 트래픽 라우팅 유형으로 TimeBasedLinear를 지정하는 경우 TimeBasedLinear 파라미터를 포함시켜야 합니다.

StepPercentage

TimeBasedLinear 배포의 각 증분이 시작될 때 이동되는 트래픽의 비율. 단계 비율은 14% 이상이어야 합니다.

필수 항목 여부: 아니요

BakeTimeMins

TimeBasedLinear 배포의 각 증분 트래픽 이동 사이의 시간(분)

필수 항목 여부: 아니요

AdditionalOptions

블루/그린 배포를 위한 추가 옵션

필수 항목 여부: 아니요

TerminationWaitTimeInMinutes

블루 리소스를 종료하기 전에 대기할 시간(분)을 지정합니다.

필수 항목 여부: 아니요

LifecycleEventHooks

수명 주기 이벤트 후크를 사용하여 CodeDeploy가 배포를 검증하기 위해 호출할 수 있는 Lambda 함수를 지정합니다. 배포 수명 주기 이벤트에 동일한 함수 또는 다른 함수를 사용할 수 있습니다. 검증 테스트가 완료되면 Lambda AfterAllowTraffic 함수는 CodeDeploy

를 다시 호출하고 Succeeded 또는 Failed 결과를 전달합니다. 자세한 내용을 알아보려면 AWS CodeDeploy 사용 설명서의 [AppSpec '후크' 섹션](#)을 참조하세요.

필수 항목 여부: 아니요

BeforeInstall

대체 작업 세트가 생성되기 전에 작업을 실행할 때 사용하는 함수

필수 항목 여부: 아니요

AfterInstall

대체 작업 세트가 생성되었고 대상 그룹 중 하나가 연결된 후 작업을 실행할 때 사용하는 함수

필수 항목 여부: 아니요

AfterAllowTestTraffic

테스트 리스너가 대체 작업 세트에 트래픽을 제공한 후 작업을 실행할 때 사용하는 함수

필수 항목 여부: 아니요

BeforeAllowTraffic

트래픽이 대체 작업 세트로 이동되기 전에 두 번째 대상 그룹이 대체 작업 세트와 연결된 후 작업을 실행할 때 사용하는 함수

필수 항목 여부: 아니요

AfterAllowTraffic

두 번째 대상 그룹이 대체 작업 세트에 트래픽을 제공한 후 작업을 실행할 때 사용하는 함수

필수 항목 여부: 아니요

ServiceRole

블루 그린 배포를 수행할 때 사용하는 CloudFormation에 대한 실행 역할. 필요한 권한의 목록은 [블루/그린 배포를 위한 IAM 권한](#) 섹션을 참조하세요.

필수 항목 여부: 아니요

Applications

Amazon ECS 애플리케이션의 속성을 지정합니다.

필수 항목 여부: 예

Target

필수 항목 여부: 예

Type

리소스의 유형.

필수 항목 여부: 예

LogicalID

리소스의 논리적 ID

필수 항목 여부: 예

ECSAttributes

Amazon ECS 애플리케이션 배포의 다양한 요구 사항을 나타내는 리소스

필수 항목 여부: 예

TaskDefinitions

Amazon ECS 애플리케이션이 포함된 Docker 컨테이너를 실행하기 위한
[AWS::ECS::TaskDefinition](#) 리소스의 논리적 ID

필수 항목 여부: 예

TaskSets

애플리케이션의 작업 세트로 사용할 [AWS::ECS::TaskSet](#) 리소스의 논리적 ID

필수 항목 여부: 예

TrafficRouting

트래픽 라우팅에 사용되는 리소스를 지정합니다.

필수 항목 여부: 예

ProdTrafficRoute

대상 그룹으로 트래픽을 보내는 데 로드 밸런서에서 사용할 리스너

필수 항목 여부: 예

Type

리소스의 유형입니다. `AWS::ElasticLoadBalancingV2::Listener`

필수 항목 여부: 예

LogicalID

리소스의 논리적 ID

필수 항목 여부: 예

TestTrafficRoute

대상 그룹으로 트래픽을 보내는 데 로드 밸런서에서 사용할 리스너

필수 항목 여부: 예

Type

리소스의 유형입니다. `AWS::ElasticLoadBalancingV2::Listener`

필수 항목 여부: 예

LogicalID

리소스의 논리적 ID

필수 항목 여부: 아니요

TargetGroups

트래픽을 등록된 대상으로 라우팅하기 위해 대상 그룹으로 사용할 리소스의 논리적 ID

필수 항목 여부: 예

블루/그린 배포 템플릿 예제

다음 예제 템플릿에서는 ECS에서의 CodeDeploy 블루/그린 배포를 설정합니다. 여기에서 단계당 트래픽 라우팅 진행률은 15%이고 각 단계 사이의 안정화 기간은 5분입니다.

템플릿을 이용해 스택을 생성하면 배포의 초기 구성이 프로비저닝됩니다. 따라서 `BlueTaskSet` 리소스(대체해야 하는 리소스)의 속성을 변경한 경우 CloudFormation은 스택 업데이트의 일부로 그린 배포를 시작합니다.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "Vpc": { "Type": "AWS::EC2::VPC::Id" },
    "Subnet1": { "Type": "AWS::EC2::Subnet::Id" },
    "Subnet2": { "Type": "AWS::EC2::Subnet::Id" }
  },
  "Transform": [ "AWS::CodeDeployBlueGreen" ],
  "Hooks": {
    "CodeDeployBlueGreenHook": {
      "Type": "AWS::CodeDeploy::BlueGreen",
      "Properties": {
        "TrafficRoutingConfig": {
          "Type": "TimeBasedCanary",
          "TimeBasedCanary": {
            "StepPercentage": 15,
            "BakeTimeMins": 5
          }
        }
      },
      "Applications": [
        {
          "Target": {
            "Type": "AWS::ECS::Service",
            "LogicalID": "ECSDemoService"
          },
          "ECSAttributes": {
            "TaskDefinitions": [ "BlueTaskDefinition", "GreenTaskDefinition" ],
            "TaskSets": [ "BlueTaskSet", "GreenTaskSet" ],
            "TrafficRouting": {
              "ProdTrafficRoute": {
                "Type": "AWS::ElasticLoadBalancingV2::Listener",
                "LogicalID": "ALBListenerProdTraffic"
              },
              "TargetGroups": [ "ALBTargetGroupBlue", "ALBTargetGroupGreen" ]
            }
          }
        }
      ]
    }
  },
  "Resources": {
```

```

"ExampleSecurityGroup":{
  "Type":"AWS::EC2::SecurityGroup",
  "Properties":{
    "GroupDescription":"Security group for ec2 access",
    "VpcId":{"Ref":"Vpc" },
    "SecurityGroupIngress":[
      {
        "IpProtocol":"tcp",
        "FromPort":80,
        "ToPort":80,
        "CidrIp":"0.0.0.0/0"
      },
      {
        "IpProtocol":"tcp",
        "FromPort":8080,
        "ToPort":8080,
        "CidrIp":"0.0.0.0/0"
      },
      {
        "IpProtocol":"tcp",
        "FromPort":22,
        "ToPort":22,
        "CidrIp":"0.0.0.0/0"
      }
    ]
  }
},
"ALBTargetGroupBlue":{
  "Type":"AWS::ElasticLoadBalancingV2::TargetGroup",
  "Properties":{
    "HealthCheckIntervalSeconds":5,
    "HealthCheckPath":"/",
    "HealthCheckPort":"80",
    "HealthCheckProtocol":"HTTP",
    "HealthCheckTimeoutSeconds":2,
    "HealthyThresholdCount":2,
    "Matcher":{"HttpCode":"200" },
    "Port":80,
    "Protocol":"HTTP",
    "Tags":[{"Key":"Group","Value":"Example" }],
    "TargetType":"ip",
    "UnhealthyThresholdCount":4,
    "VpcId":{"Ref":"Vpc" }
  }
}

```

```

},
"ALBTargetGroupGreen":{
  "Type":"AWS::ElasticLoadBalancingV2::TargetGroup",
  "Properties":{
    "HealthCheckIntervalSeconds":5,
    "HealthCheckPath":"/",
    "HealthCheckPort":"80",
    "HealthCheckProtocol":"HTTP",
    "HealthCheckTimeoutSeconds":2,
    "HealthyThresholdCount":2,
    "Matcher":{ "HttpCode":"200" },
    "Port":80,
    "Protocol":"HTTP",
    "Tags":[{ "Key":"Group","Value":"Example" }],
    "TargetType":"ip",
    "UnhealthyThresholdCount":4,
    "VpcId":{ "Ref":"Vpc" }
  }
},
"ExampleALB":{
  "Type":"AWS::ElasticLoadBalancingV2::LoadBalancer",
  "Properties":{
    "Scheme":"internet-facing",
    "SecurityGroups":[{ "Ref":"ExampleSecurityGroup" }],
    "Subnets":[{ "Ref":"Subnet1" },{ "Ref":"Subnet2" }],
    "Tags":[{ "Key":"Group","Value":"Example" }],
    "Type":"application",
    "IpAddressType":"ipv4"
  }
},
"ALBListenerProdTraffic":{
  "Type":"AWS::ElasticLoadBalancingV2::Listener",
  "Properties":{
    "DefaultActions":[
      {
        "Type":"forward",
        "ForwardConfig":{
          "TargetGroups":[
            {
              "TargetGroupArn":{ "Ref":"ALBTargetGroupBlue" },
              "Weight":1
            }
          ]
        }
      }
    ]
  }
}

```

```

    }
  ],
  "LoadBalancerArn":{ "Ref":"ExampleALB" },
  "Port":80,
  "Protocol":"HTTP"
}
},
"ALBListenerProdRule":{
  "Type":"AWS::ElasticLoadBalancingV2::ListenerRule",
  "Properties":{
    "Actions":[
      {
        "Type":"forward",
        "ForwardConfig":{
          "TargetGroups":[
            {
              "TargetGroupArn":{ "Ref":"ALBTargetGroupBlue" },
              "Weight":1
            }
          ]
        }
      }
    ],
    "Conditions":[
      {
        "Field":"http-header",
        "HttpHeaderConfig":{
          "HttpHeaderName":"User-Agent",
          "Values":[ "Mozilla" ]
        }
      }
    ]
  },
  "ListenerArn":{ "Ref":"ALBListenerProdTraffic" },
  "Priority":1
}
},
"ECSTaskExecutionRole":{
  "Type":"AWS::IAM::Role",
  "Properties":{
    "AssumeRolePolicyDocument":{
      "Version": "2012-10-17",
      "Statement":[
        {
          "Sid":"","

```

```

        "Effect": "Allow",
        "Principal": {
            "Service": "ecs-tasks.amazonaws.com"
        },
        "Action": "sts:AssumeRole"
    }
]
},
"ManagedPolicyArns": [ "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" ]
},
"BlueTaskDefinition": {
    "Type": "AWS::ECS::TaskDefinition",
    "Properties": {
        "ExecutionRoleArn": {
            "Fn::GetAtt": [ "ECSTaskExecutionRole", "Arn" ]
        },
        "ContainerDefinitions": [
            {
                "Name": "DemoApp",
                "Image": "nginxdemos/hello:latest",
                "Essential": true,
                "PortMappings": [
                    {
                        "HostPort": 80,
                        "Protocol": "tcp",
                        "ContainerPort": 80
                    }
                ]
            }
        ]
    }
},
"RequiresCompatibilities": [ "FARGATE" ],
"NetworkMode": "awsvpc",
"Cpu": "256",
"Memory": "512",
"Family": "ecs-demo"
},
"ECSDemoCluster": {
    "Type": "AWS::ECS::Cluster",
    "Properties": {}
},
"ECSDemoService": {

```

```

    "Type": "AWS::ECS::Service",
    "Properties": {
      "Cluster": { "Ref": "ECSDemoCluster" },
      "DesiredCount": 1,
      "DeploymentController": { "Type": "EXTERNAL" }
    }
  },
  "BlueTaskSet": {
    "Type": "AWS::ECS::TaskSet",
    "Properties": {
      "Cluster": { "Ref": "ECSDemoCluster" },
      "LaunchType": "FARGATE",
      "NetworkConfiguration": {
        "AwsVpcConfiguration": {
          "AssignPublicIp": "ENABLED",
          "SecurityGroups": [ { "Ref": "ExampleSecurityGroup" } ],
          "Subnets": [ { "Ref": "Subnet1" }, { "Ref": "Subnet2" } ]
        }
      }
    }
  },
  "PlatformVersion": "1.4.0",
  "Scale": {
    "Unit": "PERCENT",
    "Value": 100
  },
  "Service": { "Ref": "ECSDemoService" },
  "TaskDefinition": { "Ref": "BlueTaskDefinition" },
  "LoadBalancers": [
    {
      "ContainerName": "DemoApp",
      "ContainerPort": 80,
      "TargetGroupArn": { "Ref": "ALBTargetGroupBlue" }
    }
  ]
},
"PrimaryTaskSet": {
  "Type": "AWS::ECS::PrimaryTaskSet",
  "Properties": {
    "Cluster": { "Ref": "ECSDemoCluster" },
    "Service": { "Ref": "ECSDemoService" },
    "TaskSetId": { "Fn::GetAtt": [ "BlueTaskSet", "Id" ] }
  }
}
}

```

```
}
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Parameters:
  Vpc:
    Type: AWS::EC2::VPC::Id
  Subnet1:
    Type: AWS::EC2::Subnet::Id
  Subnet2:
    Type: AWS::EC2::Subnet::Id
Transform:
  - 'AWS::CodeDeployBlueGreen'
Hooks:
  CodeDeployBlueGreenHook:
    Type: AWS::CodeDeploy::BlueGreen
    Properties:
      TrafficRoutingConfig:
        Type: TimeBasedCanary
        TimeBasedCanary:
          StepPercentage: 15
          BakeTimeMins: 5
      Applications:
        - Target:
            Type: AWS::ECS::Service
            LogicalID: ECSDemoService
          ECSAttributes:
            TaskDefinitions:
              - BlueTaskDefinition
              - GreenTaskDefinition
            TaskSets:
              - BlueTaskSet
              - GreenTaskSet
            TrafficRouting:
              ProdTrafficRoute:
                Type: AWS::ElasticLoadBalancingV2::Listener
                LogicalID: ALBListenerProdTraffic
              TargetGroups:
                - ALBTargetGroupBlue
                - ALBTargetGroupGreen
Resources:
```



```
ExampleSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Security group for ec2 access
    VpcId: !Ref Vpc
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 80
        ToPort: 80
        CidrIp: 0.0.0.0/0
      - IpProtocol: tcp
        FromPort: 8080
        ToPort: 8080
        CidrIp: 0.0.0.0/0
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: 0.0.0.0/0
ALBTargetGroupBlue:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    HealthCheckIntervalSeconds: 5
    HealthCheckPath: /
    HealthCheckPort: '80'
    HealthCheckProtocol: HTTP
    HealthCheckTimeoutSeconds: 2
    HealthyThresholdCount: 2
    Matcher:
      HttpCode: '200'
    Port: 80
    Protocol: HTTP
    Tags:
      - Key: Group
        Value: Example
    TargetType: ip
    UnhealthyThresholdCount: 4
    VpcId: !Ref Vpc
ALBTargetGroupGreen:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    HealthCheckIntervalSeconds: 5
    HealthCheckPath: /
    HealthCheckPort: '80'
    HealthCheckProtocol: HTTP
```

```
HealthCheckTimeoutSeconds: 2
HealthyThresholdCount: 2
Matcher:
  HttpStatusCode: '200'
Port: 80
Protocol: HTTP
Tags:
  - Key: Group
    Value: Example
TargetType: ip
UnhealthyThresholdCount: 4
VpcId: !Ref Vpc
ExampleALB:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
    Scheme: internet-facing
    SecurityGroups:
      - !Ref ExampleSecurityGroup
    Subnets:
      - !Ref Subnet1
      - !Ref Subnet2
    Tags:
      - Key: Group
        Value: Example
    Type: application
    IpAddressType: ipv4
ALBListenerProdTraffic:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
      - Type: forward
        ForwardConfig:
          TargetGroups:
            - TargetGroupArn: !Ref ALBTargetGroupBlue
              Weight: 1
    LoadBalancerArn: !Ref ExampleALB
    Port: 80
    Protocol: HTTP
ALBListenerProdRule:
  Type: AWS::ElasticLoadBalancingV2::ListenerRule
  Properties:
    Actions:
      - Type: forward
        ForwardConfig:
```

```
    TargetGroups:
      - TargetGroupArn: !Ref ALBTargetGroupBlue
        Weight: 1
    Conditions:
      - Field: http-header
        HttpHeaderConfig:
          HttpHeaderName: User-Agent
          Values:
            - Mozilla
    ListenerArn: !Ref ALBListenerProdTraffic
    Priority: 1
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          - Sid: ''
            Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: 'sts:AssumeRole'
      ManagedPolicyArns:
        - 'arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy'
  BlueTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ExecutionRoleArn: !GetAtt
        - ECSTaskExecutionRole
        - Arn
      ContainerDefinitions:
        - Name: DemoApp
          Image: 'nginxdemos/hello:latest'
          Essential: true
          PortMappings:
            - HostPort: 80
              Protocol: tcp
              ContainerPort: 80
      RequiresCompatibilities:
        - FARGATE
      NetworkMode: awsvpc
      Cpu: '256'
      Memory: '512'
      Family: ecs-demo
```

```
ECSDemoCluster:
  Type: AWS::ECS::Cluster
  Properties: {}
ECSDemoService:
  Type: AWS::ECS::Service
  Properties:
    Cluster: !Ref ECSDemoCluster
    DesiredCount: 1
    DeploymentController:
      Type: EXTERNAL
BlueTaskSet:
  Type: AWS::ECS::TaskSet
  Properties:
    Cluster: !Ref ECSDemoCluster
    LaunchType: FARGATE
    NetworkConfiguration:
      AwsVpcConfiguration:
        AssignPublicIp: ENABLED
        SecurityGroups:
          - !Ref ExampleSecurityGroup
        Subnets:
          - !Ref Subnet1
          - !Ref Subnet2
    PlatformVersion: 1.4.0
    Scale:
      Unit: PERCENT
      Value: 100
    Service: !Ref ECSDemoService
    TaskDefinition: !Ref BlueTaskDefinition
    LoadBalancers:
      - ContainerName: DemoApp
        ContainerPort: 80
        TargetGroupArn: !Ref ALBTargetGroupBlue
PrimaryTaskSet:
  Type: AWS::ECS::PrimaryTaskSet
  Properties:
    Cluster: !Ref ECSDemoCluster
    Service: !Ref ECSDemoService
    TaskSetId: !GetAtt
      - BlueTaskSet
      - Id
```

CloudFormation 템플릿 스니펫

이 단원에는 CloudFormation 템플릿의 여러 부분을 선언하는 방법을 알려 주는 다양한 시나리오 예제가 나와 있습니다. 코드 조각을 출발점으로 하여 사용자 지정 템플릿의 섹션을 구성할 수도 있습니다.

Note

CloudFormation 템플릿은 JSON과 호환되어야 하기 때문에 줄 연속 문자에 대한 프로비저닝은 없습니다. 이 문서에 제시된 조각은 한 줄이 68자보다 긴 경우 임의로 줄 바꿈될 수 있습니다.

주제

- [일반 템플릿 코드 조각](#)
- [Auto Scaling CloudFormation 템플릿 코드 조각](#)
- [AWS 빌링 콘솔 템플릿 코드 조각](#)
- [CloudFormation 템플릿 코드 조각](#)
- [Amazon CloudFront 템플릿 코드 조각](#)
- [Amazon CloudWatch 템플릿 코드 조각](#)
- [Amazon CloudWatch Logs 템플릿 코드 조각](#)
- [Amazon DynamoDB 템플릿 코드 조각](#)
- [Amazon EC2 CloudFormation 템플릿 코드 조각](#)
- [Amazon Elastic Container Service 샘플 템플릿](#)
- [Amazon Elastic File System 샘플 템플릿](#)
- [Elastic Beanstalk 템플릿 코드 조각](#)
- [Elastic Load Balancing 템플릿 코드 조각](#)
- [AWS Identity and Access Management 템플릿 코드 조각](#)
- [AWS Lambda 템플릿](#)
- [Amazon Redshift 템플릿 코드 조각](#)
- [Amazon RDS 템플릿 코드 조각](#)
- [Route 53 템플릿 코드 조각](#)

- [Amazon S3 템플릿 코드 조각](#)
- [Amazon SNS 템플릿 코드 조각](#)
- [Amazon SQS 템플릿 코드 조각](#)
- [Amazon Timestream 템플릿 코드 조각](#)

일반 템플릿 코드 조각

다음은 AWS 서비스에 특정하지 않은 여러 CloudFormation 템플릿의 특징을 보여주는 예제입니다.

주제

- [Base64로 인코딩된 UserData 속성](#)
- [AccessKey 및 SecretKey가 있는 Base64로 인코딩된 UserData 속성](#)
- [리터럴 문자열 파라미터가 하나 있는 Parameters 섹션](#)
- [정규식 제약 조건이 포함된 문자열 파라미터가 있는 Parameters 섹션](#)
- [MinValue 및 MaxValue 제약 조건이 포함된 숫자 파라미터가 있는 Parameters 섹션](#)
- [AllowedValues 제약 조건이 포함된 숫자 파라미터가 있는 Parameters 섹션](#)
- [리터럴 CommaDelimitedList 파라미터가 하나 있는 Parameters 섹션](#)
- [가상 파라미터를 기반으로 한 파라미터 값이 있는 Parameters 섹션](#)
- [매핑이 3개 있는 Mapping 섹션](#)
- [리터럴 문자열 기반 Description](#)
- [리터럴 문자열 출력이 하나 있는 Outputs 섹션](#)
- [리소스 참조 및 가상 참조 출력이 각각 하나씩 있는 Outputs 섹션](#)
- [함수, 리터럴 문자열, 참조 및 가상 파라미터를 기반으로 한 출력이 있는 Outputs 섹션](#)
- [템플릿 포맷 버전](#)
- [AWS Tags 속성](#)

Base64로 인코딩된 UserData 속성

이 예제는 Fn::Base64 및 Fn::Join 함수를 사용하여 UserData 속성의 조합을 보여줍니다. 참조 MyValue 및 MyName은 템플릿의 Parameters 섹션에 정의되어야 하는 파라미터입니다. 리터럴 문자열 Hello World는 이 예제에서 UserData의 일부로 전달하는 또 다른 값입니다.

JSON

```
"UserData" : {
  "Fn::Base64" : {
    "Fn::Join" : [ ",", [
      { "Ref" : "MyValue" },
      { "Ref" : "MyName" },
      "Hello World" ] ]
  }
}
```

YAML

```
UserData:
  Fn::Base64: !Sub |
    Ref: MyValue
    Ref: MyName
    Hello World
```

AccessKey 및 SecretKey가 있는 Base64로 인코딩된 UserData 속성

이 예제는 Fn::Base64 및 Fn::Join 함수를 사용하여 UserData 속성의 조합을 보여줍니다. 여기에는 AccessKey 및 SecretKey 정보가 포함됩니다. 참조 AccessKey 및 SecretKey는 템플릿의 파라미터 섹션에 정의되어야 하는 파라미터입니다.

JSON

```
"UserData" : {
  "Fn::Base64" : {
    "Fn::Join" : [ "", [
      "ACCESS_KEY=", { "Ref" : "AccessKey" },
      "SECRET_KEY=", { "Ref" : "SecretKey" } ] ]
  }
}
```

YAML

```
UserData:
  Fn::Base64: !Sub |
    ACCESS_KEY=${AccessKey}
```

```
SECRET_KEY=${SecretKey}
```

리터럴 문자열 파라미터가 하나 있는 Parameters 섹션

다음 예제에서는 String 유형 파라미터 하나가 선언된 유효한 파라미터 섹션 선언을 보여줍니다.

JSON

```
"Parameters" : {
  "UserName" : {
    "Type" : "String",
    "Default" : "nonadmin",
    "Description" : "Assume a vanilla user if no command-line spec provided"
  }
}
```

YAML

```
Parameters:
  UserName:
    Type: String
    Default: nonadmin
    Description: Assume a vanilla user if no command-line spec provided
```

정규식 제약 조건이 포함된 문자열 파라미터가 있는 Parameters 섹션

다음 예제에서는 String 유형 파라미터 하나가 선언된 유효한 파라미터 섹션 선언을 보여줍니다.

AdminUserAccount 파라미터의 기본값은 admin입니다. 이 파라미터 값은 최소 길이가 1자, 최대 길이가 16자여야 하며 영문자 및 숫자를 포함해야 하지만 영문자로 시작해야 합니다.

JSON

```
"Parameters" : {
  "AdminUserAccount": {
    "Default": "admin",
    "NoEcho": "true",
    "Description" : "The admin account user name",
    "Type": "String",
    "MinLength": "1",
    "MaxLength": "16",
    "AllowedPattern" : "[a-zA-Z][a-zA-Z0-9]*"
```



```
}
}
```

YAML

```
Parameters:
  AdminUserAccount:
    Default: admin
    NoEcho: true
    Description: The admin account user name
    Type: String
    MinLength: 1
    MaxLength: 16
    AllowedPattern: '[a-zA-Z][a-zA-Z0-9]*'
```

MinValue 및 MaxValue 제약 조건이 포함된 숫자 파라미터가 있는 Parameters 섹션

다음 예제에서는 Number 유형 파라미터 하나가 선언된 유효한 파라미터 섹션 선언을 보여줍니다. WebServerPort 파라미터는 기본값이 80이고, 최소값은 1, 최대값은 65,535입니다.

JSON

```
"Parameters" : {
  "WebServerPort": {
    "Default": "80",
    "Description" : "TCP/IP port for the web server",
    "Type": "Number",
    "MinValue": "1",
    "MaxValue": "65535"
  }
}
```

YAML

```
Parameters:
  WebServerPort:
    Default: 80
    Description: TCP/IP port for the web server
    Type: Number
    MinValue: 1
    MaxValue: 65535
```

AllowedValues 제약 조건이 포함된 숫자 파라미터가 있는 Parameters 섹션

다음 예제에서는 Number 유형 파라미터 하나가 선언된 유효한 파라미터 섹션 선언을 보여줍니다. WebServerPort 파라미터는 기본값이 80이고, 80~8,888의 값만 허용합니다.

JSON

```
"Parameters" : {
  "WebServerPortLimited": {
    "Default": "80",
    "Description" : "TCP/IP port for the web server",
    "Type": "Number",
    "AllowedValues" : ["80", "8888"]
  }
}
```

YAML

```
Parameters:
  WebServerPortLimited:
    Default: 80
    Description: TCP/IP port for the web server
    Type: Number
    AllowedValues:
      - 80
      - 8888
```

리터럴 CommaDelimitedList 파라미터가 하나 있는 Parameters 섹션

다음 예제는 단일 CommaDelimitedList 유형 파라미터가 선언된 유효한 Parameters 섹션 선언을 보여줍니다. NoEcho 속성은 TRUE로 설정되어 있으며, 아래 지정된 위치에 저장된 정보를 제외하고 describe-stacks 출력에서 값이 별표(*****)로 마스킹 처리됩니다.

Important

NoEcho 속성을 사용해도 다음에 저장된 정보는 마스킹되지 않습니다.

- Metadata 템플릿 섹션. CloudFormation은 Metadata 섹션에 포함된 정보를 변환, 수정 또는 삭제하지 않습니다. 자세한 내용은 [메타데이터](#)를 참조하십시오.
- Outputs 템플릿 섹션. 자세한 내용은 [출력](#)을 참조하십시오.

- 리소스 정의의 Metadata 속성입니다. 자세한 내용은 [Metadata 속성](#)을 참조하세요.

이러한 메커니즘을 사용하여 암호나 보안 정보와 같은 중요한 정보를 포함하지 않는 것이 좋습니다.

Important

AWS Systems Manager Parameter Store 또는 AWS Secrets Manager와 같이 CloudFormation 외부에서 저장 및 관리되는 중요한 정보를 참조하려면 CloudFormation 템플릿에 직접 중요한 정보를 포함하는 대신 스택 템플릿에 있는 동적 파라미터를 사용하는 것이 좋습니다.

자세한 내용은 [템플릿에 자격 증명을 포함하지 않음](#)을 참조하십시오.

JSON

```
"Parameters" : {
  "UserRoles" : {
    "Type" : "CommaDelimitedList",
    "Default" : "guest,newhire",
    "NoEcho" : "TRUE"
  }
}
```

YAML

```
Parameters:
  UserRoles:
    Type: CommaDelimitedList
    Default: "guest,newhire"
    NoEcho: true
```

가상 파라미터를 기반으로 한 파라미터 값이 있는 Parameters 섹션

다음 예제에서는 가상 파라미터 `AWS::StackName` 및 `AWS::Region`을 사용하는 EC2 사용자 데이터의 명령을 보여줍니다. 가상 parameters에 대한 자세한 내용은 [가상 파라미터를 사용하여 AWS 값 가져오기](#) 단원을 참조하십시오.

JSON

```

    "UserData"      : { "Fn::Base64" : { "Fn::Join" : ["", [
        "#!/bin/bash -xe\n",
        "yum install -y aws-cfn-bootstrap\n",

        "/opt/aws/bin/cfn-init -v ",
        "      --stack ", { "Ref" : "AWS::StackName" },
        "      --resource LaunchConfig ",
        "      --region ", { "Ref" : "AWS::Region" }, "\n",

        "/opt/aws/bin/cfn-signal -e $? ",
        "      --stack ", { "Ref" : "AWS::StackName" },
        "      --resource WebServerGroup ",
        "      --region ", { "Ref" : "AWS::Region" }, "\n"
    ]]]}
  }

```

YAML

```

UserData:
  Fn::Base64: !Sub |
    #!/bin/bash -xe
    yum update -y aws-cfn-bootstrap
    /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource LaunchConfig --
region ${AWS::Region}
    /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource WebServerGroup
--region ${AWS::Region}

```

매핑이 3개 있는 Mapping 섹션

다음 예제는 매핑이 3개 있는 유효한 Mapping 섹션 선언을 보여줍니다. Stop, SlowDown 또는 Go의 매핑 키와 일치하는 경우 맵은 해당하는 RGBColor 속성에 할당된 RGB 값을 제공합니다.

JSON

```

"Mappings" : {
  "LightColor" : {
    "Stop" : {
      "Description" : "red",
      "RGBColor" : "RED 255 GREEN 0 BLUE 0"
    },

```

```

    "SlowDown" : {
      "Description" : "yellow",
      "RGBColor" : "RED 255 GREEN 255 BLUE 0"
    },
    "Go" : {
      "Description" : "green",
      "RGBColor" : "RED 0 GREEN 128 BLUE 0"
    }
  }
}

```

YAML

```

Mappings:
  LightColor:
    Stop:
      Description: red
      RGBColor: "RED 255 GREEN 0 BLUE 0"
    SlowDown:
      Description: yellow
      RGBColor: "RED 255 GREEN 255 BLUE 0"
    Go:
      Description: green
      RGBColor: "RED 0 GREEN 128 BLUE 0"

```

리터럴 문자열 기반 Description

다음 예제는 값이 리터럴 문자열을 기반으로 하는 유효한 Description 섹션 선언을 보여줍니다. 이 코드 조각은 템플릿, 파라미터, 리소스, 속성 또는 출력에 대한 코드 조각일 수 있습니다.

JSON

```

"Description" : "Replace this value"

```

YAML

```

Description: "Replace this value"

```

리터럴 문자열 출력이 하나 있는 Outputs 섹션

이 예제에서는 리터럴 문자열을 기반으로 한 출력 할당을 보여줍니다.

JSON

```
"Outputs" : {
  "MyPhone" : {
    "Value" : "Please call 555-5555",
    "Description" : "A random message for aws cloudformation describe-stacks"
  }
}
```

YAML

```
Outputs:
  MyPhone:
    Value: Please call 555-5555
    Description: A random message for aws cloudformation describe-stacks
```

리소스 참조 및 가상 참조 출력이 각각 하나씩 있는 Outputs 섹션

이 예제는 출력 할당이 2개 있는 Outputs 섹션이 표시됩니다. 출력 할당 하나는 리소스를 기반으로 하고, 다른 하나는 가상 참조를 기반으로 합니다.

JSON

```
"Outputs" : {
  "SNSTopic" : { "Value" : { "Ref" : "MyNotificationTopic" } },
  "StackName" : { "Value" : { "Ref" : "AWS::StackName" } }
}
```

YAML

```
Outputs:
  SNSTopic:
    Value: !Ref MyNotificationTopic
  StackName:
    Value: !Ref AWS::StackName
```

함수, 리터럴 문자열, 참조 및 가상 파라미터를 기반으로 한 출력이 있는 Outputs 섹션

이 예제에서는 출력 할당이 한 개 있는 출력 섹션이 표시됩니다. 퍼센트 기호를 구분 기호로 사용하여 값을 연결하기 위해 Join 함수가 사용됩니다.

JSON

```
"Outputs" : {
  "MyOutput" : {
    "Value" : { "Fn::Join" :
      [ "%", [ "A-string", {"Ref" : "AWS::StackName" } ] ]
    }
  }
}
```

YAML

```
Outputs:
  MyOutput:
    Value: !Join [ %, [ 'A-string', !Ref 'AWS::StackName' ] ]
```

템플릿 포맷 버전

다음 코드 조각은 유효한 `AWSTemplateFormatVersion` 섹션 선언을 보여줍니다.

JSON

```
"AWSTemplateFormatVersion" : "2010-09-09"
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
```

AWS Tags 속성

이 예제는 AWS Tags 속성을 보여줍니다. 이 속성은 리소스의 속성 섹션 내에서 지정합니다. 리소스가 생성되면 선언한 태그로 리소스에 태그가 지정됩니다.

JSON

```
"Tags" : [
  {
    "Key" : "keyname1",
    "Value" : "value1"
  },
  {
```

```
    "Key" : "keyname2",  
    "Value" : "value2"  
  }  
]
```

YAML

```
Tags:  
-  
  Key: "keyname1"  
  Value: "value1"  
-  
  Key: "keyname2"  
  Value: "value2"
```

Auto Scaling CloudFormation 템플릿 코드 조각

Amazon EC2 Auto Scaling을 사용하여 조정 정책 또는 예약된 조정을 통해 Amazon EC2 인스턴스를 자동으로 조정할 수 있습니다. Auto Scaling 그룹은 조정 정책, 예약된 작업, 상태 확인, 수명 주기 후크, 로드 밸런싱과 같은 자동 조정 및 플릿 관리 기능을 지원하는 Amazon EC2 인스턴스의 모음입니다.

Application Auto Scaling은 조정 정책 또는 예약된 조정으로 Amazon EC2 이외의 리소스를 자동으로 조정합니다.

CloudFormation 템플릿을 사용하여 Auto Scaling 그룹, 조정 정책, 예약된 작업, 기타 Auto Scaling 리소스를 인프라의 일부로 생성하고 구성할 수 있습니다. 템플릿을 사용하면 반복 가능하고 일관된 방식으로 Auto Scaling 리소스 배포를 쉽게 관리하고 자동화할 수 있습니다.

다음 예시 템플릿 코드 조각은 Amazon EC2 Auto Scaling 및 Application Auto Scaling을 위한 CloudFormation 리소스 또는 구성 요소에 대해 설명합니다. 이러한 코드 조각은 템플릿에 통합되도록 설계되었으며 독립적으로 실행되도록 설계되지 않았습니다.

코드 조각 카테고리

- [CloudFormation을 사용한 Amazon EC2 Auto Scaling 리소스 구성](#)
- [CloudFormation을 사용한 Application Auto Scaling 리소스 구성](#)

CloudFormation을 사용한 Amazon EC2 Auto Scaling 리소스 구성

다음 예에서는 Amazon EC2 Auto Scaling과 함께 사용할 템플릿에 포함할 수 있는 다양한 스니펫을 보여줍니다.

코드 조각 카테고리

- [단일 인스턴스 Auto Scaling 그룹 생성](#)
- [연결된 로드 밸런서를 사용하여 Auto Scaling 그룹 생성](#)
- [알림을 사용하여 Auto Scaling 그룹 생성](#)
- [CreationPolicy 및 UpdatePolicy를 사용하는 Auto Scaling 그룹 생성](#)
- [단계 조정 정책 삭제](#)
- [혼합 인스턴스 그룹 예제](#)
- [시작 구성 예제](#)

단일 인스턴스 Auto Scaling 그룹 생성

이 예에서는 시작하는 데 도움이 되는 단일 인스턴스가 포함된

[AWS::AutoScaling::AutoScalingGroup](#) 리소스를 보여줍니다. auto scaling의 VPCZoneIdentifier 속성은 3개의 서로 다른 가용 영역에 있는 기존 서브넷 목록을 지정합니다. 스택을 만들기 전에 계정에서 적용 가능한 서브넷 ID를 지정해야 합니다. LaunchTemplate 속성은 템플릿의 다른 곳에 정의된 논리명이 myLaunchTemplate인 [AWS::EC2::LaunchTemplate](#) 리소스를 참조합니다.

Note

시작 템플릿의 예는 Amazon EC2 코드 조각 섹션의 [CloudFormation을 사용하여 시작 템플릿 생성](#) 및 AWS::EC2::LaunchTemplate 리소스의 [예](#) 섹션을 참조하세요.

JSON

```
"myASG" : {
  "Type" : "AWS::AutoScaling::AutoScalingGroup",
  "Properties" : {
    "VPCZoneIdentifier" : [ "subnetIdAz1", "subnetIdAz2", "subnetIdAz3" ],
    "LaunchTemplate" : {
      "LaunchTemplateId" : {
        "Ref" : "myLaunchTemplate"
      },
    },
    "Version" : {
      "Fn::GetAtt" : [
```

```

        "myLaunchTemplate",
        "LatestVersionNumber"
    ]
}
},
"MaxSize" : "1",
"MinSize" : "1"
}
}

```

YAML

```

myASG:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    VPCZoneIdentifier:
      - subnetIdAz1
      - subnetIdAz2
      - subnetIdAz3
    LaunchTemplate:
      LaunchTemplateId: !Ref myLaunchTemplate
      Version: !GetAtt myLaunchTemplate.LatestVersionNumber
    MaxSize: '1'
    MinSize: '1'

```

연결된 로드 밸런서를 사용하여 Auto Scaling 그룹 생성

이 예에서는 여러 서버에 대한 로드 밸런싱을 위한 [AWS::AutoScaling::AutoScalingGroup](#) 리소스를 보여줍니다. 동일한 템플릿의 다른 곳에서 선언된 AWS 리소스의 논리적 이름을 지정합니다.

1. VPCZoneIdentifier 속성은 Auto Scaling 그룹의 EC2 인스턴스가 생성될 두 [AWS::EC2::Subnet](#) 리소스의 논리명인 myPublicSubnet1 및 myPublicSubnet2를 지정합니다.
2. LaunchTemplate 속성은 논리명이 myLaunchTemplate인 [AWS::EC2::LaunchTemplate](#) 리소스를 지정합니다.
3. TargetGroupARNs 속성은 트래픽을 Auto Scaling 그룹으로 라우팅하는 데 사용되는 Application Load Balancer 또는 Network Load Balancer의 대상 그룹을 나열합니다. 이 예에서는 논리명이 myTargetGroup인 [AWS::ElasticLoadBalancingV2::TargetGroup](#) 리소스에 하나의 대상 그룹이 지정됩니다.

JSON

```

"myServerGroup" : {
  "Type" : "AWS::AutoScaling::AutoScalingGroup",
  "Properties" : {
    "VPCZoneIdentifier" : [ { "Ref" : "myPublicSubnet1" }, { "Ref" :
"myPublicSubnet2" } ],
    "LaunchTemplate" : {
      "LaunchTemplateId" : {
        "Ref" : "myLaunchTemplate"
      },
      "Version" : {
        "Fn::GetAtt" : [
          "myLaunchTemplate",
          "LatestVersionNumber"
        ]
      }
    },
    "MaxSize" : "5",
    "MinSize" : "1",
    "TargetGroupARNs" : [ { "Ref" : "myTargetGroup" } ]
  }
}

```

YAML

```

myServerGroup:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    VPCZoneIdentifier:
      - !Ref myPublicSubnet1
      - !Ref myPublicSubnet2
    LaunchTemplate:
      LaunchTemplateId: !Ref myLaunchTemplate
      Version: !GetAtt myLaunchTemplate.LatestVersionNumber
    MaxSize: '5'
    MinSize: '1'
    TargetGroupARNs:
      - !Ref myTargetGroup

```

다음 사항도 참조하세요.

Application Load Balancer에 대한 ALBRequestCountPerTarget 사전 정의 지표에 따라 대상 추적 조정 정책을 사용하여 auto scaling을 생성하는 자세한 예제는 [AWS::AutoScaling::ScalingPolicy](#) 리소스의 [예제](#) 섹션을 참조하세요.

알림을 사용하여 Auto Scaling 그룹 생성

이 예에서는 지정된 이벤트가 발생하면 Amazon SNS 알림을 보내는 [AWS::AutoScaling::AutoScalingGroup](#) 리소스를 보여줍니다.

NotificationConfigurations 속성은 CloudFormation에서 알림을 보내는 SNS 주제와 CloudFormation에서 알림을 보내는 원인 이벤트를 지정합니다. NotificationTypes에 지정된 이벤트가 발생하면 CloudFormation에서 TopicARN에 지정된 SNS 주제에 알림을 보냅니다. 스택을 시작하면 CloudFormation에서 동일한 템플릿 내에 선언된 [AWS::SNS::Subscription](#) 리소스 (snsTopicForAutoScalingGroup)를 생성합니다.

auto scaling의 VPCZoneIdentifier 속성은 3개의 서로 다른 가용 영역에 있는 기존 서브넷 목록을 지정합니다. 스택을 만들기 전에 계정에서 적용 가능한 서브넷 ID를 지정해야 합니다.

LaunchTemplate 속성은 동일한 템플릿의 다른 곳에서 선언된 [AWS::EC2::LaunchTemplate](#) 리소스의 논리명을 참조합니다.

JSON

```
"myASG" : {
  "Type" : "AWS::AutoScaling::AutoScalingGroup",
  "DependsOn": [
    "snsTopicForAutoScalingGroup"
  ],
  "Properties" : {
    "VPCZoneIdentifier" : [ "subnetIdAz1", "subnetIdAz2", "subnetIdAz3" ],
    "LaunchTemplate" : {
      "LaunchTemplateId" : {
        "Ref" : "logicalName"
      },
      "Version" : {
        "Fn::GetAtt" : [
          "logicalName",
          "LatestVersionNumber"
        ]
      }
    }
  }
},
```

```

    "MaxSize" : "5",
    "MinSize" : "1",
    "NotificationConfigurations" : [
      {
        "TopicARN" : { "Ref" : "snsTopicForAutoScalingGroup" },
        "NotificationTypes" : [
          "autoscaling:EC2_INSTANCE_LAUNCH",
          "autoscaling:EC2_INSTANCE_LAUNCH_ERROR",
          "autoscaling:EC2_INSTANCE_TERMINATE",
          "autoscaling:EC2_INSTANCE_TERMINATE_ERROR",
          "autoscaling:TEST_NOTIFICATION"
        ]
      }
    ]
  }
}

```

YAML

```

myASG:
  Type: AWS::AutoScaling::AutoScalingGroup
  DependsOn:
    - snsTopicForAutoScalingGroup
  Properties:
    VPCZoneIdentifier:
      - subnetIdAz1
      - subnetIdAz2
      - subnetIdAz3
    LaunchTemplate:
      LaunchTemplateId: !Ref logicalName
      Version: !GetAtt logicalName.LatestVersionNumber
    MaxSize: '5'
    MinSize: '1'
    NotificationConfigurations:
      - TopicARN: !Ref snsTopicForAutoScalingGroup
        NotificationTypes:
          - autoscaling:EC2_INSTANCE_LAUNCH
          - autoscaling:EC2_INSTANCE_LAUNCH_ERROR
          - autoscaling:EC2_INSTANCE_TERMINATE
          - autoscaling:EC2_INSTANCE_TERMINATE_ERROR
          - autoscaling:TEST_NOTIFICATION

```

CreationPolicy 및 UpdatePolicy를 사용하는 Auto Scaling 그룹 생성

다음 예는 [AWS::AutoScaling::AutoScalingGroup](#) 리소스에 CreationPolicy 및 UpdatePolicy 속성을 추가하는 방법을 보여줍니다.

샘플 생성 정책은 auto scaling이 준비가 되면 CloudFormation이 Count 개수의 성공 신호를 받을 때까지 CREATE_COMPLETE 상태가 되지 않도록 합니다. auto scaling이 준비되었음을 알리기 위해 시작 템플릿의 사용자 데이터(표시되지 않음)에 추가된 cfn-signal 헬퍼 스크립트가 인스턴스에서 실행됩니다. 인스턴스가 지정된 Timeout 내에 신호를 보내지 않으면 CloudFormation은 인스턴스가 생성되지 않은 것으로 가정하고, 리소스 생성이 실패하고, CloudFormation은 스택을 롤백합니다.

샘플 업데이트 정책은 CloudFormation에 AutoScalingRollingUpdate 속성을 사용하여 롤링 업데이트를 수행하도록 지시합니다. 롤링 업데이트는 MaxBatchSize에 기반한 작은 배치(이 예에서는 인스턴스 단위)의 Auto Scaling 그룹, PauseTime에 기반한 업데이트의 배치 간 일시 중지 시간을 변경합니다. MinInstancesInService 속성은 CloudFormation에서 이전 인스턴스를 업데이트하는 동안 Auto Scaling 그룹 내에서 작동해야 하는 최소 인스턴스 수를 지정합니다.

WaitOnResourceSignals 속성이 true로 설정되어 있습니다. 업데이트를 계속 진행하려면 CloudFormation이 지정된 PauseTime 내에 각각의 새 인스턴스로부터 신호를 수신해야 합니다. 스택 업데이트가 진행되는 동안 HealthCheck, ReplaceUnhealthy, AZRebalance, AlarmNotification 및 ScheduledActions 같은 EC2 Auto Scaling 프로세스가 일시 중단됩니다. 참고: 이 경우 롤링 업데이트가 제대로 작동하지 않을 수 있으므로 Launch, Terminate 또는 AddToLoadBalancer(Auto Scaling 그룹이 Elastic Load Balancing와 함께 사용되는 경우) 프로세스 유형을 일시 중지하지 마십시오.

auto scaling의 VPCZoneIdentifier 속성은 3개의 서로 다른 가용 영역에 있는 기존 서브넷 목록을 지정합니다. 스택을 만들기 전에 계정에서 적용 가능한 서브넷 ID를 지정해야 합니다.

LaunchTemplate 속성은 동일한 템플릿의 다른 곳에서 선언된 [AWS::EC2::LaunchTemplate](#) 리소스의 논리명을 참조합니다.

CreationPolicy 및 UpdatePolicy 속성에 대한 자세한 내용은 [리소스 속성 참조](#) 섹션을 참조하세요.

JSON

```
{
  "Resources":{
    "myASG":{
      "CreationPolicy":{
        "ResourceSignal":{
          "Count":"3",
```

```

        "Timeout": "PT15M"
    }
},
"UpdatePolicy": {
    "AutoScalingRollingUpdate": {
        "MinInstancesInService": "3",
        "MaxBatchSize": "1",
        "PauseTime": "PT12M5S",
        "WaitOnResourceSignals": "true",
        "SuspendProcesses": [
            "HealthCheck",
            "ReplaceUnhealthy",
            "AZRebalance",
            "AlarmNotification",
            "ScheduledActions",
            "InstanceRefresh"
        ]
    }
},
"Type": "AWS::AutoScaling::AutoScalingGroup",
"Properties": {
    "VPCZoneIdentifier": [ "subnetIdAz1", "subnetIdAz2", "subnetIdAz3" ],
    "LaunchTemplate": {
        "LaunchTemplateId": {
            "Ref": "logicalName"
        },
        "Version": {
            "Fn::GetAtt": [
                "logicalName",
                "LatestVersionNumber"
            ]
        }
    },
    "MaxSize": "5",
    "MinSize": "3"
}
}
}
}

```

YAML

```
---
```

```

Resources:
  myASG:
    CreationPolicy:
      ResourceSignal:
        Count: '3'
        Timeout: PT15M
    UpdatePolicy:
      AutoScalingRollingUpdate:
        MinInstancesInService: '3'
        MaxBatchSize: '1'
        PauseTime: PT12M5S
        WaitOnResourceSignals: true
      SuspendProcesses:
        - HealthCheck
        - ReplaceUnhealthy
        - AZRebalance
        - AlarmNotification
        - ScheduledActions
        - InstanceRefresh
    Type: AWS::AutoScaling::AutoScalingGroup
    Properties:
      VPCZoneIdentifier:
        - subnetIdAz1
        - subnetIdAz2
        - subnetIdAz3
      LaunchTemplate:
        LaunchTemplateId: !Ref logicalName
        Version: !GetAtt logicalName.LatestVersionNumber
      MaxSize: '5'
      MinSize: '3'

```

단계 조정 정책 삭제

이 예에서는 단계 조정 정책을 사용하여 Auto Scaling 그룹을 스케일 아웃하는 [AWS::AutoScaling::ScalingPolicy](#) 리소스를 보여줍니다. AdjustmentType 속성은 ChangeInCapacity를 지정합니다. 이는 ScalingAdjustment가 추가하거나 (ScalingAdjustment가 양수인 경우) 삭제할(음수인 경우) 인스턴스 수를 나타낸다는 것을 의미합니다. 이 예에서 ScalingAdjustment는 1입니다. 따라서 경보 임계값을 초과하면 그룹의 EC2 인스턴스 수가 1씩 증가합니다.

[AWS::CloudWatch::Alarm](#) 리소스 CPUAlarmHigh는 경보가 ALARM 상태일 때 실행할 작업 (AlarmActions)으로 조정 정책 ASGScalingPolicyHigh를 지정합니다. Dimensions 속성은 동

일한 템플릿의 다른 곳에서 선언된 [AWS::AutoScaling::AutoScalingGroup](#) 리소스의 논리명을 참조합니다.

JSON

```
{
  "Resources":{
    "ASGScalingPolicyHigh":{
      "Type":"AWS::AutoScaling::ScalingPolicy",
      "Properties":{
        "AutoScalingGroupName":{"Ref":"logicalName" },
        "PolicyType":"StepScaling",
        "AdjustmentType":"ChangeInCapacity",
        "StepAdjustments":[
          {
            "MetricIntervalLowerBound":0,
            "ScalingAdjustment":1
          }
        ]
      }
    },
    "CPULAlarmHigh":{
      "Type":"AWS::CloudWatch::Alarm",
      "Properties":{
        "EvaluationPeriods":"2",
        "Statistic":"Average",
        "Threshold":"90",
        "AlarmDescription":"Scale out if CPU > 90% for 2 minutes",
        "Period":"60",
        "AlarmActions":[ { "Ref":"ASGScalingPolicyHigh" } ],
        "Namespace":"AWS/EC2",
        "Dimensions":[
          {
            "Name":"AutoScalingGroupName",
            "Value":{"Ref":"logicalName" }
          }
        ],
        "ComparisonOperator":"GreaterThanThreshold",
        "MetricName":"CPUUtilization"
      }
    }
  }
}
```

```
}
```

YAML

```
---
Resources:
  ASGScalingPolicyHigh:
    Type: AWS::AutoScaling::ScalingPolicy
    Properties:
      AutoScalingGroupName: !Ref logicalName
      PolicyType: StepScaling
      AdjustmentType: ChangeInCapacity
      StepAdjustments:
        - MetricIntervalLowerBound: 0
          ScalingAdjustment: 1
  CPUAlarmHigh:
    Type: AWS::CloudWatch::Alarm
    Properties:
      EvaluationPeriods: 2
      Statistic: Average
      Threshold: 90
      AlarmDescription: 'Scale out if CPU > 90% for 2 minutes'
      Period: 60
      AlarmActions:
        - !Ref ASGScalingPolicyHigh
      Namespace: AWS/EC2
      Dimensions:
        - Name: AutoScalingGroupName
          Value:
            !Ref logicalName
      ComparisonOperator: GreaterThanThreshold
      MetricName: CPUUtilization
```

다음 사항도 참조하세요.

조정 정책에 대한 추가 예제 템플릿을 알아보려면 `AWS::AutoScaling::ScalingPolicy` 리소스의 [예제](#) 섹션을 참조하세요.

혼합 인스턴스 그룹 예제

속성 기반 인스턴스 유형 선택을 사용하여 Auto Scaling 그룹 생성

이 예는 속성 기반 인스턴스 유형 선택을 사용하여 혼합 인스턴스 그룹을 시작하기 위한 정보를 포함하는 [AWS::AutoScaling::AutoScalingGroup](#) 리소스를 보여줍니다. VCpuCount 속성의 최솟값과 최댓값을 지정하고 MemoryMiB 속성의 최솟값을 지정합니다. auto scaling에서 사용하는 모든 인스턴스 유형은 필요한 인스턴스 속성과 일치해야 합니다.

auto scaling의 VPCZoneIdentifier 속성은 3개의 서로 다른 가용 영역에 있는 기존 서브넷 목록을 지정합니다. 스택을 만들기 전에 계정에서 적용 가능한 서브넷 ID를 지정해야 합니다.

LaunchTemplate 속성은 동일한 템플릿의 다른 곳에서 선언된 [AWS::EC2::LaunchTemplate](#) 리소스의 논리명을 참조합니다.

JSON

```
{
  "Resources":{
    "myASG":{
      "Type":"AWS::AutoScaling::AutoScalingGroup",
      "Properties":{
        "VPCZoneIdentifier":[
          "subnetIdAz1",
          "subnetIdAz2",
          "subnetIdAz3"
        ],
        "MixedInstancesPolicy":{
          "LaunchTemplate":{
            "LaunchTemplateSpecification":{
              "LaunchTemplateId":{
                "Ref":"logicalName"
              },
              "Version":{
                "Fn::GetAtt":[
                  "logicalName",
                  "LatestVersionNumber"
                ]
              }
            }
          },
          "Overrides":[
            {
              "InstanceRequirements":{
                "VCpuCount":{
```

```

        "Min":2,
        "Max":4
    },
    "MemoryMiB":{
        "Min":2048
    }
}
]
}
},
"MaxSize":"5",
"MinSize":"1"
}
}
}
}
}

```

YAML

```

---
Resources:
  myASG:
    Type: AWS::AutoScaling::AutoScalingGroup
    Properties:
      VPCZoneIdentifier:
        - subnetIdAz1
        - subnetIdAz2
        - subnetIdAz3
      MixedInstancesPolicy:
        LaunchTemplate:
          LaunchTemplateSpecification:
            LaunchTemplateId: !Ref logicalName
            Version: !GetAtt logicalName.LatestVersionNumber
          Overrides:
            - InstanceRequirements:
                VCpuCount:
                  Min: 2
                  Max: 4
                MemoryMiB:
                  Min: 2048
      MaxSize: '5'
      MinSize: '1'

```

시작 구성 예제

시작 구성 생성

이 예에서는 ImageId, InstanceType, SecurityGroups 속성에 대한 값을 지정하는 Auto Scaling 그룹에 대한 [AWS::AutoScaling::LaunchConfiguration](#) 리소스를 보여줍니다. SecurityGroups 속성은 템플릿에 지정된 [AWS::EC2::SecurityGroup](#) 리소스의 논리명과 myExistingEC2SecurityGroup이라는 기존 EC2 보안 그룹을 모두 지정합니다.

JSON

```
"mySimpleConfig" : {
  "Type" : "AWS::AutoScaling::LaunchConfiguration",
  "Properties" : {
    "ImageId" : "ami-02354e95b3example",
    "InstanceType" : "t3.micro",
    "SecurityGroups" : [ { "Ref" : "logicalName" }, "myExistingEC2SecurityGroup" ]
  }
}
```

YAML

```
mySimpleConfig:
  Type: AWS::AutoScaling::LaunchConfiguration
  Properties:
    ImageId: ami-02354e95b3example
    InstanceType: t3.micro
    SecurityGroups:
      - !Ref logicalName
      - myExistingEC2SecurityGroup
```

시작 구성을 사용하는 Auto Scaling 그룹 생성

이 예에서는 단일 인스턴스가 있는 [AWS::AutoScaling::AutoScalingGroup](#) 리소스를 보여줍니다. auto scaling의 VPCZoneIdentifier 속성은 3개의 서로 다른 가용 영역에 있는 기존 서브넷 목록을 지정합니다. 스택을 만들기 전에 계정에서 적용 가능한 서브넷 ID를 지정해야 합니다. LaunchConfigurationName 속성은 템플릿에 정의된 논리명이 mySimpleConfig인 [AWS::AutoScaling::LaunchConfiguration](#) 리소스를 참조합니다.

JSON

```
"myASG" : {
  "Type" : "AWS::AutoScaling::AutoScalingGroup",
  "Properties" : {
    "VPCZoneIdentifier" : [ "subnetIdAz1", "subnetIdAz2", "subnetIdAz3" ],
    "LaunchConfigurationName" : { "Ref" : "mySimpleConfig" },
    "MaxSize" : "1",
    "MinSize" : "1"
  }
}
```

YAML

```
myASG:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    VPCZoneIdentifier:
      - subnetIdAz1
      - subnetIdAz2
      - subnetIdAz3
    LaunchConfigurationName: !Ref mySimpleConfig
    MaxSize: '1'
    MinSize: '1'
```

CloudFormation을 사용한 Application Auto Scaling 리소스 구성

이 섹션에서는 다양한 AWS 리소스의 Application Auto Scaling 조정 정책 및 예약된 작업에 대한 CloudFormation 템플릿 예제를 보여줍니다.

Important

Application Auto Scaling 코드 조각이 템플릿에 포함된 경우 템플릿을 통해 생성되는 조정 가능 리소스에 대해 DependsOn 속성을 사용하여 종속성을 선언해야 할 수 있습니다. 그러면 기본 병렬 처리가 재정의되고 CloudFormation이 지정된 순서로 리소스에서 작동합니다. 이렇게 하지 않으면 리소스가 완전히 설정되기 전에 조정 구성이 적용될 수 있습니다.

자세한 내용은 [DependsOn 속성](#)을 참조하세요.

코드 조각 카테고리

- [AppStream 플릿에 대한 조정 정책 생성](#)
- [Aurora DB 클러스터에 대한 조정 정책 생성](#)
- [DynamoDB 테이블에 대한 조정 정책 생성](#)
- [Amazon ECS 서비스에 대한 조정 정책 생성\(지표: 평균 CPU 및 메모리\)](#)
- [Amazon ECS 서비스에 대한 조정 정책 생성\(지표: 대상당 평균 요청 수\)](#)
- [Lambda 함수에 대해 cron 표현식을 사용하는 예약된 작업 생성](#)
- [스팟 플릿에 대해 at 표현식을 사용하는 예약된 작업 생성](#)

AppStream 플릿에 대한 조정 정책 생성

이 코드 조각은 [AWS::ApplicationAutoScaling::ScalingPolicy](#) 리소스를 사용하여 정책을 생성하고 [AWS::AppStream::Fleet](#) 리소스에 적용하는 방법을 보여줍니다.

[AWS::ApplicationAutoScaling::ScalableTarget](#) 리소스는 해당 정책이 적용되는 조정 가능 대상을 선언합니다. Application Auto Scaling에서는 플릿 인스턴스의 수를 최소 1개와 최대 20개 사이에서 조정할 수 있습니다. 이 정책은 300초(5분)의 스케일 아웃 및 스케일 인 휴지 기간과 함께 플릿의 평균 용량 사용률을 75%로 유지합니다.

Fn::Join 및 Rev 내장 함수를 사용하여 동일한 템플릿에 지정된 AWS::AppStream::Fleet 리소스의 논리명으로 ResourceId 속성을 구성합니다. 자세한 내용은 [내장 함수 참조](#)를 참조하세요.

JSON

```
{
  "Resources" : {
    "ScalableTarget" : {
      "Type" : "AWS::ApplicationAutoScaling::ScalableTarget",
      "Properties" : {
        "MaxCapacity" : 20,
        "MinCapacity" : 1,
        "RoleARN" : { "Fn::Sub" : "arn:aws:iam::${AWS::AccountId}:role/
aws-service-role/appstream.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_AppStreamFleet" },
        "ServiceNamespace" : "appstream",
        "ScalableDimension" : "appstream:fleet:DesiredCapacity",
        "ResourceId" : {
          "Fn::Join" : [
            "/",
            [
              "fleet",
```

```

        {
            "Ref" : "logicalName"
        }
    ]
]
}
},
"ScalingPolicyAppStreamFleet" : {
    "Type" : "AWS::ApplicationAutoScaling::ScalingPolicy",
    "Properties" : {
        "PolicyName" : { "Fn::Sub" : "${AWS::StackName}-target-tracking-cpu75" },
        "PolicyType" : "TargetTrackingScaling",
        "ServiceNamespace" : "appstream",
        "ScalableDimension" : "appstream:fleet:DesiredCapacity",
        "ResourceId" : {
            "Fn::Join" : [
                "/",
                [
                    "fleet",
                    {
                        "Ref" : "logicalName"
                    }
                ]
            ]
        },
        "TargetTrackingScalingPolicyConfiguration" : {
            "TargetValue" : 75,
            "PredefinedMetricSpecification" : {
                "PredefinedMetricType" : "AppStreamAverageCapacityUtilization"
            },
            "ScaleInCooldown" : 300,
            "ScaleOutCooldown" : 300
        }
    }
}
}
}
}
}

```

YAML

```

---
Resources:

```



```

ScalableTarget:
  Type: AWS::ApplicationAutoScaling::ScalableTarget
  Properties:
    MaxCapacity: 20
    MinCapacity: 1
    RoleARN:
      Fn::Sub: 'arn:aws:iam::${AWS::AccountId}:role/aws-
service-role/appstream.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_AppStreamFleet'
    ServiceNamespace: appstream
    ScalableDimension: appstream:fleet:DesiredCapacity
    ResourceId: !Join
      - /
      - - fleet
      - !Ref logicalName
ScalingPolicyAppStreamFleet:
  Type: AWS::ApplicationAutoScaling::ScalingPolicy
  Properties:
    PolicyName: !Sub ${AWS::StackName}-target-tracking-cpu75
    PolicyType: TargetTrackingScaling
    ServiceNamespace: appstream
    ScalableDimension: appstream:fleet:DesiredCapacity
    ResourceId: !Join
      - /
      - - fleet
      - !Ref logicalName
    TargetTrackingScalingPolicyConfiguration:
      TargetValue: 75
      PredefinedMetricSpecification:
        PredefinedMetricType: AppStreamAverageCapacityUtilization
      ScaleInCooldown: 300
      ScaleOutCooldown: 300

```

Aurora DB 클러스터에 대한 조정 정책 생성

이 코드 조각에서는 [AWS::RDS::DBCluster](#) 리소스를 등록합니다.

[AWS::ApplicationAutoScaling::ScalableTarget](#) 리소스는 DB 클러스터가 1개에서 8개까지 Aurora 복제본을 보유하도록 동적으로 조정되어야 함을 나타냅니다. 또한 [AWS::ApplicationAutoScaling::ScalingPolicy](#) 리소스를 사용하여 클러스터에 대상 추적 조정 정책을 적용합니다.

이 구성에서 RDSReaderAverageCPUUtilization 사전 정의 지표는 해당 Aurora DB 클러스터에 있는 모든 Aurora 복제본에 대해 평균 CPU 사용률 40%를 기반으로 Aurora DB 클러스터를 조정합니다. 구성에서는 스케일 인 휴지 기간 10분과 스케일 아웃 휴지 기간 5분을 제공합니다.

이 예에서는 Fn::Sub 내장 함수를 사용하여 동일한 템플릿에 지정된 AWS::RDS::DBCluster 리소스의 논리명으로 ResourceId 속성을 구성합니다. 자세한 내용은 [내장 함수 참조](#)를 참조하세요.

JSON

```
{
  "Resources" : {
    "ScalableTarget" : {
      "Type" : "AWS::ApplicationAutoScaling::ScalableTarget",
      "Properties" : {
        "MaxCapacity" : 8,
        "MinCapacity" : 1,
        "RoleARN" : { "Fn::Sub" : "arn:aws:iam::${AWS::AccountId}:role/
aws-service-role/rds.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_RDSCluster" },
        "ServiceNamespace" : "rds",
        "ScalableDimension" : "rds:cluster:ReadReplicaCount",
        "ResourceId" : { "Fn::Sub" : "cluster:${logicalName}" }
      }
    },
    "ScalingPolicyDBCluster" : {
      "Type" : "AWS::ApplicationAutoScaling::ScalingPolicy",
      "Properties" : {
        "PolicyName" : { "Fn::Sub" : "${AWS::StackName}-target-tracking-cpu40" },
        "PolicyType" : "TargetTrackingScaling",
        "ServiceNamespace" : "rds",
        "ScalableDimension" : "rds:cluster:ReadReplicaCount",
        "ResourceId" : { "Fn::Sub" : "cluster:${logicalName}" },
        "TargetTrackingScalingPolicyConfiguration" : {
          "TargetValue" : 40,
          "PredefinedMetricSpecification" : {
            "PredefinedMetricType" : "RDSReaderAverageCPUUtilization"
          },
          "ScaleInCooldown" : 600,
          "ScaleOutCooldown" : 300
        }
      }
    }
  }
}
```

```
}
```

YAML

```
---
Resources:
  ScalableTarget:
    Type: AWS::ApplicationAutoScaling::ScalableTarget
    Properties:
      MaxCapacity: 8
      MinCapacity: 1
      RoleARN:
        Fn::Sub: 'arn:aws:iam::${AWS::AccountId}:role/aws-service-role/rds.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_RDSCluster'
      ServiceNamespace: rds
      ScalableDimension: rds:cluster:ReadReplicaCount
      ResourceId: !Sub cluster:${logicalName}
  ScalingPolicyDBCluster:
    Type: AWS::ApplicationAutoScaling::ScalingPolicy
    Properties:
      PolicyName: !Sub ${AWS::StackName}-target-tracking-cpu40
      PolicyType: TargetTrackingScaling
      ServiceNamespace: rds
      ScalableDimension: rds:cluster:ReadReplicaCount
      ResourceId: !Sub cluster:${logicalName}
      TargetTrackingScalingPolicyConfiguration:
        TargetValue: 40
        PredefinedMetricSpecification:
          PredefinedMetricType: RDSReaderAverageCPUUtilization
        ScaleInCooldown: 600
        ScaleOutCooldown: 300
```

DynamoDB 테이블에 대한 조정 정책 생성

이 코드 조각은 TargetTrackingScaling 정책 유형으로 정책을 생성하

고 [AWS::ApplicationAutoScaling::ScalingPolicy](#) 리소스를 사

용하는 [AWS::DynamoDB::Table](#) 리소스에 적용하는 방법을 보여줍니다.

[AWS::ApplicationAutoScaling::ScalableTarget](#) 리소스는 최소 5개 및 최대 15개의 쓰기 용량 단위로 이 정책이 적용되는 조정 가능 대상을 선언합니다. 조정 정책은 테이블의 쓰기 용량 처리량을 조정하여 DynamoDBWriteCapacityUtilization 사전 정의 지표에 따라 대상 사용률을 50%로 유지합니다.

`Fn::Join` 및 `Ref` 내장 함수를 사용하여 동일한 템플릿에 지정된 `AWS::DynamoDB::Table` 리소스의 논리명으로 `ResourceId` 속성을 구성합니다. 자세한 내용은 [내장 함수 참조](#)를 참조하세요.

Note

DynamoDB 리소스용 CloudFormation 템플릿을 생성하는 방법에 대한 자세한 내용을 알아보려면 AWS Database 블로그의 [How to use CloudFormation to configure auto scaling for Amazon DynamoDB tables and indexes](#) 게시물을 참조하세요.

JSON

```
{
  "Resources" : {
    "WriteCapacityScalableTarget" : {
      "Type" : "AWS::ApplicationAutoScaling::ScalableTarget",
      "Properties" : {
        "MaxCapacity" : 15,
        "MinCapacity" : 5,
        "RoleARN" : { "Fn::Sub" : "arn:aws:iam::${AWS::AccountId}:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable" },
        "ServiceNamespace" : "dynamodb",
        "ScalableDimension" : "dynamodb:table:WriteCapacityUnits",
        "ResourceId" : {
          "Fn::Join" : [
            "/",
            [
              "table",
              {
                "Ref" : "logicalName"
              }
            ]
          ]
        }
      }
    },
    "WriteScalingPolicy" : {
      "Type" : "AWS::ApplicationAutoScaling::ScalingPolicy",
      "Properties" : {
        "PolicyName" : "WriteScalingPolicy",
        "PolicyType" : "TargetTrackingScaling",
```

```

    "ScalingTargetId" : { "Ref" : "WriteCapacityScalableTarget" },
    "TargetTrackingScalingPolicyConfiguration" : {
      "TargetValue" : 50.0,
      "ScaleInCooldown" : 60,
      "ScaleOutCooldown" : 60,
      "PredefinedMetricSpecification" : {
        "PredefinedMetricType" : "DynamoDBWriteCapacityUtilization"
      }
    }
  }
}
}
}
}
}
}
}

```

YAML

```

---
Resources:
  WriteCapacityScalableTarget:
    Type: AWS::ApplicationAutoScaling::ScalableTarget
    Properties:
      MaxCapacity: 15
      MinCapacity: 5
      RoleARN:
        Fn::Sub: 'arn:aws:iam::${AWS::AccountId}:role/aws-
service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable'
      ServiceNamespace: dynamodb
      ScalableDimension: dynamodb:table:WriteCapacityUnits
      ResourceId: !Join
        - /
        - - table
          - !Ref logicalName
  WriteScalingPolicy:
    Type: AWS::ApplicationAutoScaling::ScalingPolicy
    Properties:
      PolicyName: WriteScalingPolicy
      PolicyType: TargetTrackingScaling
      ScalingTargetId: !Ref WriteCapacityScalableTarget
      TargetTrackingScalingPolicyConfiguration:
        TargetValue: 50.0
        ScaleInCooldown: 60
        ScaleOutCooldown: 60

```

```
PredefinedMetricSpecification:
```

```
  PredefinedMetricType: DynamoDBWriteCapacityUtilization
```

Amazon ECS 서비스에 대한 조정 정책 생성(지표: 평균 CPU 및 메모리)

이 코드 조각은 [AWS::ApplicationAutoScaling::ScalingPolicy](#) 리소스를 사용하여 정책을 생성하고 [AWS::ECS::Service](#) 리소스에 적용하는 방법을 보여줍니다.

[AWS::ApplicationAutoScaling::ScalableTarget](#) 리소스는 해당 정책이 적용되는 조정 가능 대상을 선언합니다. Application Auto Scaling에서는 작업 수를 최소 1개와 최대 6개 사이에서 조정할 수 있습니다.

이는 TargetTrackingScaling 정책 유형을 사용하여 두 개의 조정 정책을 만듭니다. 이 정책은 서비스의 평균 CPU 및 메모리 사용량을 기준으로 ECS 서비스를 확장하는 데 사용됩니다. Fn::Join 및 Ref 내장 함수를 사용하여 동일한 템플릿에 지정된 [AWS::ECS::Cluster](#)(myContainerCluster) 및 [AWS::ECS::Service](#)(myService) 리소스의 논리명으로 ResourceId 속성을 구성합니다. 자세한 내용은 [내장 함수 참조](#)를 참조하세요.

JSON

```
{
  "Resources" : {
    "ECSScalableTarget" : {
      "Type" : "AWS::ApplicationAutoScaling::ScalableTarget",
      "Properties" : {
        "MaxCapacity" : "6",
        "MinCapacity" : "1",
        "RoleARN" : { "Fn::Sub" : "arn:aws:iam::${AWS::AccountId}:role/
aws-service-role/ecs.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ECSService" },
        "ServiceNamespace" : "ecs",
        "ScalableDimension" : "ecs:service:DesiredCount",
        "ResourceId" : {
          "Fn::Join" : [
            "/",
            [
              "service",
              {
                "Ref" : "myContainerCluster"
              },
              {
                "Fn::GetAtt" : [
                  "myService",
```

```

        "Name"
      ]
    }
  ]
}
},
"ServiceScalingPolicyCPU" : {
  "Type" : "AWS::ApplicationAutoScaling::ScalingPolicy",
  "Properties" : {
    "PolicyName" : { "Fn::Sub" : "${AWS::StackName}-target-tracking-cpu70" },
    "PolicyType" : "TargetTrackingScaling",
    "ScalingTargetId" : { "Ref" : "ECSScalableTarget" },
    "TargetTrackingScalingPolicyConfiguration" : {
      "TargetValue" : 70.0,
      "ScaleInCooldown" : 180,
      "ScaleOutCooldown" : 60,
      "PredefinedMetricSpecification" : {
        "PredefinedMetricType" : "ECSServiceAverageCPUUtilization"
      }
    }
  }
},
"ServiceScalingPolicyMem" : {
  "Type" : "AWS::ApplicationAutoScaling::ScalingPolicy",
  "Properties" : {
    "PolicyName" : { "Fn::Sub" : "${AWS::StackName}-target-tracking-mem90" },
    "PolicyType" : "TargetTrackingScaling",
    "ScalingTargetId" : { "Ref" : "ECSScalableTarget" },
    "TargetTrackingScalingPolicyConfiguration" : {
      "TargetValue" : 90.0,
      "ScaleInCooldown" : 180,
      "ScaleOutCooldown" : 60,
      "PredefinedMetricSpecification" : {
        "PredefinedMetricType" : "ECSServiceAverageMemoryUtilization"
      }
    }
  }
}
}
}
```

YAML

```

---
Resources:
  ECSScalableTarget:
    Type: AWS::ApplicationAutoScaling::ScalableTarget
    Properties:
      MaxCapacity: 6
      MinCapacity: 1
      RoleARN:
        Fn::Sub: 'arn:aws:iam::${AWS::AccountId}:role/aws-service-role/ecs.application-
autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_ECSService'
      ServiceNamespace: ecs
      ScalableDimension: 'ecs:service:DesiredCount'
      ResourceId: !Join
        - /
        - - service
          - !Ref myContainerCluster
          - !GetAtt myService.Name
  ServiceScalingPolicyCPU:
    Type: AWS::ApplicationAutoScaling::ScalingPolicy
    Properties:
      PolicyName: !Sub ${AWS::StackName}-target-tracking-cpu70
      PolicyType: TargetTrackingScaling
      ScalingTargetId: !Ref ECSScalableTarget
      TargetTrackingScalingPolicyConfiguration:
        TargetValue: 70.0
        ScaleInCooldown: 180
        ScaleOutCooldown: 60
        PredefinedMetricSpecification:
          PredefinedMetricType: ECSServiceAverageCPUUtilization
  ServiceScalingPolicyMem:
    Type: AWS::ApplicationAutoScaling::ScalingPolicy
    Properties:
      PolicyName: !Sub ${AWS::StackName}-target-tracking-mem90
      PolicyType: TargetTrackingScaling
      ScalingTargetId: !Ref ECSScalableTarget
      TargetTrackingScalingPolicyConfiguration:
        TargetValue: 90.0
        ScaleInCooldown: 180
        ScaleOutCooldown: 60
        PredefinedMetricSpecification:
          PredefinedMetricType: ECSServiceAverageMemoryUtilization

```


Amazon ECS 서비스에 대한 조정 정책 생성(지표: 대상당 평균 요청 수)

다음 예에서는 ALBRequestCountPerTarget 사전 정의 지표가 있는 대상 추적 조정 정책을 ECS 서비스에 적용합니다. 이 정책은 분당 대상당 요청 수가 목표 값을 초과하는 경우 ECS 서비스에 용량을 추가하는 데 사용됩니다. DisableScaleIn의 값이 true로 설정되어 있기 때문에 대상 추적 정책은 확장 가능한 대상에서 용량을 제거하지 않습니다.

Fn::Join 및 Fn::GetAtt 내장 함수를 사용하여 동일한 템플릿에 지정된

[AWS::ElasticLoadBalancingV2::LoadBalancer](#)(myLoadBalancer) 및

[AWS::ElasticLoadBalancingV2::TargetGroup](#)(myTargetGroup) 리소스의 논리명으로

ResourceLabel 속성을 구성합니다. 자세한 내용은 [내장 함수 참조](#)를 참조하세요.

확장 가능한 대상의 MaxCapacity 및 MinCapacity 속성과 확장 정책의 TargetValue 속성은 스택을 만들거나 업데이트할 때 템플릿에 전달하는 파라미터 값을 참조합니다.

JSON

```
{
  "Resources" : {
    "ECSScalableTarget" : {
      "Type" : "AWS::ApplicationAutoScaling::ScalableTarget",
      "Properties" : {
        "MaxCapacity" : { "Ref" : "MaxCount" },
        "MinCapacity" : { "Ref" : "MinCount" },
        "RoleARN" : { "Fn::Sub" : "arn:aws:iam::${AWS::AccountId}:role/
aws-service-role/ecs.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ECSService" },
        "ServiceNamespace" : "ecs",
        "ScalableDimension" : "ecs:service:DesiredCount",
        "ResourceId" : {
          "Fn::Join" : [
            "/",
            [
              "service",
              {
                "Ref" : "myContainerCluster"
              },
              {
                "Fn::GetAtt" : [
                  "myService",
                  "Name"
                ]
              }
            ]
          ]
        }
      }
    }
  }
}
```

```
[
]
}
}
},
"ServiceScalingPolicyALB" : {
  "Type" : "AWS::ApplicationAutoScaling::ScalingPolicy",
  "Properties" : {
    "PolicyName" : "alb-requests-per-target-per-minute",
    "PolicyType" : "TargetTrackingScaling",
    "ScalingTargetId" : { "Ref" : "ECSScalableTarget" },
    "TargetTrackingScalingPolicyConfiguration" : {
      "TargetValue" : { "Ref" : "ALBPolymerTargetValue" },
      "ScaleInCooldown" : 180,
      "ScaleOutCooldown" : 30,
      "DisableScaleIn" : true,
      "PredefinedMetricSpecification" : {
        "PredefinedMetricType" : "ALBRequestCountPerTarget",
        "ResourceLabel" : {
          "Fn::Join" : [
            "/",
            [
              {
                "Fn::GetAtt" : [
                  "myLoadBalancer",
                  "LoadBalancerFullName"
                ]
              },
              {
                "Fn::GetAtt" : [
                  "myTargetGroup",
                  "TargetGroupName"
                ]
              }
            ]
          ]
        }
      }
    }
  }
}
```

YAML

```

---
Resources:
  ECSScalableTarget:
    Type: AWS::ApplicationAutoScaling::ScalableTarget
    Properties:
      MaxCapacity: !Ref MaxCount
      MinCapacity: !Ref MinCount
      RoleARN:
        Fn::Sub: 'arn:aws:iam::${AWS::AccountId}:role/aws-service-role/ecs.application-
autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_ECSService'
      ServiceNamespace: ecs
      ScalableDimension: 'ecs:service:DesiredCount'
      ResourceId: !Join
        - /
        - - service
          - !Ref myContainerCluster
          - !GetAtt myService.Name
  ServiceScalingPolicyALB:
    Type: AWS::ApplicationAutoScaling::ScalingPolicy
    Properties:
      PolicyName: alb-requests-per-target-per-minute
      PolicyType: TargetTrackingScaling
      ScalingTargetId: !Ref ECSScalableTarget
      TargetTrackingScalingPolicyConfiguration:
        TargetValue: !Ref ALBPolicyTargetValue
        ScaleInCooldown: 180
        ScaleOutCooldown: 30
        DisableScaleIn: true
        PredefinedMetricSpecification:
          PredefinedMetricType: ALBRequestCountPerTarget
          ResourceLabel: !Join
            - '/'
            - - !GetAtt myLoadBalancer.LoadBalancerFullName
              - !GetAtt myTargetGroup.TargetGroupFullName

```

Lambda 함수에 대해 cron 표현식을 사용하는 예약된 작업 생성

이 코드 조각은 [AWS::ApplicationAutoScaling::ScalableTarget](#) 리소스를 사용하여 BLUE라는 함수 별칭([AWS::Lambda::Alias](#))에 대해 프로비저닝된 동시성을 등록합니다. 또한 Cron 식을 사용하여 반복 일정으로 예약 작업을 만듭니다. 반복 일정의 시간대는 UTC입니다.

RoleARN 속성의 Fn::Join 및 Ref 내장 함수를 사용하여 서비스 연결 역할의 ARN을 지정합니다. Fn::Sub 내장 함수를 사용하여 동일한 템플릿에 지정된 [AWS::Lambda::Function](#) 또는 [AWS::Serverless::Function](#) 리소스의 논리명으로 ResourceId 속성을 구성합니다. 자세한 내용은 [내장 함수 참조](#)를 참조하세요.

Note

게시되지 않은 버전(\$LATEST)을 가리키는 별칭에는 프로비저닝된 동시성을 할당할 수 없습니다.

Lambda 리소스에 대한 CloudFormation 템플릿을 생성하는 방법에 대한 자세한 내용을 알아보려면 AWS Compute 블로그의 [Scheduling AWS Lambda Provisioned Concurrency for recurring peak usage](#) 게시물을 참조하세요.

JSON

```
{
  "ScalableTarget" : {
    "Type" : "AWS::ApplicationAutoScaling::ScalableTarget",
    "Properties" : {
      "MaxCapacity" : 250,
      "MinCapacity" : 0,
      "RoleARN" : {
        "Fn::Join" : [
          ":",
          [
            "arn:aws:iam:",
            {
              "Ref" : "AWS::AccountId"
            },
            "role/aws-service-role/lambda.application-autoscaling.amazonaws.com/"
          ]
        ]
      },
      "AWSServiceRoleForApplicationAutoScaling_LambdaConcurrency"
    ]
  },
  "ServiceNamespace" : "lambda",
  "ScalableDimension" : "lambda:function:ProvisionedConcurrency",
  "ResourceId" : { "Fn::Sub" : "function:${logicalName}:BLUE" },
  "ScheduledActions" : [
    {
      "ScalableTargetAction" : {
```

```

        "MinCapacity" : "250"
      },
      "ScheduledActionName" : "my-scale-out-scheduled-action",
      "Schedule" : "cron(0 18 * * ? *)",
      "EndTime" : "2022-12-31T12:00:00.000Z"
    }
  ]
}
}
}

```

YAML

```

ScalableTarget:
  Type: AWS::ApplicationAutoScaling::ScalableTarget
  Properties:
    MaxCapacity: 250
    MinCapacity: 0
    RoleARN: !Join
      - ':'
      - - 'arn:aws:iam:'
        - !Ref 'AWS::AccountId'
        - role/aws-service-role/lambda.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_LambdaConcurrency
  ServiceNamespace: lambda
  ScalableDimension: lambda:function:ProvisionedConcurrency
  ResourceId: !Sub function:${logicalName}:BLUE
  ScheduledActions:
    - ScalableTargetAction:
        MinCapacity: 250
        ScheduledActionName: my-scale-out-scheduled-action
        Schedule: 'cron(0 18 * * ? *)'
        EndTime: '2022-12-31T12:00:00.000Z'

```

스팟 플릿에 대해 **at** 표현식을 사용하는 예약된 작업 생성

이 코드 조각은 [AWS::ApplicationAutoScaling::ScalableTarget](#) 리소스를 사용하여 [AWS::EC2::SpotFleet](#) 리소스에 대해 한 번만 발생하는 두 개의 예약된 작업을 생성하는 방법을 보여 줍니다. 각 일회성 예약된 작업의 시간대는 UTC입니다.

Fn::Join 및 Ref 내장 함수를 통해 동일한 템플릿에 지정된 AWS::EC2::SpotFleet 리소스의 논리명으로 ResourceId 속성을 구성합니다. 자세한 내용은 [내장 함수 참조](#)를 참조하세요.

Note

스팟 플릿 요청에는 maintain 유형의 요청이 있어야 합니다. 1회 요청 또는 스팟 블록에는 자동 조정이 지원되지 않습니다.

JSON

```
{
  "Resources" : {
    "SpotFleetScalableTarget" : {
      "Type" : "AWS::ApplicationAutoScaling::ScalableTarget",
      "Properties" : {
        "MaxCapacity" : 0,
        "MinCapacity" : 0,
        "RoleARN" : { "Fn::Sub" : "arn:aws:iam::${AWS::AccountId}:role/
aws-service-role/ec2.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest" },
        "ServiceNamespace" : "ec2",
        "ScalableDimension" : "ec2:spot-fleet-request:TargetCapacity",
        "ResourceId" : {
          "Fn::Join" : [
            "/",
            [
              "spot-fleet-request",
              {
                "Ref" : "logicalName"
              }
            ]
          ]
        },
      },
      "ScheduledActions" : [
        {
          "ScalableTargetAction" : {
            "MaxCapacity" : "10",
            "MinCapacity" : "10"
          },
          "ScheduledActionName" : "my-scale-out-scheduled-action",
          "Schedule" : "at(2022-05-20T13:00:00)"
        },
        {
          "ScalableTargetAction" : {
            "MaxCapacity" : "0",
```

```

        "MinCapacity" : "0"
      },
      "ScheduledActionName" : "my-scale-in-scheduled-action",
      "Schedule" : "at(2022-05-20T21:00:00)"
    }
  ]
}
}
}
}

```

YAML

```

---
Resources:
  SpotFleetScalableTarget:
    Type: AWS::ApplicationAutoScaling::ScalableTarget
    Properties:
      MaxCapacity: 0
      MinCapacity: 0
      RoleARN:
        Fn::Sub: 'arn:aws:iam::${AWS::AccountId}:role/aws-service-role/ec2.application-
autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest'
      ServiceNamespace: ec2
      ScalableDimension: 'ec2:spot-fleet-request:TargetCapacity'
      ResourceId: !Join
        - /
        - - spot-fleet-request
          - !Ref logicalName
      ScheduledActions:
        - ScalableTargetAction:
            MaxCapacity: 10
            MinCapacity: 10
            ScheduledActionName: my-scale-out-scheduled-action
            Schedule: 'at(2022-05-20T13:00:00)'
        - ScalableTargetAction:
            MaxCapacity: 0
            MinCapacity: 0
            ScheduledActionName: my-scale-in-scheduled-action
            Schedule: 'at(2022-05-20T21:00:00)'

```

AWS 빌링 콘솔 템플릿 코드 조각

이 예제에서는 10% 글로벌 마크업 요금 규칙을 사용하여 하나의 요금제를 생성합니다. 이 요금제는 과금 그룹에 연결되어 있습니다. 과금 그룹에는 과금 그룹 총 비용에 10 USD 청구와 10% 청구를 추가로 적용하는 2개의 사용자 지정 항목도 있습니다.

JSON

```
{
  "Parameters": {
    "LinkedAccountIds": {
      "Type": "ListNumber"
    },
    "PrimaryAccountId": {
      "Type": "Number"
    }
  },
  "Resources": {
    "TestPricingRule": {
      "Type": "AWS::BillingConductor::PricingRule",
      "Properties": {
        "Name": "TestPricingRule",
        "Description": "Test pricing rule created through Cloudformation. Mark everything by 10%.",
        "Type": "MARKUP",
        "Scope": "GLOBAL",
        "ModifierPercentage": 10
      }
    },
    "TestPricingPlan": {
      "Type": "AWS::BillingConductor::PricingPlan",
      "Properties": {
        "Name": "TestPricingPlan",
        "Description": "Test pricing plan created through Cloudformation.",
        "PricingRuleArns": [
          {"Fn::GetAtt": ["TestPricingRule", "Arn"]}
        ]
      }
    },
    "TestBillingGroup": {
      "Type": "AWS::BillingConductor::BillingGroup",
      "Properties": {
        "Name": "TestBillingGroup",
```



```

        "Description": "Test billing group created through Cloudformation with 1
linked account. The linked account is also the primary account.",
        "PrimaryAccountId": {
            "Ref": "PrimaryAccountId"
        },
        "AccountGrouping": {
            "LinkedAccountIds": null
        },
        "ComputationPreference": {
            "PricingPlanArn": {
                "Fn::GetAtt": ["TestPricingPlan", "Arn"]
            }
        }
    },
    "TestFlatCustomLineItem": {
        "Type": "AWS::BillingConductor::CustomLineItem",
        "Properties": {
            "Name": "TestFlatCustomLineItem",
            "Description": "Test flat custom line item created through Cloudformation
for a $10 charge.",
            "BillingGroupArn": {
                "Fn::GetAtt": ["TestBillingGroup", "Arn"]
            },
            "CustomLineItemChargeDetails": {
                "Flat": {
                    "ChargeValue": 10
                },
                "Type": "FEE"
            }
        }
    },
    "TestPercentageCustomLineItem": {
        "Type": "AWS::BillingConductor::CustomLineItem",
        "Properties": {
            "Name": "TestPercentageCustomLineItem",
            "Description": "Test percentage custom line item created through
Cloudformation for a %10 additional charge on the overall total bill of the billing
group.",
            "BillingGroupArn": {
                "Fn::GetAtt": ["TestBillingGroup", "Arn"]
            },
            "CustomLineItemChargeDetails": {
                "Percentage": {

```

```
        "PercentageValue": 10,  
        "ChildAssociatedResources": [  
            {"Fn::GetAtt": ["TestBillingGroup", "Arn"]}  
        ]  
    },  
    "Type": "FEE"  
}  
  
}  
  
}  
  
}
```

YAML

```
Parameters:
  LinkedAccountIds:
    Type: ListNumber
  PrimaryAccountId:
    Type: Number
Resources:
  TestPricingRule:
    Type: AWS::BillingConductor::PricingRule
    Properties:
      Name: 'TestPricingRule'
      Description: 'Test pricing rule created through Cloudformation. Mark everything
by 10%.'
      Type: 'MARKUP'
      Scope: 'GLOBAL'
      ModifierPercentage: 10
  TestPricingPlan:
    Type: AWS::BillingConductor::PricingPlan
    Properties:
      Name: 'TestPricingPlan'
      Description: 'Test pricing plan created through Cloudformation.'
      PricingRuleArns:
        - !GetAtt TestPricingRule.Arn
  TestBillingGroup:
    Type: AWS::BillingConductor::BillingGroup
    Properties:
      Name: 'TestBillingGroup'
      Description: 'Test billing group created through Cloudformation with 1 linked
account. The linked account is also the primary account.'
      PrimaryAccountId: !Ref PrimaryAccountId
```

```

    AccountGrouping:
      LinkedAccountIds: !Ref LinkedAccountIds
    ComputationPreference:
      PricingPlanArn: !GetAtt TestPricingPlan.Arn
  TestFlatCustomLineItem:
    Type: AWS::BillingConductor::CustomLineItem
    Properties:
      Name: 'TestFlatCustomLineItem'
      Description: 'Test flat custom line item created through Cloudformation for a $10
charge.'
      BillingGroupArn: !GetAtt TestBillingGroup.Arn
      CustomLineItemChargeDetails:
        Flat:
          ChargeValue: 10
          Type: 'FEE'
  TestPercentageCustomLineItem:
    Type: AWS::BillingConductor::CustomLineItem
    Properties:
      Name: 'TestPercentageCustomLineItem'
      Description: 'Test percentage custom line item created through Cloudformation for
a %10 additional charge on the overall total bill of the billing group.'
      BillingGroupArn: !GetAtt TestBillingGroup.Arn
      CustomLineItemChargeDetails:
        Percentage:
          PercentageValue: 10
          ChildAssociatedResources:
            - !GetAtt TestBillingGroup.Arn
          Type: 'FEE'

```

CloudFormation 템플릿 코드 조각

주제

- [중첩 스택](#)
- [대기 조건](#)

중첩 스택

템플릿에 스택 중첩

이 예제 템플릿에는 myStack이라는 중첩 스택 리소스가 포함되어 있습니다. CloudFormation은 템플릿에서 스택을 생성할 때 myStack을 생성하는데, 이 스택에서 템플릿은 TemplateURL 속성에 지정

됩니다. 출력 값 StackRef는 myStack의 스택 ID이고 OutputFromNestedStack 값은 myStack 리소스의 출력 값 BucketName을 반환합니다. Outputs.*nestedstackoutputname* 형식은 중첩 스택의 출력 값을 지정하는 용도로 예약되어 있으며 포함 템플릿 내 어디서든 사용할 수 있습니다.

자세한 내용은 [AWS::CloudFormation::Stack](#) 섹션을 참조하세요.

JSON

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Resources" : {
    "myStack" : {
      "Type" : "AWS::CloudFormation::Stack",
      "Properties" : {
        "TemplateURL" : "https://s3.amazonaws.com/cloudformation-templates-us-east-1/
S3_Bucket.template",
        "TimeoutInMinutes" : "60"
      }
    }
  },
  "Outputs": {
    "StackRef": {"Value": { "Ref" : "myStack"}},
    "OutputFromNestedStack" : {
      "Value" : { "Fn::GetAtt" : [ "myStack", "Outputs.BucketName" ] }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myStack:
    Type: AWS::CloudFormation::Stack
    Properties:
      TemplateURL: https://s3.amazonaws.com/cloudformation-templates-us-east-1/
S3_Bucket.template
      TimeoutInMinutes: '60'
Outputs:
  StackRef:
    Value: !Ref myStack
  OutputFromNestedStack:
```

```
Value: !GetAtt myStack.Outputs.BucketName
```

템플릿 내 입력 파라미터를 사용하여 스택 중첩

이 예제 템플릿에는 입력 파라미터를 지정하는 스택 리소스가 들어 있습니다. CloudFormation이 템플릿에서 스택을 생성하는 경우, myStackWithParams 스택을 생성하는 데 사용되는 템플릿의 입력 파라미터로 Parameters 속성 안에 선언되는 값 페어를 사용합니다. 이 예제에는 InstanceType 및 KeyName 파라미터가 지정되어 있습니다.

자세한 내용은 [AWS::CloudFormation::Stack](#) 섹션을 참조하세요.

JSON

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Resources" : {
    "myStackWithParams" : {
      "Type" : "AWS::CloudFormation::Stack",
      "Properties" : {
        "TemplateURL" : "https://s3.amazonaws.com/cloudformation-templates-us-east-1/EC2ChooseAMI.template",
        "Parameters" : {
          "InstanceType" : "t2.micro",
          "KeyName" : "mykey"
        }
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myStackWithParams:
    Type: AWS::CloudFormation::Stack
    Properties:
      TemplateURL: https://s3.amazonaws.com/cloudformation-templates-us-east-1/EC2ChooseAMI.template
      Parameters:
        InstanceType: t2.micro
        KeyName: mykey
```

대기 조건

Amazon EC2 인스턴스에 대기 조건 사용

Important

Amazon EC2 및 Auto Scaling 리소스의 경우 대기 조건 대신 CreationPolicy 속성을 사용하는 것이 좋습니다. 그러한 리소스에 CreationPolicy 속성을 추가하고 cfn-signal 헬퍼 스크립트를 사용하여 인스턴스 생성 프로세스가 완료되면 신호를 보내도록 합니다.

생성 정책을 사용할 수 없는 경우 다음 예제 템플릿을 참조할 수 있습니다. 이 템플릿에서는 대기 조건을 사용하여 Amazon EC2 인스턴스를 선언합니다. myWaitCondition 대기 조건은 신호 전송에 myWaitConditionHandle을 사용하고, DependsOn 속성을 사용하여 Amazon EC2 인스턴스 리소스가 생성된 후 대기 조건이 트리거되도록 지정하고, Timeout 속성을 사용하여 대기 조건의 기간을 4,500초로 지정합니다. 또한, 대기 조건에 신호를 전송하는 미리 서명된 URL이 Ec2Instance 리소스의 UserData 속성을 사용하여 Amazon EC2 인스턴스에 전달되므로, 해당 Amazon EC2 인스턴스에서 실행 중인 애플리케이션이나 스크립트에서 미리 서명된 URL을 검색한 후 이 URL을 사용하여 대기 조건에 성공 또는 실패 신호를 전송할 수 있습니다. cfn-signal을 사용하거나 대기 조건에 신호를 전송하는 애플리케이션 또는 스크립트를 생성해야 합니다. 출력 값 ApplicationData에는 대기 조건 신호에서 다시 전달된 데이터가 포함되어 있습니다.

자세한 내용은 [CloudFormation 템플릿에서 대기 조건 생성](#) 섹션을 참조하세요.

JSON

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Mappings" : {
    "RegionMap" : {
      "us-east-1" : {
        "AMI" : "ami-0123456789abcdef0"
      },
      "us-west-1" : {
        "AMI" : "ami-0987654321fedcba0"
      },
      "eu-west-1" : {
        "AMI" : "ami-0abcdef123456789a"
      },
      "ap-northeast-1" : {
```

```

        "AMI" : "ami-0fedcba987654321b"
    },
    "ap-southeast-1" : {
        "AMI" : "ami-0c1d2e3f4a5b6c7d8"
    }
},
"Resources" : {
    "Ec2Instance" : {
        "Type" : "AWS::EC2::Instance",
        "Properties" : {
            "UserData" : { "Fn::Base64" : {"Ref" : "myWaitHandle"}},
            "ImageId" : { "Fn::FindInMap" : [ "RegionMap", { "Ref" :
"AWS::Region" }, "AMI" ]}
        }
    },
    "myWaitHandle" : {
        "Type" : "AWS::CloudFormation::WaitConditionHandle",
        "Properties" : {
        }
    },
    "myWaitCondition" : {
        "Type" : "AWS::CloudFormation::WaitCondition",
        "DependsOn" : "Ec2Instance",
        "Properties" : {
            "Handle" : { "Ref" : "myWaitHandle" },
            "Timeout" : "4500"
        }
    }
},
"Outputs" : {
    "ApplicationData" : {
        "Value" : { "Fn::GetAtt" : [ "myWaitCondition", "Data" ]},
        "Description" : "The data passed back as part of signalling the
WaitCondition."
    }
}
}

```

YAML

```

AWSTemplateFormatVersion: '2010-09-09'
Mappings:

```

```

RegionMap:
  us-east-1:
    AMI: ami-0123456789abcdef0
  us-west-1:
    AMI: ami-0987654321fedcba0
  eu-west-1:
    AMI: ami-0abcdef123456789a
  ap-northeast-1:
    AMI: ami-0fedcba987654321b
  ap-southeast-1:
    AMI: ami-0c1d2e3f4a5b6c7d8
Resources:
  Ec2Instance:
    Type: AWS::EC2::Instance
    Properties:
      UserData:
        Fn::Base64: !Ref myWaitHandle
      ImageId:
        Fn::FindInMap:
          - RegionMap
          - Ref: AWS::Region
          - AMI
  myWaitHandle:
    Type: AWS::CloudFormation::WaitConditionHandle
    Properties: {}
  myWaitCondition:
    Type: AWS::CloudFormation::WaitCondition
    DependsOn: Ec2Instance
    Properties:
      Handle: !Ref myWaitHandle
      Timeout: '4500'
Outputs:
  ApplicationData:
    Value: !GetAtt myWaitCondition.Data
    Description: The data passed back as part of signalling the WaitCondition.

```

cfn-signal 헬퍼 스크립트를 사용하여 대기 조건에 신호 전송

이 예제는 대기 조건에 성공 신호를 전송하는 cfn-signal 명령줄을 보여줍니다. EC2 인스턴스의 UserData 속성에서 명령줄을 정의해야 합니다.

JSON

```
"UserData": {
  "Fn::Base64": {
    "Fn::Join": [
      "",
      [
        "#!/bin/bash -xe\n",
        "/opt/aws/bin/cfn-signal --exit-code 0 '",
        {
          "Ref": "myWaitHandle"
        },
        "'\n"
      ]
    ]
  }
}
```

YAML

```
UserData:
  'Fn::Base64':
    'Fn::Join':
      - ''
      - - |
          #!/bin/bash -xe
          - /opt/aws/bin/cfn-signal --exit-code 0 '
          - Ref: myWaitHandle
          - |
            ,
```

Curl을 사용하여 대기 조건에 신호 전송

다음 예제에서는 대기 조건에 성공 신호를 전송하는 Curl 명령줄을 보여 줍니다.

```
curl -T /tmp/a "https://cloudformation-waitcondition-test.s3.amazonaws.com/
arn%3Aaws%3Acloudformation%3Aus-east-1%3A034017226601%3Astack
%2Fstack-gosar-20110427004224-test-stack-with-WaitCondition--VEYW
%2Fe498ce60-70a1-11e0-81a7-5081d0136786%2FmyWaitConditionHandle?
Expires=1303976584&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Signature=ik1twT6hpS4cgNAw7wy0oRejVoo
%3D"
```

이때 file /tmp/a에는 다음과 같은 JSON 구조가 포함되어 있습니다.

```
{
  "Status" : "SUCCESS",
  "Reason" : "Configuration Complete",
  "UniqueId" : "ID1234",
  "Data" : "Application has completed configuration."
}
```

이 예제에서는 명령줄에서 JSON을 파라미터로 전송하는 경우를 제외하고는 동일한 성공 신호를 전송하는 Curl 명령줄을 표시합니다.

```
curl -X PUT -H 'Content-Type:' --data-binary '{"Status" : "SUCCESS","Reason" : "Configuration Complete","UniqueId" : "ID1234","Data" : "Application has completed configuration."}' "https://cloudformation-waitcondition-test.s3.amazonaws.com/arn%3Aaws%3Acloudformation%3Aus-east-1%3A034017226601%3Astack%2Fstack-gosar-20110427004224-test-stack-with-WaitCondition--VEYW%2Fe498ce60-70a1-11e0-81a7-5081d0136786%2FmyWaitConditionHandle?Expires=1303976584&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Signature=ik1twT6hpS4cgNAw7wy0oRejVoo%3D"
```

Amazon CloudFront 템플릿 코드 조각

CloudFormation에서 이러한 샘플 템플릿 코드 조각을 Amazon CloudFront 배포 리소스에 사용합니다. 자세한 내용은 [Amazon CloudFront 리소스 유형](#)을 참조하세요.

주제

- [Amazon S3 오리진을 사용하는 Amazon CloudFront 배포 리소스](#)
- [사용자 지정 오리진을 사용하는 Amazon CloudFront 배포 리소스](#)
- [다중 오리진 지원으로 Amazon CloudFront 배포](#)
- [Lambda 함수를 오리진으로 사용하여 Amazon CloudFront 배포](#)
- [다음 사항도 참조하세요.](#)

Amazon S3 오리진을 사용하는 Amazon CloudFront 배포 리소스

다음 예제 템플릿에서는 [S3Origin](#) 및 레거시 오리진 액세스 ID(OAI)를 사용하는 Amazon CloudFront [배포](#)를 보여줍니다. 오리진 액세스 제어(OAC)를 대신 사용하는 방법에 대한 자세한 내용을 알아보려

면 Amazon CloudFront 개발자 안내서의 [Amazon Simple Storage Service 오리진에 대한 액세스 제한](#)을 참조하세요.

JSON

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Resources" : {
    "myDistribution" : {
      "Type" : "AWS::CloudFront::Distribution",
      "Properties" : {
        "DistributionConfig" : {
          "Origins" : [ {
            "DomainName" : "amzn-s3-demo-bucket.s3.amazonaws.com",
            "Id" : "myS3Origin",
            "S3OriginConfig" : {
              "OriginAccessIdentity" : "origin-access-identity/
cloudfront/E127EXAMPLE51Z"
            }
          } ],
          "Enabled" : "true",
          "Comment" : "Some comment",
          "DefaultRootObject" : "index.html",
          "Logging" : {
            "IncludeCookies" : "false",
            "Bucket" : "amzn-s3-demo-logging-bucket.s3.amazonaws.com",
            "Prefix" : "myprefix"
          },
          "Aliases" : [ "mysite.example.com", "yoursite.example.com" ],
          "DefaultCacheBehavior" : {
            "AllowedMethods" : [ "DELETE", "GET", "HEAD", "OPTIONS",
"PATCH", "POST", "PUT" ],
            "TargetOriginId" : "myS3Origin",
            "ForwardedValues" : {
              "QueryString" : "false",
              "Cookies" : { "Forward" : "none" }
            },
            "TrustedSigners" : [ "1234567890EX", "1234567891EX" ],
            "ViewerProtocolPolicy" : "allow-all"
          },
          "PriceClass" : "PriceClass_200",
          "Restrictions" : {
            "GeoRestriction" : {
              "RestrictionType" : "whitelist",
```

```
    "Locations" : [ "AQ", "CV" ]  
  },  
  "ViewerCertificate" : { "CloudFrontDefaultCertificate" : "true" }  
}  
  
}  
  
}
```

YAML

```

AWSTemplateFormatVersion: '2010-09-09'

Resources:
  myDistribution:
    Type: AWS::CloudFront::Distribution
    Properties:
      DistributionConfig:
        Origins:
          - DomainName: amzn-s3-demo-bucket.s3.amazonaws.com
            Id: myS3Origin
            S3OriginConfig:
              OriginAccessIdentity: origin-access-identity/cloudfront/E127EXAMPLE51Z
        Enabled: 'true'
        Comment: Some comment
        DefaultRootObject: index.html
        Logging:
          IncludeCookies: 'false'
          Bucket: amzn-s3-demo-logging-bucket.s3.amazonaws.com
          Prefix: myprefix
        Aliases:
          - mysite.example.com
          - yoursite.example.com
        DefaultCacheBehavior:
          AllowedMethods:
            - DELETE
            - GET
            - HEAD
            - OPTIONS
            - PATCH
            - POST
            - PUT
          TargetOriginId: myS3Origin

```

```

ForwardedValues:
  QueryString: 'false'
  Cookies:
    Forward: none
TrustedSigners:
- 1234567890EX
- 1234567891EX
ViewerProtocolPolicy: allow-all
PriceClass: PriceClass_200
Restrictions:
  GeoRestriction:
    RestrictionType: whitelist
    Locations:
      - AQ
      - CV
ViewerCertificate:
  CloudFrontDefaultCertificate: 'true'

```

사용자 지정 오리진을 사용하는 Amazon CloudFront 배포 리소스

다음 예제 템플릿에서는 [CustomOrigin](#)을 사용하는 Amazon CloudFront [배포](#)를 보여줍니다.

JSON

```

{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Resources" : {
    "myDistribution" : {
      "Type" : "AWS::CloudFront::Distribution",
      "Properties" : {
        "DistributionConfig" : {
          "Origins" : [ {
            "DomainName" : "www.example.com",
            "Id" : "myCustomOrigin",
            "CustomOriginConfig" : {
              "HTTPPort" : "80",
              "HTTPSPort" : "443",
              "OriginProtocolPolicy" : "http-only"
            }
          } ],
          "Enabled" : "true",
          "Comment" : "Somecomment",
          "DefaultRootObject" : "index.html",
          "Logging" : {

```

```

    "IncludeCookies" : "true",
    "Bucket" : "amzn-s3-demo-logging-bucket.s3.amazonaws.com",
    "Prefix": "myprefix"
  },
  "Aliases" : [
    "mysite.example.com",
    "*.yoursite.example.com"
  ],
  "DefaultCacheBehavior" : {
    "TargetOriginId" : "myCustomOrigin",
    "SmoothStreaming" : "false",
    "ForwardedValues" : {
      "QueryString" : "false",
      "Cookies" : { "Forward" : "all" }
    },
    "TrustedSigners" : [
      "1234567890EX",
      "1234567891EX"
    ],
    "ViewerProtocolPolicy" : "allow-all"
  },
  "CustomErrorResponses" : [ {
    "ErrorCode" : "404",
    "ResponsePagePath" : "/error-pages/404.html",
    "ResponseCode" : "200",
    "ErrorCachingMinTTL" : "30"
  } ],
  "PriceClass" : "PriceClass_200",
  "Restrictions" : {
    "GeoRestriction" : {
      "RestrictionType" : "whitelist",
      "Locations" : [ "AQ", "CV" ]
    }
  },
  "ViewerCertificate": { "CloudFrontDefaultCertificate" : "true" }
}
}
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myDistribution:
    Type: AWS::CloudFront::Distribution
    Properties:
      DistributionConfig:
        Origins:
          - DomainName: www.example.com
            Id: myCustomOrigin
            CustomOriginConfig:
              HTTPPort: '80'
              HTTPSPort: '443'
              OriginProtocolPolicy: http-only
        Enabled: 'true'
        Comment: Somecomment
        DefaultRootObject: index.html
        Logging:
          IncludeCookies: 'true'
          Bucket: amzn-s3-demo-logging-bucket.s3.amazonaws.com
          Prefix: myprefix
        Aliases:
          - mysite.example.com
          - "*.yoursite.example.com"
        DefaultCacheBehavior:
          TargetOriginId: myCustomOrigin
          SmoothStreaming: 'false'
          ForwardedValues:
            QueryString: 'false'
            Cookies:
              Forward: all
          TrustedSigners:
            - 1234567890EX
            - 1234567891EX
          ViewerProtocolPolicy: allow-all
      CustomErrorResponses:
        - ErrorCode: '404'
          ResponsePagePath: "/error-pages/404.html"
          ResponseCode: '200'
          ErrorCachingMinTTL: '30'
      PriceClass: PriceClass_200
      Restrictions:
        GeoRestriction:
```

```

    RestrictionType: whitelist
    Locations:
      - AQ
      - CV
  ViewerCertificate:
    CloudFrontDefaultCertificate: 'true'

```

다중 오리진 지원으로 Amazon CloudFront 배포

다음 예제 템플릿에서는 다중 오리진 지원을 사용하는 CloudFront [배포](#)를 선언하는 방법을 보여줍니다. [DistributionConfig](#)에서 오리진 목록이 제공되며 [DefaultCacheBehavior](#)가 설정됩니다.

JSON

```

{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Resources" : {
    "myDistribution" : {
      "Type" : "AWS::CloudFront::Distribution",
      "Properties" : {
        "DistributionConfig" : {
          "Origins" : [ {
            "Id" : "myS3Origin",
            "DomainName" : "amzn-s3-demo-bucket.s3.amazonaws.com",
            "S3OriginConfig" : {
              "OriginAccessIdentity" : "origin-access-identity/
cloudfront/E127EXAMPLE51Z"
            }
          },
          {
            "Id" : "myCustomOrigin",
            "DomainName" : "www.example.com",
            "CustomOriginConfig" : {
              "HTTPPort" : "80",
              "HTTPSPort" : "443",
              "OriginProtocolPolicy" : "http-only"
            }
          }
        ],
        "Enabled" : "true",
        "Comment" : "Some comment",
        "DefaultRootObject" : "index.html",
        "Logging" : {

```



```

        "IncludeCookies" : "true",
        "Bucket" : "amzn-s3-demo-logging-bucket.s3.amazonaws.com",
        "Prefix" : "myprefix"
    },
    "Aliases" : [ "mysite.example.com", "yoursite.example.com" ],
    "DefaultCacheBehavior" : {
        "TargetOriginId" : "myS3Origin",
        "ForwardedValues" : {
            "QueryString" : "false",
            "Cookies" : { "Forward" : "all" }
        },
        "TrustedSigners" : [ "1234567890EX", "1234567891EX" ],
        "ViewerProtocolPolicy" : "allow-all",
        "MinTTL" : "100",
        "SmoothStreaming" : "true"
    },
    "CacheBehaviors" : [ {
        "AllowedMethods" : [ "DELETE", "GET", "HEAD", "OPTIONS",
"PATCH", "POST", "PUT" ],
        "TargetOriginId" : "myS3Origin",
        "ForwardedValues" : {
            "QueryString" : "true",
            "Cookies" : { "Forward" : "none" }
        },
        "TrustedSigners" : [ "1234567890EX", "1234567891EX" ],
        "ViewerProtocolPolicy" : "allow-all",
        "MinTTL" : "50",
        "PathPattern" : "images1/*.jpg"
    },
    {
        "AllowedMethods" : [ "DELETE", "GET", "HEAD", "OPTIONS",
"PATCH", "POST", "PUT" ],
        "TargetOriginId" : "myCustomOrigin",
        "ForwardedValues" : {
            "QueryString" : "true",
            "Cookies" : { "Forward" : "none" }
        },
        "TrustedSigners" : [ "1234567890EX", "1234567891EX" ],
        "ViewerProtocolPolicy" : "allow-all",
        "MinTTL" : "50",
        "PathPattern" : "images2/*.jpg"
    }
    ],
    "CustomErrorResponses" : [ {

```



```
TargetOriginId: myS3Origin
ForwardedValues:
  QueryString: 'false'
  Cookies:
    Forward: all
TrustedSigners:
- 1234567890EX
- 1234567891EX
ViewerProtocolPolicy: allow-all
MinTTL: '100'
SmoothStreaming: 'true'
CacheBehaviors:
- AllowedMethods:
  - DELETE
  - GET
  - HEAD
  - OPTIONS
  - PATCH
  - POST
  - PUT
  TargetOriginId: myS3Origin
  ForwardedValues:
    QueryString: 'true'
    Cookies:
      Forward: none
  TrustedSigners:
  - 1234567890EX
  - 1234567891EX
  ViewerProtocolPolicy: allow-all
  MinTTL: '50'
  PathPattern: images1/*.jpg
- AllowedMethods:
  - DELETE
  - GET
  - HEAD
  - OPTIONS
  - PATCH
  - POST
  - PUT
  TargetOriginId: myCustomOrigin
  ForwardedValues:
    QueryString: 'true'
    Cookies:
      Forward: none
```

```

    TrustedSigners:
      - 1234567890EX
      - 1234567891EX
    ViewerProtocolPolicy: allow-all
    MinTTL: '50'
    PathPattern: images2/*.jpg
  CustomErrorResponses:
    - ErrorCode: '404'
      ResponsePagePath: "/error-pages/404.html"
      ResponseCode: '200'
      ErrorCachingMinTTL: '30'
  PriceClass: PriceClass_All
  ViewerCertificate:
    CloudFrontDefaultCertificate: 'true'

```

Lambda 함수를 오리진으로 사용하여 Amazon CloudFront 배포

다음 예에서는 지정된 Lambda 함수 URL(파라미터로 제공됨)의 프런트가 되는 CloudFront 배포를 생성하여, HTTPS 전용 액세스, 캐싱, 압축 및 글로벌 전송을 활성화합니다. Lambda URL을 사용자 지정 HTTPS 오리진으로 구성하고 표준 AWS 캐싱 정책을 적용합니다. 이 배포는 HTTP/2 및 IPv6을 통해 최적화된 성능을 제공하며, CloudFront 도메인 이름을 출력하므로 사용자가 안전한 CDN 지원 엔드포인트를 통해 Lambda 함수에 액세스할 수 있습니다. 자세한 내용은 AWS 블로그에서 [Using Amazon CloudFront with AWS Lambda as origin to accelerate your web applications](#)를 참조하세요.

JSON

```

{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "LambdaEndpoint": {
      "Type": "String",
      "Description": "The Lambda function URL endpoint without the 'https://'"
    }
  },
  "Resources": {
    "MyDistribution": {
      "Type": "AWS::CloudFront::Distribution",
      "Properties": {
        "DistributionConfig": {
          "PriceClass": "PriceClass_All",
          "HttpVersion": "http2",
          "IPv6Enabled": true,

```

```

        "Origins": [
            {
                "DomainName": {
                    "Ref": "LambdaEndpoint"
                },
                "Id": "LambdaOrigin",
                "CustomOriginConfig": {
                    "HTTPSPort": 443,
                    "OriginProtocolPolicy": "https-only"
                }
            }
        ],
        "Enabled": "true",
        "DefaultCacheBehavior": {
            "TargetOriginId": "LambdaOrigin",
            "CachePolicyId": "658327ea-f89d-4fab-a63d-7e88639e58f6",
            "ViewerProtocolPolicy": "redirect-to-https",
            "SmoothStreaming": "false",
            "Compress": "true"
        }
    }
}
},
"Outputs": {
    "CloudFrontDomain": {
        "Description": "CloudFront default domain name configured",
        "Value": {
            "Fn::Sub": "https://${MyDistribution.DomainName}/"
        }
    }
}
}
}

```

YAML

```

AWSTemplateFormatVersion: 2010-09-09
Parameters:
  LambdaEndpoint:
    Type: String
    Description: The Lambda function URL endpoint without the 'https://'
Resources:
  MyDistribution:

```

```

Type: AWS::CloudFront::Distribution
Properties:
  DistributionConfig:
    PriceClass: PriceClass_All
    HttpVersion: http2
    IPV6Enabled: true
    Origins:
      - DomainName: !Ref LambdaEndpoint
        Id: LambdaOrigin
        CustomOriginConfig:
          HTTPSPort: 443
          OriginProtocolPolicy: https-only
    Enabled: 'true'
    DefaultCacheBehavior:
      TargetOriginId: LambdaOrigin
      CachePolicyId: '658327ea-f89d-4fab-a63d-7e88639e58f6'
      ViewerProtocolPolicy: redirect-to-https
      SmoothStreaming: 'false'
      Compress: 'true'
Outputs:
  CloudFrontDomain:
    Description: CloudFront default domain name configured
    Value: !Sub https://${MyDistribution.DomainName}/

```

다음 사항도 참조하세요.

Route 53 레코드에 사용자 지정 별칭을 추가하여 CloudFront 배포에 익숙한 이름을 지정하는 예는 [CloudFront 배포에 대한 별칭 리소스 레코드 세트](#) 섹션을 참조하세요.

Amazon CloudWatch 템플릿 코드 조각

CloudFormation에서 Amazon CloudWatch 리소스를 설명하는 데 도움이 되는 샘플 템플릿 스니펫을 사용합니다. 자세한 내용은 [Amazon CloudWatch 리소스 유형](#)을 참조하세요.

주제

- [결제 경보](#)
- [CPU 사용률 경보](#)
- [Amazon Elastic Compute Cloud 인스턴스 복구](#)
- [기본 대시보드 생성](#)
- [단계별 위젯이 있는 대시보드 생성](#)

결제 경보

다음 샘플에서 Amazon CloudWatch는 AWS 계정에 대한 요금이 경보 임계값을 초과할 경우 이메일 알림을 전송합니다. 사용량 알림을 수신하려면 결제 알림을 활성화하십시오. 자세한 내용은 Amazon CloudWatch 사용 설명서의 [청구 경보를 생성하여 예상 AWS 요금 모니터링](#)을 참조하세요.

JSON

```
"SpendingAlarm": {
  "Type": "AWS::CloudWatch::Alarm",
  "Properties": {
    "AlarmDescription": { "Fn::Join": ["", [
      "Alarm if AWS spending is over $",
      { "Ref": "AlarmThreshold" }
    ]]},
    "Namespace": "AWS/Billing",
    "MetricName": "EstimatedCharges",
    "Dimensions": [{
      "Name": "Currency",
      "Value" : "USD"
    }],
    "Statistic": "Maximum",
    "Period": "21600",
    "EvaluationPeriods": "1",
    "Threshold": { "Ref": "AlarmThreshold" },
    "ComparisonOperator": "GreaterThanThreshold",
    "AlarmActions": [{
      "Ref": "BillingAlarmNotification"
    }],
    "InsufficientDataActions": [{
      "Ref": "BillingAlarmNotification"
    }]
  }
}
```

YAML

```
SpendingAlarm:
  Type: AWS::CloudWatch::Alarm
  Properties:
    AlarmDescription:
      'Fn::Join':
        - ''
```

```

    - - Alarm if AWS spending is over $
      - !Ref: AlarmThreshold
Namespace: AWS/Billing
MetricName: EstimatedCharges
Dimensions:
- Name: Currency
  Value: USD
Statistic: Maximum
Period: '21600'
EvaluationPeriods: '1'
Threshold:
  !Ref: "AlarmThreshold"
ComparisonOperator: GreaterThanThreshold
AlarmActions:
- !Ref: "BillingAlarmNotification"
InsufficientDataActions:
- !Ref: "BillingAlarmNotification"

```

CPU 사용률 경고

다음 샘플 코드 조각은 Amazon EC2 인스턴스의 평균 CPU 사용률이 세 평가 기간 동안 60초 이상 90%를 초과할 경우 알림을 전송하는 경보를 생성합니다.

JSON

```

"CPULAlarm" : {
  "Type" : "AWS::CloudWatch::Alarm",
  "Properties" : {
    "AlarmDescription" : "CPU alarm for my instance",
    "AlarmActions" : [ { "Ref" : "logical name of an AWS::SNS::Topic resource" } ],
    "MetricName" : "CPUUtilization",
    "Namespace" : "AWS/EC2",
    "Statistic" : "Average",
    "Period" : "60",
    "EvaluationPeriods" : "3",
    "Threshold" : "90",
    "ComparisonOperator" : "GreaterThanThreshold",
    "Dimensions" : [ {
      "Name" : "InstanceId",
      "Value" : { "Ref" : "logical name of an AWS::EC2::Instance resource" }
    } ]
  }
}

```


YAML

```

CPUAlarm:
  Type: AWS::CloudWatch::Alarm
  Properties:
    AlarmDescription: CPU alarm for my instance
    AlarmActions:
      - !Ref: "logical name of an AWS::SNS::Topic resource"
    MetricName: CPUUtilization
    Namespace: AWS/EC2
    Statistic: Average
    Period: '60'
    EvaluationPeriods: '3'
    Threshold: '90'
    ComparisonOperator: GreaterThanThreshold
    Dimensions:
      - Name: InstanceId
        Value: !Ref: "logical name of an AWS::EC2::Instance resource"

```

Amazon Elastic Compute Cloud 인스턴스 복구

다음 CloudWatch 경보는 15분 동안 연속해서 상태 확인에 실패할 경우 EC2 인스턴스를 복구합니다. 경보 작업에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 [EC2 인스턴스를 중지, 종료, 재부팅 또는 복구하는 경보 생성](#)을 참조하세요.

JSON

```

{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters" : {
    "RecoveryInstance" : {
      "Description" : "The EC2 instance ID to associate this alarm with.",
      "Type" : "AWS::EC2::Instance::Id"
    }
  },
  "Resources": {
    "RecoveryTestAlarm": {
      "Type": "AWS::CloudWatch::Alarm",
      "Properties": {
        "AlarmDescription": "Trigger a recovery when instance status check fails for 15 consecutive minutes.",
        "Namespace": "AWS/EC2" ,

```

```

        "MetricName": "StatusCheckFailed_System",
        "Statistic": "Minimum",
        "Period": "60",
        "EvaluationPeriods": "15",
        "ComparisonOperator": "GreaterThanThreshold",
        "Threshold": "0",
        "AlarmActions": [ {"Fn::Join" : [ "", [ "arn:aws:automate:", { "Ref" :
"AWS::Region" }, ":ec2:recover" ] ] } ],
        "Dimensions": [{"Name": "InstanceId", "Value": {"Ref": "RecoveryInstance"}}]
    }
}
}
}

```

YAML

```

AWSTemplateFormatVersion: '2010-09-09'
Parameters:
  RecoveryInstance:
    Description: The EC2 instance ID to associate this alarm with.
    Type: AWS::EC2::Instance::Id
Resources:
  RecoveryTestAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmDescription: Trigger a recovery when instance status check fails for 15
        consecutive minutes.
      Namespace: AWS/EC2
      MetricName: StatusCheckFailed_System
      Statistic: Minimum
      Period: '60'
      EvaluationPeriods: '15'
      ComparisonOperator: GreaterThanThreshold
      Threshold: '0'
      AlarmActions: [ !Sub "arn:aws:automate:${AWS::Region}:ec2:recover" ]
      Dimensions:
        - Name: InstanceId
          Value: !Ref: RecoveryInstance

```

기본 대시보드 생성

다음 예에서는 CPU 사용률을 보여주는 측정치 위젯 하나와 메시지를 보여주는 텍스트 위젯 하나가 있는 간단한 CloudWatch 대시보드를 생성합니다.

JSON

```
{
  "BasicDashboard": {
    "Type": "AWS::CloudWatch::Dashboard",
    "Properties": {
      "DashboardName": "Dashboard1",
      "DashboardBody": "{\n  \"widgets\": [\n    {\n      \"type\": \"metric\",\n      \"x\": 0,\n      \"y\": 0,\n      \"width\": 12,\n      \"height\": 6,\n      \"properties\": {\n        \"metrics\": [\n          [\n            \"AWS/EC2\",\n            \"CPUUtilization\",\n            \"InstanceId\",\n            \"i-012345\"\n          ],\n          {\n            \"period\": 300,\n            \"stat\": \"Average\",\n            \"region\": \"us-east-1\",\n            \"title\": \"EC2 Instance CPU\"\n          }\n        ]\n      },\n      {\n        \"type\": \"text\",\n        \"x\": 0,\n        \"y\": 7,\n        \"width\": 3,\n        \"height\": 3,\n        \"properties\": {\n          \"markdown\": \"Hello world\"\n        }\n      }\n    ]\n  }\n}"
    }
  }
}
```

YAML

```
BasicDashboard:
  Type: AWS::CloudWatch::Dashboard
  Properties:
    DashboardName: Dashboard1
    DashboardBody: '{"widgets":
[{"type": "metric", "x": 0, "y": 0, "width": 12, "height": 6, "properties": {"metrics": [{"AWS/
EC2", "CPUUtilization", "InstanceId", "i-012345"}], "period": 300, "stat": "Average", "region": "us-
east-1", "title": "EC2 Instance CPU"}},
{"type": "text", "x": 0, "y": 7, "width": 3, "height": 3, "properties": {"markdown": "Hello
world"}}]}'
```

단계별 위젯이 있는 대시보드 생성

다음 예에서는 단계별로 표시되는 측정치 위젯 두 개가 있는 대시보드를 생성합니다.

JSON

```
{
  "DashboardSideBySide": {
    "Type": "AWS::CloudWatch::Dashboard",
    "Properties": {
      "DashboardName": "Dashboard1",
      "DashboardBody": "{\n  \"widgets\": [\n    {\n      \"type\": \"metric\",\n      \"x\": 0,\n      \"y\": 0,\n      \"width\": 12,\n      \"height\": 6,\n      \"properties\": {\n        \"metrics\": [\n          [\n            \"AWS/EC2\",\n            \"CPUUtilization"
```

```

\","InstanceId\","i-012345\"]] ,\ "period\ ":300,\ "stat\ ":"Average\ ",\ "region\ ":
\ "us-east-1\ ",\ "title\ ":"EC2 Instance CPU\ "}}, {\ "type\ ":"metric\ ",\ "x\ ":12,\ "y\ ":0,
\ "width\ ":12,\ "height\ ":6,\ "properties\ ": {\ "metrics\ ": [\ "AWS/S3\ ",\ "BucketSizeBytes\ ",
\ "BucketName\ ",\ "amzn-s3-demo-bucket\ "]] ,\ "period\ ":86400,\ "stat\ ":"Maximum\ ",\ "region
\ ":"us-east-1\ ",\ "title\ ":"amzn-s3-demo-bucket bytes\ "}}]]]"
    }
  }
}

```

YAML

```

DashboardSideBySide:
  Type: AWS::CloudWatch::Dashboard
  Properties:
    DashboardName: Dashboard1
    DashboardBody: '{ "widgets":
[{"type": "metric", "x": 0, "y": 0, "width": 12, "height": 6, "properties": {"metrics": [
["AWS/EC2", "CPUUtilization", "InstanceId", "i-012345"], "period": 300, "stat": "Average", "region": "us-
east-1", "title": "EC2 Instance CPU"}]},
{"type": "metric", "x": 12, "y": 0, "width": 12, "height": 6, "properties":
{"metrics": [
["AWS/S3", "BucketSizeBytes", "BucketName", "amzn-s3-demo-
bucket"], "period": 86400, "stat": "Maximum", "region": "us-east-1", "title": "amzn-s3-demo-
bucket bytes"]}]}]'

```

Amazon CloudWatch Logs 템플릿 코드 조각

Amazon CloudWatch Logs는 Amazon EC2 인스턴스 또는 기타 소스의 시스템, 애플리케이션 및 사용자 지정 로그 파일을 모니터링할 수 있습니다. CloudFormation을 사용하여 로그 그룹 및 지표 필터를 프로비저닝 및 관리할 수 있습니다. Amazon CloudWatch Logs에 대한 자세한 내용은 [Amazon CloudWatch Logs 사용자 설명서](#)를 참조하세요.

주제

- [Linux 인스턴스에서 CloudWatch Logs로 로그 전송](#)
- [Windows 인스턴스에서 CloudWatch Logs로 로그 전송](#)
- [다음 사항도 참조하세요.](#)


```
    "OperatorEmail": {
      "Description": "Email address to notify when CloudWatch alarms are
triggered (404 errors or high bandwidth usage)",
      "Type": "String"
    }
  },
  "Resources": {
    "LogRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": [
                  "ec2.amazonaws.com"
                ]
              },
              "Action": [
                "sts:AssumeRole"
              ]
            }
          ]
        },
        "Path": "/",
        "Policies": [
          {
            "PolicyName": "LogRolePolicy",
            "PolicyDocument": {
              "Version": "2012-10-17",
              "Statement": [
                {
                  "Effect": "Allow",
                  "Action": [
                    "logs:PutLogEvents",
                    "logs:DescribeLogStreams",
                    "logs:DescribeLogGroups",
                    "logs:CreateLogGroup",
                    "logs:CreateLogStream"
                  ],
                  "Resource": "*"
                }
              ]
            }
          }
        ]
      }
    }
  }
}
```

```

    ]
  }
}
]
}
},
"LogRoleInstanceProfile": {
  "Type": "AWS::IAM::InstanceProfile",
  "Properties": {
    "Path": "/",
    "Roles": [{"Ref": "LogRole"}]
  }
},
"WebServerSecurityGroup": {
  "Type": "AWS::EC2::SecurityGroup",
  "Properties": {
    "GroupDescription": "Enable HTTP access via port 80 and SSH access via
port 22",
    "SecurityGroupIngress": [
      {
        "IpProtocol": "tcp",
        "FromPort": 80,
        "ToPort": 80,
        "CidrIp": "0.0.0.0/0"
      },
      {
        "IpProtocol": "tcp",
        "FromPort": 22,
        "ToPort": 22,
        "CidrIp": {"Ref": "SSHLocation"}
      }
    ]
  }
},
"WebServerHost": {
  "Type": "AWS::EC2::Instance",
  "Metadata": {
    "Comment": "Install a simple PHP application on Amazon Linux 2023",
    "AWS::CloudFormation::Init": {
      "config": {
        "packages": {
          "dnf": {
            "httpd": [],
            "php": [],

```

```

        "php-fpm": []
    },
    "files": {
        "/etc/amazon-cloudwatch-agent/amazon-cloudwatch-
agent.json": {
            "content": {
                "logs": {
                    "logs_collected": {
                        "files": {
                            "collect_list": [{
                                "file_path": "/var/log/httpd/
access_log",
                                "log_group_name": {"Ref":
"WebServerLogGroup"},
                                "log_stream_name": "{instance_id}/
apache.log",
                                "timestamp_format": "%d/%b/%Y:%H:
%M:%S %z"
                            }]
                        }
                    }
                },
                "mode": "000644",
                "owner": "root",
                "group": "root"
            },
            "/var/www/html/index.php": {
                "content": "<?php\\necho '<h1>AWS CloudFormation sample
PHP application on Amazon Linux 2023</h1>';\\n?>\\n",
                "mode": "000644",
                "owner": "apache",
                "group": "apache"
            },
            "/etc/cfn/cfn-hup.conf": {
                "content": {
                    "Fn::Join": [
                        "",
                        [
                            "[main]\\n",
                            "stack=",
                            {"Ref": "AWS::StackId"},
                            "\\n",

```



```

        "region=",
        {"Ref": "AWS::Region"},
        "\n"
    ]
]
},
"mode": "000400",
"owner": "root",
"group": "root"
},
"/etc/cfn/hooks.d/cfn-auto-reloader.conf": {
    "content": {
        "Fn::Join": [
            "",
            [
                "[cfn-auto-reloader-hook]\n",
                "triggers=post.update\n",
                "\n",
                "path=Resources.WebServerHost.Metadata.AWS::CloudFormation::Init\n",
                "action=/opt/aws/bin/cfn-init -s ",
                {"Ref": "AWS::StackId"},
                " -r WebServerHost ",
                "--region ",
                {"Ref": "AWS::Region"},
                "\n",
                "runas=root\n"
            ]
        ]
    }
},
"services": {
    "systemd": {
        "httpd": {
            "enabled": "true",
            "ensureRunning": "true"
        },
        "php-fpm": {
            "enabled": "true",
            "ensureRunning": "true"
        }
    }
}
}

```

```

    }
  },
  "CreationPolicy": {
    "ResourceSignal": {
      "Timeout": "PT5M"
    }
  },
  "Properties": {
    "ImageId": "{{resolve:ssm:/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-6.1-x86_64}}",
    "KeyName": {"Ref": "KeyName"},
    "InstanceType": "t3.micro",
    "SecurityGroupIds": [{"Ref": "WebServerSecurityGroup"}],
    "IamInstanceProfile": {"Ref": "LogRoleInstanceProfile"},
    "UserData": {"Fn::Base64": {"Fn::Join": [ "", [
      "#!/bin/bash\n",
      "dnf update -y aws-cfn-bootstrap\n",
      "dnf install -y amazon-cloudwatch-agent\n",
      "/opt/aws/bin/cfn-init -v --stack ", {"Ref": "AWS::StackName"}, "
--resource WebServerHost --region ", {"Ref": "AWS::Region"}, "\n",
      "\n",
      "# Verify Apache log directory exists and create if needed\n",
      "mkdir -p /var/log/httpd\n",
      "\n",
      "# Start CloudWatch agent\n",
      "/opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -
a fetch-config -m ec2 -c file:/etc/amazon-cloudwatch-agent/amazon-cloudwatch-agent.json
-s\n",
      "\n",
      "# Signal success\n",
      "/opt/aws/bin/cfn-signal -e $? --stack ", {"Ref":
"AWS::StackName"}, " --resource WebServerHost --region ", {"Ref": "AWS::Region"}, "\n"
]]]]}
  }
},
  "WebServerLogGroup": {
    "Type": "AWS::Logs::LogGroup",
    "DeletionPolicy": "Retain",
    "UpdateReplacePolicy": "Retain",
    "Properties": {
      "RetentionInDays": 7
    }
  },
  "404MetricFilter": {

```

```

        "Type": "AWS::Logs::MetricFilter",
        "Properties": {
            "LogGroupName": {"Ref": "WebServerLogGroup"},
            "FilterPattern": "[ip, identity, user_id, timestamp, request,
status_code = 404, size, ...]",
            "MetricTransformations": [
                {
                    "MetricValue": "1",
                    "MetricNamespace": "test/404s",
                    "MetricName": "test404Count"
                }
            ]
        }
    },
    "BytesTransferredMetricFilter": {
        "Type": "AWS::Logs::MetricFilter",
        "Properties": {
            "LogGroupName": {"Ref": "WebServerLogGroup"},
            "FilterPattern": "[ip, identity, user_id, timestamp, request,
status_code, size, ...]",
            "MetricTransformations": [
                {
                    "MetricValue": "$size",
                    "MetricNamespace": "test/BytesTransferred",
                    "MetricName": "testBytesTransferred"
                }
            ]
        }
    },
    "404Alarm": {
        "Type": "AWS::CloudWatch::Alarm",
        "Properties": {
            "AlarmDescription": "The number of 404s is greater than 2 over 2
minutes",
            "MetricName": "test404Count",
            "Namespace": "test/404s",
            "Statistic": "Sum",
            "Period": "60",
            "EvaluationPeriods": "2",
            "Threshold": "2",
            "AlarmActions": [{"Ref": "AlarmNotificationTopic"}],
            "ComparisonOperator": "GreaterThanThreshold"
        }
    },

```

```

    "BandwidthAlarm": {
      "Type": "AWS::CloudWatch::Alarm",
      "Properties": {
        "AlarmDescription": "The average volume of traffic is greater 3500 KB
over 10 minutes",
        "MetricName": "testBytesTransferred",
        "Namespace": "test/BytesTransferred",
        "Statistic": "Average",
        "Period": "300",
        "EvaluationPeriods": "2",
        "Threshold": "3500",
        "AlarmActions": [{"Ref": "AlarmNotificationTopic"}],
        "ComparisonOperator": "GreaterThanThreshold"
      }
    },
    "AlarmNotificationTopic": {
      "Type": "AWS::SNS::Topic",
      "Properties": {
        "Subscription": [{"Endpoint": {"Ref": "OperatorEmail"}, "Protocol":
"email"}]
      }
    }
  },
  "Outputs": {
    "InstanceId": {
      "Description": "The instance ID of the web server",
      "Value": {"Ref": "WebServerHost"}
    },
    "WebsiteURL": {
      "Value": {"Fn::Sub": "http://${WebServerHost.PublicDnsName}"},
      "Description": "URL for the web server"
    },
    "PublicIP": {
      "Description": "Public IP address of the web server",
      "Value": {"Fn::GetAtt": ["WebServerHost", "PublicIp"]}
    },
    "CloudWatchLogGroupName": {
      "Description": "The name of the CloudWatch log group",
      "Value": {"Ref": "WebServerLogGroup"}
    }
  }
}

```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: Sample template that sets up and configures CloudWatch Logs on Amazon
  Linux 2023 instance.
Parameters:
  KeyName:
    Description: Name of an existing EC2 KeyPair to enable SSH access to the instances
    Type: AWS::EC2::KeyPair::KeyName
    ConstraintDescription: must be the name of an existing EC2 KeyPair.
  SSHLocation:
    Description: The IP address range that can be used to SSH to the EC2 instances
    Type: String
    MinLength: '9'
    MaxLength: '18'
    Default: 0.0.0.0/0
    AllowedPattern: '(\d{1,3})\.\d{1,3})\.\d{1,3})\.\d{1,3})/(\d{1,2})'
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.
  OperatorEmail:
    Description: Email address to notify when CloudWatch alarms are triggered (404
      errors or high bandwidth usage)
    Type: String
Resources:
  LogRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - ec2.amazonaws.com
            Action:
              - 'sts:AssumeRole'
      Path: /
    Policies:
      - PolicyName: LogRolePolicy
        PolicyDocument:
          Version: 2012-10-17
          Statement:
            - Effect: Allow
              Action:
                - 'logs:PutLogEvents'
```

```

        - 'logs:DescribeLogStreams'
        - 'logs:DescribeLogGroups'
        - 'logs:CreateLogGroup'
        - 'logs:CreateLogStream'
    Resource: '*'
LogRoleInstanceProfile:
  Type: AWS::IAM::InstanceProfile
  Properties:
    Path: /
    Roles:
      - !Ref LogRole
WebServerSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Enable HTTP access via port 80 and SSH access via port 22
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 80
        ToPort: 80
        CidrIp: 0.0.0.0/0
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: !Ref SSHLocation
WebServerHost:
  Type: AWS::EC2::Instance
  Metadata:
    Comment: Install a simple PHP application on Amazon Linux 2023
    'AWS::CloudFormation::Init':
      config:
        packages:
          dnf:
            httpd: []
            php: []
            php-fpm: []
        files:
          /etc/amazon-cloudwatch-agent/amazon-cloudwatch-agent.json:
            content: !Sub |
              {
                "logs": {
                  "logs_collected": {
                    "files": {
                      "collect_list": [
                        {

```

```

        "file_path": "/var/log/httpd/access_log",
        "log_group_name": "${WebServerLogGroup}",
        "log_stream_name": "{instance_id}/apache.log",
        "timestamp_format": "%d/%b/%Y:%H:%M:%S %z"
    }
  ]
}
}
}
mode: '000644'
owner: root
group: root
/var/www/html/index.php:
content: |
    <?php echo '<h1>AWS CloudFormation sample PHP application on Amazon
Linux 2023</h1>';
    ?>
mode: '000644'
owner: apache
group: apache
/etc/cfn/cfn-hup.conf:
content: !Sub |
    [main]
    stack=${AWS::StackId}
    region=${AWS::Region}
mode: '000400'
owner: root
group: root
/etc/cfn/hooks.d/cfn-auto-reloader.conf:
content: !Sub |
    [cfn-auto-reloader-hook]
    triggers=post.update
    path=Resources.WebServerHost.Metadata.AWS::CloudFormation::Init
    action=/opt/aws/bin/cfn-init -s ${AWS::StackId} -r WebServerHost --
region ${AWS::Region}
    runas=root
services:
  systemd:
    httpd:
      enabled: 'true'
      ensureRunning: 'true'
    php-fpm:
      enabled: 'true'

```

```

        ensureRunning: 'true'
CreationPolicy:
  ResourceSignal:
    Timeout: PT5M
Properties:
  ImageId: '{{resolve:ssm:/aws/service/ami-amazon-linux-latest/al2023-ami-
kernel-6.1-x86_64}}'
  KeyName: !Ref KeyName
  InstanceType: t3.micro
  SecurityGroupIds:
    - !Ref WebServerSecurityGroup
  IamInstanceProfile: !Ref LogRoleInstanceProfile
  UserData: !Base64
    Fn::Sub: |
      #!/bin/bash
      dnf update -y aws-cfn-bootstrap
      dnf install -y amazon-cloudwatch-agent
      /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource WebServerHost
--region ${AWS::Region}

      # Verify Apache log directory exists and create if needed
      mkdir -p /var/log/httpd

      # Start CloudWatch agent
      /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetch-
config -m ec2 -c file:/etc/amazon-cloudwatch-agent/amazon-cloudwatch-agent.json -s

      # Signal success
      /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource
WebServerHost --region ${AWS::Region}
      echo "Done"
WebServerLogGroup:
  Type: AWS::Logs::LogGroup
  DeletionPolicy: Retain
  UpdateReplacePolicy: Retain
  Properties:
    RetentionInDays: 7
404MetricFilter:
  Type: AWS::Logs::MetricFilter
  Properties:
    LogGroupName: !Ref WebServerLogGroup
    FilterPattern: >-
      [ip, identity, user_id, timestamp, request, status_code = 404, size, ...]
    MetricTransformations:

```



```
- MetricValue: '1'
  MetricNamespace: test/404s
  MetricName: test404Count
BytesTransferredMetricFilter:
  Type: AWS::Logs::MetricFilter
  Properties:
    LogGroupName: !Ref WebServerLogGroup
    FilterPattern: '[ip, identity, user_id, timestamp, request, status_code,
size, ...]'
    MetricTransformations:
      - MetricValue: $size
        MetricNamespace: test/BytesTransferred
        MetricName: testBytesTransferred
404Alarm:
  Type: AWS::CloudWatch::Alarm
  Properties:
    AlarmDescription: The number of 404s is greater than 2 over 2 minutes
    MetricName: test404Count
    Namespace: test/404s
    Statistic: Sum
    Period: '60'
    EvaluationPeriods: '2'
    Threshold: '2'
    AlarmActions:
      - !Ref AlarmNotificationTopic
    ComparisonOperator: GreaterThanThreshold
BandwidthAlarm:
  Type: AWS::CloudWatch::Alarm
  Properties:
    AlarmDescription: The average volume of traffic is greater 3500 KB over 10
minutes
    MetricName: testBytesTransferred
    Namespace: test/BytesTransferred
    Statistic: Average
    Period: '300'
    EvaluationPeriods: '2'
    Threshold: '3500'
    AlarmActions:
      - !Ref AlarmNotificationTopic
    ComparisonOperator: GreaterThanThreshold
AlarmNotificationTopic:
  Type: AWS::SNS::Topic
  Properties:
    Subscription:
```

```
- Endpoint: !Ref OperatorEmail
  Protocol: email
```

Outputs:

```
InstanceId:
  Description: The instance ID of the web server
  Value: !Ref WebServerHost
WebsiteURL:
  Value: !Sub 'http://${WebServerHost.PublicDnsName}'
  Description: URL for the web server
PublicIP:
  Description: Public IP address of the web server
  Value: !GetAtt WebServerHost.PublicIp
CloudWatchLogGroupName:
  Description: The name of the CloudWatch log group
  Value: !Ref WebServerLogGroup
```

Windows 인스턴스에서 CloudWatch Logs로 로그 전송

다음 템플릿은 Windows 2012R2 인스턴스에 대한 CloudWatch Logs를 구성합니다.

Windows의 CloudWatch Logs 에이전트(Windows 2012R2 및 Windows 2016 AMI의 SSM 에이전트)는 시작된 후에만 로그를 전송하므로 스타트업 전에 생성된 모든 로그는 전송되지 않습니다. 이를 해결하기 위해 템플릿은 로그를 작성하기 전에 에이전트가 시작되도록 합니다.

- `cfn-init config`의 첫 번째 `configSets` 항목으로 에이전트 설정을 구성합니다.
- `waitAfterCompletion`을 사용하여 에이전트를 시작하는 명령 뒤에 일시 중지를 삽입합니다.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Sample template that sets up and configures CloudWatch Logs on Windows 2012R2 instance.",
  "Parameters": {
    "KeyPair": {
      "Description": "Name of an existing EC2 KeyPair to enable RDP access to the instances",
      "Type": "AWS::EC2::KeyPair::KeyName",
      "ConstraintDescription": "must be the name of an existing EC2 KeyPair."
    },
    "RDPLocation": {
```

```
"Description": "The IP address range that can be used to RDP to the EC2 instances",
    "Type": "String",
    "MinLength": "9",
    "MaxLength": "18",
    "Default": "0.0.0.0/0",
    "AllowedPattern": "(\\d{1,3})\\.((\\d{1,3})|\\.((\\d{1,3})|\\.((\\d{1,3})|\\.((\\d{1,2})|\\.x.x.x.x/x."
  },
  "OperatorEmail": {
    "Description": "Email address to notify when CloudWatch alarms are triggered (404 errors)",
    "Type": "String"
  }
},
"Resources": {
  "WebServerSecurityGroup": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
      "GroupDescription": "Enable HTTP access via port 80 and RDP access via port 3389",
      "SecurityGroupIngress": [
        {
          "IpProtocol": "tcp",
          "FromPort": "80",
          "ToPort": "80",
          "CidrIp": "0.0.0.0/0"
        },
        {
          "IpProtocol": "tcp",
          "FromPort": "3389",
          "ToPort": "3389",
          "CidrIp": {"Ref": "RDPLocation"}
        }
      ]
    }
  },
  "LogRole": {
    "Type": "AWS::IAM::Role",
    "Properties": {
      "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
```

```

        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "Service": [
                        "ec2.amazonaws.com"
                    ]
                },
                "Action": [
                    "sts:AssumeRole"
                ]
            }
        ],
        "ManagedPolicyArns": [
            "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore"
        ],
        "Path": "/",
        "Policies": [
            {
                "PolicyName": "LogRolePolicy",
                "PolicyDocument": {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": [
                                "logs:Create*",
                                "logs:PutLogEvents",
                                "s3:GetObject"
                            ],
                            "Resource": [
                                "arn:aws:logs:*:*:*",
                                "arn:aws:s3:::*"
                            ]
                        }
                    ]
                }
            }
        ]
    },
    "LogRoleInstanceProfile": {
        "Type": "AWS::IAM::InstanceProfile",

```

```

        "Properties": {
            "Path": "/",
            "Roles": [{"Ref": "LogRole"}]
        },
    },
    "WebServerHost": {
        "Type": "AWS::EC2::Instance",
        "CreationPolicy": {
            "ResourceSignal": {
                "Timeout": "PT15M"
            }
        },
    },
    "Metadata": {
        "AWS::CloudFormation::Init": {
            "configSets": {
                "config": [
                    "00-ConfigureCWLogs",
                    "01-InstallWebServer",
                    "02-ConfigureApplication",
                    "03-Finalize"
                ]
            },
            "00-ConfigureCWLogs": {
                "files": {
                    "C:\\Program Files\\Amazon\\SSM\\Plugins\\awsCloudWatch\\
\\AWS.EC2.Windows.CloudWatch.json": {
                        "content": {
                            "EngineConfiguration": {
                                "Components": [
                                    {
                                        "FullName":
"\"AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch\"",
                                        "Id": "ApplicationEventLog",
                                        "Parameters": {
                                            "Levels": "7",
                                            "LogName": "Application"
                                        }
                                    },
                                    {
                                        "FullName":
"\"AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch\"",
                                        "Id": "SystemEventLog",
                                        "Parameters": {
                                            "Levels": "7",

```

```

        "LogName": "System"
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
        "Id": "SecurityEventLog",
        "Parameters": {
            "Levels": "7",
            "LogName": "Security"
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CustomLog.CustomLogInputComponent,AWS.EC2.Windows.CloudWatch",
        "Id": "EC2ConfigLog",
        "Parameters": {
            "CultureName": "en-US",
            "Encoding": "ASCII",
            "Filter": "EC2ConfigLog.txt",
            "LogDirectoryPath": "C:\\Program
Files\\Amazon\\Ec2ConfigService\\Logs",
            "TimeZoneKind": "UTC",
            "TimestampFormat": "yyyy-MM-
ddTHH:mm:ss.fffZ:"
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CustomLog.CustomLogInputComponent,AWS.EC2.Windows.CloudWatch",
        "Id": "CfnInitLog",
        "Parameters": {
            "CultureName": "en-US",
            "Encoding": "ASCII",
            "Filter": "cfn-init.log",
            "LogDirectoryPath": "C:\\cfn\\log",
            "TimeZoneKind": "Local",
            "TimestampFormat": "yyyy-MM-dd
HH:mm:ss,fff"
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CustomLog.CustomLogInputComponent,AWS.EC2.Windows.CloudWatch",

```

```

        "Id": "IISLogs",
        "Parameters": {
            "CultureName": "en-US",
            "Encoding": "UTF-8",
            "Filter": "",
            "LineCount": "3",
            "LogDirectoryPath": "C:\\inetpub\\
\\logs\\LogFiles\\W3SVC1",

            "TimeZoneKind": "UTC",
            "TimestampFormat": "yyyy-MM-dd
HH:mm:ss"
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.PerformanceCounterComponent.PerformanceCounterInputComponent,AWS.E
        "Id": "MemoryPerformanceCounter",
        "Parameters": {
            "CategoryName": "Memory",
            "CounterName": "Available MBytes",
            "DimensionName": "",
            "DimensionValue": "",
            "InstanceName": "",
            "MetricName": "Memory",
            "Unit": "Megabytes"
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
        "Id": "CloudWatchApplicationEventLog",
        "Parameters": {
            "AccessKey": "",
            "LogGroup": {"Ref": "LogGroup"},
            "LogStream": "{instance_id}/
ApplicationEventLog",

            "Region": {"Ref": "AWS::Region"},
            "SecretKey": ""
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
        "Id": "CloudWatchSystemEventLog",

```

```

        "Parameters": {
            "AccessKey": "",
            "LogGroup": {"Ref": "LogGroup"},
            "LogStream": "{instance_id}/
SystemEventLog",

            "Region": {"Ref": "AWS::Region"},
            "SecretKey": ""
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
        "Id": "CloudWatchSecurityEventLog",
        "Parameters": {
            "AccessKey": "",
            "LogGroup": {"Ref": "LogGroup"},
            "LogStream": "{instance_id}/
SecurityEventLog",

            "Region": {"Ref": "AWS::Region"},
            "SecretKey": ""
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
        "Id": "CloudWatchEC2ConfigLog",
        "Parameters": {
            "AccessKey": "",
            "LogGroup": {"Ref": "LogGroup"},
            "LogStream": "{instance_id}/
EC2ConfigLog",

            "Region": {"Ref": "AWS::Region"},
            "SecretKey": ""
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
        "Id": "CloudWatchCfnInitLog",
        "Parameters": {
            "AccessKey": "",
            "LogGroup": {"Ref": "LogGroup"},
            "LogStream": "{instance_id}/
CfnInitLog",

```



```

        "Region": {"Ref": "AWS::Region"},
        "SecretKey": ""
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
        "Id": "CloudWatchIISLogs",
        "Parameters": {
            "AccessKey": "",
            "LogGroup": {"Ref": "LogGroup"},
            "LogStream": "{instance_id}/
IISLogs",

            "Region": {"Ref": "AWS::Region"},
            "SecretKey": ""
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CloudWatch.CloudWatchOutputComponent,AWS.EC2.Windows.CloudWatch",
        "Id": "CloudWatch",
        "Parameters": {
            "AccessKey": "",
            "NameSpace": "Windows/Default",
            "Region": {"Ref": "AWS::Region"},
            "SecretKey": ""
        }
    }
],
"Flows": {
    "Flows": [

"ApplicationEventLog,CloudWatchApplicationEventLog",

"SystemEventLog,CloudWatchSystemEventLog",

"SecurityEventLog,CloudWatchSecurityEventLog",

"EC2ConfigLog,CloudWatchEC2ConfigLog",
"CfnInitLog,CloudWatchCfnInitLog",
"IISLogs,CloudWatchIISLogs",
"MemoryPerformanceCounter,CloudWatch"

    ]
},
"PollInterval": "00:00:05"

```

```

        },
        "IsEnabled": true
    }
}
},
"commands": {
    "0-enableSSM": {
        "command": "powershell.exe -Command \"Set-Service -Name
AmazonSSMAgent -StartupType Automatic\" ",
        "waitAfterCompletion": "0"
    },
    "1-restartSSM": {
        "command": "powershell.exe -Command \"Restart-Service
AmazonSSMAgent \"",
        "waitAfterCompletion": "30"
    }
},
"01-InstallWebServer": {
    "commands": {
        "01_install_webserver": {
            "command": "powershell.exe -Command \"Install-
WindowsFeature Web-Server -IncludeAllSubFeature\"",
            "waitAfterCompletion": "0"
        }
    }
},
"02-ConfigureApplication": {
    "files": {
        "c:\\Inetpub\\wwwroot\\index.htm": {
            "content": "<html> <head> <title>Test Application
Page</title> </head> <body> <h1>Congratulations!! Your IIS server is configured.</h1>
</body> </html>"
        }
    }
},
"03-Finalize": {
    "commands": {
        "00_signal_success": {
            "command": {
                "Fn::Sub": "cfn-signal.exe -e 0 --resource
WebServerHost --stack ${AWS::StackName} --region ${AWS::Region}"
            },
            "waitAfterCompletion": "0"
        }
    }
}

```

```

        }
    }
}
},
"Properties": {
    "KeyName": {
        "Ref": "KeyPair"
    },
    "ImageId": "{{resolve:ssm:/aws/service/ami-windows-latest/
Windows_Server-2012-R2_RTM-English-64Bit-Base}}",
    "InstanceType": "t2.xlarge",
    "SecurityGroupIds": [{"Ref": "WebServerSecurityGroup"}],
    "IamInstanceProfile": {"Ref": "LogRoleInstanceProfile"},
    "UserData": {
        "Fn::Base64": {
            "Fn::Join": [
                "",
                [
                    "<script>\n",
                    "wmic product where \"description='Amazon SSM Agent' \"
uninstall\n",
                    "wmic product where \"description='aws-cfn-bootstrap'
\" uninstall\n",
                    "start /wait c:\\\\Windows\\\\system32\\\\msiexec /passive /
qn /i https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-win64-
latest.msi\n",
                    "powershell.exe -Command \"iwr https://
s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/windows_amd64/
AmazonSSMAgentSetup.exe -UseBasicParsing -OutFile C:\\\\AmazonSSMAgentSetup.exe\\\"\\n",
                    "start /wait C:\\\\AmazonSSMAgentSetup.exe /install /
quiet\n",
                    "cfn-init.exe -v -c config -s ", {"Ref":
"AWS::StackName"}, " --resource WebServerHost --region ", {"Ref": "AWS::Region"}, "
\n",
                    "</script>\n"
                ]
            ]
        }
    },
    "LogGroup": {
        "Type": "AWS::Logs::LogGroup",

```

```

        "Properties": {
            "RetentionInDays": 7
        },
        "404MetricFilter": {
            "Type": "AWS::Logs::MetricFilter",
            "Properties": {
                "LogGroupName": {"Ref": "LogGroup"},
                "FilterPattern": "[timestamps, serverip, method, uri, query, port,
dash, clientip, useragent, status_code = 404, ...]",
                "MetricTransformations": [
                    {
                        "MetricValue": "1",
                        "MetricNamespace": "test/404s",
                        "MetricName": "test404Count"
                    }
                ]
            }
        },
        "404Alarm": {
            "Type": "AWS::CloudWatch::Alarm",
            "Properties": {
                "AlarmDescription": "The number of 404s is greater than 2 over 2
minutes",
                "MetricName": "test404Count",
                "Namespace": "test/404s",
                "Statistic": "Sum",
                "Period": "60",
                "EvaluationPeriods": "2",
                "Threshold": "2",
                "AlarmActions": [{"Ref": "AlarmNotificationTopic"}],
                "ComparisonOperator": "GreaterThanThreshold"
            }
        },
        "AlarmNotificationTopic": {
            "Type": "AWS::SNS::Topic",
            "Properties": {
                "Subscription": [{"Endpoint": {"Ref": "OperatorEmail"}, "Protocol":
"email"}]
            }
        },
        "Outputs": {
            "InstanceId": {

```

```

        "Description": "The instance ID of the web server",
        "Value": {"Ref": "WebServerHost"}
    },
    "WebsiteURL": {
        "Value": {"Fn::Sub": "http://${WebServerHost.PublicDnsName}"},
        "Description": "URL for the web server"
    },
    "PublicIP": {
        "Description": "Public IP address of the web server",
        "Value": {"Fn::GetAtt": ["WebServerHost", "PublicIp"]}
    },
    "CloudWatchLogGroupName": {
        "Description": "The name of the CloudWatch log group",
        "Value": {"Ref": "LogGroup"}
    }
}
}

```

YAML

```

AWSTemplateFormatVersion: 2010-09-09
Description: >-
  Sample template that sets up and configures CloudWatch Logs on Windows 2012R2
  instance.
Parameters:
  KeyPair:
    Description: Name of an existing EC2 KeyPair to enable RDP access to the instances
    Type: AWS::EC2::KeyPair::KeyName
    ConstraintDescription: must be the name of an existing EC2 KeyPair.
  RDPLocation:
    Description: The IP address range that can be used to RDP to the EC2 instances
    Type: String
    MinLength: '9'
    MaxLength: '18'
    Default: 0.0.0.0/0
    AllowedPattern: '(\d{1,3})\.\d{1,3})\.\d{1,3})\.\d{1,3})/(\d{1,2})'
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.
  OperatorEmail:
    Description: Email address to notify when CloudWatch alarms are triggered (404
    errors)
    Type: String
Resources:
  WebServerSecurityGroup:

```

Type: AWS::EC2::SecurityGroup

Properties:

GroupDescription: Enable HTTP access via port 80 and RDP access via port 3389

SecurityGroupIngress:

- IpProtocol: tcp
FromPort: '80'
ToPort: '80'
CidrIp: 0.0.0.0/0
- IpProtocol: tcp
FromPort: '3389'
ToPort: '3389'
CidrIp: !Ref RDPLocation

LogRole:

Type: AWS::IAM::Role

Properties:

AssumeRolePolicyDocument:

Version: 2012-10-17

Statement:

- Effect: Allow
Principal:
Service:
 - ec2.amazonaws.com

Action:

- 'sts:AssumeRole'

ManagedPolicyArns:

- 'arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore'

Path: /

Policies:

- PolicyName: LogRolePolicy

PolicyDocument:

Version: 2012-10-17

Statement:

- Effect: Allow
Action:
 - 'logs:Create*'
 - 'logs:PutLogEvents'
 - 's3:GetObject'
Resource:
 - 'arn:aws:logs:*:*:*'
 - 'arn:aws:s3:::*'

LogRoleInstanceProfile:

Type: AWS::IAM::InstanceProfile

Properties:

Path: /

```

    Roles:
      - !Ref LogRole
  WebServerHost:
    Type: AWS::EC2::Instance
    CreationPolicy:
      ResourceSignal:
        Timeout: PT15M
  Metadata:
    'AWS::CloudFormation::Init':
      configSets:
        config:
          - 00-ConfigureCWLogs
          - 01-InstallWebServer
          - 02-ConfigureApplication
          - 03-Finalize
      00-ConfigureCWLogs:
        files:
          'C:\Program Files\Amazon\SSM\Plugins\awsCloudWatch
\AWS.EC2.Windows.CloudWatch.json':
            content: !Sub |
              {
                "EngineConfiguration": {
                  "Components": [
                    {
                      "FullName":
"AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
                      "Id": "ApplicationEventLog",
                      "Parameters": {
                        "Levels": "7",
                        "LogName": "Application"
                      }
                    },
                    {
                      "FullName":
"AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
                      "Id": "SystemEventLog",
                      "Parameters": {
                        "Levels": "7",
                        "LogName": "System"
                      }
                    }
                  ]
                }
              }

```

```

        "Id": "SecurityEventLog",
        "Parameters": {
            "Levels": "7",
            "LogName": "Security"
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CustomLog.CustomLogInputComponent,AWS.EC2.Windows.CloudWatch",
        "Id": "EC2ConfigLog",
        "Parameters": {
            "CultureName": "en-US",
            "Encoding": "ASCII",
            "Filter": "EC2ConfigLog.txt",
            "LogDirectoryPath": "C:\\Program Files\\Amazon\\
\\Ec2ConfigService\\Logs",
            "TimeZoneKind": "UTC",
            "TimestampFormat": "yyyy-MM-ddTHH:mm:ss.fffZ:"
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CustomLog.CustomLogInputComponent,AWS.EC2.Windows.CloudWatch",
        "Id": "CfnInitLog",
        "Parameters": {
            "CultureName": "en-US",
            "Encoding": "ASCII",
            "Filter": "cfn-init.log",
            "LogDirectoryPath": "C:\\cfn\\log",
            "TimeZoneKind": "Local",
            "TimestampFormat": "yyyy-MM-dd HH:mm:ss,fff"
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CustomLog.CustomLogInputComponent,AWS.EC2.Windows.CloudWatch",
        "Id": "IISLogs",
        "Parameters": {
            "CultureName": "en-US",
            "Encoding": "UTF-8",
            "Filter": "",
            "LineCount": "3",
            "LogDirectoryPath": "C:\\inetpub\\logs\\LogFiles\\
\\W3SVC1",

```



```

        "TimeZoneKind": "UTC",
        "TimestampFormat": "yyyy-MM-dd HH:mm:ss"
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.PerformanceCounterComponent.PerformanceCounterInputComponent,AWS.E
        "Id": "MemoryPerformanceCounter",
        "Parameters": {
            "CategoryName": "Memory",
            "CounterName": "Available MBytes",
            "DimensionName": "",
            "DimensionValue": "",
            "InstanceName": "",
            "MetricName": "Memory",
            "Unit": "Megabytes"
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
        "Id": "CloudWatchApplicationEventLog",
        "Parameters": {
            "AccessKey": "",
            "LogGroup": "${LogGroup}",
            "LogStream": "{instance_id}/ApplicationEventLog",
            "Region": "${AWS::Region}",
            "SecretKey": ""
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
        "Id": "CloudWatchSystemEventLog",
        "Parameters": {
            "AccessKey": "",
            "LogGroup": "${LogGroup}",
            "LogStream": "{instance_id}/SystemEventLog",
            "Region": "${AWS::Region}",
            "SecretKey": ""
        }
    },
    {

```

```

        "FullName":
"AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
        "Id": "CloudWatchSecurityEventLog",
        "Parameters": {
            "AccessKey": "",
            "LogGroup": "${LogGroup}",
            "LogStream": "{instance_id}/SecurityEventLog",
            "Region": "${AWS::Region}",
            "SecretKey": ""
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
        "Id": "CloudWatchEC2ConfigLog",
        "Parameters": {
            "AccessKey": "",
            "LogGroup": "${LogGroup}",
            "LogStream": "{instance_id}/EC2ConfigLog",
            "Region": "${AWS::Region}",
            "SecretKey": ""
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
        "Id": "CloudWatchCfnInitLog",
        "Parameters": {
            "AccessKey": "",
            "LogGroup": "${LogGroup}",
            "LogStream": "{instance_id}/CfnInitLog",
            "Region": "${AWS::Region}",
            "SecretKey": ""
        }
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
        "Id": "CloudWatchIISLogs",
        "Parameters": {
            "AccessKey": "",
            "LogGroup": "${LogGroup}",
            "LogStream": "{instance_id}/IISLogs",
            "Region": "${AWS::Region}",

```

```

        "SecretKey": ""
    },
    {
        "FullName":
"AWS.EC2.Windows.CloudWatch.CloudWatch.CloudWatchOutputComponent,AWS.EC2.Windows.CloudWatch",
        "Id": "CloudWatch",
        "Parameters": {
            "AccessKey": "",
            "NameSpace": "Windows/Default",
            "Region": "${AWS::Region}",
            "SecretKey": ""
        }
    }
],
"Flows": {
    "Flows": [
        "ApplicationEventLog,CloudWatchApplicationEventLog",
        "SystemEventLog,CloudWatchSystemEventLog",
        "SecurityEventLog,CloudWatchSecurityEventLog",
        "EC2ConfigLog,CloudWatchEC2ConfigLog",
        "CfnInitLog,CloudWatchCfnInitLog",
        "IISLogs,CloudWatchIISLogs",
        "MemoryPerformanceCounter,CloudWatch"
    ]
},
"PollInterval": "00:00:05"
},
"IsEnabled": true
}
commands:
  0-enableSSM:
    command: >-
      powershell.exe -Command "Set-Service -Name AmazonSSMAgent
        -StartupType Automatic"
    waitAfterCompletion: '0'
  1-restartSSM:
    command: powershell.exe -Command "Restart-Service AmazonSSMAgent "
    waitAfterCompletion: '30'
01-InstallWebServer:
  commands:
    01_install_webserver:
      command: >-
        powershell.exe -Command "Install-WindowsFeature Web-Server

```

```

        -IncludeAllSubFeature"
        waitAfterCompletion: '0'
    02-ConfigureApplication:
        files:
            'c:\Inetpub\wwwroot\index.htm':
                content: >-
                    <html> <head> <title>Test Application Page</title> </head>
                    <body> <h1>Congratulations !! Your IIS server is
                    configured.</h1> </body> </html>
    03-Finalize:
        commands:
            00_signal_success:
                command: !Sub >-
                    cfn-signal.exe -e 0 --resource WebServerHost --stack
                    ${AWS::StackName} --region ${AWS::Region}
                waitAfterCompletion: '0'
Properties:
    KeyName: !Ref KeyPair
    ImageId: "{{resolve:ssm:/aws/service/ami-windows-latest/Windows_Server-2012-
R2_RTM-English-64Bit-Base}}"
    InstanceType: t2.xlarge
    SecurityGroupIds:
        - !Ref WebServerSecurityGroup
    IamInstanceProfile: !Ref LogRoleInstanceProfile
    UserData: !Base64
        'Fn::Sub': >
            <script>

            wmic product where "description='Amazon SSM Agent' " uninstall

            wmic product where "description='aws-cfn-bootstrap' " uninstall

            start /wait c:\Windows\system32\msiexec /passive /qn /i
            https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-win64-
latest.msi

            powershell.exe -Command "iwr
            https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/windows_amd64/
AmazonSSMAgentSetup.exe
            -UseBasicParsing -OutFile C:\\AmazonSSMAgentSetup.exe"

            start /wait C:\\AmazonSSMAgentSetup.exe /install /quiet

            cfn-init.exe -v -c config -s ${AWS::StackName} --resource

```

```

    WebServerHost --region ${AWS::Region}

</script>
LogGroup:
  Type: AWS::Logs::LogGroup
  Properties:
    RetentionInDays: 7
404MetricFilter:
  Type: AWS::Logs::MetricFilter
  Properties:
    LogGroupName: !Ref LogGroup
    FilterPattern: >-
      [timestamps, serverip, method, uri, query, port, dash, clientip,
       useragent, status_code = 404, ...]
    MetricTransformations:
      - MetricValue: '1'
        MetricNamespace: test/404s
        MetricName: test404Count
404Alarm:
  Type: AWS::CloudWatch::Alarm
  Properties:
    AlarmDescription: The number of 404s is greater than 2 over 2 minutes
    MetricName: test404Count
    Namespace: test/404s
    Statistic: Sum
    Period: '60'
    EvaluationPeriods: '2'
    Threshold: '2'
    AlarmActions:
      - !Ref AlarmNotificationTopic
    ComparisonOperator: GreaterThanThreshold
AlarmNotificationTopic:
  Type: AWS::SNS::Topic
  Properties:
    Subscription:
      - Endpoint: !Ref OperatorEmail
        Protocol: email
Outputs:
  InstanceId:
    Description: The instance ID of the web server
    Value: !Ref WebServerHost
  WebsiteURL:
    Value: !Sub 'http://${WebServerHost.PublicDnsName}'
    Description: URL for the web server

```

```

PublicIP:
  Description: Public IP address of the web server
  Value: !GetAtt
    - WebServerHost
    - PublicIp
CloudWatchLogGroupName:
  Description: The name of the CloudWatch log group
  Value: !Ref LogGroup

```

다음 사항도 참조하세요.

CloudWatch Logs 리소스에 대한 자세한 내용을 알아보려면 [AWS::Logs::LogGroup](#) 또는 [AWS::Logs::MetricFilter](#)를 참조하세요.

Amazon DynamoDB 템플릿 코드 조각

주제

- [Amazon DynamoDB 테이블을 통한 Application Auto Scaling](#)
- [다음 사항도 참조하세요.](#)

Amazon DynamoDB 테이블을 통한 Application Auto Scaling

이 예에서는 `AWS::DynamoDB::Table` 리소스에 대해 Application Auto Scaling을 설정합니다. 이 템플릿은 테이블에 대한 TargetTrackingScaling 처리량을 확장하는 WriteCapacityUnits 조정 정책을 정의합니다.

JSON

```

{
  "Resources": {
    "DDBTable": {
      "Type": "AWS::DynamoDB::Table",
      "Properties": {
        "AttributeDefinitions": [
          {
            "AttributeName": "ArtistId",
            "AttributeType": "S"
          },
          {
            "AttributeName": "Concert",
            "AttributeType": "S"
          }
        ]
      }
    }
  }
}

```

```
    },
    {
      "AttributeName": "TicketSales",
      "AttributeType": "S"
    }
  ],
  "KeySchema": [
    {
      "AttributeName": "ArtistId",
      "KeyType": "HASH"
    },
    {
      "AttributeName": "Concert",
      "KeyType": "RANGE"
    }
  ],
  "GlobalSecondaryIndexes": [
    {
      "IndexName": "GSI",
      "KeySchema": [
        {
          "AttributeName": "TicketSales",
          "KeyType": "HASH"
        }
      ],
      "Projection": {
        "ProjectionType": "KEYS_ONLY"
      },
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 5,
        "WriteCapacityUnits": 5
      }
    }
  ],
  "ProvisionedThroughput": {
    "ReadCapacityUnits": 5,
    "WriteCapacityUnits": 5
  }
},
"WriteCapacityScalableTarget": {
  "Type": "AWS::ApplicationAutoScaling::ScalableTarget",
  "Properties": {
    "MaxCapacity": 15,
```

```

        "MinCapacity": 5,
        "ResourceId": {
            "Fn::Join": [
                "/",
                [
                    "table",
                    {
                        "Ref": "DDBTable"
                    }
                ]
            ],
            "RoleARN" : { "Fn::Sub" : "arn:aws:iam::${AWS::AccountId}:role/
aws-service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable" },
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "ServiceNamespace": "dynamodb"
        }
    },
    "WriteScalingPolicy": {
        "Type": "AWS::ApplicationAutoScaling::ScalingPolicy",
        "Properties": {
            "PolicyName": "WriteAutoScalingPolicy",
            "PolicyType": "TargetTrackingScaling",
            "ScalingTargetId": {
                "Ref": "WriteCapacityScalableTarget"
            },
            "TargetTrackingScalingPolicyConfiguration": {
                "TargetValue": 50,
                "ScaleInCooldown": 60,
                "ScaleOutCooldown": 60,
                "PredefinedMetricSpecification": {
                    "PredefinedMetricType": "DynamoDBWriteCapacityUtilization"
                }
            }
        }
    }
}

```

YAML

Resources:


```

DDBTable:
  Type: AWS::DynamoDB::Table
  Properties:
    AttributeDefinitions:
      - AttributeName: "ArtistId"
        AttributeType: "S"
      - AttributeName: "Concert"
        AttributeType: "S"
      - AttributeName: "TicketSales"
        AttributeType: "S"
    KeySchema:
      - AttributeName: "ArtistId"
        KeyType: "HASH"
      - AttributeName: "Concert"
        KeyType: "RANGE"
    GlobalSecondaryIndexes:
      - IndexName: "GSI"
        KeySchema:
          - AttributeName: "TicketSales"
            KeyType: "HASH"
        Projection:
          ProjectionType: "KEYS_ONLY"
        ProvisionedThroughput:
          ReadCapacityUnits: 5
          WriteCapacityUnits: 5
    ProvisionedThroughput:
      ReadCapacityUnits: 5
      WriteCapacityUnits: 5
  WriteCapacityScalableTarget:
    Type: AWS::ApplicationAutoScaling::ScalableTarget
    Properties:
      MaxCapacity: 15
      MinCapacity: 5
      ResourceId: !Join
        - /
        - - table
          - !Ref DDBTable
      RoleARN:
        Fn::Sub: 'arn:aws:iam::${AWS::AccountId}:role/aws-
service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable'
      ScalableDimension: dynamodb:table:WriteCapacityUnits
      ServiceNamespace: dynamodb
  WriteScalingPolicy:

```

```
Type: AWS::ApplicationAutoScaling::ScalingPolicy
Properties:
  PolicyName: WriteAutoScalingPolicy
  PolicyType: TargetTrackingScaling
  ScalingTargetId: !Ref WriteCapacityScalableTarget
  TargetTrackingScalingPolicyConfiguration:
    TargetValue: 50.0
    ScaleInCooldown: 60
    ScaleOutCooldown: 60
    PredefinedMetricSpecification:
      PredefinedMetricType: DynamoDBWriteCapacityUtilization
```

다음 사항도 참조하세요.

자세한 내용은 AWS 데이터베이스 블로그의 [CloudFormation을 사용하여 DynamoDB 테이블 및 인덱스에 대한 Auto Scaling을 구성하는 방법](#) 게시물을 참조하세요.

DynamoDB 리소스에 대한 자세한 내용은 [AWS::DynamoDB::Table](#)을 참조하십시오.

Amazon EC2 CloudFormation 템플릿 코드 조각

Amazon EC2는 AWS 클라우드에서 확장 가능한 컴퓨팅 용량을 제공합니다. Amazon EC2를 사용하여 원하는 수의 가상 서버를 빌드하고 보안 및 네트워킹을 구성하며 스토리지를 관리할 수 있습니다. 인스턴스라고 하는 이러한 가상 서버는 다양한 운영 체제와 애플리케이션을 실행할 수 있으며 특정 요구 사항에 맞게 사용자 지정할 수 있습니다. Amazon EC2를 사용하면 손쉽게 규모를 확장 또는 축소하여 요구 사항 변경이나 사용량 급증에 대처할 수 있습니다.

CloudFormation 템플릿을 사용하여 Amazon EC2 인스턴스를 인프라의 일부로 정의하고 프로비저닝할 수 있습니다. 템플릿을 사용하면 반복 가능하고 일관된 방식으로 Amazon EC2 리소스 배포를 쉽게 관리하고 자동화할 수 있습니다.

다음 예제 템플릿 코드 조각은 Amazon EC2용 CloudFormation 리소스 또는 구성 요소를 설명합니다. 이러한 코드 조각은 템플릿에 통합되도록 설계되었으며 독립적으로 실행되도록 설계되지 않았습니다.

코드 조각 카테고리

- [CloudFormation을 사용하여 Amazon EC2 인스턴스 구성](#)
- [CloudFormation을 사용하여 시작 템플릿 생성](#)
- [CloudFormation을 사용하여 보안 그룹 관리](#)
- [CloudFormation을 사용하여 탄력적 IP 주소 할당 및 연결](#)
- [CloudFormation을 사용하여 Amazon VPC 리소스 구성](#)

CloudFormation을 사용하여 Amazon EC2 인스턴스 구성

다음 코드 조각은 CloudFormation을 사용하여 Amazon EC2 인스턴스를 구성하는 방법을 보여줍니다.

코드 조각 카테고리

- [일반적인 Amazon EC2 구성](#)
- [인스턴스의 블록 디바이스 매핑 지정](#)

일반적인 Amazon EC2 구성

다음 코드 조각은 CloudFormation을 사용하는 Amazon EC2 인스턴스의 일반적인 구성을 보여줍니다.

코드 조각 예제

- [지정된 가용 영역에서 Amazon EC2 인스턴스 생성](#)
- [EBS 볼륨 및 사용자 데이터를 사용하여 태그가 지정된 Amazon EC2 인스턴스 구성](#)
- [Amazon EC2 인스턴스 시작을 위한 사용자 데이터에 DynamoDB 테이블 이름 정의](#)
- [DeletionPolicy을\(를\) 사용하여 Amazon EBS 볼륨 생성](#)

지정된 가용 영역에서 Amazon EC2 인스턴스 생성

다음 코드 조각은 [AWS::EC2::Instance](#) 리소스를 사용하여 지정된 가용 영역에서 Amazon EC2 인스턴스를 생성합니다. 가용 영역의 코드는 리전 코드와 문자 식별자를 조합한 것입니다. 단일 가용 영역에서 인스턴스를 시작할 수 있습니다.

JSON

```
"Ec2Instance": {
  "Type": "AWS::EC2::Instance",
  "Properties": {
    "AvailabilityZone": "aa-example-1a",
    "ImageId": "ami-1234567890abcdef0"
  }
}
```

YAML

```
Ec2Instance:
  Type: AWS::EC2::Instance
  Properties:
```

```
AvailabilityZone: aa-example-1a
ImageId: ami-1234567890abcdef0
```

EBS 볼륨 및 사용자 데이터를 사용하여 태그가 지정된 Amazon EC2 인스턴스 구성

다음 코드 조각은 태그, EBS 볼륨 및 사용자 데이터를 사용하여 Amazon EC2 인스턴스를 생성합니다. [AWS::EC2::Instance](#) 리소스를 사용합니다. 동일한 템플릿에서 [AWS::EC2::SecurityGroup](#) 리소스, [AWS::SNS::Topic](#) 리소스 및 [AWS::EC2::Volume](#) 리소스를 정의해야 합니다. KeyName은(는) 템플릿의 Parameters 섹션에서 정의해야 합니다.

태그를 사용하면 용도, 소유자 또는 환경 등 원하는 기준에 따라 AWS 리소스를 분류할 수 있습니다. 사용자 데이터를 사용하면 시작 중에 인스턴스에 사용자 지정 스크립트 또는 데이터를 프로비저닝할 수 있습니다. 이 데이터는 초기화 중에 인스턴스에 대한 작업 자동화, 소프트웨어 구성, 패키지 설치 및 기타 작업을 용이하게 합니다.

리소스 태그 지정에 대한 자세한 내용은 Amazon EC2 사용 설명서의 [Amazon EC2 리소스 태깅](#)을 참조하세요.

사용자 데이터에 대한 자세한 내용은 Amazon EC2 사용 설명서의 [인스턴스 메타데이터를 사용하여 EC2 인스턴스를 관리합니다](#)를 참조하세요.

JSON

```
"Ec2Instance": {
  "Type": "AWS::EC2::Instance",
  "Properties": {
    "KeyName": { "Ref": "KeyName" },
    "SecurityGroups": [ { "Ref": "Ec2SecurityGroup" } ],
    "UserData": {
      "Fn::Base64": {
        "Fn::Join": [ ":", [
          "PORT=80",
          "TOPIC=",
          { "Ref": "MySNSTopic" }
        ]
        ]
      }
    },
    "InstanceType": "aa.size",
    "AvailabilityZone": "aa-example-1a",
    "ImageId": "ami-1234567890abcdef0",
    "Volumes": [
```

```

    {
      "VolumeId": { "Ref": "MyVolumeResource" },
      "Device": "/dev/sdk"
    }
  ],
  "Tags": [ { "Key": "Name", "Value": "MyTag" } ]
}
}

```

YAML

```

Ec2Instance:
  Type: AWS::EC2::Instance
  Properties:
    KeyName: !Ref KeyName
    SecurityGroups:
      - !Ref Ec2SecurityGroup
    UserData:
      Fn::Base64:
        Fn::Join:
          - ":"
          - - "PORT=80"
            - "TOPIC="
            - !Ref MySNSTopic
    InstanceType: aa.size
    AvailabilityZone: aa-example-1a
    ImageId: ami-1234567890abcdef0
    Volumes:
      - VolumeId: !Ref MyVolumeResource
        Device: "/dev/sdk"
    Tags:
      - Key: Name
        Value: MyTag

```

Amazon EC2 인스턴스 시작을 위한 사용자 데이터에 DynamoDB 테이블 이름 정의

다음 코드 조각은 Amazon EC2 인스턴스를 생성하고 시작 시 인스턴스에 전달할 DynamoDB 테이블 이름을 사용자 데이터에 정의합니다. [AWS::EC2::Instance](#) 리소스를 사용합니다. 사용자 데이터에 파라미터 또는 동적 값을 정의하여 시작 시 EC2 인스턴스를 전달할 수 있습니다.

사용자 데이터에 대한 자세한 내용은 Amazon EC2 사용 설명서의 [인스턴스 메타데이터를 사용하여 EC2 인스턴스를 관리합니다](#)를 참조하세요.

JSON

```

"Ec2Instance": {
  "Type": "AWS::EC2::Instance",
  "Properties": {
    "UserData": {
      "Fn::Base64": {
        "Fn::Join": [
          "",
          [
            "TableName=",
            {
              "Ref": "DynamoDBTableName"
            }
          ]
        ]
      },
      "AvailabilityZone": "aa-example-1a",
      "ImageId": "ami-1234567890abcdef0"
    }
  }
}

```

YAML

```

Ec2Instance:
  Type: AWS::EC2::Instance
  Properties:
    UserData:
      Fn::Base64:
        Fn::Join:
          - ''
          - - 'TableName='
            - Ref: DynamoDBTableName
    AvailabilityZone: aa-example-1a
    ImageId: ami-1234567890abcdef0

```

DeletionPolicy을(를) 사용하여 Amazon EBS 볼륨 생성

다음 코드 조각은 Amazon EC2 [AWS::EC2::Volume](#) 리소스를 사용하여 Amazon EBS 볼륨을 생성합니다. Size 또는 SnapshotID 속성을 사용하여 볼륨을 정의할 수 있지만 둘 다 사용할 수는 없습니다. 스택을 삭제할 때 볼륨의 스냅샷을 생성하도록 DeletionPolicy 속성이 설정됩니다.

DeletionPolicy 속성에 대한 자세한 내용은 [DeletionPolicy 속성](#)을 참조하세요.

Amazon EBS 볼륨 생성에 대한 자세한 내용은 [Amazon EBS 볼륨 생성](#)을 참조하세요.

JSON

이 코드 조각은 지정된 크기로 Amazon EBS 볼륨을 생성합니다. 크기는 10으로 설정되어 있지만 필요에 따라 조정할 수 있습니다. [AWS::EC2::Volume](#) 리소스를 사용하면 크기 또는 스냅샷 ID를 지정할 수 있지만 둘 다 지정할 수는 없습니다.

```
"MyEBSVolume": {
  "Type": "AWS::EC2::Volume",
  "Properties": {
    "Size": "10",
    "AvailabilityZone": {
      "Ref": "AvailabilityZone"
    }
  },
  "DeletionPolicy": "Snapshot"
}
```

이 코드 조각은 제공된 스냅샷 ID를 사용하여 Amazon EBS 볼륨을 생성합니다. [AWS::EC2::Volume](#) 리소스를 사용하면 크기 또는 스냅샷 ID를 지정할 수 있지만 둘 다 지정할 수는 없습니다.

```
"MyEBSVolume": {
  "Type": "AWS::EC2::Volume",
  "Properties": {
    "SnapshotId" : "snap-1234567890abcdef0",
    "AvailabilityZone": {
      "Ref": "AvailabilityZone"
    }
  },
  "DeletionPolicy": "Snapshot"
}
```

YAML

이 코드 조각은 지정된 크기로 Amazon EBS 볼륨을 생성합니다. 크기는 10으로 설정되어 있지만 필요에 따라 조정할 수 있습니다. [AWS::EC2::Volume](#) 리소스를 사용하면 크기 또는 스냅샷 ID를 지정할 수 있지만 둘 다 지정할 수는 없습니다.

```
MyEBSVolume:
```

```
Type: AWS::EC2::Volume
Properties:
  Size: 10
  AvailabilityZone:
    Ref: AvailabilityZone
  DeletionPolicy: Snapshot
```

이 코드 조각은 제공된 스냅샷 ID를 사용하여 Amazon EBS 볼륨을 생성합니다. [AWS::EC2::Volume](#) 리소스를 사용하면 크기 또는 스냅샷 ID를 지정할 수 있지만 둘 다 지정할 수는 없습니다.

```
MyEBSVolume:
  Type: AWS::EC2::Volume
  Properties:
    SnapshotId: snap-1234567890abcdef0
    AvailabilityZone:
      Ref: AvailabilityZone
    DeletionPolicy: Snapshot
```

인스턴스의 블록 디바이스 매핑 지정

블록 디바이스 매핑은 인스턴스에 연결할 블록 디바이스(인스턴스 스토어 볼륨 및 EBS 볼륨 포함)를 정의합니다. AMI를 생성할 때 블록 디바이스 매핑을 지정하면 AMI에서 실행되는 모든 인스턴스가 해당 매핑을 사용할 수 있습니다. 또는 인스턴스를 실행할 때 블록 디바이스 매핑을 지정하여 해당 매핑이 인스턴스가 실행된 AMI에서 지정된 매핑을 재정의하도록 할 수 있습니다.

다음 템플릿 코드 조각을 사용하면 [AWS::EC2::Instance](#) 리소스의 BlockDeviceMappings 속성을 사용하여 EBS 또는 인스턴스 스토어 볼륨의 블록 디바이스 매핑을 지정할 수 있습니다.

블록 디바이스 매핑에 대한 자세한 내용은 Amazon EC2 사용 설명서의 [Amazon EC2 인스턴스의 볼륨에 대한 블록 디바이스 매핑](#)을 참조하세요.

시나리오

- [두 EBS 볼륨의 블록 디바이스 매핑 지정](#)
- [인스턴스 스토어 볼륨의 인스턴스 블록 디바이스 매핑 지정](#)

두 EBS 볼륨의 블록 디바이스 매핑 지정

JSON

```
"Ec2Instance": {
```



```

    "Type": "AWS::EC2::Instance",
    "Properties": {
      "ImageId": "{{resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-
x86_64-gp2}}",
      "KeyName": { "Ref": "KeyName" },
      "InstanceType": { "Ref": "InstanceType" },
      "SecurityGroups": [{ "Ref": "Ec2SecurityGroup" }],
      "BlockDeviceMappings": [
        {
          "DeviceName": "/dev/sda1",
          "Ebs": { "VolumeSize": "50" }
        },
        {
          "DeviceName": "/dev/sdm",
          "Ebs": { "VolumeSize": "100" }
        }
      ]
    }
  }
}

```

YAML

```

EC2Instance:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: '{{resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-
x86_64-gp2}}'
    KeyName: !Ref KeyName
    InstanceType: !Ref InstanceType
    SecurityGroups:
      - !Ref Ec2SecurityGroup
    BlockDeviceMappings:
      -
        DeviceName: /dev/sda1
        Ebs:
          VolumeSize: 50
      -
        DeviceName: /dev/sdm
        Ebs:
          VolumeSize: 100

```

인스턴스 스토어 볼륨의 인스턴스 블록 디바이스 매핑 지정

JSON

```
"Ec2Instance" : {
  "Type" : "AWS::EC2::Instance",
  "Properties" : {
    "ImageId" : "{{resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2}}",
    "KeyName" : { "Ref" : "KeyName" },
    "InstanceType": { "Ref": "InstanceType" },
    "SecurityGroups" : [ { "Ref" : "Ec2SecurityGroup" } ],
    "BlockDeviceMappings" : [
      {
        "DeviceName" : "/dev/sdc",
        "VirtualName" : "ephemeral0"
      }
    ]
  }
}
```

YAML

```
EC2Instance:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: '{{resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2}}'
    KeyName: !Ref KeyName
    InstanceType: !Ref InstanceType
    SecurityGroups:
      - !Ref Ec2SecurityGroup
    BlockDeviceMappings:
      - DeviceName: /dev/sdc
        VirtualName: ephemeral0
```

CloudFormation을 사용하여 시작 템플릿 생성

이 섹션에서는 CloudFormation을 사용하여 Amazon EC2 시작 템플릿을 생성하는 예를 제공합니다. 시작 템플릿을 사용하면 AWS 내에서 Amazon EC2 인스턴스를 구성하고 프로비저닝하기 위한 템플릿을 생성할 수 있습니다. 시작 템플릿을 사용하면 인스턴스를 시작할 때마다 지정할 필요가 없도록 시작

파라미터를 저장할 수 있습니다. 더 많은 예는 `AWS::EC2::LaunchTemplate` 리소스의 [예](#) 섹션을 참조하세요.

시작 템플릿에 대한 자세한 내용은 Amazon EC2 사용 설명서의 [Amazon EC2 시작 템플릿에 인스턴스 시작 파라미터 저장](#)을 참조하세요.

Auto Scaling 그룹에 사용할 시작 템플릿을 생성하는 방법에 대한 자세한 내용은 Amazon EC2 Auto Scaling 사용 설명서의 [Auto Scaling 시작 템플릿](#)을 참조하세요.

코드 조각 카테고리

- [보안 그룹, 태그, 사용자 데이터와 IAM 역할을 지정하는 시작 템플릿 생성](#)

보안 그룹, 태그, 사용자 데이터와 IAM 역할을 지정하는 시작 템플릿 생성

이 코드 조각은 인스턴스를 시작하기 위한 구성 정보를 포함하는 `AWS::EC2::LaunchTemplate` 리소스를 보여줍니다. `ImageId`, `InstanceType`, `SecurityGroups`, `UserData` 및 `TagSpecifications` 속성의 값을 지정합니다. `SecurityGroups` 속성은 기존 EC2 보안 그룹과 새 보안 그룹을 지정합니다. `Ref` 함수는 스택 템플릿의 다른 곳에 선언된 `AWS::EC2::SecurityGroup` 리소스 `myNewEC2SecurityGroup`의 ID를 가져옵니다.

시작 템플릿에는 사용자 정의 사용자 데이터에 대한 부분이 포함되어 있습니다. 이 섹션에서 인스턴스가 시작될 때 실행되는 구성 태스크 및 스크립트를 전달할 수 있습니다. 이 예제에서 사용자 데이터는 AWS Systems Manager Agent를 설치하고 시작합니다.

시작 템플릿에는 인스턴스에서 실행되는 애플리케이션이 사용자를 대신하여 작업을 수행할 수 있도록 하는 IAM 역할도 포함되어 있습니다. 이 예제는 `IamInstanceProfile` 속성을 사용하여 IAM 역할을 지정하는 시작 템플릿에 대한 `AWS::IAM::Role` 리소스를 보여줍니다. `Ref` 함수는 `AWS::IAM::InstanceProfile` 리소스 `myInstanceProfile`의 이름을 가져옵니다. IAM 역할의 권한을 구성하려면 `ManagedPolicyArns` 속성 값을 지정합니다.

JSON

```
{
  "Resources":{
    "myLaunchTemplate":{
      "Type":"AWS::EC2::LaunchTemplate",
      "Properties":{
        "LaunchTemplateName":{"Fn::Sub": "${AWS::StackName}-launch-template" },
        "LaunchTemplateData":{
          "ImageId":"ami-02354e95b3example",
          "InstanceType":"t3.micro",
```

```

    "IamInstanceProfile":{
      "Name":{
        "Ref":"myInstanceProfile"
      }
    },
    "SecurityGroupIds":[
      {
        "Ref":"myNewEC2SecurityGroup"
      },
      "sg-083cd3bfb8example"
    ],
    "UserData":{
      "Fn::Base64":{
        "Fn::Join": [
          "", [
            "#!/bin/bash\n",
            "cd /tmp\n",
            "yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_amd64/amazon-ssm-agent.rpm\n",
            "systemctl enable amazon-ssm-agent\n",
            "systemctl start amazon-ssm-agent\n"
          ]
        ]
      }
    },
    "TagSpecifications":[
      {
        "ResourceType":"instance",
        "Tags":[
          {
            "Key":"environment",
            "Value":"development"
          }
        ]
      },
      {
        "ResourceType":"volume",
        "Tags":[
          {
            "Key":"environment",
            "Value":"development"
          }
        ]
      }
    ]
  }
}

```

```

    ]
  }
}
},
"myInstanceRole":{
  "Type":"AWS::IAM::Role",
  "Properties":{
    "RoleName":"InstanceRole",
    "AssumeRolePolicyDocument":{
      "Version": "2012-10-17",
      "Statement":[
        {
          "Effect":"Allow",
          "Principal":{
            "Service":[
              "ec2.amazonaws.com"
            ]
          },
          "Action":[
            "sts:AssumeRole"
          ]
        }
      ]
    }
  },
  "ManagedPolicyArns":[
    "arn:aws:iam::aws:policy/myCustomerManagedPolicy"
  ]
},
"myInstanceProfile":{
  "Type":"AWS::IAM::InstanceProfile",
  "Properties":{
    "Path":"/",
    "Roles":[
      {
        "Ref":"myInstanceRole"
      }
    ]
  }
}
}
}
}
}

```

YAML

```

---
Resources:
  myLaunchTemplate:
    Type: AWS::EC2::LaunchTemplate
    Properties:
      LaunchTemplateName: !Sub ${AWS::StackName}-launch-template
      LaunchTemplateData:
        ImageId: ami-02354e95b3example
        InstanceType: t3.micro
        IamInstanceProfile:
          Name: !Ref myInstanceProfile
        SecurityGroupIds:
          - !Ref myNewEC2SecurityGroup
          - sg-083cd3bfb8example
        UserData:
          Fn::Base64: !Sub |
            #!/bin/bash
            cd /tmp
            yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
            systemctl enable amazon-ssm-agent
            systemctl start amazon-ssm-agent
      TagSpecifications:
        - ResourceType: instance
          Tags:
            - Key: environment
              Value: development
        - ResourceType: volume
          Tags:
            - Key: environment
              Value: development
  myInstanceRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: InstanceRole
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: 'Allow'
            Principal:
              Service:
                - 'ec2.amazonaws.com'

```

```

    Action:
      - 'sts:AssumeRole'
    ManagedPolicyArns:
      - 'arn:aws:iam::aws:policy/myCustomerManagedPolicy'
  myInstanceProfile:
    Type: AWS::IAM::InstanceProfile
    Properties:
      Path: '/'
      Roles:
        - !Ref myInstanceRole

```

CloudFormation을 사용하여 보안 그룹 관리

다음 코드 조각은 CloudFormation을 사용하여 보안 그룹과 Amazon EC2 인스턴스를 관리하여 AWS 리소스에 대한 액세스를 제어하는 방법을 보여줍니다.

코드 조각 카테고리

- [보안 그룹과 Amazon EC2 인스턴스 연결](#)
- [인그레스 규칙이 있는 보안 그룹 생성](#)
- [보안 그룹 인그레스 규칙이 있는 Elastic Load Balancer 생성](#)

보안 그룹과 Amazon EC2 인스턴스 연결

다음 예제 코드 조각은 CloudFormation을 사용하여 Amazon EC2 인스턴스를 기본 Amazon VPC 보안 그룹과 연결하는 방법을 보여줍니다.

코드 조각 예제

- [기본 VPC 보안 그룹과 Amazon EC2 인스턴스 연결](#)
- [연결된 볼륨 및 보안 그룹을 포함하는 Amazon EC2 인스턴스를 생성합니다.](#)

기본 VPC 보안 그룹과 Amazon EC2 인스턴스 연결

다음 코드 조각은 Amazon VPC, VPC 내의 서브넷, Amazon EC2 인스턴스를 생성합니다. VPC는 [AWS::EC2::VPC](#) 리소스를 사용하여 생성됩니다. VPC의 IP 주소 범위는 더 큰 템플릿에서 정의되며 MyVPCCIDRRange 파라미터에 의해 참조됩니다.

서브넷은 [AWS::EC2::Subnet](#) 리소스를 사용하여 VPC 내에 생성됩니다. 서브넷은 MyVPC(으)로 참조되는 VPC와 연결됩니다.

EC2 인스턴스는 [AWS::EC2::Instance](#) 리소스를 사용하여 VPC와 서브넷 내에서 시작됩니다. 이 리소스는 인스턴스를 시작하는 데 사용할 Amazon Machine Image(AMI), 인스턴스를 실행할 서브넷, 인스턴스와 연결할 보안 그룹을 지정합니다. ImageId는 Systems Manager 파라미터를 사용하여 최신 Amazon Linux 2 AMI를 동적으로 검색합니다.

보안 그룹 ID는 MyVPC 리소스에서 기본 보안 그룹을 검색하는 Fn::GetAtt 함수를 사용하여 가져옵니다.

인스턴스는 코드 조각에 정의된 MySubnet 리소스 내에 배치됩니다.

CloudFormation을 사용하여 VPC를 생성하는 경우 AWS는 VPC 내에 기본 보안 그룹을 포함한 기본 리소스를 자동으로 생성합니다. 하지만 CloudFormation 템플릿 내에 VPC를 정의하면 템플릿을 생성할 때 이러한 기본 리소스의 ID에 액세스하지 못할 수 있습니다. 템플릿에 지정된 기본 리소스에 액세스하고 사용하려면 Fn::GetAtt와(과) 같은 내장 함수를 사용할 수 있습니다. 이 함수를 사용하면 CloudFormation에서 자동으로 생성한 기본 리소스를 사용하여 작업할 수 있습니다.

JSON

```
"MyVPC": {
  "Type": "AWS::EC2::VPC",
  "Properties": {
    "CidrBlock": {
      "Ref": "MyVPCCIDRRange"
    },
    "EnableDnsSupport": false,
    "EnableDnsHostnames": false,
    "InstanceTenancy": "default"
  }
},
"MySubnet": {
  "Type": "AWS::EC2::Subnet",
  "Properties": {
    "CidrBlock": {
      "Ref": "MyVPCCIDRRange"
    },
    "VpcId": {
      "Ref": "MyVPC"
    }
  }
},
"MyInstance": {
  "Type": "AWS::EC2::Instance",
```



```

    "Properties": {
      "ImageId": "{{resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2}}",
      "SecurityGroupIds": [
        {
          "Fn::GetAtt": [
            "MyVPC",
            "DefaultSecurityGroup"
          ]
        }
      ],
      "SubnetId": {
        "Ref": "MySubnet"
      }
    }
  }
}

```

YAML

```

MyVPC:
  Type: AWS::EC2::VPC
  Properties:
    CidrBlock:
      Ref: MyVPCCIDRRange
    EnableDnsSupport: false
    EnableDnsHostnames: false
    InstanceTenancy: default
MySubnet:
  Type: AWS::EC2::Subnet
  Properties:
    CidrBlock:
      Ref: MyVPCCIDRRange
    VpcId:
      Ref: MyVPC
MyInstance:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: '{{resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2}}'
    SecurityGroupIds:
      - Fn::GetAtt:
          - MyVPC
          - DefaultSecurityGroup

```

```
SubnetId:
  Ref: MySubnet
```

연결된 볼륨 및 보안 그룹을 포함하는 Amazon EC2 인스턴스를 생성합니다.

다음 코드 조각은 지정된 AMI에서 시작되는 [AWS::EC2::Instance](#) 리소스를 사용하여 Amazon EC2 인스턴스를 생성합니다. 인스턴스는 [AWS::EC2::SecurityGroup](#) 리소스를 사용하여 지정된 IP 주소로부터 포트 22로 들어오는 SSH 트래픽을 허용하는 보안 그룹과 연결됩니다. [AWS::EC2::Volume](#) 리소스를 사용하여 100GB Amazon EBS 볼륨을 생성합니다. 볼륨은 GetAtt 함수에 지정된 대로 인스턴스와 동일한 가용 영역에 생성되며 /dev/sdh 디바이스의 인스턴스에 마운트됩니다.

Amazon EBS 볼륨 생성에 대한 자세한 내용은 [Amazon EBS 볼륨 생성](#)을 참조하세요.

JSON

```
"Ec2Instance": {
  "Type": "AWS::EC2::Instance",
  "Properties": {
    "SecurityGroups": [
      {
        "Ref": "InstanceSecurityGroup"
      }
    ],
    "ImageId": "ami-1234567890abcdef0"
  }
},
"InstanceSecurityGroup": {
  "Type": "AWS::EC2::SecurityGroup",
  "Properties": {
    "GroupDescription": "Enable SSH access via port 22",
    "SecurityGroupIngress": [
      {
        "IpProtocol": "tcp",
        "FromPort": "22",
        "ToPort": "22",
        "CidrIp": "192.0.2.0/24"
      }
    ]
  }
},
"NewVolume": {
  "Type": "AWS::EC2::Volume",
  "Properties": {
```

```

        "Size": "100",
        "AvailabilityZone": {
            "Fn::GetAtt": [
                "Ec2Instance",
                "AvailabilityZone"
            ]
        }
    },
    "MountPoint": {
        "Type": "AWS::EC2::VolumeAttachment",
        "Properties": {
            "InstanceId": {
                "Ref": "Ec2Instance"
            },
            "VolumeId": {
                "Ref": "NewVolume"
            },
            "Device": "/dev/sdh"
        }
    }
}

```

YAML

```

Ec2Instance:
  Type: AWS::EC2::Instance
  Properties:
    SecurityGroups:
      - !Ref InstanceSecurityGroup
    ImageId: ami-1234567890abcdef0
InstanceSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Enable SSH access via port 22
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: 192.0.2.0/24
NewVolume:
  Type: AWS::EC2::Volume
  Properties:
    Size: 100

```

```

    AvailabilityZone: !GetAtt [Ec2Instance, AvailabilityZone]
MountPoint:
  Type: AWS::EC2::VolumeAttachment
  Properties:
    InstanceId: !Ref Ec2Instance
    VolumeId: !Ref NewVolume
    Device: /dev/sdh

```

인그레스 규칙이 있는 보안 그룹 생성

다음 코드 조각 예제는 CloudFormation을 사용하여 특정 인그레스 규칙이 있는 보안 그룹을 구성하는 방법을 보여줍니다.

코드 조각

- [SSH 및 HTTP 액세스를 위한 인그레스 규칙이 있는 보안 그룹 생성](#)
- [지정된 CIDR 범위에서의 HTTP 및 SSH 액세스에 대한 인그레스 규칙이 있는 보안 그룹 생성](#)
- [인그레스 규칙이 있는 교차 참조 보안 그룹 생성](#)

SSH 및 HTTP 액세스를 위한 인그레스 규칙이 있는 보안 그룹 생성

다음 코드 조각은 [AWS::EC2::SecurityGroup](#) 리소스를 사용하는 두 가지 보안 그룹 인그레스 규칙을 설명합니다. 첫 번째 인그레스 규칙은 계정 번호가 1111-2222-3333인 AWS 계정이 소유한 MyAdminSecurityGroup이라는 기존 보안 그룹에서의 SSH(포트 22) 액세스를 허용합니다. 두 번째 인그레스 규칙은 동일한 템플릿에서 생성된 MySecurityGroupCreatedInCFN이라는 다른 보안 그룹에서의 HTTP(포트 80) 액세스를 허용합니다. Ref 함수는 동일한 템플릿에 생성된 보안 그룹의 논리명을 참조하는 데 사용됩니다.

첫 번째 인그레스 규칙에서는 SourceSecurityGroupName 및 SourceSecurityGroupOwnerId 속성 모두에 값을 추가해야 합니다. 두 번째 인그레스 규칙에서 MySecurityGroupCreatedInCFNTemplate은(는) 동일한 템플릿에 생성된 다른 보안 그룹을 참조합니다. 논리명 MySecurityGroupCreatedInCFNTemplate이(가) 더 큰 템플릿에서 지정한 보안 그룹 리소스의 실제 논리명과 일치하는지 확인합니다.

보안 그룹에 대한 자세한 내용은 Amazon EC2 사용 설명서의 [Amazon EC2 인스턴스에 대한 Amazon EC2 보안 그룹](#)을 참조하세요.

JSON

```

"SecurityGroup": {

```

```

    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
      "GroupDescription": "Allow connections from specified source security group",
      "SecurityGroupIngress": [
        {
          "IpProtocol": "tcp",
          "FromPort": "22",
          "ToPort": "22",
          "SourceSecurityGroupName": "MyAdminSecurityGroup",
          "SourceSecurityGroupOwnerId": "1111-2222-3333"
        },
        {
          "IpProtocol": "tcp",
          "FromPort": "80",
          "ToPort": "80",
          "SourceSecurityGroupName": {
            "Ref": "MySecurityGroupCreatedInCFNTemplate"
          }
        }
      ]
    }
  }
}

```

YAML

```

SecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Allow connections from specified source security group
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: '22'
        ToPort: '22'
        SourceSecurityGroupName: MyAdminSecurityGroup
        SourceSecurityGroupOwnerId: '1111-2222-3333'
      - IpProtocol: tcp
        FromPort: '80'
        ToPort: '80'
        SourceSecurityGroupName:
          Ref: MySecurityGroupCreatedInCFNTemplate

```

지정된 CIDR 범위에서의 HTTP 및 SSH 액세스에 대한 인그레스 규칙이 있는 보안 그룹 생성

다음 코드 조각은 두 가지 인바운드 규칙이 있는 Amazon EC2 인스턴스용 보안 그룹을 생성합니다. 인바운드 규칙은 지정된 CIDR 범위에서 지정된 포트에 들어오는 TCP 트래픽을 허용합니다. 규칙을 지정하는 데 [AWS::EC2::SecurityGroup](#) 리소스가 사용됩니다. 각 규칙마다 프로토콜을 지정해야 합니다. TCP의 경우 포트 또는 포트 범위도 지정해야 합니다. 소스 보안 그룹 또는 CIDR 범위를 지정하지 않으면 스택이 성공적으로 시작되지만 규칙이 보안 그룹에 적용되지 않습니다.

보안 그룹에 대한 자세한 내용은 Amazon EC2 사용 설명서의 [Amazon EC2 인스턴스에 대한 Amazon EC2 보안 그룹](#)을 참조하세요.

JSON

```
"ServerSecurityGroup": {
  "Type": "AWS::EC2::SecurityGroup",
  "Properties": {
    "GroupDescription": "Allow connections from specified CIDR ranges",
    "SecurityGroupIngress": [
      {
        "IpProtocol": "tcp",
        "FromPort": "80",
        "ToPort": "80",
        "CidrIp": "192.0.2.0/24"
      },
      {
        "IpProtocol": "tcp",
        "FromPort": "22",
        "ToPort": "22",
        "CidrIp": "192.0.2.0/24"
      }
    ]
  }
}
```

YAML

```
ServerSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Allow connections from specified CIDR ranges
    SecurityGroupIngress:
      - IpProtocol: tcp
```

```

    FromPort: 80
    ToPort: 80
    CidrIp: 192.0.2.0/24
  - IpProtocol: tcp
    FromPort: 22
    ToPort: 22
    CidrIp: 192.0.2.0/24

```

인그레스 규칙이 있는 교차 참조 보안 그룹 생성

다음 코드 조각은 [AWS::EC2::SecurityGroup](#) 리소스를 사용하여 두 개의 Amazon EC2 보안 그룹 SGroup1 및 SGroup2을(를) 생성합니다. 두 보안 그룹 간의 통신을 허용하는 인그레스 규칙은 [AWS::EC2::SecurityGroupIngress](#) 리소스를 사용하여 생성됩니다. SGroup1Ingress은(는) 소스 보안 그룹 SGroup2에서 포트 80으로 들어오는 TCP 트래픽을 허용하는 SGroup1의 인그레스 규칙을 설정합니다. SGroup2Ingress은(는) 소스 보안 그룹 SGroup1에서 포트 80으로 들어오는 TCP 트래픽을 허용하는 SGroup2의 인그레스 규칙을 설정합니다.

JSON

```

"SGroup1": {
  "Type": "AWS::EC2::SecurityGroup",
  "Properties": {
    "GroupDescription": "EC2 instance access"
  }
},
"SGroup2": {
  "Type": "AWS::EC2::SecurityGroup",
  "Properties": {
    "GroupDescription": "EC2 instance access"
  }
},
"SGroup1Ingress": {
  "Type": "AWS::EC2::SecurityGroupIngress",
  "Properties": {
    "GroupName": {
      "Ref": "SGroup1"
    },
    "IpProtocol": "tcp",
    "ToPort": "80",
    "FromPort": "80",
    "SourceSecurityGroupName": {
      "Ref": "SGroup2"
    }
  }
}

```

```

    }
  },
  "SGroup2Ingress": {
    "Type": "AWS::EC2::SecurityGroupIngress",
    "Properties": {
      "GroupName": {
        "Ref": "SGroup2"
      },
      "IpProtocol": "tcp",
      "ToPort": "80",
      "FromPort": "80",
      "SourceSecurityGroupName": {
        "Ref": "SGroup1"
      }
    }
  }
}

```

YAML

```

SGroup1:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: EC2 Instance access
SGroup2:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: EC2 Instance access
SGroup1Ingress:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupName: !Ref SGroup1
    IpProtocol: tcp
    ToPort: 80
    FromPort: 80
    SourceSecurityGroupName: !Ref SGroup2
SGroup2Ingress:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupName: !Ref SGroup2
    IpProtocol: tcp
    ToPort: 80
    FromPort: 80
    SourceSecurityGroupName: !Ref SGroup1

```


보안 그룹 인그레스 규칙이 있는 Elastic Load Balancer 생성

다음 템플릿은 지정된 가용 영역에 [AWS::ElasticLoadBalancing::LoadBalancer](#) 리소스를 생성합니다. [AWS::ElasticLoadBalancing::LoadBalancer](#) 리소스는 포트 80에서 HTTP 트래픽을 수신하고 포트 80에서 인스턴스로 직접 요청을 보내도록 구성됩니다. Elastic Load Balancer는 인스턴스 간에 들어오는 HTTP 트래픽의 로드 밸런싱을 담당합니다.

또한 이 템플릿은 로드 밸런서와 연결된 [AWS::EC2::SecurityGroup](#) 리소스를 생성합니다. 이 보안 그룹은 포트 80에서 들어오는 TCP 트래픽을 허용하는 단일 인그레스 규칙(ELB ingress group(으)로 설명됨)으로 생성됩니다. 이 인그레스 규칙의 소스는 로드 밸런서 리소스에서 속성을 검색하는 Fn::GetAtt 함수를 사용하여 정의됩니다. SourceSecurityGroupOwnerId은(는) Fn::GetAtt을 사용하여 로드 밸런서 소스 보안 그룹의 OwnerAlias을(를) 가져옵니다. SourceSecurityGroupName은(는) Fn::GetAtt을(를) 사용하여 ELB 소스 보안 그룹의 GroupName을(를) 가져옵니다.

이 설정은 ELB와 인스턴스 간의 보안 통신을 보장합니다.

로드 밸런싱에 대한 자세한 내용은 [Elastic Load Balancing 사용 설명서](#)를 참조하세요.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Resources": {
    "MyELB": {
      "Type": "AWS::ElasticLoadBalancing::LoadBalancer",
      "Properties": {
        "AvailabilityZones": [
          "aa-example-1a"
        ],
        "Listeners": [
          {
            "LoadBalancerPort": "80",
            "InstancePort": "80",
            "Protocol": "HTTP"
          }
        ]
      }
    },
    "MyELBIngressGroup": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
```

```
"GroupDescription": "ELB ingress group",
"SecurityGroupIngress": [
    {
        "IpProtocol": "tcp",
        "FromPort": 80,
        "ToPort": 80,
        "SourceSecurityGroupId": {
            "Fn::GetAtt": [
                "MyELB",
                "SourceSecurityGroup.OwnerAlias"
            ]
        },
        "SourceSecurityGroupName": {
            "Fn::GetAtt": [
                "MyELB",
                "SourceSecurityGroup.GroupName"
            ]
        }
    }
]
}
```

YAML

```

AWSTemplateFormatVersion: '2010-09-09'

Resources:
  MyELB:
    Type: AWS::ElasticLoadBalancing::LoadBalancer
    Properties:
      AvailabilityZones:
        - aa-example-1a
      Listeners:
        - LoadBalancerPort: '80'
          InstancePort: '80'
          Protocol: HTTP
  MyELBIngressGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: ELB ingress group
      SecurityGroupIngress:

```

```

- IpProtocol: tcp
  FromPort: '80'
  ToPort: '80'
  SourceSecurityGroupOwnerId:
    Fn::GetAtt:
      - MyELB
      - SourceSecurityGroup.OwnerAlias
  SourceSecurityGroupName:
    Fn::GetAtt:
      - MyELB
      - SourceSecurityGroup.GroupName

```

CloudFormation을 사용하여 탄력적 IP 주소 할당 및 연결

다음 템플릿 코드 조각은 Amazon EC2의 탄력적 IP 주소(EIP)와 관련된 예입니다. 이 예에서는 인스턴스의 EIP 할당, 연결 및 관리를 다룹니다.

코드 조각 예제

- [탄력적 IP 주소 할당 및 Amazon EC2 인스턴스와 연결](#)
- [IP 주소를 지정하여 Amazon EC2 인스턴스에 탄력적 IP 주소 연결](#)
- [IP 주소의 할당 ID를 지정하여 Amazon EC2 인스턴스에 탄력적 IP 주소 연결](#)

탄력적 IP 주소 할당 및 Amazon EC2 인스턴스와 연결

다음 코드 조각은 [AWS::EC2::EIP](#) 리소스를 사용하여 Amazon EC2 탄력적 IP(EIP)를 할당하고 해당 주소를 Amazon EC2 인스턴스와 연결합니다. AWS이(가) 소유한 주소 풀 또는 [고유 IP 주소 가져오기 \(BYOIP\)](#)를 사용하여 AWS 리소스와 함께 사용하기 위해 AWS(으)로 가져온 퍼블릭 IPv4 주소 범위에서 생성한 주소 풀의 EIP 주소를 할당할 수 있습니다. 이 예에서는 AWS이(가) 소유한 주소 풀에서 EIP를 할당합니다.

탄력적 IP 주소에 대한 자세한 내용은 Amazon EC2 사용 설명서의 [탄력적 IP 주소](#)를 참조하세요.

JSON

```

"ElasticIP": {
  "Type": "AWS::EC2::EIP",
  "Properties": {
    "InstanceId": {
      "Ref": "Ec2Instance"
    }
  }
}

```

```
}
}
```

YAML

```
ElasticIP:
  Type: AWS::EC2::EIP
  Properties:
    InstanceId: !Ref EC2Instance
```

IP 주소를 지정하여 Amazon EC2 인스턴스에 탄력적 IP 주소 연결

다음 코드 조각은 [AWS::EC2::EIPAssociation](#) 리소스를 사용하여 기존 Amazon EC2 탄력적 IP 주소를 EC2 인스턴스에 연결합니다. 먼저 계정에서 사용할 탄력적 IP 주소를 할당해야 합니다. 탄력적 IP 주소는 하나의 인스턴스와 연결할 수 있습니다.

JSON

```
"IPAssoc": {
  "Type": "AWS::EC2::EIPAssociation",
  "Properties": {
    "InstanceId": {
      "Ref": "Ec2Instance"
    },
    "EIP": "192.0.2.0"
  }
}
```

YAML

```
IPAssoc:
  Type: AWS::EC2::EIPAssociation
  Properties:
    InstanceId: !Ref EC2Instance
    EIP: 192.0.2.0
```

IP 주소의 할당 ID를 지정하여 Amazon EC2 인스턴스에 탄력적 IP 주소 연결

다음 코드 조각은 [AWS::EC2::EIPAssociation](#) 리소스를 사용하여 할당 ID를 지정하여 기존 탄력적 IP 주소를 Amazon EC2 인스턴스에 연결합니다. 탄력적 IP 주소 할당 시 탄력적 IP 주소에 할당 ID가 할당됩니다.

JSON

```
"IPAssoc": {
  "Type": "AWS::EC2::EIPAssociation",
  "Properties": {
    "InstanceId": {
      "Ref": "Ec2Instance"
    },
    "AllocationId": "eipalloc-1234567890abcdef0"
  }
}
```

YAML

```
IPAssoc:
  Type: AWS::EC2::EIPAssociation
  Properties:
    InstanceId: !Ref EC2Instance
    AllocationId: eipalloc-1234567890abcdef0
```

CloudFormation을 사용하여 Amazon VPC 리소스 구성

이 섹션에서는 CloudFormation을 사용하여 Amazon VPC 리소스를 구성하는 예를 제공합니다. VPC를 사용하면 AWS 내에서 가상 네트워크를 생성할 수 있으며, 다음 코드 조각은 네트워킹 요구 사항에 맞게 VPC의 측면을 구성하는 방법을 보여 줍니다.

코드 조각 예제

- [VPC에서 IPv6 외부 전용 인터넷 액세스 활성화](#)
- [탄력적 네트워크 인터페이스\(ENI\) 템플릿 코드 조각](#)

VPC에서 IPv6 외부 전용 인터넷 액세스 활성화

외부 전용 인터넷 게이트웨이를 사용하면 VPC 내의 인스턴스가 인터넷에 액세스할 수 있으며 인터넷의 리소스가 인스턴스와 통신하는 것을 방지할 수 있습니다. 다음 코드 조각은 VPC 내에서 IPv6 외부 전용 인터넷 액세스를 활성화합니다. [AWS::EC2::VPC](#) 리소스를 사용하여 IPv4 주소 범위가 10.0.0/16인 VPC를 생성합니다. 라우팅 테이블은 [AWS::EC2::RouteTable](#) 리소스를 사용하여 이 VPC 리소스와 연결됩니다. 라우팅 테이블은 VPC 내 인스턴스의 경로를 관리합니다. [AWS::EC2::EgressOnlyInternetGateway](#)는 인바운드 트래픽을 차단하면서 VPC 내 인스턴스로부터의 아웃바운드 트래픽에 대해 IPv6 통신을 활성화하는 외부 전용 인터넷 게이트웨이를 생성하는 데 사용

됩니다. [AWS::EC2::Route](#) 리소스는 모든 아웃바운드 IPv6 트래픽(::/0)을 외부 전용 인터넷 게이트웨이로 보내는 IPv6 경로를 라우팅 테이블에 생성하도록 지정됩니다.

송신 전용 인터넷 게이트웨이에 대한 자세한 내용은 Amazon VPC 사용 설명서의 [송신 전용 인터넷 게이트웨이를 사용하여 아웃바운드 IPv6 트래픽 활성화](#)를 참조하세요.

JSON

```
"DefaultIpv6Route": {
  "Type": "AWS::EC2::Route",
  "Properties": {
    "DestinationIpv6CidrBlock": "::/0",
    "EgressOnlyInternetGatewayId": {
      "Ref": "EgressOnlyInternetGateway"
    },
    "RouteTableId": {
      "Ref": "RouteTable"
    }
  }
},
"EgressOnlyInternetGateway": {
  "Type": "AWS::EC2::EgressOnlyInternetGateway",
  "Properties": {
    "VpcId": {
      "Ref": "VPC"
    }
  }
},
"RouteTable": {
  "Type": "AWS::EC2::RouteTable",
  "Properties": {
    "VpcId": {
      "Ref": "VPC"
    }
  }
},
"VPC": {
  "Type": "AWS::EC2::VPC",
  "Properties": {
    "CidrBlock": "10.0.0.0/16"
  }
}
```

YAML

```

DefaultIpv6Route:
  Type: AWS::EC2::Route
  Properties:
    DestinationIpv6CidrBlock: ":::/0"
    EgressOnlyInternetGatewayId:
      Ref: "EgressOnlyInternetGateway"
    RouteTableId:
      Ref: "RouteTable"
EgressOnlyInternetGateway:
  Type: AWS::EC2::EgressOnlyInternetGateway
  Properties:
    VpcId:
      Ref: "VPC"
RouteTable:
  Type: AWS::EC2::RouteTable
  Properties:
    VpcId:
      Ref: "VPC"
VPC:
  Type: AWS::EC2::VPC
  Properties:
    CidrBlock: "10.0.0.0/16"

```

탄력적 네트워크 인터페이스(ENI) 템플릿 코드 조각

탄력적 네트워크 인터페이스(ENI)를 사용하여 Amazon EC2 인스턴스 생성

다음 예제 스니펫은 지정된 Amazon VPC 및 서브넷에서 [AWS::EC2::Instance](#) 리소스를 사용하여 Amazon EC2 인스턴스를 생성합니다. 인스턴스에 두 개의 네트워크 인터페이스(ENI)를 연결하고, 연결된 ENI를 통해 탄력적 IP 주소를 인스턴스에 연결하고, SSH 및 HTTP 액세스를 위한 보안 그룹을 구성합니다. 사용자 데이터는 인스턴스 생성 시 시작 구성의 일부로 인스턴스에 제공됩니다. 사용자 데이터에는 인스턴스에 전달되도록 base64 형식으로 인코딩된 스크립트가 포함되어 있습니다. 인스턴스가 시작되면 스크립트는 부트스트랩 프로세스의 일부로 자동으로 실행됩니다. ec2-net-utils을(를) 설치하고 네트워크 인터페이스를 구성한 다음 HTTP 서비스를 시작합니다.

선택한 리전을 기반으로 적절한 Amazon Machine Image(AMI)를 결정하기 위해 코드 조각은 RegionMap 매핑에서 값을 조회하는 Fn::FindInMap 함수를 사용합니다. 이 매핑은 더 큰 템플릿에서 정의해야 합니다. 두 네트워크 인터페이스는 [AWS::EC2::NetworkInterface](#) 리소스를 사용하여 생성됩니다. 탄력적 IP 주소는 vpc 도메인에 할당된 [AWS::EC2::EIP](#) 리소스를 사용하여 지정됩니다. 해당 탄력적 IP 주소는 [AWS::EC2::EIPAssociation](#) 리소스를 사용하는 네트워크 인터페이스와 연결됩니다.

Outputs 섹션에서는 스택이 생성된 후 액세스하려는 값이나 리소스를 정의합니다. 이 코드 조각에서 정의된 출력은 InstancePublicIp이며 스택에서 생성한 EC2 인스턴스의 퍼블릭 IP 주소를 나타냅니다. CloudFormation 콘솔의 출력 탭이나 [describe-stacks](#) 명령을 사용하여 이 출력을 검색할 수 있습니다.

탄력적 네트워크 인터페이스에 대한 자세한 내용은 [탄력적 네트워크 인터페이스](#)를 참조하세요.

JSON

```
"Resources": {
  "ControlPortAddress": {
    "Type": "AWS::EC2::EIP",
    "Properties": {
      "Domain": "vpc"
    }
  },
  "AssociateControlPort": {
    "Type": "AWS::EC2::EIPAssociation",
    "Properties": {
      "AllocationId": {
        "Fn::GetAtt": [
          "ControlPortAddress",
          "AllocationId"
        ]
      },
      "NetworkInterfaceId": {
        "Ref": "controlXface"
      }
    }
  },
  "WebPortAddress": {
    "Type": "AWS::EC2::EIP",
    "Properties": {
      "Domain": "vpc"
    }
  },
  "AssociateWebPort": {
    "Type": "AWS::EC2::EIPAssociation",
    "Properties": {
      "AllocationId": {
        "Fn::GetAtt": [
          "WebPortAddress",
          "AllocationId"
        ]
      }
    }
  }
}
```



```

        ]
      },
      "NetworkInterfaceId": {
        "Ref": "webXface"
      }
    }
  },
  "SSHSecurityGroup": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
      "VpcId": {
        "Ref": "VpcId"
      },
      "GroupDescription": "Enable SSH access via port 22",
      "SecurityGroupIngress": [
        {
          "CidrIp": "0.0.0.0/0",
          "FromPort": 22,
          "IpProtocol": "tcp",
          "ToPort": 22
        }
      ]
    }
  },
  "WebSecurityGroup": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
      "VpcId": {
        "Ref": "VpcId"
      },
      "GroupDescription": "Enable HTTP access via user-defined port",
      "SecurityGroupIngress": [
        {
          "CidrIp": "0.0.0.0/0",
          "FromPort": 80,
          "IpProtocol": "tcp",
          "ToPort": 80
        }
      ]
    }
  },
  "controlXface": {
    "Type": "AWS::EC2::NetworkInterface",
    "Properties": {

```

```
        "SubnetId": {
            "Ref": "SubnetId"
        },
        "Description": "Interface for controlling traffic such as SSH",
        "GroupSet": [
            {
                "Fn::GetAtt": [
                    "SSHSecurityGroup",
                    "GroupId"
                ]
            }
        ],
        "SourceDestCheck": true,
        "Tags": [
            {
                "Key": "Network",
                "Value": "Control"
            }
        ]
    },
    "webXface": {
        "Type": "AWS::EC2::NetworkInterface",
        "Properties": {
            "SubnetId": {
                "Ref": "SubnetId"
            },
            "Description": "Interface for web traffic",
            "GroupSet": [
                {
                    "Fn::GetAtt": [
                        "WebSecurityGroup",
                        "GroupId"
                    ]
                }
            ],
            "SourceDestCheck": true,
            "Tags": [
                {
                    "Key": "Network",
                    "Value": "Web"
                }
            ]
        }
    }
}
```

```

    },
    "Ec2Instance": {
      "Type": "AWS::EC2::Instance",
      "Properties": {
        "ImageId": {
          "Fn::FindInMap": [
            "RegionMap",
            {
              "Ref": "AWS::Region"
            }
          ],
          "AMI"
        },
        "KeyName": {
          "Ref": "KeyName"
        },
        "NetworkInterfaces": [
          {
            "NetworkInterfaceId": {
              "Ref": "controlXface"
            },
            "DeviceIndex": "0"
          },
          {
            "NetworkInterfaceId": {
              "Ref": "webXface"
            },
            "DeviceIndex": "1"
          }
        ],
        "Tags": [
          {
            "Key": "Role",
            "Value": "Test Instance"
          }
        ],
        "UserData": {
          "Fn::Base64": {
            "Fn::Sub": "#!/bin/bash -xe\nyum install ec2-net-utils -y\nec2ifup\neth1\nservice httpd start\n"
          }
        }
      }
    }
  }
}

```

```

},
"Outputs": {
  "InstancePublicIp": {
    "Description": "Public IP Address of the EC2 Instance",
    "Value": {
      "Fn::GetAtt": [
        "Ec2Instance",
        "PublicIp"
      ]
    }
  }
}
}

```

YAML

```

Resources:
  ControlPortAddress:
    Type: AWS::EC2::EIP
    Properties:
      Domain: vpc
  AssociateControlPort:
    Type: AWS::EC2::EIPAssociation
    Properties:
      AllocationId:
        Fn::GetAtt:
          - ControlPortAddress
          - AllocationId
      NetworkInterfaceId:
        Ref: controlXface
  WebPortAddress:
    Type: AWS::EC2::EIP
    Properties:
      Domain: vpc
  AssociateWebPort:
    Type: AWS::EC2::EIPAssociation
    Properties:
      AllocationId:
        Fn::GetAtt:
          - WebPortAddress
          - AllocationId
      NetworkInterfaceId:
        Ref: webXface
  SSHSecurityGroup:

```

```
Type: AWS::EC2::SecurityGroup
Properties:
  VpcId:
    Ref: VpcId
  GroupDescription: Enable SSH access via port 22
  SecurityGroupIngress:
    - CidrIp: 0.0.0.0/0
      FromPort: 22
      IpProtocol: tcp
      ToPort: 22
WebSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    VpcId:
      Ref: VpcId
    GroupDescription: Enable HTTP access via user-defined port
    SecurityGroupIngress:
      - CidrIp: 0.0.0.0/0
        FromPort: 80
        IpProtocol: tcp
        ToPort: 80
controlXface:
  Type: AWS::EC2::NetworkInterface
  Properties:
    SubnetId:
      Ref: SubnetId
    Description: Interface for controlling traffic such as SSH
    GroupSet:
      - Fn::GetAtt:
          - SSHSecurityGroup
          - GroupId
    SourceDestCheck: true
    Tags:
      - Key: Network
        Value: Control
webXface:
  Type: AWS::EC2::NetworkInterface
  Properties:
    SubnetId:
      Ref: SubnetId
    Description: Interface for web traffic
    GroupSet:
      - Fn::GetAtt:
          - WebSecurityGroup
```

```
    - GroupId
SourceDestCheck: true
Tags:
  - Key: Network
    Value: Web
Ec2Instance:
  Type: AWS::EC2::Instance
  Properties:
    ImageId:
      Fn::FindInMap:
        - RegionMap
        - Ref: AWS::Region
        - AMI
    KeyName:
      Ref: KeyName
    NetworkInterfaces:
      - NetworkInterfaceId:
          Ref: controlXface
          DeviceIndex: "0"
      - NetworkInterfaceId:
          Ref: webXface
          DeviceIndex: "1"
    Tags:
      - Key: Role
        Value: Test Instance
    UserData:
      Fn::Base64: !Sub |
        #!/bin/bash -xe
        yum install ec2-net-utils -y
        ec2ifup eth1
        service httpd start
Outputs:
  InstancePublicIp:
    Description: Public IP Address of the EC2 Instance
    Value:
      Fn::GetAtt:
        - Ec2Instance
        - PublicIp
```

Amazon Elastic Container Service 샘플 템플릿

Amazon Elastic Container Service(Amazon ECS)는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 클러스터에서 도커 컨테이너를 손쉽게 실행, 중지 및 관리할 수 있게 해 주는 컨테이너 관리 서비스입니다.

AL2023 Amazon ECS-Optimized-AMI를 사용하여 클러스터 생성

Amazon EC2에서 AL2023 인스턴스를 실행하는 용량 공급자를 사용하는 클러스터를 정의합니다.

Important

최신 AMI ID는 Amazon Elastic Container Service 개발자 안내서의 [Amazon ECS 최적화 AMI](#)를 참조하세요.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "EC2 ECS cluster that starts out empty, with no EC2 instances yet. An ECS capacity provider automatically launches more EC2 instances as required on the fly when you request ECS to launch services or standalone tasks.",
  "Parameters": {
    "InstanceType": {
      "Type": "String",
      "Description": "EC2 instance type",
      "Default": "t2.medium",
      "AllowedValues": [
        "t1.micro",
        "t2.2xlarge",
        "t2.large",
        "t2.medium",
        "t2.micro",
        "t2.nano",
        "t2.small",
        "t2.xlarge",
        "t3.2xlarge",
        "t3.large",
        "t3.medium",
        "t3.micro",
        "t3.nano",
```

```

        "t3.small",
        "t3.xlarge"
    ]
},
"DesiredCapacity": {
    "Type": "Number",
    "Default": "0",
    "Description": "Number of EC2 instances to launch in your ECS cluster."
},
"MaxSize": {
    "Type": "Number",
    "Default": "100",
    "Description": "Maximum number of EC2 instances that can be launched in your
ECS cluster."
},
"ECSAMI": {
    "Description": "The Amazon Machine Image ID used for the cluster",
    "Type": "AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>",
    "Default": "/aws/service/ecs/optimized-ami/amazon-linux-2023/recommended/
image_id"
},
"VpcId": {
    "Type": "AWS::EC2::VPC::Id",
    "Description": "VPC ID where the ECS cluster is launched",
    "Default": "vpc-1234567890abcdef0"
},
"SubnetIds": {
    "Type": "List<AWS::EC2::Subnet::Id>",
    "Description": "List of subnet IDs where the EC2 instances will be launched",
    "Default": "subnet-021345abcdef67890"
}
},
"Resources": {
    "ECSCluster": {
        "Type": "AWS::ECS::Cluster",
        "Properties": {
            "ClusterSettings": [
                {
                    "Name": "containerInsights",
                    "Value": "enabled"
                }
            ]
        }
    }
}
},

```



```
"ECSAutoScalingGroup": {
  "Type": "AWS::AutoScaling::AutoScalingGroup",
  "DependsOn": [
    "ECSCluster",
    "EC2Role"
  ],
  "Properties": {
    "VPCZoneIdentifier": {
      "Ref": "SubnetIds"
    },
    "LaunchTemplate": {
      "LaunchTemplateId": {
        "Ref": "ContainerInstances"
      },
      "Version": {
        "Fn::GetAtt": [
          "ContainerInstances",
          "LatestVersionNumber"
        ]
      }
    },
    "MinSize": 0,
    "MaxSize": {
      "Ref": "MaxSize"
    },
    "DesiredCapacity": {
      "Ref": "DesiredCapacity"
    },
    "NewInstancesProtectedFromScaleIn": true
  },
  "UpdatePolicy": {
    "AutoScalingReplacingUpdate": {
      "WillReplace": "true"
    }
  }
},
"ContainerInstances": {
  "Type": "AWS::EC2::LaunchTemplate",
  "Properties": {
    "LaunchTemplateName": "asg-launch-template",
    "LaunchTemplateData": {
      "ImageId": {
        "Ref": "ECSAMI"
      }
    }
  },
```

```

        "InstanceType": {
            "Ref": "InstanceType"
        },
        "IamInstanceProfile": {
            "Name": {
                "Ref": "EC2InstanceProfile"
            }
        },
        "SecurityGroupIds": [
            {
                "Ref": "ContainerHostSecurityGroup"
            }
        ],
        "UserData": {
            "Fn::Base64": {
                "Fn::Sub": "#!/bin/bash -xe\n echo ECS_CLUSTER=${ECSCluster}
>> /etc/ecs/ecs.config\n yum install -y aws-cfn-bootstrap\n /opt/aws/bin/cfn-init -
v --stack ${AWS::StackId} --resource ContainerInstances --configsets full_install --
region ${AWS::Region} &\n"
            }
        },
        "MetadataOptions": {
            "HttpEndpoint": "enabled",
            "HttpTokens": "required"
        }
    }
},
    "EC2InstanceProfile": {
        "Type": "AWS::IAM::InstanceProfile",
        "Properties": {
            "Path": "/",
            "Roles": [
                {
                    "Ref": "EC2Role"
                }
            ]
        }
    },
    "CapacityProvider": {
        "Type": "AWS::ECS::CapacityProvider",
        "Properties": {
            "AutoScalingGroupProvider": {
                "AutoScalingGroupArn": {

```

```

        "Ref": "ECSAutoScalingGroup"
    },
    "ManagedScaling": {
        "InstanceWarmupPeriod": 60,
        "MinimumScalingStepSize": 1,
        "MaximumScalingStepSize": 100,
        "Status": "ENABLED",
        "TargetCapacity": 100
    },
    "ManagedTerminationProtection": "ENABLED"
}
},
"CapacityProviderAssociation": {
    "Type": "AWS::ECS::ClusterCapacityProviderAssociations",
    "Properties": {
        "CapacityProviders": [
            {
                "Ref": "CapacityProvider"
            }
        ],
        "Cluster": {
            "Ref": "ECSCluster"
        },
        "DefaultCapacityProviderStrategy": [
            {
                "Base": 0,
                "CapacityProvider": {
                    "Ref": "CapacityProvider"
                },
                "Weight": 1
            }
        ]
    }
},
"ContainerHostSecurityGroup": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
        "GroupDescription": "Access to the EC2 hosts that run containers",
        "VpcId": {
            "Ref": "VpcId"
        }
    }
},
},

```

```

    "EC2Role": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": [
                  "ec2.amazonaws.com"
                ]
              },
              "Action": [
                "sts:AssumeRole"
              ]
            }
          ]
        },
        "Path": "/",
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AmazonEC2ContainerServiceforEC2Role",
          "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore"
        ]
      }
    },
    "ECSTaskExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": [
                  "ecs-tasks.amazonaws.com"
                ]
              },
              "Action": [
                "sts:AssumeRole"
              ],
              "Condition": {
                "ArnLike": {
                  "aws:SourceArn": {

```

```

        "Fn::Sub": "arn:${AWS::Partition}:ecs:
${AWS::Region}:${AWS::AccountId}:*"
    },
    "StringEquals": {
        "aws:SourceAccount": {
            "Fn::Sub": "${AWS::AccountId}"
        }
    }
},
"Path": "/",
"ManagedPolicyArns": [
    "arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy"
]
}
},
"Outputs": {
    "ClusterName": {
        "Description": "The ECS cluster into which to launch resources",
        "Value": "ECSCluster"
    },
    "ECSTaskExecutionRole": {
        "Description": "The role used to start up a task",
        "Value": "ECSTaskExecutionRole"
    },
    "CapacityProvider": {
        "Description": "The cluster capacity provider that the service should use to
request capacity when it wants to start up a task",
        "Value": "CapacityProvider"
    }
}
}

```

YAML

```

AWSTemplateFormatVersion: 2010-09-09
Description: EC2 ECS cluster that starts out empty, with no EC2 instances yet.

```

An ECS capacity provider automatically launches more EC2 instances as required on the fly when you request ECS to launch services or standalone tasks.

Parameters:

InstanceType:

Type: String

Description: EC2 instance type

Default: "t2.medium"

AllowedValues:

- t1.micro
- t2.2xlarge
- t2.large
- t2.medium
- t2.micro
- t2.nano
- t2.small
- t2.xlarge
- t3.2xlarge
- t3.large
- t3.medium
- t3.micro
- t3.nano
- t3.small
- t3.xlarge

DesiredCapacity:

Type: Number

Default: "0"

Description: Number of EC2 instances to launch in your ECS cluster.

MaxSize:

Type: Number

Default: "100"

Description: Maximum number of EC2 instances that can be launched in your ECS cluster.

ECSAMI:

Description: The Amazon Machine Image ID used for the cluster

Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>

Default: /aws/service/ecs/optimized-ami/amazon-linux-2023/recommended/image_id

VpcId:

Type: AWS::EC2::VPC::Id

Description: VPC ID where the ECS cluster is launched

Default: vpc-1234567890abcdef0

SubnetIds:

Type: List<AWS::EC2::Subnet::Id>

Description: List of subnet IDs where the EC2 instances will be launched

Default: "subnet-021345abcdef67890"

```
Resources:
# This is authorizes ECS to manage resources on your
# account on your behalf. This role is likely already created on your account
# ECSRole:
#   Type: AWS::IAM::ServiceLinkedRole
#   Properties:
#     AWSServiceName: 'ecs.amazonaws.com'

# ECS Resources
ECSCluster:
  Type: AWS::ECS::Cluster
  Properties:
    ClusterSettings:
      - Name: containerInsights
        Value: enabled

# Autoscaling group. This launches the actual EC2 instances that will register
# themselves as members of the cluster, and run the docker containers.
ECSAutoScalingGroup:
  Type: AWS::AutoScaling::AutoScalingGroup
  DependsOn:
    # This is to ensure that the ASG gets deleted first before these
    # resources, when it comes to stack teardown.
    - ECSCluster
    - EC2Role
  Properties:
    VPCZoneIdentifier:
      Ref: SubnetIds
    LaunchTemplate:
      LaunchTemplateId: !Ref ContainerInstances
      Version: !GetAtt ContainerInstances.LatestVersionNumber
    MinSize: 0
    MaxSize:
      Ref: MaxSize
    DesiredCapacity:
      Ref: DesiredCapacity
    NewInstancesProtectedFromScaleIn: true
    UpdatePolicy:
      AutoScalingReplacingUpdate:
        WillReplace: "true"
# The config for each instance that is added to the cluster
ContainerInstances:
  Type: AWS::EC2::LaunchTemplate
  Properties:
```

```

LaunchTemplateName: "asg-launch-template"
LaunchTemplateData:
  ImageId:
    Ref: ECSAMI
  InstanceType:
    Ref: InstanceType
  IamInstanceProfile:
    Name: !Ref EC2InstanceProfile
  SecurityGroupIds:
    - !Ref ContainerHostSecurityGroup
  # This injected configuration file is how the EC2 instance
  # knows which ECS cluster on your AWS account it should be joining
  UserData:
    Fn::Base64: !Sub |
      #!/bin/bash -xe
      echo ECS_CLUSTER=${ECSCluster} >> /etc/ecs/ecs.config
      yum install -y aws-cfn-bootstrap
      /opt/aws/bin/cfn-init -v --stack ${AWS::StackId} --resource
ContainerInstances --configsets full_install --region ${AWS::Region} &
      # Disable IMDSv1, and require IMDSv2
  MetadataOptions:
    HttpEndpoint: enabled
    HttpTokens: required
EC2InstanceProfile:
  Type: AWS::IAM::InstanceProfile
  Properties:
    Path: /
    Roles:
      - !Ref EC2Role
  # Create an ECS capacity provider to attach the ASG to the ECS cluster
  # so that it autoscales as we launch more containers
CapacityProvider:
  Type: AWS::ECS::CapacityProvider
  Properties:
    AutoScalingGroupProvider:
      AutoScalingGroupArn: !Ref ECSAutoScalingGroup
    ManagedScaling:
      InstanceWarmupPeriod: 60
      MinimumScalingStepSize: 1
      MaximumScalingStepSize: 100
      Status: ENABLED
      # Percentage of cluster reservation to try to maintain
      TargetCapacity: 100
    ManagedTerminationProtection: ENABLED

```



```

# Create a cluster capacity provider association so that the cluster
# will use the capacity provider
CapacityProviderAssociation:
  Type: AWS::ECS::ClusterCapacityProviderAssociations
  Properties:
    CapacityProviders:
      - !Ref CapacityProvider
    Cluster: !Ref ECSCluster
    DefaultCapacityProviderStrategy:
      - Base: 0
        CapacityProvider: !Ref CapacityProvider
        Weight: 1
# A security group for the EC2 hosts that will run the containers.
# This can be used to limit incoming traffic to or outgoing traffic
# from the container's host EC2 instance.
ContainerHostSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Access to the EC2 hosts that run containers
    VpcId:
      Ref: VpcId
# Role for the EC2 hosts. This allows the ECS agent on the EC2 hosts
# to communicate with the ECS control plane, as well as download the docker
# images from ECR to run on your host.
EC2Role:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Statement:
        - Effect: Allow
          Principal:
            Service:
              - ec2.amazonaws.com
          Action:
            - sts:AssumeRole
    Path: /
    ManagedPolicyArns:
      # See reference: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/
security-iam-awsmanpol.html#security-iam-awsmanpol-AmazonEC2ContainerServiceforEC2Role
      - arn:aws:iam::aws:policy/service-role/AmazonEC2ContainerServiceforEC2Role
      # This managed policy allows us to connect to the instance using SSM
      - arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
# This is a role which is used within Fargate to allow the Fargate agent
# to download images, and upload logs.

```

```

ECSTaskExecutionRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Statement:
        - Effect: Allow
          Principal:
            Service:
              - ecs-tasks.amazonaws.com
          Action:
            - sts:AssumeRole
          Condition:
            ArnLike:
              aws:SourceArn: !Sub arn:${AWS::Partition}:ecs:${AWS::Region}:
${AWS::AccountId}:*
            StringEquals:
              aws:SourceAccount: !Sub ${AWS::AccountId}
      Path: /
      # This role enables all features of ECS. See reference:
      # https://docs.aws.amazon.com/AmazonECS/latest/developerguide/security-iam-awsmanpol.html#security-iam-awsmanpol-AmazonECSTaskExecutionRolePolicy
    ManagedPolicyArns:
      - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
Outputs:
  ClusterName:
    Description: The ECS cluster into which to launch resources
    Value: ECSCluster
  ECSTaskExecutionRole:
    Description: The role used to start up a task
    Value: ECSTaskExecutionRole
  CapacityProvider:
    Description: The cluster capacity provider that the service should use to
      request capacity when it wants to start up a task
    Value: CapacityProvider

```

서비스를 배포합니다.

다음 템플릿은 용량 공급자를 통해 실행할 AL2023 용량을 요청하는 서비스를 정의합니다. 컨테이너가 온라인 상태가 되면 AL2023 인스턴스에서 실행됩니다.

JSON

```
{
```

```
"AWSTemplateFormatVersion": "2010-09-09",
"Description": "An example service that deploys in AWS VPC networking mode on EC2
capacity. Service uses a capacity provider to request EC2 instances to run on. Service
runs with networking in private subnets, but still accessible to the internet via a
load balancer hosted in public subnets.",
"Parameters": {
  "VpcId": {
    "Type": "String",
    "Description": "The VPC that the service is running inside of"
  },
  "PublicSubnetIds": {
    "Type": "List<AWS::EC2::Subnet::Id>",
    "Description": "List of public subnet ID's to put the load balancer in"
  },
  "PrivateSubnetIds": {
    "Type": "List<AWS::EC2::Subnet::Id>",
    "Description": "List of private subnet ID's that the AWS VPC tasks are in"
  },
  "ClusterName": {
    "Type": "String",
    "Description": "The name of the ECS cluster into which to launch capacity."
  },
  "ECSTaskExecutionRole": {
    "Type": "String",
    "Description": "The role used to start up an ECS task"
  },
  "CapacityProvider": {
    "Type": "String",
    "Description": "The cluster capacity provider that the service should use to
request capacity when it wants to start up a task"
  },
  "ServiceName": {
    "Type": "String",
    "Default": "web",
    "Description": "A name for the service"
  },
  "ImageUrl": {
    "Type": "String",
    "Default": "public.ecr.aws/docker/library/nginx:latest",
    "Description": "The url of a docker image that contains the application
process that will handle the traffic for this service"
  },
  "ContainerCpu": {
    "Type": "Number",
```

```

        "Default": 256,
        "Description": "How much CPU to give the container. 1024 is 1 CPU"
    },
    "ContainerMemory": {
        "Type": "Number",
        "Default": 512,
        "Description": "How much memory in megabytes to give the container"
    },
    "ContainerPort": {
        "Type": "Number",
        "Default": 80,
        "Description": "What port that the application expects traffic on"
    },
    "DesiredCount": {
        "Type": "Number",
        "Default": 2,
        "Description": "How many copies of the service task to run"
    }
},
"Resources": {
    "TaskDefinition": {
        "Type": "AWS::ECS::TaskDefinition",
        "Properties": {
            "Family": {
                "Ref": "ServiceName"
            },
            "Cpu": {
                "Ref": "ContainerCpu"
            },
            "Memory": {
                "Ref": "ContainerMemory"
            },
            "NetworkMode": "awsvpc",
            "RequiresCompatibilities": [
                "EC2"
            ],
            "ExecutionRoleArn": {
                "Ref": "ECSTaskExecutionRole"
            },
            "ContainerDefinitions": [
                {
                    "Name": {
                        "Ref": "ServiceName"
                    },

```

```

        "Cpu": {
            "Ref": "ContainerCpu"
        },
        "Memory": {
            "Ref": "ContainerMemory"
        },
        "Image": {
            "Ref": "ImageUrl"
        },
        "PortMappings": [
            {
                "ContainerPort": {
                    "Ref": "ContainerPort"
                },
                "HostPort": {
                    "Ref": "ContainerPort"
                }
            }
        ],
        "LogConfiguration": {
            "LogDriver": "awslogs",
            "Options": {
                "mode": "non-blocking",
                "max-buffer-size": "25m",
                "awslogs-group": {
                    "Ref": "LogGroup"
                },
                "awslogs-region": {
                    "Ref": "AWS::Region"
                },
                "awslogs-stream-prefix": {
                    "Ref": "ServiceName"
                }
            }
        }
    }
}

},
"Service": {
    "Type": "AWS::ECS::Service",
    "DependsOn": "PublicLoadBalancerListener",
    "Properties": {
        "ServiceName": {

```

```
        "Ref": "ServiceName"
    },
    "Cluster": {
        "Ref": "ClusterName"
    },
    "PlacementStrategies": [
        {
            "Field": "attribute:ecs.availability-zone",
            "Type": "spread"
        },
        {
            "Field": "cpu",
            "Type": "binpack"
        }
    ],
    "CapacityProviderStrategy": [
        {
            "Base": 0,
            "CapacityProvider": {
                "Ref": "CapacityProvider"
            },
            "Weight": 1
        }
    ],
    "NetworkConfiguration": {
        "AwsvpcConfiguration": {
            "SecurityGroups": [
                {
                    "Ref": "ServiceSecurityGroup"
                }
            ],
            "Subnets": {
                "Ref": "PrivateSubnetIds"
            }
        }
    },
    "DeploymentConfiguration": {
        "MaximumPercent": 200,
        "MinimumHealthyPercent": 75
    },
    "DesiredCount": {
        "Ref": "DesiredCount"
    },
    "TaskDefinition": {
```

```

        "Ref": "TaskDefinition"
    },
    "LoadBalancers": [
        {
            "ContainerName": {
                "Ref": "ServiceName"
            },
            "ContainerPort": {
                "Ref": "ContainerPort"
            },
            "TargetGroupArn": {
                "Ref": "ServiceTargetGroup"
            }
        }
    ]
},
"ServiceSecurityGroup": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
        "GroupDescription": "Security group for service",
        "VpcId": {
            "Ref": "VpcId"
        }
    }
},
"ServiceTargetGroup": {
    "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
    "Properties": {
        "HealthCheckIntervalSeconds": 6,
        "HealthCheckPath": "/",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 5,
        "HealthyThresholdCount": 2,
        "TargetType": "ip",
        "Port": {
            "Ref": "ContainerPort"
        },
        "Protocol": "HTTP",
        "UnhealthyThresholdCount": 10,
        "VpcId": {
            "Ref": "VpcId"
        }
    },
    "TargetGroupAttributes": [

```

```

        {
            "Key": "deregistration_delay.timeout_seconds",
            "Value": 0
        }
    ]
}
},
"PublicLoadBalancerSG": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
        "GroupDescription": "Access to the public facing load balancer",
        "VpcId": {
            "Ref": "VpcId"
        },
        "SecurityGroupIngress": [
            {
                "CidrIp": "0.0.0.0/0",
                "IpProtocol": -1
            }
        ]
    }
},
"PublicLoadBalancer": {
    "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
    "Properties": {
        "Scheme": "internet-facing",
        "LoadBalancerAttributes": [
            {
                "Key": "idle_timeout.timeout_seconds",
                "Value": "30"
            }
        ],
        "Subnets": {
            "Ref": "PublicSubnetIds"
        },
        "SecurityGroups": [
            {
                "Ref": "PublicLoadBalancerSG"
            }
        ]
    }
},
"PublicLoadBalancerListener": {
    "Type": "AWS::ElasticLoadBalancingV2::Listener",

```



```

    "Properties": {
      "DefaultActions": [
        {
          "Type": "forward",
          "ForwardConfig": {
            "TargetGroups": [
              {
                "TargetGroupArn": {
                  "Ref": "ServiceTargetGroup"
                },
                "Weight": 100
              }
            ]
          }
        }
      ],
      "LoadBalancerArn": {
        "Ref": "PublicLoadBalancer"
      },
      "Port": 80,
      "Protocol": "HTTP"
    }
  },
  "ServiceIngressfromLoadBalancer": {
    "Type": "AWS::EC2::SecurityGroupIngress",
    "Properties": {
      "Description": "Ingress from the public ALB",
      "GroupId": {
        "Ref": "ServiceSecurityGroup"
      },
      "IpProtocol": -1,
      "SourceSecurityGroupId": {
        "Ref": "PublicLoadBalancerSG"
      }
    }
  },
  "LogGroup": {
    "Type": "AWS::Logs::LogGroup"
  }
}

```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: >-
  An example service that deploys in AWS VPC networking mode on EC2 capacity.
  Service uses a capacity provider to request EC2 instances to run on. Service
  runs with networking in private subnets, but still accessible to the internet
  via a load balancer hosted in public subnets.
Parameters:
  VpcId:
    Type: String
    Description: The VPC that the service is running inside of
  PublicSubnetIds:
    Type: 'List<AWS::EC2::Subnet::Id>'
    Description: List of public subnet ID's to put the load balancer in
  PrivateSubnetIds:
    Type: 'List<AWS::EC2::Subnet::Id>'
    Description: List of private subnet ID's that the AWS VPC tasks are in
  ClusterName:
    Type: String
    Description: The name of the ECS cluster into which to launch capacity.
  ECSTaskExecutionRole:
    Type: String
    Description: The role used to start up an ECS task
  CapacityProvider:
    Type: String
    Description: >-
      The cluster capacity provider that the service should use to request
      capacity when it wants to start up a task
  ServiceName:
    Type: String
    Default: web
    Description: A name for the service
  ImageUrl:
    Type: String
    Default: 'public.ecr.aws/docker/library/nginx:latest'
    Description: >-
      The url of a docker image that contains the application process that will
      handle the traffic for this service
  ContainerCpu:
    Type: Number
    Default: 256
    Description: How much CPU to give the container. 1024 is 1 CPU
  ContainerMemory:
```

```
Type: Number
Default: 512
Description: How much memory in megabytes to give the container
ContainerPort:
  Type: Number
  Default: 80
  Description: What port that the application expects traffic on
DesiredCount:
  Type: Number
  Default: 2
  Description: How many copies of the service task to run
Resources:
  TaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      Family: !Ref ServiceName
      Cpu: !Ref ContainerCpu
      Memory: !Ref ContainerMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - EC2
      ExecutionRoleArn: !Ref ECSTaskExecutionRole
      ContainerDefinitions:
        - Name: !Ref ServiceName
          Cpu: !Ref ContainerCpu
          Memory: !Ref ContainerMemory
          Image: !Ref ImageUrl
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              HostPort: !Ref ContainerPort
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-group: !Ref LogGroup
              awslogs-region: !Ref AWS::Region
              awslogs-stream-prefix: !Ref ServiceName
  Service:
    Type: AWS::ECS::Service
    DependsOn: PublicLoadBalancerListener
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ClusterName
```

```
PlacementStrategies:
  - Field: 'attribute:ecs.availability-zone'
    Type: spread
  - Field: cpu
    Type: binpack
CapacityProviderStrategy:
  - Base: 0
    CapacityProvider: !Ref CapacityProvider
    Weight: 1
NetworkConfiguration:
  AwsVpcConfiguration:
    SecurityGroups:
      - !Ref ServiceSecurityGroup
    Subnets: !Ref PrivateSubnetIds
DeploymentConfiguration:
  MaximumPercent: 200
  MinimumHealthyPercent: 75
DesiredCount: !Ref DesiredCount
TaskDefinition: !Ref TaskDefinition
LoadBalancers:
  - ContainerName: !Ref ServiceName
    ContainerPort: !Ref ContainerPort
    TargetGroupArn: !Ref ServiceTargetGroup
ServiceSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Security group for service
    VpcId: !Ref VpcId
ServiceTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    HealthCheckIntervalSeconds: 6
    HealthCheckPath: /
    HealthCheckProtocol: HTTP
    HealthCheckTimeoutSeconds: 5
    HealthyThresholdCount: 2
    TargetType: ip
    Port: !Ref ContainerPort
    Protocol: HTTP
    UnhealthyThresholdCount: 10
    VpcId: !Ref VpcId
    TargetGroupAttributes:
      - Key: deregistration_delay.timeout_seconds
        Value: 0
```

```
PublicLoadBalancerSG:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Access to the public facing load balancer
    VpcId: !Ref VpcId
    SecurityGroupIngress:
      - CidrIp: 0.0.0.0/0
        IpProtocol: -1
PublicLoadBalancer:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
    Scheme: internet-facing
    LoadBalancerAttributes:
      - Key: idle_timeout.timeout_seconds
        Value: '30'
    Subnets: !Ref PublicSubnetIds
    SecurityGroups:
      - !Ref PublicLoadBalancerSG
PublicLoadBalancerListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
      - Type: forward
        ForwardConfig:
          TargetGroups:
            - TargetGroupArn: !Ref ServiceTargetGroup
              Weight: 100
    LoadBalancerArn: !Ref PublicLoadBalancer
    Port: 80
    Protocol: HTTP
ServiceIngressfromLoadBalancer:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    Description: Ingress from the public ALB
    GroupId: !Ref ServiceSecurityGroup
    IpProtocol: -1
    SourceSecurityGroupId: !Ref PublicLoadBalancerSG
LogGroup:
  Type: AWS::Logs::LogGroup
```

Amazon Elastic File System 샘플 템플릿

Amazon Elastic File System(Amazon EFS)는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 대한 파일 스토리지 서비스입니다. Amazon EFS를 사용하면 파일을 추가하고 제거할 때마다 스토리지 용량이 자동으로 확장 및 축소되므로 애플리케이션에서 스토리지가 필요할 때 확보할 수 있습니다.

다음 샘플 템플릿은 Amazon EFS 파일 시스템과 연결되는 EC2 인스턴스를 Auto Scaling 그룹에 배포합니다. 인스턴스를 파일 시스템과 연결하기 위해 인스턴스에서 cfn-init 헬퍼 스크립트를 실행합니다. 그러면 헬퍼 스크립트에서 nfs-utils yum 패키지를 다운로드하여 설치하고, 새 디렉터리를 생성한 다음 파일 시스템의 DNS 이름을 사용하여 해당 디렉터리에 파일 시스템을 탑재합니다. 파일 시스템의 DNS 이름은 Amazon EC2 인스턴스의 가용 영역에서 탑재 대상의 IP 주소를 확인합니다. DNS 이름 구조에 대한 자세한 내용을 알아보려면 Amazon Elastic File System User Guide(Amazon Elastic File System 사용 설명서)의 [Mounting File Systems](#)(EFS 파일 시스템 탑재)를 참조하세요.

네트워크 파일 시스템 활동을 측정하기 위해 템플릿에 사용자 지정 Amazon CloudWatch 지표가 포함되어 있습니다. 또한 템플릿은 VPC, 서브넷 및 보안 그룹을 생성합니다. 인스턴스가 파일 시스템과 통신하도록 허용하려면 VPC에서 DNS를 활성화하고 탑재 대상과 EC2 인스턴스가 동일한 가용 영역(AZ)에 있어야 합니다. 가용 영역은 서브넷에 의해 지정됩니다.

탑재 대상의 보안 그룹은 TCP 포트 2049에 대한 네트워크 연결을 활성화합니다. NFSv4 클라이언트가 파일 시스템을 탑재하려면 네트워크 연결이 필요합니다. EC2 인스턴스 및 탑재 대상의 보안 그룹에 대한 자세한 내용을 알아보려면 [Amazon Elastic File System User Guide](#)(Amazon Elastic File System 사용 설명서)의 [Security](#)(보안)를 참조하세요.

Note

탑재 대상을 업데이트하여 대체될 경우 연결된 파일 시스템을 사용하는 인스턴스 또는 애플리케이션이 중단될 수 있습니다. 그러면 작성한 커밋되지 않은 내용이 손실될 수 있습니다. 중단을 방지하려면 원하는 용량을 0으로 설정하여 탑재 대상을 업데이트할 때 인스턴스를 중지합니다. 그러면 탑재 대상이 삭제되기 이전에 인스턴스에서 파일 시스템의 탑재를 해제할 수 있습니다. 탑재 업데이트가 완료된 이후에 원하는 용량을 설정하여 후속 업데이트에서 인스턴스를 시작합니다.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
```

```
"Description": "This template creates an Amazon EFS file system and mount target and associates it with Amazon EC2 instances in an Auto Scaling group. **WARNING** This template creates Amazon EC2 instances and related resources. You will be billed for the AWS resources used if you create a stack from this template.",
```

```
"Parameters": {
  "InstanceType" : {
    "Description" : "WebServer EC2 instance type",
    "Type" : "String",
    "Default" : "t2.small",
    "AllowedValues" : [
      "t1.micro",
      "t2.nano",
      "t2.micro",
      "t2.small",
      "t2.medium",
      "t2.large",
      "m1.small",
      "m1.medium",
      "m1.large",
      "m1.xlarge",
      "m2.xlarge",
      "m2.2xlarge",
      "m2.4xlarge",
      "m3.medium",
      "m3.large",
      "m3.xlarge",
      "m3.2xlarge",
      "m4.large",
      "m4.xlarge",
      "m4.2xlarge",
      "m4.4xlarge",
      "m4.10xlarge",
      "c1.medium",
      "c1.xlarge",
      "c3.large",
      "c3.xlarge",
      "c3.2xlarge",
      "c3.4xlarge",
      "c3.8xlarge",
      "c4.large",
      "c4.xlarge",
      "c4.2xlarge",
      "c4.4xlarge",
      "c4.8xlarge",
```

```

        "g2.2xlarge",
        "g2.8xlarge",
        "r3.large",
        "r3.xlarge",
        "r3.2xlarge",
        "r3.4xlarge",
        "r3.8xlarge",
        "i2.xlarge",
        "i2.2xlarge",
        "i2.4xlarge",
        "i2.8xlarge",
        "d2.xlarge",
        "d2.2xlarge",
        "d2.4xlarge",
        "d2.8xlarge",
        "hi1.4xlarge",
        "hs1.8xlarge",
        "cr1.8xlarge",
        "cc2.8xlarge",
        "cg1.4xlarge"
    ],
    "ConstraintDescription" : "must be a valid EC2 instance type."
},
"KeyName": {
    "Type": "AWS::EC2::KeyPair::KeyName",
    "Description": "Name of an existing EC2 key pair to enable SSH access to the EC2 instances"
},
"AsgMaxSize": {
    "Type": "Number",
    "Description": "Maximum size and initial desired capacity of Auto Scaling Group",
    "Default": "2"
},
"SSHLocation" : {
    "Description" : "The IP address range that can be used to connect to the EC2 instances by using SSH",
    "Type": "String",
    "MinLength": "9",
    "MaxLength": "18",
    "Default": "0.0.0.0/0",
    "AllowedPattern": "(\\d{1,3})\\. (\\d{1,3})\\. (\\d{1,3})\\. (\\d{1,3})/(\\d{1,2})",
    "ConstraintDescription": "must be a valid IP CIDR range of the form x.x.x.x/x."
},
"VolumeName" : {

```



```

    "Description" : "The name to be used for the EFS volume",
    "Type": "String",
    "MinLength": "1",
    "Default": "myEFSvolume"
  },
  "MountPoint" : {
    "Description" : "The Linux mount point for the EFS volume",
    "Type": "String",
    "MinLength": "1",
    "Default": "myEFSvolume"
  }
},
"Mappings" : {
  "AWSInstanceType2Arch" : {
    "t1.micro"      : { "Arch" : "HVM64" },
    "t2.nano"      : { "Arch" : "HVM64" },
    "t2.micro"      : { "Arch" : "HVM64" },
    "t2.small"     : { "Arch" : "HVM64" },
    "t2.medium"    : { "Arch" : "HVM64" },
    "t2.large"     : { "Arch" : "HVM64" },
    "m1.small"     : { "Arch" : "HVM64" },
    "m1.medium"    : { "Arch" : "HVM64" },
    "m1.large"     : { "Arch" : "HVM64" },
    "m1.xlarge"    : { "Arch" : "HVM64" },
    "m2.xlarge"    : { "Arch" : "HVM64" },
    "m2.2xlarge"   : { "Arch" : "HVM64" },
    "m2.4xlarge"   : { "Arch" : "HVM64" },
    "m3.medium"    : { "Arch" : "HVM64" },
    "m3.large"     : { "Arch" : "HVM64" },
    "m3.xlarge"    : { "Arch" : "HVM64" },
    "m3.2xlarge"   : { "Arch" : "HVM64" },
    "m4.large"     : { "Arch" : "HVM64" },
    "m4.xlarge"    : { "Arch" : "HVM64" },
    "m4.2xlarge"   : { "Arch" : "HVM64" },
    "m4.4xlarge"   : { "Arch" : "HVM64" },
    "m4.10xlarge"  : { "Arch" : "HVM64" },
    "c1.medium"    : { "Arch" : "HVM64" },
    "c1.xlarge"    : { "Arch" : "HVM64" },
    "c3.large"     : { "Arch" : "HVM64" },
    "c3.xlarge"    : { "Arch" : "HVM64" },
    "c3.2xlarge"   : { "Arch" : "HVM64" },
    "c3.4xlarge"   : { "Arch" : "HVM64" },
    "c3.8xlarge"   : { "Arch" : "HVM64" },
    "c4.large"     : { "Arch" : "HVM64" },

```

```

    "c4.xlarge" : { "Arch" : "HVM64" },
    "c4.2xlarge" : { "Arch" : "HVM64" },
    "c4.4xlarge" : { "Arch" : "HVM64" },
    "c4.8xlarge" : { "Arch" : "HVM64" },
    "g2.2xlarge" : { "Arch" : "HVMG2" },
    "g2.8xlarge" : { "Arch" : "HVMG2" },
    "r3.large" : { "Arch" : "HVM64" },
    "r3.xlarge" : { "Arch" : "HVM64" },
    "r3.2xlarge" : { "Arch" : "HVM64" },
    "r3.4xlarge" : { "Arch" : "HVM64" },
    "r3.8xlarge" : { "Arch" : "HVM64" },
    "i2.xlarge" : { "Arch" : "HVM64" },
    "i2.2xlarge" : { "Arch" : "HVM64" },
    "i2.4xlarge" : { "Arch" : "HVM64" },
    "i2.8xlarge" : { "Arch" : "HVM64" },
    "d2.xlarge" : { "Arch" : "HVM64" },
    "d2.2xlarge" : { "Arch" : "HVM64" },
    "d2.4xlarge" : { "Arch" : "HVM64" },
    "d2.8xlarge" : { "Arch" : "HVM64" },
    "hi1.4xlarge" : { "Arch" : "HVM64" },
    "hs1.8xlarge" : { "Arch" : "HVM64" },
    "cr1.8xlarge" : { "Arch" : "HVM64" },
    "cc2.8xlarge" : { "Arch" : "HVM64" }
  },
  "AWSRegionArch2AMI" : {
    "us-east-1" : { "HVM64" : "ami-0ff8a91507f77f867", "HVMG2" :
"ami-0a584ac55a7631c0c"},
    "us-west-2" : { "HVM64" : "ami-a0cfeed8", "HVMG2" :
"ami-0e09505bc235aa82d"},
    "us-west-1" : { "HVM64" : "ami-0bdb828fd58c52235", "HVMG2" :
"ami-066ee5fd4a9ef77f1"},
    "eu-west-1" : { "HVM64" : "ami-047bb4163c506cd98", "HVMG2" :
"ami-0a7c483d527806435"},
    "eu-west-2" : { "HVM64" : "ami-f976839e", "HVMG2" : "NOT_SUPPORTED"},
    "eu-west-3" : { "HVM64" : "ami-0ebc281c20e89ba4b", "HVMG2" :
"NOT_SUPPORTED"},
    "eu-central-1" : { "HVM64" : "ami-0233214e13e500f77", "HVMG2" :
"ami-06223d46a6d0661c7"},
    "ap-northeast-1" : { "HVM64" : "ami-06cd52961ce9f0d85", "HVMG2" :
"ami-053cdd503598e4a9d"},
    "ap-northeast-2" : { "HVM64" : "ami-0a10b2721688ce9d2", "HVMG2" :
"NOT_SUPPORTED"},
    "ap-northeast-3" : { "HVM64" : "ami-0d98120a9fb693f07", "HVMG2" :
"NOT_SUPPORTED"},

```

```

    "ap-southeast-1" : {"HVM64" : "ami-08569b978cc4dfa10", "HVMG2" :
"ami-0be9df32ae9f92309"},
    "ap-southeast-2" : {"HVM64" : "ami-09b42976632b27e9b", "HVMG2" :
"ami-0a9ce9fecc3d1daf8"},
    "ap-south-1" : {"HVM64" : "ami-0912f71e06545ad88", "HVMG2" :
"ami-097b15e89dbdcfcf4"},
    "us-east-2" : {"HVM64" : "ami-0b59bfac6be064b78", "HVMG2" :
"NOT_SUPPORTED"},
    "ca-central-1" : {"HVM64" : "ami-0b18956f", "HVMG2" : "NOT_SUPPORTED"},
    "sa-east-1" : {"HVM64" : "ami-07b14488da8ea02a0", "HVMG2" :
"NOT_SUPPORTED"},
    "cn-north-1" : {"HVM64" : "ami-0a4eaf6c4454eda75", "HVMG2" :
"NOT_SUPPORTED"},
    "cn-northwest-1" : {"HVM64" : "ami-6b6a7d09", "HVMG2" : "NOT_SUPPORTED"}
  }
},
"Resources": {
  "CloudWatchPutMetricsRole" : {
    "Type" : "AWS::IAM::Role",
    "Properties" : {
      "AssumeRolePolicyDocument" : {
        "Statement" : [ {
          "Effect" : "Allow",
          "Principal" : {
            "Service" : [ "ec2.amazonaws.com" ]
          },
          "Action" : [ "sts:AssumeRole" ]
        } ]
      },
      "Path" : "/"
    }
  },
  "CloudWatchPutMetricsRolePolicy" : {
    "Type" : "AWS::IAM::Policy",
    "Properties" : {
      "PolicyName" : "CloudWatch_PutMetricData",
      "PolicyDocument" : {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Sid": "CloudWatchPutMetricData",
            "Effect": "Allow",
            "Action": ["cloudwatch:PutMetricData"],
            "Resource": ["*"]
          }
        ]
      }
    }
  }
}

```

```

        }
      ]
    },
    "Roles" : [ { "Ref" : "CloudWatchPutMetricsRole" } ]
  }
},
"CloudWatchPutMetricsInstanceProfile" : {
  "Type" : "AWS::IAM::InstanceProfile",
  "Properties" : {
    "Path" : "/",
    "Roles" : [ { "Ref" : "CloudWatchPutMetricsRole" } ]
  }
},
"VPC": {
  "Type": "AWS::EC2::VPC",
  "Properties": {
    "EnableDnsSupport" : "true",
    "EnableDnsHostnames" : "true",
    "CidrBlock": "10.0.0.0/16",
    "Tags": [ { "Key": "Application", "Value": { "Ref": "AWS::StackId" } } ]
  }
},
"InternetGateway" : {
  "Type" : "AWS::EC2::InternetGateway",
  "Properties" : {
    "Tags" : [
      { "Key" : "Application", "Value" : { "Ref" : "AWS::StackName" } },
      { "Key" : "Network", "Value" : "Public" }
    ]
  }
},
"GatewayToInternet" : {
  "Type" : "AWS::EC2::VPCGatewayAttachment",
  "Properties" : {
    "VpcId" : { "Ref" : "VPC" },
    "InternetGatewayId" : { "Ref" : "InternetGateway" }
  }
},
"RouteTable":{
  "Type":"AWS::EC2::RouteTable",
  "Properties":{
    "VpcId": {"Ref":"VPC"}
  }
},

```

```

    "SubnetRouteTableAssoc": {
      "Type" : "AWS::EC2::SubnetRouteTableAssociation",
      "Properties" : {
        "RouteTableId" : {"Ref":"RouteTable"},
        "SubnetId" : {"Ref":"Subnet"}
      }
    },
    "InternetGatewayRoute": {
      "Type": "AWS::EC2::Route",
      "Properties": {
        "DestinationCidrBlock": "0.0.0.0/0",
        "RouteTableId": {"Ref": "RouteTable"},
        "GatewayId": {"Ref": "InternetGateway"}
      }
    },
    "Subnet": {
      "Type": "AWS::EC2::Subnet",
      "Properties": {
        "VpcId": { "Ref": "VPC" },
        "CidrBlock": "10.0.0.0/24",
        "Tags": [ { "Key": "Application", "Value": { "Ref": "AWS::StackId" } } ]
      }
    },
    "InstanceSecurityGroup": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "VpcId": { "Ref": "VPC" },
        "GroupDescription": "Enable SSH access via port 22",
        "SecurityGroupIngress": [
          { "IpProtocol": "tcp", "FromPort": 22, "ToPort": 22, "CidrIp": { "Ref":
"SSHLocation" } },
          { "IpProtocol": "tcp", "FromPort": 80, "ToPort": 80, "CidrIp": "0.0.0.0/0" }
        ]
      }
    },
    "MountTargetSecurityGroup": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "VpcId": { "Ref": "VPC" },
        "GroupDescription": "Security group for mount target",
        "SecurityGroupIngress": [
          {
            "IpProtocol": "tcp",
            "FromPort": 2049,

```

```

        "ToPort": 2049,
        "CidrIp": "0.0.0.0/0"
    }
]
},
"FileSystem": {
    "Type": "AWS::EFS::FileSystem",
    "Properties": {
        "PerformanceMode": "generalPurpose",
        "FileSystemTags": [
            {
                "Key": "Name",
                "Value": { "Ref" : "VolumeName" }
            }
        ]
    }
},
"MountTarget": {
    "Type": "AWS::EFS::MountTarget",
    "Properties": {
        "FileSystemId": { "Ref": "FileSystem" },
        "SubnetId": { "Ref": "Subnet" },
        "SecurityGroups": [ { "Ref": "MountTargetSecurityGroup" } ]
    }
},
"LaunchConfiguration": {
    "Type": "AWS::AutoScaling::LaunchConfiguration",
    "Metadata" : {
        "AWS::CloudFormation::Init" : {
            "configSets" : {
                "MountConfig" : [ "setup", "mount" ]
            },
            "setup" : {
                "packages" : {
                    "yum" : {
                        "nfs-utils" : []
                    }
                },
                "files" : {
                    "/home/ec2-user/post_nfsstat" : {
                        "content" : { "Fn::Join" : [ "", [
                            "#!/bin/bash\n",
                            "\n",

```

```

        "INPUT=\"$(cat)\"\n",
        "CW_JSON_OPEN='{ \"Namespace\": \"EFS\", \"MetricData\": [ '\n",
        "CW_JSON_CLOSE=' ] }'\n",
        "CW_JSON_METRIC=''\n",
        "METRIC_COUNTER=0\n",
        "\n",
        "for COL in 1 2 3 4 5 6; do\n",
        "\n",
        "  COUNTER=0\n",
        "  METRIC_FIELD=$COL\n",
        "  DATA_FIELD=$(( $COL + ( $COL - 1 )))\n",
        "\n",
        "  while read line; do\n",
        "    if [[ COUNTER -gt 0 ]]; then\n",
        "\n",
        "      LINE=`echo $line | tr -s ' ' '\n",
        "      AWS_COMMAND=\"aws cloudwatch put-metric-data --region ",
{ "Ref": "AWS::Region" }, "\n",
        "      MOD=$(( $COUNTER % 2 ))\n",
        "\n",
        "      if [ $MOD -eq 1 ]; then\n",
        "        METRIC_NAME=`echo $LINE | cut -d ' ' -f $METRIC_FIELD`\n",
\n",
        "        else\n",
        "          METRIC_VALUE=`echo $LINE | cut -d ' ' -f $DATA_FIELD`\n",
        "        fi\n",
        "\n",
        "        if [[ -n \"$METRIC_NAME\" && -n \"$METRIC_VALUE\" ]]; then\n",
\n",
        "          INSTANCE_ID=$(curl -s http://169.254.169.254/latest/meta-
data/instance-id)\n",
        "          CW_JSON_METRIC=\"${CW_JSON_METRIC} { \\\"MetricName\\\": \\\"
\\\"$METRIC_NAME\\\", \\\"Dimensions\\\": [{\\\"Name\\\": \\\"InstanceId\\\", \\\"Value\\\"
\\\": \\\"$INSTANCE_ID\\\"} ], \\\"Value\\\": $METRIC_VALUE },\n",
        "          unset METRIC_NAME\n",
        "          unset METRIC_VALUE\n",
        "\n",
        "          METRIC_COUNTER=$((METRIC_COUNTER+1))\n",
        "          if [ $METRIC_COUNTER -eq 20 ]; then\n",
        "            # 20 is max metric collection size, so we have to
submit here\n",
        "            aws cloudwatch put-metric-data --region ", { "Ref":
"AWS::Region" }, " --cli-input-json \"`echo $CW_JSON_OPEN ${CW_JSON_METRIC%?}
$CW_JSON_CLOSE`\n",

```

```

        "\n",
        "        # reset\n",
        "        METRIC_COUNTER=0\n",
        "        CW_JSON_METRIC=' '\n",
        "    fi\n",
        "fi\n",
        "\n",
        "\n",
        "\n",
        "    COUNTER=$((COUNTER+1))\n",
        "    fi\n",
        "\n",
        "    if [[ \"$line\" == \"Client nfs v4:\" ]]; then\n",
        "        # the next line is the good stuff\n",
        "        COUNTER=$((COUNTER+1))\n",
        "    fi\n",
        "done <<< \"$INPUT\"\n",
        "done\n",
        "\n",
        "# submit whatever is left\n",
        "aws cloudwatch put-metric-data --region ", { "Ref":
"AWS::Region" }, " --cli-input-json \"`echo $CW_JSON_OPEN ${CW_JSON_METRIC%?}
$CW_JSON_CLOSE`\n"
    ] ] },
    "mode": "000755",
    "owner": "ec2-user",
    "group": "ec2-user"
},
"/home/ec2-user/crontab" : {
    "content" : { "Fn::Join" : [ "", [
        "** * * * * /usr/sbin/nfsstat | /home/ec2-user/post_nfsstat\n"
    ] ] },
    "owner": "ec2-user",
    "group": "ec2-user"
}
},
"commands" : {
    "01_createdir" : {
        "command" : {"Fn::Join" : [ "", [ "mkdir /", { "Ref" :
"MountPoint" }]]}
    }
}
},
"mount" : {

```



```

        "commands" : {
            "01_mount" : {
                "command" : { "Fn::Sub": "sudo mount -t nfs4 -o
nfsvers=4.1,rsize=1048576,wsiz=1048576,hard,timeo=600,retrans=2 ${FileSystem}.efs.
${AWS::Region}.amazonaws.com:/ /${MountPoint}" }
            },
            "02_permissions" : {
                "command" : { "Fn::Join" : [ "", [ "chown ec2-user:ec2-user /",
{ "Ref" : "MountPoint" } ] ] ] }
            }
        }
    },
    "Properties": {
        "AssociatePublicIpAddress" : true,
        "ImageId": {
            "Fn::FindInMap": [ "AWSRegionArch2AMI", { "Ref": "AWS::Region" }, {
            "Fn::FindInMap": [ "AWSInstanceType2Arch", { "Ref": "InstanceType" },
"Arch" ]
        } ]
    },
    "InstanceType": { "Ref": "InstanceType" },
    "KeyName": { "Ref": "KeyName" },
    "SecurityGroups": [ { "Ref": "InstanceSecurityGroup" } ],
    "IamInstanceProfile" : { "Ref" : "CloudWatchPutMetricsInstanceProfile" },
    "UserData" : { "Fn::Base64" : { "Fn::Join" : [ "", [
        "#!/bin/bash -xe\n",
        "yum install -y aws-cfn-bootstrap\n",

        "/opt/aws/bin/cfn-init -v ",
        "    --stack ", { "Ref" : "AWS::StackName" },
        "    --resource LaunchConfiguration ",
        "    --configsets MountConfig ",
        "    --region ", { "Ref" : "AWS::Region" }, "\n",

        "crontab /home/ec2-user/crontab\n",

        "/opt/aws/bin/cfn-signal -e $? ",
        "    --stack ", { "Ref" : "AWS::StackName" },
        "    --resource AutoScalingGroup ",
        "    --region ", { "Ref" : "AWS::Region" }, "\n"
    ] ] ] }
    }
}

```

```

    },
    "AutoScalingGroup": {
      "Type": "AWS::AutoScaling::AutoScalingGroup",
      "DependsOn": ["MountTarget", "GatewayToInternet"],
      "CreationPolicy" : {
        "ResourceSignal" : {
          "Timeout" : "PT15M",
          "Count" : { "Ref": "AsgMaxSize" }
        }
      },
    },
    "Properties": {
      "VPCZoneIdentifier": [ { "Ref": "Subnet" } ],
      "LaunchConfigurationName": { "Ref": "LaunchConfiguration" },
      "MinSize": "1",
      "MaxSize": { "Ref": "AsgMaxSize" },
      "DesiredCapacity": { "Ref": "AsgMaxSize" },
      "Tags": [ {
        "Key": "Name",
        "Value": "EFS FileSystem Mounted Instance",
        "PropagateAtLaunch": "true"
      } ]
    }
  }
},
"Outputs" : {
  "MountTargetID" : {
    "Description" : "Mount target ID",
    "Value" : { "Ref" : "MountTarget" }
  },
  "FileSystemID" : {
    "Description" : "File system ID",
    "Value" : { "Ref" : "FileSystem" }
  }
}
}
}

```

YAML

AWSTemplateFormatVersion: '2010-09-09'

Description: This template creates an Amazon EFS file system and mount target and associates it with Amazon EC2 instances in an Auto Scaling group. ****WARNING**** This template creates Amazon EC2 instances and related resources. You will be billed for the AWS resources used if you create a stack from this template.

Parameters:**InstanceType:**

Description: WebServer EC2 instance type

Type: String

Default: t2.small

AllowedValues:

- t1.micro
- t2.nano
- t2.micro
- t2.small
- t2.medium
- t2.large
- m1.small
- m1.medium
- m1.large
- m1.xlarge
- m2.xlarge
- m2.2xlarge
- m2.4xlarge
- m3.medium
- m3.large
- m3.xlarge
- m3.2xlarge
- m4.large
- m4.xlarge
- m4.2xlarge
- m4.4xlarge
- m4.10xlarge
- c1.medium
- c1.xlarge
- c3.large
- c3.xlarge
- c3.2xlarge
- c3.4xlarge
- c3.8xlarge
- c4.large
- c4.xlarge
- c4.2xlarge
- c4.4xlarge
- c4.8xlarge
- g2.2xlarge
- g2.8xlarge
- r3.large
- r3.xlarge

- r3.2xlarge
- r3.4xlarge
- r3.8xlarge
- i2.xlarge
- i2.2xlarge
- i2.4xlarge
- i2.8xlarge
- d2.xlarge
- d2.2xlarge
- d2.4xlarge
- d2.8xlarge
- hi1.4xlarge
- hs1.8xlarge
- cr1.8xlarge
- cc2.8xlarge
- cg1.4xlarge

ConstraintDescription: must be a valid EC2 instance type.

KeyName:

Type: AWS::EC2::KeyPair::KeyName

Description: Name of an existing EC2 key pair to enable SSH access to the ECS instances

AsgMaxSize:

Type: Number

Description: Maximum size and initial desired capacity of Auto Scaling Group

Default: '2'

SSHLocation:

Description: The IP address range that can be used to connect to the EC2 instances by using SSH

Type: String

MinLength: '9'

MaxLength: '18'

Default: 0.0.0.0/0

AllowedPattern: "(\d{1,3})\.\.(\d{1,3})\.\.(\d{1,3})\.\.(\d{1,3})/(\d{1,2})"

ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.

VolumeName:

Description: The name to be used for the EFS volume

Type: String

MinLength: '1'

Default: myEFSvolume

MountPoint:

Description: The Linux mount point for the EFS volume

Type: String

MinLength: '1'

Default: myEFSvolume

Mappings:**AWSInstanceType2Arch:**

```
t1.micro:
  Arch: HVM64
t2.nano:
  Arch: HVM64
t2.micro:
  Arch: HVM64
t2.small:
  Arch: HVM64
t2.medium:
  Arch: HVM64
t2.large:
  Arch: HVM64
m1.small:
  Arch: HVM64
m1.medium:
  Arch: HVM64
m1.large:
  Arch: HVM64
m1.xlarge:
  Arch: HVM64
m2.xlarge:
  Arch: HVM64
m2.2xlarge:
  Arch: HVM64
m2.4xlarge:
  Arch: HVM64
m3.medium:
  Arch: HVM64
m3.large:
  Arch: HVM64
m3.xlarge:
  Arch: HVM64
m3.2xlarge:
  Arch: HVM64
m4.large:
  Arch: HVM64
m4.xlarge:
  Arch: HVM64
m4.2xlarge:
  Arch: HVM64
m4.4xlarge:
  Arch: HVM64
```

```
m4.10xlarge:
  Arch: HVM64
c1.medium:
  Arch: HVM64
c1.xlarge:
  Arch: HVM64
c3.large:
  Arch: HVM64
c3.xlarge:
  Arch: HVM64
c3.2xlarge:
  Arch: HVM64
c3.4xlarge:
  Arch: HVM64
c3.8xlarge:
  Arch: HVM64
c4.large:
  Arch: HVM64
c4.xlarge:
  Arch: HVM64
c4.2xlarge:
  Arch: HVM64
c4.4xlarge:
  Arch: HVM64
c4.8xlarge:
  Arch: HVM64
g2.2xlarge:
  Arch: HVMG2
g2.8xlarge:
  Arch: HVMG2
r3.large:
  Arch: HVM64
r3.xlarge:
  Arch: HVM64
r3.2xlarge:
  Arch: HVM64
r3.4xlarge:
  Arch: HVM64
r3.8xlarge:
  Arch: HVM64
i2.xlarge:
  Arch: HVM64
i2.2xlarge:
  Arch: HVM64
```

```
i2.4xlarge:
  Arch: HVM64
i2.8xlarge:
  Arch: HVM64
d2.xlarge:
  Arch: HVM64
d2.2xlarge:
  Arch: HVM64
d2.4xlarge:
  Arch: HVM64
d2.8xlarge:
  Arch: HVM64
hi1.4xlarge:
  Arch: HVM64
hs1.8xlarge:
  Arch: HVM64
cr1.8xlarge:
  Arch: HVM64
cc2.8xlarge:
  Arch: HVM64
AWSRegionArch2AMI:
us-east-1:
  HVM64: ami-0ff8a91507f77f867
  HVMG2: ami-0a584ac55a7631c0c
us-west-2:
  HVM64: ami-a0cfeed8
  HVMG2: ami-0e09505bc235aa82d
us-west-1:
  HVM64: ami-0bdb828fd58c52235
  HVMG2: ami-066ee5fd4a9ef77f1
eu-west-1:
  HVM64: ami-047bb4163c506cd98
  HVMG2: ami-0a7c483d527806435
eu-west-2:
  HVM64: ami-f976839e
  HVMG2: NOT_SUPPORTED
eu-west-3:
  HVM64: ami-0ebc281c20e89ba4b
  HVMG2: NOT_SUPPORTED
eu-central-1:
  HVM64: ami-0233214e13e500f77
  HVMG2: ami-06223d46a6d0661c7
ap-northeast-1:
  HVM64: ami-06cd52961ce9f0d85
```

```

    HVMG2: ami-053cdd503598e4a9d
  ap-northeast-2:
    HVM64: ami-0a10b2721688ce9d2
    HVMG2: NOT_SUPPORTED
  ap-northeast-3:
    HVM64: ami-0d98120a9fb693f07
    HVMG2: NOT_SUPPORTED
  ap-southeast-1:
    HVM64: ami-08569b978cc4dfa10
    HVMG2: ami-0be9df32ae9f92309
  ap-southeast-2:
    HVM64: ami-09b42976632b27e9b
    HVMG2: ami-0a9ce9fecc3d1daf8
  ap-south-1:
    HVM64: ami-0912f71e06545ad88
    HVMG2: ami-097b15e89dbdcfcf4
  us-east-2:
    HVM64: ami-0b59bfac6be064b78
    HVMG2: NOT_SUPPORTED
  ca-central-1:
    HVM64: ami-0b18956f
    HVMG2: NOT_SUPPORTED
  sa-east-1:
    HVM64: ami-07b14488da8ea02a0
    HVMG2: NOT_SUPPORTED
  cn-north-1:
    HVM64: ami-0a4eaf6c4454eda75
    HVMG2: NOT_SUPPORTED
  cn-northwest-1:
    HVM64: ami-6b6a7d09
    HVMG2: NOT_SUPPORTED

```

Resources:

CloudWatchPutMetricsRole:

Type: AWS::IAM::Role

Properties:

AssumeRolePolicyDocument:

Statement:

- Effect: Allow

Principal:

Service:

- ec2.amazonaws.com

Action:

- sts:AssumeRole

Path: "/"


```
CloudWatchPutMetricsRolePolicy:
  Type: AWS::IAM::Policy
  Properties:
    PolicyName: CloudWatch_PutMetricData
    PolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Sid: CloudWatchPutMetricData
          Effect: Allow
          Action:
            - cloudwatch:PutMetricData
          Resource:
            - "*"

    Roles:
      - Ref: CloudWatchPutMetricsRole
CloudWatchPutMetricsInstanceProfile:
  Type: AWS::IAM::InstanceProfile
  Properties:
    Path: "/"
    Roles:
      - Ref: CloudWatchPutMetricsRole
VPC:
  Type: AWS::EC2::VPC
  Properties:
    EnableDnsSupport: 'true'
    EnableDnsHostnames: 'true'
    CidrBlock: 10.0.0.0/16
    Tags:
      - Key: Application
        Value:
          Ref: AWS::StackId
InternetGateway:
  Type: AWS::EC2::InternetGateway
  Properties:
    Tags:
      - Key: Application
        Value:
          Ref: AWS::StackName
      - Key: Network
        Value: Public
GatewayToInternet:
  Type: AWS::EC2::VPCEGatewayAttachment
  Properties:
    VpcId:
```

```
    Ref: VPC
  InternetGatewayId:
    Ref: InternetGateway
RouteTable:
  Type: AWS::EC2::RouteTable
  Properties:
    VpcId:
      Ref: VPC
SubnetRouteTableAssoc:
  Type: AWS::EC2::SubnetRouteTableAssociation
  Properties:
    RouteTableId:
      Ref: RouteTable
    SubnetId:
      Ref: Subnet
InternetGatewayRoute:
  Type: AWS::EC2::Route
  Properties:
    DestinationCidrBlock: 0.0.0.0/0
    RouteTableId:
      Ref: RouteTable
    GatewayId:
      Ref: InternetGateway
Subnet:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId:
      Ref: VPC
    CidrBlock: 10.0.0.0/24
    Tags:
      - Key: Application
        Value:
          Ref: AWS::StackId
InstanceSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    VpcId:
      Ref: VPC
    GroupDescription: Enable SSH access via port 22
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp:
```

```
    Ref: SSHLocation
  - IpProtocol: tcp
    FromPort: 80
    ToPort: 80
    CidrIp: 0.0.0.0/0
MountTargetSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    VpcId:
      Ref: VPC
    GroupDescription: Security group for mount target
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 2049
        ToPort: 2049
        CidrIp: 0.0.0.0/0
FileSystem:
  Type: AWS::EFS::FileSystem
  Properties:
    PerformanceMode: generalPurpose
    FileSystemTags:
      - Key: Name
        Value:
          Ref: VolumeName
MountTarget:
  Type: AWS::EFS::MountTarget
  Properties:
    FileSystemId:
      Ref: FileSystem
    SubnetId:
      Ref: Subnet
    SecurityGroups:
      - Ref: MountTargetSecurityGroup
LaunchConfiguration:
  Type: AWS::AutoScaling::LaunchConfiguration
  Metadata:
    AWS::CloudFormation::Init:
      configSets:
        MountConfig:
          - setup
          - mount
      setup:
        packages:
          yum:
```

```

        nfs-utils: []
files:
  "/home/ec2-user/post_nfsstat":
    content: !Sub |
      #!/bin/bash

      INPUT="$(cat)"
      CW_JSON_OPEN='{ "Namespace": "EFS", "MetricData": [ '
      CW_JSON_CLOSE=' ] }'
      CW_JSON_METRIC=''
      METRIC_COUNTER=0

      for COL in 1 2 3 4 5 6; do

        COUNTER=0
        METRIC_FIELD=$COL
        DATA_FIELD=$(( $COL + ( $COL - 1 ) ))

        while read line; do
          if [[ COUNTER -gt 0 ]]; then

            LINE=`echo $line | tr -s ' ' `
            AWS_COMMAND="aws cloudwatch put-metric-data --region
${AWS::Region}"
            MOD=$(( $COUNTER % 2 ))

            if [ $MOD -eq 1 ]; then
              METRIC_NAME=`echo $LINE | cut -d ' ' -f $METRIC_FIELD`
            else
              METRIC_VALUE=`echo $LINE | cut -d ' ' -f $DATA_FIELD`
            fi

            if [[ -n "$METRIC_NAME" && -n "$METRIC_VALUE" ]]; then
              INSTANCE_ID=$(curl -s http://169.254.169.254/latest/meta-data/
instance-id)
              CW_JSON_METRIC="$CW_JSON_METRIC { \"MetricName\": \"$METRIC_NAME
\", \"Dimensions\": [{\"Name\": \"InstanceId\", \"Value\": \"$INSTANCE_ID\"} ], \"Value
\": $METRIC_VALUE },"
              unset METRIC_NAME
              unset METRIC_VALUE

              METRIC_COUNTER=$((METRIC_COUNTER+1))
              if [ $METRIC_COUNTER -eq 20 ]; then
                # 20 is max metric collection size, so we have to submit here

```

```

        aws cloudwatch put-metric-data --region ${AWS::Region} --cli-
input-json "`echo $CW_JSON_OPEN ${!CW_JSON_METRIC%?} $CW_JSON_CLOSE`"

        # reset
        METRIC_COUNTER=0
        CW_JSON_METRIC=''
    fi
fi

    COUNTER=$((COUNTER+1))
fi

    if [[ "$line" == "Client nfs v4:" ]]; then
        # the next line is the good stuff
        COUNTER=$((COUNTER+1))
    fi
done <<< "$INPUT"
done

# submit whatever is left
aws cloudwatch put-metric-data --region ${AWS::Region} --cli-input-json
"`echo $CW_JSON_OPEN ${!CW_JSON_METRIC%?} $CW_JSON_CLOSE`"
mode: '000755'
owner: ec2-user
group: ec2-user
"/home/ec2-user/crontab":
    content: "* * * * * /usr/sbin/nfsstat | /home/ec2-user/post_nfsstat\n"
    owner: ec2-user
    group: ec2-user
commands:
    01_createdir:
        command: !Sub "mkdir /${MountPoint}"
mount:
    commands:
        01_mount:
            command: !Sub >
                mount -t nfs4 -o nfsvers=4.1 ${FileSystem}.efs.
${AWS::Region}.amazonaws.com:/ /${MountPoint}
        02_permissions:
            command: !Sub "chown ec2-user:ec2-user /${MountPoint}"
Properties:
    AssociatePublicIpAddress: true

```

```

ImageId:
  Fn::FindInMap:
    - AWSRegionArch2AMI
    - Ref: AWS::Region
    - Fn::FindInMap:
        - AWSInstanceType2Arch
        - Ref: InstanceType
        - Arch
InstanceType:
  Ref: InstanceType
KeyName:
  Ref: KeyName
SecurityGroups:
  - Ref: InstanceSecurityGroup
IamInstanceProfile:
  Ref: CloudWatchPutMetricsInstanceProfile
UserData:
  Fn::Base64: !Sub |
    #!/bin/bash -xe
    yum install -y aws-cfn-bootstrap
    /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --resource
LaunchConfiguration --configsets MountConfig --region ${AWS::Region}
    crontab /home/ec2-user/crontab
    /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource
AutoScalingGroup --region ${AWS::Region}
AutoScalingGroup:
  Type: AWS::AutoScaling::AutoScalingGroup
  DependsOn:
    - MountTarget
    - GatewayToInternet
  CreationPolicy:
    ResourceSignal:
      Timeout: PT15M
      Count:
        Ref: AsgMaxSize
  Properties:
    VPCZoneIdentifier:
      - Ref: Subnet
    LaunchConfigurationName:
      Ref: LaunchConfiguration
    MinSize: '1'
    MaxSize:
      Ref: AsgMaxSize
    DesiredCapacity:

```

```

    Ref: AsgMaxSize
  Tags:
    - Key: Name
      Value: EFS FileSystem Mounted Instance
      PropagateAtLaunch: 'true'
Outputs:
  MountTargetID:
    Description: Mount target ID
    Value:
      Ref: MountTarget
  FileSystemID:
    Description: File system ID
    Value:
      Ref: FileSystem

```

Elastic Beanstalk 템플릿 코드 조각

Elastic Beanstalk 사용을 통해 애플리케이션을 실행하는 인프라에 관계없이 AWS에서 애플리케이션을 신속하게 배포하고 관리할 수 있습니다. 다음 샘플 템플릿은 CloudFormation 템플릿에서 Elastic Beanstalk 리소스를 설명하는 데 도움이 될 수 있습니다.

Elastic Beanstalk 샘플 PHP

다음 샘플 템플릿은 Amazon S3 버킷에 저장되어 있는 샘플 PHP 웹 애플리케이션을 배포합니다. 또한 이 환경은 최소 2개 및 최대 6개의 Amazon EC2 인스턴스가 있는 Auto Scaling, 로드 밸런싱 환경입니다. 레거시 시작 구성을 사용하는 Elastic Beanstalk 환경을 보여줍니다. 대신 시작 템플릿을 사용하는 방법에 대한 자세한 내용은 AWS Elastic Beanstalk 개발자 안내서의 [시작 템플릿을 참조하세요](#).

solution-stack을 솔루션 스택 이름(플랫폼 버전)으로 바꿉니다. 사용 가능한 솔루션 스택 목록을 보려면 AWS CLI 명령 `aws elasticbeanstalk list-available-solution-stacks`를 사용합니다.

JSON

```

{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Resources": {
    "sampleApplication": {
      "Type": "AWS::ElasticBeanstalk::Application",
      "Properties": {
        "Description": "AWS Elastic Beanstalk Sample Application"
      }
    }
  },

```

```

"sampleApplicationVersion": {
  "Type": "AWS::ElasticBeanstalk::ApplicationVersion",
  "Properties": {
    "ApplicationName": {
      "Ref": "sampleApplication"
    },
    "Description": "AWS ElasticBeanstalk Sample Application Version",
    "SourceBundle": {
      "S3Bucket": {
        "Fn::Sub": "elasticbeanstalk-samples-${AWS::Region}"
      },
      "S3Key": "php-newsample-app.zip"
    }
  }
},
"sampleConfigurationTemplate": {
  "Type": "AWS::ElasticBeanstalk::ConfigurationTemplate",
  "Properties": {
    "ApplicationName": {
      "Ref": "sampleApplication"
    },
    "Description": "AWS ElasticBeanstalk Sample Configuration Template",
    "OptionSettings": [
      {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "MinSize",
        "Value": "2"
      },
      {
        "Namespace": "aws:autoscaling:asg",
        "OptionName": "MaxSize",
        "Value": "6"
      },
      {
        "Namespace": "aws:elasticbeanstalk:environment",
        "OptionName": "EnvironmentType",
        "Value": "LoadBalanced"
      },
      {
        "Namespace": "aws:autoscaling:launchconfiguration",
        "OptionName": "IamInstanceProfile",
        "Value": {
          "Ref": "MyInstanceProfile"
        }
      }
    ]
  }
}

```



```

        }
    ],
    "SolutionStackName": "solution-stack"
}
},
"sampleEnvironment": {
    "Type": "AWS::ElasticBeanstalk::Environment",
    "Properties": {
        "ApplicationName": {
            "Ref": "sampleApplication"
        },
        "Description": "AWS ElasticBeanstalk Sample Environment",
        "TemplateName": {
            "Ref": "sampleConfigurationTemplate"
        },
        "VersionLabel": {
            "Ref": "sampleApplicationVersion"
        }
    }
},
"MyInstanceRole": {
    "Type": "AWS::IAM::Role",
    "Properties": {
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "Service": [
                            "ec2.amazonaws.com"
                        ]
                    },
                    "Action": [
                        "sts:AssumeRole"
                    ]
                }
            ]
        }
    },
    "Description": "Beanstalk EC2 role",
    "ManagedPolicyArns": [
        "arn:aws:iam::aws:policy/AWSElasticBeanstalkWebTier",
        "arn:aws:iam::aws:policy/AWSElasticBeanstalkMulticontainerDocker",
        "arn:aws:iam::aws:policy/AWSElasticBeanstalkWorkerTier"
    ]
}

```

```

        ]
      }
    },
    "MyInstanceProfile": {
      "Type": "AWS::IAM::InstanceProfile",
      "Properties": {
        "Roles": [
          {
            "Ref": "MyInstanceRole"
          }
        ]
      }
    }
  }
}

```

YAML

```

AWSTemplateFormatVersion: '2010-09-09'
Resources:
  sampleApplication:
    Type: AWS::ElasticBeanstalk::Application
    Properties:
      Description: AWS Elastic Beanstalk Sample Application
  sampleApplicationVersion:
    Type: AWS::ElasticBeanstalk::ApplicationVersion
    Properties:
      ApplicationName:
        Ref: sampleApplication
      Description: AWS ElasticBeanstalk Sample Application Version
      SourceBundle:
        S3Bucket: !Sub "elasticbeanstalk-samples-${AWS::Region}"
        S3Key: php-newsample-app.zip
  sampleConfigurationTemplate:
    Type: AWS::ElasticBeanstalk::ConfigurationTemplate
    Properties:
      ApplicationName:
        Ref: sampleApplication
      Description: AWS ElasticBeanstalk Sample Configuration Template
      OptionSettings:
        - Namespace: aws:autoscaling:asg
          OptionName: MinSize
          Value: '2'

```

```
- Namespace: aws:autoscaling:asg
  OptionName: MaxSize
  Value: '6'
- Namespace: aws:elasticbeanstalk:environment
  OptionName: EnvironmentType
  Value: LoadBalanced
- Namespace: aws:autoscaling:launchconfiguration
  OptionName: IamInstanceProfile
  Value: !Ref MyInstanceProfile
SolutionStackName: solution-stack
sampleEnvironment:
  Type: AWS::ElasticBeanstalk::Environment
  Properties:
    ApplicationName:
      Ref: sampleApplication
    Description: AWS ElasticBeanstalk Sample Environment
    TemplateName:
      Ref: sampleConfigurationTemplate
    VersionLabel:
      Ref: sampleApplicationVersion
MyInstanceRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: 2012-10-17
      Statement:
        - Effect: Allow
          Principal:
            Service:
              - ec2.amazonaws.com
          Action:
            - sts:AssumeRole
    Description: Beanstalk EC2 role
    ManagedPolicyArns:
      - arn:aws:iam::aws:policy/AWSElasticBeanstalkWebTier
      - arn:aws:iam::aws:policy/AWSElasticBeanstalkMulticontainerDocker
      - arn:aws:iam::aws:policy/AWSElasticBeanstalkWorkerTier
MyInstanceProfile:
  Type: AWS::IAM::InstanceProfile
  Properties:
    Roles:
      - !Ref MyInstanceRole
```

Elastic Load Balancing 템플릿 코드 조각

Application Load Balancer, Network Load Balancer 또는 Gateway Load Balancer를 생성하려면 `AWS::ElasticLoadBalancingV2`로 시작하는 V2 리소스 유형을 사용합니다. Classic Load Balancer를 생성하려면 `AWS::ElasticLoadBalancing`으로 시작하는 리소스 유형을 사용합니다.

내용

- [ELBv2 리소스](#)
- [Classic Load Balancer 리소스](#)

ELBv2 리소스

이 예제에서는 HTTP 리스너와 대상 그룹으로 트래픽을 전달하는 기본 작업을 사용하여 Application Load Balancer를 정의합니다. 로드 밸런서는 기본 상태 확인 설정을 사용합니다. 대상 그룹에는 2개의 등록된 EC2 인스턴스가 있습니다.

YAML

```
Resources:
  myLoadBalancer:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      Name: my-alb
      Type: application
      Scheme: internal
      Subnets:
        - !Ref subnet-AZ1
        - !Ref subnet-AZ2
      SecurityGroups:
        - !Ref mySecurityGroup

  myHTTPListener:
    Type: AWS::ElasticLoadBalancingV2::Listener
    Properties:
      LoadBalancerArn: !Ref myLoadBalancer
      Protocol: HTTP
      Port: 80
      DefaultActions:
        - Type: "forward"
          TargetGroupArn: !Ref myTargetGroup
```

```
myTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    Name: "my-target-group"
    Protocol: HTTP
    Port: 80
    TargetType: instance
    VpcId: !Ref myVPC
    Targets:
      - Id: !GetAtt Instance1.InstanceId
        Port: 80
      - Id: !GetAtt Instance2.InstanceId
        Port: 80
```

JSON

```
{
  "Resources": {
    "myLoadBalancer": {
      "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
      "Properties": {
        "Name": "my-alb",
        "Type": "application",
        "Scheme": "internal",
        "Subnets": [
          {
            "Ref": "subnet-AZ1"
          },
          {
            "Ref": "subnet-AZ2"
          }
        ],
        "SecurityGroups": [
          {
            "Ref": "mySecurityGroup"
          }
        ]
      }
    },
    "myHTTPListener": {
      "Type": "AWS::ElasticLoadBalancingV2::Listener",
      "Properties": {
        "LoadBalancerArn": {
```

```

        "Ref": "myLoadBalancer"
      },
      "Protocol": "HTTP",
      "Port": 80,
      "DefaultActions": [
        {
          "Type": "forward",
          "TargetGroupArn": {
            "Ref": "myTargetGroup"
          }
        }
      ]
    }
  },
  "myTargetGroup": {
    "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
    "Properties": {
      "Name": "my-target-group",
      "Protocol": "HTTP",
      "Port": 80,
      "TargetType": "instance",
      "VpcId": {
        "Ref": "myVPC"
      },
      "Targets": [
        {
          "Id": {
            "Fn::GetAtt": [
              "Instance1",
              "InstanceId"
            ]
          },
          "Port": 80
        },
        {
          "Id": {
            "Fn::GetAtt": [
              "Instance2",
              "InstanceId"
            ]
          },
          "Port": 80
        }
      ]
    }
  }
]

```

```

    }
  }
}

```

Classic Load Balancer 리소스

이 예제에서는 1개의 HTTP 리스너가 있고 등록된 EC2 인스턴스가 없는 Classic Load Balancer를 정의합니다. 로드 밸런서는 기본 상태 확인 설정을 사용합니다.

YAML

```

myLoadBalancer:
  Type: AWS::ElasticLoadBalancing::LoadBalancer
  Properties:
    AvailabilityZones:
      - "us-east-1a"
    Listeners:
      - LoadBalancerPort: '80'
        InstancePort: '80'
        Protocol: HTTP

```

JSON

```

"myLoadBalancer" : {
  "Type" : "AWS::ElasticLoadBalancing::LoadBalancer",
  "Properties" : {
    "AvailabilityZones" : [ "us-east-1a" ],
    "Listeners" : [ {
      "LoadBalancerPort" : "80",
      "InstancePort" : "80",
      "Protocol" : "HTTP"
    } ]
  }
}

```

이 예제에서는 1개의 HTTP 리스너, 2개의 등록된 EC2 인스턴스 및 사용자 지정 상태 확인 설정이 있는 Classic Load Balancer를 정의합니다.

YAML

```
myClassicLoadBalancer:
  Type: AWS::ElasticLoadBalancing::LoadBalancer
  Properties:
    AvailabilityZones:
      - "us-east-1a"
    Instances:
      - Ref: Instance1
      - Ref: Instance2
    Listeners:
      - LoadBalancerPort: '80'
        InstancePort: '80'
        Protocol: HTTP
    HealthCheck:
      Target: HTTP:80/
      HealthyThreshold: '3'
      UnhealthyThreshold: '5'
      Interval: '30'
      Timeout: '5'
```

JSON

```
"myClassicLoadBalancer" : {
  "Type" : "AWS::ElasticLoadBalancing::LoadBalancer",
  "Properties" : {
    "AvailabilityZones" : [ "us-east-1a" ],
    "Instances" : [
      { "Ref" : "Instance1" },
      { "Ref" : "Instance2" }
    ],
    "Listeners" : [ {
      "LoadBalancerPort" : "80",
      "InstancePort" : "80",
      "Protocol" : "HTTP"
    } ],
    "HealthCheck" : {
      "Target" : "HTTP:80/",
      "HealthyThreshold" : "3",
      "UnhealthyThreshold" : "5",
      "Interval" : "30",
      "Timeout" : "5"
    }
  }
}
```



```
}
    }
}
```

AWS Identity and Access Management 템플릿 코드 조각

이 단원에는 AWS Identity and Access Management 템플릿 조각이 포함되어 있습니다.

주제

- [IAM 사용자 리소스 선언](#)
- [IAM 액세스 키 리소스 선언](#)
- [IAM 그룹 리소스 선언](#)
- [그룹에 사용자 추가](#)
- [IAM 정책 선언](#)
- [Amazon S3 버킷 정책 선언](#)
- [Amazon SNS 주제 정책 선언](#)
- [Amazon SQS 정책 선언](#)
- [IAM 역할 템플릿의 예](#)

Important

IAM 리소스가 포함된 템플릿을 사용하여 스택을 생성 또는 업데이트하는 경우 IAM 기능의 사용을 승인해야 합니다. 자세한 내용은 [CloudFormation 템플릿에서 IAM 리소스 승인](#) 섹션을 참조하세요.

IAM 사용자 리소스 선언

이 코드 조각에서는 IAM 사용자를 생성하기 위해 [AWS::IAM::User](#) 리소스를 선언하는 방법을 보여줍니다. 사용자는 경로("/")와 암호가 포함된 로그인 프로파일(myP@ssW0rd)로 선언됩니다.

정책 문서 giveaccesstoqueueonly는 Amazon SQS 대기열 리소스 myqueue에 대해 모든 Amazon SQS 작업을 수행할 수 있는 사용자 권한을 부여하고 기타 모든 Amazon SQS 대기열 리소스에 대한 액세스를 거부합니다. Fn::GetAtt 함수는 [AWS::SQS::Queue](#) 리소스 myqueue의 Arn 속성을 가져옵니다.

Amazon SNS 주제 리소스 mytopic에 대해 모든 Amazon SNS 작업을 수행하고 기타 모든 Amazon SNS 리소스에 대한 액세스를 거부할 수 있는 사용자 권한을 부여하는 정책 문서 giveaccesstotopiconly가 사용자에게 추가됩니다. Ref 함수는 [AWS::SNS::Topic](#) 리소스 mytopic의 ARN을 가져옵니다.

JSON

```
"myuser" : {
  "Type" : "AWS::IAM::User",
  "Properties" : {
    "Path" : "/",
    "LoginProfile" : {
      "Password" : "myP@ssW0rd"
    },
    "Policies" : [ {
      "PolicyName" : "giveaccesstoqueueonly",
      "PolicyDocument" : {
        "Version": "2012-10-17",
        "Statement" : [ {
          "Effect" : "Allow",
          "Action" : [ "sqs:*" ],
          "Resource" : [ {
            "Fn::GetAtt" : [ "myqueue", "Arn" ]
          } ]
        }, {
          "Effect" : "Deny",
          "Action" : [ "sqs:*" ],
          "NotResource" : [ {
            "Fn::GetAtt" : [ "myqueue", "Arn" ]
          } ]
        }
      ]
    }, {
      "PolicyName" : "giveaccesstotopiconly",
      "PolicyDocument" : {
        "Version": "2012-10-17",
        "Statement" : [ {
          "Effect" : "Allow",
          "Action" : [ "sns:*" ],
          "Resource" : [ { "Ref" : "mytopic" } ]
        }, {
          "Effect" : "Deny",
          "Action" : [ "sns:*" ],
```

```

        "NotResource" : [ { "Ref" : "mytopic" } ]
      } ]
    } ]
  }
}

```

YAML

```

myuser:
  Type: AWS::IAM::User
  Properties:
    Path: "/"
    LoginProfile:
      Password: myP@ssW0rd
    Policies:
      - PolicyName: giveaccesstoqueueonly
        PolicyDocument:
          Version: '2012-10-17'
          Statement:
            - Effect: Allow
              Action:
                - sqs:*
              Resource:
                - !GetAtt myqueue.Arn
            - Effect: Deny
              Action:
                - sqs:*
              NotResource:
                - !GetAtt myqueue.Arn
      - PolicyName: giveaccesstotopiconly
        PolicyDocument:
          Version: '2012-10-17'
          Statement:
            - Effect: Allow
              Action:
                - sns:*
              Resource:
                - !Ref mytopic
            - Effect: Deny
              Action:
                - sns:*
              NotResource:

```

- !Ref mytopic

IAM 액세스 키 리소스 선언

이 코드 조각은 [AWS::IAM::AccessKey](#) 리소스를 보여줍니다. myaccesskey 리소스는 액세스 키를 생성하여 템플릿에 [AWS::IAM::User](#) 리소스로 선언된 IAM 사용자에게 할당합니다.

JSON

```
"myaccesskey" : {
  "Type" : "AWS::IAM::AccessKey",
  "Properties" : {
    "UserName" : { "Ref" : "myuser" }
  }
}
```

YAML

```
myaccesskey:
  Type: AWS::IAM::AccessKey
  Properties:
    UserName:
      !Ref myuser
```

`AWS::IAM::AccessKey` 함수를 사용하여 `Fn::GetAtt` 리소스에 대한 보안 키를 가져올 수 있습니다. 보안 키는 Output 값에 보안 키를 입력한 경우에만 검색할 수 있습니다. Ref 함수를 사용하면 액세스 키를 가져올 수 있습니다. 다음 Output 값 선언에서는 myaccesskey에 대한 액세스 키 및 보안 키를 가져옵니다.

JSON

```
"AccessKeyformyaccesskey" : {
  "Value" : { "Ref" : "myaccesskey" }
},
"SecretKeyformyaccesskey" : {
  "Value" : {
    "Fn::GetAtt" : [ "myaccesskey", "SecretAccessKey" ]
  }
}
```

```
}
```

YAML

```
AccessKeyformyaccesskey:
  Value:
    !Ref myaccesskey
SecretKeyformyaccesskey:
  Value: !GetAtt myaccesskey.SecretAccessKey
```

또한 템플릿에 정의된 Amazon EC2 인스턴스 또는 오토 스케일링에 AWS 액세스 키 및 보안 키를 전달할 수도 있습니다. 다음 [AWS::EC2::Instance](#) 선언에서는 UserData 속성을 사용하여 myaccesskey 리소스에 대한 액세스 키 및 보안 암호 키를 전달합니다.

JSON

```
"myinstance" : {
  "Type" : "AWS::EC2::Instance",
  "Properties" : {
    "AvailabilityZone" : "us-east-1a",
    "ImageId" : "ami-0ff8a91507f77f867",
    "UserData" : {
      "Fn::Base64" : {
        "Fn::Join" : [
          "", [
            "ACCESS_KEY=", {
              "Ref" : "myaccesskey"
            },
            "&",
            "SECRET_KEY=",
            {
              "Fn::GetAtt" : [
                "myaccesskey",
                "SecretAccessKey"
              ]
            }
          ]
        ]
      }
    }
  ]
}
```

```
}
```

YAML

```
myinstance:
  Type: AWS::EC2::Instance
  Properties:
    AvailabilityZone: "us-east-1a"
    ImageId: ami-0ff8a91507f77f867
    UserData:
      Fn::Base64: !Sub "ACCESS_KEY=${myaccesskey}&SECRET_KEY=
${myaccesskey.SecretAccessKey}"
```

IAM 그룹 리소스 선언

이 코드 조각은 [AWS::IAM::Group](#) 리소스를 보여줍니다. 그룹에는 경로("/myapplication/")가 있습니다. 그룹의 사용자가 Amazon SQS 대기열 리소스 myqueue에 대해 모든 Amazon SQS 작업을 수행하고 myqueue를 제외한 기타 모든 Amazon SQS 리소스에 대한 액세스를 거부하도록 허용하는 정책 문서 myappppolicy가 그룹에 추가됩니다.

리소스에 정책을 할당하려면 IAM에는 해당 리소스에 대한 Amazon 리소스 이름(ARN)이 필요합니다. 코드 조각에서 Fn::GetAtt 함수는 [AWS::SQS::Queue](#) 리소스 대기열의 ARN을 가져옵니다.

JSON

```
"mygroup" : {
  "Type" : "AWS::IAM::Group",
  "Properties" : {
    "Path" : "/myapplication/",
    "Policies" : [ {
      "PolicyName" : "myappppolicy",
      "PolicyDocument" : {
        "Version": "2012-10-17",
        "Statement" : [ {
          "Effect" : "Allow",
          "Action" : [ "sqs:*" ],
          "Resource" : [ {
            "Fn::GetAtt" : [ "myqueue", "Arn" ]
          } ]
        } ]
      },
      {
```

```

        "Effect" : "Deny",
        "Action" : [ "sqs:*" ],
        "NotResource" : [ { "Fn::GetAtt" : [ "myqueue", "Arn" ] } ]
    }
  ] }
} ]
}
}

```

YAML

```

mygroup:
  Type: AWS::IAM::Group
  Properties:
    Path: "/myapplication/"
    Policies:
      - PolicyName: myapppolicy
        PolicyDocument:
          Version: '2012-10-17'
          Statement:
            - Effect: Allow
              Action:
                - sqs:*
              Resource: !GetAtt myqueue.Arn
            - Effect: Deny
              Action:
                - sqs:*
              NotResource: !GetAtt myqueue.Arn

```

그룹에 사용자 추가

[AWS::IAM::UserToGroupAddition](#) 리소스는 그룹에 사용자를 추가합니다. 다음 코드 조각에서 `addUserToGroup` 리소스는 기존 사용자 `existinguser1` 및 템플릿에 [AWS::IAM::User](#) 리소스로 선언된 사용자 `myuser`를 기존 그룹 `myexistinggroup2`에 추가합니다.

JSON

```

"addUserToGroup" : {
  "Type" : "AWS::IAM::UserToGroupAddition",
  "Properties" : {
    "GroupName" : "myexistinggroup2",
    "Users" : [ "existinguser1", { "Ref" : "myuser" } ]
  }
}

```

```
}
}
```

YAML

```
addUserToGroup:
  Type: AWS::IAM::UserToGroupAddition
  Properties:
    GroupName: myexistinggroup2
    Users:
      - existinguser1
      - !Ref myuser
```

IAM 정책 선언

이 코드 조각은 mypolicy라는 [AWS::IAM::Policy](#) 리소스를 사용하여 정책을 생성해 여러 그룹에 적용하는 방법을 보여줍니다. mypolicy 리소스에는 ARN PolicyDocument으로 표시되는 S3 버킷에 있는 객체에 대해 GetObject, PutObject 및 PutObjectAcl 작업을 허용하는 arn:aws:s3:::myAWSBucket 속성이 포함되어 있습니다. mypolicy 리소스는 기존 그룹 myexistinggroup1 및 템플릿에 [AWS::IAM::Group](#) 리소스로 선언된 그룹 mygroup에 정책을 적용합니다. 이 예에서는 Groups 속성을 사용하여 그룹에 정책을 적용하는 방법을 보여주지만 대신 Users 속성을 사용하여 사용자 목록에 정책 문서를 추가할 수도 있습니다.

JSON

```
"mypolicy" : {
  "Type" : "AWS::IAM::Policy",
  "Properties" : {
    "PolicyName" : "mygrouppolicy",
    "PolicyDocument" : {
      "Version": "2012-10-17",
      "Statement" : [ {
        "Effect" : "Allow",
        "Action" : [
          "s3:GetObject" , "s3:PutObject" , "s3:PutObjectAcl" ],
        "Resource" : "arn:aws:s3:::myAWSBucket/*"
      } ]
    },
    "Groups" : [ "myexistinggroup1", { "Ref" : "mygroup" } ]
  }
}
```


YAML

```

mypolicy:
  Type: AWS::IAM::Policy
  Properties:
    PolicyName: mygrouppolicy
    PolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Effect: Allow
          Action:
            - s3:GetObject
            - s3:PutObject
            - s3:PutObjectAcl
          Resource: arn:aws:s3:::myAWSBucket/*
    Groups:
      - myexistinggroup1
      - !Ref mygroup

```

Amazon S3 버킷 정책 선언

이 코드 조각은 [AWS::S3::BucketPolicy](#) 리소스를 사용하여 정책을 생성해 Amazon S3 버킷에 적용하는 방법을 보여줍니다. mybucketpolicy 리소스는 user1 IAM 사용자가 정책이 적용되는 S3 버킷의 모든 객체에 대해 GetObject 작업을 수행하도록 허용하는 정책 문서를 선언합니다. 코드 조각에서 Fn::GetAtt 함수는 user1 리소스의 ARN을 가져옵니다. mybucketpolicy 리소스는 AWS::S3::BucketPolicy 리소스 mybucket에 정책을 적용합니다. Ref 함수는 mybucket 리소스의 버킷 이름을 가져옵니다.

JSON

```

"mybucketpolicy" : {
  "Type" : "AWS::S3::BucketPolicy",
  "Properties" : {
    "PolicyDocument" : {
      "Id" : "MyPolicy",
      "Version": "2012-10-17",
      "Statement" : [ {
        "Sid" : "ReadAccess",
        "Action" : [ "s3:GetObject" ],
        "Effect" : "Allow",
        "Resource" : { "Fn::Join" : [
          "", [ "arn:aws:s3:::", { "Ref" : "mybucket" } , "/*" ]

```

```

        ] },
        "Principal" : {
            "AWS" : { "Fn::GetAtt" : [ "user1", "Arn" ] }
        }
    } ]
},
"Bucket" : { "Ref" : "mybucket" }
}
}

```

YAML

```

mybucketpolicy:
  Type: AWS::S3::BucketPolicy
  Properties:
    PolicyDocument:
      Id: MyPolicy
      Version: '2012-10-17'
      Statement:
        - Sid: ReadAccess
          Action:
            - s3:GetObject
          Effect: Allow
          Resource: !Sub "arn:aws:s3:::${mybucket}/*"
          Principal:
            AWS: !GetAtt user1.Arn
      Bucket: !Ref mybucket

```

Amazon SNS 주제 정책 선언

이 코드 조각은 [AWS::SNS::TopicPolicy](#) 리소스를 사용하여 정책을 생성해 Amazon SNS 주제에 적용하는 방법을 보여줍니다. mysnspolicy 리소스에는 [AWS::IAM::User](#) 리소스 myuser가 [AWS::SNS::Topic](#) 리소스 mytopic에서 Publish 작업을 수행하도록 허용하는 PolicyDocument 속성이 포함되어 있습니다. 이 코드 조각에서 Fn::GetAtt 함수는 myuser 리소스의 ARN을 가져오고, Ref 함수는 mytopic 리소스의 ARN을 가져옵니다.

JSON

```

"mysnspolicy" : {
  "Type" : "AWS::SNS::TopicPolicy",
  "Properties" : {

```

```

    "PolicyDocument" : {
      "Id" : "MyTopicPolicy",
      "Version": "2012-10-17",
      "Statement" : [ {
        "Sid" : "My-statement-id",
        "Effect" : "Allow",
        "Principal" : {
          "AWS" : { "Fn::GetAtt" : [ "myuser", "Arn" ] }
        },
        "Action" : "sns:Publish",
        "Resource" : "*"
      } ]
    },
    "Topics" : [ { "Ref" : "mytopic" } ]
  }
}

```

YAML

```

mysnspolicy:
  Type: AWS::SNS::TopicPolicy
  Properties:
    PolicyDocument:
      Id: MyTopicPolicy
      Version: '2012-10-17'
      Statement:
        - Sid: My-statement-id
          Effect: Allow
          Principal:
            AWS: !GetAtt myuser.Arn
          Action: sns:Publish
          Resource: "*"
    Topics:
      - !Ref mytopic

```

Amazon SQS 정책 선언

이 코드 조각은 [AWS::SQS::QueuePolicy](#) 리소스를 사용하여 정책을 생성해 Amazon SQS 대기열에 적용하는 방법을 보여줍니다. PolicyDocument 속성은 기존 사용자 myapp(ARN으로 지정됨)이 기존 대기열(해당 URL로 지정됨) 및 [AWS::SQS::Queue](#) 리소스 myqueue에서 SendMessage 작업을 수행하도록 허용합니다. [Ref](#) 함수는 myqueue 리소스 URL을 가져옵니다.

JSON

```

"mysqspolicy" : {
  "Type" : "AWS::SQS::QueuePolicy",
  "Properties" : {
    "PolicyDocument" : {
      "Id" : "MyQueuePolicy",
      "Version": "2012-10-17",
      "Statement" : [ {
        "Sid" : "Allow-User-SendMessage",
        "Effect" : "Allow",
        "Principal" : {
          "AWS" : "arn:aws:iam::123456789012:user/myapp"
        },
        "Action" : [ "sqs:SendMessage" ],
        "Resource" : "*"
      } ]
    },
    "Queues" : [
      "https://sqs.us-east-2aws-region.amazonaws.com/123456789012/myexistingqueue",
      { "Ref" : "myqueue" }
    ]
  }
}

```

YAML

```

mysqspolicy:
  Type: AWS::SQS::QueuePolicy
  Properties:
    PolicyDocument:
      Id: MyQueuePolicy
      Version: '2012-10-17'
      Statement:
        - Sid: Allow-User-SendMessage
          Effect: Allow
          Principal:
            AWS: arn:aws:iam::123456789012:user/myapp
          Action:
            - sqs:SendMessage
          Resource: "*"
    Queues:
      - https://sqs.aws-region.amazonaws.com/123456789012/myexistingqueue

```

- !Ref myqueue

IAM 역할 템플릿의 예

이 섹션에서는 EC2 인스턴스에 대한 IAM 역할의 CloudFormation 템플릿 예제를 제공합니다.

자세한 내용은 Amazon EC2 사용 설명서의 [Amazon EC2의 IAM 역할](#)을 참조하세요.

EC2 관련 IAM 역할

이 예제에서 인스턴스 프로파일은 EC2 인스턴스의 `IamInstanceProfile` 속성에서 참조합니다. 인스턴스 정책 및 역할 정책 둘 다에서 [AWS::IAM::Role](#)을 참조합니다.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Resources": {
    "myEC2Instance": {
      "Type": "AWS::EC2::Instance",
      "Version": "2009-05-15",
      "Properties": {
        "ImageId": "ami-0ff8a91507f77f867",
        "InstanceType": "m1.small",
        "Monitoring": "true",
        "DisableApiTermination": "false",
        "IamInstanceProfile": {
          "Ref": "RootInstanceProfile"
        }
      }
    },
    "RootRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [ {
            "Effect": "Allow",
            "Principal": {
              "Service": [ "ec2.amazonaws.com" ]
            },
            "Action": [ "sts:AssumeRole" ]
          } ]
        }
      }
    }
  }
}
```

```

        },
        "Path": "/"
    }
},
"RolePolicies": {
    "Type": "AWS::IAM::Policy",
    "Properties": {
        "PolicyName": "root",
        "PolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [ {
                "Effect": "Allow",
                "Action": "*",
                "Resource": "*"
            } ]
        },
        "Roles": [ { "Ref": "RootRole" } ]
    }
},
"RootInstanceProfile": {
    "Type": "AWS::IAM::InstanceProfile",
    "Properties": {
        "Path": "/",
        "Roles": [ { "Ref": "RootRole" } ]
    }
}
}
}
}

```

YAML

```

AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myEC2Instance:
    Type: AWS::EC2::Instance
    Version: '2009-05-15'
    Properties:
      ImageId: ami-0ff8a91507f77f867
      InstanceType: m1.small
      Monitoring: 'true'
      DisableApiTermination: 'false'
      IamInstanceProfile:
        !Ref RootInstanceProfile

```

```

RootRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Effect: Allow
          Principal:
            Service:
              - ec2.amazonaws.com
          Action:
            - sts:AssumeRole
      Path: "/"
    RolePolicies:
      Type: AWS::IAM::Policy
      Properties:
        PolicyName: root
        PolicyDocument:
          Version: '2012-10-17'
          Statement:
            - Effect: Allow
              Action: "*"
              Resource: "*"
        Roles:
          - !Ref RootRole
    RootInstanceProfile:
      Type: AWS::IAM::InstanceProfile
      Properties:
        Path: "/"
        Roles:
          - !Ref RootRole

```

Auto Scaling 그룹 관련 IAM 역할

이 예제에서 인스턴스 프로파일은 Amazon EC2 Auto Scaling 시작 구성의 `IamInstanceProfile` 속성에서 참조합니다.

JSON

```

{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Resources": {
    "myLCOne": {

```

```

    "Type": "AWS::AutoScaling::LaunchConfiguration",
    "Version": "2009-05-15",
    "Properties": {
        "ImageId": "ami-0ff8a91507f77f867",
        "InstanceType": "m1.small",
        "InstanceMonitoring": "true",
        "IamInstanceProfile": { "Ref": "RootInstanceProfile" }
    }
},
"myASGrpOne": {
    "Type": "AWS::AutoScaling::AutoScalingGroup",
    "Version": "2009-05-15",
    "Properties": {
        "AvailabilityZones": [ "us-east-1a" ],
        "LaunchConfigurationName": { "Ref": "myLCOne" },
        "MinSize": "0",
        "MaxSize": "0",
        "HealthCheckType": "EC2",
        "HealthCheckGracePeriod": "120"
    }
},
"RootRole": {
    "Type": "AWS::IAM::Role",
    "Properties": {
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [ {
                "Effect": "Allow",
                "Principal": {
                    "Service": [ "ec2.amazonaws.com" ]
                },
                "Action": [ "sts:AssumeRole" ]
            } ]
        },
        "Path": "/"
    }
},
"RolePolicies": {
    "Type": "AWS::IAM::Policy",
    "Properties": {
        "PolicyName": "root",
        "PolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [ {

```



```

        "Effect": "Allow",
        "Action": "*",
        "Resource": "*"
    } ]
},
"Roles": [ { "Ref": "RootRole" } ]
}
},
"RootInstanceProfile": {
    "Type": "AWS::IAM::InstanceProfile",
    "Properties": {
        "Path": "/",
        "Roles": [ { "Ref": "RootRole" } ]
    }
}
}
}
}

```

YAML

```

AWSTemplateFormatVersion: '2010-09-09'
Resources:
  myLCOne:
    Type: AWS::AutoScaling::LaunchConfiguration
    Version: '2009-05-15'
    Properties:
      ImageId: ami-0ff8a91507f77f867
      InstanceType: m1.small
      InstanceMonitoring: 'true'
      IamInstanceProfile:
        !Ref RootInstanceProfile
  myASGrpOne:
    Type: AWS::AutoScaling::AutoScalingGroup
    Version: '2009-05-15'
    Properties:
      AvailabilityZones:
        - "us-east-1a"
      LaunchConfigurationName:
        !Ref myLCOne
      MinSize: '0'
      MaxSize: '0'
      HealthCheckType: EC2
      HealthCheckGracePeriod: '120'

```

```

RootRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Effect: Allow
          Principal:
            Service:
              - ec2.amazonaws.com
          Action:
            - sts:AssumeRole
      Path: "/"
    RolePolicies:
      Type: AWS::IAM::Policy
      Properties:
        PolicyName: root
        PolicyDocument:
          Version: '2012-10-17'
          Statement:
            - Effect: Allow
              Action: "*"
              Resource: "*"
        Roles:
          - !Ref RootRole
    RootInstanceProfile:
      Type: AWS::IAM::InstanceProfile
      Properties:
        Path: "/"
        Roles:
          - !Ref RootRole

```

AWS Lambda 템플릿

다음 템플릿에서는 AWS Lambda(Lambda) 함수와 사용자 지정 리소스를 사용하여 기존 보안 그룹 목록에 새 보안 그룹을 추가합니다. 이 함수는 목록에 새 보안 그룹과 기존 보안 그룹이 모두 포함되도록 보안 그룹의 목록을 동적으로 빌드하려는 경우에 유용합니다. 예를 들어, 기존 보안 그룹의 목록을 파라미터 값으로 전달하고, 새 값을 목록에 추가한 다음 모든 값을 EC2 인스턴스에 연결할 수 있습니다. Lambda 함수 리소스 유형에 대한 자세한 내용은 [AWS::Lambda::Function](#)을 참조하세요.

이 예제에서 CloudFormation은 AllSecurityGroups 사용자 지정 리소스를 생성할 때 AppendItemToListFunction Lambda 함수를 간접적으로 호출합니다. CloudFormation은 기존 보

안 그룹 목록과 새 보안 그룹(NewSecurityGroup)을 함수에 전달하고, 해당 함수는 새 보안 그룹을 목록에 추가한 다음 수정된 목록을 반환합니다. CloudFormation은 수정된 목록을 사용하여 모든 보안 그룹을 MyEC2Instance 리소스와 연결합니다.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "ExistingSecurityGroups": {
      "Type": "List<AWS::EC2::SecurityGroup::Id>"
    },
    "ExistingVPC": {
      "Type": "AWS::EC2::VPC::Id",
      "Description": "The VPC ID that includes the security groups in the ExistingSecurityGroups parameter."
    },
    "InstanceType": {
      "Type": "String",
      "Default": "t2.micro",
      "AllowedValues": [
        "t2.micro",
        "t3.micro"
      ]
    }
  },
  "Resources": {
    "SecurityGroup": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "GroupDescription": "Allow HTTP traffic to the host",
        "VpcId": {
          "Ref": "ExistingVPC"
        }
      },
      "SecurityGroupIngress": [
        {
          "IpProtocol": "tcp",
          "FromPort": 80,
          "ToPort": 80,
          "CidrIp": "0.0.0.0/0"
        }
      ]
    }
  },
}
```

```

        "SecurityGroupEgress": [
            {
                "IpProtocol": "tcp",
                "FromPort": 80,
                "ToPort": 80,
                "CidrIp": "0.0.0.0/0"
            }
        ]
    },
    "AllSecurityGroups": {
        "Type": "Custom::Split",
        "Properties": {
            "ServiceToken": {
                "Fn::GetAtt": [
                    "AppendItemToListFunction",
                    "Arn"
                ]
            },
            "List": {
                "Ref": "ExistingSecurityGroups"
            },
            "AppendedItem": {
                "Ref": "SecurityGroup"
            }
        }
    },
    "AppendItemToListFunction": {
        "Type": "AWS::Lambda::Function",
        "Properties": {
            "Handler": "index.handler",
            "Role": {
                "Fn::GetAtt": [
                    "LambdaExecutionRole",
                    "Arn"
                ]
            },
            "Code": {
                "ZipFile": {
                    "Fn::Join": [
                        "",
                        [
                            "var response = require('cfn-response');",
                            "exports.handler = function(event, context) {"

```

```

        "    var responseData = {Value:
event.ResourceProperties.List}};",
        "
responseData.Value.push(event.ResourceProperties.AppendedItem);",
        "    response.send(event, context, response.SUCCESS,
responseData);",
        "};"
    ]
}
},
"Runtime": "nodejs20.x"
}
},
"MyEC2Instance": {
    "Type": "AWS::EC2::Instance",
    "Properties": {
        "ImageId": "{{resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-
ami-hvm-x86_64-gp2}}",
        "SecurityGroupIds": {
            "Fn::GetAtt": [
                "AllSecurityGroups",
                "Value"
            ]
        },
        "InstanceType": {
            "Ref": "InstanceType"
        }
    }
},
"LambdaExecutionRole": {
    "Type": "AWS::IAM::Role",
    "Properties": {
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "Service": [
                            "lambda.amazonaws.com"
                        ]
                    },
                    "Action": [

```

```

        "sts:AssumeRole"
      ]
    }
  ]
},
"Path": "/",
"Policies": [
  {
    "PolicyName": "root",
    "PolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": [
            "logs:*"
          ],
          "Resource": "arn:aws:logs*:*:*"
        }
      ]
    }
  }
]
}
}
}
},
"Outputs": {
  "AllSecurityGroups": {
    "Description": "Security Groups that are associated with the EC2 instance",
    "Value": {
      "Fn::Join": [
        ", ",
        {
          "Fn::GetAtt": [
            "AllSecurityGroups",
            "Value"
          ]
        }
      ]
    }
  }
}
}
}
}

```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Parameters:
  ExistingSecurityGroups:
    Type: List<AWS::EC2::SecurityGroup::Id>
  ExistingVPC:
    Type: AWS::EC2::VPC::Id
    Description: The VPC ID that includes the security groups in the
ExistingSecurityGroups parameter.
  InstanceType:
    Type: String
    Default: t2.micro
    AllowedValues:
      - t2.micro
      - t3.micro
Resources:
  SecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Allow HTTP traffic to the host
      VpcId: !Ref ExistingVPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
      SecurityGroupEgress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
  AllSecurityGroups:
    Type: Custom::Split
    Properties:
      ServiceToken: !GetAtt AppendItemToListFunction.Arn
      List: !Ref ExistingSecurityGroups
      AppendedItem: !Ref SecurityGroup
  AppendItemToListFunction:
    Type: AWS::Lambda::Function
    Properties:
      Handler: index.handler
      Role: !GetAtt LambdaExecutionRole.Arn
```

```

Code:
  ZipFile: !Join
    - ''
    - - var response = require('cfn-response');
      - exports.handler = function(event, context) {
      - '   var responseData = {Value: event.ResourceProperties.List};'
      - '   responseData.Value.push(event.ResourceProperties.AppendedItem);'
      - '   response.send(event, context, response.SUCCESS, responseData);'
      - '};'

  Runtime: nodejs20.x

MyEC2Instance:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: '{{resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2}}'
    SecurityGroupIds: !GetAtt AllSecurityGroups.Value
    InstanceType: !Ref InstanceType

LambdaExecutionRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Effect: Allow
          Principal:
            Service:
              - lambda.amazonaws.com
          Action:
            - sts:AssumeRole

    Path: /

  Policies:
    - PolicyName: root
      PolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Action:
              - logs:*
            Resource: arn:aws:logs:*:*:*

Outputs:
  AllSecurityGroups:
    Description: Security Groups that are associated with the EC2 instance
    Value: !Join
      - ', '

```


- !GetAtt AllSecurityGroups.Value

Amazon Redshift 템플릿 코드 조각

Amazon Redshift는 클라우드에서 완전히 관리되는 페타바이트급 데이터 웨어하우스 서비스입니다. CloudFormation을 사용하여 Amazon Redshift 클러스터를 프로비저닝하고 관리할 수 있습니다.

Amazon Redshift 클러스터

다음 샘플 템플릿은 스택 생성 시 지정된 파라미터 값에 따라 Amazon Redshift 클러스터를 생성합니다. Amazon Redshift 클러스터와 연결된 클러스터 파라미터 그룹은 사용자 활동 로깅을 가능하게 합니다. 또한 이 템플릿은 템플릿에 정의된 Amazon VPC에서 Amazon Redshift 클러스터를 시작합니다. VPC에는 인터넷 게이트웨이가 포함되어 있어 인터넷에서 Amazon Redshift 클러스터에 액세스할 수 있습니다. 그러나 클러스터와 인터넷 게이트웨이 간에 통신이 활성화되어 있어야 하며, 통신은 라우팅 테이블 항목을 통해 활성화합니다.

Note

템플릿에는 IsMultiNodeCluster 파라미터 값이 NumberOfNodes로 설정된 경우에만 ClusterType 파라미터가 선언되는 multi-node 조건이 포함되어 있습니다.

이 예에서는 MySQLRootPassword 속성이 NoEcho로 설정된 true 파라미터를 정의합니다. NoEcho 속성을 true로 설정한 경우, 아래 지정된 위치에 저장된 정보를 제외하고 CloudFormation은 스택 또는 스택 이벤트를 설명하는 모든 호출에 대해 별표(****)로 마스킹 처리된 파라미터 값을 반환합니다.

Important

NoEcho 속성을 사용해도 다음에 저장된 정보는 마스킹되지 않습니다.

- Metadata 템플릿 섹션. CloudFormation은 Metadata 섹션에 포함된 정보를 변환, 수정 또는 삭제하지 않습니다. 자세한 내용은 [메타데이터](#)를 참조하십시오.
- Outputs 템플릿 섹션. 자세한 내용은 [출력](#)을 참조하십시오.
- 리소스 정의의 Metadata 속성입니다. 자세한 내용은 [Metadata 속성](#)을 참조하세요.

이러한 메커니즘을 사용하여 암호나 보안 정보와 같은 중요한 정보를 포함하지 않는 것이 좋습니다.

Important

AWS Systems Manager Parameter Store 또는 AWS Secrets Manager와 같이 CloudFormation 외부에서 저장 및 관리되는 중요한 정보를 참조하려면 CloudFormation 템플릿에 직접 중요한 정보를 포함하는 대신 스택 템플릿에 있는 동적 파라미터를 사용하는 것이 좋습니다.

자세한 내용은 [템플릿에 자격 증명을 포함하지 않음](#)을 참조하십시오.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters" : {
    "DatabaseName" : {
      "Description" : "The name of the first database to be created when the cluster is
created",
      "Type" : "String",
      "Default" : "dev",
      "AllowedPattern" : "([a-z]|[0-9])+"
    },
    "ClusterType" : {
      "Description" : "The type of cluster",
      "Type" : "String",
      "Default" : "single-node",
      "AllowedValues" : [ "single-node", "multi-node" ]
    },
    "NumberOfNodes" : {
      "Description" : "The number of compute nodes in the cluster. For multi-node
clusters, the NumberOfNodes parameter must be greater than 1",
      "Type" : "Number",
      "Default" : "1"
    },
    "NodeType" : {
      "Description" : "The type of node to be provisioned",
      "Type" : "String",
```

```
"Default" : "ds2.xlarge",
    "AllowedValues" : [ "ds2.xlarge", "ds2.8xlarge", "dc1.large", "dc1.8xlarge" ]
},
"MasterUsername" : {
    "Description" : "The user name that is associated with the master user account
for the cluster that is being created",
    "Type" : "String",
    "Default" : "defaultuser",
    "AllowedPattern" : "([a-z])([a-z]|[0-9])*"
},
"MasterUserPassword" : {
    "Description" : "The password that is associated with the master user account for
the cluster that is being created.",
    "Type" : "String",
    "NoEcho" : "true"
},
"InboundTraffic" : {
    "Description" : "Allow inbound traffic to the cluster from this CIDR range.",
    "Type" : "String",
    "MinLength": "9",
    "MaxLength": "18",
    "Default" : "0.0.0.0/0",
    "AllowedPattern" : "(\\d{1,3})\\.((\\d{1,3})|\\.((\\d{1,3})|\\.((\\d{1,3})|\\.(\n\\d{1,2})))",
    "ConstraintDescription" : "must be a valid CIDR range of the form x.x.x.x/x."
},
"PortNumber" : {
    "Description" : "The port number on which the cluster accepts incoming
connections.",
    "Type" : "Number",
    "Default" : "5439"
}
},
"Conditions" : {
    "IsMultiNodeCluster" : {
        "Fn::Equals" : [{ "Ref" : "ClusterType" }, "multi-node" ]
    }
},
"Resources" : {
    "RedshiftCluster" : {
        "Type" : "AWS::Redshift::Cluster",
        "DependsOn" : "AttachGateway",
        "Properties" : {
            "ClusterType" : { "Ref" : "ClusterType" },
```

```

    "NumberOfNodes" : { "Fn::If" : [ "IsMultiNodeCluster", { "Ref" :
"NumberOfNodes" }, { "Ref" : "AWS::NoValue" } ] },
    "NodeType" : { "Ref" : "NodeType" },
    "DBName" : { "Ref" : "DatabaseName" },
    "MasterUsername" : { "Ref" : "MasterUsername" },
    "MasterUserPassword" : { "Ref" : "MasterUserPassword" },
    "ClusterParameterGroupName" : { "Ref" : "RedshiftClusterParameterGroup" },
    "VpcSecurityGroupIds" : [ { "Ref" : "SecurityGroup" } ],
    "ClusterSubnetGroupName" : { "Ref" : "RedshiftClusterSubnetGroup" },
    "PubliclyAccessible" : "true",
    "Port" : { "Ref" : "PortNumber" }
  }
},
"RedshiftClusterParameterGroup" : {
  "Type" : "AWS::Redshift::ClusterParameterGroup",
  "Properties" : {
    "Description" : "Cluster parameter group",
    "ParameterGroupFamily" : "redshift-1.0",
    "Parameters" : [{
      "ParameterName" : "enable_user_activity_logging",
      "ParameterValue" : "true"
    }]
  }
},
"RedshiftClusterSubnetGroup" : {
  "Type" : "AWS::Redshift::ClusterSubnetGroup",
  "Properties" : {
    "Description" : "Cluster subnet group",
    "SubnetIds" : [ { "Ref" : "PublicSubnet" } ]
  }
},
"VPC" : {
  "Type" : "AWS::EC2::VPC",
  "Properties" : {
    "CidrBlock" : "10.0.0.0/16"
  }
},
"PublicSubnet" : {
  "Type" : "AWS::EC2::Subnet",
  "Properties" : {
    "CidrBlock" : "10.0.0.0/24",
    "VpcId" : { "Ref" : "VPC" }
  }
},

```

```

"SecurityGroup" : {
  "Type" : "AWS::EC2::SecurityGroup",
  "Properties" : {
    "GroupDescription" : "Security group",
    "SecurityGroupIngress" : [ {
      "CidrIp" : { "Ref": "InboundTraffic" },
      "FromPort" : { "Ref" : "PortNumber" },
      "ToPort" : { "Ref" : "PortNumber" },
      "IpProtocol" : "tcp"
    } ],
    "VpcId" : { "Ref" : "VPC" }
  }
},
"myInternetGateway" : {
  "Type" : "AWS::EC2::InternetGateway"
},
"AttachGateway" : {
  "Type" : "AWS::EC2::VPCGatewayAttachment",
  "Properties" : {
    "VpcId" : { "Ref" : "VPC" },
    "InternetGatewayId" : { "Ref" : "myInternetGateway" }
  }
},
"PublicRouteTable" : {
  "Type" : "AWS::EC2::RouteTable",
  "Properties" : {
    "VpcId" : {
      "Ref" : "VPC"
    }
  }
},
"PublicRoute" : {
  "Type" : "AWS::EC2::Route",
  "DependsOn" : "AttachGateway",
  "Properties" : {
    "RouteTableId" : {
      "Ref" : "PublicRouteTable"
    },
    "DestinationCidrBlock" : "0.0.0.0/0",
    "GatewayId" : {
      "Ref" : "myInternetGateway"
    }
  }
},

```

```

    "PublicSubnetRouteTableAssociation" : {
      "Type" : "AWS::EC2::SubnetRouteTableAssociation",
      "Properties" : {
        "SubnetId" : {
          "Ref" : "PublicSubnet"
        },
        "RouteTableId" : {
          "Ref" : "PublicRouteTable"
        }
      }
    },
  },
  "Outputs" : {
    "ClusterEndpoint" : {
      "Description" : "Cluster endpoint",
      "Value" : { "Fn::Join" : [ ":", [ { "Fn::GetAtt" : [ "RedshiftCluster",
"Endpoint.Address" ] }, { "Fn::GetAtt" : [ "RedshiftCluster",
"Endpoint.Port" ] } ] ] }
    },
    "ClusterName" : {
      "Description" : "Name of cluster",
      "Value" : { "Ref" : "RedshiftCluster" }
    },
    "ParameterGroupName" : {
      "Description" : "Name of parameter group",
      "Value" : { "Ref" : "RedshiftClusterParameterGroup" }
    },
    "RedshiftClusterSubnetGroupName" : {
      "Description" : "Name of cluster subnet group",
      "Value" : { "Ref" : "RedshiftClusterSubnetGroup" }
    },
    "RedshiftClusterSecurityGroupName" : {
      "Description" : "Name of cluster security group",
      "Value" : { "Ref" : "SecurityGroup" }
    }
  }
}

```

YAML

```

AWSTemplateFormatVersion: '2010-09-09'
Parameters:
  DatabaseName:

```

```

    Description: The name of the first database to be created when the cluster is
        created
    Type: String
    Default: dev
    AllowedPattern: "([a-z]|[0-9])+"
ClusterType:
    Description: The type of cluster
    Type: String
    Default: single-node
    AllowedValues:
        - single-node
        - multi-node
NumberOfNodes:
    Description: The number of compute nodes in the cluster. For multi-node clusters,
        the NumberOfNodes parameter must be greater than 1
    Type: Number
    Default: '1'
NodeType:
    Description: The type of node to be provisioned
    Type: String
    Default: ds2.xlarge
    AllowedValues:
        - ds2.xlarge
        - ds2.8xlarge
        - dc1.large
        - dc1.8xlarge
MasterUsername:
    Description: The user name that is associated with the master user account for
        the cluster that is being created
    Type: String
    Default: defaultuser
    AllowedPattern: "([a-z])([a-z]|[0-9])*"
MasterUserPassword:
    Description: The password that is associated with the master user account for
        the cluster that is being created.
    Type: String
    NoEcho: 'true'
InboundTraffic:
    Description: Allow inbound traffic to the cluster from this CIDR range.
    Type: String
    MinLength: '9'
    MaxLength: '18'
    Default: 0.0.0.0/0
    AllowedPattern: "(\\d{1,3})\\.\\.\\.\\.\\. (\\d{1,3})\\.\\.\\.\\.\\. (\\d{1,3})\\.\\.\\.\\.\\. (\\d{1,3})/ (\\d{1,2})"

```

```
    ConstraintDescription: must be a valid CIDR range of the form x.x.x.x/x.
  PortNumber:
    Description: The port number on which the cluster accepts incoming connections.
    Type: Number
    Default: '5439'
  Conditions:
    IsMultiNodeCluster:
      Fn::Equals:
        - Ref: ClusterType
        - multi-node
  Resources:
    RedshiftCluster:
      Type: AWS::Redshift::Cluster
      DependsOn: AttachGateway
      Properties:
        ClusterType:
          Ref: ClusterType
        NumberOfNodes:
          Fn::If:
            - IsMultiNodeCluster
            - Ref: NumberOfNodes
            - Ref: AWS::NoValue
        NodeType:
          Ref: NodeType
        DBName:
          Ref: DatabaseName
        MasterUsername:
          Ref: MasterUsername
        MasterUserPassword:
          Ref: MasterUserPassword
        ClusterParameterGroupName:
          Ref: RedshiftClusterParameterGroup
        VpcSecurityGroupIds:
          - Ref: SecurityGroup
        ClusterSubnetGroupName:
          Ref: RedshiftClusterSubnetGroup
        PubliclyAccessible: 'true'
        Port:
          Ref: PortNumber
    RedshiftClusterParameterGroup:
      Type: AWS::Redshift::ClusterParameterGroup
      Properties:
        Description: Cluster parameter group
        ParameterGroupFamily: redshift-1.0
```



```
Parameters:
- ParameterName: enable_user_activity_logging
  ParameterValue: 'true'
RedshiftClusterSubnetGroup:
Type: AWS::Redshift::ClusterSubnetGroup
Properties:
  Description: Cluster subnet group
  SubnetIds:
  - Ref: PublicSubnet
VPC:
Type: AWS::EC2::VPC
Properties:
  CidrBlock: 10.0.0.0/16
PublicSubnet:
Type: AWS::EC2::Subnet
Properties:
  CidrBlock: 10.0.0.0/24
  VpcId:
    Ref: VPC
SecurityGroup:
Type: AWS::EC2::SecurityGroup
Properties:
  GroupDescription: Security group
  SecurityGroupIngress:
  - CidrIp:
      Ref: InboundTraffic
    FromPort:
      Ref: PortNumber
    ToPort:
      Ref: PortNumber
    IpProtocol: tcp
  VpcId:
    Ref: VPC
myInternetGateway:
Type: AWS::EC2::InternetGateway
AttachGateway:
Type: AWS::EC2::VPCGatewayAttachment
Properties:
  VpcId:
    Ref: VPC
  InternetGatewayId:
    Ref: myInternetGateway
PublicRouteTable:
Type: AWS::EC2::RouteTable
```

```

Properties:
  VpcId:
    Ref: VPC
PublicRoute:
  Type: AWS::EC2::Route
  DependsOn: AttachGateway
  Properties:
    RouteTableId:
      Ref: PublicRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    GatewayId:
      Ref: myInternetGateway
PublicSubnetRouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
  Properties:
    SubnetId:
      Ref: PublicSubnet
    RouteTableId:
      Ref: PublicRouteTable
Outputs:
  ClusterEndpoint:
    Description: Cluster endpoint
    Value: !Sub "${RedshiftCluster.Endpoint.Address}:${RedshiftCluster.Endpoint.Port}"
  ClusterName:
    Description: Name of cluster
    Value:
      Ref: RedshiftCluster
  ParameterGroupName:
    Description: Name of parameter group
    Value:
      Ref: RedshiftClusterParameterGroup
  RedshiftClusterSubnetGroupName:
    Description: Name of cluster subnet group
    Value:
      Ref: RedshiftClusterSubnetGroup
  RedshiftClusterSecurityGroupName:
    Description: Name of cluster security group
    Value:
      Ref: SecurityGroup

```

다음 사항도 참조하세요.

[AWS::Redshift::Cluster](#)

Amazon RDS 템플릿 코드 조각

주제

- [Amazon RDS DB 인스턴스 리소스](#)
- [Amazon RDS Oracle Database DB 인스턴스 리소스](#)
- [CIDR 범위에 대한 Amazon RDS DBSecurityGroup 리소스](#)
- [Amazon EC2 보안 그룹을 포함하는 Amazon RDS DBSecurityGroup](#)
- [여러 VPC 보안 그룹](#)
- [VPC 보안 그룹 내 Amazon RDS 데이터베이스 인스턴스](#)

Amazon RDS DB 인스턴스 리소스

이 예에서는 관리형 마스터 사용자 암호를 사용하는 Amazon RDS DB 인스턴스 리소스를 보여 줍니다. 자세한 내용은 Amazon RDS 사용 설명서의 [AWS Secrets Manager를 통한 암호 관리](#) 및 Aurora 사용 설명서의 [AWS Secrets Manager를 통한 암호 관리](#)를 참조하세요. 선택 사항인 EngineVersion 속성을 지정하지 않은 경우 기본 엔진 버전이 이 DB 인스턴스에 사용됩니다. 기본 엔진 버전과 기타 기본 설정에 대한 자세한 내용은 [CreateDBInstance](#)를 참조하십시오. DBSecurityGroups 속성은 MyDbSecurityByEC2SecurityGroup과 MyDbSecurityByCIDRIPGroup이라는 AWS::RDS::DBSecurityGroup 리소스에 대한 네트워크 수신을 승인합니다. 자세한 내용은 섹션을 참조하세요 [AWS::RDS::DBInstance](#) 또한 DB 인스턴스 리소스에서는 DeletionPolicy 속성을 Snapshot으로 설정합니다. Snapshot DeletionPolicy 세트를 사용하여 CloudFormation에서는 스택 삭제 중 이 DB 인스턴스를 삭제하기 전에 스냅샷을 생성합니다.

JSON

```
"MyDB" : {
  "Type" : "AWS::RDS::DBInstance",
  "Properties" : {
    "DBSecurityGroups" : [
      {"Ref" : "MyDbSecurityByEC2SecurityGroup"}, {"Ref" :
"MyDbSecurityByCIDRIPGroup"} ],
    "AllocatedStorage" : "5",
    "DBInstanceClass" : "db.t2.small",
    "Engine" : "MySQL",
    "MasterUsername" : "MyName",
    "ManageMasterUserPassword" : true,
    "MasterUserSecret" : {
```

```

        "KmsKeyId" : {"Ref" : "KMSKey"}
    },
    "DeletionPolicy" : "Snapshot"
}

```

YAML

```

MyDB:
  Type: AWS::RDS::DBInstance
  Properties:
    DBSecurityGroups:
      - Ref: MyDbSecurityByEC2SecurityGroup
      - Ref: MyDbSecurityByCIDRIPGroup
    AllocatedStorage: '5'
    DBInstanceClass: db.t2.small
    Engine: MySQL
    MasterUsername: MyName
    ManageMasterUserPassword: true
    MasterUserSecret:
      KmsKeyId: !Ref KMSKey
    DeletionPolicy: Snapshot

```

Amazon RDS Oracle Database DB 인스턴스 리소스

이 예에서는 관리형 마스터 사용자 암호를 사용하는 Oracle Database DB 인스턴스 리소스를 생성합니다. 자세한 내용은 Amazon RDS 사용 설명서의 [AWS Secrets Manager를 통한 암호 관리](#)를 참조하세요. 이 예제는 Engine을 oracle-ee로 지정하고 라이선스 모델로 기존 보유 라이선스 사용을 사용합니다. Oracle Database DB 인스턴스 설정에 대한 자세한 내용은 [CreateDBInstance](#)를 참조하십시오. DBSecurityGroups 속성은 MyDbSecurityByEC2SecurityGroup 및 MyDbSecurityByCIDRIPGroup이라는 AWS::RDS::DBSecurityGroup 리소스에 네트워크 진입을 허가합니다. 자세한 내용은 섹션을 참조하세요. [AWS::RDS::DBInstance](#) 또한 DB 인스턴스 리소스에서는 DeletionPolicy 속성을 Snapshot으로 설정합니다. Snapshot DeletionPolicy 세트를 사용하여 CloudFormation에서는 스택 삭제 중 이 DB 인스턴스를 삭제하기 전에 스냅샷을 생성합니다.

JSON

```

"MyDB" : {
  "Type" : "AWS::RDS::DBInstance",
  "Properties" : {
    "DBSecurityGroups" : [

```

```

        {"Ref" : "MyDbSecurityByEC2SecurityGroup"}, {"Ref" :
"MyDbSecurityByCIDRIPGroup"} ],
        "AllocatedStorage" : "5",
        "DBInstanceClass" : "db.t2.small",
        "Engine" : "oracle-ee",
        "LicenseModel" : "bring-your-own-license",
        "MasterUsername" : "master",
        "ManageMasterUserPassword" : true,
        "MasterUserSecret" : {
            "KmsKeyId" : {"Ref" : "KMSKey"}
        }
    },
    "DeletionPolicy" : "Snapshot"
}

```

YAML

```

MyDB:
  Type: AWS::RDS::DBInstance
  Properties:
    DBSecurityGroups:
      - Ref: MyDbSecurityByEC2SecurityGroup
      - Ref: MyDbSecurityByCIDRIPGroup
    AllocatedStorage: '5'
    DBInstanceClass: db.t2.small
    Engine: oracle-ee
    LicenseModel: bring-your-own-license
    MasterUsername: master
    ManageMasterUserPassword: true
    MasterUserSecret:
      KmsKeyId: !Ref KMSKey
    DeletionPolicy: Snapshot

```

CIDR 범위에 대한 Amazon RDS DBSecurityGroup 리소스

이 예제는 ddd.ddd.ddd.ddd/dd 형식으로 지정된 CIDR 범위에 대한 수신 권한이 부여된 Amazon RDS DBSecurityGroup 리소스를 보여줍니다. 자세한 내용은 [AWS::RDS::DBSecurityGroup](#) 및 [Ingress](#)를 참조하십시오.

JSON

```

"MyDbSecurityByCIDRIPGroup" : {
  "Type" : "AWS::RDS::DBSecurityGroup",

```

```

"Properties" : {
  "GroupDescription" : "Ingress for CIDRIP",
  "DBSecurityGroupIngress" : {
    "CIDRIP" : "192.168.0.0/32"
  }
}
}

```

YAML

```

MyDbSecurityByCIDRIPGroup:
  Type: AWS::RDS::DBSecurityGroup
  Properties:
    GroupDescription: Ingress for CIDRIP
    DBSecurityGroupIngress:
      CIDRIP: "192.168.0.0/32"

```

Amazon EC2 보안 그룹을 포함하는 Amazon RDS DBSecurityGroup

이 예제는 MyEc2SecurityGroup에서 참조되는 Amazon EC2 보안 그룹으로부터 수신 권한 부여를 받은 [AWS::RDS::DBSecurityGroup](#) 리소스를 보여줍니다.

이렇게 하려면 EC2 보안 그룹을 정의한 다음 내장 Ref 함수를 사용하여 DBSecurityGroup 내에서 EC2 보안 그룹을 참조합니다.

JSON

```

"DBInstance" : {
  "Type": "AWS::RDS::DBInstance",
  "Properties": {
    "DBName"          : { "Ref" : "DBName" },
    "Engine"          : "MySQL",
    "MasterUsername"  : { "Ref" : "DBUsername" },
    "DBInstanceClass" : { "Ref" : "DBClass" },
    "DBSecurityGroups" : [ { "Ref" : "DBSecurityGroup" } ],
    "AllocatedStorage" : { "Ref" : "DBAllocatedStorage" },
    "MasterUserPassword": { "Ref" : "DBPassword" }
  }
},

"DBSecurityGroup": {
  "Type": "AWS::RDS::DBSecurityGroup",

```

```

"Properties": {
  "DBSecurityGroupIngress": {
    "EC2SecurityGroupName": {
      "Fn::GetAtt": ["WebServerSecurityGroup", "GroupName"]
    }
  },
  "GroupDescription" : "Frontend Access"
},
},

"WebServerSecurityGroup" : {
  "Type" : "AWS::EC2::SecurityGroup",
  "Properties" : {
    "GroupDescription" : "Enable HTTP access via port 80 and SSH access",
    "SecurityGroupIngress" : [
      {"IpProtocol" : "tcp", "FromPort" : 80, "ToPort" : 80, "CidrIp" :
"0.0.0.0/0"},
      {"IpProtocol" : "tcp", "FromPort" : 22, "ToPort" : 22, "CidrIp" : "0.0.0.0/0"}
    ]
  }
}
}

```

YAML

이 예는 다음 전체 예제에서 추출되었습니다. [Drupal_Single_Instance_With_RDS.template](#)

```

DBInstance:
  Type: AWS::RDS::DBInstance
  Properties:
    DBName:
      Ref: DBName
    Engine: MySQL
    MasterUsername:
      Ref: DBUsername
    DBInstanceClass:
      Ref: DBClass
    DBSecurityGroups:
      - Ref: DBSecurityGroup
    AllocatedStorage:
      Ref: DBAllocatedStorage
    MasterUserPassword:
      Ref: DBPassword
  DBSecurityGroup:
    Type: AWS::RDS::DBSecurityGroup

```

```

Properties:
  DBSecurityGroupIngress:
    EC2SecurityGroupName:
      Ref: WebServerSecurityGroup
    GroupDescription: Frontend Access
WebServerSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Enable HTTP access via port 80 and SSH access
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 80
        ToPort: 80
        CidrIp: 0.0.0.0/0
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: 0.0.0.0/0

```

여러 VPC 보안 그룹

이 예제에서는 [AWS::RDS::DBSecurityGroupIngress](#)에서 여러 Amazon EC2 VPC 보안 그룹에 대한 수신 권한 부여를 받은 [AWS::RDS::DBSecurityGroup](#) 리소스를 보여줍니다.

JSON

```

{
  "Resources" : {
    "DBInstance" : {
      "Type" : "AWS::RDS::DBInstance",
      "Properties" : {
        "AllocatedStorage" : "5",
        "DBInstanceClass" : "db.t2.small",
        "DBName" : { "Ref": "MyDBName" },
        "DBSecurityGroups" : [ { "Ref" : "DbSecurityByEC2SecurityGroup" } ],
        "DBSubnetGroupName" : { "Ref" : "MyDBSubnetGroup" },
        "Engine" : "MySQL",
        "MasterUserPassword": { "Ref" : "MyDBPassword" },
        "MasterUsername" : { "Ref" : "MyDBUsername" }
      },
      "DeletionPolicy" : "Snapshot"
    },
    "DbSecurityByEC2SecurityGroup" : {

```



```

    "Type" : "AWS::RDS::DBSecurityGroup",
    "Properties" : {
      "GroupDescription" : "Ingress for Amazon EC2 security group",
      "EC2VpcId" : { "Ref" : "MyVPC" },
      "DBSecurityGroupIngress" : [ {
        "EC2SecurityGroupId" : "sg-b0ff1111",
        "EC2SecurityGroupOwnerId" : "111122223333"
      }, {
        "EC2SecurityGroupId" : "sg-ffd72222",
        "EC2SecurityGroupOwnerId" : "111122223333"
      } ]
    }
  }
}

```

YAML

```

Resources:
  DBInstance:
    Type: AWS::RDS::DBInstance
    Properties:
      AllocatedStorage: '5'
      DBInstanceClass: db.t2.small
      DBName:
        Ref: MyDBName
      DBSecurityGroups:
        - Ref: DbSecurityByEC2SecurityGroup
      DBSubnetGroupName:
        Ref: MyDBSubnetGroup
      Engine: MySQL
      MasterUserPassword:
        Ref: MyDBPassword
      MasterUsername:
        Ref: MyDBUsername
      DeletionPolicy: Snapshot
  DbSecurityByEC2SecurityGroup:
    Type: AWS::RDS::DBSecurityGroup
    Properties:
      GroupDescription: Ingress for Amazon EC2 security group
      EC2VpcId:
        Ref: MyVPC
      DBSecurityGroupIngress:

```

- EC2SecurityGroupId: sg-b0ff1111
EC2SecurityGroupOwnerId: '111122223333'
- EC2SecurityGroupId: sg-ffd72222
EC2SecurityGroupOwnerId: '111122223333'

VPC 보안 그룹 내 Amazon RDS 데이터베이스 인스턴스

이 예에서는 Amazon EC2 VPC 보안 그룹과 연결된 Amazon RDS 데이터베이스 인스턴스를 보여줍니다.

JSON

```
{
  "DBEC2SecurityGroup": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
      "GroupDescription": "Open database for access",
      "SecurityGroupIngress": [{
        "IpProtocol": "tcp",
        "FromPort": 3306,
        "ToPort": 3306,
        "SourceSecurityGroupName": { "Ref": "WebServerSecurityGroup" }
      }]
    }
  },
  "DBInstance": {
    "Type": "AWS::RDS::DBInstance",
    "Properties": {
      "DBName": { "Ref": "DBName" },
      "Engine": "MySQL",
      "MultiAZ": { "Ref": "MultiAZDatabase" },
      "MasterUsername": { "Ref": "DBUser" },
      "DBInstanceClass": { "Ref": "DBClass" },
      "AllocatedStorage": { "Ref": "DBAllocatedStorage" },
      "MasterUserPassword": { "Ref": "DBPassword" },
      "VPCSecurityGroups": [ { "Fn::GetAtt": [ "DBEC2SecurityGroup", "GroupId" ] } ]
    }
  }
}
```

YAML

```
DBEC2SecurityGroup:
```

```

Type: AWS::EC2::SecurityGroup
Properties:
  GroupDescription: Open database for access
  SecurityGroupIngress:
    - IpProtocol: tcp
      FromPort: 3306
      ToPort: 3306
      SourceSecurityGroupName:
        Ref: WebServerSecurityGroup
DBInstance:
  Type: AWS::RDS::DBInstance
  Properties:
    DBName:
      Ref: DBName
    Engine: MySQL
    MultiAZ:
      Ref: MultiAZDatabase
    MasterUsername:
      Ref: DBUser
    DBInstanceClass:
      Ref: DBClass
    AllocatedStorage:
      Ref: DBAllocatedStorage
    MasterUserPassword:
      Ref: DBPassword
    VPCSecurityGroups:
      - !GetAtt DBEC2SecurityGroup.GroupId

```

Route 53 템플릿 코드 조각

주제

- [호스팅 영역 이름 또는 ID를 사용하는 Amazon Route 53 리소스 레코드 세트](#)
- [RecordSetGroup을 사용하여 가중치 기반 리소스 레코드 세트 설정](#)
- [RecordSetGroup을 사용하여 별칭 리소스 레코드 세트 설정](#)
- [CloudFront 배포에 대한 별칭 리소스 레코드 세트](#)

호스팅 영역 이름 또는 ID를 사용하는 Amazon Route 53 리소스 레코드 세트

Amazon Route 53 리소스 레코드 세트 생성 시 이러한 레코드 세트를 추가할 호스팅 영역을 지정해야 합니다. CloudFormation은 호스팅 영역을 지정하는 두 가지 방법을 제공합니다.

- HostedZoneId 속성을 사용하여 호스팅 영역을 명시적으로 지정할 수 있습니다.
- HostedZoneName 속성을 사용하여 CloudFormation에서 호스팅 영역을 찾으려 할 수 있습니다. HostedZoneName 속성을 사용하며 이름이 동일한 여러 호스팅 영역이 있는 경우 CloudFormation은 스택을 생성하지 않습니다.

HostedZoneId를 사용하여 RecordSet 추가

이 예제는 HostedZoneId 속성을 사용하여 호스팅 영역을 지정하는 도메인 이름 `mysite.example.com`에 대한 SPF 레코드가 들어 있는 Amazon Route 53 리소스 레코드 세트를 추가합니다.

JSON

```
"myDNSRecord" : {
  "Type" : "AWS::Route53::RecordSet",
  "Properties" :
  {
    "HostedZoneId" : "Z3DG6IL3SJCGPX",
    "Name" : "mysite.example.com.",
    "Type" : "SPF",
    "TTL" : "900",
    "ResourceRecords" : [ "\"v=spf1 ip4:192.168.0.1/16 -all\"" ]
  }
}
```

YAML

```
myDNSRecord:
  Type: AWS::Route53::RecordSet
  Properties:
    HostedZoneId: Z3DG6IL3SJCGPX
    Name: mysite.example.com.
    Type: SPF
    TTL: '900'
    ResourceRecords:
      - '"v=spf1 ip4:192.168.0.1/16 -all"'
```

HostedZoneName를 사용하여 RecordSet 추가

이 예제에서는 HostedZoneName 속성을 사용하여 호스팅 영역을 지정하는 도메인 이름 `"mysite.example.com"`에 대한 Amazon Route 53 리소스 레코드 세트를 추가합니다.

JSON

```
"myDNSRecord2" : {
    "Type" : "AWS::Route53::RecordSet",
    "Properties" : {
        "HostedZoneName" : "example.com.",
        "Name" : "mysite.example.com.",
        "Type" : "A",
        "TTL" : "900",
        "ResourceRecords" : [
            "192.168.0.1",
            "192.168.0.2"
        ]
    }
}
```

YAML

```
myDNSRecord2:
  Type: AWS::Route53::RecordSet
  Properties:
    HostedZoneName: example.com.
    Name: mysite.example.com.
    Type: A
    TTL: '900'
    ResourceRecords:
      - 192.168.0.1
      - 192.168.0.2
```

RecordSetGroup을 사용하여 가중치 기반 리소스 레코드 세트 설정

이 예제에서는 [AWS::Route53::RecordSetGroup](#)을 사용하여 'example.com'에 대한 2개의 CNAME 레코드를 설정합니다. 호스팅 영역입니다. RecordSets 속성에는 "mysite.example.com" DNS 이름의 CNAME 레코드 세트가 들어 있습니다. 각 레코드 세트에는 식별자(SetIdentifier)와 가중치(Weight)가 포함되어 있습니다. 리소스로 라우팅되는 인터넷 트래픽의 비율은 다음 계산을 기반으로 합니다.

- Frontend One: $140 / (140 + 60) = 140 / 200 = 70\%$
- Frontend Two: $60 / (140 + 60) = 60 / 200 = 30\%$

가중치 기반 리소스 레코드 세트에 대한 자세한 내용을 알아보려면 Amazon Route 53 개발자 안내서의 [가중치 기반 라우팅](#)을 참조하세요.

JSON

```
"myDNSOne" : {
  "Type" : "AWS::Route53::RecordSetGroup",
  "Properties" : {
    "HostedZoneName" : "example.com.",
    "Comment" : "Weighted RR for my frontends.",
    "RecordSets" : [
      {
        "Name" : "mysite.example.com.",
        "Type" : "CNAME",
        "TTL" : "900",
        "SetIdentifier" : "Frontend One",
        "Weight" : "140",
        "ResourceRecords" : ["example-ec2.amazonaws.com"]
      },
      {
        "Name" : "mysite.example.com.",
        "Type" : "CNAME",
        "TTL" : "900",
        "SetIdentifier" : "Frontend Two",
        "Weight" : "60",
        "ResourceRecords" : ["example-ec2-larger.amazonaws.com"]
      }
    ]
  }
}
```

YAML

```
myDNSOne:
  Type: AWS::Route53::RecordSetGroup
  Properties:
    HostedZoneName: example.com.
    Comment: Weighted RR for my frontends.
    RecordSets:
      - Name: mysite.example.com.
        Type: CNAME
        TTL: '900'
        SetIdentifier: Frontend One
```

```

    Weight: '140'
    ResourceRecords:
      - example-ec2.amazonaws.com
  - Name: mysite.example.com.
    Type: CNAME
    TTL: '900'
    SetIdentifier: Frontend Two
    Weight: '60'
    ResourceRecords:
      - example-ec2-larger.amazonaws.com

```

RecordSetGroup을 사용하여 별칭 리소스 레코드 세트 설정

다음 예제에서는 [AWS::Route53::RecordSetGroup](#)을 사용하여 ELB 버전 1(Classic) 로드 밸런서 및 버전 2(애플리케이션 또는 네트워크) 로드 밸런서로 트래픽을 라우팅하는 example.com이라는 별칭 리소스 레코드 세트를 설정합니다. [AliasTarget](#) 속성은 GetAtt 내장 함수를 사용하여 myELB LoadBalancer의 호스팅 영역 ID와 DNS 이름을 지정합니다. GetAtt는 트래픽을 버전 1 또는 버전 2 중에서 어떤 로드 밸런서로 라우팅하는지에 따라 myELB 리소스의 다른 속성을 검색합니다.

- 버전 1 로드 밸런서: CanonicalHostedZoneNameID 및 DNSName
- 버전 2 로드 밸런서: CanonicalHostedZoneID 및 DNSName

별칭 리소스 레코드 세트에 대한 자세한 내용을 알아보려면 Route 53 개발자 안내서의 [별칭 및 비별칭 레코드 선택](#)을 참조하세요.

버전 1 로드 밸런서용 JSON

```

"myELB" : {
  "Type" : "AWS::ElasticLoadBalancing::LoadBalancer",
  "Properties" : {
    "AvailabilityZones" : [ "us-east-1a" ],
    "Listeners" : [ {
      "LoadBalancerPort" : "80",
      "InstancePort" : "80",
      "Protocol" : "HTTP"
    } ]
  }
},
"myDNS" : {
  "Type" : "AWS::Route53::RecordSetGroup",
  "Properties" : {

```

```

    "HostedZoneName" : "example.com.",
    "Comment" : "Zone apex alias targeted to myELB LoadBalancer.",
    "RecordSets" : [
      {
        "Name" : "example.com.",
        "Type" : "A",
        "AliasTarget" : {
          "HostedZoneId" : { "Fn::GetAtt" : ["myELB",
"CanonicalHostedZoneNameID"] },
          "DNSName" : { "Fn::GetAtt" : ["myELB","DNSName"] }
        }
      }
    ]
  }
}

```

버전 1 로드 밸런서용 YAML

```

myELB:
  Type: AWS::ElasticLoadBalancing::LoadBalancer
  Properties:
    AvailabilityZones:
      - "us-east-1a"
    Listeners:
      - LoadBalancerPort: '80'
        InstancePort: '80'
        Protocol: HTTP
myDNS:
  Type: AWS::Route53::RecordSetGroup
  Properties:
    HostedZoneName: example.com.
    Comment: Zone apex alias targeted to myELB LoadBalancer.
    RecordSets:
      - Name: example.com.
        Type: A
        AliasTarget:
          HostedZoneId: !GetAtt 'myELB.CanonicalHostedZoneNameID'
          DNSName: !GetAtt 'myELB.DNSName'

```

버전 2 로드 밸런서용 JSON

```

"myELB" : {
  "Type" : "AWS::ElasticLoadBalancing::LoadBalancer",

```



```

    "Properties" : {
      "Subnets" : [
        {"Ref": "SubnetAZ1"},
        {"Ref" : "SubnetAZ2"}
      ]
    }
  },
  "myDNS" : {
    "Type" : "AWS::Route53::RecordSetGroup",
    "Properties" : {
      "HostedZoneName" : "example.com.",
      "Comment" : "Zone apex alias targeted to myELB LoadBalancer.",
      "RecordSets" : [
        {
          "Name" : "example.com.",
          "Type" : "A",
          "AliasTarget" : {
            "HostedZoneId" : { "Fn::GetAtt" : ["myELB",
"CanonicalHostedZoneID"] },
            "DNSName" : { "Fn::GetAtt" : ["myELB","DNSName"] }
          }
        }
      ]
    }
  }
}

```

버전 2 로드 밸런서용 YAML

```

myELB:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
    Subnets:
      - Ref: SubnetAZ1
      - Ref: SubnetAZ2
myDNS:
  Type: AWS::Route53::RecordSetGroup
  Properties:
    HostedZoneName: example.com.
    Comment: Zone apex alias targeted to myELB LoadBalancer.
    RecordSets:
      - Name: example.com.
        Type: A
        AliasTarget:

```

```
HostedZoneId: !GetAtt 'myELB.CanonicalHostedZoneID'
DNSName: !GetAtt 'myELB.DNSName'
```

CloudFront 배포에 대한 별칭 리소스 레코드 세트

다음 예에서는 사용자 지정 도메인 이름을 기존 CloudFront 배포에 매핑하는 별칭 A 레코드를 생성합니다. myHostedZoneId는 동일한 템플릿의 실제 AWS::Route53::HostedZone 리소스 또는 파라미터에 대한 참조로 간주됩니다. myCloudFrontDistribution는 동일한 템플릿 내의 AWS::CloudFront::Distribution 리소스를 참조합니다. 별칭 레코드는 표준 CloudFront 호스팅 영역 ID(Z2FDTNDATAQYW2)를 사용하며, Fn::GetAtt를 사용하여 배포의 도메인 이름을 자동으로 확인합니다. 이 설정을 사용하면 IP 주소 없이 웹 트래픽을 사용자 지정 도메인에서 CloudFront 배포로 라우팅할 수 있습니다.

Note

별칭 리소스 레코드 세트를 생성할 때 Z2FDTNDATAQYW2 속성을 HostedZoneId로 지정해야 합니다. 프라이빗 영역에서는 CloudFront에 대한 별칭 리소스 레코드 세트를 생성할 수 없습니다.

JSON

```
{
  "myDNS": {
    "Type": "AWS::Route53::RecordSetGroup",
    "Properties": {
      "HostedZoneId": {
        "Ref": "myHostedZoneID"
      },
      "RecordSets": [
        {
          "Name": {
            "Ref": "myRecordSetDomainName"
          },
          "Type": "A",
          "AliasTarget": {
            "HostedZoneId": "Z2FDTNDATAQYW2",
            "DNSName": {
              "Fn::GetAtt": [
                "myCloudFrontDistribution",
                "DomainName"
              ]
            }
          }
        }
      ]
    }
  }
}
```

```

    ],
    },
    "EvaluateTargetHealth": false
  }
}
]
}
}
}

```

YAML

```

myDNS:
  Type: AWS::Route53::RecordSetGroup
  Properties:
    HostedZoneId: !Ref myHostedZoneID
    RecordSets:
      - Name: !Ref myRecordSetDomainName
        Type: A
        AliasTarget:
          HostedZoneId: Z2FDTNDATAQYW2
          DNSName: !GetAtt
            - myCloudFrontDistribution
            - DomainName
        EvaluateTargetHealth: false

```

Amazon S3 템플릿 코드 조각

이러한 Amazon S3 샘플 템플릿을 사용하면 CloudFormation에서 Amazon S3 버킷을 쉽게 설명할 수 있습니다. 더 많은 예는 `AWS::S3::Bucket` 리소스의 [예](#) 섹션을 참조하세요.

주제

- [기본값을 사용하여 Amazon S3 버킷 생성](#)
- [DeletionPolicy를 사용하여 웹 사이트 호스팅용 Amazon S3 버킷 생성](#)
- [사용자 지정 도메인을 사용하여 정적 웹 사이트 생성](#)

기본값을 사용하여 Amazon S3 버킷 생성

이 예제에서는 [AWS::S3::Bucket](#)을 사용하여 기본 설정으로 버킷을 생성합니다.

JSON

```
"myS3Bucket" : {
  "Type" : "AWS::S3::Bucket"
}
```

YAML

```
MyS3Bucket:
  Type: AWS::S3::Bucket
```

DeletionPolicy를 사용하여 웹 사이트 호스팅용 Amazon S3 버킷 생성

이 예제에서는 버킷을 웹 사이트로 만들고 퍼블릭 액세스 차단은 비활성화합니다(웹 사이트 호스팅용으로 설정된 버킷에는 퍼블릭 읽기 권한 필요). 이후 퍼블릭 버킷 정책이 버킷에 추가됩니다. 이 버킷 리소스에서 DeletionPolicy 속성이 Retain으로 설정되었으므로 CloudFormation에서는 스택을 삭제할 때 이 버킷을 삭제하지 않습니다. Output 섹션에서는 Fn::GetAtt를 사용하여 S3Bucket 리소스의 WebsiteURL 속성과 DomainName 속성을 검색합니다.

 Note

다음 예제에서는 퍼블릭 액세스 차단 설정 BlockPublicPolicy 및 RestrictPublicBuckets가 계정 수준에서 비활성화되었다고 가정합니다.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Resources": {
    "S3Bucket": {
      "Type": "AWS::S3::Bucket",
      "Properties": {
        "PublicAccessBlockConfiguration": {
          "BlockPublicAcls": false,
          "BlockPublicPolicy": false,
          "IgnorePublicAcls": false,
          "RestrictPublicBuckets": false
        },
        "WebsiteConfiguration": {
          "IndexDocument": "index.html",
```

```

        "ErrorDocument": "error.html"
    },
    "DeletionPolicy": "Retain",
    "UpdateReplacePolicy": "Retain"
},
"BucketPolicy": {
    "Type": "AWS::S3::BucketPolicy",
    "Properties": {
        "PolicyDocument": {
            "Id": "MyPolicy",
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Sid": "PublicReadForGetBucketObjects",
                    "Effect": "Allow",
                    "Principal": "*",
                    "Action": "s3:GetObject",
                    "Resource": {
                        "Fn::Join": [
                            "",
                            [
                                "arn:aws:s3:::",
                                {
                                    "Ref": "S3Bucket"
                                },
                                "/*"
                            ]
                        ]
                    }
                }
            ]
        }
    },
    "Bucket": {
        "Ref": "S3Bucket"
    }
}
},
"Outputs": {
    "WebsiteURL": {
        "Value": {
            "Fn::GetAtt": [
                "S3Bucket",

```

```

        "WebsiteURL"
      ],
    },
    "Description": "URL for website hosted on S3"
  },
  "S3BucketSecureURL": {
    "Value": {
      "Fn::Join": [
        "",
        [
          "https://",
          {
            "Fn::GetAtt": [
              "S3Bucket",
              "DomainName"
            ]
          }
        ]
      ]
    },
    "Description": "Name of S3 bucket to hold website content"
  }
}
}
}

```

YAML

```

AWSTemplateFormatVersion: 2010-09-09
Resources:
  S3Bucket:
    Type: AWS::S3::Bucket
    Properties:
      PublicAccessBlockConfiguration:
        BlockPublicAcls: false
        BlockPublicPolicy: false
        IgnorePublicAcls: false
        RestrictPublicBuckets: false
      WebsiteConfiguration:
        IndexDocument: index.html
        ErrorDocument: error.html
      DeletionPolicy: Retain
      UpdateReplacePolicy: Retain
      BucketPolicy:

```

```

Type: AWS::S3::BucketPolicy
Properties:
  PolicyDocument:
    Id: MyPolicy
    Version: 2012-10-17
    Statement:
      - Sid: PublicReadForGetBucketObjects
        Effect: Allow
        Principal: '*'
        Action: 's3:GetObject'
        Resource: !Join
          - ''
          - - 'arn:aws:s3:::'
            - !Ref S3Bucket
            - /*
          - Bucket: !Ref S3Bucket
Outputs:
  WebsiteURL:
    Value: !GetAtt
      - S3Bucket
      - WebsiteURL
    Description: URL for website hosted on S3
  S3BucketSecureURL:
    Value: !Join
      - ''
      - - 'https://'
        - !GetAtt
          - S3Bucket
          - DomainName
    Description: Name of S3 bucket to hold website content

```

사용자 지정 도메인을 사용하여 정적 웹 사이트 생성

등록된 도메인과 함께 Route 53을 사용할 수 있습니다. 다음 샘플에서는 도메인에 대해 Route 53에서 호스팅 영역을 이미 생성했다고 가정합니다. 이 예제는 웹 사이트 호스팅용 버킷 두 개를 생성합니다. 루트 버킷은 콘텐츠를 호스팅하고 다른 버킷은 `www.domainname.com` 요청을 루트 버킷으로 리디렉션합니다. 레코드 세트는 도메인 이름을 Amazon S3 엔드포인트로 매핑합니다.

위 예에 표시된 대로 버킷 정책도 추가해야 합니다.

사용자 지정 도메인 사용에 대한 자세한 내용을 알아보려면 Amazon Simple Storage Service 사용 설명서의 [자습서: Route 53에 등록된 사용자 지정 도메인을 사용하여 정적 웹 사이트 구성](#)을 참조하세요.

Note

다음 예제에서는 퍼블릭 액세스 차단 설정 BlockPublicPolicy 및 RestrictPublicBuckets가 계정 수준에서 비활성화되었다고 가정합니다.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Mappings" : {
    "RegionMap" : {
      "us-east-1" : { "S3hostedzoneID" : "Z3AQBSTGFYJSTF", "websiteendpoint" :
"s3-website-us-east-1.amazonaws.com" },
      "us-west-1" : { "S3hostedzoneID" : "Z2F56UZL2M1ACD", "websiteendpoint" :
"s3-website-us-west-1.amazonaws.com" },
      "us-west-2" : { "S3hostedzoneID" : "Z3BJ6K6RIION7M", "websiteendpoint" :
"s3-website-us-west-2.amazonaws.com" },
      "eu-west-1" : { "S3hostedzoneID" : "Z1BKCTXD74EZPE", "websiteendpoint" :
"s3-website-eu-west-1.amazonaws.com" },
      "ap-southeast-1" : { "S3hostedzoneID" : "Z300J2DXBE1FTB",
"websiteendpoint" : "s3-website-ap-southeast-1.amazonaws.com" },
      "ap-southeast-2" : { "S3hostedzoneID" : "Z1WCIGYICN2BYD",
"websiteendpoint" : "s3-website-ap-southeast-2.amazonaws.com" },
      "ap-northeast-1" : { "S3hostedzoneID" : "Z2M4EHUR26P7ZW",
"websiteendpoint" : "s3-website-ap-northeast-1.amazonaws.com" },
      "sa-east-1" : { "S3hostedzoneID" : "Z31GFT0UA1I2HV", "websiteendpoint" :
"s3-website-sa-east-1.amazonaws.com" }
    }
  },
  "Parameters": {
    "RootDomainName": {
      "Description": "Domain name for your website (example.com)",
      "Type": "String"
    }
  },
  "Resources": {
    "RootBucket": {
      "Type": "AWS::S3::Bucket",
      "Properties": {
        "BucketName" : {"Ref":"RootDomainName"},
        "PublicAccessBlockConfiguration": {
          "BlockPublicAcls": false,
```



```

        "BlockPublicPolicy": false,
        "IgnorePublicAcls": false,
        "RestrictPublicBuckets": false
    },
    "WebsiteConfiguration": {
        "IndexDocument": "index.html",
        "ErrorDocument": "404.html"
    }
},
"WWWBucket": {
    "Type": "AWS::S3::Bucket",
    "Properties": {
        "BucketName": {
            "Fn::Join": ["", ["www.", {"Ref": "RootDomainName"}]]
        },
        "AccessControl": "BucketOwnerFullControl",
        "WebsiteConfiguration": {
            "RedirectAllRequestsTo": {
                "HostName": {"Ref": "RootBucket"}
            }
        }
    }
},
"myDNS": {
    "Type": "AWS::Route53::RecordSetGroup",
    "Properties": {
        "HostedZoneName": {
            "Fn::Join": ["", [{"Ref": "RootDomainName"}, "."]]
        },
        "Comment": "Zone apex alias.",
        "RecordSets": [
            {
                "Name": {"Ref": "RootDomainName"},
                "Type": "A",
                "AliasTarget": {
                    "HostedZoneId": {"Fn::FindInMap" : [ "RegionMap", { "Ref" :
"AWS::Region" }, "S3hostedzoneID"]},
                    "DNSName": {"Fn::FindInMap" : [ "RegionMap", { "Ref" :
"AWS::Region" }, "websiteendpoint"]}
                }
            },
            {
                "Name": {

```

```

        "Fn::Join": [ "", [ "www.", { "Ref": "RootDomainName" } ] ]
    },
    "Type": "CNAME",
    "TTL" : "900",
    "ResourceRecords" : [
        { "Fn::GetAtt": [ "WWWBucket", "DomainName" ] }
    ]
    }
    ]
    }
    },
    "Outputs": {
        "WebsiteURL": {
            "Value": { "Fn::GetAtt": [ "RootBucket", "WebsiteURL" ] },
            "Description": "URL for website hosted on S3"
        }
    }
}

```

YAML

```

Parameters:
  RootDomainName:
    Description: Domain name for your website (example.com)
    Type: String
Mappings:
  RegionMap:
    us-east-1:
      S3hostedzoneID: Z3AQBSTGFYJSTF
      websiteendpoint: s3-website-us-east-1.amazonaws.com
    us-west-1:
      S3hostedzoneID: Z2F56UZL2M1ACD
      websiteendpoint: s3-website-us-west-1.amazonaws.com
    us-west-2:
      S3hostedzoneID: Z3BJ6K6RIION7M
      websiteendpoint: s3-website-us-west-2.amazonaws.com
    eu-west-1:
      S3hostedzoneID: Z1BKCTXD74EZPE
      websiteendpoint: s3-website-eu-west-1.amazonaws.com
    ap-southeast-1:
      S3hostedzoneID: Z300J2DXBE1FTB
      websiteendpoint: s3-website-ap-southeast-1.amazonaws.com

```

```

ap-southeast-2:
  S3hostedzoneID: Z1WCIGYICN2BYD
  websiteendpoint: s3-website-ap-southeast-2.amazonaws.com
ap-northeast-1:
  S3hostedzoneID: Z2M4EHUR26P7ZW
  websiteendpoint: s3-website-ap-northeast-1.amazonaws.com
sa-east-1:
  S3hostedzoneID: Z31GFT0UA1I2HV
  websiteendpoint: s3-website-sa-east-1.amazonaws.com

```

Resources:

RootBucket:

Type: AWS::S3::Bucket

Properties:

```

BucketName: !Ref RootDomainName
PublicAccessBlockConfiguration:
  BlockPublicAcls: false
  BlockPublicPolicy: false
  IgnorePublicAcls: false
  RestrictPublicBuckets: false
WebsiteConfiguration:
  IndexDocument: index.html
  ErrorDocument: 404.html

```

WWWBucket:

Type: AWS::S3::Bucket

Properties:

```

BucketName: !Sub
  - www.${Domain}
  - Domain: !Ref RootDomainName
AccessControl: BucketOwnerFullControl
WebsiteConfiguration:
  RedirectAllRequestsTo:
    HostName: !Ref RootBucket

```

myDNS:

Type: AWS::Route53::RecordSetGroup

Properties:

```

HostedZoneName: !Sub
  - ${Domain}.
  - Domain: !Ref RootDomainName
Comment: Zone apex alias.

```

RecordSets:

```

- Name: !Ref RootDomainName
  Type: A
  AliasTarget:
    HostedZoneId: !FindInMap [ RegionMap, !Ref 'AWS::Region', S3hostedzoneID]

```

```

        DNSName: !FindInMap [ RegionMap, !Ref 'AWS::Region', websiteendpoint]
    - Name: !Sub
      - www.${Domain}
      - Domain: !Ref RootDomainName
    Type: CNAME
    TTL: 900
    ResourceRecords:
      - !GetAtt WWWBucket.DomainName

Outputs:
  WebsiteURL:
    Value: !GetAtt RootBucket.WebsiteURL
    Description: URL for website hosted on S3

```

Amazon SNS 템플릿 코드 조각

이 예에서는 Amazon SNS 주제 리소스를 보여줍니다. 여기에는 유효한 이메일 주소가 필요합니다.

JSON

```

"MySNSTopic" : {
  "Type" : "AWS::SNS::Topic",
  "Properties" : {
    "Subscription" : [ {
      "Endpoint" : "add valid email address",
      "Protocol" : "email"
    } ]
  }
}

```

YAML

```

MySNSTopic:
  Type: AWS::SNS::Topic
  Properties:
    Subscription:
      - Endpoint: "add valid email address"
        Protocol: email

```

Amazon SQS 템플릿 코드 조각

이 예제에서는 Amazon SQS 대기열을 보여줍니다.

JSON

```
"MyQueue" : {
  "Type" : "AWS::SQS::Queue",
  "Properties" : {
    "VisibilityTimeout" : "value"
  }
}
```

YAML

```
MyQueue:
  Type: AWS::SQS::Queue
  Properties:
    VisibilityTimeout: value
```

Amazon Timestream 템플릿 코드 조각

InfluxDB용 Amazon Timestream을 사용하면 애플리케이션 개발자와 DevOps 팀이 오픈 소스 API를 사용하여 AWS에서 실시간 시계열 애플리케이션을 위한 완전관리형 InfluxDB 데이터베이스를 쉽게 실행할 수 있습니다. 까다로운 시계열 워크로드를 처리하는 InfluxDB 데이터베이스를 빠르게 생성할 수 있습니다. 몇 번의 간단한 API 호출로 AWS에서 자동 소프트웨어 패치, 백업 및 복구를 통해 InfluxDB 데이터베이스의 설정, 마이그레이션, 운영 및 확장을 수행할 수 있습니다. GitHub의 [awslabs/amazon-timestream-tools/tree/mainline/integrations/cloudformation/timestream-influxdb](https://github.com/aws-labs/amazon-timestream-tools/tree/mainline/integrations/cloudformation/timestream-influxdb)에서도 이러한 샘플을 찾을 수 있습니다.

주제

- [기본값을 사용하는 최소 샘플](#)
- [파라미터가 포함된 보다 완전한 예제](#)

이러한 CloudFormation 템플릿은 InfluxDB용 Amazon Timestream 인스턴스를 성공적으로 생성, 연결 및 모니터링하는 데 필요한 다음과 같은 리소스를 생성합니다.

Amazon VPC

- VPC
- 하나 이상의 Subnet
- InternetGateway

- RouteTable
- SecurityGroup

Amazon S3

- Bucket

Amazon Timestream

- InfluxDBInstance

기본값을 사용하는 최소 샘플

이 예제에서는 가능한 경우 기본값을 사용하여 다중 AZ 및 퍼블릭 액세스 가능 인스턴스를 배포합니다.

JSON

```
{
  "Metadata": {
    "AWS::CloudFormation::Interface": {
      "ParameterGroups": [
        {
          "Label": {"default": "Amazon Timestream for InfluxDB Configuration"},
          "Parameters": [
            "DbInstanceName",
            "InfluxDBPassword"
          ]
        }
      ],
      "ParameterLabels": {
        "VPCCIDR": {"default": "VPC CIDR"}
      }
    }
  },
  "Parameters": {
    "DbInstanceName": {
      "Description": "The name that uniquely identifies the DB instance when interacting with the Amazon Timestream for InfluxDB API and CLI commands. This name will also be a prefix included in the endpoint. DB instance names must be unique per customer and per Region.",
```

```

    "Type": "String",
    "Default": "mydbinstance",
    "MinLength": 3,
    "MaxLength": 40,
    "AllowedPattern": "^[a-zA-z][a-zA-Z0-9]*(-[a-zA-Z0-9]+)*$"
  },
  "InfluxDBPassword": {
    "Description": "The password of the initial admin user created in InfluxDB. This
password will allow you to access the InfluxDB UI to perform various administrative
tasks and also use the InfluxDB CLI to create an operator token. These attributes will
be stored in a Secret created in AWS Secrets Manager in your account.",
    "Type": "String",
    "NoEcho": true,
    "MinLength": 8,
    "MaxLength": 64,
    "AllowedPattern": "^[a-zA-Z0-9]+$"
  }
},
"Resources": {
  "VPC": {
    "Type": "AWS::EC2::VPC",
    "Properties": {"CidrBlock": "10.0.0.0/16"}
  },
  "InternetGateway": {"Type": "AWS::EC2::InternetGateway"},
  "InternetGatewayAttachment": {
    "Type": "AWS::EC2::VPCGatewayAttachment",
    "Properties": {
      "InternetGatewayId": {"Ref": "InternetGateway"},
      "VpcId": {"Ref": "VPC"}
    }
  },
  "Subnet1": {
    "Type": "AWS::EC2::Subnet",
    "Properties": {
      "VpcId": {"Ref": "VPC"},
      "AvailabilityZone": {
        "Fn::Select": [
          0,
          {"Fn::GetAZs": ""}
        ]
      }
    }
  },
  "CidrBlock": {
    "Fn::Select": [
      0,

```

```

        {
            "Fn::Cidr": [
                {
                    "Fn::GetAtt": [
                        "VPC",
                        "CidrBlock"
                    ]
                },
                2,
                12
            ]
        }
    ],
    "MapPublicIpOnLaunch": true
},
"Subnet2": {
    "Type": "AWS::EC2::Subnet",
    "Properties": {
        "VpcId": {"Ref": "VPC"},
        "AvailabilityZone": {
            "Fn::Select": [
                1,
                {"Fn::GetAZs": ""}
            ]
        },
        "CidrBlock": {
            "Fn::Select": [
                1,
                {
                    "Fn::Cidr": [
                        {
                            "Fn::GetAtt": [
                                "VPC",
                                "CidrBlock"
                            ]
                        },
                        2,
                        12
                    ]
                }
            ]
        }
    }
},

```



```

        "MapPublicIpOnLaunch": true
    }
},
"RouteTable": {
    "Type": "AWS::EC2::RouteTable",
    "Properties": {
        "VpcId": {"Ref": "VPC"}
    }
},
"DefaultRoute": {
    "Type": "AWS::EC2::Route",
    "DependsOn": "InternetGatewayAttachment",
    "Properties": {
        "RouteTableId": {"Ref": "RouteTable"},
        "DestinationCidrBlock": "0.0.0.0/0",
        "GatewayId": {"Ref": "InternetGateway"}
    }
},
"Subnet1RouteTableAssociation": {
    "Type": "AWS::EC2::SubnetRouteTableAssociation",
    "Properties": {
        "RouteTableId": {"Ref": "RouteTable"},
        "SubnetId": {"Ref": "Subnet1"}
    }
},
"Subnet2RouteTableAssociation": {
    "Type": "AWS::EC2::SubnetRouteTableAssociation",
    "Properties": {
        "RouteTableId": {"Ref": "RouteTable"},
        "SubnetId": {"Ref": "Subnet2"}
    }
},
"InfluxDBSecurityGroup": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
        "GroupName": "influxdb-sg",
        "GroupDescription": "Security group allowing port 8086 ingress for InfluxDB",
        "VpcId": {"Ref": "VPC"}
    }
},
"InfluxDBSecurityGroupIngress": {
    "Type": "AWS::EC2::SecurityGroupIngress",
    "Properties": {
        "GroupId": {"Ref": "InfluxDBSecurityGroup"},

```

```

        "IpProtocol": "tcp",
        "CidrIp": "0.0.0.0/0",
        "FromPort": 8086,
        "ToPort": 8086
    }
},
"InfluxDBLogsS3Bucket": {
    "Type": "AWS::S3::Bucket",
    "DeletionPolicy": "Retain"
},
"InfluxDBLogsS3BucketPolicy": {
    "Type": "AWS::S3::BucketPolicy",
    "Properties": {
        "Bucket": {"Ref": "InfluxDBLogsS3Bucket"},
        "PolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Action": "s3:PutObject",
                    "Effect": "Allow",
                    "Resource": {"Fn::Sub": "arn:aws:s3:::${InfluxDBLogsS3Bucket}/InfluxLogs/
*"},
                    "Principal": {"Service": "timestream-influxdb.amazonaws.com"}
                },
                {
                    "Action": "s3:*",
                    "Effect": "Deny",
                    "Resource": [
                        {"Fn::Sub": "arn:aws:s3:::${InfluxDBLogsS3Bucket}/*"},
                        {"Fn::Sub": "arn:aws:s3:::${InfluxDBLogsS3Bucket}"}
                    ],
                    "Principal": "*",
                    "Condition": {
                        "Bool": {"aws:SecureTransport": false}
                    }
                }
            ]
        }
    }
},
"DbInstance": {
    "Type": "AWS::Timestream::InfluxDBInstance",
    "DependsOn": "InfluxDBLogsS3BucketPolicy",
    "Properties": {

```

```

        "AllocatedStorage": 20,
        "DbInstanceType": "db.influx.medium",
        "Name": {"Ref": "DbInstanceName"},
        "Password": {"Ref": "InfluxDBPassword"},
        "PubliclyAccessible": true,
        "DeploymentType": "WITH_MULTIAZ_STANDBY",
        "VpcSecurityGroupIds": [
            {"Ref": "InfluxDBSecurityGroup"}
        ],
        "VpcSubnetIds": [
            {"Ref": "Subnet1"},
            {"Ref": "Subnet2"}
        ],
        "LogDeliveryConfiguration": {
            "S3Configuration": {
                "BucketName": {"Ref": "InfluxDBLogsS3Bucket"},
                "Enabled": true
            }
        }
    },
    "Outputs": {
        "VPC": {
            "Description": "A reference to the VPC used to create network resources",
            "Value": {"Ref": "VPC"}
        },
        "Subnets": {
            "Description": "A list of the subnets created",
            "Value": {
                "Fn::Join": [
                    ",",
                    [
                        {"Ref": "Subnet1"},
                        {"Ref": "Subnet2"}
                    ]
                ]
            }
        },
        "Subnet1": {
            "Description": "A reference to the subnet in the 1st Availability Zone",
            "Value": {"Ref": "Subnet1"}
        },
        "Subnet2": {

```

```

    "Description": "A reference to the subnet in the 2nd Availability Zone",
    "Value": {"Ref": "Subnet2"}
  },
  "InfluxDBSecurityGroup": {
    "Description": "Security group with port 8086 ingress rule",
    "Value": {"Ref": "InfluxDBSecurityGroup"}
  },
  "InfluxDBLogsS3Bucket": {
    "Description": "S3 Bucket containing InfluxDB logs from the DB instance",
    "Value": {"Ref": "InfluxDBLogsS3Bucket"}
  },
  "DbInstance": {
    "Description": "A reference to the Timestream for InfluxDB DB instance",
    "Value": {"Ref": "DbInstance"}
  },
  "InfluxAuthParametersSecretArn": {
    "Description": "The Amazon Resource Name (ARN) of the AWS Secrets Manager secret
containing the initial InfluxDB authorization parameters. The secret value is a JSON
formatted key-value pair holding InfluxDB authorization values: organization, bucket,
username, and password.",
    "Value": {
      "Fn::GetAtt": [
        "DbInstance",
        "InfluxAuthParametersSecretArn"
      ]
    }
  },
  "Endpoint": {
    "Description": "The endpoint URL to connect to InfluxDB",
    "Value": {
      "Fn::Join": [
        "",
        [
          "https://",
          {
            "Fn::GetAtt": [
              "DbInstance",
              "Endpoint"
            ]
          },
          ":8086"
        ]
      ]
    }
  }
}

```

```

    }
  }
}
```

YAML

```

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      -
        Label:
          default: "Amazon Timestream for InfluxDB Configuration"
        Parameters:
          - DbInstanceName
          - InfluxDBPassword
    ParameterLabels:
      VPCCIDR:
        default: VPC CIDR

Parameters:
  DbInstanceName:
    Description: The name that uniquely identifies the DB instance when interacting
    with the Amazon Timestream for InfluxDB API and CLI commands. This name will also be a
    prefix included in the endpoint. DB instance names must be unique per customer and per
    Region.
    Type: String
    Default: mydbinstance
    MinLength: 3
    MaxLength: 40
    AllowedPattern: ^[a-zA-z][a-zA-Z0-9]*(-[a-zA-Z0-9]+)*$
  InfluxDBPassword:
    Description: The password of the initial admin user created in InfluxDB. This
    password will allow you to access the InfluxDB UI to perform various administrative
    tasks and also use the InfluxDB CLI to create an operator token. These attributes will
    be stored in a Secret created in AWS Secrets Manager in your account.
    Type: String
    NoEcho: true
    MinLength: 8
    MaxLength: 64
    AllowedPattern: ^[a-zA-Z0-9]+$

Resources:
  VPC:
```

```
Type: AWS::EC2::VPC
Properties:
  CidrBlock: 10.0.0.0/16
InternetGateway:
  Type: AWS::EC2::InternetGateway
InternetGatewayAttachment:
  Type: AWS::EC2::VPCGatewayAttachment
Properties:
  InternetGatewayId: !Ref InternetGateway
  VpcId: !Ref VPC
Subnet1:
  Type: AWS::EC2::Subnet
Properties:
  VpcId: !Ref VPC
  AvailabilityZone: !Select [0, !GetAZs '']
  CidrBlock: !Select [0, !Cidr [!GetAtt VPC.CidrBlock, 2, 12 ]]
  MapPublicIpOnLaunch: true
Subnet2:
  Type: AWS::EC2::Subnet
Properties:
  VpcId: !Ref VPC
  AvailabilityZone: !Select [1, !GetAZs '']
  CidrBlock: !Select [1, !Cidr [!GetAtt VPC.CidrBlock, 2, 12 ]]
  MapPublicIpOnLaunch: true
RouteTable:
  Type: AWS::EC2::RouteTable
Properties:
  VpcId: !Ref VPC
DefaultRoute:
  Type: AWS::EC2::Route
  DependsOn: InternetGatewayAttachment
Properties:
  RouteTableId: !Ref RouteTable
  DestinationCidrBlock: 0.0.0.0/0
  GatewayId: !Ref InternetGateway
Subnet1RouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
Properties:
  RouteTableId: !Ref RouteTable
  SubnetId: !Ref Subnet1
Subnet2RouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
Properties:
  RouteTableId: !Ref RouteTable
```

```

    SubnetId: !Ref Subnet2
InfluxDBSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupName: "influxdb-sg"
    GroupDescription: "Security group allowing port 8086 ingress for InfluxDB"
    VpcId: !Ref VPC
InfluxDBSecurityGroupIngress:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !Ref InfluxDBSecurityGroup
    IpProtocol: tcp
    CidrIp: 0.0.0.0/0
    FromPort: 8086
    ToPort: 8086
InfluxDBLogsS3Bucket:
  Type: AWS::S3::Bucket
  DeletionPolicy: Retain
InfluxDBLogsS3BucketPolicy:
  Type: AWS::S3::BucketPolicy
  Properties:
    Bucket: !Ref InfluxDBLogsS3Bucket
    PolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Action: "s3:PutObject"
          Effect: Allow
          Resource: !Sub arn:aws:s3:::${InfluxDBLogsS3Bucket}/InfluxLogs/*
          Principal:
            Service: timestream-influxdb.amazonaws.com
        - Action: "s3:*"
          Effect: Deny
          Resource:
            - !Sub arn:aws:s3:::${InfluxDBLogsS3Bucket}/*
            - !Sub arn:aws:s3:::${InfluxDBLogsS3Bucket}
          Principal: "*"
          Condition:
            Bool:
              aws:SecureTransport: false
DbInstance:
  Type: AWS::Timestream::InfluxDBInstance
  DependsOn: InfluxDBLogsS3BucketPolicy
  Properties:
    AllocatedStorage: 20

```

```

    DbInstanceType: db.influx.medium
    Name: !Ref DbInstanceName
    Password: !Ref InfluxDBPassword
    PubliclyAccessible: true
    DeploymentType: WITH_MULTIAZ_STANDBY
    VpcSecurityGroupIds:
      - !Ref InfluxDBSecurityGroup
    VpcSubnetIds:
      - !Ref Subnet1
      - !Ref Subnet2
    LogDeliveryConfiguration:
      S3Configuration:
        BucketName: !Ref InfluxDBLogsS3Bucket
        Enabled: true

```

Outputs:

Network Resources

VPC:

```

    Description: A reference to the VPC used to create network resources
    Value: !Ref VPC

```

Subnets:

```

    Description: A list of the subnets created
    Value: !Join [",", [!Ref Subnet1, !Ref Subnet2]]

```

Subnet1:

```

    Description: A reference to the subnet in the 1st Availability Zone
    Value: !Ref Subnet1

```

Subnet2:

```

    Description: A reference to the subnet in the 2nd Availability Zone
    Value: !Ref Subnet2

```

InfluxDBSecurityGroup:

```

    Description: Security group with port 8086 ingress rule
    Value: !Ref InfluxDBSecurityGroup

```

Timestream for InfluxDB Resources

InfluxDBLogsS3Bucket:

```

    Description: S3 Bucket containing InfluxDB logs from the DB instance
    Value: !Ref InfluxDBLogsS3Bucket

```

DbInstance:

```

    Description: A reference to the Timestream for InfluxDB DB instance
    Value: !Ref DbInstance

```

InfluxAuthParametersSecretArn:

```

    Description: "The Amazon Resource Name (ARN) of the AWS Secrets Manager secret
    containing the initial InfluxDB authorization parameters. The secret value is a JSON

```


formatted key-value pair holding InfluxDB authorization values: organization, bucket, username, and password."

Value: !GetAtt DbInstance.InfluxAuthParametersSecretArn

Endpoint:

Description: The endpoint URL to connect to InfluxDB

Value: !Join ["", ["https://", !GetAtt DbInstance.Endpoint, ":8086"]]

파라미터가 포함된 보다 완전한 예제

이 예제 템플릿에서는 제공된 파라미터에 따라 네트워크 리소스를 동적으로 변경합니다. 파라미터에는 PubliclyAccessible 및 DeploymentType이 포함됩니다.

JSON

```
{
  "Metadata": {
    "AWS::CloudFormation::Interface": {
      "ParameterGroups": [
        {
          "Label": {"default": "Network Configuration"},
          "Parameters": ["VPCCIDR"]
        },
        {
          "Label": {"default": "Amazon Timestream for InfluxDB Configuration"},
          "Parameters": [
            "DbInstanceName",
            "InfluxDBUsername",
            "InfluxDBPassword",
            "InfluxDBOrganization",
            "InfluxDBBucket",
            "DbInstanceType",
            "DbStorageType",
            "AllocatedStorage",
            "PubliclyAccessible",
            "DeploymentType"
          ]
        }
      ],
      "ParameterLabels": {
        "VPCCIDR": {"default": "VPC CIDR"}
      }
    }
  }
}
```

```

},
"Parameters": {
  "VPCCIDR": {
    "Description": "Please enter the IP range (CIDR notation) for the new VPC",
    "Type": "String",
    "Default": "10.0.0.0/16"
  },
  "DbInstanceName": {
    "Description": "The name that uniquely identifies the DB instance when
interacting with the Amazon Timestream for InfluxDB API and CLI commands. This name
will also be a prefix included in the endpoint. DB instance names must be unique per
customer and per Region.",
    "Type": "String",
    "Default": "mydbinstance",
    "MinLength": 3,
    "MaxLength": 40,
    "AllowedPattern": "^[a-zA-z][a-zA-Z0-9]*(-[a-zA-Z0-9]+)*$"
  },
  "InfluxDBUsername": {
    "Description": "The username of the initial admin user created in InfluxDB. Must
start with a letter and can't end with a hyphen or contain two consecutive hyphens.
For example, my-user1. This username will allow you to access the InfluxDB UI to
perform various administrative tasks and also use the InfluxDB CLI to create an
operator token. These attributes will be stored in a Secret created in AWS Secrets
Manager in your account.",
    "Type": "String",
    "Default": "admin",
    "MinLength": 1,
    "MaxLength": 64
  },
  "InfluxDBPassword": {
    "Description": "The password of the initial admin user created in InfluxDB. This
password will allow you to access the InfluxDB UI to perform various administrative
tasks and also use the InfluxDB CLI to create an operator token. These attributes will
be stored in a Secret created in AWS Secrets Manager in your account.",
    "Type": "String",
    "NoEcho": true,
    "MinLength": 8,
    "MaxLength": 64,
    "AllowedPattern": "^[a-zA-Z0-9]+$"
  },
  "InfluxDBOrganization": {
    "Description": "The name of the initial organization for the initial admin user
in InfluxDB. An InfluxDB organization is a workspace for a group of users.",

```

```

    "Type": "String",
    "Default": "org",
    "MinLength": 1,
    "MaxLength": 64
  },
  "InfluxDBBucket": {
    "Description": "The name of the initial InfluxDB bucket. All InfluxDB data
is stored in a bucket. A bucket combines the concept of a database and a retention
period (the duration of time that each data point persists). A bucket belongs to an
organization.",
    "Type": "String",
    "Default": "bucket",
    "MinLength": 2,
    "MaxLength": 64,
    "AllowedPattern": "^[^_\\\\""][^\\\\""]*$"
  },
  "DeploymentType": {
    "Description": "Specifies whether the Timestream for InfluxDB is deployed as
Single-AZ or with a MultiAZ Standby for High availability",
    "Type": "String",
    "Default": "WITH_MULTIAZ_STANDBY",
    "AllowedValues": [
      "SINGLE_AZ",
      "WITH_MULTIAZ_STANDBY"
    ]
  },
  "AllocatedStorage": {
    "Description": "The amount of storage to allocate for your DB storage type in GiB
(gibibytes).",
    "Type": "Number",
    "Default": 400,
    "MinValue": 20,
    "MaxValue": 16384
  },
  "DbInstanceType": {
    "Description": "The Timestream for InfluxDB DB instance type to run InfluxDB
on.",
    "Type": "String",
    "Default": "db.influx.medium",
    "AllowedValues": [
      "db.influx.medium",
      "db.influx.large",
      "db.influx.xlarge",
      "db.influx.2xlarge",

```

```

        "db.influx.4xlarge",
        "db.influx.8xlarge",
        "db.influx.12xlarge",
        "db.influx.16xlarge"
    ]
},
"DbStorageType": {
    "Description": "The Timestream for InfluxDB DB storage type to read and write
InfluxDB data.",
    "Type": "String",
    "Default": "InfluxIOIncludedT1",
    "AllowedValues": [
        "InfluxIOIncludedT1",
        "InfluxIOIncludedT2",
        "InfluxIOIncludedT3"
    ]
},
"PubliclyAccessible": {
    "Description": "Configures the DB instance with a public IP to facilitate
access.",
    "Type": "String",
    "Default": true,
    "AllowedValues": [
        true,
        false
    ]
}
},
"Conditions": {
    "IsMultiAZ": {
        "Fn::Equals": [
            {"Ref": "DeploymentType"},
            "WITH_MULTIAZ_STANDBY"
        ]
    },
    "IsPublic": {
        "Fn::Equals": [
            {"Ref": "PubliclyAccessible"},
            true
        ]
    }
},
"Resources": {
    "VPC": {

```

```

    "Type": "AWS::EC2::VPC",
    "Properties": {
      "CidrBlock": {"Ref": "VPCCIDR"}
    }
  },
  "InternetGateway": {
    "Type": "AWS::EC2::InternetGateway",
    "Condition": "IsPublic"
  },
  "InternetGatewayAttachment": {
    "Type": "AWS::EC2::VPCGatewayAttachment",
    "Condition": "IsPublic",
    "Properties": {
      "InternetGatewayId": {"Ref": "InternetGateway"},
      "VpcId": {"Ref": "VPC"}
    }
  },
  "Subnet1": {
    "Type": "AWS::EC2::Subnet",
    "Properties": {
      "VpcId": {"Ref": "VPC"},
      "AvailabilityZone": {
        "Fn::Select": [
          0,
          {"Fn::GetAZs": ""}
        ]
      },
      "CidrBlock": {
        "Fn::Select": [
          0,
          {
            "Fn::Cidr": [
              {
                "Fn::GetAtt": [
                  "VPC",
                  "CidrBlock"
                ]
              },
              2,
              12
            ]
          }
        ]
      }
    }
  },
},

```

```

    "MapPublicIpOnLaunch": {
      "Fn::If": [
        "IsPublic",
        true,
        false
      ]
    }
  },
  "Subnet2": {
    "Type": "AWS::EC2::Subnet",
    "Condition": "IsMultiAZ",
    "Properties": {
      "VpcId": {"Ref": "VPC"},
      "AvailabilityZone": {
        "Fn::Select": [
          1,
          {"Fn::GetAZs": ""}
        ]
      },
      "CidrBlock": {
        "Fn::Select": [
          1,
          {
            "Fn::Cidr": [
              {
                "Fn::GetAtt": [
                  "VPC",
                  "CidrBlock"
                ]
              },
              2,
              12
            ]
          }
        ]
      }
    }
  },
  "MapPublicIpOnLaunch": {
    "Fn::If": [
      "IsPublic",
      true,
      false
    ]
  }
}

```

```

    }
  },
  "RouteTable": {
    "Type": "AWS::EC2::RouteTable",
    "Properties": {
      "VpcId": {"Ref": "VPC"}
    }
  },
  "DefaultRoute": {
    "Type": "AWS::EC2::Route",
    "Condition": "IsPublic",
    "DependsOn": "InternetGatewayAttachment",
    "Properties": {
      "RouteTableId": {"Ref": "RouteTable"},
      "DestinationCidrBlock": "0.0.0.0/0",
      "GatewayId": {"Ref": "InternetGateway"}
    }
  },
  "Subnet1RouteTableAssociation": {
    "Type": "AWS::EC2::SubnetRouteTableAssociation",
    "Properties": {
      "RouteTableId": {"Ref": "RouteTable"},
      "SubnetId": {"Ref": "Subnet1"}
    }
  },
  "Subnet2RouteTableAssociation": {
    "Type": "AWS::EC2::SubnetRouteTableAssociation",
    "Condition": "IsMultiAZ",
    "Properties": {
      "RouteTableId": {"Ref": "RouteTable"},
      "SubnetId": {"Ref": "Subnet2"}
    }
  },
  "InfluxDBSecurityGroup": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
      "GroupName": "influxdb-sg",
      "GroupDescription": "Security group allowing port 8086 ingress for InfluxDB",
      "VpcId": {"Ref": "VPC"}
    }
  },
  "InfluxDBSecurityGroupIngress": {
    "Type": "AWS::EC2::SecurityGroupIngress",
    "Properties": {

```

```

    "GroupId": {"Ref": "InfluxDBSecurityGroup"},
    "IpProtocol": "tcp",
    "CidrIp": "0.0.0.0/0",
    "FromPort": 8086,
    "ToPort": 8086
  }
},
"InfluxDBLogsS3Bucket": {
  "Type": "AWS::S3::Bucket",
  "DeletionPolicy": "Retain"
},
"InfluxDBLogsS3BucketPolicy": {
  "Type": "AWS::S3::BucketPolicy",
  "Properties": {
    "Bucket": {"Ref": "InfluxDBLogsS3Bucket"},
    "PolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Action": "s3:PutObject",
          "Effect": "Allow",
          "Resource": {"Fn::Sub": "arn:aws:s3:::${InfluxDBLogsS3Bucket}/InfluxLogs/
*"},
          "Principal": {"Service": "timestream-influxdb.amazonaws.com"}
        },
        {
          "Action": "s3:*",
          "Effect": "Deny",
          "Resource": [
            {"Fn::Sub": "arn:aws:s3:::${InfluxDBLogsS3Bucket}/*"},
            {"Fn::Sub": "arn:aws:s3:::${InfluxDBLogsS3Bucket}"}
          ],
          "Principal": "*",
          "Condition": {
            "Bool": {"aws:SecureTransport": false}
          }
        }
      ]
    }
  }
},
"DbInstance": {
  "Type": "AWS::Timestream::InfluxDBInstance",
  "DependsOn": "InfluxDBLogsS3BucketPolicy",

```



```

"Properties": {
  "DbStorageType": {"Ref": "DbStorageType"},
  "AllocatedStorage": {"Ref": "AllocatedStorage"},
  "DbInstanceType": {"Ref": "DbInstanceType"},
  "Name": {"Ref": "DbInstanceName"},
  "Username": {"Ref": "InfluxDBUsername"},
  "Password": {"Ref": "InfluxDBPassword"},
  "Organization": {"Ref": "InfluxDBOrganization"},
  "Bucket": {"Ref": "InfluxDBBucket"},
  "PubliclyAccessible": {
    "Fn::If": [
      "IsPublic",
      true,
      false
    ]
  },
  "DeploymentType": {"Ref": "DeploymentType"},
  "VpcSecurityGroupIds": [
    {"Ref": "InfluxDBSecurityGroup"}
  ],
  "VpcSubnetIds": {
    "Fn::If": [
      "IsMultiAZ",
      [
        {"Ref": "Subnet1"},
        {"Ref": "Subnet2"}
      ],
      [
        {"Ref": "Subnet1"}
      ]
    ]
  },
  "LogDeliveryConfiguration": {
    "S3Configuration": {
      "BucketName": {"Ref": "InfluxDBLogsS3Bucket"},
      "Enabled": true
    }
  }
},
"Outputs": {
  "VPC": {
    "Description": "A reference to the VPC used to create network resources",

```

```

    "Value": {"Ref": "VPC"}
  },
  "Subnets": {
    "Description": "A list of the subnets created",
    "Value": {
      "Fn::If": [
        "IsMultiAZ",
        {
          "Fn::Join": [
            ",",
            [
              {"Ref": "Subnet1"},
              {"Ref": "Subnet2"}
            ]
          ]
        },
        {"Ref": "Subnet1"}
      ]
    }
  },
  "Subnet1": {
    "Description": "A reference to the subnet in the 1st Availability Zone",
    "Value": {"Ref": "Subnet1"}
  },
  "Subnet2": {
    "Condition": "IsMultiAZ",
    "Description": "A reference to the subnet in the 2nd Availability Zone",
    "Value": {"Ref": "Subnet2"}
  },
  "InfluxDBSecurityGroup": {
    "Description": "Security group with port 8086 ingress rule",
    "Value": {"Ref": "InfluxDBSecurityGroup"}
  },
  "InfluxDBLogsS3Bucket": {
    "Description": "S3 Bucket containing InfluxDB logs from the DB instance",
    "Value": {"Ref": "InfluxDBLogsS3Bucket"}
  },
  "DbInstance": {
    "Description": "A reference to the Timestream for InfluxDB DB instance",
    "Value": {"Ref": "DbInstance"}
  },
  "InfluxAuthParametersSecretArn": {
    "Description": "The Amazon Resource Name (ARN) of the AWS Secrets Manager secret
containing the initial InfluxDB authorization parameters. The secret value is a JSON

```

```
formatted key-value pair holding InfluxDB authorization values: organization, bucket,
username, and password.",
  "Value": {
    "Fn::GetAtt": [
      "DbInstance",
      "InfluxAuthParametersSecretArn"
    ]
  }
},
"Endpoint": {
  "Description": "The endpoint URL to connect to InfluxDB",
  "Value": {
    "Fn::Join": [
      "",
      [
        "https://",
        {
          "Fn::GetAtt": [
            "DbInstance",
            "Endpoint"
          ]
        },
        ":8086"
      ]
    ]
  }
}
}
```

YAML

```
Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      -
        Label:
          default: "Network Configuration"
        Parameters:
          - VPCCIDR
      -
        Label:
          default: "Amazon Timestream for InfluxDB Configuration"
```

Parameters:

- DbInstanceName
- InfluxDBUsername
- InfluxDBPassword
- InfluxDBOrganization
- InfluxDBBucket
- DbInstanceType
- DbStorageType
- AllocatedStorage
- PubliclyAccessible
- DeploymentType

ParameterLabels:**VPCCIDR:**

default: VPC CIDR

Parameters:**# Network Configuration****VPCCIDR:**

Description: Please enter the IP range (CIDR notation) for the new VPC

Type: String

Default: 10.0.0.0/16

Timestream for InfluxDB Configuration**DbInstanceName:**

Description: The name that uniquely identifies the DB instance when interacting with the Amazon Timestream for InfluxDB API and CLI commands. This name will also be a prefix included in the endpoint. DB instance names must be unique per customer and per Region.

Type: String

Default: mydbinstance

MinLength: 3

MaxLength: 40

AllowedPattern: ^[a-zA-z][a-zA-Z0-9]*(-[a-zA-Z0-9]+)*\$

InfluxDB initial user configurations**InfluxDBUsername:**

Description: The username of the initial admin user created in InfluxDB. Must start with a letter and can't end with a hyphen or contain two consecutive hyphens. For example, my-user1. This username will allow you to access the InfluxDB UI to perform various administrative tasks and also use the InfluxDB CLI to create an operator token. These attributes will be stored in a Secret created in AWS Secrets Manager in your account.

Type: String

Default: admin

MinLength: 1

MaxLength: 64

InfluxDBPassword:

Description: The password of the initial admin user created in InfluxDB. This password will allow you to access the InfluxDB UI to perform various administrative tasks and also use the InfluxDB CLI to create an operator token. These attributes will be stored in a Secret created in AWS in your account.

Type: String

NoEcho: true

MinLength: 8

MaxLength: 64

AllowedPattern: ^[a-zA-Z0-9]+\$

InfluxDBOrganization:

Description: The name of the initial organization for the initial admin user in InfluxDB. An InfluxDB organization is a workspace for a group of users.

Type: String

Default: org

MinLength: 1

MaxLength: 64

InfluxDBBucket:

Description: The name of the initial InfluxDB bucket. All InfluxDB data is stored in a bucket. A bucket combines the concept of a database and a retention period (the duration of time that each data point persists). A bucket belongs to an organization.

Type: String

Default: bucket

MinLength: 2

MaxLength: 64

AllowedPattern: ^[^_\""]*[^_\""]*\$

DeploymentType:

Description: Specifies whether the Timestream for InfluxDB is deployed as Single-AZ or with a MultiAZ Standby for High availability

Type: String

Default: WITH_MULTIAZ_STANDBY

AllowedValues:

- SINGLE_AZ

- WITH_MULTIAZ_STANDBY

AllocatedStorage:

Description: The amount of storage to allocate for your DB storage type in GiB (gibibytes).

Type: Number

Default: 400

MinValue: 20

MaxValue: 16384

DbInstanceType:

Description: The Timestream for InfluxDB DB instance type to run InfluxDB on.

Type: String

Default: db.influx.medium

AllowedValues:

- db.influx.medium
- db.influx.large
- db.influx.xlarge
- db.influx.2xlarge
- db.influx.4xlarge
- db.influx.8xlarge
- db.influx.12xlarge
- db.influx.16xlarge

DbStorageType:

Description: The Timestream for InfluxDB DB storage type to read and write InfluxDB data.

Type: String

Default: InfluxIOIncludedT1

AllowedValues:

- InfluxIOIncludedT1
- InfluxIOIncludedT2
- InfluxIOIncludedT3

PubliclyAccessible:

Description: Configures the DB instance with a public IP to facilitate access.

Type: String

Default: true

AllowedValues:

- true
- false

Conditions:

IsMultiAZ: !Equals [!Ref DeploymentType, WITH_MULTIAZ_STANDBY]

IsPublic: !Equals [!Ref PubliclyAccessible, true]

Resources:

VPC:

Type: AWS::EC2::VPC

Properties:

CidrBlock: !Ref VPCCIDR

InternetGateway:

Type: AWS::EC2::InternetGateway

Condition: IsPublic

InternetGatewayAttachment:

Type: AWS::EC2::VPCElasticNetworkInterfaceAttachment

Condition: IsPublic

Properties:

InternetGatewayId: !Ref InternetGateway

```
VpcId: !Ref VPC
Subnet1:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref VPC
    AvailabilityZone: !Select [0, !GetAZs '']
    CidrBlock: !Select [0, !Cidr [!GetAtt VPC.CidrBlock, 2, 12 ]]
    MapPublicIpOnLaunch: !If [IsPublic, true, false]
Subnet2:
  Type: AWS::EC2::Subnet
  Condition: IsMultiAZ
  Properties:
    VpcId: !Ref VPC
    AvailabilityZone: !Select [1, !GetAZs '']
    CidrBlock: !Select [1, !Cidr [!GetAtt VPC.CidrBlock, 2, 12 ]]
    MapPublicIpOnLaunch: !If [IsPublic, true, false]
RouteTable:
  Type: AWS::EC2::RouteTable
  Properties:
    VpcId: !Ref VPC
DefaultRoute:
  Type: AWS::EC2::Route
  Condition: IsPublic
  DependsOn: InternetGatewayAttachment
  Properties:
    RouteTableId: !Ref RouteTable
    DestinationCidrBlock: 0.0.0.0/0
    GatewayId: !Ref InternetGateway
Subnet1RouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
  Properties:
    RouteTableId: !Ref RouteTable
    SubnetId: !Ref Subnet1
Subnet2RouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
  Condition: IsMultiAZ
  Properties:
    RouteTableId: !Ref RouteTable
    SubnetId: !Ref Subnet2
InfluxDBSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupName: "influxdb-sg"
    GroupDescription: "Security group allowing port 8086 ingress for InfluxDB"
```

```

    VpcId: !Ref VPC
InfluxDBSecurityGroupIngress:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !Ref InfluxDBSecurityGroup
    IpProtocol: tcp
    CidrIp: 0.0.0.0/0
    FromPort: 8086
    ToPort: 8086
InfluxDBLogsS3Bucket:
  Type: AWS::S3::Bucket
  DeletionPolicy: Retain
InfluxDBLogsS3BucketPolicy:
  Type: AWS::S3::BucketPolicy
  Properties:
    Bucket: !Ref InfluxDBLogsS3Bucket
    PolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Action: "s3:PutObject"
          Effect: Allow
          Resource: !Sub arn:aws:s3:::${InfluxDBLogsS3Bucket}/InfluxLogs/*
          Principal:
            Service: timestream-influxdb.amazonaws.com
        - Action: "s3:*"
          Effect: Deny
          Resource:
            - !Sub arn:aws:s3:::${InfluxDBLogsS3Bucket}/*
            - !Sub arn:aws:s3:::${InfluxDBLogsS3Bucket}
          Principal: "*"
      Condition:
        Bool:
          aws:SecureTransport: false
DbInstance:
  Type: AWS::Timestream::InfluxDBInstance
  DependsOn: InfluxDBLogsS3BucketPolicy
  Properties:
    DbStorageType: !Ref DbStorageType
    AllocatedStorage: !Ref AllocatedStorage
    DbInstanceType: !Ref DbInstanceType
    Name: !Ref DbInstanceName
    Username: !Ref InfluxDBUsername
    Password: !Ref InfluxDBPassword
    Organization: !Ref InfluxDBOrganization

```



```

    Bucket: !Ref InfluxDBBucket
    PubliclyAccessible: !If [IsPublic, true, false]
    DeploymentType: !Ref DeploymentType
    VpcSecurityGroupIds:
      - !Ref InfluxDBSecurityGroup
    VpcSubnetIds: !If
      - IsMultiAZ
      -
        - !Ref Subnet1
        - !Ref Subnet2
      -
        - !Ref Subnet1
    LogDeliveryConfiguration:
      S3Configuration:
        BucketName: !Ref InfluxDBLogsS3Bucket
        Enabled: true

```

Outputs:

Network Resources

VPC:

Description: A reference to the VPC used to create network resources
Value: !Ref VPC

Subnets:

Description: A list of the subnets created
Value: !If
 - IsMultiAZ
 - !Join [",", [!Ref Subnet1, !Ref Subnet2]]
 - !Ref Subnet1

Subnet1:

Description: A reference to the subnet in the 1st Availability Zone
Value: !Ref Subnet1

Subnet2:

Condition: IsMultiAZ
Description: A reference to the subnet in the 2nd Availability Zone
Value: !Ref Subnet2

InfluxDBSecurityGroup:

Description: Security group with port 8086 ingress rule
Value: !Ref InfluxDBSecurityGroup

Timestream for InfluxDB Resources

InfluxDBLogsS3Bucket:

Description: S3 Bucket containing InfluxDB logs from the DB instance
Value: !Ref InfluxDBLogsS3Bucket

DbInstance:

Description: A reference to the Timestream for InfluxDB DB instance

Value: !Ref DbInstance

InfluxAuthParametersSecretArn:

Description: "The Amazon Resource Name (ARN) of the AWS Secrets Manager secret containing the initial InfluxDB authorization parameters. The secret value is a JSON formatted key-value pair holding InfluxDB authorization values: organization, bucket, username, and password."

Value: !GetAtt DbInstance.InfluxAuthParametersSecretArn

Endpoint:

Description: The endpoint URL to connect to InfluxDB

Value: !Join ["", ["https://", !GetAtt DbInstance.Endpoint, ":8086"]]

CloudFormation을 사용하여 Windows 기반 스택 배포

이 페이지에서는 Windows 기반 배포에 일반적으로 사용되는 CloudFormation 리소스에 대한 기술 참조 설명서 링크를 제공합니다.

CloudFormation은 Infrastructure as Code (IaC)를 통해 Microsoft Windows 스택을 배포하고 관리할 수 있도록 지원합니다. CloudFormation을 사용하여 Windows 기반 EC2 인스턴스, Amazon RDS의 SQL Server, Directory Service를 통한 Microsoft Active Directory를 자동으로 프로비저닝할 수 있습니다.

AWS는 Amazon EC2에 애플리케이션을 빠르게 배포할 수 있도록 Windows 플랫폼용으로 특별히 설계되어 사전 구성된 Amazon Machine Image(AMI)를 제공합니다. 이러한 AMI에는 기본 Microsoft 설정과 AWS 특정 사용자 지정이 포함됩니다. CloudFormation을 사용하면 다른 Windows Server와 마찬가지로 적절한 AMI를 선택하고, 인스턴스를 시작하고, 원격 데스크톱 연결을 사용하여 액세스할 수 있습니다. AMI에는 최적의 성능과 AWS 서비스와의 호환성을 보장하기 위해 EC2Launch(버전은 Windows Server 에디션에 따라 다름), AWS Systems Manager, CloudFormation, AWS Tools for PowerShell, 다양한 네트워크, 스토리지 및 그래픽 드라이버를 포함한 필수 소프트웨어 구성 요소가 포함되어 있습니다. 자세한 내용은 [AWS Windows AMI 참조](#)를 참조하세요.

CloudFormation은 EC2 인스턴스가 처음 부팅될 때 PowerShell 또는 배치 명령을 실행할 수 있는 UserData 스크립트와 같은 소프트웨어 구성 도구도 지원합니다. 또한 헬퍼 스크립트(cfn-init, cfn-signal, cfn-get-metadata, cfn-hup)를 제공하며 Windows 인스턴스에서 패키지, 파일, 서비스를 관리하기 위한 AWS::CloudFormation::Init 메타데이터를 지원합니다.

엔터프라이즈 환경의 경우 CloudFormation은 도메인 조인, EC2 라이선스 모델을 통한 Windows 라이선스 관리, AWS Secrets Manager를 통한 보안 자격 증명 처리를 지원합니다. CloudFormation은 버전 제어 템플릿 및 반복 가능한 배포와 결합하여 조직이 여러 AWS 리전 및 계정에서 일관되고 안전하며 확장 가능한 Windows 환경을 유지하도록 지원합니다.

Windows 기반 배포에 일반적으로 사용되는 CloudFormation 리소스에 대한 자세한 내용은 다음 기술 참조 주제를 참조하세요.

리소스 유형	설명
AWS::EC2::Instance	Windows EC2 인스턴스를 시작하기 위한 리소스입니다.
AWS::EC2::SecurityGroup	Windows 워크로드에 대한 방화벽 규칙을 정의하기 위한 리소스입니다.
AWS::AutoScaling::AutoScalingGroup AWS::EC2::LaunchTemplate	Windows EC2 인스턴스 크기를 조정하기 위한 리소스입니다.
AWS::DirectoryService::MicrosoftAD	Microsoft Active Directory를 배포하기 위한 리소스입니다.
AWS::FSx::FileSystem	FSx for Windows File Server를 배포하기 위한 리소스입니다.
AWS::RDS::DBInstance	Amazon RDS에 SQL Server를 프로비저닝하기 위한 리소스입니다.
AWS::CloudFormation::Init	EC2 메타데이터 내에서 인스턴스 구성을 위해 사용되는 리소스입니다. 자세한 내용은 Windows 기반 CloudFormation 스택 부트스트래핑 섹션을 참조하세요.
AWS::SecretsManager::Secret	자격 증명 및 Windows 암호를 안전하게 관리하기 위한 리소스입니다.
AWS::SSM::Parameter	구성 값을 안전하게 저장하기 위한 리소스입니다.
AWS::IAM::InstanceProfile AWS::IAM::Role	EC2 인스턴스에서 실행되는 애플리케이션에 권한을 부여하기 위한 리소스입니다.

Windows 기반 CloudFormation 스택 부트스트래핑

이 주제에서는 Windows 스택을 부트스트랩하고 스택 생성 문제를 해결하는 방법을 설명합니다.

주제

- [EC2 인스턴스의 사용자 데이터](#)
- [CloudFormation 헬퍼 스크립트](#)
- [Windows 스택 부트스트래핑 예](#)
- [Windows 파일 경로의 백슬래시 이스케이프 처리](#)
- [Windows 서비스 관리](#)
- [스택 생성 문제 해결](#)

EC2 인스턴스의 사용자 데이터

사용자 데이터는 Amazon EC2 실행 시에 스크립트 또는 구성 정보를 EC2 인스턴스에 전달할 수 있는 Amazon EC2 기능입니다.

Windows EC2 인스턴스의 경우:

- 배치 스크립트(<script> 태그 사용) 또는 PowerShell 스크립트(<powershell> 태그 사용)를 사용할 수 있습니다.
- 스크립트 실행은 EC2Launch에 의해 처리됩니다.

Important

CloudFormation에서 사용할 자체 Windows AMI를 생성하는 경우 EC2Launch v2가 올바르게 구성되었는지 확인합니다. 스택 생성 중에 CloudFormation 부트스트래핑 도구가 Windows 인스턴스를 올바르게 초기화하고 구성하려면 EC2Launch v2가 필요합니다. 자세한 내용은 Amazon EC2 사용 설명서의 [EC2Launch v2 에이전트를 사용하여 EC2 Windows 인스턴스 시작 중 태스크 수행](#)을 참조하세요.

AWS Windows AMI에 대한 자세한 내용은 [AWSWindows AMI 참조](#)를 참조하세요.

CloudFormation 헬퍼 스크립트

헬퍼 스크립트는 부트스트래핑 프로세스 중에 인스턴스를 구성하기 위한 유틸리티로, Amazon EC2 사용자 데이터와 함께 사용되며 강력한 구성 옵션을 제공합니다.

CloudFormation에서는 스택의 일부로 생성한 Amazon EC2 인스턴스에서 소프트웨어를 설치하고 서비스를 시작하는 데 사용할 수 있도록 다음과 같은 Python 헬퍼 스크립트 세트를 제공합니다.

- `cfn-init` - 리소스 메타데이터를 검색 및 해석하고, 패키지를 설치하고, 파일을 생성하고, 서비스를 시작하는 데 사용됩니다.
- `cfn-signal` - 필수 리소스나 애플리케이션이 준비될 때 스택에서 다른 리소스를 동기화할 수 있도록 `CreationPolicy`에서 신호를 전송하는 데 사용됩니다.
- `cfn-get-metadata` - 특정 키에 대한 리소스나 경로의 메타데이터를 검색하는 데 사용됩니다.
- `cfn-hup` - 메타데이터에 대한 업데이트가 있는지 확인하고 변경 사항이 감지된 경우 사용자 지정 후크를 실행하는 데 사용됩니다.

템플릿에서 직접 헬퍼 스크립트를 호출할 수 있습니다. 헬퍼 스크립트는 동일한 템플릿에서 정의한 리소스 메타데이터와 함께 작동합니다. 헬퍼 스크립트는 스택 생성 과정에서 Amazon EC2 인스턴스에서 실행됩니다.

자세한 내용은 CloudFormation 템플릿 참조 가이드의 [CloudFormation 헬퍼 스크립트 참조](#)를 참조하세요.

Windows 스택 부트스트래핑 예

다음 작업을 수행하는 Windows Server 서버 템플릿의 예제 코드 조각을 살펴보겠습니다.

- Windows Server 2022 AMI에서 `TestInstance`라는 EC2 인스턴스를 시작합니다.
- 간단한 테스트 파일을 생성하여 `cfn-init`가 작동하는지 확인합니다.
- 지속적인 구성 관리를 위해 `cfn-hup`를 구성합니다.
- `CreationPolicy`를 사용하여 인스턴스가 성공적인 완료를 알리는지 확인합니다.

`cfn-init` 헬퍼 스크립트는 템플릿에 있는 `AWS::CloudFormation::Init` 리소스의 정보를 기반으로 이러한 각 작업을 수행하는 데 사용됩니다.

`AWS::CloudFormation::Init` 섹션 이름은 `TestInstance`이고 이 섹션은 표준 선언으로 시작합니다.

```

TestInstance:
  Type: AWS::EC2::Instance
  Metadata:
    AWS::CloudFormation::Init:
      configSets:
        default:
          - create_files
          - start_services

```

이후에 `AWS::CloudFormation::Init`의 `files` 섹션이 선언됩니다.

```

create_files:
  files:
    c:\cfn\test.txt:
      content: !Sub |
        Hello from ${AWS::StackName}
    c:\cfn\cfn-hup.conf:
      content: !Sub |
        [main]
        stack=${AWS::StackName}
        region=${AWS::Region}
        interval=2
    c:\cfn\hooks.d\cfn-auto-reloader.conf:
      content: !Sub |
        [cfn-auto-reloader-hook]
        triggers=post.update
        path=Resources.TestInstance.Metadata.AWS::CloudFormation::Init
        action=cfn-init.exe -v -s ${AWS::StackName} -r TestInstance -c default --
region ${AWS::Region}

```

여기서 3개 파일이 생성되어 서버 인스턴스의 `C:\cfn` 디렉터리에 배치됩니다.

- `test.txt`: `cfn-init`가 올바르게 작동하는지 확인하고 동적 콘텐츠로 파일을 생성할 수 있는 간단한 테스트 파일.
- `cfn-hup.conf`: 검사 간격이 2분인 `cfn-hup`에 대한 구성 파일.
- `cfn-auto-reloader.conf` - `AWS::CloudFormation::Init`의 메타데이터가 변경될 때 `cfn-hup`가 업데이트를 시작(`cfn-init` 직접 호출)하는 데 사용되는 후크에 대한 구성 파일입니다.

다음은 Windows 서비스를 구성하는 `start_services` 섹션입니다.

```

start_services:
  services:
    windows:
      cfn-hup:
        enabled: true
        ensureRunning: true
        files:
          - c:\cfn\cfn-hup.conf
          - c:\cfn\hooks.d\cfn-auto-reloader.conf

```

이 섹션에서는 cfn-hup 서비스가 시작되고 구성 파일이 수정되면 자동으로 다시 시작되도록 보장합니다. 서비스는 CloudFormation 메타데이터의 변경 사항을 모니터링하고 업데이트가 감지되면 cfn-init를 다시 실행합니다.

다음은 Properties 섹션입니다.

```

TestInstance:
  Type: AWS::EC2::Instance
  CreationPolicy:
    ResourceSignal:
      Timeout: PT20M
  Metadata:
    AWS::CloudFormation::Init:
      # ... metadata configuration ...
  Properties:
    InstanceType: t2.large
    ImageId: '{{resolve:ssm:/aws/service/ami-windows-latest/Windows_Server-2022-English-Full-Base}}'
    SecurityGroupIds:
      - !Ref InstanceSecurityGroup
    KeyName: !Ref KeyPairName
    UserData:
      Fn::Base64: !Sub |
        <powershell>
          cfn-init.exe -v -s ${AWS::StackName} -r TestInstance -c default --region
${AWS::Region}
          cfn-signal.exe -e $lastexitcode --stack ${AWS::StackName} --resource
TestInstance --region ${AWS::Region}
        </powershell>

```

이 섹션에서 UserData 속성에는 EC2Launch에 의해 실행될 PowerShell 스크립트가 <powershell> 태그로 둘러싸여 있습니다. 스크립트는 default configSet와 함께 cfn-init를 실행한 다음 cfn-

signal을 사용하여 종료 코드를 CloudFormation에 다시 보고합니다. CreationPolicy는 스택 생성이 완료된 것으로 간주되기 전에 인스턴스가 올바르게 구성되었는지 확인하는 데 사용됩니다.

ImageId 속성은 Systems Manager Parameter Store 퍼블릭 파라미터를 사용하여 최신 Windows Server 2022 AMI ID를 자동으로 검색합니다. 이 접근 방식을 사용하면 리전별 AMI 매핑이 필요하지 않으며 항상 최신 AMI를 얻을 수 있습니다. AMI ID에 대한 Systems Manager 파라미터를 사용하는 것이 최신 AMI 참조를 유지하는 모범 사례입니다. 인스턴스에 연결하려는 경우 SecurityGroupIds 속성이 RDP 액세스를 허용하는 보안 그룹을 참조하는지 확인합니다.

CreationPolicy는 리소스 속성의 일부로 선언되며 제한 시간을 지정합니다. 인스턴스 구성이 완료되면 사용자 데이터에서 cfn-signal 명령이 신호를 보냅니다.

```
TestInstance:
  Type: AWS::EC2::Instance
  CreationPolicy:
    ResourceSignal:
      Timeout: PT20M
  Properties:
    # ... other properties ...
```

부트스트래핑 프로세스는 최소이며 파일을 생성하고 서비스만 시작하기 때문에 CreationPolicy는 제한 시간이 초과되기 전에 20분(PT20M)을 기다립니다. 제한 시간은 ISO 8601 지속 시간 형식을 사용하여 지정됩니다. Windows 인스턴스는 Linux 인스턴스보다 일반적으로 시작하는 데 시간이 오래 걸리므로 철저하게 테스트하여 요구 사항에 가장 적합한 제한 시간 값을 결정합니다.

모두 잘 진행되면 CreationPolicy가 성공적으로 완료되며 해당 퍼블릭 IP 주소를 사용하여 Windows Server 인스턴스에 액세스할 수 있습니다. 스택 생성이 완료되면 인스턴스 ID 및 퍼블릭 IP 주소가 CloudFormation 콘솔의 출력 탭에 표시됩니다.

```
Outputs:
  InstanceId:
    Value: !Ref TestInstance
    Description: Instance ID of the Windows Server
  PublicIP:
    Value: !GetAtt TestInstance.PublicIp
    Description: Public IP address of the Windows Server
```

또한 RDP를 통해 인스턴스에 연결하고 C:\cfn\test.txt 파일이 존재하고 예상 콘텐츠를 포함하는지 확인하여 부트스트래핑이 올바르게 작동하는지 수동으로 확인할 수 있습니다. Windows 인스턴스

에 연결에 대한 자세한 내용은 Amazon EC2 사용자 설명서의 [RDP를 사용하여 Windows 인스턴스에 연결](#)을 참조하세요.

Windows 파일 경로의 백슬래시 이스케이프 처리

CloudFormation 템플릿에서 Windows 경로를 참조하는 경우 항상 사용 중인 템플릿 형식에 따라 백슬래시(\)를 올바르게 이스케이프해야 합니다.

- JSON 템플릿의 경우 JSON은 백슬래시를 이스케이프 문자로 취급하므로 Windows 파일 경로에서 이중 백슬래시를 사용해야 합니다. 첫 번째 백슬래시는 두 번째 백슬래시를 이스케이프 처리하므로 단일 리터럴 백슬래시로 해석됩니다.

```
"commands" : {
  "1-extract" : {
    "command" : "C:\\\\SharePoint\\SharePointFoundation2010.exe /extract:C:\\\\SharePoint
\\SPF2010 /quiet /log:C:\\\\SharePoint\\SharePointFoundation2010-extract.log"
  }
}
```

- YAML 템플릿의 경우 일반적으로 단일 백슬래시로 충분합니다.

```
commands:
  1-extract:
    command: C:\SharePoint\SharePointFoundation2010.exe /extract:C:\SharePoint
\SPF2010 /quiet /log:C:\SharePoint\SharePointFoundation2010-extract.log
```

Windows 서비스 관리

sysvinit 대신 windows 키를 사용한다는 점을 제외하면 Linux 서비스와 동일한 방법으로 Windows 서비스를 관리합니다. 다음 예제에서는 cfn-hup 서비스를 시작하고 Automatic으로 설정한 다음 cfn-init가 c:\cfn\cfn-hup.conf 또는 c:\cfn\hooks.d\cfn-auto-reloader.conf 구성 파일을 수정하면 서비스를 다시 시작합니다.

```
services:
  windows:
    cfn-hup:
      enabled: true
      ensureRunning: true
      files:
        - c:\cfn\cfn-hup.conf
```

```
- c:\cfn\hooks.d\cfn-auto-reloader.conf
```

이름(표시 이름 아님)으로 서비스를 참조하여 동일한 방식으로 다른 Windows 서비스를 관리할 수 있습니다.

스택 생성 문제 해결

생성 중에 스택이 실패할 경우 기본 동작은 실패 시 롤백입니다. 일반적으로 기본 동작은 불필요한 과금을 피할 수 있다는 점은 좋지만 스택 생성이 실패한 원인을 디버깅하기 어렵습니다.

CloudFormation 콘솔을 사용하여 스택을 생성하거나 업데이트할 때 이 동작을 끄려면 스택 실패 옵션에서 성공적으로 프로비저닝된 리소스 보존 옵션을 선택합니다. 자세한 내용은 [리소스 프로비저닝 시 실패 처리 방법 선택](#) 섹션을 참조하세요. 그러면 인스턴스에 로그인한 후 로그 파일을 보고 시작 스크립트를 실행할 때 발생한 문제를 정확히 파악할 수 있습니다.

검토할 중요 로그는 다음과 같습니다.

- %ProgramData%\Amazon\EC2Launch\log\agent.log의 EC2 구성 로그
- C:\cfn\log\cfn-init.log의 cfn-init 로그(특정 장애 지점에 대한 종료 코드 및 오류 메시지 확인)

자세한 로그는 Amazon EC2 사용 설명서에서 다음 주제를 참조하세요.

- [EC2Launch 디렉터리 구조](#)
- [EC2Launch v2 디렉터리 구조](#)

부트스트래핑 문제 해결에 대한 자세한 내용은 [Windows 인스턴스의 CloudFormation 스택에서 부트스트랩되지 않는 도우미 스크립트 문제를 해결하려면 어떻게 해야 하나요?](#)를 참조하세요.

CloudFormation에서 제공하는 리소스 유형을 사용하여 템플릿의 기능 확장

CloudFormation은 기능을 단순한 스택 템플릿 이상으로 확대할 수 있는 여러 리소스 유형을 제공합니다.

이러한 리소스 유형에는 다음이 포함됩니다.

리소스 유형	설명	설명서
사용자 지정 리소스	AWS::CloudFormation::Custom Resource 리소스 유형에서는 특정 프로비저닝 태스크를 수행하거나 CloudFormation 리소스 유형으로 사용할 수 없는 리소스를 포함할 수 있는 사용자 지정 리소스를 생성할 수 있습니다.	사용자 지정 리소스
매크로	AWS::CloudFormation::Macro 리소스 유형에서는 CloudFormation 템플릿에서 사용자 지정 프로세싱을 수행할 수 있는 재사용 가능한 코드를 정의합니다. 매크로에서는 스택 생성 또는 업데이트 중에 템플릿을 수정하거나, 추가 리소스를 생성하거나, 기타 사용자 지정 작업을 수행할 수 있습니다.	템플릿 매크로
중첩 스택	AWS::CloudFormation::Stack 리소스 유형에서는 보다 모듈화되고 재사용 가능한 스택 아키텍처를 위해 CloudFormation 템플릿 내에 중첩 스택을 생성할 수 있습니다.	중첩 스택
StackSet	AWS::CloudFormation::StackSet 리소스 유형에서는 여러 AWS 계정 및 리전에 배포될 수 있는 스택용 컨테이너인 CloudFormation StackSet를 생성하거나 업데이트합니다.	StackSets를 사용하여 스택 관리
대기 조건	AWS::CloudFormation::WaitCondition 리소스 유형에서는 장기 실행 프로세스의 성공적인 완료 또는 외부 리소스 가용성과 같은 특정 조건이 충족될 때까지 스택 생성 또는 업데이트를 일시 중지합니다.	대기 조건
대기 조건 핸들	AWS::CloudFormation::WaitConditionHandle 리소스 유형은 AWS::CloudFormation::WaitCondition 리소스 유형과 함께 작동합니다. 특정 조건이 충족되었음을 나타내는 신호를 전송하는 데 사용되는 미리 서명된 URL을 제공합니	대기 조건

리소스 유형	설명	설명서
	다. 이러한 신호를 통해 스택 생성 또는 업데이트 프로세스를 진행할 수 있습니다.	

사용자 지정 리소스를 사용하여 사용자 지정 프로비저닝 로직 생성

사용자 지정 리소스는 CloudFormation 템플릿에 사용자 지정 프로비저닝 로직을 작성하고 스택을 생성, 업데이트(사용자 지정 리소스를 변경하는 경우) 또는 삭제할 때마다 CloudFormation이 실행되도록 하는 방법을 제공합니다. 이는 프로비저닝 요구 사항이 CloudFormation의 기본 제공 리소스 유형으로 표현할 수 없는 복잡한 로직 또는 워크플로와 관련된 경우 유용할 수 있습니다.

예를 들어 CloudFormation 리소스 유형으로 사용할 수 없는 리소스를 포함하려고 할 수 있습니다. 이러한 리소스는 사용자 지정 리소스를 사용하여 포함할 수 있습니다. 이러한 방식으로 단일 스택에서 관련 리소스를 모두 관리할 수 있습니다.

CloudFormation 템플릿에서 사용자 지정 리소스를 정의하려면

AWS::CloudFormation::CustomResource 또는 Custom::*MyCustomResourceTypeName* 리소스 유형을 사용합니다. 사용자 지정 리소스에는 하나의 속성 즉, 서비스 토큰이 필요합니다. 서비스 토큰은 CloudFormation에서 요청을 보내는 위치(예: Amazon SNS 주제 또는 Lambda 함수)를 지정합니다.

다음 주제는 사용자 지정 리소스 사용 방법에 관한 정보를 제공합니다.

주제

- [사용자 지정 리소스가 작동하는 방식](#)
- [응답 제한 시간](#)
- [CloudFormation 사용자 지정 리소스 요청 및 응답 참조](#)
- [Amazon SNS 지원 사용자 지정 리소스](#)
- [Lambda 지원 사용자 지정 리소스](#)

Note

CloudFormation 레지스트리와 사용자 지정 리소스에는 각각 고유한 이점이 있습니다. 사용자 지정 리소스는 다음과 같은 이점을 제공합니다.

- 리소스를 등록할 필요는 없습니다.

- 등록하지 않고도 전체 리소스를 템플릿의 일부로 포함할 수 있습니다.
- Create, Update 및 Delete 작업을 지원합니다.

레지스트리 기반 리소스가 제공하는 이점은 다음과 같습니다.

- 서드 파티 애플리케이션 리소스의 모델링, 프로비저닝 및 관리 지원
- Create, Read, Update, Delete 및 List(CRUDL) 작업 지원
- 프라이빗 및 서드 파티 리소스 유형에 대한 드리프트 감지 지원

사용자 지정 리소스와 달리 레지스트리 기반 리소스의 경우 CRUDL 작업을 수행하기 위해 Amazon SNS 주제 또는 Lambda 함수를 연결할 필요가 없습니다. 자세한 내용은 [CloudFormation 레지스트리를 통해 익스텐션 관리](#) 섹션을 참조하세요.

사용자 지정 리소스가 작동하는 방식

새 사용자 지정 리소스를 설정하는 일반적인 프로세스에는 다음 단계가 포함됩니다. 이러한 단계에는 사용자 지정 리소스를 소유하는 사용자 지정 리소스 공급자와 사용자 지정 리소스 유형이 포함된 템플릿을 생성하는 템플릿 개발자라는 두 가지 역할이 포함됩니다. 한 사람이 두 역할을 모두 수행할 수도 있지만, 그렇지 않은 경우 사용자 지정 리소스 공급자와 템플릿 개발자가 협력해야 합니다.

1. 사용자 지정 리소스 공급자는 CloudFormation의 요청을 처리하고 사용자 지정 리소스에서 작업을 수행하는 방법을 결정하는 로직을 작성합니다.
2. 사용자 지정 리소스 공급자는 CloudFormation이 요청을 전송할 수 있는 Amazon SNS 주제 또는 Lambda 함수를 생성합니다. Amazon SNS 주제 또는 Lambda 함수는 스택이 생성될 리전과 동일한 리전에 있어야 합니다.
3. 사용자 지정 리소스 공급자가 템플릿 개발자에게 Amazon SNS 주제 ARN 또는 Lambda 함수 ARN을 제공합니다.
4. 템플릿 개발자는 자신의 CloudFormation 템플릿에서 사용자 지정 리소스를 정의합니다. 여기에는 서비스 토큰 및 모든 입력 데이터 파라미터가 포함됩니다. 서비스 토큰 및 입력 데이터 구조는 사용자 지정 리소스 공급자가 정의합니다. 서비스 토큰은 Amazon SNS 주제 ARN 또는 Lambda 함수 ARN을 지정하며 항상 필요하지만 입력 데이터는 사용자 지정 리소스에 따르는 선택 사항입니다.

이제 사용자가 템플릿을 사용하여 사용자 지정 리소스를 생성, 업데이트 또는 삭제할 때마다 CloudFormation에서 지정된 서비스 토큰에 대한 요청을 보낸 다음 스택 작업을 진행하기 전에 응답을 기다립니다.

다음은 템플릿에서 스택을 생성하는 흐름을 요약한 것입니다.

1. CloudFormation은 지정된 서비스 토큰에 대한 요청을 보냅니다. 이 요청에는 요청 유형, 미리 서명된 Amazon S3 버킷 URL, 사용자 지정 리소스가 응답을 전송하는 위치와 같은 정보가 포함됩니다. 요청에 포함되는 정보에 대한 자세한 내용은 [CloudFormation 사용자 지정 리소스 요청 및 응답 참조](#) 단원을 참조하십시오.

다음 샘플 데이터는 CloudFormation에서 Create 요청에 포함하는 정보를 보여줍니다. 이 예제에서 ResourceProperties는 CloudFormation이 Lambda 함수로 보낼 사용자 지정 페이로드를 생성하도록 합니다.

```
{
  "RequestType" : "Create",
  "RequestId" : "unique id for this create request",
  "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/mystack/5b918d10-cd98-11ea-90d5-0a9cd3354c10",
  "ResponseURL" : "http://pre-signed-S3-url-for-response",
  "ResourceType" : "Custom::TestResource",
  "LogicalResourceId" : "MyTestResource",
  "ResourceProperties" : {
    "Name" : "Value",
    "List" : [ "1", "2", "3" ]
  }
}
```

2. 사용자 지정 리소스 공급자는 CloudFormation 요청을 처리하고 미리 서명된 URL에 SUCCESS 또는 FAILED의 응답을 반환합니다. custom resource provider는 JSON 형식 파일로 응답을 제공하고 미리 서명된 S3 URL에 업로드합니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 [미리 서명된 URL을 통해 객체 공유](#)를 참조하세요.

또한 custom resource provider는 응답에 template developer가 액세스할 수 있는 이름-값 페어를 포함할 수 있습니다. 예를 들어, 응답에는 요청에 성공한 경우에는 출력 데이터를, 요청에 실패한 경우에는 오류 메시지를 포함할 수 있습니다. 응답에 대한 자세한 내용은 [CloudFormation 사용자 지정 리소스 요청 및 응답 참조](#) 단원을 참조하십시오.

⚠ Important

이름-값 페어에 중요한 정보가 들어 있는 경우 NoEcho 필드를 사용하여 사용자 지정 리소스의 출력을 마스킹해야 합니다. 그러지 않으면 속성값(예: DescribeStackEvents)을 드러내는 API를 통해 값이 표시될 수 있습니다.

NoEcho를 사용하여 민감한 정보를 마스킹 처리하는 방법에 대한 자세한 내용은 [템플릿에 자격 증명을 포함하지 않음](#) 모범 사례를 참조하세요.

custom resource provider는 요청 수신 및 요청에 대한 응답을 담당합니다. 예를 들어 Amazon SNS 알림의 경우 사용자 지정 리소스 공급자는 특정 주제 ARN으로 전송된 알림을 수신하고 해당 알림에 응답해야 합니다. CloudFormation은 미리 서명된 URL 위치에서 응답을 대기해 수신합니다.

다음 샘플 데이터는 사용자 지정 리소스가 응답에 포함할 수 있는 정보를 보여줍니다.

```
{
  "Status" : "SUCCESS",
  "RequestId" : "unique id for this create request",
  "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/mystack/5b918d10-cd98-11ea-90d5-0a9cd3354c10",
  "LogicalResourceId" : "MyTestResource",
  "PhysicalResourceId" : "TestResource1",
  "Data" : {
    "OutputName1" : "Value1",
    "OutputName2" : "Value2",
  }
}
```

3. SUCCESS 응답을 수신한 후 CloudFormation에서는 스택 작업을 진행합니다. FAILED 응답 또는 응답 없음이 반환되면 작업에 실패합니다. 사용자 지정 리소스의 모든 출력 데이터는 미리 서명된 URL 위치에 저장됩니다. 템플릿 개발자는 [Fn::GetAtt](#) 함수를 사용하여 해당 데이터를 검색할 수 있습니다.

i Note

AWS PrivateLink를 사용하는 경우 VPC의 사용자 지정 리소스는 CloudFormation 고유의 S3 버킷에 액세스할 수 있어야 합니다. 사용자 지정 리소스는 미리 서명된 Amazon S3 URL로 응답을 전송해야 합니다. Amazon S3에 응답을 보내지 못하면 CloudFormation이 응답을

수신하지 않고 스택 작업이 실패합니다. 자세한 내용은 [인터페이스 엔드포인트를 사용하여 CloudFormation에 액세스\(AWS PrivateLink\)](#) 섹션을 참조하세요.

응답 제한 시간

사용자 지정 리소스의 기본 제한 시간은 3600초(1시간)입니다. 이 시간 동안 응답이 수신되지 않으면 스택 작업이 실패합니다.

사용자 지정 리소스의 응답에 소요될 것으로 예상되는 시간에 따라 제한 시간 값을 조정할 수 있습니다. 예를 들어 5분 이내에 응답할 것으로 예상되는 Lambda 함수를 호출하는 사용자 지정 리소스를 프로비저닝하는 경우 `ServiceTimeout` 속성을 지정하여 스택 템플릿에서 시간 제한을 5분으로 설정할 수 있습니다. 자세한 내용은 [CloudFormation 사용자 지정 리소스 요청 및 응답 참조](#) 섹션을 참조하세요. 이렇게 하면 Lambda 함수에서 오류가 발생하여 중단되는 경우 CloudFormation이 전체 시간 동안 기다리지 않고 5분 후에 스택 작업이 실패합니다.

하지만 시간 제한 값을 너무 작게 설정하지 않도록 주의하세요. 예상치 못한 시간 제한을 피하려면 사용자 지정 리소스에 필요한 작업을 수행하고 응답을 반환할 수 있는 충분한 시간이 있는지 확인하세요.

CloudFormation 사용자 지정 리소스 요청 및 응답 참조

CloudFormation은 사용자 지정 리소스 공급자와 통신하는 요청-응답 프로토콜을 통해 사용자 지정 리소스를 관리합니다. 각 요청에는 요청 유형(Create, Update 또는 Delete)이 포함되며 이 상위 수준 워크플로를 따릅니다.

1. 템플릿 개발자는 템플릿에서 `ServiceToken` 및 `ServiceTimeout`을 사용하여 사용자 지정 리소스를 정의하고 스택 작업을 시작합니다.
2. CloudFormation은 SNS 또는 Lambda를 통해 사용자 지정 리소스 공급자로 JSON 요청을 보냅니다.
3. 그러면 사용자 지정 리소스 공급자가 요청을 처리하고 제한 시간이 만료되기 전에 미리 서명된 Amazon S3 버킷 URL로 JSON 응답을 반환합니다.
4. CloudFormation은 응답을 읽고 스택 작업을 진행합니다. 제한 시간이 끝날 때까지 응답을 받지 못하면 요청이 실패로 간주되고 스택 작업에 실패합니다.

자세한 내용은 [사용자 지정 리소스가 작동하는 방식](#) 섹션을 참조하세요.

이 섹션에서는 각 요청 유형에 대한 구조, 파라미터 및 예상 응답을 설명합니다.

 Note

응답 본문의 총 크기는 4,096바이트를 초과할 수 없습니다.

템플릿 설정

템플릿에서 사용자 지정 리소스를 정의할 때 템플릿 개발자는 다음 속성과 함께 [AWS::CloudFormation::CustomResource](#)를 사용합니다.

ServiceToken

스택과 동일한 리전의 Amazon SNS 주제 ARN 또는 Lambda 함수 ARN.

필수 항목 여부: 예

유형: 문자열

ServiceTimeout

사용자 지정 리소스 작업의 제한 시간이 초과되기 전 최대 시간(초). 값은 1~3,600이어야 합니다. 기본값: 3,600초(1시간).

필수 항목 여부: 아니요

유형: 문자열

추가 리소스 속성이 지원됩니다. 리소스 속성은 요청에 `ResourceProperties`로 포함됩니다. 사용자 지정 리소스 공급자가 유효한 속성과 허용 가능한 값을 결정해야 합니다.

요청 객체

Create

템플릿 개발자가 사용자 지정 리소스를 포함하는 스택을 생성하면 CloudFormation에서 `RequestType`이 `Create`로 설정된 요청을 전송합니다.

생성 요청에는 다음과 같은 필드가 포함됩니다.

RequestType

`Create`.

필수 항목 여부: 예

유형: 문자열

RequestId

요청에 대한 고유 ID입니다.

StackId와 RequestId를 조합하여 만든 값으로 특정한 사용자 지정 리소스에 대한 요청을 고유하게 식별할 수 있습니다.

필수 항목 여부: 예

유형: 문자열

StackId

사용자 지정 리소스가 들어 있는 스택을 나타내는 Amazon 리소스 이름(ARN).

StackId와 RequestId를 조합하여 만든 값으로 특정한 사용자 지정 리소스에 대한 요청을 고유하게 식별할 수 있습니다.

필수 항목 여부: 예

유형: 문자열

ResponseURL

응답 URL은 사용자 지정 리소스 공급자에서 CloudFormation에 보내는 응답을 수신할 미리 서명된 S3 버킷을 식별합니다.

필수 항목 여부: 예

유형: 문자열

ResourceType

CloudFormation 템플릿에서 사용자 지정 리소스의 템플릿 개발자가 선택한 리소스 유형입니다. 사용자 지정 리소스 유형 이름은 최대 60자이고 영숫자 및 다음 문자를 포함할 수 있습니다. _@-

필수 항목 여부: 예

유형: 문자열

LogicalResourceId

CloudFormation 템플릿에 있는 사용자 지정 리소스의 템플릿 개발자 선택 이름(논리적 ID).

필수 항목 여부: 예

유형: 문자열

ResourceProperties

이 필드에는 템플릿 개발자가 보낸 Properties 객체의 내용이 들어 있습니다. 해당 콘텐츠는 사용자 지정 리소스 공급자에 의해 정의됩니다.

필수 항목 여부: 아니요

유형: JSON 객체

예제

```
{
  "RequestType" : "Create",
  "RequestId" : "unique-request-id",
  "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/mystack/id",
  "ResponseURL" : "pre-signed-url-for-create-response",
  "ResourceType" : "Custom::MyCustomResourceType",
  "LogicalResourceId" : "resource-logical-id",
  "ResourceProperties" : {
    "key1" : "string",
    "key2" : [ "list" ],
    "key3" : { "key4" : "map" }
  }
}
```

Update

템플릿 개발자가 템플릿 내 사용자 지정 리소스의 속성을 변경하고 스택을 업데이트하면 CloudFormation에서는 RequestType이 Update로 설정된 요청을 사용자 지정 리소스 공급자에게 전송합니다. 즉, 요청 유형이 Update인 경우 사용자 지정 리소스 코드에서 해당 속성이 변경되는 것을 알고 있으므로 변경 사항을 감지할 필요가 없습니다.

업데이트 요청에는 다음과 같은 필드가 포함됩니다.

RequestType

Update.

필수 항목 여부: 예

유형: 문자열

RequestId

요청에 대한 고유 ID입니다.

StackId와 RequestId를 조합하여 만든 값으로 특정한 사용자 지정 리소스에 대한 요청을 고유하게 식별할 수 있습니다.

필수 항목 여부: 예

유형: 문자열

StackId

사용자 지정 리소스가 들어 있는 스택을 나타내는 Amazon 리소스 이름(ARN).

StackId와 RequestId를 조합하여 만든 값으로 특정한 사용자 지정 리소스에 대한 요청을 고유하게 식별할 수 있습니다.

필수 항목 여부: 예

유형: 문자열

ResponseURL

응답 URL은 사용자 지정 리소스 공급자에서 CloudFormation에 보내는 응답을 수신할 미리 서명된 S3 버킷을 식별합니다.

필수 항목 여부: 예

유형: 문자열

ResourceType

CloudFormation 템플릿에서 사용자 지정 리소스의 템플릿 개발자가 선택한 리소스 유형입니다. 사용자 지정 리소스 유형 이름은 최대 60자이고 영숫자 및 다음 문자를 포함할 수 있습니다. _@- 업데이트 중에는 유형을 변경할 수 없습니다.

필수 항목 여부: 예

유형: 문자열

LogicalResourceId

CloudFormation 템플릿에 있는 사용자 지정 리소스의 템플릿 개발자 선택 이름(논리적 ID).

필수 항목 여부: 예

유형: 문자열

PhysicalResourceId

사용자 지정 리소스 공급자가 제공하는 물리적 ID(해당 공급자에서 고유함).

필수 항목 여부: 예

유형: 문자열

ResourceProperties

이 필드에는 템플릿 개발자가 보낸 Properties 객체의 내용이 들어 있습니다. 해당 콘텐츠는 사용자 지정 리소스 공급자에 의해 정의됩니다.

필수 항목 여부: 아니요

유형: JSON 객체

OldResourceProperties

Update 요청에만 사용됩니다. CloudFormation 템플릿에서 템플릿 개발자가 이전에 선언한 새 리소스 속성값입니다.

필수 항목 여부: 예

유형: JSON 객체

예제

```
{
  "RequestType" : "Update",
  "RequestId" : "unique-request-id",
  "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/mystack/id",
  "ResponseURL" : "pre-signed-url-for-update-response",
  "ResourceType" : "Custom::MyCustomResourceType",
```

```
"LogicalResourceId" : "resource-logical-id",
"PhysicalResourceId" : "provider-defined-physical-id",
"ResourceProperties" : {
    "key1" : "new-string",
    "key2" : [ "new-list" ],
    "key3" : { "key4" : "new-map" }
},
"OldResourceProperties" : {
    "key1" : "string",
    "key2" : [ "list" ],
    "key3" : { "key4" : "map" }
}
}
```

Delete

템플릿 개발자가 스택을 삭제하거나 스택에서 사용자 지정 리소스를 제거하면 CloudFormation에서 RequestType이 Delete로 설정된 요청을 전송합니다.

삭제 요청에는 다음과 같은 필드가 포함됩니다.

RequestType

Delete.

필수 항목 여부: 예

유형: 문자열

RequestId

요청에 대한 고유 ID입니다.

필수 항목 여부: 예

유형: 문자열

StackId

사용자 지정 리소스가 들어 있는 스택을 나타내는 Amazon 리소스 이름(ARN).

필수 항목 여부: 예

유형: 문자열

ResponseURL

응답 URL은 사용자 지정 리소스 공급자에서 CloudFormation에 보내는 응답을 수신할 미리 서명된 S3 버킷을 식별합니다.

필수 항목 여부: 예

유형: 문자열

ResourceType

CloudFormation 템플릿에서 사용자 지정 리소스의 템플릿 개발자가 선택한 리소스 유형입니다. 사용자 지정 리소스 유형 이름은 최대 60자이고 영숫자 및 다음 문자를 포함할 수 있습니다. _@-

필수 항목 여부: 예

유형: 문자열

LogicalResourceId

CloudFormation 템플릿에 있는 사용자 지정 리소스의 템플릿 개발자 선택 이름(논리적 ID).

필수 항목 여부: 예

유형: 문자열

PhysicalResourceId

사용자 지정 리소스 공급자가 제공하는 물리적 ID(해당 공급자에서 고유함).

필수 항목 여부: 예

유형: 문자열

ResourceProperties

이 필드에는 템플릿 개발자가 보낸 Properties 객체의 내용이 들어 있습니다. 해당 콘텐츠는 사용자 지정 리소스 공급자에 의해 정의됩니다.

필수 항목 여부: 아니요

유형: JSON 객체

예제

```
{
  "RequestType" : "Delete",
  "RequestId" : "unique-request-id",
  "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/mystack/id",
  "ResponseURL" : "pre-signed-url-for-delete-response",
  "ResourceType" : "Custom::MyCustomResourceType",
  "LogicalResourceId" : "resource-logical-id",
  "PhysicalResourceId" : "provider-defined-physical-id",
  "ResourceProperties" : {
    "key1" : "string",
    "key2" : [ "list" ],
    "key3" : { "key4" : "map" }
  }
}
```

응답 객체

사용자 지정 리소스 공급자는 모든 요청 유형에 대해 미리 서명된 URL로 응답을 전송합니다. 사용자 지정 리소스 공급자가 응답을 전송하지 않으면 CloudFormation은 작업의 제한 시간이 초과될 때까지 기다립니다.

이 응답은 다음 필드를 포함하는 JSON 객체여야 합니다.

Status

SUCCESS 또는 FAILED여야 합니다.

필수 항목 여부: 예

유형: 문자열

RequestId

요청에 대한 고유 ID입니다. 이 값을 요청에 표시된 그대로 복사합니다.

필수 항목 여부: 예

유형: 문자열

StackId

사용자 지정 리소스가 들어 있는 스택을 나타내는 Amazon 리소스 이름(ARN). 이 값을 요청에 표시된 그대로 복사합니다.

필수 항목 여부: 예

유형: 문자열

LogicalResourceId

CloudFormation 템플릿에 있는 사용자 지정 리소스의 템플릿 개발자 선택 이름(논리적 ID). 이 값을 요청에 표시된 그대로 복사합니다.

필수 항목 여부: 예

유형: 문자열

PhysicalResourceId

이 값은 사용자 지정 리소스 공급업체 고유의 ID여야 하며, 크기는 최대 1KB까지 가능합니다. 값은 빈 문자열일 수 없으며, 동일한 리소스의 모든 응답에 대해 동일해야 합니다.

사용자 지정 리소스를 업데이트할 때 PhysicalResourceId에 대해 반환되는 값에 따라 업데이트 동작이 결정됩니다. 값이 동일하게 유지되는 경우 CloudFormation은 이를 일반 업데이트로 간주합니다. 값이 변경되는 경우 CloudFormation은 업데이트를 교체로 이해하고 기존 리소스에 삭제 요청을 전송합니다. 자세한 내용은 [AWS::CloudFormation::CustomResource](#) 섹션을 참조하세요.

필수 항목 여부: 예

유형: 문자열

Reason

실패 응답의 이유를 설명합니다.

Status이 FAILED인 경우 필수입니다. 그렇지 않은 경우 선택 사항입니다.

필수 항목 여부: 조건부

유형: 문자열

NoEcho

Fn::GetAtt 함수를 사용하여 조회할 때 사용자 지정 리소스의 출력을 마스킹할지 여부를 나타냅니다. true로 설정하면 템플릿의 Metadata 섹션에 저장된 값을 제외하고 반환된 모든 값은 별표(*****)로 마스킹됩니다. CloudFormation은 Metadata 섹션에 포함된 정보를 변환, 수정 또는 삭제하지 않습니다. 기본값은 false입니다.

NoEcho를 사용하여 민감한 정보를 마스킹 처리하는 방법에 대한 자세한 내용은 [템플릿에 자격 증명을 포함하지 않음](#) 모범 사례를 참조하세요.

Create 및 Update 응답에 대해서만 사용할 수 있습니다. Delete 응답에 대해서는 지원되지 않습니다.

필수 항목 여부: 아니요

유형: 부울

Data

응답과 함께 보낼 사용자 지정 리소스 공급자 정의 이름-값 페어. Fn::GetAtt와 템플릿의 이름을 사용해 여기 제공된 값에 액세스할 수 있습니다.

Create 및 Update 응답에 대해서만 사용할 수 있습니다. Delete 응답에 대해서는 지원되지 않습니다.

Important

이름-값 페어에 중요한 정보가 들어 있는 경우 NoEcho 필드를 사용하여 사용자 지정 리소스의 출력을 마스킹해야 합니다. 그러지 않으면 속성값(예: DescribeStackEvents)을 드러내는 API를 통해 값이 표시될 수 있습니다.

필수 항목 여부: 아니요

유형: JSON 객체

성공 응답 예제

Create 및 Update 응답

```
{
  "Status": "SUCCESS",
  "RequestId": "unique-request-id",
  "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/name/id",
  "LogicalResourceId": "resource-logical-id",
  "PhysicalResourceId": "provider-defined-physical-id",
  "NoEcho": true,
  "Data": {
    "key1": "value1",
```

```

    "key2": "value2"
  }
}

```

Delete 응답

```

{
  "Status": "SUCCESS",
  "RequestId": "unique-request-id",
  "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/name/id",
  "LogicalResourceId": "resource-logical-id",
  "PhysicalResourceId": "provider-defined-physical-id"
}

```

실패 응답 예제

```

{
  "Status": "FAILED",
  "RequestId": "unique-request-id",
  "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/name/id",
  "LogicalResourceId": "resource-logical-id",
  "PhysicalResourceId": "provider-defined-physical-id",
  "Reason": "Required failure reason string"
}

```

Amazon SNS 지원 사용자 지정 리소스

다음 주제는 CloudFormation이 요청을 보내는 Amazon SNS 주제를 지정하는 서비스 토큰으로 사용자 지정 리소스를 구성하는 방법을 보여줍니다. 또한 사용자 지정 리소스 스택 생성, 업데이트 및 삭제의 결과로 전송 및 수신되는 이벤트 및 메시지 시퀀스를 학습합니다.

사용자 지정 리소스와 Amazon SNS를 사용하면 스택에 새로운 리소스를 추가하고 동적 데이터를 스택에 주입하는 것과 같은 시나리오를 활성화할 수 있습니다. 예를 들어, 스택을 생성할 때 CloudFormation은 Amazon EC2 인스턴스에서 실행 중인 애플리케이션이 모니터링하는 주제에 Create 요청을 전송할 수 있습니다. Amazon SNS 알림은 허용 목록에 있는 탄력적 IP 주소의 풀 가져 오기와 같은 추가 프로비저닝 태스크를 수행하도록 애플리케이션을 트리거합니다. 완료된 후 애플리케이션은 CloudFormation에 스택 작업을 계속하도록 알리는 응답(및 출력 데이터)을 전송합니다.

Amazon SNS 주제를 사용자 지정 리소스의 대상으로 지정하면 CloudFormation은 사용자 지정 리소스와 연관된 스택 작업 도중 지정된 SNS 주제에 메시지를 전송합니다. 이러한 메시지를 처리하고 필요한 작업을 수행하려면 지원되는 엔드포인트에서 SNS 주제를 구독해야 합니다.

사용자 지정 리소스의 소개와 작동 방식은 [사용자 지정 리소스가 작동하는 방식](#) 섹션을 참조하세요. Amazon SNS와 작동 방식에 대한 자세한 설명은 [Amazon Simple Notification Service 개발자 안내서](#)를 참조하세요.

Amazon SNS를 사용하여 사용자 지정 리소스 생성

주제

- [1단계: 스택 생성](#)
- [2단계: 스택 업데이트](#)
- [3단계: 스택 삭제](#)

1단계: 스택 생성

1. 템플릿 개발자는 사용자 지정 리소스가 포함된 CloudFormation 스택을 생성합니다.

아래 템플릿 예제에서는 논리적 ID가 *MySeleniumTest*인 사용자 지정 리소스에 대해 사용자 지정 리소스 이름 `Custom::SeleniumTester`를 사용합니다. 사용자 지정 리소스 유형 이름은 영숫자여야 하며 최대 60자 길이일 수 있습니다.

사용자 지정 리소스 유형은 사용자 지정 리소스 공급자에서 정의되는 서비스 토큰, 선택적 공급자별 속성 및 선택적 `Fn::GetAtt` 속성을 사용하여 선언됩니다. 이러한 속성은 template developer에서 custom resource provider로 및 반대로 정보를 전달하는 데 사용할 수 있습니다. 서비스 토큰은 리소스 공급자가 구성한 Amazon SNS 주제를 지정합니다.

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Resources" : {
    "MySeleniumTest" : {
      "Type": "Custom::SeleniumTester",
      "Version" : "1.0",
      "Properties" : {
        "ServiceToken": "arn:aws:sns:us-west-2:123456789012:CRTest",
        "seleniumTester" : "SeleniumTest()",
        "endpoints" : [ "http://mysite.com", "http://myecommercesite.com/",
          "http://search.mysite.com" ],
        "frequencyOfTestsPerHour" : [ "3", "2", "4" ]
      }
    }
  },
  "Outputs" : {
```

```

    "topItem" : {
      "Value" : { "Fn::GetAtt" : [ "MySeleniumTest", "resultsPage" ] }
    },
    "numRespondents" : {
      "Value" : { "Fn::GetAtt" : [ "MySeleniumTest", "lastUpdate" ] }
    }
  }
}

```

Note

공급자가 CloudFormation에 응답하는 동안 사용자 지정 리소스 공급자는 Fn::GetAtt를 사용하여 액세스하는 데이터의 이름과 값을 반환합니다. custom resource provider가 타사인 경우 template developer는 custom resource provider에서 이러한 반환 값의 이름을 얻어야 합니다.

2. CloudFormation은 스택에 대한 정보, 스택 템플릿의 사용자 지정 리소스 속성 및 응답을 위한 S3 URL이 포함된 "RequestType" : "Create"을 사용하여 Amazon SNS 알림을 리소스 공급자에게 전송합니다.

알림을 전송하는 데 사용되는 SNS 주제는 ServiceToken 속성의 템플릿에 포함됩니다. 하드 코딩된 값을 사용하지 않으려면 템플릿 개발자는 스택이 시작되는 시간에 값이 입력되도록 템플릿 파라미터를 사용할 수 있습니다.

다음 예제에서는 Create의 Custom::SeleniumTester를 사용하여 생성된 사용자 지정 리소스 유형 이름인 LogicalResourceId가 포함되어 있는 사용자 지정 리소스 MySeleniumTester 요청을 보여 줍니다.

```

{
  "RequestType" : "Create",
  "RequestId" : "unique-request-id",
  "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/mystack/5b918d10-cd98-11ea-90d5-0a9cd3354c10",
  "ResponseURL" : "http://pre-signed-S3-url-for-response",
  "ResourceType" : "Custom::SeleniumTester",
  "LogicalResourceId" : "MySeleniumTester",
  "ResourceProperties" : {
    "seleniumTester" : "SeleniumTest()",
    "endpoints" : [ "http://mysite.com", "http://myecommercesite.com/", "http://search.mysite.com" ],
    "frequencyOfTestsPerHour" : [ "3", "2", "4" ]
  }
}

```

```
}
}
```

Create 요청에 대한 요청 객체에 관한 자세한 내용은 [요청 및 응답 참조](#) 주제를 참조하세요.

3. custom resource provider는 template developer에서 전송된 데이터를 처리하고 Create 요청이 성공적이었는지 여부를 결정합니다. 그런 다음 리소스 공급자는 CloudFormation에서 전송된 S3 URL을 사용하여 SUCCESS 또는 FAILED의 응답을 전송합니다.

응답 유형에 따라 다른 응답 필드가 CloudFormation에서 예상됩니다. 특정 요청 유형의 응답 필드에 대한 자세한 내용은 [요청 및 응답 참조](#) 섹션에서 해당 요청 유형에 대한 설명서를 참조하세요.

생성 또는 업데이트 요청에 응답하여 사용자 지정 리소스 공급자가 응답의 Data 필드에서 데이터 요소를 반환할 수 있습니다. 이러한 요소는 이름/값 페어이며, 이름은 스택 템플릿에서 사용자 지정 리소스와 함께 사용된 Fn::GetAtt 속성에 해당합니다. 값은 템플릿 개발자가 속성 이름을 사용하여 리소스에서 Fn::GetAtt를 호출할 때 반환되는 데이터입니다.

다음은 사용자 지정 리소스 응답의 예입니다.

```
{
  "Status" : "SUCCESS",
  "RequestId" : "unique-request-id",
  "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/mystack/5b918d10-cd98-11ea-90d5-0a9cd3354c10",
  "LogicalResourceId" : "MySeleniumTester",
  "PhysicalResourceId" : "Tester1",
  "Data" : {
    "resultsPage" : "http://www.myexampledomain/test-results/guid",
    "lastUpdate" : "2012-11-14T03:30Z"
  }
}
```

Create 요청에 대한 응답 객체에 관한 자세한 내용은 [요청 및 응답 참조](#) 주제를 참조하세요.

요청의 StackId, RequestId 및 LogicalResourceId 필드를 정확히 복사해야 합니다.

4. CloudFormation은 스택 상태를 CREATE_COMPLETE 또는 CREATE_FAILED로 선언합니다. 스택이 성공적으로 생성된 경우 템플릿 개발자는 [Fn::GetAtt](#)를 통해 생성된 사용자 지정 리소스의 출력값에 액세스하여 해당 값을 사용할 수 있습니다.

예를 들어, 그림에 사용된 사용자 지정 리소스 템플릿은 Fn::GetAtt를 사용하여 리소스 출력을 스택 출력에 복사했습니다.

```

"Outputs" : {
  "topItem" : {
    "Value" : { "Fn::GetAtt" : ["MySeleniumTest", "resultsPage"] }
  },
  "numRespondents" : {
    "Value" : { "Fn::GetAtt" : ["MySeleniumTest", "lastUpdate"] }
  }
}

```

2단계: 스택 업데이트

기존 스택을 업데이트하려면 아래 예제와 같이 스택에서 리소스의 속성에 대한 업데이트를 지정하는 템플릿을 제출해야 합니다. CloudFormation은 템플릿에 지정된 변경 사항이 있는 리소스만 업데이트합니다. 자세한 내용은 [스택 리소스의 업데이트 동작 이해](#) 섹션을 참조하세요.

기본 물리적 리소스를 교체해야 하는 사용자 지정 리소스를 업데이트할 수 있습니다. CloudFormation 템플릿에서 사용자 지정 리소스를 업데이트할 경우 CloudFormation은 해당 사용자 지정 리소스에 업데이트 요청을 전송합니다. 사용자 지정 리소스에 교체가 필요한 경우, 새 사용자 지정 리소스는 새 물리적 ID로 응답을 전송해야 합니다. 응답을 수신한 CloudFormation은 기존 및 신규 사용자 지정 리소스의 PhysicalResourceId를 비교합니다. [3단계: 스택 삭제](#)에서 설명한 바와 같이, 두 값이 다른 경우 CloudFormation은 업데이트를 교체로 인식하고 기존 리소스에 삭제 요청을 전송합니다.

Note

사용자 지정 리소스에 변경 사항을 적용하지 않으면 CloudFormation은 스택 업데이트 중 해당 리소스에 요청을 전송하지 않습니다.

1. template developer는 사용자 지정 리소스가 포함된 스택에 대한 업데이트를 시작합니다. 업데이트 중 template developer는 스택 템플릿에서 새로운 속성을 지정할 수 있습니다.

다음은 사용자 지정 리소스 유형을 사용하는 스택 템플릿에 대한 Update의 예입니다.

```

{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Resources" : {
    "MySeleniumTest" : {
      "Type": "Custom::SeleniumTester",
      "Version" : "1.0",

```

```

    "Properties" : {
        "ServiceToken": "arn:aws:sns:us-west-2:123456789012:CRTest",
        "seleniumTester" : "SeleniumTest()",
        "endpoints" : [ "http://mysite.com", "http://myecommercesite.com/",
"http://search.mysite.com",
        "http://mynewsite.com" ],
        "frequencyOfTestsPerHour" : [ "3", "2", "4", "3" ]
    }
},
"Outputs" : {
    "topItem" : {
        "Value" : { "Fn::GetAtt" : ["MySeleniumTest", "resultsPage"] }
    },
    "numRespondents" : {
        "Value" : { "Fn::GetAtt" : ["MySeleniumTest", "lastUpdate"] }
    }
}
}

```

2. CloudFormation은 OldResourceProperties 필드에 기존 리소스 속성이 포함되고 ResourceProperties에 업데이트된(있는 경우) 리소스 속성이 포함된다는 점을 제외하고 Create 호출과 비슷한 정보가 포함되는 "RequestType" : "Update"를 사용하여 Amazon SNS 알림을 리소스 공급자에게 전송합니다.

다음은 Update 요청의 예입니다.

```

{
    "RequestType" : "Update",
    "RequestId" : "unique-request-id",
    "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/mystack/5b918d10-cd98-11ea-90d5-0a9cd3354c10",
    "ResponseURL" : "http://pre-signed-S3-url-for-response",
    "ResourceType" : "Custom::SeleniumTester",
    "LogicalResourceId" : "MySeleniumTester",
    "PhysicalResourceId" : "Tester1",
    "ResourceProperties" : {
        "seleniumTester" : "SeleniumTest()",
        "endpoints" : [ "http://mysite.com", "http://myecommercesite.com/", "http://search.mysite.com",
        "http://mynewsite.com" ],
        "frequencyOfTestsPerHour" : [ "3", "2", "4", "3" ]
    },
}

```



```

"OldResourceProperties" : {
    "seleniumTester" : "SeleniumTest()",
    "endpoints" : [ "http://mysite.com", "http://myecommercesite.com/", "http://search.mysite.com" ],
    "frequencyOfTestsPerHour" : [ "3", "2", "4" ]
}
}

```

Update 요청에 대한 요청 객체에 관한 자세한 내용은 [요청 및 응답 참조](#) 주제를 참조하세요.

3. 사용자 지정 리소스 공급자는 CloudFormation에서 전송된 데이터를 처리합니다. 사용자 지정 리소스는 업데이트를 수행하고 SUCCESS 또는 FAILED의 응답을 S3 URL에 전송합니다. 그런 다음 CloudFormation은 기존 및 신규 사용자 지정 리소스의 PhysicalResourceIDs를 비교합니다. 값이 다른 경우 CloudFormation은 업데이트에 교체가 필요하다고 인식하고 기존 리소스에 삭제 요청을 전송합니다. 다음 예제에서는 Update 요청에 대한 custom resource provider 응답을 보여줍니다.

```

{
    "Status" : "SUCCESS",
    "RequestId" : "unique-request-id",
    "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/mystack/5b918d10-cd98-11ea-90d5-0a9cd3354c10",
    "LogicalResourceId" : "MySeleniumTester",
    "PhysicalResourceId" : "Tester2"
}

```

Update 요청에 대한 응답 객체에 관한 자세한 내용은 [요청 및 응답 참조](#) 주제를 참조하세요.

요청의 StackId, RequestId 및 LogicalResourceId 필드를 정확히 복사해야 합니다.

4. CloudFormation은 스택 상태를 UPDATE_COMPLETE 또는 UPDATE_FAILED로 선언합니다. 업데이트가 실패하면 스택은 롤백합니다. 스택이 성공적으로 업데이트된 경우 template developer는 Fn::GetAtt를 사용하여 생성된 사용자 지정 리소스의 새로운 출력값에 액세스할 수 있습니다.

3단계: 스택 삭제

1. 템플릿 개발자는 사용자 지정 리소스가 포함된 스택을 삭제합니다. CloudFormation은 SNS 주제와 함께 스택 템플릿에 지정된 현재 속성을 가져오고, 사용자 지정 리소스 공급자에게 요청하기 위해 준비합니다.

2. CloudFormation은 스택에 대한 정보, 스택 템플릿의 사용자 지정 리소스 속성 및 응답을 위한 S3 URL이 포함된 "RequestType" : "Delete"을 사용하여 Amazon SNS 알림을 리소스 공급자에게 전송합니다.

스택을 삭제하거나 사용자 지정 리소스를 제거하거나 교체하는 업데이트를 수행할 때마다 CloudFormation은 기존 및 신규 사용자 지정 리소스 간에 PhysicalResourceId를 비교합니다. 값이 다른 경우 다음 Delete 요청 예제와 같이 CloudFormation은 업데이트를 교체로 인식하고 기존 리소스(OldPhysicalResource)에 대한 삭제 요청을 전송합니다.

```
{
  "RequestType" : "Delete",
  "RequestId" : "unique-request-id",
  "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/mystack/5b918d10-cd98-11ea-90d5-0a9cd3354c10",
  "ResponseURL" : "http://pre-signed-S3-url-for-response",
  "ResourceType" : "Custom::SeleniumTester",
  "LogicalResourceId" : "MySeleniumTester",
  "PhysicalResourceId" : "Tester1",
  "ResourceProperties" : {
    "seleniumTester" : "SeleniumTest()",
    "endpoints" : [ "http://mysite.com", "http://myecommercesite.com/", "http://search.mysite.com",
      "http://mynewsite.com" ],
    "frequencyOfTestsPerHour" : [ "3", "2", "4", "3" ]
  }
}
```

Delete 요청에 대한 요청 객체에 관한 자세한 내용은 [요청 및 응답 참조](#) 주제를 참조하세요.

DescribeStackResource, DescribeStackResources 및 ListStackResources는 지정된 경우 사용자 정의 이름을 표시합니다.

3. 사용자 지정 리소스 공급자는 CloudFormation에서 전송된 데이터를 처리하고 Delete 요청이 성공적이었는지 여부를 결정합니다. 그런 다음 리소스 공급자는 CloudFormation에서 전송된 S3 URL을 사용하여 SUCCESS 또는 FAILED의 응답을 전송합니다. 사용자 지정 리소스가 포함된 스택을 성공적으로 삭제하려면 custom resource provider에서 삭제 요청에 성공적으로 응답해야 합니다.

다음은 Delete 요청에 대한 custom resource provider 응답의 예입니다.

```
{
  "Status" : "SUCCESS",
```

```

    "RequestId" : "unique-request-id",
    "StackId" : "arn:aws:cloudformation:us-west-2:123456789012:stack/mystack/5b918d10-
cd98-11ea-90d5-0a9cd3354c10",
    "LogicalResourceId" : "MySeleniumTester",
    "PhysicalResourceId" : "Tester1"
}

```

Delete 요청에 대한 응답 객체에 관한 자세한 내용은 [요청 및 응답 참조](#) 주제를 참조하세요.

요청의 StackId, RequestId 및 LogicalResourceId 필드를 정확히 복사해야 합니다.

4. CloudFormation은 스택 상태를 DELETE_COMPLETE 또는 DELETE_FAILED로 선언합니다.

Lambda 지원 사용자 지정 리소스

Lambda 함수를 사용자 지정 리소스와 연결하면 해당 사용자 지정 리소스가 생성, 업데이트 또는 삭제될 때마다 이 함수가 호출됩니다. CloudFormation은 Lambda API를 호출해 이러한 함수를 호출하고 모든 요청 데이터(예: 요청 유형 및 리소스 속성)를 이 함수로 전달합니다. CloudFormation과 함께 Lambda 함수의 이점 및 사용자 지정 가능성 덕분에 스택 생성 중 동적으로 AMI ID 조회 또는 유틸리티 함수(예: 문자열 반전 함수) 구현 및 사용과 같은 광범위한 시나리오를 처리할 수 있습니다.

사용자 지정 리소스의 소개와 작동 방식은 [사용자 지정 리소스가 작동하는 방식](#) 섹션을 참조하세요.

주제

- [연습: Lambda 지원 사용자 지정 리소스를 사용하여 지연 메커니즘 생성](#)
- [cfn-response 모듈](#)

연습: Lambda 지원 사용자 지정 리소스를 사용하여 지연 메커니즘 생성

이 연습에서는 샘플 CloudFormation 템플릿을 사용하여 Lambda 지원 사용자 지정 리소스를 구성하고 시작하는 방법을 보여줍니다. 이 템플릿은 지정된 시간 동안 스택 배포를 일시 중지하는 지연 메커니즘을 생성합니다. 이는 종속 리소스가 생성되기 전에 리소스가 안정화될 때까지 기다리는 경우와 같이 리소스 프로비저닝 중에 의도적으로 지연시켜야 하는 경우에 유용할 수 있습니다.

Note

이전에는 AMI ID를 검색하는 데 Lambda 지원 사용자 지정 리소스가 권장되었지만 이제 AWS Systems Manager 파라미터를 사용하는 것이 좋습니다. 이러한 접근 방식을 통해 더 쉽게 템플

릿을 재사용하고 유지 관리할 수 있습니다. 자세한 내용은 [Systems Manager Parameter Store에서 일반 텍스트 값 가져오기](#) 섹션을 참조하세요.

주제

- [개요](#)
- [샘플 템플릿](#)
- [샘플 템플릿 연습](#)
- [사전 조건](#)
- [스택 시작](#)
- [리소스 정리](#)
- [관련 정보](#)

개요

이 연습에서 사용된 샘플 스택 템플릿은 Lambda 지원 사용자 지정 리소스를 생성합니다. 이 사용자 지정 리소스는 스택 생성 중에 구성 가능한 지연(기본값 60초)을 도입합니다. 지연은 사용자 지정 리소스의 속성이 수정되는 경우에만 스택 업데이트 중에 발생합니다.

템플릿은 다음 리소스를 프로비저닝합니다.

- 사용자 지정 리소스,
- Lambda 함수, 및
- Lambda가 CloudWatch에 로그를 작성할 수 있도록 하는 IAM 역할.

이는 또한 다음 두 가지 출력도 정의합니다.

- 함수가 대기한 실제 시간입니다.
- Lambda 함수의 각 실행 동안 생성되는 고유 식별자입니다.

Note

CloudFormation은 무료 서비스이지만 Lambda는 함수에 대한 요청 수 및 코드 실행 시간에 따라 요금을 부과합니다. Lambda 요금에 대한 자세한 내용은 [AWS Lambda 요금](#)을 참조하세요.

샘플 템플릿

아래 지연 메커니즘을 사용하여 Lambda 지원 사용자 지정 리소스 샘플 템플릿을 확인할 수 있습니다.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Resources": {
    "LambdaExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Statement": [{
            "Effect": "Allow",
            "Principal": { "Service": ["lambda.amazonaws.com"] },
            "Action": ["sts:AssumeRole"]
          }]
        },
        "Path": "/",
        "Policies": [{
          "PolicyName": "AllowLogs",
          "PolicyDocument": {
            "Statement": [{
              "Effect": "Allow",
              "Action": ["logs:*"],
              "Resource": "*"
            }]
          }
        ]
      }
    },
    "CFNWaiter": {
      "Type": "AWS::Lambda::Function",
      "Properties": {
        "Handler": "index.handler",
        "Runtime": "python3.9",
        "Timeout": 900,
        "Role": { "Fn::GetAtt": ["LambdaExecutionRole", "Arn"] },
        "Code": {
          "ZipFile": { "Fn::Join": ["\n", [
            "from time import sleep",
            "import json",
            "import cfnresponse",
```

```

        "import uuid",
        "",
        "def handler(event, context):",
        "    wait_seconds = 0",
        "    id = str(uuid.uuid1())",
        "    if event[\"RequestType\"] in [\"Create\", \"Update\"]:",
        "        wait_seconds = int(event[\"ResourceProperties\"].get(\"ServiceTimeout", 0))",
        "        sleep(wait_seconds)",
        "        response = {",
        "            \"TimeWaited\": wait_seconds,",
        "            \"Id\": id ",
        "        }",
        "        cfnresponse.send(event, context, cfnresponse.SUCCESS, response, \"Waiter-\"+id)",
        "    ]",
        "}"
    ],
    "CFNWaiterCustomResource": {
        "Type": "AWS::CloudFormation::CustomResource",
        "Properties": {
            "ServiceToken": { "Fn::GetAtt": ["CFNWaiter", "Arn"] },
            "ServiceTimeout": 60
        }
    },
    "Outputs": {
        "TimeWaited": {
            "Value": { "Fn::GetAtt": ["CFNWaiterCustomResource", "TimeWaited"] },
            "Export": { "Name": "TimeWaited" }
        },
        "WaiterId": {
            "Value": { "Fn::GetAtt": ["CFNWaiterCustomResource", "Id"] },
            "Export": { "Name": "WaiterId" }
        }
    }
}

```

YAML

```

AWSTemplateFormatVersion: "2010-09-09"
Resources:

```

```
LambdaExecutionRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Statement:
        - Effect: "Allow"
          Principal:
            Service:
              - "lambda.amazonaws.com"
          Action:
            - "sts:AssumeRole"
    Path: "/"
    Policies:
      - PolicyName: "AllowLogs"
        PolicyDocument:
          Statement:
            - Effect: "Allow"
              Action:
                - "logs:*"
              Resource: "*"
CFNWaiter:
  Type: AWS::Lambda::Function
  Properties:
    Handler: index.handler
    Runtime: python3.9
    Timeout: 900
    Role: !GetAtt LambdaExecutionRole.Arn
    Code:
      ZipFile:
        !Sub |
        from time import sleep
        import json
        import cfnresponse
        import uuid

        def handler(event, context):
            wait_seconds = 0
            id = str(uuid.uuid1())
            if event["RequestType"] in ["Create", "Update"]:
                wait_seconds = int(event["ResourceProperties"].get("ServiceTimeout", 0))
                sleep(wait_seconds)
            response = {
                "TimeWaited": wait_seconds,
                "Id": id
```

```

    }
    cfnresponse.send(event, context, cfnresponse.SUCCESS, response,
"Waiter-"+id)
  CFNWaiterCustomResource:
    Type: AWS::CloudFormation::CustomResource
    Properties:
      ServiceToken: !GetAtt CFNWaiter.Arn
      ServiceTimeout: 60
Outputs:
  TimeWaited:
    Value: !GetAtt CFNWaiterCustomResource.TimeWaited
    Export:
      Name: TimeWaited
  WaiterId:
    Value: !GetAtt CFNWaiterCustomResource.Id
    Export:
      Name: WaiterId

```

샘플 템플릿 연습

다음 코드 조각은 샘플 템플릿의 관련 부분을 설명하여 Lambda 함수를 사용자 지정 리소스와 연결하는 방법과 해당 출력을 이해하도록 도움이 됩니다.

[AWS::Lambda::Function](#) 리소스 CFNWaiter

`AWS::Lambda::Function` 리소스는 함수의 소스 코드, 핸들러 이름, 런타임 환경 및 실행 역할 Amazon 리소스 이름(ARN)을 지정합니다.

`Handler` 속성은 Python 소스 코드를 사용하기 때문에 `index.handler`로 설정됩니다. 인라인 함수 소스 코드를 사용할 때 허용되는 핸들러 식별자에 대한 자세한 내용은 [AWS::Lambda::Function Code](#)를 참조하세요.

`Runtime`은 소스 파일이 Python 코드이므로는 `python3.9`로 지정됩니다.

`Timeout`은 900초로 설정됩니다.

`Role` 속성은 `Fn::GetAtt` 함수를 사용하여 템플릿의 `AWS::IAM::Role` 리소스에 선언된 `LambdaExecutionRole` 실행 역할의 ARN을 가져옵니다.

`Code` 속성은 Python 함수를 사용하여 함수 코드를 인라인으로 정의합니다. 샘플 템플릿의 Python 함수는 다음을 수행합니다.

- UUID를 사용하여 고유 ID 생성

- 요청이 생성 요청인지 또는 업데이트 요청인지 확인
- Create 또는 Update 요청 중 ServiceTimeout에 지정된 기간만큼 대기
- 대기 시간 및 고유 ID 반환

JSON

```
...
    "CFNWaiter": {
      "Type": "AWS::Lambda::Function",
      "Properties": {
        "Handler": "index.handler",
        "Runtime": "python3.9",
        "Timeout": 900,
        "Role": { "Fn::GetAtt": ["LambdaExecutionRole", "Arn"] },
        "Code": {
          "ZipFile": { "Fn::Join": ["\n", [
            "from time import sleep",
            "import json",
            "import cfnresponse",
            "import uuid",
            "",
            "def handler(event, context):",
            "    wait_seconds = 0",
            "    id = str(uuid.uuid1())",
            "    if event[\"RequestType\"] in [\"Create\", \"Update\"]:",
            "        wait_seconds = int(event[\"ResourceProperties\"].get(\"ServiceTimeout", 0))",
            "        sleep(wait_seconds)",
            "        response = {",
            "            \"TimeWaited\": wait_seconds,",
            "            \"Id\": id ",
            "        }",
            "        cfnresponse.send(event, context, cfnresponse.SUCCESS, response,",
            "\"Waiter-\"+id)"
          ]]]
        }
      }
    },
    ...
```

YAML

```
...
CFNWaiter:
  Type: AWS::Lambda::Function
  Properties:
    Handler: index.handler
    Runtime: python3.9
    Timeout: 900
    Role: !GetAtt LambdaExecutionRole.Arn
    Code:
      ZipFile:
        !Sub |
        from time import sleep
        import json
        import cfnresponse
        import uuid

        def handler(event, context):
            wait_seconds = 0
            id = str(uuid.uuid1())
            if event["RequestType"] in ["Create", "Update"]:
                wait_seconds = int(event["ResourceProperties"].get("ServiceTimeout", 0))
                sleep(wait_seconds)
            response = {
                "TimeWaited": wait_seconds,
                "Id": id
            }
            cfnresponse.send(event, context, cfnresponse.SUCCESS, response,
"Waiter-"+id)
...

```

[AWS::IAM::Role](#) 리소스 LambdaExecutionRole

`AWS::IAM::Role` 리소스는 Lambda 함수의 실행 역할을 생성합니다. 여기에는 Lambda가 이를 사용할 수 있도록 허용하는 수임 역할 정책이 포함됩니다. 또한 CloudWatch Logs 액세스를 허용하는 정책도 포함됩니다.

JSON

```
...
  "LambdaExecutionRole": {

```

```

    "Type": "AWS::IAM::Role",
    "Properties": {
      "AssumeRolePolicyDocument": {
        "Statement": [{
          "Effect": "Allow",
          "Principal": { "Service": ["lambda.amazonaws.com"] },
          "Action": ["sts:AssumeRole"]
        }]
      },
      "Path": "/",
      "Policies": [{
        "PolicyName": "AllowLogs",
        "PolicyDocument": {
          "Statement": [{
            "Effect": "Allow",
            "Action": ["logs:*"],
            "Resource": "*"
          }]
        }
      }]
    }
  },
  ...

```

YAML

```

...
LambdaExecutionRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Statement:
        - Effect: "Allow"
          Principal:
            Service:
              - "lambda.amazonaws.com"
          Action:
            - "sts:AssumeRole"
    Path: "/"
    Policies:
      - PolicyName: "AllowLogs"
        PolicyDocument:
          Statement:

```

```

    - Effect: "Allow"
      Action:
        - "logs:*"
      Resource: "*"
  ...

```

[AWS::CloudFormation::CustomResource](#) 리소스 CFNWaiterCustomResource

사용자 지정 리소스는 !GetAtt CFNWaiter.Arn을 사용하여 ARN으로 Lambda 함수에 연결됩니다. ServiceTimeout에 설정된 대로 생성 및 업데이트 작업에 대해 60초의 대기 시간을 구현합니다. 속성이 수정된 경우에만 업데이트 작업에 대해서만 리소스가 간접 호출됩니다.

JSON

```

...
"CFNWaiterCustomResource": {
  "Type": "AWS::CloudFormation::CustomResource",
  "Properties": {
    "ServiceToken": { "Fn::GetAtt": ["CFNWaiter", "Arn"] },
    "ServiceTimeout": 60
  }
},
...

```

YAML

```

...
CFNWaiterCustomResource:
  Type: AWS::CloudFormation::CustomResource
  Properties:
    ServiceToken: !GetAtt CFNWaiter.Arn
    ServiceTimeout: 60
...

```

Outputs

이 템플릿의 Outputs은 TimeWaited 및 WaiterId입니다. TimeWaited 값은 Fn::GetAtt 함수를 사용하여 대기자 리소스가 실제로 대기한 시간을 제공합니다. WaiterId는 Fn::GetAtt 함수를 사용하여 생성되고 실행과 연결된 고유 ID를 제공합니다.

JSON

```
...
  "Outputs": {
    "TimeWaited": {
      "Value": { "Fn::GetAtt": ["CFNWaiterCustomResource", "TimeWaited"] },
      "Export": { "Name": "TimeWaited" }
    },
    "WaiterId": {
      "Value": { "Fn::GetAtt": ["CFNWaiterCustomResource", "Id"] },
      "Export": { "Name": "WaiterId" }
    }
  }
}
...
```

YAML

```
...
Outputs:
  TimeWaited:
    Value: !GetAtt CFNWaiterCustomResource.TimeWaited
    Export:
      Name: TimeWaited
  WaiterId:
    Value: !GetAtt CFNWaiterCustomResource.Id
    Export:
      Name: WaiterId
...
```

사전 조건

해당하는 서비스(예: Lambda 및 CloudFormation)를 모두 사용하려면 IAM 권한도 있어야 합니다.

스택 시작

스택을 생성하는 방법

1. [샘플 템플릿](#) 섹션에서 원하는 템플릿(YAML 또는 JSON)을 찾아 컴퓨터에 `samplelambdabackedcustomresource.template` 이름으로 저장하세요.
2. <https://console.aws.amazon.com/cloudformation/>에서 CloudFormation 콘솔을 엽니다.
3. 스택 페이지의 오른쪽 상단에서 스택 생성을 선택하고 새 리소스 사용(표준)을 선택합니다.

4. 사전 조건 - 템플릿 준비에서 기존 템플릿 선택을 선택합니다.
5. 템플릿 지정에서 템플릿 파일 업로드를 선택한 다음 파일 선택을 선택하세요.
6. 이전에 저장한 `samplelambdabackedcustomresource.template` 템플릿 파일을 선택하세요.
7. 다음을 선택합니다.
8. 스택 이름에 **SampleCustomResourceStack**을 입력하고 다음을 선택하세요.
9. 이 연습에서는 태그를 추가하거나 고급 설정을 지정할 필요가 없으므로 다음을 선택합니다.
10. 스택 이름이 올바른지 확인한 다음 업데이트를 선택하세요.

CloudFormation에서 스택을 생성하는 데 몇 분 정도 걸릴 수 있습니다. 진행 상황을 모니터링하려면 스택 이벤트를 확인합니다. 자세한 내용은 [CloudFormation 콘솔에서 스택 정보 보기](#) 섹션을 참조하세요.

스택 생성에 성공하면 스택의 모든 리소스(예: Lambda 함수 및 사용자 지정 리소스)가 생성됩니다. Lambda 함수 및 사용자 지정 리소스를 성공적으로 사용했습니다.

Lambda 함수가 오류를 반환하면 CloudWatch Logs [콘솔](#)에서 함수 로그를 확인합니다. 로그 스트림의 이름은 사용자 지정 리소스의 물리적 ID로, 스택의 리소스를 보면 확인할 수 있습니다. 자세한 내용을 알아보려면 Amazon CloudWatch 사용 설명서의 [로그 데이터 보기](#)를 참조하세요.

리소스 정리

스택을 삭제해 생성한 스택 리소스를 모두 정리하면 불필요한 리소스에 대해 비용이 청구되지 않습니다.

스택을 삭제하려면

1. CloudFormation 콘솔에서 SampleCustomResourceStack 스택을 선택하세요.
2. 작업을 선택한 다음 스택 삭제를 선택합니다.
3. 확인 메시지에서 예, 삭제를 선택합니다.

생성한 리소스가 모두 삭제됩니다.

Lambda 지원 사용자 지정 리소스를 생성하고 사용하는 방법을 이해했으므로 이 연습의 샘플 템플릿과 코드를 사용하여 기타 스택 및 함수를 빌드하고 이를 사용해 실험할 수 있습니다.

관련 정보

- [CloudFormation 사용자 지정 리소스 참조](#)

- [AWS::CloudFormation::CustomResource](#)

cfm-response 모듈

CloudFormation 템플릿에서는 사용자 지정 리소스의 대상으로 Lambda 함수를 지정할 수 있습니다. ZipFile 속성을 사용하여 [함수](#)의 소스 코드를 지정하는 경우 Lambda 함수에서 사용자 지정 리소스로 응답을 전송할 cfm-response 모듈을 로드할 수 있습니다. cfm-response 모듈은 Lambda 함수를 호출한 사용자 지정 리소스로의 응답 전송을 간소화하는 라이브러리입니다. 이 모듈에는 send 메서드가 있고, 이 메서드는 Amazon S3 미리 서명된 URL(ResponseURL)을 사용하여 [응답 객체](#)를 사용자 지정 리소스로 전송합니다.

cfm-response 모듈은 ZipFile 속성을 사용하여 소스 코드를 작성할 때만 사용할 수 있습니다. Amazon S3 버킷에 저장된 소스 코드에 사용할 수 없습니다. 버킷 내 코드의 경우, 응답을 전송할 고유 함수를 작성해야 합니다.

Note

send 메서드를 실행하고 나면 Lambda 함수가 종료하므로 해당 메서드 이후에 작성하는 모든 내용이 무시됩니다.

cfm-response 모듈 로드

Node.js 함수의 경우 require() 함수를 사용하여 cfm-response 모듈을 불러옵니다. 예를 들면 다음 코드 예제에서는 cfm-response 이름의 response 객체를 생성합니다.

```
var response = require('cfm-response');
```

Python의 경우 다음 예제에서와 같이 import 문을 사용하여 cfmresponse 모듈을 로드합니다.

Note

이 정확한 가져오기 문을 사용합니다. 가져오기 문의 다양한 변형을 사용하는 경우 CloudFormation에서 응답 모듈을 포함하지 않습니다.

```
import cfmresponse
```

send 메서드 파라미터

send 메서드와 함께 다음 파라미터를 사용할 수 있습니다.

event

[사용자 지정 리소스 요청](#)의 필드입니다.

context

함수 및 콜백이 Lambda 실행 환경 내에서 정보에 액세스했거나 실행을 완료한 경우를 지정하는 데 사용할 수 있는 Lambda 함수 관련 객체입니다. 자세한 내용은 AWS Lambda 개발자 안내서의 [Node.js를 사용하여 Lambda 함수 빌드](#)를 참조하세요.

responseStatus

함수가 성공적으로 완료되었는지 여부를 나타냅니다. cfntemplate 모듈 제약을 사용하여 실행 성공에 대해 SUCCESS를 지정하고 실행 실패에 대해 FAILED를 지정합니다.

responseData

사용자 지정 [응답 객체](#)의 Data 필드입니다. 데이터는 이름-값 페어 목록입니다.

physicalResourceId

선택 사항. 함수를 호출한 사용자 지정 리소스의 고유 식별자입니다. 기본적으로 이 모듈은 Lambda 함수와 연결된 Amazon CloudWatch Logs 로그 스트림의 이름을 사용합니다.

PhysicalResourceId에 대해 반환된 값은 사용자 지정 리소스 업데이트 작업을 변경할 수 있습니다. 반환된 값이 같은 경우 일반 업데이트로 간주됩니다. 반환된 값이 다른 경우 CloudFormation은 업데이트를 교체로 인식하고 기존 리소스에 삭제 요청을 전송합니다. 자세한 내용은 [AWS::CloudFormation::CustomResource](#) 섹션을 참조하세요.

noEcho

선택 사항. Fn::GetAtt 함수를 사용하여 조회할 때 사용자 지정 리소스의 출력을 마스킹할지 여부를 나타냅니다. true로 설정하면 아래 지정된 위치에 저장된 정보를 제외하고, 반환된 모든 값은 별표(*****)로 마스킹됩니다. 기본적으로 이 값은 false입니다.

Important

NoEcho 속성을 사용해도 다음에 저장된 정보는 마스킹되지 않습니다.

- Metadata 템플릿 섹션. CloudFormation은 Metadata 섹션에 포함된 정보를 변환, 수정 또는 삭제하지 않습니다. 자세한 내용은 [메타데이터](#)를 참조하십시오.
- Outputs 템플릿 섹션. 자세한 내용은 [출력](#)을 참조하십시오.

- 리소스 정의의 Metadata 속성입니다. 자세한 내용은 [Metadata 속성](#)을 참조하세요. 이러한 메커니즘을 사용하여 암호나 보안 정보와 같은 중요한 정보를 포함하지 않는 것이 좋습니다.

NoEcho를 사용하여 민감한 정보를 마스킹 처리하는 방법에 대한 자세한 내용은 [템플릿에 자격 증명을 포함하지 않음](#) 모범 사례를 참조하세요.

예제

Node.js

다음 Node.js 예제에서 인라인 Lambda 함수는 입력 값을 받아서 그 값에 5를 곱합니다. 인라인 함수를 사용하면 패키지를 생성하여 Amazon S3 버킷에 업로드하는 대신에 템플릿에서 소스 코드를 직접 지정할 수 있으므로 인라인 함수는 작은 함수에 특히 유용합니다. 이 함수는 cfn-response send 메서드를 사용하여 호출했던 사용자 지정 리소스로 결과를 다시 전송합니다.

JSON

```
"ZipFile": { "Fn::Join": ["", [
  "var response = require('cfn-response');",
  "exports.handler = function(event, context) {",
  "  var input = parseInt(event.ResourceProperties.Input);",
  "  var responseData = {Value: input * 5};",
  "  response.send(event, context, response.SUCCESS, responseData);",
  "};",
]] }
```

YAML

```
ZipFile: >
  var response = require('cfn-response');
  exports.handler = function(event, context) {
    var input = parseInt(event.ResourceProperties.Input);
    var responseData = {Value: input * 5};
    response.send(event, context, response.SUCCESS, responseData);
  };
```

Python

다음 Python 예제에서 인라인 Lambda 함수는 정수 값을 받아서 그 값에 5를 곱합니다.

JSON

```
"ZipFile" : { "Fn::Join" : ["\n", [  
    "import json",  
    "import cfnresponse",  
    "def handler(event, context):",  
    "    responseValue = int(event['ResourceProperties']['Input']) * 5",  
    "    responseData = {}",  
    "    responseData['Data'] = responseValue",  
    "    cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,  
    \"CustomResourcePhysicalID\")",  
    ]}] }
```

YAML

```
ZipFile: |  
    import json  
    import cfnresponse  
    def handler(event, context):  
        responseValue = int(event['ResourceProperties']['Input']) * 5  
        responseData = {}  
        responseData['Data'] = responseValue  
        cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,  
        "CustomResourcePhysicalID")
```

모듈 소스 코드

주제

- [비동기 Node.js 소스 코드](#)
- [Node.js 소스 코드](#)
- [Python 소스 코드](#)

비동기 Node.js 소스 코드

다음은 핸들러가 비동기식인 경우의 Node.js 함수에 대한 응답 모듈 소스 코드입니다. 이 코드를 검토하면 모듈의 기능을 이해할 수 있으며 고유의 응답 함수를 구현하는 데 도움이 됩니다.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.  
// SPDX-License-Identifier: MIT-0
```

```
exports.SUCCESS = "SUCCESS";
exports.FAILED = "FAILED";

exports.send = function(event, context, responseStatus, responseData,
    physicalResourceId, noEcho) {

    return new Promise((resolve, reject) => {
        var responseBody = JSON.stringify({
            Status: responseStatus,
            Reason: "See the details in CloudWatch Log Stream: " +
context.logStreamName,
            PhysicalResourceId: physicalResourceId || context.logStreamName,
            StackId: event.StackId,
            RequestId: event.RequestId,
            LogicalResourceId: event.LogicalResourceId,
            NoEcho: noEcho || false,
            Data: responseData
        });

        console.log("Response body:\n", responseBody);

        var https = require("https");
        var url = require("url");

        var parsedUrl = url.parse(event.ResponseURL);
        var options = {
            hostname: parsedUrl.hostname,
            port: 443,
            path: parsedUrl.path,
            method: "PUT",
            headers: {
                "content-type": "",
                "content-length": responseBody.length
            }
        };

        var request = https.request(options, function(response) {
            console.log("Status code: " + parseInt(response.statusCode));
            resolve(context.done());
        });

        request.on("error", function(error) {
            console.log("send(..) failed executing https.request(..): " +
maskCredentialsAndSignature(error));
        });
    });
};
```

```

        reject(context.done(error));
    });

    request.write(responseBody);
    request.end();
  })
}

function maskCredentialsAndSignature(message) {
  return message.replace(/X-Amz-Credential=[^&\s]+/i, 'X-Amz-Credential=*****')
    .replace(/X-Amz-Signature=[^&\s]+/i, 'X-Amz-Signature=*****');
}

```

Node.js 소스 코드

다음은 핸들러가 비동기식이 아닌 경우의 Node.js 함수에 대한 응답 모듈 소스 코드입니다. 이 코드를 검토하면 모듈의 기능을 이해할 수 있으며 고유의 응답 함수를 구현하는 데 도움이 됩니다.

```

// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: MIT-0

exports.SUCCESS = "SUCCESS";
exports.FAILED = "FAILED";

exports.send = function(event, context, responseStatus, responseData,
  physicalResourceId, noEcho) {

  var responseBody = JSON.stringify({
    Status: responseStatus,
    Reason: "See the details in CloudWatch Log Stream: " + context.logStreamName,
    PhysicalResourceId: physicalResourceId || context.logStreamName,
    StackId: event.StackId,
    RequestId: event.RequestId,
    LogicalResourceId: event.LogicalResourceId,
    NoEcho: noEcho || false,
    Data: responseData
  });

  console.log("Response body:\n", responseBody);

  var https = require("https");
  var url = require("url");

```

```

var parsedUrl = url.parse(event.ResponseURL);
var options = {
    hostname: parsedUrl.hostname,
    port: 443,
    path: parsedUrl.path,
    method: "PUT",
    headers: {
        "content-type": "",
        "content-length": responseBody.length
    }
};

var request = https.request(options, function(response) {
    console.log("Status code: " + parseInt(response.statusCode));
    context.done();
});

request.on("error", function(error) {
    console.log("send(..) failed executing https.request(..): " +
maskCredentialsAndSignature(error));
    context.done();
});

request.write(responseBody);
request.end();
}

```

Python 소스 코드

다음은 Python 함수에 대한 응답 모듈 소스 코드입니다.

```

# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: MIT-0

from __future__ import print_function
import urllib3
import json
import re

SUCCESS = "SUCCESS"
FAILED = "FAILED"

http = urllib3.PoolManager()

```

```
def send(event, context, responseStatus, responseData, physicalResourceId=None,
noEcho=False, reason=None):
    responseUrl = event['ResponseURL']

    responseBody = {
        'Status' : responseStatus,
        'Reason' : reason or "See the details in CloudWatch Log Stream:
{}".format(context.log_stream_name),
        'PhysicalResourceId' : physicalResourceId or context.log_stream_name,
        'StackId' : event['StackId'],
        'RequestId' : event['RequestId'],
        'LogicalResourceId' : event['LogicalResourceId'],
        'NoEcho' : noEcho,
        'Data' : responseData
    }

    json_responseBody = json.dumps(responseBody)

    print("Response body:")
    print(json_responseBody)

    headers = {
        'content-type' : '',
        'content-length' : str(len(json_responseBody))
    }

    try:
        response = http.request('PUT', responseUrl, headers=headers,
body=json_responseBody)
        print("Status code:", response.status)

    except Exception as e:

        print("send(..) failed executing http.request(..):",
mask_credentials_and_signature(e))

def mask_credentials_and_signature(message):
    message = re.sub(r'X-Amz-Credential=[^\s]+', 'X-Amz-Credential=*****', message,
flags=re.IGNORECASE)
```

```
return re.sub(r'X-Amz-Signature=[^&\s]+', 'X-Amz-Signature=*****', message,
flags=re.IGNORECASE)
```

템플릿 매크로를 사용하여 CloudFormation 템플릿에서 사용자 지정 처리 수행

매크로를 사용하면 찾기 및 바꾸기 작업 같은 간단한 작업부터 전체 템플릿 변형과 같은 작업까지 템플릿에 대한 사용자 지정 처리를 수행할 수 있습니다.

가능성의 폭에 대한 아이디어를 얻으려면 `AWS::Include` 및 `AWS::Serverless` 변형을 고려하세요. CloudFormation에 의해 호스팅되는 매크로입니다.

- [AWS::Include 변환](#)을 사용하면 보일러플레이트 템플릿 코드 조각을 템플릿에 삽입할 수 있습니다.
- [AWS::Serverless 변환](#)에서는 AWS Serverless Application Model(AWS SAM) 구문에서 작성된 전체 템플릿을 가져온 다음 이를 변형 후 규정 준수 CloudFormation 템플릿으로 확장합니다. 서버리스 애플리케이션 및 AWS SAM에 대한 자세한 내용은 [AWS Serverless Application Model 개발자 안내서](#)를 참조하세요.

주제

- [결제](#)
- [매크로 예제](#)
- [관련 리소스](#)
- [CloudFormation 매크로 개요](#)
- [CloudFormation 매크로 정의 생성](#)
- [간단한 문자열 교체 매크로 예제](#)
- [처리된 템플릿 문제 해결](#)

결제

매크로가 실행 중일 때 Lambda 함수의 소유자에게 해당 함수의 실행과 관련한 요금이 청구됩니다.

`AWS::Include` 및 `AWS::Serverless` 변환은 CloudFormation에서 호스팅하는 매크로입니다. 사용에 대해 부과되는 요금은 없습니다.

매크로 예제

이 섹션의 예제에 추가로 [GitHub 리포지토리](#)에서 소스 코드 및 템플릿을 포함한 예제 매크로를 확인할 수 있습니다. 이러한 예제는 교육용으로 있는 그대로 제공됩니다.

관련 리소스

- [AWS::CloudFormation::Macro](#)
- [CloudFormation 템플릿 Transform 섹션](#)
- [Fn::Transform](#)
- [AWS::Serverless 변환](#)
- [AWS::Include 변환](#)

CloudFormation 매크로 개요

매크로를 사용한 템플릿 처리에 대한 두 가지 주요 단계가 있습니다. 매크로 자체의 생성과 매크로 사용을 통한 템플릿의 처리 수행입니다.

매크로 정의를 생성하려면 다음을 생성해야 합니다.

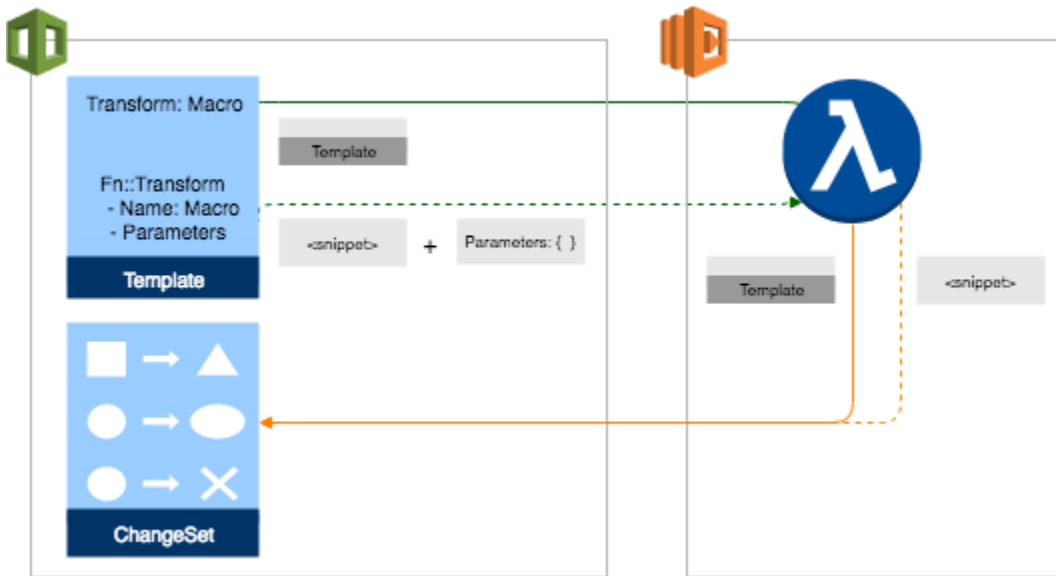
- 템플릿 처리를 수행하는 Lambda 함수. 이 Lambda 함수는 코드 조각 또는 전체 템플릿 및 정의하는 추가 파라미터를 수락합니다. 응답으로 처리된 템플릿 코드 조각 또는 전체 템플릿을 반환합니다.
- [AWS::CloudFormation::Macro](#) 유형의 리소스로, 이를 통해 사용자는 CloudFormation 템플릿 내부에서 Lambda 함수를 직접적으로 호출할 수 있습니다. 이 리소스는 Lambda 함수의 ARN을 지정하여 이 매크로를 호출하고, 추가 선택적 속성을 지정하여 디버깅을 지원합니다. 계정 내에서 이 리소스를 생성하려면 `AWS::CloudFormation::Macro` 리소스를 포함하는 템플릿을 작성한 다음 템플릿에서 자체 관리 권한이 있는 스택 또는 스택 세트를 생성합니다. CloudFormation StackSets는 현재 매크로를 참조하는 템플릿에서 서비스 관리형 권한으로 스택 세트를 생성하거나 업데이트하는 것을 지원하지 않습니다.

매크로를 사용하려면 템플릿에서 매크로를 참조합니다.

- 템플릿의 섹션 또는 일부를 처리하려면 변환하려는 템플릿과 관련된 위치에 있는 `Fn::Transform` 함수의 매크로를 참조합니다. `Fn::Transform` 사용 시 필요한 지정된 파라미터를 전달할 수 있습니다.
- 전체 템플릿을 처리하려면 템플릿의 [Transform](#) 섹션에 있는 매크로를 참조합니다.

그런 다음 변경 세트를 생성하고 실행하십시오. (매크로 처리 시 알지 못할 수도 있는 여러 리소스가 추가될 수 있습니다. 매크로가 수행하는 모든 변경을 인식하려면 변경 세트를 사용합니다.) CloudFormation은 추가로 지정된 파라미터와 함께 지정된 템플릿 콘텐츠를 매크로 리소스에 지정된 Lambda 함수에 전달합니다. Lambda 함수는 처리된 템플릿 콘텐츠를 반환하며, 이는 코드 조각 또는 전체 템플릿입니다.

템플릿의 모든 매크로가 직접적으로 호출된 이후 CloudFormation에서 처리된 템플릿 콘텐츠를 포함하는 변경 세트를 생성합니다. 변경 세트를 검토한 후 실행하여 해당 변경 내용을 적용합니다.



스택을 직접 생성하는 방법

매크로를 참조하는 템플릿을 사용하여 스택을 생성하거나 업데이트하려면 일반적으로 변경 세트를 생성하고 나서 실행하십시오. 변경 세트는 처리된 템플릿에 기초하여 CloudFormation에서 수행할 작업을 설명합니다. 매크로 처리 시 알지 못할 수도 있는 여러 리소스가 추가될 수 있습니다. 매크로가 수행하는 모든 변경을 항상 파악하려면 변경 세트를 사용합니다. 변경 세트를 검토한 후 실행하여 변경을 적용할 수 있습니다.

매크로는 IAM 리소스를 템플릿에 추가할 수 있습니다. 이러한 리소스의 경우 CloudFormation에서는 [해당 기능을 승인](#)해야 합니다. 템플릿을 처리하기 전에는 CloudFormation에서 추가되는 리소스를 알 수 없으므로 참조되는 매크로가 IAM 리소스를 포함하는지 여부에 따라 변경 세트를 생성할 때 IAM 기능을 승인해야 할 수도 있습니다. 즉, 변경 세트를 실행할 경우 CloudFormation은 IAM 리소스 생성 시 필요한 기능을 갖추게 됩니다.

변경 세트에 제시된 변경을 먼저 검토하지 않고 처리된 템플릿에서 직접 스택을 생성하거나 업데이트하려면 CreateStack 또는 UpdateStack 요청 중 CAPABILITY_AUTO_EXPAND 기능을 지정합니다. 매크로가 수행하는 처리를 알고 있을 경우에만 매크로를 포함하는 스택 템플릿에서 직접 스택을 생성

해야 합니다. 스택 세트 매크로에는 변경 세트를 사용할 수 없습니다. 스택 세트를 직접 업데이트해야 합니다.

자세한 내용은 AWS CloudFormation API 참조의 [CreateStack](#) 또는 [UpdateStack](#) 섹션을 참조하세요.

Important

스택 세트 템플릿에서 하나 이상의 매크로를 참조하는 경우 먼저 변경 세트의 결과 변경 사항을 먼저 검토하지 않고 처리된 템플릿에서 직접 스택 세트를 생성해야 합니다. 매크로 처리 시 알지 못할 수도 있는 여러 리소스가 추가될 수 있습니다. 매크로를 직접 참조하는 템플릿에서 스택 세트를 생성 또는 업데이트하기 전에 매크로가 수행하는 처리 작업을 알고 있어야 합니다.

매크로를 참조하는 템플릿에서 스택 시작에 필요한 단계 수를 줄이려면 package 및 deploy AWS CLI 명령을 사용합니다. 자세한 내용은 [AWS CLI를 사용하여 S3 버킷에 로컬 아티팩트 업로드 및 변환이 포함된 스택 생성](#)(을)를 참조하세요.

고려 사항

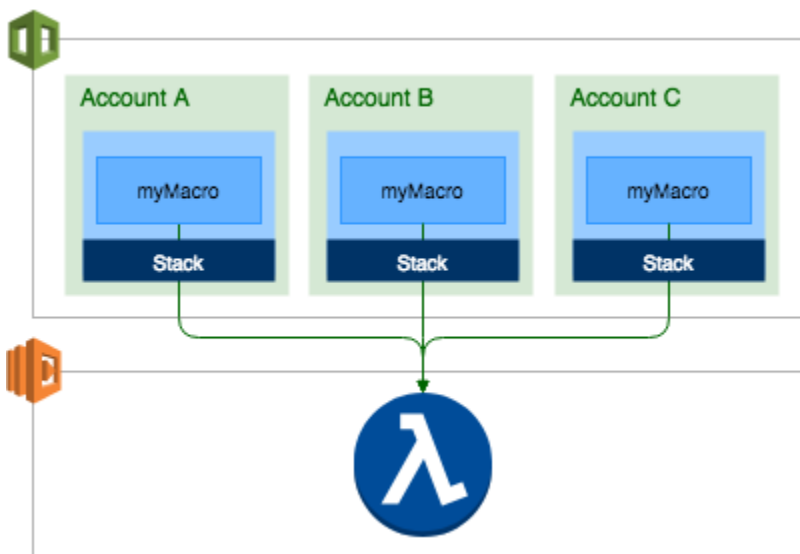
매크로를 사용하는 경우 다음 고려 사항 및 제한 사항에 유의하세요.

- 매크로는 Lambda 사용이 가능한 AWS 리전에서만 지원됩니다. Lambda를 사용할 수 있는 리전 목록을 알아보려면 [AWS Lambda 엔드포인트 및 할당량](#)을 참조하세요.
- 처리된 템플릿 조각은 유효한 JSON이어야 합니다.
- 모든 처리된 템플릿 조각은 스택 생성, 스택 업데이트, 스택 세트 생성 또는 스택 세트 업데이트 작업에 대한 검증 확인을 전달해야 합니다.
- CloudFormation은 매크로를 먼저 확인하고 템플릿을 처리합니다. 결과 템플릿은 유효한 JSON 또는 이어야 하며 템플릿 크기 제한을 초과해서는 안 됩니다.
- CloudFormation이 템플릿에서 요소를 처리하는 순서가 있기 때문에 매크로는 CloudFormation에 반환하는 처리된 템플릿 콘텐츠에 모듈을 포함할 수 없습니다. 자세한 내용은 [매크로 평가 순서](#) 섹션을 참조하세요.
- 업데이트 롤백 기능을 사용하면 CloudFormation은 원본 템플릿의 사본을 사용합니다. 포함된 코드 조각이 변경된 경우에도 원본 템플릿으로 롤백됩니다.
- 매크로는 재귀적으로 처리되지 않기 때문에 매크로 내에 매크로를 포함시키면 작동하지 않습니다.
- 현재 매크로에서 Fn::ImportValue 내장 함수를 지원하지 않습니다.

- 템플릿에 포함된 내장 함수는 모든 매크로 이후 평가됩니다. 따라서 매크로가 반환하는 처리된 템플릿 콘텐츠에는 내장 함수에 대한 호출이 포함될 수 있고, 이는 평소와 같이 평가됩니다.
- StackSets는 현재 CloudFormation 매크로를 참조하는 템플릿에서 서비스 관리형 권한으로 스택 세트를 생성하거나 업데이트하는 것을 지원하지 않습니다.

매크로 계정 범위 및 권한

리소스로 생성된 계정에서만 매크로를 사용할 수 있습니다. 매크로의 이름은 해당 계정 내에서 고유해야 합니다. 하지만 기본 Lambda 함수에서 교차 계정 액세스를 활성화하고 여러 계정에서 해당 함수를 참조하는 매크로 정의를 생성함으로써 여러 계정에서 동일한 기능을 사용할 수 있도록 할 수 있습니다. 아래 예제에서는 3개의 계정에 동일한 Lambda 함수를 가리키는 매크로 정의가 포함되어 있습니다.



매크로 정의를 생성하려면 사용자에게 지정된 계정 내에서 스택 또는 스택 세트를 생성할 수 있는 권한이 있어야 합니다.

CloudFormation에서 템플릿에 포함된 매크로를 실행하려면 사용자가 기본 Lambda 함수에 대해 Invoke 권한을 보유해야 합니다. 잠재적인 권한 상승을 방지하도록 CloudFormation에서는 매크로 실행 도중 사용자를 가장합니다.

자세한 내용은 AWS Lambda 개발자 안내서의 [AWS Lambda에서 권한 관리](#) 및 서비스 승인 참조의 [Actions, resources, and condition keys for AWS Lambda](#)를 참조하세요.

CloudFormation 매크로 정의 생성

매크로 정의를 생성할 때 매크로 정의에서 지정된 계정의 기본 Lambda 함수를 사용할 수 있도록 하여 CloudFormation에서 이를 간접적으로 호출하고 템플릿을 처리합니다.

이벤트 매핑

CloudFormation은 매크로의 Lambda 함수를 간접적으로 호출할 때 다음 구조의 JSON 형식으로 요청을 전송합니다.

```
{
  "region" : "us-east-1",
  "accountId" : "$ACCOUNT_ID",
  "fragment" : { ... },
  "transformId" : "$TRANSFORM_ID",
  "params" : { ... },
  "requestId" : "$REQUEST_ID",
  "templateParameterValues" : { ... }
}
```

- region

매크로가 상주하는 리전입니다.

- accountId

매크로가 Lambda 함수를 호출하는 계정의 계정 ID입니다.

- fragment

사용자 지정 처리 사용이 가능한 템플릿 콘텐츠로 JSON 형식입니다.

- Transform 템플릿 섹션에 포함된 매크로의 경우 Transform 섹션을 제외한 전체 템플릿입니다.
- Fn::Transform 내장 함수 호출에 포함된 매크로의 경우 템플릿 내에 있는 내장 함수의 위치를 기반으로 한 모든 형제(및 하위) 노드가 포함됩니다. 단, Fn::Transform 함수는 예외입니다. 자세한 내용은 [매크로 템플릿 범위](#) 섹션을 참조하세요.

- transformId

이 함수를 호출하는 매크로의 이름입니다.

- params

Fn::Transform 함수 직접 호출의 경우 함수에 대한 지정된 파라미터입니다. CloudFormation은 이러한 파라미터를 함수에 전달하기 전에 평가하지 않습니다.

Transform 템플릿 섹션에 포함된 매크로의 경우 이 섹션은 비어 있습니다.

- requestId

이 함수를 호출하는 요청의 ID입니다.

- `templateParameterValues`

템플릿의 [Parameters](#) 섹션에 지정된 파라미터입니다. CloudFormation은 이러한 파라미터를 함수에 전달하기 전에 평가합니다.

응답 형식

CloudFormation에서는 Lambda 함수가 다음 JSON 형식의 응답을 반환할 것으로 예상합니다.

```
{
  "requestId" : "$REQUEST_ID",
  "status" : "$STATUS",
  "fragment" : { ... },
  "errorMessage": "optional error message for failures"
}
```

- `requestId`

이 함수를 호출하는 요청의 ID입니다. 함수 호출 시 CloudFormation에서 제공하는 요청 ID와 일치해야 합니다.

- `status`

요청의 상태입니다(대/소문자를 구분하지 않음). `success`로 설정되어야 합니다. CloudFormation에서 다른 모든 응답을 실패로 처리합니다.

- `fragment`

형제 템플릿을 포함하여 처리된 템플릿에 포함할 CloudFormation의 처리된 템플릿 콘텐츠입니다. CloudFormation이 Lambda 함수에 전달된 템플릿 콘텐츠를 Lambda 응답에서 수신하는 템플릿 조각으로 대체합니다.

처리된 템플릿 콘텐츠는 유효한 JSON이어야 하고, 처리된 템플릿에 포함된 사항은 유효한 템플릿이 되어야 합니다.

함수가 CloudFormation에서 전달하는 템플릿 콘텐츠를 실제로 변경하지 않지만 이러한 콘텐츠를 처리된 템플릿에 포함해야 하는 경우 함수가 응답에서 해당 템플릿 콘텐츠를 CloudFormation으로 반환해야 합니다.

- `errorMessage`

변환이 실패한 이유를 설명하는 오류 메시지입니다. CloudFormation은 스택에 대한 Stack details(스택 세부 정보) 페이지의 Events(이벤트) 창에 이 오류 메시지를 표시합니다.

예제:

```
Error creating change set: Transform
    AWS ## account
    number::macro name failed with:
    error message string.
```

매크로 정의 생성

CloudFormation 매크로 정의를 생성하려면 다음을 수행하세요.

1. 템플릿 콘텐츠 처리를 처리할 [Lambda 함수를 빌드](#)합니다. 함수가 템플릿의 일부분부터 전체 템플릿까지 처리할 수 있습니다.
2. AWS::CloudFormation::Macro 리소스 유형이 포함된 CloudFormation 템플릿을 생성하고 Name 및 FunctionName 속성을 지정합니다. FunctionName 속성은 CloudFormation에서 매크로 실행 시 간접적으로 호출할 Lambda 함수의 ARN을 포함해야 합니다.
3. (선택 사항) 디버깅을 돕기 위해 매크로에 대한 AWS::CloudFormation::Macro 리소스 유형 생성 시 LogGroupName 및 LogRoleArn 속성을 지정할 수 있습니다. 이러한 속성을 통해 매크로의 기본 Lambda 함수를 간접적으로 호출할 때 CloudFormation에서 오류 로깅 정보를 전송하는 CloudWatch Logs 로그 그룹을 지정할 수 있으며, 로그 항목을 로그로 전송할 때 CloudFormation에서 수입해야 하는 역할을 지정할 수 있습니다.
4. 스택을 사용하려는 계정에서 매크로가 포함된 템플릿을 사용하여 [스택을 생성](#)합니다. 또는 관리자 계정에서 매크로가 포함된 템플릿을 사용하여 [자체 관리형 권한으로 스택 세트를 생성](#)한 다음 대상 계정에서 스택 인스턴스를 생성합니다.
5. CloudFormation에서 매크로 정의가 포함된 스택을 생성한 이후 매크로는 해당 계정 내에서 사용 가능합니다. 처리하고자 하는 템플릿 콘텐츠와 관련이 있는 적절한 위치에서 템플릿을 참조하여 매크로를 사용합니다.

매크로 템플릿 범위

템플릿의 Transform 섹션에서 참조되는 매크로는 해당 템플릿의 전체 콘텐츠를 처리할 수 있습니다.

`Fn::Transform` 함수에서 참조되는 매크로는 템플릿의 `Fn::Transform` 함수에 대한 형제 요소(하위 요소 포함)의 콘텐츠를 처리할 수 있습니다.

예를 들어 아래 템플릿 샘플에서 `AWS::Include`는 `Fn::Transform` 함수의 위치를 기준으로 `MyBucket` 속성을 모두 처리할 수 있습니다. `MyMacro`는 `Transform` 섹션에 포함되어 있으므로 전체 템플릿의 콘텐츠를 처리할 수 있습니다.

```
# Start of processable content for MyMacro
AWSTemplateFormatVersion: 2010-09-09
Transform: [MyMacro]
Resources:
  WaitCondition:
    Type: AWS::CloudFormation::WaitCondition
  MyBucket:
    Type: AWS::S3::Bucket
    # Start of processable content for AWS::Include
    Properties:
      BucketName: amzn-s3-demo-bucket1
      Tags: [{"key": "value"}]
      'Fn::Transform':
        - Name: 'AWS::Include'
          Parameters:
            Location: s3://amzn-s3-demo-bucket2/MyFileName.yaml
      CorsConfiguration: []
    # End of processable content for AWS::Include
  MyEc2Instance:
    Type: AWS::EC2::Instance
    Properties:
      ImageID: ami-1234567890abcdef0
# End of processable content for MyMacro
```

매크로 평가 순서

[AWS::Include](#) 및 [AWS::Serverless](#) 등과 같이 CloudFormation에서 호스팅되는 변형을 포함하여 해당 템플릿의 여러 매크로를 참조할 수 있습니다.

매크로는 템플릿에서의 위치를 기준으로 가장 깊이 중첩된 외부부터 가장 일반적인 순서로 평가됩니다. 템플릿의 동일한 위치에 있는 매크로는 나열된 순서를 기준으로 순차적으로 평가됩니다.

`AWS::Include` 및 `AWS::Transform`과 같은 변형은 작업 순서 및 범위 측면에서 다른 매크로와 동일하게 취급됩니다.

예를 들어, 아래의 템플릿 샘플에서 CloudFormation은 템플릿에서 가장 깊이 중첩된 매크로인 PolicyAdder 매크로를 먼저 평가합니다. 그런 다음 CloudFormation은 Transform 섹션에서 AWS::Serverless보다 먼저 나열되어 있기 때문에 MyMacro를 AWS::Serverless보다 먼저 평가합니다.

```
AWSTemplateFormatVersion: 2010-09-09
Transform: [MyMacro, AWS::Serverless]
Resources:
  WaitCondition:
    Type: AWS::CloudFormation::WaitCondition
  MyBucket:
    Type: AWS::S3::Bucket
    Properties:
      BucketName: amzn-s3-demo-bucket
      Tags: [{"key": "value"}]
      'Fn::Transform':
        - Name: PolicyAdder
      CorsConfiguration: []
  MyEc2Instance:
    Type: AWS::EC2::Instance
    Properties:
      ImageID: ami-1234567890abcdef0
```

간단한 문자열 교체 매크로 예제

다음 예제는 템플릿에서 매크로를 정의하는 것부터 매크로에 대한 Lambda 함수를 생성하고 템플릿에 있는 매크로를 사용하는 것에 이르기까지 매크로 사용 프로세스를 연습합니다.

이 예제에서 지정된 문자열을 처리된 콘텐츠의 지정된 대상 콘텐츠를 대신하여 삽입하는 단순한 매크로를 생성합니다. 또한 이를 사용하여 처리된 템플릿의 지정된 위치에 빈 WaitHandleCondition을 삽입합니다.

매크로 생성

매크로를 사용하기 전에 2가지를 완료해야 합니다. 원하는 템플릿 처리를 수행하는 Lambda 함수를 생성하고, 매크로 정의를 생성함으로써 CloudFormation에서 Lambda 함수를 사용할 수 있도록 합니다.

다음 샘플 템플릿에는 예제 매크로에 대한 정의가 포함되어 있습니다. 특정 AWS 계정에서 매크로를 사용할 수 있도록 하려면 템플릿에서 스택을 생성합니다. 매크로 정의는 매크로 이름, 간단한 설명을 지정하고, 이 매크로가 템플릿에서 사용될 때 CloudFormation에서 호출하는 Lambda 함수의 ARN을 참조합니다. (오류 로깅을 위해 LogGroupName 또는 LogRoleARN 속성을 포함하지 않았습니다.)

이 예제에서는 이 템플릿에서 생성된 스택의 이름이 JavaMacroFunc라고 가정합니다. 매크로 Name 속성이 스택 이름으로 설정되었기 때문에 결과로 나오는 매크로 이름은 JavaMacroFunc입니다.

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  Macro:
    Type: AWS::CloudFormation::Macro
    Properties:
      Name: !Sub '${AWS::StackName}'
      Description: Adds a blank WaitConditionHandle named WaitHandle
      FunctionName: 'arn:aws:lambda:us-east-1:012345678910:function:JavaMacroFunc'
```

매크로 사용

매크로를 사용하기 위해 Fn::Transform 내장 함수를 사용하여 이를 템플릿에 포함합니다.

아래 템플릿을 사용하여 스택을 생성할 때 CloudFormation에서 예제 매크로를 호출합니다. 기본 Lambda 함수는 지정된 문자열을 다른 지정된 문자열로 바꿉니다. 이 경우에는 결과적으로 빈 AWS::CloudFormation::WaitConditionHandle이 처리된 템플릿으로 삽입됩니다.

```
Parameters:
  ExampleParameter:
    Type: String
    Default: 'SampleMacro'

Resources:
  2a:
    Fn::Transform:
      Name: "JavaMacroFunc"
      Parameters:
        replacement: 'AWS::CloudFormation::WaitConditionHandle'
        target: '$$REPLACEMENT$$'
      Type: '$$REPLACEMENT$$'
```

- 간접적으로 호출할 매크로는 이전 매크로 정의 예제에서 가져온 JavaMacroFunc로 지정됩니다.
- 매크로는 전달되는 2개의 파라미터로, target 및 replacement입니다. 대상 문자열과 원하는 대체 값을 나타냅니다.
- 매크로는 Type 노드의 콘텐츠를 처리할 수 있는데, Type이 매크로를 참조하는 Fn::Transform 함수의 형제이기 때문입니다.
- 결과 AWS::CloudFormation::WaitConditionHandle의 이름은 2a입니다.

- 템플릿에는 또한 템플릿 파라미터인 `ExampleParameter`가 포함되어 있습니다. 매크로에서 여기에 액세스할 수 있습니다(단, 이 경우 사용하지 않음).

Lambda 입력 데이터

CloudFormation에서 스택 생성 도중 예제 템플릿을 처리할 때 다음 이벤트 매핑을 `JavaMacroFunc` 매크로 정의에서 참조되는 Lambda 함수에 전달합니다.

- `region`: `us-east-1`
- `accountId`: `012345678910`
- `fragment`:

```
{
  "Type": "$$REPLACEMENT$$"
}
```

- `transformId`: `012345678910::JavaMacroFunc`
- `params`:

```
{
  "replacement": "AWS::CloudFormation::WaitConditionHandle",
  "target": "$$REPLACEMENT$$"
}
```

- `requestId`: `5dba79b5-f117-4de0-9ce4-d40363bfb6ab`
- `templateParameterValues`:

```
{
  "ExampleParameter": "SampleMacro"
}
```

`fragment`에는 매크로에서 처리할 수 있는 템플릿 조각을 나타내는 JSON이 포함되어 있습니다. 이 조각에는 `Fn::Transform` 함수 호출의 형제가 포함되어 있지만 함수 호출 자체가 포함되어 있지는 않습니다. 또한 `params`에는 매크로 파라미터를 나타내는 JSON이 포함되어 있습니다. 이 경우 대체 및 대상입니다. 비슷하게 `templateParameterValues`에는 전체적으로 템플릿에 지정된 파라미터를 나타내는 JSON이 포함되어 있습니다.

Lambda 함수 코드

다음은 기본 JavaMacroFunc 예제 매크로의 Lambda 함수에 대한 실제 코드입니다. 응답에 포함된 템플릿 조각(문자열, 목록 또는 맵 형식)을 반복하여 지정된 대상 문자열을 찾습니다. 지정된 대상 문자열을 찾으면 Lambda 함수에서 대상 문자열을 지정된 대체 문자열로 바꿉니다. 그렇지 않은 경우 함수에서 템플릿 조각을 변경하지 않은 채로 둡니다. 그런 다음 함수에서 아래 세부 정보에서 논의하는 예상 속성의 맵을 CloudFormation에 반환합니다.

```
package com.macroexample.lambda.demo;

import java.util.List;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;

import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.RequestHandler;

public class LambdaFunctionHandler implements RequestHandler<Map<String, Object>,
    Map<String, Object>> {

    private static final String REPLACEMENT = "replacement";
    private static final String TARGET = "target";
    private static final String PARAMS = "params";
    private static final String FRAGMENT = "fragment";
    private static final String REQUESTID = "requestId";
    private static final String STATUS = "status";
    private static final String SUCCESS = "SUCCESS";
    private static final String FAILURE = "FAILURE";

    @Override
    public Map<String, Object> handleRequest(Map<String, Object> event, Context
context) {
        // TODO: implement your handler
        final Map<String, Object> responseMap = new HashMap<String, Object>();
        responseMap.put(REQUESTID, event.get(REQUESTID));
        responseMap.put(STATUS, FAILURE);
        try {
            if (!event.containsKey(PARAMS)) {
                throw new RuntimeException("Params are required");
            }

            final Map<String, Object> params = (Map<String, Object>) event.get(PARAMS);
            if (!params.containsKey(REPLACEMENT) || !params.containsKey(TARGET)) {
```

```

        throw new RuntimeException("replacement or target under Params are
required");
    }

    final String replacement = (String) params.get(REPLACEMENT);
    final String target = (String) params.get(TARGET);
    final Object fragment = event.getOrDefault(FRAGMENT, new HashMap<String,
Object>());
    final Object retFragment;
    if (fragment instanceof String) {
        retFragment = iterateAndReplace(replacement, target, (String) fragment);
    } else if (fragment instanceof List) {
        retFragment = iterateAndReplace(replacement, target, (List<Object>) fragment);
    } else if (fragment instanceof Map) {
        retFragment = iterateAndReplace(replacement, target, (Map<String, Object>)
fragment);
    } else {
        retFragment = fragment;
    }
    responseMap.put(STATUS, SUCCESS);
    responseMap.put(FRAGMENT, retFragment);
    return responseMap;
} catch (Exception e) {
    e.printStackTrace();
    context.getLogger().log(e.getMessage());
    return responseMap;
}
}

private Map<String, Object> iterateAndReplace(final String replacement, final
String target, final Map<String, Object> fragment) {
    final Map<String, Object> retFragment = new HashMap<String, Object>();
    final List<String> replacementKeys = new ArrayList<>();
    fragment.forEach((k, v) -> {
        if (v instanceof String) {
            retFragment.put(k, iterateAndReplace(replacement, target, (String)v));
        } else if (v instanceof List) {
            retFragment.put(k, iterateAndReplace(replacement, target, (List<Object>)v));
        } else if (v instanceof Map ) {
            retFragment.put(k, iterateAndReplace(replacement, target, (Map<String, Object>)
v));
        } else {
            retFragment.put(k, v);
        }
    });
}

```

```

    });
    return retFragment;
}

private List<Object> iterateAndReplace(final String replacement, final String
target, final List<Object> fragment) {
    final List<Object> retFragment = new ArrayList<>();
    fragment.forEach(o -> {
        if (o instanceof String) {
            retFragment.add(iterateAndReplace(replacement, target, (String) o));
        } else if (o instanceof List) {
            retFragment.add(iterateAndReplace(replacement, target, (List<Object>) o));
        } else if (o instanceof Map) {
            retFragment.add(iterateAndReplace(replacement, target, (Map<String, Object>)
o));
        } else {
            retFragment.add(o);
        }
    });
    return retFragment;
}

private String iterateAndReplace(final String replacement, final String target,
final String fragment) {
    System.out.println(replacement + " == " + target + " == " + fragment );
    if (fragment != null AND_AND fragment.equals(target))
        return replacement;
    return fragment;
}
}

```

Lambda 함수 응답

다음은 Lambda 함수에서 처리를 위해 CloudFormation으로 반환하는 매핑입니다.

- requestId : 5dba79b5-f117-4de0-9ce4-d40363bfb6ab
- status : SUCCESS
- fragment :

```

{
  "Type": "AWS::CloudFormation::WaitConditionHandle"
}

```

requestId가 CloudFormation에서 전송된 것과 일치하는 경우 status의 SUCCESS는 Lambda 함수가 요청에 포함된 템플릿 조각을 처리했음을 의미합니다. 이 응답에서 fragment에는 기존 템플릿 조각을 대신하여 처리된 템플릿에 삽입할 콘텐츠를 나타내는 JSON이 포함되어 있습니다.

처리된 템플릿 결과

CloudFormation에서 Lambda 함수로부터 성공적인 응답을 수신한 이후 반환된 템플릿 조각을 처리된 템플릿에 삽입합니다.

아래는 예제에 해당하는 처리된 템플릿 결과입니다. JavaMacroFunc 매크로를 참조하는 Fn::Transform 내장 함수 직접 호출은 더 이상 포함되지 않습니다. Lambda 함수에서 반환하는 템플릿 조각은 적절한 위치에 포함되며, 결과적으로 콘텐츠 "Type": "\$\$REPLACEMENT\$\$"는 "Type": "AWS::CloudFormation::WaitConditionHandle"로 변경되었습니다.

```
{
  "Parameters": {
    "ExampleParameter": {
      "Default": "SampleMacro",
      "Type": "String"
    }
  },
  "Resources": {
    "2a": {
      "Type": "AWS::CloudFormation::WaitConditionHandle"
    }
  }
}
```

처리된 템플릿 문제 해결

매크로를 사용할 때 CloudFormation 콘솔에서 처리된 템플릿을 찾을 수 있습니다.

템플릿의 스테이지는 처리 상태를 나타냅니다.

- **Original:** 사용자가 스택 또는 스택 세트를 생성하거나 업데이트하기 위해 최초로 제출한 템플릿입니다.
- **Processed:** CloudFormation에서 참조된 매크로 처리 이후 스택 또는 스택 세트를 생성하거나 업데이트할 때 사용하는 템플릿입니다. 원본 템플릿이 YAML 형식인 경우에도 처리된 템플릿은 JSON 형식입니다.

문제 해결을 위해 처리된 템플릿을 사용합니다. 템플릿이 매크로를 참조하지 않는 경우 원본 템플릿과 처리된 템플릿이 동일합니다.

자세한 내용은 [CloudFormation 콘솔에서 스택 정보 보기](#) 섹션을 참조하세요.

AWS CLI를 사용하여 처리된 템플릿을 가져오려면 [get-template](#) 명령을 사용합니다.

크기 제한

CreateStack, UpdateStack 또는 ValidateTemplate으로 직접 전달될 경우 처리되는 스택 템플릿의 최대 크기는 51,200바이트이고, Amazon S3 템플릿 URL을 사용하여 S3 객체로 처리될 경우에는 1MB입니다. 그러나 처리 중 CloudFormation은 템플릿에 포함된 매크로를 연속적으로 처리할 때 템플릿의 임시 상태를 업데이트합니다. 이 때문에 처리 중 템플릿의 크기가 전체 처리된 템플릿의 허용 크기를 일시적으로 초과할 수 있습니다. CloudFormation은 이러한 프로세스 내 템플릿에 대한 일부 버퍼를 허용합니다. 그러나 처리되는 스택 템플릿의 최대 허용 크기를 고려하여 템플릿과 매크로를 설계해야 합니다.

템플릿을 처리할 때 CloudFormation이 Transformation data limit exceeded 오류를 반환할 경우, CloudFormation이 처리 중 허용하는 최대 템플릿 크기를 초과한 것입니다.

이 문제를 해결하려면 다음과 같이 해 보십시오.

- 템플릿을 여러 개의 템플릿으로 재구성하여 처리 과정의 템플릿에 대한 최대 크기를 넘지 않도록 하십시오. 예제:
 - 중첩 스택 템플릿을 사용하여 템플릿의 부분을 캡슐화합니다. 자세한 내용은 [중첩 스택을 사용하여 템플릿을 재사용 가능한 조각으로 분할](#) 섹션을 참조하세요.
 - 여러 개의 스택을 생성하고 교차 스택 참조를 사용하여 스택 간에 정보를 교환합니다. 자세한 내용은 [다른 CloudFormation 스택의 리소스 출력 참조](#) 섹션을 참조하세요.
- 특정 매크로가 반환하는 템플릿 조각의 크기를 줄입니다. CloudFormation은 매크로가 반환한 조각의 내용을 변경하지 않습니다.

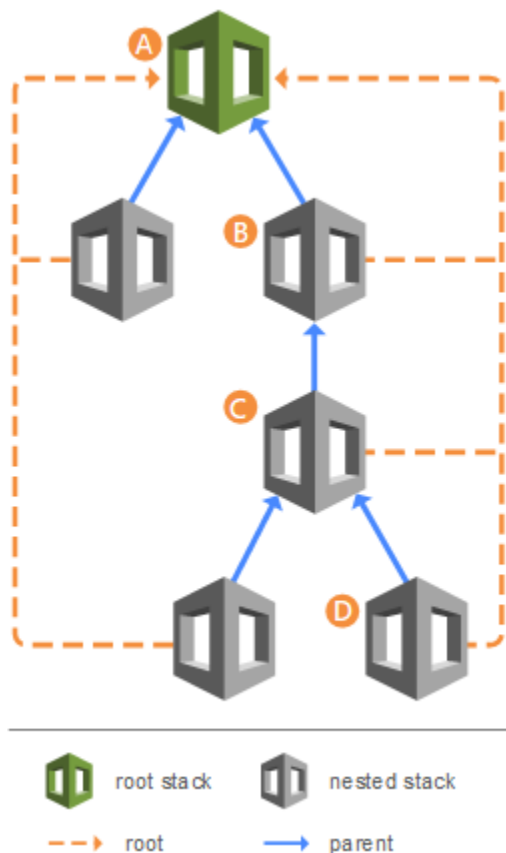
중첩 스택을 사용하여 템플릿을 재사용 가능한 조각으로 분할

인프라가 증가함에 따라 여러 템플릿에서 동일한 리소스 구성을 반복적으로 생성할 수 있습니다. 이러한 중복성을 방지하려면 이러한 공통 구성을 전용 템플릿으로 분리할 수 있습니다. 그런 다음 다른 템플릿의 [AWS::CloudFormation::Stack](#) 리소스를 사용하여 이러한 전용 템플릿을 참조하여 중첩 스택을 생성할 수 있습니다.

예를 들어 대부분의 스택에 사용하는 로드 밸런서 구성이 있다고 가정합니다. 동일한 구성을 복사하여 템플릿에 붙여 넣는 대신 로드 밸런서 전용 템플릿을 생성할 수 있습니다. 그런 다음 동일한 로드 밸런서 구성이 필요한 다른 템플릿 내에서 이 템플릿을 참조할 수 있습니다.

중첩 스택에는 다른 중첩 스택이 포함될 수 있으며 아래 다이어그램에서와 같이 스택 계층 구조가 생성됩니다. 루트 스택은 모든 중첩 스택이 궁극적으로 속하는 최상위 스택입니다. 각 중첩 스택에는 직속 상위 스택이 있습니다. 중첩 스택의 첫 레벨의 경우 루트 스택이 상위 스택이기도 합니다.

- 스택 A는 계층 구조에 있는 다른 모든 중첩 스택의 루트 스택입니다.
- 스택 B의 경우에는 스택 A가 상위 스택이자 루트 스택입니다.
- 스택 C의 경우 스택 B가 상위 스택이고 스택 D의 상위 스택은 스택 C입니다.



주제

- [템플릿 분할 이전 및 이후 예](#)
- [중첩 스택 아키텍처의 예](#)
- [중첩 스택에서 스택 작업 수행](#)
- [관련 정보](#)

템플릿 분할 이전 및 이후 예

이 예제에서는 어떻게 큰 CloudFormation 템플릿 하나를 가져와서 중첩 템플릿을 사용하여 더 정형화 되고 재사용 가능한 설계로 재구성할 수 있는지 보여줍니다. 처음에 "스택 중첩 전" 템플릿에는 파일 하나에 정의된 모든 리소스가 표시됩니다. 리소스 수가 늘어날수록 이 과정이 복잡해지고 관리하기 어려워질 수 있습니다. '스택 중첩 후' 템플릿에서는 리소스를 더 작은 별도의 템플릿으로 분할합니다. 각 중첩 스택에서는 특정 관련 리소스 세트를 처리하므로 전체 구조가 더 체계적으로 구성되고 유지 관리가 더 쉬워집니다.

스택 중첩 전	스택 중첩 후
<pre> AWS::CloudFormation::Template AWSTemplateFormatVersion: 2010-09-09 Parameters: InstanceType: Type: String Default: t2.micro Description: The EC2 instance type Environment: Type: String Default: Production Description: The deployment environment Resources: MyEC2Instance: Type: AWS::EC2::Instance Properties: ImageId: ami-1234567890abcdef0 InstanceType: !Ref InstanceType MyS3Bucket: Type: AWS::S3::Bucket </pre>	<pre> AWS::CloudFormation::Template AWSTemplateFormatVersion: 2010-09-09 Resources: MyFirstNestedStack: Type: AWS::CloudFormation::Stack Properties: TemplateURL: https://s3.amazonaws.com/amzn-s3-demo-bucket/first-nested-stack.yaml Parameters: # Pass parameters to the nested stack if needed InstanceType: t3.micro MySecondNestedStack: Type: AWS::CloudFormation::Stack Properties: TemplateURL: https://s3.amazonaws.com/amzn-s3-demo-bucket/second-nested-stack.yaml Parameters: # Pass parameters to the nested stack if needed Environment: Testing DependsOn: MyFirstNestedStack </pre>

중첩 스택 아키텍처의 예

이 섹션에서는 중첩 스택을 참조하는 최상위 스택으로 구성된 중첩 스택 아키텍처를 보여줍니다. 중첩 스택은 Node.js Lambda 함수를 배포하고, 최상위 스택에서 파라미터 값을 수신하며, 최상위 스택을 통해 노출된 출력을 반환합니다.

주제

- [1단계: 로컬 시스템의 중첩 스택에 대한 템플릿 생성](#)
- [2단계: 로컬 시스템의 최상위 스택에 대한 템플릿 생성](#)
- [3단계: 템플릿 패키징 및 배포](#)

1단계: 로컬 시스템의 중첩 스택에 대한 템플릿 생성

다음 예제에서는 중첩 스택 템플릿의 형식을 보여줍니다.

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: Nested stack template for Lambda function deployment
Parameters:
  MemorySize:
    Type: Number
    Default: 128
    MinValue: 128
    MaxValue: 10240
    Description: Lambda function memory allocation (128-10240 MB)
Resources:
  LambdaFunction:
    Type: AWS::Lambda::Function
    Properties:
      FunctionName: !Sub "${AWS::StackName}-Function"
      Runtime: nodejs18.x
      Handler: index.handler
      Role: !GetAtt LambdaExecutionRole.Arn
      Code:
        ZipFile: |
          exports.handler = async (event) => {
            return {
              statusCode: 200,
              body: JSON.stringify('Hello from Lambda!')
            };
          };
```

```

    };
    MemorySize: !Ref MemorySize
  LambdaExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: lambda.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - 'arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole'
  Outputs:
    LambdaArn:
      Description: ARN of the created Lambda function
      Value: !GetAtt LambdaFunction.Arn

```

2단계: 로컬 시스템의 최상위 스택에 대한 템플릿 생성

다음 예제에서는 최상위 스택 템플릿의 형식과 이전 단계에서 생성한 스택을 참조하는 [AWS::CloudFormation::Stack](#) 리소스를 보여줍니다.

YAML

```

AWSTemplateFormatVersion: 2010-09-09
Description: Top-level stack template that deploys a nested stack
Resources:
  NestedStack:
    Type: AWS::CloudFormation::Stack
    Properties:
      TemplateURL: /path_to_template/nested-template.yaml
      Parameters:
        MemorySize: 256
  Outputs:
    NestedStackLambdaArn:
      Description: ARN of the Lambda function from nested stack
      Value: !GetAtt NestedStack.Outputs.LambdaArn

```

3단계: 템플릿 패키징 및 배포

Note

로컬에서 템플릿으로 작업할 때 AWS CLI `package` 명령을 사용하면 배포를 위한 템플릿을 준비하는 데 도움이 될 수 있습니다. 이는 Amazon S3(TemplateURL 포함)에 대한 로컬 아티팩트 업로드를 자동으로 처리하고 이러한 S3 위치에 대한 업데이트된 참조가 있는 새 템플릿 파일을 생성합니다. 자세한 내용은 [AWS CLI를 사용하여 S3 버킷에 로컬 아티팩트 업로드](#) 섹션을 참조하세요.

그런 다음 `package` 명령을 사용하여 중첩된 템플릿을 Amazon S3 버킷에 업로드할 수 있습니다.

```
aws cloudformation package \
  --s3-bucket amzn-s3-demo-bucket \
  --template /path_to_template/top-level-template.yaml \
  --output-template-file packaged-template.yaml \
  --output json
```

명령은 `--output-template-file`에 지정된 경로에서 새 템플릿을 생성합니다. 이는 아래와 같이 TemplateURL 참조를 Amazon S3 위치로 바꿉니다.

결과 템플릿

```
AWSTemplateFormatVersion: 2010-09-09
Description: Top-level stack template that deploys a nested stack
Resources:
  NestedStack:
    Type: AWS::CloudFormation::Stack
    Properties:
      TemplateURL: https://s3.us-west-2.amazonaws.com/amzn-s3-demo-bucket/8b3bb7aa7abfc6e37e2d06b869484bed.template
      Parameters:
        MemorySize: 256
Outputs:
  NestedStackLambdaArn:
    Description: ARN of the Lambda function from nested stack
    Value:
      Fn::GetAtt:
        - NestedStack
        - Outputs.LambdaArn
```

package 명령을 실행한 후 [deploy](#) 명령을 사용하여 처리된 템플릿을 배포할 수 있습니다. IAM 리소스를 포함하는 중첩 스택의 경우 `--capabilities` 옵션을 포함하여 IAM 기능을 승인해야 합니다.

```
aws cloudformation deploy \  
  --template-file packaged-template.yaml \  
  --stack-name stack-name \  
  --capabilities CAPABILITY_NAMED_IAM
```

중첩 스택에서 스택 작업 수행

중첩 스택으로 작업할 때는 작업 중에 신중하게 처리해야 합니다. 스택 업데이트와 같은 특정 스택 작업은 중첩 스택에서 직접 수행하는 것이 아니라 루트 스택에서 시작해야 합니다. 루트 스택을 업데이트하면 템플릿 변경 사항이 있는 중첩 스택만 업데이트됩니다.

추가로 중첩 스택의 존재가 루트 스택의 작업에 영향을 미칠 수 있습니다. 예를 들어 중첩 스택 하나가 UPDATE_ROLLBACK_IN_PROGRESS 상태로 멈춘 경우 루트 스택은 해당 중첩 스택이 롤백을 완료할 때까지 기다렸다가 계속 진행합니다. 업데이트 작업을 진행하기 전에 롤백 시 스택 업데이트를 취소할 수 있는 IAM 권한이 있는지 확인합니다. 자세한 내용은 [AWS Identity and Access Management를 사용하여 CloudFormation 액세스 제어](#) 섹션을 참조하세요.

다음 절차를 사용하여 루트 스택과 중첩 스택을 찾습니다.

중첩 스택의 루트 스택을 보려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 스택 페이지에서 루트 스택을 확인하고자 하는 중첩 스택의 이름을 선택합니다.

중첩 스택은 스택 이름 위에 NESTED가 표시됩니다.

3. 스택 정보 탭에서 개요 섹션의 루트 스택으로 나열된 스택 이름을 선택합니다.

루트 스택에 속한 중첩 스택을 보려면

1. 중첩 스택을 보려는 루트 스택에서 리소스 탭을 선택합니다.
2. 유형 열에 AWS::CloudFormation::Stack 유형의 리소스를 찾습니다.

관련 정보

- [기존 스택 중첩](#)

- [스택 리소스의 업데이트 동작 이해](#)
- [실패한 중첩 스택 업데이트에서 계속 롤백](#)
- [중첩 스택 롤백 실패](#)

CloudFormation 템플릿에서 대기 조건 생성

이 주제에서는 템플릿에서 대기 조건을 생성하여 스택 리소스의 생성을 조정하고 구성 프로세스의 진행을 추적하는 방법을 설명합니다. 예를 들어, 애플리케이션 구성이 부분적으로 완료된 후 다른 리소스의 생성을 시작하거나, 설치 및 구성 프로세스 중 신호를 전송하여 진행을 추적할 수 있습니다.

CloudFormation에서 대기 조건이 포함된 스택을 생성하는 경우

- 다른 리소스와 마찬가지로 대기 조건을 생성하고 대기 조건의 상태를 CREATE_IN_PROGRESS로 설정합니다.
- CloudFormation은 필요한 수의 성공 신호를 수신하거나 대기 조건의 제한 기간이 만료될 때까지 기다립니다.
- 제한 기간이 만료되기 전에 필요한 수의 성공 신호를 수신하는 경우
 - 대기 조건 상태가 CREATE_COMPLETE로 변경됨
 - 스택 생성이 계속됨
- 제한 시간이 만료되거나 실패 신호가 수신되는 경우
 - 대기 조건 상태가 CREATE_FAILED로 변경됨
 - 스택 롤백

Important

Amazon EC2 및 Auto Scaling 리소스의 경우 대기 조건 대신 CreationPolicy 속성을 사용하는 것이 좋습니다. 그러한 리소스에 CreationPolicy 속성을 추가하고 cfn-signal 헬퍼 스크립트를 사용하여 인스턴스 생성 프로세스가 완료되면 신호를 보내도록 합니다.

자세한 내용은 [CreationPolicy 속성](#)을 참조하세요.

Note

AWS PrivateLink를 사용하는 경우 대기 조건에 응답하는 VPC의 리소스는 CloudFormation의 Amazon Simple Storage Service(Amazon S3) 버킷에 대해 액세스 권한이 있어야 합니다. 리소

스는 미리 서명된 Amazon S3 URL로 대기 조건 응답을 보내야 합니다. Amazon S3에 응답을 보내지 못하면 CloudFormation이 응답을 수신하지 않고 스택 작업이 실패합니다. 자세한 내용은 [인터페이스 엔드포인트를 사용하여 CloudFormation에 액세스\(AWS PrivateLink\)](#) 및 [버킷 정책을 사용하여 VPC 엔드포인트에서 액세스 제어](#)를 참조하세요.

주제

- [템플릿에서 대기 조건 생성](#)
- [대기 조건 신호 구문](#)
- [신호 데이터에 액세스](#)

템플릿에서 대기 조건 생성

1. 대기 조건 핸들

스택의 템플릿에서 [AWS::CloudFormation::WaitConditionHandle](#) 리소스를 정의하는 것으로 시작합니다. 이 리소스는 신호를 보내는 데 필요한 미리 서명된 URL을 생성합니다. 이를 사용하면 AWS 자격 증명을 제공할 필요 없이 신호를 전송할 수 있습니다. 예제:

Resources:

MyWaitHandle:

Type: AWS::CloudFormation::WaitConditionHandle

2. 대기 조건

다음으로, 스택의 템플릿에서 [AWS::CloudFormation::WaitCondition](#) 리소스를 정의합니다. AWS::CloudFormation::WaitCondition의 기본 구조는 다음과 같습니다.

MyWaitCondition:

Type: AWS::CloudFormation::WaitCondition

Properties:

Handle: *String*

Timeout: *String*

Count: *Integer*

AWS::CloudFormation::WaitCondition 리소스에는 2개의 필수 속성과 1개의 선택적 속성이 있습니다.

- Handle(필수) - 템플릿에 선언된 WaitConditionHandle에 대한 참조입니다.

- **Timeout(필수)** - CloudFormation이 필요한 수의 신호를 수신할 때까지 대기하는 시간(초)입니다. Timeout은 최소 경계 속성이므로, 제한 시간은 지정한 시간이 되자마자 발생하지만 잠시 후에 발생할 수도 있습니다. 지정할 수 있는 최대 시간은 43,200초(12시간)입니다.
- **Count(선택 사항)** - CloudFormation이 해당 대기 조건의 상태를 CREATE_COMPLETE로 설정하고 스택 생성을 재개하기 전에 수신해야 하는 성공 신호 수입니다. 지정하지 않은 경우 기본값은 1입니다.

일반적으로 특정 리소스가 생성된 직후에 대기 조건이 시작되기를 원합니다. DependsOn 속성을 대기 조건에 추가하여 이를 수행할 수 있습니다. 대기 조건에 DependsOn 속성을 추가하면 CloudFormation은 먼저 DependsOn 속성에 리소스를 생성한 다음 대기 조건을 생성합니다. 자세한 내용은 [DependsOn 속성](#)을 참조하세요.

다음 예제는 다음과 같은 대기 조건을 보여줍니다.

- MyEC2Instance 리소스가 성공적으로 생성된 후 시작
- WaitConditionHandle로 MyWaitHandle 리소스 사용
- 제한 시간 4,500초
- 기본 Count 1(Count 속성이 지정되지 않았기 때문)

MyWaitCondition:

```
Type: AWS::CloudFormation::WaitCondition
DependsOn: MyEC2Instance
Properties:
  Handle: !Ref MyWaitHandle
  Timeout: '4500'
```

3. 신호 전송

CloudFormation에 성공 또는 실패를 알리기 위해 일반적으로 일부 코드 또는 스크립트를 실행합니다. 예를 들어 EC2 인스턴스에서 실행되는 애플리케이션은 몇 가지 추가 구성 태스크를 수행한 다음 CloudFormation에 완료를 나타내는 신호를 전송할 수 있습니다.

대기 조건 핸들에 의해 생성된 미리 서명된 URL로 신호가 전송되어야 합니다. 미리 서명된 URL을 사용하여 성공 또는 실패를 알립니다.

신호를 전송하려면 다음을 수행하세요.

1. 템플릿 내에서 미리 서명된 URL을 검색하려면 대기 조건 핸들의 논리명과 함께 Ref 내장 함수를 사용합니다.

다음 예제와 같이 템플릿은 Amazon EC2 UserData 속성을 사용하여 Amazon EC2 인스턴스를 선언하고 미리 서명된 URL을 Amazon EC2 인스턴스에 전달할 수 있습니다. 이를 통해 해당 인스턴스에서 실행되는 스크립트 또는 애플리케이션이 CloudFormation에 성공 또는 실패를 알릴 수 있습니다.

```
MyEC2Instance:
  Type: AWS::EC2::Instance
  Properties:
    InstanceType: t2.micro # Example instance type
    ImageId: ami-055e3d4f0bbeb5878 # Change this as needed (Amazon Linux 2023 in
us-west-2)
    UserData:
      Fn::Base64:
        Fn::Join:
          - ""
          - - "SignalURL="
            - { "Ref": "MyWaitHandle" }
```

결과로 다음과 유사한 UserData 출력이 반환됩니다.

```
SignalURL=https://amzn-s3-demo-bucket.s3.amazonaws.com/....
```

참고: AWS Management Console 및 명령줄 도구에서 미리 서명된 URL은 대기 조건 핸들 리소스의 물리적 ID로 표시됩니다.

2. (선택 사항) 스택이 대기 조건에 들어가는 시점을 감지하려면 다음 방법 중 하나를 사용합니다.
 - 알림이 활성화된 상태로 스택을 생성하면 CloudFormation은 모든 스택 이벤트에 대한 알림을 지정된 주제에 게시합니다. 사용자나 사용자의 애플리케이션이 해당 주제를 구독하는 경우 알림에서 대기 조건 핸들 생성 이벤트를 모니터링하고 알림 메시지에서 미리 서명된 URL을 가져올 수 있습니다.
 - AWS Management Console, AWS CLI 또는 SDK를 사용하여 스택의 이벤트를 모니터링할 수도 있습니다.
3. 신호를 전송하려면 미리 서명된 URL을 사용하여 HTTP 요청 메시지를 전송합니다. 요청 메시드는 PUT이어야 하며 Content-Type 헤더는 빈 문자열이거나 생략되어야 합니다. 요청 메시지는 [대기 조건 신호 구문](#)에 지정된 형식의 JSON 구조여야 합니다.

CloudFormation에서 스택 생성을 계속하려면 Count 속성으로 지정된 수의 성공 신호를 전송해야 합니다. Count이(가) 1보다 큰 경우 특정 대기 조건에 전송된 모든 신호에서 각 신호의 UniqueId 값이 고유해야 합니다. UniqueId은(는) 임의의 영숫자 문자열입니다.

curl 명령은 신호를 전송하는 한 가지 방법입니다. 다음 예에서는 대기 조건에 성공 신호를 전송하는 curl 명령줄을 보여 줍니다.

```
$ curl -T /tmp/a \
  "https://amzn-s3-demo-bucket.s3.amazonaws.com/arn%3Aaws%3Acloudformation%3Aus-west-2%3A034017226601%3Astack%2Fstack-gosar-20110427004224-test-stack-with-WaitCondition--VEYW%2Fe498ce60-70a1-11e0-81a7-5081d0136786%2FmyWaitConditionHandle?Expires=1303976584&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Signature=ik1twT6hpS4cgNAw7wy0oRejVo%3D"
```

이때 `/tmp/a` 파일에는 다음과 같은 JSON 구조가 포함되어 있습니다.

```
{
  "Status" : "SUCCESS",
  "Reason" : "Configuration Complete",
  "UniqueId" : "ID1234",
  "Data" : "Application has completed configuration."
}
```

이 예에서는 명령줄에서 JSON 구조를 파라미터로 전송하는 경우를 제외하고 동일한 성공 신호를 전송하는 curl 명령줄을 보여줍니다.

```
$ curl -X PUT \
  -H 'Content-Type:' --data-binary '{"Status" : "SUCCESS","Reason" : "Configuration Complete","UniqueId" : "ID1234","Data" : "Application has completed configuration."}' \
  "https://amzn-s3-demo-bucket.s3.amazonaws.com/arn%3Aaws%3Acloudformation%3Aus-west-2%3A034017226601%3Astack%2Fstack-gosar-20110427004224-test-stack-with-WaitCondition--VEYW%2Fe498ce60-70a1-11e0-81a7-5081d0136786%2FmyWaitConditionHandle?Expires=1303976584&AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Signature=ik1twT6hpS4cgNAw7wy0oRejVo%3D"
```

대기 조건 신호 구문

대기 조건 핸들에 의해 생성된 URL로 신호를 전송할 때는 다음 JSON 형식을 사용해야 합니다.

```
{
  "Status" : "StatusValue",
  "UniqueId" : "Some UniqueId",
  "Data" : "Some Data",
  "Reason" : "Some Reason"
}
```

속성

Status 필드는 다음 값 중 하나여야 합니다.

- SUCCESS
- FAILURE

UniqueId 필드는 CloudFormation에 대한 신호를 식별합니다. 대기 조건의 Count 속성이 1보다 큰 경우 UniqueId 값은 특정 대기 조건에 대해 전송된 모든 신호에서 고유해야 합니다. 그렇지 않으면 CloudFormation은 신호를 동일한 UniqueId로 이전에 전송된 신호를 재전송한 것으로 간주하고 무시합니다.

Data 필드에는 신호와 함께 다시 전송하려는 모든 정보가 포함될 수 있습니다. 템플릿 내에서 [Fn::GetAtt](#) 함수를 사용하여 Data 값에 액세스할 수 있습니다.

Reason 필드는 콘텐츠에 대해 JSON 규정 이외의 다른 제한 사항이 없는 문자열입니다.

신호 데이터에 액세스

유효한 신호에서 전송된 데이터에 액세스하려면 CloudFormation 템플릿에서 대기 조건에 대한 출력 값을 생성할 수 있습니다. 예제:

Outputs:

WaitConditionData:

Description: *The data passed back as part of signalling the WaitCondition*

Value: *!GetAtt MyWaitCondition.Data*

그런 다음 [describe-stacks](#) 명령이나 CloudFormation 콘솔의 출력 탭을 사용하여 이 데이터를 볼 수 있습니다.

Fn::GetAtt 함수는 UniqueId와 Data를 JSON 구조 내 이름/값 페어로 반환합니다. 예제:

```
{"Signal1": "Application has completed configuration."}
```

CloudFormation 모듈을 사용하여 템플릿 전체에 포함할 수 있는 재사용 가능한 리소스 구성 생성

모듈을 사용하면 모든 스택 템플릿에 포함하기 위한 리소스 구성을 투명하고 관리가 편리하면서도 반복적으로 패키징할 수 있습니다. 모듈은 공통적 서비스 구성과 모범 사례를 모듈식 맞춤형 구성 요소로 캡슐화하고 스택 템플릿에 포함할 수 있습니다. 모듈을 사용하면 리소스 구성을 포함할 수 있습니다. 리소스 구현에 대한 복잡한 지식을 심층적으로 배우지 않더라도 리소스 구성을 통해 모범 사례, 전문가 영역 지식, 허용되는 가이드라인(보안, 규정 준수, 거버넌스, 산업 규정 등의 분야)을 템플릿에 포함할 수 있습니다.

예를 들어 네트워킹 영역 전문가는 기본 보안 그룹과 보안 가이드라인을 준수하는 수신/송신 규칙이 포함된 모듈을 만들 수 있습니다. 그런 다음, 템플릿에 모듈을 포함하여 스택에 보안 네트워킹 인프라를 프로비저닝할 수 있습니다. VPC, 서브넷, 보안 그룹, 게이트웨이의 작동 원리를 배우지 않아도 됩니다. 모듈에 버전이 지정되기 때문에 시간이 지나서 보안 가이드라인이 변경될 경우, 모듈 작성자는 이런 변경 사항을 통합하는 새 버전의 모듈을 만들 수 있습니다.

템플릿에서 모듈을 사용하는 방법의 특징은 다음과 같습니다.

- 예측 가능성 - 모듈은 CloudFormation 레지스트리에 등록하는 스키마를 준수해야 합니다. 그래야 템플릿에 모듈을 포함한 후 어떤 리소스를 해석할 수 있는지 알 수 있습니다.
- 재사용 가능성 - 여러 템플릿과 계정에서 동일한 모듈을 사용할 수 있습니다.
- 추적 가능성 - CloudFormation은 모듈에서 스택의 어떤 리소스가 프로비저닝되었는지 알고 있으므로, 리소스 변경 사항의 소스를 쉽게 이해할 수 있습니다.
- 관리 용이성 - 모듈에 등록하고 나면 버전 관리, 계정 및 리전 가용성을 포함하여 CloudFormation 레지스트리를 통해 모듈을 관리할 수 있습니다.

모듈에는 다음을 포함할 수 있습니다.

- 모듈에서 프로비저닝되는 하나 이상의 리소스 및 관련 데이터(예: 출력, 조건).
- 모듈 파라미터(모듈을 사용할 때마다 사용자 지정 값을 지정할 수 있게 됩니다).

모듈 개발에 대한 자세한 내용은 CloudFormation CLI User Guide의 [Developing modules](#)를 참조하세요.

주제

- [모듈 사용 시 고려 사항](#)

- [모듈 버전 관리 이해](#)
- [CloudFormation 프라이빗 레지스트리의 모듈 사용](#)
- [파라미터를 사용하여 모듈 값 지정](#)
- [CloudFormation 템플릿의 모듈 리소스 참조](#)

모듈 사용 시 고려 사항

- 모듈을 사용하더라도 추가 요금은 없습니다. 모듈이 스택에서 해석하는 리소스에 대해서만 요금을 지불합니다.
- CloudFormation 할당량(스택에 허용되는 리소스의 최대 개수, 템플릿 본문의 최대 크기)을 처리된 템플릿에 적용합니다. 이는 해당 템플릿에 포함된 리소스가 모듈의 리소스인지 여부와는 관계가 없습니다. 자세한 내용은 [CloudFormation 할당량 이해](#) 섹션을 참조하세요.
- 스택 수준에서 지정하는 태그는 모듈에서 파생된 각 리소스에 할당됩니다.
- CloudFormation이 템플릿을 처리할 때 모듈 수준에서 지정된 도우미 스크립트는 모듈에 포함된 개별 리소스로 전달되지 않습니다.
- 모듈에 지정된 출력은 템플릿 수준의 출력에 전달됩니다.

각 출력에는 모듈의 논리적 이름과 모듈에 정의된 출력 이름을 연결하는 논리적 ID가 할당됩니다. 자세한 내용은 [배포된 CloudFormation 스택에서 내보낸 출력 가져오기](#) 섹션을 참조하세요.

- 모듈에 지정된 파라미터는 템플릿 수준의 파라미터에 전달되지 않습니다.

그러나 모듈 수준 파라미터를 참조하는 템플릿 수준 파라미터를 생성할 수 있습니다. 자세한 내용은 [파라미터를 사용하여 모듈 값 지정](#) 섹션을 참조하세요.

모듈 버전 관리 이해

CloudFormation 레지스트리는 AWS 계정과 리전 내에서 사용할 모듈을 등록하고 관리할 수 있는 리포지토리 역할을 합니다. 계정과 리전 내에서 AWS, 서드 파티 게시자 및 사용자 지정 확장 프로그램을 포함한 다양한 소스의 모듈을 등록할 수 있습니다. 자세한 내용은 [CloudFormation 레지스트리를 통해 익스텐션 관리](#) 섹션을 참조하세요.

모듈에는 여러 버전이 있을 수 있으므로 사용하려는 모듈의 버전을 지정할 수 있습니다. 이 버전 관리 기능은 모듈에 종속된 기존 스택을 손상시키지 않고 모듈을 업데이트하거나 수정해야 할 때 특히 유용합니다.

여러 버전의 모듈을 사용할 때는 다음 고려 사항에 유의하세요.

- 스택 작업 중 CloudFormation은 현재 스택 작업이 수행되는 AWS 계정과 리전에 기본 버전으로 등록된 모듈 버전을 사용합니다. 여기에는 다른 모듈에 중첩된 모듈이 포함됩니다.

따라서 다른 계정이나 리전에 동일한 모듈의 다른 버전이 기본 버전으로 등록되어 있는 경우 같은 템플릿을 사용하더라도 결과가 다를 수 있습니다.

- 스택 작업 중에 CloudFormation은 현재 스택 작업이 수행되는 AWS 계정과 리전에 기본 버전으로 등록된 리소스 버전을 사용합니다. 여기에는 모듈을 포함하여 생성된 리소스가 포함됩니다.
- 모듈의 기본 버전을 바꾸더라도 스택 업데이트 작업이 시작되지 않습니다. 그러나 다음에 해당 모듈을 포함하는 템플릿으로 스택 작업을 실행할 경우(예: 스택 업데이트) CloudFormation은 작업에 새 기본 버전을 사용합니다.

한 가지 예외는 이전 템플릿 사용 옵션을 지정한 스택 업데이트입니다. 자세한 내용은 아래에 설명되어 있습니다.

- 스택 업데이트 작업에서 이전 템플릿 사용 옵션을 지정할 경우, CloudFormation은 이전에 스택 업데이트에서 처리된 템플릿을 사용하고 변경 사항에 대해 모듈을 다시 처리하지 않습니다.
- 일관된 결과를 보장하려면 스택 세트와 함께 사용할 스택 템플릿에 모듈을 포함할 경우 스택 인스턴스를 배포하고자 하는 모든 계정과 리전에서 동일한 모듈 버전을 기본 버전으로 설정해야 합니다. 여기에는 다른 모듈에 중첩된 모듈의 버전도 포함됩니다. 자세한 내용은 [StackSets를 사용하여 여러 계정 및 리전의 스택 관리](#) 섹션을 참조하세요.

서드 파티 퍼블릭 모듈 활성화를 위한 요구 사항

계정 및 리전에서 서드 파티 퍼블릭 모듈을 성공적으로 활성화하려면 모듈에 포함된 각 서드 파티 공개 확장 프로그램(리소스 또는 모듈)에 대해 다음 조건이 충족되어야 합니다.

- 확장 프로그램 활성화 - 확장 프로그램을 사용하려는 계정과 리전에서 확장 프로그램을 활성화해야 합니다. 자세한 내용은 [CloudFormation 레지스트리에서 서드 파티 공개 확장 프로그램 사용](#) 섹션을 참조하세요.
- 별칭 등록 - 모듈의 확장 프로그램이 유형 이름 별칭을 사용하는 경우 확장 프로그램은 동일한 유형 이름 별칭을 사용하여 계정과 리전에 등록되어야 합니다. 자세한 내용은 [별칭을 사용하여 확장 프로그램 참조](#) 섹션을 참조하세요.
- 버전 호환성 - 현재 활성화된 확장 프로그램 버전은 모듈에 지정된 해당 확장 프로그램의 지원되는 메이저 버전 중 하나여야 합니다.

올바른 서드 파티 공개 확장 프로그램 및 확장 프로그램 버전을 활성화하지 않은 경우 CloudFormation은 모듈을 성공적으로 활성화하기 전에 활성화해야 하는 확장 프로그램과 버전을 나열하는 오류와 함께 작업에 실패합니다.

CloudFormation 프라이빗 레지스트리의 모듈 사용

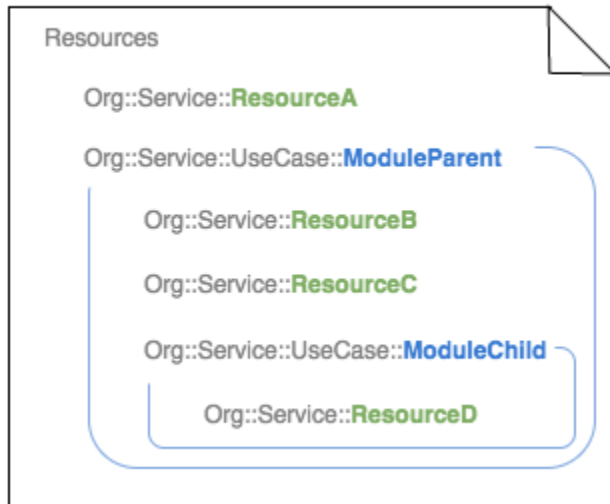
이 주제에서는 CloudFormation 템플릿에서 모듈을 사용하는 방법을 설명합니다. 모듈은 템플릿에 추가할 수 있는 미리 만들어진 리소스 번들이라고 생각하면 됩니다.

모듈을 사용하려면 다음 단계를 수행하세요.

- **모듈 등록** - CloudFormation 레지스트리에 모듈을 비공개 확장 프로그램으로 등록할 수 있습니다. 작업 중인 AWS 계정과 리전에 모듈이 등록되어 있는지 확인하세요. 자세한 내용은 [CloudFormation 레지스트리 개념](#) 섹션을 참조하세요.
- **템플릿에 포함** - 다른 리소스와 마찬가지로 CloudFormation 템플릿의 [Resources](#) 섹션에 모듈을 추가합니다. 모듈에 필요한 속성도 제공해야 합니다.
- **스택 생성 또는 업데이트** - 스택 작업을 시작하면 CloudFormation이 처리된 템플릿을 생성합니다. 이 템플릿은 포함된 모듈을 적절한 리소스로 해석합니다.
- **변경 사항 미리 보기** - 변경하기 전에 변경 세트를 사용하여 추가 또는 변경될 리소스를 확인할 수 있습니다. 자세한 내용은 [변경 세트를 사용하여 CloudFormation 스택 업데이트](#) 섹션을 참조하세요.

리소스와 모듈을 모두 포함한 템플릿을 예시로 생각해 보세요. 템플릿에는 개별 리소스(ResourceA)와 모듈(ModuleParent)이 포함되어 있습니다. 이 모듈에는 ResourceB 및 ResourceC 리소스는 물론 중첩된 모듈인 ModuleChild가 포함되어 있습니다. ModuleChild에는 단일 리소스인 ResourceD가 포함되어 있습니다. 이 템플릿에서 스택을 생성하면 CloudFormation이 템플릿을 처리하고 모듈을 적절한 리소스로 해석합니다. 결과로 나오는 스택에는 ResourceA, ResourceB, ResourceC, ResourceD라는 4개의 리소스가 있습니다.

Template



Processed Template



CloudFormation은 스택의 어떤 리소스가 모듈에서 생성되었는지 추적합니다. [이벤트], [리소스], [드립트] 탭에서 지정된 스택에 대한 정보를 확인할 수 있으며, 이는 변경 세트 미리 보기에도 포함됩니다.

모듈은 템플릿의 리소스와 구별됩니다. 리소스는 일반적으로 세 부분으로 구성된 규칙을 준수하는 것과 달리, 모듈은 네 부분으로 구성된 이름 지정 규칙을 준수하기 때문입니다.

`organization::service::use-case::MODULE`

파라미터를 사용하여 모듈 값 지정

CloudFormation에서 템플릿 파라미터로 스택 생성 또는 업데이트 중 입력 값을 제공하여 스택을 사용자 지정할 수 있습니다. 이러한 파라미터를 사용하면 필요에 따라 스택의 특정 측면을 변경할 수 있습니다. 템플릿 파라미터 정의에 대한 자세한 내용은 [CloudFormation 템플릿 Parameters 구문](#) 섹션을 참조하세요.

마찬가지로 모듈에도 파라미터가 있을 수 있습니다. 이러한 모듈 파라미터를 사용하면 해당 모듈을 사용하는 템플릿 또는 다른 모듈에서 모듈에 사용자 지정 값을 입력할 수 있습니다. 그런 다음 모듈은 이러한 사용자 지정 값을 사용하여 포함된 리소스의 속성 값을 설정할 수 있습니다.

또한 스택 작업 시 모듈에 전달되는 값을 입력할 수 있도록 모듈 속성을 설정하는 템플릿 파라미터를 정의할 수 있습니다.

모듈에 자체 모듈 파라미터가 있는 중첩 모듈이 있는 경우 다음 중 하나를 수행할 수 있습니다.

- 중첩 모듈 파라미터에 대한 값을 상위 모듈에 직접 지정합니다.

- 상위 모듈에서 해당 모듈 파라미터를 정의합니다. 상위 모듈이 포함된 템플릿(또는 모듈)에서 중첩 모듈 파라미터를 설정할 수 있습니다.

템플릿 파라미터를 사용하여 모듈 파라미터 값 지정

다음 예제는 값을 모듈에 전달하는 템플릿 파라미터를 정의하는 방법을 나타냅니다.

My::S3::SampleBucket::MODULE을 포함하는 이 템플릿은 사용자가 스택 작업 중 S3 버킷 이름을 지정할 수 있는 템플릿 파라미터인 BucketName을 정의합니다.

```
# Template containing My::S3::SampleBucket::MODULE
Parameters:
  BucketName:
    Description: Name for your sample bucket
    Type: String
Resources:
  MyBucket:
    Type: 'My::S3::SampleBucket::MODULE'
    Properties:
      BucketName: !Ref BucketName
```

상위 모듈에서 하위 모듈의 리소스에 대한 속성 지정

다음 예제는 다른 모듈 안에 중첩된 모듈에서 파라미터 값을 지정하는 방법을 나타냅니다.

이 첫 번째 모듈 My::S3::SampleBucketPrivate::MODULE이 하위 모듈이 됩니다. 이 하위 모듈이 파라미터 2개(BucketName과 AccessControl)를 정의합니다. 두 가지 파라미터에 지정된 값을 사용하여 모듈에 포함된 AWS::S3::Bucket 리소스의 BucketName과 AccessControl을 지정합니다. 다음의 My::S3::SampleBucketPrivate::MODULE에 대한 템플릿 프래그먼트입니다.

```
# My::S3::SampleBucketPrivate::MODULE
AWSTemplateFormatVersion: 2010-09-09
Description: A sample S3 Bucket with Versioning and DeletionPolicy.
Parameters:
  BucketName:
    Description: Name for the bucket
    Type: String
  AccessControl:
    Description: AccessControl for the bucket
    Type: String
```

```
Resources:
  S3Bucket:
    Type: AWS::S3::Bucket
    Properties:
      BucketName: !Ref BucketName
      AccessControl: !Ref AccessControl
      DeletionPolicy: Retain
      VersioningConfiguration:
        Status: Enabled
```

그 다음에는 이전 모듈이 상위 모듈 `My::S3::SampleBucket::MODULE`에 중첩됩니다. 상위 모듈 `My::S3::SampleBucket::MODULE`은 다음과 같은 방법으로 하위 모듈 파라미터를 설정합니다.

- `My::S3::SampleBucketPrivate::MODULE`의 `AccessControl` 파라미터를 `Private`으로 설정합니다.
- `BucketName`은 모듈 파라미터를 정의합니다. `My::S3::SampleBucket::MODULE`을 포함하는 템플릿(또는 모듈)에서 버킷 이름을 지정할 수 있게 됩니다.

```
# My::S3::SampleBucket::MODULE
AWSTemplateFormatVersion: 2010-09-09
Description: A sample S3 Bucket. With Private AccessControl.
Parameters:
  BucketName:
    Description: Name for your sample bucket
    Type: String
Resources:
  MyBucket:
    Type: 'My::S3::SampleBucketPrivate::MODULE'
    Properties:
      BucketName: !Ref BucketName
      AccessControl: Private
```

모듈 파라미터에 대한 제약 조건 지정

모듈 파라미터는 제약 조건의 적용을 지원하지 않습니다. 모듈 파라미터에 대해 제약 조건 검사를 실행하려면 원하는 제약 조건으로 템플릿 파라미터를 생성합니다. 그런 다음 모듈 파라미터에서 템플릿 파라미터를 참조합니다. 템플릿 파라미터 정의에 대한 자세한 내용은 [CloudFormation 템플릿 Parameters 구문](#) 섹션을 참조하세요.

CloudFormation 템플릿의 모듈 리소스 참조

CloudFormation에서 다른 리소스의 이름이나 속성을 기반으로 한 리소스의 속성을 설정해야 하는 경우가 있습니다. 자세한 내용은 [참조 리소스](#) 섹션을 참조하세요.

CloudFormation 템플릿의 모듈에 포함된 리소스를 참조하려면 두 논리명을 결합해야 합니다.

- 템플릿에 모듈을 포함할 때 모듈 자체에 지정한 논리명입니다.
- 해당 모듈 내 특정 리소스의 논리명입니다.

이 두 논리명을 마침표(.)를 사용하거나 사용하지 않고 결합할 수 있습니다. 예를 들어, 모듈의 논리명이 MyModule이고 리소스의 논리명이 MyBucket인 경우 해당 리소스를 MyModule.MyBucket 또는 MyModuleMyBucket으로 참조할 수 있습니다.

모듈 내부 리소스의 논리명을 찾으려면 CloudFormation 레지스트리에서 사용할 수 있는 모듈 스키마를 참조하거나 [DescribeType](#) 작업을 사용합니다. 스키마에는 모듈의 일부인 모든 리소스와 해당 논리명이 나열됩니다.

전체 논리명을 알면 GetAtt 및 Ref와 같은 CloudFormation 함수를 사용하여 모듈 리소스의 속성 값에 액세스할 수 있습니다.

예를 들어, 논리명이 S3Bucket인 AWS::S3::Bucket 리소스를 포함하는 My::S3::SampleBucket::MODULE 모듈이 있다고 가정해 보겠습니다. Ref 함수를 사용하여 이 버킷의 이름을 참조하려면 템플릿의 모듈 이름(MyBucket)과 모듈의 리소스 논리명(S3Bucket)을 결합합니다. 전체 논리명은 MyBucket.S3Bucket 또는 MyBucketS3Bucket입니다.

예제 템플릿

다음 예 템플릿은 My::S3::SampleBucket::MODULE 모듈을 사용하여 S3 버킷을 생성합니다. 또한 Amazon SQS 대기열을 생성하고 해당 이름을 모듈의 버킷 이름과 동일하게 설정합니다. 생성된 S3 버킷의 Amazon 리소스 이름(ARN)도 출력합니다.

```
# Template that uses My::S3::SampleBucket::MODULE
Parameters:
  BucketName:
    Description: Name for your sample bucket
    Type: String
Resources:
  MyBucket:
    Type: My::S3::SampleBucket::MODULE
```

```
Properties:
  BucketName: !Ref BucketName
exampleQueue:
  Type: AWS::SQS::Queue
  Properties:
    QueueName: !Ref MyBucket.S3Bucket
Outputs:
  BucketArn:
    Value: !GetAtt MyBucket.S3Bucket.Arn
```

CloudFormation 스택을 통해 하나의 단위로 AWS 리소스 관리

스택이란 하나의 단위로 관리할 수 있는 AWS 리소스의 모음입니다. 다시 말해서 스택을 생성, 업데이트 또는 삭제하여 리소스 모음을 생성, 업데이트 및 삭제할 수 있습니다.

스택 생성에는 리소스와 해당 구성을 지정하는 CloudFormation 템플릿 배포가 포함됩니다. CloudFormation에서 이를 프로비저닝하고 구성합니다.

스택 업데이트에는 템플릿 또는 파라미터에 변경 사항을 적용하는 것이 포함됩니다. CloudFormation에서는 제출하는 변경 사항을 스택의 현재 상태를 비교하여 변경된 리소스만 업데이트합니다. CloudFormation에서는 업데이트하는 속성에 따라 리소스를 중단하거나 업데이트된 리소스를 대체할 수 있습니다. 리소스 업데이트 동작에 대한 자세한 내용은 [스택 리소스의 업데이트 동작 이해](#) 단원을 참조하십시오.

CloudFormation에서는 스택을 업데이트하는 두 가지 방법을 제공합니다.

- 변경 세트 – 변경 세트를 사용하면 CloudFormation에서 스택에 대해 변경한 사항을 미리 보고 이러한 변경 사항을 적용할지 여부를 결정할 수 있습니다. 변경 세트는 JSON 형식 문서로, CloudFormation에서 스택에 대해 변경한 사항이 요약되어 있습니다. CloudFormation에서 의도하지 않은 변경이 발생하지 않도록 하거나 여러 옵션을 고려하려는 경우 변경 세트를 사용합니다. 예를 들어, 변경 세트를 사용하여 업데이트 중에 CloudFormation에서 스택의 데이터베이스 인스턴스를 대체할 수 없는지 확인할 수 있습니다.
- 직접 업데이트 – 스택을 직접 업데이트하는 경우 변경 사항을 제출하면 CloudFormation에서 즉시 해당 변경 사항을 배포합니다. 업데이트를 빠르게 배포하려면 직접 업데이트를 사용합니다.

스택을 삭제하면 해당 스택과 연결된 리소스가 삭제됩니다. 예를 들어 스택에는 웹 애플리케이션을 실행하는 데 필요한 모든 리소스(예: 웹 서버, 데이터베이스 및 네트워킹 규칙)가 포함될 수 있습니다. 해당 웹 애플리케이션이 더 이상 필요하지 않은 경우 간단히 스택을 삭제하면 관련 리소스가 모두 삭제됩니다.

Note

스택을 곧바로 삭제했더라도 스택 리소스가 작동된 시간 동안 스택 리소스에 대한 책임은 사용자에게 있습니다.

CloudFormation은 모든 스택 리소스가 적절히 생성 또는 삭제되도록 합니다. CloudFormation은 스택 리소스를 하나의 단위로 처리하기 때문에 스택을 생성하거나 삭제하려면 모든 리소스를 생성하거나 삭제해야 합니다. 리소스를 생성할 수 없는 경우 CloudFormation에서 해당 스택을 롤백하고 생성된 모든 리소스를 자동으로 삭제합니다. 리소스를 삭제할 수 없는 경우, 스택이 삭제될 때까지 나머지 모든 리소스가 그대로 유지됩니다.

주제

- [스택 관리를 위한 인터페이스](#)
- [CloudFormation 콘솔에서 스택 생성](#)
- [CloudFormation 콘솔에서 스택 정보 보기](#)
- [스택 템플릿 업데이트](#)
- [스택 리소스의 업데이트 동작 이해](#)
- [변경 세트를 사용하여 CloudFormation 스택 업데이트](#)
- [스택 배포 검증](#)
- [직접 스택 업데이트](#)
- [스택 업데이트 취소](#)
- [CloudFormation 콘솔에서 스택 삭제](#)
- [스택 진행 상황 모니터링](#)
- [롤백 트리거를 사용하여 경보 위반 시 CloudFormation 스택을 롤백합니다.](#)
- [드리프트 감지를 사용하여 스택 및 리소스에 대한 비관리형 구성 변경 감지](#)
- [CloudFormation 스택에 AWS 리소스 가져오기](#)
- [스택 리팩터링](#)
- [리소스 유형 지원](#)
- [빠른 생성 링크를 사용하여 CloudFormation 스택 생성](#)
- [AWS CLI 및 PowerShell에 대한 CloudFormation 스택 작업 명령의 예](#)

스택 관리를 위한 인터페이스

다음과 같은 인터페이스를 사용하여 CloudFormation 스택을 관리할 수 있습니다.

- CloudFormation 콘솔 – 스택에 액세스할 때 사용할 수 있는 웹 인터페이스를 제공합니다. AWS Management Console에 로그인하고 탐색 모음의 검색 상자를 사용하여 CloudFormation을 검색한 다음 검색 결과에서 CloudFormation을 선택하여 CloudFormation 콘솔에 액세스할 수 있습니다.

- AWS Command Line Interface – CloudFormation을 포함한 다양한 AWS 서비스에서 사용되는 명령을 제공하며 Windows, Mac 및 Linux에서 지원됩니다. CloudFormation 명령에 대한 자세한 내용은 AWS CLI 명령 참조의 [cloudformation](#)을 참조하세요.
- AWS Tools for PowerShell – SDK for .NET에 의해 노출되는 기능을 기반으로 구축된 PowerShell 모듈 세트입니다. Tools for PowerShell을 사용하면 PowerShell 명령줄에서 AWS 리소스에 대한 작업을 스크립팅할 수 있습니다. CloudFormation용 cmdlet은 [AWS Tools for PowerShell Cmdlet 참조](#)에서 찾아볼 수 있습니다.
- 쿼리 API - HTTPS 요청을 사용하여 호출하는 하위 수준의 API 작업을 제공합니다. 애플리케이션에서 API를 직접적으로 호출할 경우 요청에 서명할 해시 생성과 같은 하위 수준의 세부 정보를 처리하는 코드를 작성해야 합니다. CloudFormation의 API 작업에 대한 자세한 내용은 AWS CloudFormation API 참조의 [작업](#)을 참조하세요.
- AWS SDK – 언어별 API를 제공하고, 서명 계산, 요청 재시도 처리 및 오류 처리와 같은 많은 연결 세부 정보를 관리합니다. 자세한 내용은 [AWS를 기반으로 구축하기 위한 도구](#)를 참조하세요.
- AWS Cloud Development Kit (AWS CDK) – AWS CDK는 TypeScript, Python, Java, .NET과 같은 친숙한 프로그래밍 언어를 사용하여 AWS 인프라를 정의할 수 있는 오픈 소스 소프트웨어 개발 프레임워크입니다. CDK를 사용하면 애플리케이션 리소스를 모델링한 다음 통합 개발 환경(IDE)에서 직접 CloudFormation을 사용하여 프로비저닝할 수 있습니다. [자세한 내용은 AWS Cloud Development Kit \(AWS CDK\) 단원을 참조하세요.](#)

CloudFormation 콘솔에서 스택 생성

스택 템플릿을 생성한 다음 CloudFormation 콘솔 또는 명령줄 도구를 사용하여 스택을 생성할 수 있습니다. 콘솔은 사전 정의된 옵션이 포함된 마법사 기반 인터페이스를 제공합니다. 그러므로 스택 생성 프로세스가 간소화됩니다.

주제

- [스택 생성](#)
- [스택 옵션 구성](#)
- [스택 구성 미리 보기](#)

스택 생성

이 섹션의 단계에 따라 템플릿을 배포하고 스택을 생성합니다.

사전 조건

- 스택 템플릿을 생성해야 합니다. 자세한 내용은 [CloudFormation 템플릿 작업](#) 섹션을 참조하세요.

스택을 생성하려면(콘솔)

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 화면 상단의 탐색 모음에서 스택을 생성할 AWS 리전을 선택합니다.
3. 스택 페이지에서 오른쪽 상단의 스택 생성을 선택하고 새 리소스 사용(표준)을 선택합니다.

또는 기존 리소스 사용(리소스 가져오기) 옵션을 선택하여 템플릿에 설명된 기존 AWS 리소스를 가져올 수 있습니다. 이 옵션에 대한 자세한 내용은 [CloudFormation 스택에 AWS 리소스 가져오기](#) 섹션을 참조하세요.

4. 스택 생성 페이지에서 다음 중 하나를 수행합니다.
 - 기존 템플릿을 사용하려면 사전 조건 - 템플릿 준비에서 기존 템플릿 선택을 선택합니다. 그런 다음에 템플릿 지정에서 템플릿의 위치에 따라 Amazon S3 URL 또는 템플릿 파일 업로드를 선택합니다.
 - Amazon S3 URL을 선택하는 경우 S3 버킷의 템플릿 파일에 URL을 제공합니다.

템플릿에 중첩 스택(예: 하위 디렉터리에 있는 다른 템플릿 문서에서 설명하는 스택)이 포함되어 있으면 S3 버킷에 필수 파일 및 디렉터리가 있는지 확인합니다.

템플릿이 버전 관리가 활성화된 버킷에 있는 경우 ?versionId=**version-id** 추가를 통해 특정 템플릿 버전을 URL에 지정할 수 있습니다. 버전 관리가 활성화된 버킷에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 [버전 관리가 활성화된 버킷의 객체 작업](#)을 참조하세요.

URL은 사용자에게 읽기 권한이 있는 S3 버킷에 저장된 템플릿(최대 크기 1MB)을 가리켜야 합니다. URL은 최대 1,024자 길이일 수 있습니다. 일부 리소스의 경우 버킷이 스택과 같은 리전에 있어야 할 수 있습니다.

- 템플릿 파일 업로드를 선택하는 경우 파일 선택을 선택하여 로컬 컴퓨터에서 템플릿 파일을 선택합니다. 템플릿 파일 크기는 1MB 이하여야 합니다.

템플릿을 선택하면 CloudFormation이 파일을 업로드하고 S3 URL을 표시합니다.

CloudFormation에서는 이를 AWS 계정의 Amazon S3 버킷에 업로드합니다. CloudFormation에서 생성한 S3 버킷이 AWS 계정에 이미 있는 경우 CloudFormation이 템플릿을 해당 버킷

에 추가합니다. CloudFormation에서 생성한 버킷이 아직 없는 경우 템플릿 파일을 업로드하는 리전마다 고유한 버킷이 생성됩니다.

다음은 CloudFormation에서 생성한 S3 버킷 사용 시 고려 사항입니다.

- AWS 계정의 Amazon S3 권한이 있는 모든 사용자가 버킷에 액세스할 수 있습니다.
- CloudFormation은 기본적으로 서버 측 암호화가 활성화된 버킷을 생성하여 버킷에 저장된 모든 객체를 암호화합니다.

CloudFormation에서 생성한 버킷에 대한 암호화 옵션을 직접 관리할 수 있습니다(예: <https://console.aws.amazon.com/s3/>의 Amazon S3 콘솔 또는 AWS CLI 사용). 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 [Amazon S3 버킷에 대한 기본 서버 측 암호화 동작 설정](#)을 참조하세요.

- 템플릿을 Amazon S3에 수동으로 업로드하여 자체 버킷을 사용하고 버킷의 권한을 관리할 수 있습니다. 스택을 생성하거나 업데이트할 때마다 템플릿 파일의 Amazon S3 URL을 지정합니다.
- 템플릿이 준비되지 않은 경우 Infrastructure Composer에서 빌드를 선택하여 Infrastructure Composer로 템플릿을 생성할 수 있습니다. 자세한 내용은 [Infrastructure Composer](#) 섹션을 참조하세요.

5. 계속하여 템플릿을 검증하려면 다음을 선택합니다.

계속하기 전에 CloudFormation에서 템플릿을 검증하여 구문 및 일부 의미 체계 오류(예: 순환 종속성)를 파악합니다. 확인 중에 CloudFormation에서는 먼저 템플릿이 유효한 JSON인지를 확인합니다. 유효한 JSON이 아니면 CloudFormation은 템플릿이 유효한 YAML인지를 확인합니다. 두 확인이 모두 실패한 경우 CloudFormation에서는 템플릿 확인 오류를 반환합니다.

6. 스택 세부 정보 지정 페이지에서 스택 이름 상자에 스택 이름을 입력합니다.

스택 이름은 스택 목록에서 특정 스택을 찾을 수 있도록 도와주는 식별자입니다. 스택 이름에는 영숫자(대소문자 구분)와 하이픈만 사용할 수 있습니다. 영문자로 시작해야 하고 128자 이하여야 합니다.

7. 파라미터 섹션에서 템플릿에 정의된 파라미터의 값을 지정합니다.

8. 스택 생성을 계속하려면 다음을 선택합니다.

9. (선택 사항) 스택 옵션 구성 페이지에서 기본 스택 옵션을 변경합니다. 자세한 내용은 [스택 옵션 구성](#) 섹션을 참조하세요.

10. 템플릿에 IAM 리소스가 포함되어 있는 경우 기능에서 이 템플릿이 IAM 리소스를 생성할 수 있음을 확인합니다를 선택하여 템플릿에서 IAM 리소스를 사용하도록 지정합니다. 자세한 내용은 [CloudFormation 템플릿에서 IAM 리소스 승인](#) 섹션을 참조하세요.

11. 다음을 선택하여 계속 진행합니다.
12. 검토 및 생성 페이지에서 스택의 세부 정보를 검토합니다.

스택 시작 전에 값을 변경하려면 변경하려는 설정이 있는 섹션에서 편집을 선택합니다.

13. (선택 사항) 변경 세트를 생성하여 스택을 생성하기 전에 해당 구성을 미리 볼 수 있습니다. 검토 및 생성 페이지에서 변경 세트 생성을 선택하고 지침을 따릅니다. 자세한 내용은 [스택 구성 미리 보기](#) 섹션을 참조하세요.
14. 제출을 선택하여 스택을 시작합니다.

그러면 CloudFormation에서는 템플릿에 정의된 모든 리소스 생성을 진행합니다.

새 스택의 이벤트 탭에서 스택 생성 진행 상황과 상태를 모니터링할 수 있습니다. 자세한 내용은 [스택 진행 상황 모니터링](#) 섹션을 참조하세요.

명령줄을 사용하여 스택을 생성하려면

다음 명령 중 하나를 사용할 수 있습니다:

- [create-stack](#) (AWS CLI)
- [New-CFNStack](#) (AWS Tools for Windows PowerShell)

명령줄을 사용하여 스택을 생성하는 예제는 [AWS CLI 및 PowerShell에 대한 CloudFormation 스택 작업 명령의 예](#) 단원을 참조하세요.

스택 옵션 구성

스택 옵션 구성 페이지에서 CloudFormation 스택의 옵션(예: 태그, 스택 이벤트 알림 또는 스택 정책)을 구성할 수 있습니다.

다음과 같은 스택 옵션을 설정할 수 있습니다.

Tags

스택과 CloudFormation에서 태깅을 지원하는 리소스에 태그 키 페어를 50개까지 추가할 수 있습니다. 태그는 비용 추적과 같은 목적으로 AWS 리소스에 할당될 수 있는 고객 정의 키 및 값입니다.

키는 영숫자 또는 공백으로 구성됩니다. 태그 키는 최대 127자일 수 있습니다.

값은 영숫자 또는 공백으로 구성됩니다. 태그 값은 최대 255자일 수 있습니다.

스택 수준 태그를 생성, 추가, 업데이트 또는 제거하면 스택 업데이트가 시작됩니다. 스택 수준 태그 전파를 지원하는 모든 리소스도 그에 따라 업데이트됩니다.

권한

CloudFormation이 수입할 수 있는 기존 IAM 서비스 역할입니다. CloudFormation에서는 사용자의 계정 자격 증명을 사용하는 대신 역할의 자격 증명을 사용하여 스택을 생성합니다. 자세한 내용은 [CloudFormation 서비스 역할](#) 섹션을 참조하세요.

스택 실패 옵션

모든 스택 배포 및 변경 세트 작업에 대한 프로비저닝 실패 옵션을 지정합니다. 자세한 내용은 [리소스 프로비저닝 시 실패 처리 방법 선택](#) 섹션을 참조하세요.

Roll back all stack resources(모든 스택 리소스 롤백) 옵션은 스택 상태가 CREATE_FAILED 또는 UPDATE_FAILED일 때 템플릿에 지정된 모든 리소스를 롤백합니다.

생성 작업의 경우 Preserve successfully provisioned resources(성공적으로 프로비저닝된 리소스 보존) 옵션은 성공한 리소스의 상태를 유지하는 반면 실패한 리소스는 다음 업데이트 작업이 수행될 때까지 실패 상태로 유지됩니다.

업데이트 및 변경 세트 작업의 경우 Preserve successfully provisioned resources(성공적으로 프로비저닝된 리소스 보존) 옵션은 성공한 리소스의 상태를 유지하면서 실패한 리소스를 마지막으로 알려진 안정 상태로 롤백합니다. 실패한 리소스는 UPDATE_FAILED 상태가 됩니다. 마지막으로 알려진 안정 상태가 없는 리소스는 다음 스택 작업 시 삭제됩니다.

스택 생성을 위해 다음과 같은 고급 옵션을 설정할 수도 있습니다.

스택 정책

스택 업데이트 중 의도치 않게 업데이트되지 않도록 하려는 리소스를 정의합니다. 기본적으로 스택 업데이트 중에는 모든 리소스가 업데이트될 수 있습니다.

스택 정책을 바로 JSON으로 입력하거나 스택 정책이 포함된 JSON 파일을 업로드할 수 있습니다. 자세한 내용은 [스택 리소스에 대한 업데이트 방지](#) 섹션을 참조하세요.

롤백 구성

스택 생성 및 업데이트 중에 CloudFormation에서 스택의 상태를 모니터링하도록 하여 지정한 경보의 임계값을 스택에서 위반하는 경우 해당 작업을 롤백할 수 있습니다. CloudFormation에서 모니터링해야 하는 CloudWatch 경보를 지정합니다. 스택 작업 도중 또는 모니터링 기간 동안 ALARM 상태로 전환되는 경보가 하나라도 있을 경우 CloudFormation이 전체 스택 작업을 롤백합니다. 자세한

내용은 [롤백 트리거를 사용하여 경보 위반 시 CloudFormation 스택을 롤백합니다](#) 섹션을 참조하세요.

Notification options(알림 옵션)

스택 이벤트에 대한 알림을 전송하는 기존 또는 새 Amazon Simple Notification Service 주제를 지정할 수 있습니다.

Amazon SNS 주제를 생성하면 스택 이벤트 알림을 보낼 이메일 주소와 이름을 지정해야 합니다.

스택 생성 옵션

다음 옵션은 스택 생성에 포함되지만 스택 업데이트의 일부로 사용할 수 없습니다.

제한 시간

스택 생성 작업 시간이 초과되기 전에 CloudFormation이 할당해야 하는 시간(분)을 지정합니다. CloudFormation이 할당된 시간 내에 전체 스택을 생성하지 못할 경우, 시간 초과로 인해 스택 생성이 실패하고 스택이 롤백됩니다.

기본적으로 스택 생성에는 시간 초과가 없습니다. 하지만 개인 리소스에는 해당 리소스가 구현하는 서비스의 특성에 따라 자체 시간 초과가 설정될 수 있습니다. 예를 들어 스택의 개별 리소스가 시간 초과될 경우, 스택 생성에 대해 지정한 시간 초과 시간이 아직 지나지 않았더라도 스택 생성이 시간 초과됩니다.

종료 방지

스택이 실수로 삭제되는 것을 방지합니다. 종료 방지 기능이 활성화된 스택을 삭제하려고 시도하면 삭제가 실패하고 스택은 상태를 포함하여 변함없이 그대로 유지됩니다. 자세한 내용은 [CloudFormation 스택 삭제 방지](#) 섹션을 참조하세요.

종료 방지 기능은 기본적으로 비활성화됩니다.

스택 구성 미리 보기

스택을 생성하기 이전에 CloudFormation 스택이 어떻게 구성될지 미리 보려면 변경 세트를 생성합니다. 이 기능을 사용하면 다양한 구성을 검사한 다음 변경 세트를 실행하기 이전에 스택을 수정 및 변경할 수 있습니다. 변경 세트에 대한 자세한 내용은 [변경 세트를 사용하여 CloudFormation 스택 업데이트](#) 단원을 참조하십시오.

새 스택에 대한 변경 세트 생성

새 스택에 대한 변경 세트를 만들려면 스택 템플릿을 선택하고 새 스택을 만드는 경우와 같이 스택 구성을 지정한 다음, 새 스택 대신 새 변경 세트를 만들도록 선택합니다.

새 스택에 대한 변경 세트 생성

1. 검토 및 생성 페이지에서 변경 세트 생성을 선택합니다.
2. 변경 세트 생성 대화 상자에서 변경 세트의 이름과 원하는 경우 설명을 입력합니다. 변경 세트 생성을 선택합니다.

새 스택에 대한 변경 세트를 만들면 CloudFormation이 다음을 수행합니다.

- 상태가 REVIEW_IN_PROGRESS인 새 스택을 시작합니다.
- 이전 단계에서 지정한 스택 구성을 반영하는 새 스택에 대해 변경 세트를 만듭니다.

CloudFormation에서 제안한 스택에 대한 [변경 세트(Change sets)] 페이지가 표시됩니다. CloudFormation이 변경 세트를 생성하는 동안 상태는 CREATE_IN_PROGRESS이고 실행 상태는 UNAVAILABLE입니다. CloudFormation이 변경 세트 생성을 성공적으로 완료하면 변경 세트 상태는 CREATE_COMPLETE로 설정되고 실행 상태는 AVAILABLE입니다. 스택 상태가 REVIEW_IN_PROGRESS로 업데이트됩니다. 이 시점에 변경 세트를 실행하여 새 스택 생성을 완료할 수 있습니다.

CloudFormation의 [변경(Changes)] 창에 제안된 스택 구성이 표시됩니다.

CloudFormation이 변경 세트 생성에 실패하면, 변경 세트 상태는 CREATE_FAILED로 설정됩니다. 상태 사유 필드에 표시된 오류를 수정한 다음 새 변경 세트를 만듭니다. 이 단계에서 다양한 구성을 시도한 후 다음 변경 세트를 실행하기 이전에 스택을 수정 및 변경할 수 있습니다.

3. 변경 세트를 기반으로 새 스택 생성을 완료하려면 Execute(실행)를 선택하고 롤백 구성을 지정한 다음 Execute change set(변경 세트 실행)를 선택합니다.

CloudFormation 콘솔에서 스택 정보 보기

CloudFormation 스택을 생성한 후에는 AWS Management Console을 사용하여 스택의 데이터 및 리소스를 볼 수 있습니다.

CloudFormation 스택에 대한 정보를 보려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 스택이 위치한 AWS 리전을 선택합니다.

3. 스택 페이지에서 실패한 스택 이름을 선택합니다. CloudFormation에 선택한 스택에 대한 스택 세부 정보가 표시됩니다.

Note

스택이 삭제된 경우 필터 상태 옵션을 사용하여 스택을 찾을 수 있습니다. 자세한 내용은 [CloudFormation 콘솔에서 삭제된 스택 보기](#) 섹션을 참조하세요.

다음 스택 정보를 볼 수 있습니다.

스택 정보

다음은 포함하여 스택 및 해당 구성에 대한 일반 정보를 표시합니다.

개요

스택 상태, 드리프트 상태 및 종료 방지와 같은 상태 정보와 함께 스택 이름, 스택 ID, 루트 스택 및 [IAM 역할](#)을 표시합니다.

Tags

스택과 연결된 태그를 모두 표시합니다.

스택 정책

스택 업데이트 시 업데이트되지 않는 스택 리소스를 설명합니다. 이러한 리소스를 업데이트하려면 스택 업데이트 중 명시적으로 허용되어야 합니다. 자세한 내용은 [스택 리소스에 대한 업데이트 방지](#) 섹션을 참조하세요.

롤백 구성

스택 작업 중 또는 지정된 모니터링 기간 동안 CloudFormation이 모니터링해야 하는, 지정된 CloudWatch 경보를 표시합니다. 스택 작업 도중 또는 모니터링 기간 동안 ALARM 상태로 전환되는 경보가 하나라도 있을 경우 CloudFormation이 전체 스택 작업을 롤백합니다. 자세한 내용은 [롤백 트리거를 사용하여 경보 위반 시 CloudFormation 스택을 롤백합니다.](#) 섹션을 참조하세요.

Notification options(알림 옵션)

스택 이벤트에 대한 알림이 전송되는 Amazon Simple Notification Service 주제를 표시합니다 (지정된 경우).

이벤트

스택을 생성, 업데이트 또는 삭제할 때 추적되는 작업을 표시합니다. 자세한 내용은 [스택 진행 상황 모니터링](#) 섹션을 참조하세요.

지정된 스택 작업에 의해 트리거되는 모든 이벤트는 동일한 클라이언트 요청 토큰에 할당됩니다. 이 토큰은 작업을 추적하는 데 사용할 수 있습니다. 콘솔에서 시작되는 스택 작업은 Console-StackOperation-ID 토큰 형식을 사용합니다. 이러한 형식은 스택 작업을 쉽게 식별하는 데 도움이 됩니다. 예를 들어, 콘솔을 사용하여 스택을 생성할 경우, 각 결과 스택 이벤트는 Console-CreateStack-7f59c3cf-00d2-40c7-b2ff-e75db0987002 형식의 동일한 토큰에 할당됩니다.

리소스

스택의 일부인 리소스를 표시합니다.

출력

스택의 템플릿에서 선언된 출력을 표시합니다. 자세한 내용은 [배포된 CloudFormation 스택에서 내보낸 출력 가져오기](#) 섹션을 참조하세요.

파라미터

스택의 파라미터와 그 값을 표시합니다.

Systems Manager 파라미터가 포함된 스택의 경우, 확인된 값 옆에 Systems Manager 파라미터에 대한 스택 정의에 사용된 값이 표시됩니다. 자세한 내용은 [CloudFormation에서 제공하는 파라미터 유형을 사용하여 런타임 시 기존 리소스 지정](#) 섹션을 참조하세요.

템플릿

스택의 템플릿을 표시합니다.

매크로가 포함된 스택의 경우 View original template(원본 템플릿 보기)을 선택하여 사용자가 제출한 템플릿을 보거나 View processed template(처리된 템플릿 보기)를 선택하여 CloudFormation에서 참조된 매크로를 처리한 후 템플릿을 볼 수 있습니다. CloudFormation은 처리된 템플릿을 사용하여 스택을 생성하거나 업데이트합니다.

변경 세트

스택의 변경 세트를 표시합니다.

자세한 내용은 [CloudFormation 스택에 대한 변경 세트 보기](#) 섹션을 참조하세요.

Git 동기화

스택의 Git 동기화 대시보드를 표시합니다.

자세한 내용은 [Git 동기화 상태 대시보드](#) 섹션을 참조하세요.

스택 템플릿 업데이트

CloudFormation 스택의 리소스 또는 속성을 수정하려면 스택의 템플릿을 업데이트해야 합니다. 해당 스택의 기존 템플릿으로 시작하여 변경 사항을 적용합니다. 소스 제어 시스템에 저장된 템플릿이 있으면 해당 템플릿의 사본을 시작 지점으로 사용합니다. 아니면 CloudFormation에서 템플릿의 사본을 가져올 수 있습니다.

스택의 파라미터 또는 설정(예: 스택의 Amazon SNS 주제)만 변경하려면 사본을 가져오지 않고 기존 템플릿을 재사용할 수 있습니다.

[Infrastructure Composer](#) 또는 텍스트 편집기를 사용하여 CloudFormation 스택 템플릿을 업데이트할 수 있습니다.

Infrastructure Composer를 사용하여 기존 스택 템플릿을 업데이트하려면 다음을 수행하세요.

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 스택 페이지에서 업데이트하려는 스택 이름을 선택하세요.
3. 템플릿 탭을 선택한 다음 인프라 컴포저에서 보기를 선택하세요.

CloudFormation에서는 Infrastructure Composer에서 템플릿을 엽니다.

4. 다음 방법 중 하나를 사용하여 템플릿을 업데이트하세요.
 - 캔버스 인터페이스: 여기서 리소스 팔레트로부터 끌어서 놓을 수 있습니다. 카드를 두 번 클릭하여 리소스 속성 패널을 열어 리소스를 구성하세요. 필요에 따라 리소스를 연결하세요. Canvas 인터페이스 사용에 대한 자세한 지침은 [How to compose in AWS 인프라 컴포저](#)를 참조하세요.
 - 템플릿 인터페이스: 캔버스에서 템플릿 인터페이스로 전환합니다. 템플릿 코드에 대한 인라인 업데이트를 수행하세요. 필요에 따라 JSON 및 YAML 형식 간 전환을 수행하세요.
5. 검증을 선택하여 템플릿에서 구문 오류가 있는지 확인하세요.
6. 변경 사항을 CloudFormation으로 내보낼 준비가 되면 템플릿 업데이트를 선택하세요.

AWS CLI를 사용하여 기존 스택 템플릿을 업데이트하는 방법

1. 업데이트할 스택의 템플릿을 가져오려면 [get-template](#) CLI 명령을 사용합니다.
2. 템플릿을 복사하고 텍스트 파일에 붙여 넣은 다음 수정한 후 저장합니다. 템플릿만 복사합니다. 이 명령에서는 템플릿이 따옴표 안에 지정되지만, 따옴표는 복사하지 마십시오. 템플릿 자체는 열기 괄호에서 시작하여 닫기 괄호로 끝납니다. 이 파일에 스택 리소스에 대한 변경 내용을 지정합니다.

템플릿에 변경 사항을 적용할 때는 다음 사항에 유의하세요.

- 업데이트를 지원하지 않는 리소스에서 사용되는 파라미터를 추가, 수정 또는 삭제할 수 없습니다.
- 대부분 리소스의 경우, 리소스의 논리적 이름을 변경하면 해당 리소스가 삭제되고 새 리소스로 대체됩니다. 이름이 변경된 리소스에 종속된 다른 리소스는 모두 업데이트해야 하며 대체될 수도 있습니다. 그 밖의 리소스는 업데이트를 시작하기 위해 (논리적 이름뿐만 아니라) 속성도 업데이트해야 합니다.
- 일부 리소스에는 특정 속성에 설정할 수 있는 값에 대한 제약 조건이 있을 수 있습니다. 예를 들어, RDS 데이터베이스 인스턴스의 `AllocatedStorage` 속성에 대한 변경은 현재 값보다 커야 합니다. 업데이트에서 이러한 규칙을 위반하면 해당 부분이 실패합니다.
- 하나의 리소스를 업데이트하면 해당 리소스를 참조하는 다른 리소스에도 영향을 미칠 수 있습니다. [Ref 함수](#) 또는 [Fn::GetAtt 함수](#) 같은 함수를 사용하여 다른 리소스를 기반으로 속성을 설정하는 경우 CloudFormation에서는 참조된 리소스가 변경되면 참조하는 리소스도 업데이트합니다.
- 특정 리소스 속성 업데이트에 따른 영향에 대한 자세한 내용은 [AWS 리소스 및 속성 유형 참조](#)를 참조하세요. 각 속성마다 업데이트의 영향은 다음 중 하나입니다.
 - 업데이트 필요조건: [무중단](#)
 - 업데이트 필요조건: [일부 중단](#)
 - 업데이트 필요: [대체](#)
- [validate-template](#) CLI 명령을 사용하거나 콘솔에서 템플릿을 지정하여 템플릿의 JSON 또는 YAML 구문을 확인할 수 있습니다. 콘솔은 유효성 검사를 자동으로 수행합니다. 하지만 이러한 방법은 템플릿의 구문만 확인하고 리소스에 지정한 속성 값이 해당 리소스에 유효한지는 확인하지 않습니다. 더 복잡한 유효성 검사 또는 모범 사례 검사의 경우 [CloudFormation Linter\(cfn-lint\)](#) 및 [CloudFormation Rain\(rain fmt\)](#) 같은 추가 도구를 사용할 수 있습니다.

Note

가끔 CloudFormation에서는 적용하려는 특정 변경 사항을 허용하지 않고, 변경이 허용되지 않는다고 알려줍니다. 이 메시는 비동기적으로 발생할 수 있지만, 리소스가 CloudFormation에 의해 기본적으로 비확정적인 순서에 따라 생성 및 업데이트됩니다.

스택 리소스의 업데이트 동작 이해

업데이트를 제출하면 CloudFormation에서는 제출하는 내용과 스택의 현재 템플릿 간의 차이에 따라 리소스를 업데이트합니다. 변경되지 않은 리소스는 업데이트 프로세스 동안 중단 없이 실행됩니다. 업데이트한 리소스의 경우, CloudFormation에서는 다음 업데이트 동작 중 하나를 사용합니다.

업데이트(무중단)

CloudFormation에서는 해당 리소스의 작동을 중단하지 않고 리소스의 물리적 ID를 변경하지 않는 상태에서 리소스를 업데이트합니다. 예를 들면 [AWS::CloudTrail::Trail](#) 리소스에 대해 특정 속성을 업데이트하는 경우 CloudFormation에서 추적 파일이 중단 없이 업데이트됩니다.

업데이트(중단)

CloudFormation이 리소스를 업데이트하지만, 다소 중단이 발생합니다. 예를 들면 [AWS::EC2::Instance](#) 리소스에 대해 특정 속성을 업데이트하는 경우 CloudFormation 및 Amazon EC2에서 인스턴스를 재구성하는 동안 해당 인스턴스가 중단될 수도 있습니다.

대체

CloudFormation에서는 업데이트 도중 리소스를 다시 생성하며, 이때 새 물리적 ID도 생성합니다. 일반적으로 CloudFormation에서는 리소스를 먼저 생성한 후, 대체 리소스를 가리키도록 다른 종속 리소스의 참조를 변경한 다음, 이전 리소스를 삭제합니다. 예를 들면 [AWS::EC2::DBInstance](#) 리소스 유형의 AvailabilityZone 속성을 업데이트하는 경우, CloudFormation에서는 새 리소스를 생성하고 현재의 EC2 인스턴스 리소스를 새 리소스로 대체합니다.

교체가 필요한 속성을 추가하거나 제거하는 경우 업데이트도 트리거됩니다. 업데이트는 속성의 실제 값이 변경되지 않는 경우에도 발생합니다.

CloudFormation에서 사용하는 메서드는 해당 리소스 유형에 대해 업데이트하는 속성에 따라 달라집니다. 각 속성에 대한 업데이트 동작은 [AWS 리소스 및 속성 유형 참조](#)에 설명되어 있습니다.

업데이트 동작에 따라 이러한 변경 내용이 애플리케이션에 미치는 영향을 줄이도록 리소스를 수정해야 하는 경우를 설정할 수 있습니다. 특히 업데이트 중에 리소스를 대체해야 하는 경우를 계획할 수 있습니다. 예를 들면 [AWS::RDS::DBInstance](#) 리소스 유형의 Port 속성을 업데이트하는 경우, CloudFormation에서는 업데이트된 포트 설정을 사용하여 새 DB 인스턴스를 생성함으로써 DB 인스턴스를 대체하고 이전 DB 인스턴스를 삭제합니다. 업데이트를 수행하기 전에 다음을 수행하여 데이터베이스 대체를 준비할 계획을 세워야 합니다.

- 현재 데이터베이스의 스냅샷을 생성합니다.
- DB 인스턴스를 바꾸는 동안 해당 인스턴스를 사용하는 애플리케이션에서 종단을 처리할 방법에 대한 전략을 준비합니다.
- 해당 DB 인스턴스를 사용하는 애플리케이션에서 업데이트된 포트 설정과 수행된 기타 업데이트를 고려하는지 확인합니다.
- DB 스냅샷을 사용하여 새 DB 인스턴스에서 데이터베이스를 복원합니다.

이 예제는 포괄적이지 않지만, 업데이트 중에 리소스가 교체되는 경우를 대비하여 계획해야 할 사항을 알려주기 위한 목적으로 제공되었습니다.

Note

템플릿에 하나 이상의 [중첩 스택](#)이 포함되는 경우 CloudFormation에서는 모든 중첩 스택에 대해서도 업데이트를 시작합니다. 이 작업은 중첩된 스택이 수정되었는지 여부를 확인하기 위해 필요합니다. CloudFormation은 중첩된 스택에서 해당 템플릿에 지정된 변경 사항이 있는 리소스만 업데이트합니다.

변경 세트를 사용하여 CloudFormation 스택 업데이트

스택을 업데이트해야 하는 경우 변경 사항을 구현하기 전에 변경 사항이 실행 중인 리소스에 미치는 영향에 대해 이해하면 확신을 가지고 스택을 업데이트할 수 있습니다. 변경 세트를 사용하면 제안된 스택 변경 사항이 리소스 속성 및 속성에 미치는 영향을 포함하여 실행 중인 리소스에 어떤 영향을 미칠 수 있는지 미리 확인할 수 있습니다. 변경 사항이 중요한 리소스를 삭제하거나 바꿀지 여부를 결정하고, 변경 세트를 실행하도록 결정한 경우에만 CloudFormation에서 스택을 변경하고, 제안된 변경 사항을 진행하거나 다른 변경 세트를 만들어 다른 변경 사항을 탐색할 수 있습니다. CloudFormation 콘솔, AWS CLI 또는 CloudFormation API를 사용하여 변경 세트를 생성하고 관리할 수 있습니다.

주제

- [CloudFormation 스택에 대한 변경 세트 생성](#)

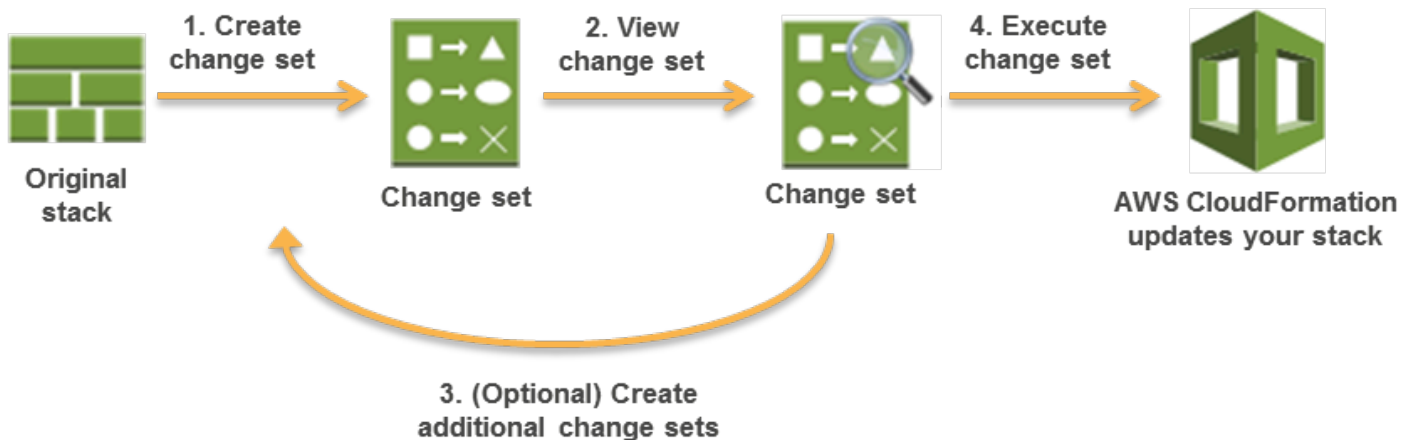
- [CloudFormation 스택에 대한 변경 세트 보기](#)
- [드리프트 인식 변경 세트 사용](#)
- [CloudFormation 스택에 대한 변경 세트 실행](#)
- [CloudFormation 스택에 대한 변경 세트 삭제](#)
- [CloudFormation 스택에 대한 예제 변경 세트](#)
- [중첩 스택에 대한 변경 세트](#)

⚠ Important

변경 세트는 CloudFormation에서 스택 업데이트에 성공했는지 여부는 나타내지 않습니다. 예를 들어, 변경 세트는 계정 할당량을 초과하는 경우, 업데이트를 지원하지 않는 리소스를 업데이트하는 경우 또는 리소스를 수정할 권한이 불충분한 경우를 확인하지 않습니다. 이러한 경우는 모두 스택 업데이트 실패를 일으킬 수 있습니다. 업데이트에 실패한 경우 CloudFormation에서는 리소스를 원래 상태로 롤백하려고 시도합니다.

변경 세트 개요

다음 다이어그램은 변경 세트를 사용하여 스택을 업데이트하는 방법을 요약해서 보여줍니다.



1. 업데이트하려는 스택의 변경 사항을 제출하여 변경 세트를 생성합니다. 수정된 스택 템플릿 또는 수정된 입력 파라미터값을 제출할 수 있습니다. CloudFormation에서는 변경 세트를 생성하기 위해 제출한 변경 사항과 스택을 비교합니다. 이 시점에 스택은 변경되지 않습니다.
2. 변경 세트를 확인하여 어떤 스택 설정 및 리소스가 변경되었는지 확인합니다. 예를 들어 CloudFormation에서 추가, 수정 또는 삭제할 리소스를 확인할 수 있습니다. 또한 CloudFormation이 수정할 리소스 속성 및 속성(예: 태그)의 이전/이후 상태를 비교하여 확인할 수 있습니다.

3. 선택사항: 다른 변경 사항을 고려하는 경우 변경할 사항을 결정하기 전에 추가 변경 세트를 생성합니다. 여러 변경 세트를 생성하면 다른 변경 사항이 리소스 및 속성에 미치는 영향을 이해하고 평가하는 데 도움이 됩니다. 필요한 만큼 변경 세트를 생성할 수 있습니다.
4. 스택에 적용하려는 변경 사항이 포함된 변경 세트를 실행합니다. CloudFormation은 이러한 변경 사항으로 스택을 업데이트합니다.

Note

변경을 실행한 후에는 업데이트된 스택에 변경 세트를 적용할 수 없기 때문에 CloudFormation에서는 스택과 관련된 모든 변경 세트를 제거합니다.

변경 세트를 삭제하면 적용하면 안 되는 변경 세트를 실행하지 않도록 방지할 수 있습니다.

CloudFormation 스택에 대한 변경 세트 생성

실행 중인 스택에 대한 변경 세트를 생성하려면 수정된 템플릿, 새 입력 파라미터값 또는 둘 다를 제공하여 수행하려는 변경 사항을 제출합니다. CloudFormation은 제출한 변경 사항과 스택을 비교하여 변경 세트를 생성합니다.

변경 세트를 생성하기 전에 또는 변경 세트 생성 중에 템플릿을 수정할 수 있습니다.

Create a change set (console)

변경 세트를 생성하려면 다음을 수행하세요.

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 스택 페이지에서 변경 세트를 생성하려는 실행 중인 스택을 선택합니다.
4. 스택 세부 정보 창에서 스택 작업을 선택하고 변경 세트 생성을 선택하세요.
5. Create change set for **stack-name(## ##**에 대한 변경 세트 만들기) 페이지에서 다음 중 하나를 수행하여 입력 파라미터값을 수정하거나 업데이트된 템플릿의 위치를 지정하거나 템플릿을 수정하십시오.

Task	작업
입력 파라미터값을 수정하려면	기존 템플릿 사용을 선택하고 다음을 선택하여 입력 파라미터값을 입력하거나 수정합니다.
업데이트된 템플릿의 위치를 지정하려면	템플릿을 수정한 경우 기존 템플릿 교체를 선택하고 다음 중 하나를 수행하세요. - Amazon S3 버킷에 저장된 템플릿의 경우 Amazon S3 URL을 선택합니다. 템플릿에 대한 URL을 입력하거나 붙여 넣고 다음을 선택합니다. 템플릿이 버전 관리가 활성화된 버킷에 있는 경우 URL에 ?versionId= <i>version-id</i> 를 추가하여 특정 템플릿 버전을 지정할 수 있습니다. 자세한 내용을 알아보려면 Amazon Simple Storage Service 사용 설명서의 버전 관리를 사용하는 버킷의 객체 관리를 참조하세요. - 컴퓨터에 로컬로 저장된 템플릿의 경우 템플릿 파일에 업로드를 선택합니다. [파일 선택(Choose file)]을 선택하고 해당 파일을 찾아 선택한 후 [다음(Next)]을 선택합니다.
템플릿을 수정하려면	템플릿을 수정하지 않은 경우 Infrastructure Composer에서 템플릿 편집을 선택한 다음 Infrastructure Composer에서 편집을 선택합니다. AWS 인프라 컴포저로 리디렉션됩니다. 템플릿을 수정한 후 변경 세트 생성을 선택한 다음 확인 후 CloudFormation 계속을 선택하여 ## ## 에 대한 변경 세트 생성 페이지로 돌아간 후 다음을 선택합니다.

6. 스택 세부 정보 지정 페이지에서 변경 세트의 이름을 지정하고 선택적으로 변경 세트의 설명을 지정하여 개요 섹션에서 스택의 용도를 식별합니다. 템플릿에 파라미터가 포함되어 있는 경우 Specify stack details(스택 세부 정보 지정) 페이지에서 해당 입력 파라미터값을 입력하거나 수정하고 다음을 선택합니다.

스택 템플릿을 재사용할 경우 CloudFormation에서는 NoEcho 속성으로 선언된 파라미터를 제외하고 스택의 현재 값으로 각 파라미터를 채웁니다. 이러한 파라미터에 기존 값을 사용하려면 기존 값 사용을 선택합니다.

동적 파라미터를 사용하여 보안 암호를 관리하는 방법 및 NoEcho를 사용하여 민감한 정보를 마스킹 처리하는 방법에 대한 자세한 내용은 [템플릿에 자격 증명을 포함하지 않음](#) 모범 사례를 참조하세요.

- 구성 스택 옵션 페이지에서 스택 태그, IAM 서비스 역할, 스택 정책, 롤백 구성, Amazon SNS 알림 주제(해당할 경우) 또는 변경 세트를 업데이트합니다.

Note

중첩 스택에 대한 변경 세트는 기본적으로 활성화됩니다. 그러면 템플릿에 지정된 모든 중첩 세트에 대해 변경 세트가 생성됩니다. 현재 스택에 대해서만 변경 세트를 생성하려면 [Disabled(비활성화)]를 선택합니다. 중첩 스택에 대한 변경 세트에 대한 자세한 내용은 [중첩 스택에 대한 변경 세트](#) 섹션을 참조하세요.

- 템플릿에 IAM 리소스가 포함된 경우 기능에서 CloudFormation에서 IAM 리소스를 생성할 수 있음을 확인함을 선택합니다. IAM 리소스는 AWS 계정의 권한을 수정할 수 있습니다. 이러한 리소스를 검토하여 의도한 작업만 허용하는지 확인하세요. 자세한 내용은 [CloudFormation 템플릿에서 IAM 리소스 승인](#) 섹션을 참조하세요.
- 다음을 선택하여 계속 진행합니다.
- ## ## 검토 페이지에서 이 변경 세트의 변경 사항을 검토하십시오.
- 제출을 선택합니다.

변경 세트 세부 정보 페이지의 변경 사항 탭으로 리디렉션됩니다. CloudFormation에서 변경 세트를 생성하는 동안 변경 세트의 상태는 CREATE_PENDING입니다. 변경 세트가 생성된 후 CloudFormation에서 상태가 CREATE_COMPLETE로 설정됩니다. [변경(Changes)] 섹션의 CloudFormation에서 스택에 적용된 모든 변경 사항을 나열합니다. 자세한 내용은 [CloudFormation 스택에 대한 변경 세트 보기](#) 섹션을 참조하세요.

속성 수준에 적용된 변경 사항을 보려면 속성 수준 변경 사항 열의 세부 정보 보기를 선택합니다.

CloudFormation에서 변경 세트 생성에 실패하면(FAILED 상태 보고) [상태(Status)] 필드에 표시된 오류를 수정한 다음 변경 세트를 다시 생성합니다.

- 변경 사항이 올바른지 확인한 후 변경 세트 실행을 선택합니다.

Create a change set for nested stacks (console)

중첩 스택에 대해 변경 세트를 생성하려면 다음을 수행하세요.

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 스택 페이지에서 변경 세트를 생성하려는 실행 중인 스택을 선택합니다.
4. 스택 세부 정보 창에서 스택 작업을 선택하고 변경 세트 생성을 선택하세요.
5. Create change set for **stack-name(## ##**에 대한 변경 세트 만들기) 페이지에서 다음 중 하나를 수행하여 입력 파라미터값을 수정하거나 업데이트된 템플릿의 위치를 지정하거나 템플릿을 수정하십시오.

Task	작업
입력 파라미터값을 수정하려면	기존 템플릿 사용을 선택하고 다음을 선택하여 입력 파라미터값을 입력하거나 수정합니다.
업데이트된 템플릿의 위치를 지정하려면	템플릿을 수정한 경우 기존 템플릿 교체를 선택하고 다음 중 하나를 수행하세요. • Amazon S3 버킷에 저장된 템플릿의 경우 Amazon S3 URL을 선택합니다. 템플릿에 대한 URL을 입력하거나 붙여 넣고 다음을 선택합니다. 템플릿이 버전 관리가 활성화된 버킷에 있는 경우 URL에 ? versionId= version-id 를 추가하여 특정 템플릿 버전을 지정할 수 있습니다. 자세한 내용을 알아보려면 Amazon Simple Storage Service 사용 설명서의 버전 관리를 사용하는 버킷의 객체 관리를 참조하세요. • 컴퓨터에 로컬로 저장된 템플릿의 경우 템플릿 파일에 업로드를 선택합니다. [파일 선택(Choose file)]을 선택하고 해당 파일을 찾아 선택한 후 [다음(Next)]을 선택합니다.
템플릿을 수정하려면	템플릿을 수정하지 않은 경우 Infrastructure Composer에서 템플릿 편집을 선택한 다음 Infrastructure Composer에서 편집을 선택합니다. AWS 인프라 컴포저로 리디렉션됩니다. 템플릿을 수정한 후 변경 세트 생성을 선택한 다음 확인 후 CloudFormation 계속을

Task	작업
	선택하여 ## ## 에 대한 변경 세트 생성 페이지로 돌아간 후 다음을 선택합니다.

6. 스택 세부 정보 지정 페이지에서 변경 세트의 이름을 지정하고 선택적으로 변경 세트의 설명을 지정하여 개요 섹션에서 스택의 용도를 식별합니다. 템플릿에 파라미터가 포함되어 있는 경우 Specify stack details(스택 세부 정보 지정) 페이지에서 해당 입력 파라미터값을 입력하거나 수정하고 다음을 선택합니다.

스택 템플릿을 재사용할 경우 CloudFormation에서는 NoEcho 속성으로 선언된 파라미터를 제외하고 스택의 현재 값으로 각 파라미터를 채웁니다. 이러한 파라미터에 기존 값을 사용하려면 기존 값 사용을 선택합니다.

동적 파라미터를 사용하여 보안 암호를 관리하는 방법 및 NoEcho를 사용하여 민감한 정보를 마스킹 처리하는 방법에 대한 자세한 내용은 [템플릿에 자격 증명을 포함하지 않음](#) 모범 사례를 참조하세요.

7. 구성 스택 옵션 페이지에서 스택 태그, IAM 서비스 역할, 스택 정책, 롤백 구성, Amazon SNS 알림 주제(해당할 경우) 또는 변경 세트를 업데이트합니다. 자세한 내용은 [스택 옵션 구성](#) 섹션을 참조하세요.

Note

중첩 스택에 대한 변경 세트는 기본적으로 활성화됩니다. 그러면 템플릿에 지정된 모든 중첩 세트에 대해 변경 세트가 생성됩니다. 중첩 스택에 대한 변경 세트에 대한 자세한 내용은 [중첩 스택에 대한 변경 세트](#) 섹션을 참조하세요.

8. 템플릿에 IAM 리소스가 포함된 경우 기능에서 CloudFormation에서 IAM 리소스를 생성할 수 있음을 확인함을 선택합니다. IAM 리소스는 AWS 계정의 권한을 수정할 수 있습니다. 이러한 리소스를 검토하여 의도한 작업만 허용하는지 확인하세요. 자세한 내용은 [CloudFormation 템플릿에서 IAM 리소스 승인](#) 섹션을 참조하세요.
9. 다음을 선택하여 계속 진행합니다.
10. **## ##** 검토 페이지에서 이 변경 세트의 변경 사항을 검토하십시오.
11. 제출을 선택합니다.

Note

중첩 스택에 대한 변경 세트를 생성할 때 CloudFormation 속성 수준 변경 세트는 교차 스택 참조를 확인하지 않습니다. 변경 세트는 상위 스택의 출력을 참조하고 상위 스택이 수정된 경우 조건부 교체를 위해 하위 스택의 리소스에 표시를 할 수 있습니다.

변경 세트 세부 정보 페이지의 변경 사항 탭으로 리디렉션됩니다. CloudFormation에서 변경 세트를 생성하는 동안 변경 세트의 상태는 CREATE_PENDING입니다. 변경 세트가 생성된 후 CloudFormation에서 상태가 CREATE_COMPLETE로 설정됩니다. [변경(Changes)] 섹션의 CloudFormation에서 스택에 적용된 모든 변경 사항을 나열합니다. 자세한 내용은 [CloudFormation 스택에 대한 변경 세트 보기](#) 섹션을 참조하세요.

CloudFormation에서 변경 세트 생성에 실패하면(FAILED 상태 보고) [상태(Status)] 필드에 표시된 오류를 수정한 다음 변경 세트를 다시 생성합니다.

12. 변경 사항이 올바른지 확인한 후 변경 세트 실행을 선택합니다.

변경 세트를 생성하려면(AWS CLI)

- [create-change-set](#) 명령을 사용합니다.

변경 사항을 명령 옵션으로 제출합니다. 새 파라미터값, 수정된 템플릿 또는 둘 다를 지정할 수 있습니다. 예를 들어 다음 명령은 SampleChangeSet 스택에 대해 MyStack이라는 변경 세트를 생성합니다. 변경 세트에서 현재 스택의 템플릿을 사용하지만 다음 Purpose 파라미터의 값이 다릅니다.

```
aws cloudformation create-change-set --stack-name MyStack \
  --change-set-name SampleChangeSet --use-previous-template \
  --parameters \
    ParameterKey="InstanceType",UsePreviousValue=true
    ParameterKey="KeyPairName",UsePreviousValue=true
    ParameterKey="Purpose",ParameterValue="production"
```

CloudFormation 스택에 대한 변경 세트 보기

변경 세트를 생성한 후 변경 세트를 실행하기 전에 제안된 변경 사항을 확인할 수 있습니다.

CloudFormation 콘솔, AWS CLI 또는 CloudFormation API를 사용하여 변경 세트를 확인할 수 있습니다.

다. CloudFormation 콘솔에서는 변경 사항 요약 및 자세한 변경 사항 목록을 JSON 형식으로 제공합니다. AWS CLI와 AWS CloudFormation API는 자세한 변경 사항 목록을 JSON 형식으로 반환합니다.

View a change set (console)

변경 세트를 보려면 다음을 수행하세요.

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 스택 페이지에서 확인하려는 변경 세트가 포함된 스택의 이름을 선택합니다.
4. 탐색 창에서 변경 세트를 선택하여 스택의 변경 세트 목록을 봅니다.
5. 확인하고자 하는 변경 세트의 이름을 선택합니다.

CloudFormation 콘솔에서 변경 세트의 세부 정보 페이지로 이동합니다. 여기서 변경 세트 생성 시간, 상태, 변경 세트 생성에 사용된 입력 및 변경 요약을 볼 수 있습니다.

[변경 사항(Changes)] 섹션에서 각 행은 CloudFormation이 추가, 수정 또는 제거할 리소스를 나타냅니다.

- [추가(Add)] – CloudFormation이 스택 템플릿에 리소스를 추가할 때 리소스를 생성합니다.
- [수정(Modify)] – CloudFormation이 스택 템플릿에서 리소스 속성을 변경할 때 리소스를 수정합니다.
- [제거(Remove)] – CloudFormation이 스택 템플릿에서 리소스를 삭제할 때 리소스를 삭제합니다.

Note

수정으로 인해 리소스가 중단되거나 바뀔(다시 생성될) 수 있습니다. 리소스 업데이트 동작에 대한 자세한 내용은 [스택 리소스의 업데이트 동작 이해](#) 단원을 참조하십시오.

특정 변경 사항을 중점적으로 살펴보려면 필터 보기를 사용합니다. 예를 들어, 특정 리소스 유형(예: AWS::EC2::Instance)을 필터링합니다. 특정 리소스를 필터링하려면 논리적 ID 또는 물리적 ID(예: myWebServer 또는 i-123abcd4)를 지정합니다.

6. 리소스에 적용된 속성값 변경 사항을 보려면 변경 섹션에서 속성 수준 변경 사항 열의 세부 정보 보기를 선택합니다.

7. CloudFormation 콘솔이 리소스의 속성 수준 변경 사항 페이지로 안내합니다. 이 페이지에서 변경 세트를 실행하기 전의 리소스 템플릿 구성과 변경 세트를 실행한 후의 템플릿 구성을 확인할 수 있습니다.

속성 수준 변경 사항 섹션 표에는 영향을 받는 속성의 경로, 변경 유형, 이전 값 및 이후 값이 표시됩니다. 표에서 템플릿의 이전 및 이후 보기에서 강조 표시하려는 각 변경 사항의 확인란을 선택하여 속성 수준에서 어떤 변경 사항이 적용되는지 확인할 수 있습니다.

- 추가 - 추가된 속성이 녹색으로 강조 표시됩니다.
- 수정 - 수정된 속성이 파란색으로 강조 표시됩니다.
- 제거 - 제거된 속성이 빨간색으로 강조 표시됩니다.

View a change set for nested stack (console)

중첩 스택에 대한 변경 사항 확인(콘솔)

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 스택 페이지에서 확인하려는 변경 세트가 포함된 스택의 이름을 선택합니다.
4. 탐색 창에서 변경 세트를 선택하여 스택의 변경 세트 목록을 봅니다.
5. 확인하고자 하는 변경 세트의 이름을 선택합니다.

CloudFormation 콘솔에서 변경 세트의 세부 정보 페이지로 이동합니다. 여기서 변경 세트 생성 시간, 상태, 변경 세트 생성에 사용된 입력 및 변경 요약 볼 수 있습니다.

[변경 사항(Changes)] 섹션에서 각 행은 CloudFormation이 추가, 수정, 제거하거나 동적인 상태를 표시할 리소스를 나타냅니다.

- [추가(Add)] – CloudFormation이 스택 템플릿에 리소스를 추가할 때 리소스를 생성합니다.
- [수정(Modify)] – CloudFormation이 스택 템플릿에서 리소스 속성을 변경할 때 리소스를 수정합니다.
- [제거(Remove)] – CloudFormation이 스택 템플릿에서 리소스를 삭제할 때 리소스를 삭제합니다.
- [동적(Dynamic)] – CloudFormation이 중첩 스택 템플릿에서 정확한 리소스 변경 작업을 확인할 수 없습니다.

Note

수정으로 인해 리소스가 중단되거나 바뀔(다시 생성될) 수 있습니다. 리소스 업데이트 동작에 대한 자세한 내용은 [스택 리소스의 업데이트 동작 이해](#) 단원을 참조하십시오.

특정 변경 사항을 중점적으로 살펴보려면 필터 보기를 사용합니다. 예를 들어, 특정 리소스 유형(예: **AWS::CloudFormation::Stack**)을 필터링합니다. 특정 리소스를 필터링하려면 논리적 ID 또는 물리적 ID(예: **DeadLetterQueue** 또는 **NestedStack**)를 지정합니다.

6. 변경 사항 섹션에서 확인하고자 하는 중첩된 변경 세트의 [중첩된 변경 세트 보기]를 선택합니다.

CloudFormation 콘솔에서 중첩된 변경 세트의 세부 정보 페이지로 이동합니다. [루트 변경 세트로 이동]을 선택하여 루트 변경 세트를 확인하거나 [상위 변경 세트 보기]를 선택하여 상위 변경 세트를 확인할 수도 있습니다. 자세한 내용은 [중첩 스택에 대한 변경 세트](#) 단원을 참조하십시오.

Note

중첩 스택에 대한 변경 세트를 생성할 때 CloudFormation 속성 수준 변경 세트는 교차 스택 참조를 확인하지 않습니다. 변경 세트는 상위 스택의 출력을 참조하고 상위 스택이 수정된 경우 조건부 교체를 위해 하위 스택의 리소스에 표시를 할 수 있습니다.

변경 세트를 보려면(AWS CLI)

1. 변경 세트의 ID를 가져오려면 [change-sets](#) 명령을 실행합니다.

다음 예에서처럼 보려는 변경 세트가 있는 스택의 이름을 지정합니다.

```
aws cloudformation list-change-sets --stack-name MyStack
```

CloudFormation에서는 다음과 유사한 변경 세트 목록을 반환합니다.

```
{
  "Summaries": [
    {
```

```

    "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/MyStack/1a2345b6-0000-00a0-a123-00abc0abc000",
    "Status": "CREATE_COMPLETE",
    "ChangeSetName": "SampleChangeSet",
    "CreationTime": "2020-11-18T20:44:05.889Z",
    "StackName": "MyStack",
    "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/SampleChangeSet/1a2345b6-0000-00a0-a123-00abc0abc000"
  },
  {
    "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/MyStack/1a2345b6-0000-00a0-a123-00abc0abc000",
    "Status": "CREATE_COMPLETE",
    "ChangeSetName": "SampleChangeSet-conditional",
    "CreationTime": "2020-11-18T21:15:56.398Z",
    "StackName": "MyStack",
    "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/SampleChangeSet-conditional/1a2345b6-0000-00a0-a123-00abc0abc000"
  },
  {
    "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/MyStack/1a2345b6-0000-00a0-a123-00abc0abc000",
    "Status": "CREATE_COMPLETE",
    "ChangeSetName": "SampleChangeSet-replacement",
    "CreationTime": "2020-11-18T21:03:37.706Z",
    "StackName": "MyStack",
    "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/SampleChangeSet-replacement/1a2345b6-0000-00a0-a123-00abc0abc000"
  }
]
}

```

2. [describe-change-set](#) 명령을 실행하여 보려는 변경 세트의 ID를 지정합니다. 예제:

```

aws cloudformation describe-change-set \
  --change-set-name arn:aws:cloudformation:us-east-1:123456789012:changeSet/SampleChangeSet/1a2345b6-0000-00a0-a123-00abc0abc000

```

CloudFormation에서는 지정된 변경 세트에 대한 정보를 반환합니다.

```
{
```

```
"StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
MyStack/1a2345b6-0000-00a0-a123-00abc0abc000",
"Status": "CREATE_COMPLETE",
"ChangeSetName": "SampleChangeSet-direct",
"Parameters": [
  {
    "ParameterValue": "testing",
    "ParameterKey": "Purpose"
  },
  {
    "ParameterValue": "ellioty-useast1",
    "ParameterKey": "KeyPairName"
  },
  {
    "ParameterValue": "t2.micro",
    "ParameterKey": "InstanceType"
  }
],
"Changes": [
  {
    "ResourceChange": {
      "ResourceType": "AWS::EC2::Instance",
      "PhysicalResourceId": "i-1abc23d4",
      "Details": [
        {
          "ChangeSource": "DirectModification",
          "Evaluation": "Static",
          "Target": {
            "Attribute": "Tags",
            "RequiresRecreation": "Never"
          }
        }
      ],
      "Action": "Modify",
      "Scope": [
        "Tags"
      ],
      "LogicalResourceId": "MyEC2Instance",
      "Replacement": "False"
    },
    "Type": "Resource"
  }
],
"CreationTime": "2020-11-18T23:35:25.813Z",
```

```

    "Capabilities": [],
    "StackName": "MyStack",
    "NotificationARNs": [],
    "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/
SampleChangeSet-direct/9edde307-960d-4e6e-ad66-b09ea2f20255"
}

```

속성 수준 변경 사항을 나열하려면 `describe-change-set`과 함께 `--include-property-values`를 사용합니다.

Changes 키는 리소스에 대한 변경 사항을 나열합니다. 이 변경 세트를 실행하면 CloudFormation에서는 `i-1abc23d4` EC2 인스턴스의 태그를 업데이트해야 합니다. 각 필드의 설명은 AWS CloudFormation API 참조의 [Change](#) 데이터 유형을 참조하세요.

변경 세트에 대한 예를 더 보려면 [CloudFormation 스택에 대한 예제 변경 세트](#) 단원을 참조하십시오.

변경 세트의 속성 수준 변경 사항을 보려면(AWS CLI)

- 다음 명령은 `AWS::EC2::NetworkInterface` 리소스의 변경 세트와 관련된 속성 수준 변경 사항을 나열합니다. 이 변경 세트에서는 `Ipv4Prefixes` 속성을 제거하고 리소스의 `Description`을 수정하고 `Tag`를 추가합니다.

```

aws cloudformation describe-change-set --include-property-values \
  --change-set-name arn:aws:cloudformation:us-east-1:123456789012:changeSet/
ExampleChangeSet/9f7b541b-126b-44f7-998e-932174557841

```

출력의 예제는 다음과 같습니다.

```

"ChangeSetName": "ExampleChangeSet",
  "ChangeSetId": "arn:aws:cloudformation:us-east-1:803642222207:changeSet/
ExampleChangeSet/9f7b541b-126b-44f7-998e-932174557841",
  "StackId": "arn:aws:cloudformation:us-east-1:803642222207:stack/ExampleStack/
ab664180-f686-11ee-9e29-12cd92393671",
  "StackName": "ExampleStack",
  "Description": null,
  "Parameters": null,
  "CreationTime": "2024-04-09T18:04:59.935000+00:00",
  "ExecutionStatus": "AVAILABLE",
  "Status": "CREATE_COMPLETE",

```



```

    "StatusReason": null,
    "NotificationARNs": [],
    "RollbackConfiguration": {
        "RollbackTriggers": []
    },
    "Capabilities": [],
    "Tags": null,
    "ParentChangeSetId": null,
    "IncludeNestedStacks": true,
    "RootChangeSetId": null,
    "OnStackFailure": null,
  {
    "Changes": [
      {
        "Type": "Resource",
        "ResourceChange": {
          "Action": "Modify",
          "LogicalResourceId":
"EC2NetworkInterface00eni067fd35b649a05b7100Tpyls",
          "PhysicalResourceId": "eni-067fd35b649a05b71",
          "ResourceType": "AWS::EC2::NetworkInterface",
          "Replacement": "False",
          "Scope": [
            "Properties",
            "Tags"
          ],
          "Details": [
            {
              "Target": {
                "Attribute": "Properties",
                "Name": "Ipv4Prefixes",
                "RequiresRecreation": "Never",
                "Path": "/Properties/Ipv4Prefixes",
                "BeforeValue": "[]",
                "AttributeChangeType": "Remove"
              },
              "Evaluation": "Static",
              "ChangeSource": "DirectModification"
            },
            {
              "Target": {
                "Attribute": "Properties",
                "Name": "Description",
                "RequiresRecreation": "Never",

```

```

        "Path": "/Properties/Description",
        "BeforeValue": "",
        "AfterValue": "Description",
        "AttributeChangeType": "Modify"
    },
    "Evaluation": "Static",
    "ChangeSource": "DirectModification"
},
{
    "Target": {
        "Attribute": "Tags",
        "RequiresRecreation": "Never",
        "Path": "/Properties/Tags/0",
        "AfterValue": "{\"Key\": \"Test\", \"Value\": \"Test\"}",
        "AttributeChangeType": "Add"
    },
    "Evaluation": "Static",
    "ChangeSource": "DirectModification"
}
],
    "BeforeContext": "{\"Properties\": {\"Description\": \"\",
    \"PrivateIpAddress\": \"172.31.76.2\", \"PrivateIpAddresses\": [{\"PrivateIpAddress
    \": \"172.31.76.2\", \"Primary\": \"true\"}], \"SecondaryPrivateIpAddressCount\": \"0\",
    \"Ipv6PrefixCount\": \"0\", \"Ipv4Prefixes\": [], \"Ipv4PrefixCount\": \"0\", \"GroupSet
    \": [\"sg-05a45689b1059e82d\"], \"Ipv6Prefixes\": [], \"SubnetId\": \"subnet-455e8969\",
    \"SourceDestCheck\": \"true\", \"InterfaceType\": \"interface\", \"Tags\": []},
    \"UpdateReplacePolicy\": \"Retain\", \"DeletionPolicy\": \"Retain\"}",
    "AfterContext": "{\"Properties\": {\"Description\": \"Description\",
    \"PrivateIpAddress\": \"172.31.76.2\", \"PrivateIpAddresses\": [{\"PrivateIpAddress
    \": \"172.31.76.2\", \"Primary\": \"true\"}], \"SecondaryPrivateIpAddressCount
    \": \"0\", \"Ipv6PrefixCount\": \"0\", \"Ipv4PrefixCount\": \"0\", \"GroupSet\":
    [\"sg-05a45689b1059e82d\"], \"Ipv6Prefixes\": [], \"SubnetId\": \"subnet-455e8969\",
    \"SourceDestCheck\": \"true\", \"InterfaceType\": \"interface\", \"Tags\": [{\"Value\":
    \"Test\", \"Key\": \"Test\"}]}, \"UpdateReplacePolicy\": \"Retain\", \"DeletionPolicy\":
    \"Retain\"}"
    }
},
    "ChangeSetName": "ExampleChangeSet",
    "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/
    ExampleChangeSet/9f7b541b-126b-44f7-998e-932174557841",
    "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/ExampleStack/
    ab664180-f686-11ee-9e29-12cd92393671",

```

드리프트 인식 변경 세트 사용

드리프트 인식 변경 세트는 스택 드리프트를 안전하게 식별 및 관리할 수 있는 향상된 CloudFormation 변경 세트입니다. 기본 서비스의 AWS Management Console, AWS CLI 또는 SDK를 사용하여 CloudFormation 외부에서 스택 리소스가 변경될 때 스택이 예상 템플릿 구성에서 드리프트됩니다. 드리프트 인식 변경 세트는 템플릿을 스택 리소스의 실제 상태와 비교하고 드리프트된 리소스를 템플릿 정의에 맞게 조정합니다. 리소스의 템플릿 정의가 실제 상태와 일치하도록 업데이트되면 드리프트 인식 변경 세트는 리소스를 수정하지 않고 리소스의 드리프트 상태를 재설정합니다.

드리프트 인식 변경 세트를 사용하면 다음과 같은 이점이 있습니다.

- 드리프트의 미리 보기 덮어쓰기 - 배포가 인시던트 응답 중에 수행된 중요한 대역 외 변경 내용을 되돌릴지를 감지합니다.
- 체계적 드리프트 조정 - 드리프트된 속성을 템플릿 값으로 덮어쓰거나 실제 리소스 상태와 일치하도록 템플릿을 업데이트하여 드리프트된 리소스를 규정 준수 템플릿 정의에 맞게 안전하게 조정합니다.
- 향상된 롤백 기능 - 배포에 실패하면 리소스를 이전 템플릿 상태가 아닌 실제 사전 배포 상태로 복원합니다.
- 완전한 가시성 - 변경하기 전에 배포가 실제 리소스 구성에 미치는 영향을 정확하게 확인합니다.

기존 변경 세트는 새 템플릿을 스택의 이전 템플릿과 비교하지만 스택 드리프트는 고려하지 않습니다. 드리프트 인식 변경 세트는 다음을 3방향 비교를 통해 이 문제를 해결합니다.

- 실제 상태 - 리소스의 라이브 구성. CloudFormation은 기본 서비스 API를 통해 변경 세트 생성 시 계정에서 리소스 구성을 읽습니다.
- 이전 배포 상태 - 마지막 CloudFormation 배포의 템플릿에 정의된 구성.
- 원하는 상태 - 새 템플릿에 정의된 구성.

드리프트 인식 변경 세트는 리소스가 템플릿에서 명시적으로 변경되지 않은 경우에도 원하는 상태와 일치하도록 모든 스택 리소스의 실제 상태를 업데이트합니다.

고려 사항

- 리소스 유형 지원 - 드리프트 인식 변경 세트는 원하는 상태를 수백 개의 리소스 유형에 대한 실제 상태와 비교할 수 있도록 지원합니다. 지원되지 않는 리소스 유형의 경우 드리프트 인식 변경 세트는 이전 배포 상태를 원하는 상태와 비교하는 기능으로 대체됩니다. 자세한 내용은 지원되는 리소스 유형 섹션을 참조하세요.

- 쓰기 전용 속성 - 민감한 데이터(암호, 보안 암호)가 포함된 속성의 경우 드리프트 인식 변경 세트는 실제 값이 아닌 이전 배포 값과 비교됩니다.
- AWS 관리형 속성 - 드리프트 인식 변경 세트는 Auto Scaling 그룹의 원하는 용량과 같이 관리형 속성에 대해 AWS 서비스에서 수행한 대역 외 변경 내용을 보존합니다. 자세한 내용은 AWS 관리형 속성 섹션을 참조하세요.
- 외부 태그 키 - 드리프트 인식 변경 세트는 템플릿에 지정되지 않은 태그 키를 제거하거나 수정하지 않으므로 속성 기반 액세스 제어(ABAC) 시스템과의 충돌을 방지합니다.
- 드리프트된 리소스 교체 - 드리프트 인식 변경 세트는 변경 불가능한 속성에 대한 드리프트 조정을 지원하지 않습니다.
- 교차 스택 연결 - `AWS::IAM::Policy`와 같은 일부 리소스 유형은 `AWS::IAM::Role`과 같은 다른 리소스 유형의 실제 상태에 영향을 미칠 수 있습니다. 드리프트 인식 변경 세트는 스택 내 리소스 연결을 처리합니다. 리소스가 다른 스택의 보조 리소스 연결을 통해 수정되는 경우 드리프트 인식 변경 세트는 수정을 드리프트로 감지하고 연결을 되돌릴 수 있습니다. 자주 사용되는 연결 가능한 리소스로, `AWS::IAM::Policy`, `AWS::IAM::ManagedPolicy`, `AWS::EC2::SecurityGroupIngress`, `AWS::EC2::SecurityGroupEgress`가 있습니다.

AWS 관리형 속성

활성 AWS 관리를 위해 특정 리소스 속성을 구성할 수 있습니다. 예를 들어 Amazon Relational Database Service(Amazon RDS)에서 Amazon RDS 테이블의 마이너 엔진 버전을 자동으로 업그레이드하도록 허용할 수 있습니다. 이러한 변경 내용은 CloudFormation에 스택 드리프트로 표시될 수 있습니다. 드리프트 인식 변경 세트는 AWS 관리형 속성에 대해 드리프트가 예상됨을 인식하고 템플릿에서 속성을 수정하지 않은 경우 실제 값을 그대로 둡니다. 다음은 AWS 관리형 속성의 주요 예제입니다.

- Amazon RDS 테이블의 `AutoMinorVersionUpgrade` 속성을 활성화하여 엔진 버전의 자동 업데이트 허용.
- `AWS::ApplicationAutoScaling::ScalableTarget` 리소스를 사용하여 Amazon DynamoDB 테이블의 읽기/쓰기 용량 단위 및 원하는 Amazon Elastic Container Service 클러스터 수와 같은 속성에 대한 자동 오토 스케일링 활성화.
- Auto Scaling 그룹에 대한 `AWS::AutoScaling::ScalingPolicy` 사용.

드리프트 인식 변경 세트는 AWS 관리형으로 식별된 속성을 명확히 합니다. 자세한 내용은 AWS CLI 섹션을 참조하세요.

드리프트 인식 변경 세트 사용(콘솔)

배포 모드에 대한 추가 옵션과 함께 기존 변경 세트와 동일한 워크플로를 사용하여 CloudFormation 콘솔을 통해 드리프트 인식 변경 세트를 생성 및 관리할 수 있습니다.

드리프트 인식 변경 세트 생성

드리프트 인식 변경 세트를 생성하는 방법(콘솔)

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 스택 페이지에서 업데이트하려는 스택을 선택합니다.
3. 스택 작업을 선택한 후 현재 스택에 대한 변경 세트 생성을 선택하세요.
4. 변경 세트 생성 페이지의 변경 세트 유형에서 드리프트 인식 변경 세트를 선택하세요.
5. 변경 세트 생성을 선택합니다.
6. 변경 세트 세부 정보 페이지에서 영향을 받는 각 리소스의 실제, 이전 배포 및 원하는 상태를 보여주는 세 가지 방향의 비교를 검토하세요.
7. 변경 내용에 만족하면 변경 세트 실행을 선택하세요.

드리프트 인식 변경 세트 검토

콘솔에서 드리프트 인식 변경 세트를 보면 기존 변경 세트에 비해 향상된 정보가 표시됩니다.

- 스택 드리프트 상태 - 스택이 마지막 배포에서 드리프트되었는지를 나타냅니다.
- 속성 비교 - 영향을 받는 리소스의 실제 상태와 원하는 상태 간 JSON 차이를 표시합니다.
- 드리프트 표시기 - 드리프트된 JSON 차이 내 속성을 명확하게 표시합니다. 드리프트 보기를 클릭하여 속성의 이전 배포 값을 확인합니다.
- 값 소스 표시기 - 속성의 이전 값이 실제 상태에 기반하는지, 이전 배포 상태에 기반하는지를 표시합니다.

드리프트 인식 변경 세트 사용(AWS CLI)

create-change-set 명령에 --deployment-mode REVERT_DRIFT 파라미터를 추가함으로써 AWS CLI를 사용하여 드리프트 인식 변경 세트를 생성하고 관리할 수 있습니다.

드리프트 인식 변경 세트 생성

드리프트 인식 변경 세트를 생성하는 방법

[create-change-set](#) 명령을 `--deployment-mode REVERT_DRIFT` 파라미터와 함께 사용하여 드리프트 인식 변경 세트를 생성합니다.

```
aws cloudformation create-change-set \  
  --stack-name my-stack \  
  --change-set-name my-drift-aware-changeset \  
  --template-body file://updated-template.yaml \  
  --deployment-mode REVERT_DRIFT \  
  --capabilities CAPABILITY_IAM
```

드리프트 인식 변경 세트 검토

드리프트 인식 변경 세트의 세부 정보를 검토하는 방법

[describe-change-set](#) 명령을 사용하여 드리프트 인식 변경 세트의 세부 정보를 검토합니다.

```
aws cloudformation describe-change-set \  
  --change-set-name my-drift-aware-changeset \  
  --stack-name my-stack
```

응답에는 드리프트 인식 변경 세트에 대한 향상된 정보가 포함됩니다.

- `StackDriftStatus` - 스택의 드리프트 여부를 표시합니다(DRIFTED, IN_SYNC, NOT_CHECKED 또는 UNKNOWN).
- `ResourceDriftStatus` - 각 리소스의 드리프트 상태를 표시합니다(DELETED, MODIFIED, IN_SYNC 또는 NOT_CHECKED).
- `BeforeValueFrom` - 속성의 이전 값이 ACTUAL_STATE 또는 PREVIOUS_DEPLOYMENT_STATE에서 비롯되는지를 나타냅니다.
- `Drift` - `PreviousValue`, `ActualValue`, `DriftDetectionTimestamp`를 비롯한 속성에 대한 드리프트 세부 정보를 포함합니다.
- `ResourceDriftIgnoredProperties` - 변경 세트가 드리프트를 되돌리지 않는 리소스의 속성과 드리프트를 무시하는 이유를 포함합니다.

자세한 내용은 AWS CloudFormation API 참조의 [DescribeChangeSet](#)를 참조하세요.

드리프트 인식 변경 세트 실행

드리프트 인식 변경 세트를 실행하는 방법

변경 세트를 검토한 후 [execute-change-set](#) 명령을 사용하여 변경 내용을 적용합니다.

```
aws cloudformation execute-change-set \  
  --change-set-name my-drift-aware-changeset \  
  --stack-name my-stack
```

배포에 실패하면 CloudFormation은 리소스를 실제 사전 배포 상태로 롤백하고 마지막 배포 이후 변경된 대역 외 변경 내용을 유지합니다.

지원되는 리소스 유형

드리프트 인식 변경 세트는 다음 리소스 유형을 제외하고 드리프트 감지를 지원하는 모든 [리소스 유형](#)에 대해 실제 상태, 이전 배포 상태 및 원하는 상태의 3방향 비교를 지원합니다.

- AWS::SageMaker::MlflowTrackingServer
- AWS::Route53Resolver::FirewallRuleGroup
- AWS::MediaLive::Multiplexprogram
- AWS::VpcLattice::ResourceConfiguration
- AWS::S3::StorageLensGroup
- AWS::Bedrock::AgentAlias
- AWS::MSK::Cluster
- AWS::RDS::DBProxy
- AWS::Redshift::ClusterParameterGroup
- AWS::QBusiness::Index
- AWS::NetworkManager::CoreNetwork
- AWS::IAM::OIDCProvider
- AWS::Organizations::ResourcePolicy
- AWS::SNS::TopicInlinePolicy
- AWS::Route53::KeySigningKey
- AWS::DataZone::PolicyGrant
- AWS::Transfer::Certificate

- AWS::SageMaker::ImageVersion
- AWS::Neptune::DBParameterGroup
- AWS::ODBC::CloudVmCluster
- AWS::RolesAnywhere::TrustAnchor
- AWS::Detective::Graph
- AWS::Maester::DocumentType
- AWS::SageMaker::ModelPackageGroup
- AWS::S3Express::BucketPolicy
- AWS::Panorama::PackageVersion
- AWS::S3Tables::TableBucketPolicy

드리프트 인식 변경 세트는 3방향 비교를 지원하지 않는 리소스에 대해 이전 배포 상태와 원하는 상태의 비교로 돌아갑니다.

CloudFormation 스택에 대한 변경 세트 실행

변경 세트에 설명된 변경 사항을 스택에 적용하려면 변경 세트를 실행합니다.

Important

변경 세트를 실행한 후에는 업데이트된 스택에 대해 해당 변경 세트가 더 이상 유효하지 않기 때문에 CloudFormation에서는 스택과 관련된 모든 추가 변경 세트를 삭제합니다. 업데이트에 실패하면 새 변경 세트를 생성해야 합니다.

스택 정책 및 변경 세트 실행

연결된 스택 정책이 있는 스택에 대해 변경 세트를 실행하면 CloudFormation에서는 스택 업데이트 시 정책을 적용합니다. 변경 세트를 실행할 때 기존 정책을 덮어쓰는 임시 스택 정책을 지정할 수 없습니다. 보호되는 리소스를 업데이트하려면 스택 정책을 업데이트하거나 직접 업데이트 방법을 사용해야 합니다. 자세한 내용은 [직접 스택 업데이트](#) 섹션을 참조하세요.

Execute a change set (console)

변경 세트를 실행하려면 다음을 수행하세요.

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.

2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 스택 페이지에서 업데이트하려는 스택의 이름을 선택합니다.
4. 탐색 창에서 변경 세트를 선택하여 스택의 변경 세트 목록을 봅니다.
5. 실행하고자 하는 변경 세트의 이름을 선택합니다.
6. 변경 세트의 세부 정보 페이지에서 변경 세트 실행을 선택합니다.

CloudFormation에서는 즉시 스택을 업데이트하기 시작합니다. CloudFormation 콘솔에서 [이벤트(Events)] 탭으로 연결되며, 여기에서 스택 업데이트 진행률을 모니터링할 수 있습니다. 자세한 내용은 [스택 진행 상황 모니터링](#) 섹션을 참조하세요.

Execute a change set for nested stacks (console)

중첩 스택에 대한 변경 세트를 실행하려면 다음을 수행하세요.

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 스택 페이지에서 업데이트하려는 스택의 이름을 선택합니다. 루트 변경 세트와 연결된 스택 이름을 선택해야 합니다.
4. 탐색 창에서 변경 세트를 선택하여 스택의 변경 세트 목록을 봅니다.
5. 실행하고자 하는 루트 변경 세트의 이름을 선택합니다.
6. 변경 세트의 세부 정보 페이지에서 변경 세트 실행을 선택합니다.

Note

[CloudFormation 스택에 대한 변경 세트 생성](#) 프로세스 중에 중첩 스택의 변경 세트에 대해 [활성화됨(Enabled)] 상태인 경우, CloudFormation이 루트 변경 세트와 중첩된 변경 세트에서 설명한 변경 사항을 실행합니다.

CloudFormation에서는 즉시 스택을 업데이트하기 시작합니다. CloudFormation 콘솔에서 [이벤트(Events)] 탭으로 연결되며, 여기에서 스택 업데이트 진행률을 모니터링할 수 있습니다. 자세한 내용은 [스택 진행 상황 모니터링](#) 섹션을 참조하세요.

변경 세트를 실행하려면(AWS CLI)

- [execute-change-set](#) 명령을 실행합니다.

다음 예에서처럼 실행하려는 변경 세트의 변경 세트 ID를 지정합니다.

```
aws cloudformation execute-change-set \  
  --change-set-name \  
    arn:aws:cloudformation:us-east-1:123456789012:changeSet/  
    SampleChangeSet/1a2345b6-0000-00a0-a123-00abc0abc000
```

위 예에서 명령은 다음 ID를 사용하여 변경 세트를 실행합니다.

```
arn:aws:cloudformation:us-east-1:123456789012:changeSet/  
SampleChangeSet/1a2345b6-0000-00a0-a123-00abc0abc000.
```

명령을 실행한 후 CloudFormation에서 스택 업데이트를 시작합니다. 스택의 진행 상황을 보려면 [describe-stacks](#) 명령을 사용합니다.

CloudFormation 스택에 대한 변경 세트 삭제

변경 세트를 삭제하면 스택의 변경 세트 목록에서 제거됩니다. 변경 세트를 삭제하면 자신 또는 다른 사용자가 적용하면 안 되는 변경 세트를 실수로 실행할 수 없게 됩니다. CloudFormation에서는 변경 세트를 삭제하지 않는 한 스택을 업데이트할 때까지 모든 변경 세트를 보존합니다.

Delete a change set

변경 세트를 삭제하려면(콘솔)

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 스택 페이지에서 삭제할 변경 세트가 포함된 스택의 이름을 선택합니다.
4. 탐색 창에서 변경 세트를 선택하여 스택의 변경 세트 목록을 봅니다.
5. 삭제하고자 하는 변경 세트의 이름을 선택합니다.
6. 변경 세트의 세부 정보 페이지에서 변경 세트 삭제를 선택합니다.

CloudFormation이 즉시 스택의 변경 세트 목록에서 변경 세트를 삭제하기 시작하고, [스택 (Stacks)] 페이지로 리디렉션됩니다.

Delete a change set for nested stacks (console)

중첩된 세트에 대한 변경 세트를 삭제하려면 다음을 수행하세요.

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 스택 페이지에서 루트 변경 세트와 연결된 스택 이름을 선택합니다.
4. 탐색 창에서 변경 세트를 선택하여 스택의 변경 세트 목록을 봅니다.
5. 삭제하고자 하는 변경 세트의 이름을 선택합니다.
6. 변경 세트의 세부 사항 페이지에서 삭제를 선택하십시오. 변경 세트 삭제를 선택하면 중첩된 변경 세트의 계층 전체가 삭제됩니다.

Note

중첩 스택에 대한 변경 세트를 대상으로 한 삭제 작업은 비동기식이고 DELETE_PENDING 상태를 표시한 다음, DELETE_IN_PROGRESS 상태를 표시합니다. 변경 세트 삭제 작업이 완료되면 변경 세트가 목록에서 제거됩니다. REVIEW_IN_PROGRESS 상태의 중첩 스택은 변경 세트를 생성할 때 생성되었다면 마찬가지로 삭제됩니다.

CloudFormation이 스택의 변경 세트 목록에서 변경 세트를 즉시 삭제하기 시작합니다.

Note

중첩 스택이 진행 중 작업에서 멈춘 경우 [중첩 스택 롤백 실패](#)의 오류 문제 해결을 참조하세요.

변경 세트를 삭제하려면(AWS CLI)

- [delete-change-set](#) 명령을 실행하여 다음 예에서처럼 삭제하려는 변경 세트의 변경 세트 ID를 지정합니다.

```
aws cloudformation delete-change-set \  
--change-set-name \  

```

```
arn:aws:cloudformation:us-east-1:123456789012:changeSet/
SampleChangeSet/1a2345b6-0000-00a0-a123-00abc0abc000
```

CloudFormation 스택에 대한 예제 변경 세트

이 단원에서는 CloudFormation에서 일반적인 스택 변경을 위해 생성하는 변경 세트의 예를 제공합니다. 이러한 예에서는 템플릿을 직접 편집하는 방법, 단일 입력 파라미터를 수정하는 방법, 백업되지 않은 데이터 손실을 방지하거나 스택에서 실행 중인 애플리케이션이 중단되지 않도록 리소스 재생성(대체)을 계획하는 방법, 그리고 리소스를 추가 및 제거하는 방법을 보여줍니다. 변경 세트의 작동 방식을 설명하기 위해 제출된 변경 사항을 살펴보고 그 결과 생성된 변경 세트에 대해 알아봅니다. 각 예는 여러분이 이전의 예를 이해하고 있다고 가정하고 작성되었으므로 해당하는 예를 순서대로 읽어보는 것이 좋습니다. 변경 세트 내 각 필드에 대한 설명은 AWS CloudFormation API 참조의 [Change](#) 데이터 유형을 참조하세요.

[콘솔](#), AWS CLI 또는 CloudFormation [DescribeChangeSet](#) API 작업을 사용하여 변경 세트 세부 정보를 확인할 수 있습니다.

다음 각 변경 세트는 [샘플 템플릿](#)을 사용하여 스택에서 생성했습니다.

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Description" : "A sample EC2 instance template for testing change sets.",
  "Parameters" : {
    "Purpose" : {
      "Type" : "String",
      "Default" : "testing",
      "AllowedValues" : ["testing", "production"],
      "Description" : "The purpose of this instance."
    },
    "KeyName" : {
      "Type": "AWS::EC2::KeyPair::KeyName",
      "Description" : "Name of an existing EC2 KeyPair to enable SSH access to the
instance"
    },
    "InstanceType" : {
      "Type" : "String",
      "Default" : "t2.micro",
      "AllowedValues" : ["t2.micro", "t2.small", "t2.medium"],
      "Description" : "The EC2 instance type."
    }
  },
}
```

```

"Resources" : {
  "MyEC2Instance" : {
    "Type" : "AWS::EC2::Instance",
    "Properties" : {
      "KeyName" : { "Ref" : "KeyPairName" },
      "InstanceType" : { "Ref" : "InstanceType" },
      "ImageId" : "ami-8fcee4e5",
      "Tags" : [
        {
          "Key" : "Purpose",
          "Value" : { "Ref" : "Purpose" }
        }
      ]
    }
  }
}
}
}

```

직접 템플릿 편집

스택의 템플릿에서 직접 리소스를 수정해 변경 세트를 생성하면 업데이트된 파라미터값으로 시작되는 변경 사항과 달리 CloudFormation은 이러한 변경 사항을 직접적인 수정 사항으로 분류합니다. i-1abc23d4 인스턴스에 새 태그를 추가한 다음 변경 세트는 직접 수정의 예입니다. 파라미터값 및 기능과 같은 기타 모든 입력이 변경되지 않으므로 Changes 구조에 중점을 둘 수 있습니다.

```

{
  "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/MyStack/1a2345b6-0000-00a0-a123-00abc0abc000",
  "Status": "CREATE_COMPLETE",
  "ChangeSetName": "SampleChangeSet-direct",
  "Parameters": [
    {
      "ParameterValue": "testing",
      "ParameterKey": "Purpose"
    },
    {
      "ParameterValue": "MyKeyName",
      "ParameterKey": "KeyPairName"
    },
    {
      "ParameterValue": "t2.micro",
      "ParameterKey": "InstanceType"
    }
  ]
}

```

```

    ],
    "Changes": [
      {
        "ResourceChange": {
          "ResourceType": "AWS::EC2::Instance",
          "PhysicalResourceId": "i-1abc23d4",
          "Details": [
            {
              "ChangeSource": "DirectModification",
              "Evaluation": "Static",
              "Target": {
                "Attribute": "Tags",
                "RequiresRecreation": "Never"
              }
            }
          ],
          "Action": "Modify",
          "Scope": [
            "Tags"
          ],
          "LogicalResourceId": "MyEC2Instance",
          "Replacement": "False"
        },
        "Type": "Resource"
      }
    ],
    "CreationTime": "2020-11-18T23:35:25.813Z",
    "Capabilities": [],
    "StackName": "MyStack",
    "NotificationARNs": [],
    "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/SampleChangeSet-direct/1a2345b6-0000-00a0-a123-00abc0abc000"
  }

```

Changes 구조에는 ResourceChange 구조가 하나뿐입니다. 이 구조는 CloudFormation에서 변경할 리소스 유형, CloudFormation에서 취할 조치, 리소스 ID, 변경 범위 및 변경 시 리소스를 바꿔야 하는지 여부(CloudFormation이 새 리소스를 만든 다음 이전 리소스 삭제) 등과 같은 정보를 설명합니다. 이 예에서 변경 세트는 CloudFormation에서 i-1abc23d4 EC2 인스턴스의 Tags 속성을 수정하고 인스턴스 대체가 필요하지 않음을 나타냅니다.

Details 구조에서 CloudFormation은 이러한 변경 사항에 직접 수정이라는 레이블을 붙입니다. 직접 수정 시에는 인스턴스를 다시 생성할(바꿀) 필요가 없습니다. CloudFormation에서 인스턴스를 바꾸지 않을 것이므로 사용자는 이 변경 사항을 자신 있게 실행할 수 있습니다.

CloudFormation에서는 이러한 변경을 Static 평가로 표시합니다. 정적 평가는 CloudFormation이 변경 세트를 실행하기 전에 태그의 값을 결정할 수 있음을 의미합니다. 변경 세트를 실행한 후에만 CloudFormation에서 값을 결정할 수 있는 경우도 있습니다. CloudFormation은 Dynamic 평가라는 레이블을 지정하여 이러한 변경 사항을 분류합니다. 예를 들어 조건부로 바뀌는 업데이트된 리소스를 참조하는 경우, CloudFormation은 업데이트된 리소스에 대한 참조가 변경될지 여부를 결정할 수 없습니다.

입력 파라미터값 수정

입력 파라미터값을 수정하면 CloudFormation에서는 업데이트된 파라미터값을 사용하는 각 리소스에 대해 변경 사항 2개를 생성 합니다. 이 예에서는 이러한 변경 사항이 어떻게 표시되는지 그리고 어떤 정보를 중점적으로 살펴봐야 하는지 알아보려고 합니다. 다음 예는 Purpose 입력 파라미터의 값만 변경하여 생성되었습니다.

Purpose 파라미터는 EC2 인스턴스에 대한 태그 키 값을 지정합니다. 이 예에서 파라미터값은 testing에서 production으로 변경되었습니다. 새 값은 Parameters 구조로 표시됩니다.

```
{
  "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/MyStack/1a2345b6-0000-00a0-a123-00abc0abc000",
  "Status": "CREATE_COMPLETE",
  "ChangeSetName": "SampleChangeSet",
  "Parameters": [
    {
      "ParameterValue": "production",
      "ParameterKey": "Purpose"
    },
    {
      "ParameterValue": "MyKeyName",
      "ParameterKey": "KeyPairName"
    },
    {
      "ParameterValue": "t2.micro",
      "ParameterKey": "InstanceType"
    }
  ],
  "Changes": [
    {
```

```

    "ResourceChange": {
      "ResourceType": "AWS::EC2::Instance",
      "PhysicalResourceId": "i-1abc23d4",
      "Details": [
        {
          "ChangeSource": "DirectModification",
          "Evaluation": "Dynamic",
          "Target": {
            "Attribute": "Tags",
            "RequiresRecreation": "Never"
          }
        },
        {
          "CausingEntity": "Purpose",
          "ChangeSource": "ParameterReference",
          "Evaluation": "Static",
          "Target": {
            "Attribute": "Tags",
            "RequiresRecreation": "Never"
          }
        }
      ],
      "Action": "Modify",
      "Scope": [
        "Tags"
      ],
      "LogicalResourceId": "MyEC2Instance",
      "Replacement": "False"
    },
    "Type": "Resource"
  ],
  "CreationTime": "2020-11-18T23:59:18.447Z",
  "Capabilities": [],
  "StackName": "MyStack",
  "NotificationARNs": [],
  "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/SampleChangeSet/1a2345b6-0000-00a0-a123-00abc0abc000"
}

```

Changes 구조는 [직접 템플릿 편집](#) 예제에서 수행한 것과 유사한 방법으로 작동합니다.

ResourceChange 구조가 하나뿐이고, 이 구조는 Tags EC2 인스턴스의 i-1abc23d4 속성에 대한 변경 사항을 설명합니다.

그러나 Details 구조에서는 단일 파라미터값이 변경되었다더라도 변경 세트에는 Tags 속성에 대한 변경 사항이 2개 표시됩니다. (Ref 내장 함수를 사용하여) 변경된 파라미터값을 참조하는 리소스에는 변경 사항이 항상 2개 있습니다. 하나는 Dynamic 평가와 관련된 변경 사항이고 다른 하나는 Static 평가와 관련된 변경 사항입니다. 이러한 유형의 변경 사항은 다음 필드를 보고 확인할 수 있습니다.

- Static 평가 변경의 경우 ChangeSource 필드를 확인합니다. 이 예에서 ChangeSource 필드는 ParameterReference와 동일합니다. 즉, 이 변경은 업데이트된 파라미터 참조 값의 결과입니다. 변경 세트에는 유사한 Dynamic 평가 변경이 포함되어 있어야 합니다.
- 두 변경 사항에 대한 Dynamic 구조를 비교해 일치하는 Target 평가 변경을 찾을 수 있습니다. 이러한 구조에는 동일한 정보가 포함됩니다. 이 예에서 두 변경 사항에 대한 Target 구조에는 Attribute 및 RequireRecreation 필드에 대해 동일한 값이 포함되어 있습니다.

이러한 변경 유형의 경우, 변경 사항에 대해 가장 자세한 정보를 제공하는 정적 평가를 중점적으로 살펴보겠습니다. 이 예에서 정적 평가는 변경 사항이 파라미터 참조 값(ParameterReference)을 변경한 결과임을 보여줍니다. 변경된 정확한 파라미터는 CauseEntity 필드(Purpose 파라미터)로 표시됩니다.

대체 필드의 값 확인

ResourceChange 구조의 Replacement 필드는 CloudFormation에서 리소스를 다시 생성할지 여부를 나타냅니다. 리소스 재생성(대체)을 계획하면 백업되지 않은 데이터 손실 또는 스택에서 실행 중인 애플리케이션 중단이 방지됩니다.

Replacement 필드의 값은 변경 시 리소스를 바꿔야 하는지 여부에 따라 달라지며, 변경의 RequiresRecreation 구조 내 Target 필드로 표시됩니다. 예를 들어, RequiresRecreation 필드가 Never이면 Replacement 필드는 False입니다. 그러나 한 리소스에 대한 변경 사항이 여러 개인데 변경마다 RequiresRecreation 필드 값이 다른 경우, CloudFormation은 가장 적극적인 개입 동작을 사용하여 리소스를 업데이트합니다. 다시 말해, 여러 변경 사항 중 리소스를 바꿔야 하는 것이 하나뿐인 경우 CloudFormation은 해당 리소스를 바꾸기 위해 Replacement 필드를 True로 설정합니다.

다음 변경 세트는 전부 EC2 인스턴스에서 사용되는 모든 파라미터(Purpose, InstanceType 및 KeyPairName)의 값을 변경하여 생성된 것입니다. Replacement 필드가 True이기 때문에 이러한 변경이 있는 경우 CloudFormation은 인스턴스를 바꿔야 합니다.

```
{
  "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/MyStack/1a2345b6-0000-00a0-a123-00abc0abc000",
  "Status": "CREATE_COMPLETE",
```

```

"ChangeSetName": "SampleChangeSet-multiple",
"Parameters": [
  {
    "ParameterValue": "production",
    "ParameterKey": "Purpose"
  },
  {
    "ParameterValue": "MyNewKeyName",
    "ParameterKey": "KeyPairName"
  },
  {
    "ParameterValue": "t2.small",
    "ParameterKey": "InstanceType"
  }
],
"Changes": [
  {
    "ResourceChange": {
      "ResourceType": "AWS::EC2::Instance",
      "PhysicalResourceId": "i-7bef86f8",
      "Details": [
        {
          "ChangeSource": "DirectModification",
          "Evaluation": "Dynamic",
          "Target": {
            "Attribute": "Properties",
            "Name": "KeyName",
            "RequiresRecreation": "Always"
          }
        },
        {
          "ChangeSource": "DirectModification",
          "Evaluation": "Dynamic",
          "Target": {
            "Attribute": "Properties",
            "Name": "InstanceType",
            "RequiresRecreation": "Conditionally"
          }
        },
        {
          "ChangeSource": "DirectModification",
          "Evaluation": "Dynamic",
          "Target": {
            "Attribute": "Tags",

```

```

        "RequiresRecreation": "Never"
    },
    {
        "CausingEntity": "KeyPairName",
        "ChangeSource": "ParameterReference",
        "Evaluation": "Static",
        "Target": {
            "Attribute": "Properties",
            "Name": "KeyName",
            "RequiresRecreation": "Always"
        }
    },
    {
        "CausingEntity": "InstanceType",
        "ChangeSource": "ParameterReference",
        "Evaluation": "Static",
        "Target": {
            "Attribute": "Properties",
            "Name": "InstanceType",
            "RequiresRecreation": "Conditionally"
        }
    },
    {
        "CausingEntity": "Purpose",
        "ChangeSource": "ParameterReference",
        "Evaluation": "Static",
        "Target": {
            "Attribute": "Tags",
            "RequiresRecreation": "Never"
        }
    }
],
"Action": "Modify",
"Scope": [
    "Tags",
    "Properties"
],
"LogicalResourceId": "MyEC2Instance",
"Replacement": "True"
},
"Type": "Resource"
},
],

```

```

    "CreationTime": "2020-11-18T00:39:35.974Z",
    "Capabilities": [],
    "StackName": "MyStack",
    "NotificationARNs": [],
    "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/
SampleChangeSet-multiple/1a2345b6-0000-00a0-a123-00abc0abc000"
}

```

각 변경(Details 구조의 정적 평가)을 보고 리소스를 바꿔야 하는 변경을 찾습니다. 이 예제에서는 변경 사항마다 RequireRecreation 필드 값이 다르지만, KeyName 속성의 변경에는 가장 적극적인 업데이트 동작이 포함되어 있어 항상 재생성해야 합니다. 키 이름이 변경되었으므로 CloudFormation이 인스턴스를 대체합니다.

키 이름이 변경되지 않은 경우 InstanceType 속성의 변경에는 가장 적극적인 업데이트 동작(Conditionally)이 포함되어 있어 Replacement 필드가 Conditionally가 됩니다.

CloudFormation이 인스턴스를 교체해야 하는 조건을 찾아보려면 [AWS::EC2::Instance](#) 리소스 유형의 InstanceType 속성에 대한 업데이트 동작을 확인합니다.

리소스 추가 및 제거

다음은 EC2 인스턴스를 제거하고 Auto Scaling 그룹 및 시작 구성을 추가하는 수정된 템플릿을 제출하여 생성된 예제입니다.

```

{
  "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
MyStack/1a2345b6-0000-00a0-a123-00abc0abc000",
  "Status": "CREATE_COMPLETE",
  "ChangeSetName": "SampleChangeSet-addremove",
  "Parameters": [
    {
      "ParameterValue": "testing",
      "ParameterKey": "Purpose"
    },
    {
      "ParameterValue": "MyKeyName",
      "ParameterKey": "KeyPairName"
    },
    {
      "ParameterValue": "t2.micro",
      "ParameterKey": "InstanceType"
    }
  ],
}

```

```

    "Changes": [
      {
        "ResourceChange": {
          "Action": "Add",
          "ResourceType": "AWS::AutoScaling::AutoScalingGroup",
          "Scope": [],
          "Details": [],
          "LogicalResourceId": "AutoScalingGroup"
        },
        "Type": "Resource"
      },
      {
        "ResourceChange": {
          "Action": "Add",
          "ResourceType": "AWS::AutoScaling::LaunchConfiguration",
          "Scope": [],
          "Details": [],
          "LogicalResourceId": "LaunchConfig"
        },
        "Type": "Resource"
      },
      {
        "ResourceChange": {
          "ResourceType": "AWS::EC2::Instance",
          "PhysicalResourceId": "i-1abc23d4",
          "Details": [],
          "Action": "Remove",
          "Scope": [],
          "LogicalResourceId": "MyEC2Instance"
        },
        "Type": "Resource"
      }
    ],
    "CreationTime": "2020-11-18T01:44:08.444Z",
    "Capabilities": [],
    "StackName": "MyStack",
    "NotificationARNs": [],
    "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/SampleChangeSet-addremove/1a2345b6-0000-00a0-a123-00abc0abc000"
  }

```

Changes 구조에는 리소스마다 하나씩 세 가지 ResourceChange 구조가 있습니다. 각 리소스의 Action 필드는 CloudFormation에서 리소스를 추가하는지 아니면 제거하는지 여부를 나타냅니다. Scope 및 Details 필드는 수정된 리소스에만 적용되므로 비어 있습니다.

새 리소스의 경우, 변경 세트를 실행할 때까지 CloudFormation이 일부 필드의 값을 확인할 수 없습니다. 예를 들어 Auto Scaling의 물리적 ID 및 시작 구성이 아직 존재하지 않기 때문에 CloudFormation은 이러한 정보를 제공하지 않습니다. CloudFormation은 변경 세트를 실행할 때 새 리소스를 생성합니다.

속성 수준 변경 사항 보기

다음 예제에서는 Amazon EC2 인스턴스의 Tag 속성에 대한 속성 수준 변경 사항을 보여줍니다. Value 및 Key 태그가 Test로 변경됩니다.

```
"ChangeSetName": "SampleChangeSet",
  "ChangeSetId": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/
SampleChangeSet/38d91d27-798d-4736-9bf1-fb7c46207807",
  "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
SampleEc2Template/68edcdc0-f6b6-11ee-966c-126d572cdd11",
  "StackName": "SampleEc2Template",
  "Description": "A sample EC2 instance template for testing change sets.",
  "Parameters": [
    {
      "ParameterKey": "KeyPairName",
      "ParameterValue": "BatchTest"
    },
    {
      "ParameterKey": "Purpose",
      "ParameterValue": "testing"
    },
    {
      "ParameterKey": "InstanceType",
      "ParameterValue": "t2.micro"
    }
  ],
  "CreationTime": "2024-04-09T21:29:10.759000+00:00",
  "ExecutionStatus": "AVAILABLE",
  "Status": "CREATE_COMPLETE",
  "StatusReason": null,
  "NotificationARNs": [],
  "RollbackConfiguration": {
    "...skipping..."
  },
  "Changes": [
```

```

{
  "Type": "Resource",
  "ResourceChange": {
    "Action": "Modify",
    "LogicalResourceId": "MyEC2Instance",
    "PhysicalResourceId": "i-0cc7856a36315e62b",
    "ResourceType": "AWS::EC2::Instance",
    "Replacement": "False",
    "Scope": [
      "Tags"
    ],
    "Details": [
      {
        "Target": {
          "Attribute": "Tags",
          "RequiresRecreation": "Never",
          "Path": "/Properties/Tags/0/Value",
          "BeforeValue": "testing",
          "AfterValue": "Test",
          "AttributeChangeType": "Modify"
        },
        "Evaluation": "Static",
        "ChangeSource": "DirectModification"
      },
      {
        "Target": {
          "Attribute": "Tags",
          "RequiresRecreation": "Never",
          "Path": "/Properties/Tags/0/Key",
          "BeforeValue": "Purpose",
          "AfterValue": "Test",
          "AttributeChangeType": "Modify"
        },
        "Evaluation": "Static",
        "ChangeSource": "DirectModification"
      }
    ],
    "BeforeContext": "{\"Properties\":{\"KeyName\":\"BatchTest\",\"ImageId\":\"ami-8fcee4e5\",\"InstanceType\":\"t2.micro\",\"Tags\":[{\"Value\":\"testing\",\"Key\":\"Purpose\"]}}}",
    "AfterContext": "{\"Properties\":{\"KeyName\":\"BatchTest\",\"ImageId\":\"ami-8fcee4e5\",\"InstanceType\":\"t2.micro\",\"Tags\":[{\"Value\":\"Test\",\"Key\":\"Test\"]}}}"
  }
}

```

```
}
]
```

Details 구조에는 변경 세트가 실행되기 전의 Key 및 Value에 대한 값과 변경 세트가 실행된 후의 해당 값이 표시됩니다.

중첩 스택에 대한 변경 세트

중첩 스택에 대한 변경 세트를 사용하면 모든 중첩 스택 계층의 애플리케이션과 인프라 리소스 변경 사항을 미리 확인하고, 모든 변경 사항이 의도한 대로인지 확인한 후 업데이트를 진행할 수 있습니다.

중첩 스택에 대한 변경 세트를 설명하는 자세한 내용을 알아보려면 다음 섹션을 참조하세요.

주제

- [변경 세트 및 중첩 스택 개요](#)
- [중첩 스택에 대한 변경 세트 작업\(콘솔\)](#)
- [중첩 스택에 대한 변경 세트 작업\(AWS CLI\)](#)

변경 세트 및 중첩 스택 개요

중첩 스택에 대한 변경 세트는 다음의 기능을 함께 묶어서 전체 스택 계층의 변경 사항을 미리 확인하는 범위를 확장합니다.

- 변경 세트는 스택에 제안된 변경 사항이 기존 또는 새로 생성된 리소스에 미치는 영향을 미리 볼 수 있는 CloudFormation 기능입니다. 변경 세트를 생성하면 CloudFormation이 스택과 제출한 리소스 변경 사항을 비교하여 제안된 변경 사항 목록을 제공합니다. 변경 세트에 대한 자세한 내용은 [변경 세트를 사용하여 CloudFormation 스택 업데이트](#)의 내용을 참조하세요.
- 중첩 스택은 다른 스택의 일부로 생성된 스택입니다. 예를 들어 하나의 중첩 스택에 네트워킹 및 보안 관련 리소스가 있고 다른 중첩 스택에 애플리케이션 리소스가 있을 수 있습니다. 이렇게 애플리케이션 모델의 파티션을 나누면 코드 유지 관리와 재사용에 도움이 됩니다. 중첩 스택에 대한 자세한 내용은 [중첩 스택을 사용하여 템플릿을 재사용 가능한 조각으로 분할](#) 섹션을 참조하세요.

중첩 스택에 대한 변경 세트 작업(콘솔)

- 변경 세트 생성 – 스택 계층의 아무 수준에서 변경 사항을 제출하여 변경 세트를 생성합니다. 수정된 스택 템플릿 또는 수정된 입력 파라미터 값을 제출하면 CloudFormation이 중첩 스택과 제출된 변경 사항을 비교하여 변경 세트를 생성합니다. 중첩 스택에 대한 변경 세트는 기본적으로

CloudFormation 콘솔에서 활성화됩니다. 자세한 내용은 [CloudFormation 스택에 대한 변경 세트 생성](#) 섹션을 참조하세요.

▼ Change sets

Change sets for nested stacks

Creates change sets for all nested stacks specified in your template. [Learn more](#)

☒ Enabled

Creates a change set for all nested stacks specified in the template.

☐ Disabled

Create a change set only on the target stack.

Note

루트 변경 세트는 스택과 연결된 변경 세트이고, 여기에서 변경 세트의 전체 계층이 생성됩니다. 루트 변경 세트에서 중첩 스택에 대한 변경 세트를 실행하거나 삭제해야 합니다. 자세한 내용은 [중첩 스택에서 스택 작업 수행](#) 섹션을 참조하세요.

- 변경 세트 보기 – 중첩 스택 내부의 리소스 변경 사항을 실행하기 전에 시각화합니다. 변경 세트의 변경 사항 섹션에서 현재 스택과 중첩 변경 세트를 탐색하여 제안된 변경 사항을 확인할 수 있습니다. 자세한 내용은 [CloudFormation 스택에 대한 변경 세트 보기](#) 섹션을 참조하세요.
- 변경 세트 실행 – 현재 스택 및 하위 항목과 관련된 변경 세트에 나와 있는 변경 사항을 실행합니다. 실행 작업은 루트 변경 세트에서 시작해야 합니다. 자세한 내용은 [CloudFormation 스택에 대한 변경 세트 실행](#) 섹션을 참조하세요.
- 변경 세트 삭제 – 현재 스택에서 변경 세트를 제거합니다. 변경 세트를 삭제하면 자신 또는 다른 사용자가 적용하지 말아야 할 변경 세트를 시작하는 실수를 예방하는 데 도움이 됩니다. 삭제 작업은 루트 변경 세트에서 실행해야 합니다. 자세한 내용은 [CloudFormation 스택에 대한 변경 세트 삭제](#) 섹션을 참조하세요.

중첩 스택에 대한 변경 세트 작업(AWS CLI)

create-change-set

- [create-change-set](#) - 중첩 스택에 대한 변경 세트는 기본적으로 AWS CLI에서는 활성화되어 있지 않습니다. 전체 스택 계층에 대한 변경 세트를 생성하려면 `--include-nested-stacks` 옵션을 지정하세요. 자세한 내용은 [CloudFormation 스택에 대한 변경 세트 생성](#) 섹션을 참조하세요.

다음 AWS CLI 예제는 지정된 루트 스택에 대한 변경 세트를 생성합니다.

```
aws cloudformation create-change-set \
  --stack-name my-root-stack \
  --change-set-name my-root-stack-change-set \
  --template-body file://template.yaml \
  --capabilities CAPABILITY_IAM \
  --include-nested-stacks
```

다음은 예제 출력입니다.

```
{
  "Id": "arn:aws:cloudformation:us-west-2:123456789012:changeSet/my-root-stack-change-set/4eca1a01-e285-xmpl-8026-9a1967bfb4b0",
  "StackId": "arn:aws:cloudformation:us-west-2:123456789012:Stack/my-root-stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204"
}
```

describe-change-set

- [describe-change-set](#) – 변경 세트를 실행하면 CloudFormation이 생성할 변경 사항 목록을 반환합니다. 지정된 변경 세트에 중첩 스택에 속하는 하위 변경 세트가 포함되어 있으면 ChangeSetId가 해당 변경 세트에 대한 정보를 반환합니다. 자세한 내용은 [CloudFormation 스택에 대한 변경 세트 보기](#) 섹션을 참조하세요.

다음 AWS CLI 예제는 지정된 루트 스택에 대한 변경 세트를 설명합니다.

```
aws cloudformation describe-change-set \
  --change-set-name my-root-stack-change-set \
  --stack-name my-root-stack
```

다음은 예제 출력입니다.

```
{
  "Changes": [
    {
      "Type": "Resource",
      "ResourceChange": {
        "Action": "Modify",
        "LogicalResourceId": "ChildStack",
        "PhysicalResourceId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-nested-stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99205",

```

```

        "ResourceType": "AWS::CloudFormation::Stack",
        "Replacement": "False",
        "ChangeSetId": "arn:aws:cloudformation:us-
west-2:123456789012:changeSet/my-nested-stack-change-set/4eca1a01-e285-
xmpl-8026-9a1967bfb4b0",
        "Scope": [
            "Properties"
        ],
        "Details": [
            {
                "Target": {
                    "Attribute": "Properties",
                    "RequiresRecreation": "Never"
                },
                "Evaluation": "Dynamic",
                "ChangeSource": "Automatic"
            }
        ]
    }
},
    "ChangeSetName": "my-root-stack-change-set",
    "ChangeSetId": "arn:aws:cloudformation:us-west-2:123456789012:changeSet/my-root-
stack-change-set/4eca1a01-e285-xmpl-8026-9a1967bfb4b0",
    "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-root-stack/
d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
    "StackName": "my-root-stack",
    "IncludeNestedStacks": true,
    "ParentChangeSetId": null,
    "RootChangeSetId": null,
    "Description": null,
    "Parameters": null,
    "CreationTime": "2020-11-18T05:20:56.651Z",
    "ExecutionStatus": "AVAILABLE",
    "Status": "CREATE_COMPLETE",
    "StatusReason": null,
    "NotificationARNs": [

    ],
    "RollbackConfiguration": {

    },
    "Capabilities": [
        "CAPABILITY_IAM"
    ]
}

```

```
],  
  "Tags": null  
}
```

다음 AWS CLI 예제는 지정된 중첩 스택에 대한 변경 세트를 설명합니다.

```
aws cloudformation describe-change-set \  
  --change-set-name my-nested-stack-change-set \  
  --stack-name my-nested-stack
```

다음은 예제 출력입니다.

```
{  
  "Changes": [  
    {  
      "Type": "Resource",  
      "ResourceChange": {  
        "Action": "Modify",  
        "LogicalResourceId": "function",  
        "PhysicalResourceId": "my-function",  
        "ResourceType": "AWS::Lambda::Function",  
        "Replacement": "False",  
        "ChangeSetId": null,  
        "Scope": [  
          "Properties"  
        ],  
        "Details": [  
          {  
            "Target": {  
              "Attribute": "Properties",  
              "Name": "Timeout",  
              "RequiresRecreation": "Never"  
            },  
            "Evaluation": "Static",  
            "ChangeSource": "DirectModification"  
          },  
          {}  
        ],  
        "Properties": {}  
      },  
      "ChangeSetName": "my-nested-stack-change-set",  
      "ChangeSetId": "arn:aws:cloudformation:us-west-2:123456789012:changeSet/my-nested-stack-change-set/4eca1a01-e285-xmpl-8026-9a1967bfb4b0",  
    }  
  ],  
  "ChangeSetName": "my-nested-stack-change-set",  
  "ChangeSetId": "arn:aws:cloudformation:us-west-2:123456789012:changeSet/my-nested-stack-change-set/4eca1a01-e285-xmpl-8026-9a1967bfb4b0",  
}
```

```

    "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-nested-stack/
d0a825a0-e4cd-xmpl-b9fb-061c69e99205",
    "ParentChangeSetId": "arn:aws:cloudformation:us-west-2:123456789012:changeSet/my-
root-stack-change-set/4eca1a01-e285-xmpl-8026-9a1967bfb4b0",
    "RootChangeSetId": "arn:aws:cloudformation:us-west-2:123456789012:changeSet/my-
root-stack-change-set/4eca1a01-e285-xmpl-8026-9a1967bfb4b0",
    "IncludeNestedStacks": true,
    "StackName": "my-nested-stack",
    "Description": null,
    "Parameters": null,
    "CreationTime": "2020-11-18T05:20:56.651Z",
    "ExecutionStatus": "UNAVAILABLE",
    "Status": "CREATE_COMPLETE",
    "StatusReason": "Executable from root change set",
    "NotificationARNs": [

    ],
    "RollbackConfiguration": {

    },
    "Capabilities": [
        "CAPABILITY_IAM"
    ],
    "Tags": null
}

```

execute-change-set

- [execute-change-set](#) – 지정된 변경 세트가 생성되었을 때 제공한 입력 정보를 사용하여 스택을 생성하거나 업데이트합니다. 전체 스택 계층에 대한 변경 세트를 생성하려면 create-change-set 작업 중에 `--include-nested-stacks` 옵션을 지정해야 합니다. 자세한 내용은 [CloudFormation 스택에 대한 변경 세트 실행](#) 섹션을 참조하세요.

Note

execute-change-set는 루트 변경 세트에서 실행해야 하고, 전체 스택 계층에 변경 세트를 적용합니다.

다음 AWS CLI 예제는 지정된 루트 스택에 대한 변경 세트를 실행합니다.

```
aws cloudformation execute-change-set \
```

```
--stack-name my-root-stack \  
--change-set-name my-root-stack-change-set
```

delete-change-set

- [delete-change-set](#) – 지정된 변경 세트를 삭제합니다. 변경 세트를 삭제하면 아무도 잘못된 변경 세트를 사용할 수 없습니다. 변경 세트 삭제는 `--include-nested-stacks` 옵션으로 생성된 변경 세트에 대해서는 비동기 방식입니다. 자세한 내용은 [CloudFormation 스택에 대한 변경 세트 삭제](#) 섹션을 참조하세요.

Note

delete-change-set는 루트 변경 세트에서 실행해야 하고, 변경 세트의 전체 계층을 삭제합니다. REVIEW_IN_PROGRESS 상태의 중첩 스택은 create-change-set 작업 중에 생성되었다면 역시 삭제됩니다.

다음 AWS CLI 예제는 지정된 루트 스택에 대한 변경 세트를 삭제합니다.

```
aws cloudformation delete-change-set \  
--stack-name my-root-stack \  
--change-set-name my-root-stack-change-set
```

스택 배포 검증

사전 배포 검증을 통해 CloudFormation 변경 세트를 실행하기 전에 잠재적 배포 문제를 식별하고 해결할 수 있습니다. 이 기능은 일반적인 장애 시나리오에 대해 템플릿을 검증하므로 개발 주기 초반에 문제를 감지하는 데 도움이 됩니다.

주제

- [사전 배포 검증 작동 방식](#)
- [고려 사항](#)
- [사전 조건](#)
- [스택 배포 검증\(콘솔\)](#)
- [스택 배포 검증\(AWS CLI\)](#)
- [검증 유형](#)

- [리소스 제한 사항](#)

사전 배포 검증 작동 방식

사전 배포 검증에는 다음 단계가 포함됩니다.

1. 변경 세트 생성 - 일반적으로 CloudFormation 스택 업데이트와 같이 변경 세트를 생성합니다. 변경 세트를 생성할 때 사전 배포 검증이 기본적으로 활성화됩니다.
2. 검증 실행 - CloudFormation은 템플릿 및 대상 환경에 대해 여러 검증 검사를 실행합니다. 현재 3 가지 유형의 검증이 지원됩니다. 리소스 스키마에 대한 속성 구문 검증, 기존 리소스와의 리소스 이름 충돌 감지, 삭제 작업에 대한 S3 버킷 비어 있음 검증이 이에 해당합니다.
3. 검증 결과 검토 - CloudFormation은 템플릿에서 문제 위치를 정확히 식별하는 정확한 경로를 포함하여 발견된 모든 문제에 대한 자세한 피드백을 제공함으로써 수동 템플릿 디버깅을 수행하지 않아도 됩니다.
4. 문제 해결 - 배포를 진행하기 전에 템플릿을 업데이트하거나 충돌을 해결하여 식별된 문제를 해결합니다.
5. 자신 있게 실행 - 일반적인 장애 시나리오가 미리 검증되었음을 알고 변경 세트를 배포합니다.

고려 사항

사전 배포 검증을 사용할 경우 다음 사항에 유의하세요.

- 사전 배포 검증은 세 가지 일반적인 배포 장애 시나리오에 중점을 둡니다. 배포의 성공을 보장하지는 않지만 일반적인 장애의 가능성을 줄입니다.
- 다음과 같이 검증 모드는 서로 다르게 작동합니다.
 - FAIL 모드에서는 검증에서 오류를 감지하면 변경 세트 실행을 방지하여 문제가 있는 템플릿을 배포하지 않도록 합니다. 이는 속성 구문 오류 및 리소스 이름 충돌에 적용됩니다.
 - WARN 모드에서는 검증 실패에도 불구하고 변경 세트를 생성하며 개발자가 실행 전에 검토하고 해결할 수 있는 경고를 제공합니다. 수동 개입을 통해 해결할 수 있는 S3 버킷 비어 있음과 같은 제약 조건 위반에 적용됩니다.
- 검증 결과는 특정 변경 세트와 연결됩니다. 템플릿을 수정하는 경우 업데이트된 검증 결과를 얻으려면 새 변경 세트를 생성해야 합니다.
- S3 버킷 검증은 객체 존재만 확인하며 삭제를 방지할 수 있는 버킷 정책 또는 기타 제약 조건은 확인하지 않습니다.

사전 조건

사전 배포 검증을 사용하려면 다음이 필요합니다.

- 계정에서 변경 세트를 생성하고 리소스를 읽는 데 필요한 IAM 권한. S3 버킷 비어 있음 검사の場合 s3:ListBucket 권한이 필요합니다.
- 스택이 배포된 AWS 리전에 대한 액세스 권한.
- 배포 전에 검증하려는 CloudFormation 템플릿.

스택 배포 검증(콘솔)

콘솔에서 다음 절차를 통해 스택 배포를 검증합니다.

배포 전에 템플릿을 검증하는 방법

- AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
- 화면 상단의 탐색 모음에서 스택이 위치한 AWS 리전을 선택하세요.
- 스택 페이지에서 변경 세트를 생성하려는 실행 중인 스택을 선택합니다.
- 스택 세부 정보 창에서 스택 업데이트를 선택하고 변경 세트 생성을 선택하세요.
- stack-name**에 대한 변경 세트 생성 페이지에서 업데이트된 템플릿을 업로드하거나 템플릿 소스를 지정하세요.
- 다음을 선택하여 나머지 변경 세트 구성 단계를 진행하세요.
- 템플릿에 IAM 리소스가 포함된 경우 기능에서 CloudFormation에서 IAM 리소스를 생성할 수 있음을 확인함을 선택합니다. IAM 리소스는 AWS 계정의 권한을 수정할 수 있습니다. 이러한 리소스를 검토하여 의도한 작업만 허용하는지 확인하세요. 자세한 내용은 [CloudFormation 템플릿에서 IAM 리소스 승인](#)을 참조하세요.
- 검토 및 생성 페이지에서 변경 세트 생성을 선택하세요.
- CloudFormation은 변경 세트를 생성하고 검증 검사를 실행합니다. 배포 검증 탭에서 검증 결과를 검토하세요.
- 검증에 통과하거나 경고에 만족하는 경우 변경 세트 실행을 선택하여 변경 내용을 배포하세요.
- 검증에 실패하면 문제를 해결하고 새 변경 세트를 생성하여 배포를 다시 검증하세요.

스택 배포 검증(AWS CLI)

사전 배포 검증을 위한 AWS CLI 명령은 다음을 포함합니다.

- [create-change-set](#): 변경 세트 생성 중 자동 검증.
- [describe-change-set](#): 변경 세트 상태 확인
- [describe-events](#): 검증 결과 검토.

AWS CLI를 사용하여 다음 절차를 통해 스택 배포를 검증합니다.

배포 전에 템플릿을 검증하는 방법

1. [create-stack-set](#) 명령을 사용하세요.

```
aws cloudformation create-change-set \  
  --stack-name MyStack \  
  --change-set-name MyChangeSet \  
  --change-set-type "CREATE" \  
  --template-body file://updated-template.yaml
```

명령은 변경 세트 ARN 및 스택 ARN을 모두 반환하세요.

2. [describe-events](#) 명령을 변경 세트 ARN 또는 변경 세트 이름과 함께 사용하여 검증 상태 및 결과를 검토하세요.

```
aws cloudformation describe-events \  
  --change-set-name "arn:aws:cloudformation:us-east-1:123456789012:changeSet/MyChangeSet/94498df5-1afb-43b1-9869-9f82b2d877ac"
```

검증 오류 출력 예제:

```
{  
  "OperationEvents":[  
    {  
      "EventId":"9b5c9a29-4704-4ad0-8082-afb49418d55b",  
      "StackId":"arn:aws:cloudformation:us-east-1:123456789012:stack/MyStack/c3908380-b357-11f0-a97f-0ad08f35df65",  
      "OperationId":"f558b823-e1e3-4de3-a222-e6b930ddcad4",  
      "OperationType":"CREATE_CHANGESET",  
      "OperationStatus":"FAILED",
```

```

    "EventType": "STACK_EVENT",
    "Timestamp": "2025-10-27T17:10:02.923Z",
    "StartTime": "2025-10-27T17:09:57.537Z",
    "EndTime": "2025-10-27T17:10:02.923Z"
  },
  {
    "EventId": "2d8c3262-3468-4283-82fb-6e780e9e4f1d",
    "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/MyStack/c3908380-b357-11f0-a97f-0ad08f35df65",
    "OperationId": "f558b823-e1e3-4de3-a222-e6b930ddcad4",
    "OperationType": "CREATE_CHANGESET",
    "EventType": "VALIDATION_ERROR",
    "LogicalResourceId": "NotificationBucket",
    "PhysicalResourceId": "",
    "ResourceType": "AWS::S3::Bucket",
    "Timestamp": "2025-10-27T17:10:02.461Z",
    "ValidationFailureMode": "FAIL",
    "ValidationName": "PROPERTY_VALIDATION",
    "ValidationStatus": "FAILED",
    "ValidationStatusReason": "#/NotificationConfiguration/QueueConfigurations/0: required key [Event] not found",
    "ValidationPath": "/Resources/NotificationBucket/Properties/NotificationConfiguration/QueueConfigurations/0"
  }
]
}

```

3. 템플릿을 업데이트하여 검증 오류를 해결한 다음 새 변경 세트를 생성하세요.
4. 검증에 통과되면 변경 세트를 실행하세요.

```

aws cloudformation execute-change-set \
  --change-set-name MyChangeSet \
  --stack-name MyStack

```

검증 유형

사전 배포 검증에는 다음과 같은 유형의 검사가 포함됩니다.

- 속성 구문 검증 - AWS 리소스 스키마를 기준으로 리소스 속성을 검증합니다. 필요한 속성과 유효한 속성 값을 확인하고 더 이상 사용되지 않거나 지원되지 않는 속성 조합을 식별합니다.

- 리소스 이름 충돌 감지 - 기존 AWS 리소스와의 이름 충돌이 있는지 확인합니다. 리소스 이름이 AWS 이름 지정 요구 사항을 충족하는지 검증하고 배포 시도 전에 잠재적 충돌을 식별합니다.
- S3 버킷 비어 있음 검증 - 객체가 포함된 S3 버킷을 삭제하려고 할 때 경고합니다. 삭제 영향을 평가하고 일반적인 S3 삭제 실패를 방지하는 데 도움이 되도록 객체 수를 제공합니다.

각 검증 유형은 문제를 신속하게 해결하는 데 도움이 되도록 템플릿에 특정 오류 메시지와 오류 위치를 함께 제공합니다.

리소스 제한 사항

사전 배포 검증에 대해서는 다음 리소스 유형이 지원되지 않습니다.

- `AWS::ApiGatewayV2::ApiGatewayManagedOverrides`
- `AWS::ApiGatewayV2::Stage`
- `AWS::AppMesh::GatewayRoute`
- `AWS::AppMesh::Mesh`
- `AWS::AppMesh::Route`
- `AWS::AppMesh::VirtualGateway`
- `AWS::AppMesh::VirtualNode`
- `AWS::AppMesh::VirtualRouter`
- `AWS::AppMesh::VirtualService`
- `AWS::AppStream::Fleet`
- `AWS::AppStream::Stack`
- `AWS::AppStream::StackFleetAssociation`
- `AWS::AppStream::StackUserAssociation`
- `AWS::AppStream::User`
- `AWS::AppSync::ApiCache`
- `AWS::AppSync::ApiKey`
- `AWS::AppSync::GraphQLSchema`
- `AWS::AutoScalingPlans::ScalingPlan`
- `AWS::Budgets::Budget`
- `AWS::CertificateManager::Certificate`

- `AWS::Cloud9::EnvironmentEC2`
- `AWS::CloudFormation::CustomResource`
- `AWS::CloudFormation::Macro`
- `AWS::CloudFormation::WaitCondition`
- `AWS::CloudFormation::WaitConditionHandle`
- `AWS::CloudFront::StreamingDistribution`
- `AWS::CloudWatch::AnomalyDetector`
- `AWS::CloudWatch::InsightRule`
- `AWS::CodeBuild::Project`
- `AWS::CodeBuild::ReportGroup`
- `AWS::CodeBuild::SourceCredential`
- `AWS::CodeCommit::Repository`
- `AWS::CodeDeploy::DeploymentGroup`
- `AWS::CodeStar::GitHubRepository`
- `AWS::Config::ConfigurationRecorder`
- `AWS::Config::DeliveryChannel`
- `AWS::Config::OrganizationConfigRule`
- `AWS::Config::RemediationConfiguration`
- `AWS::DAX::Cluster`
- `AWS::DAX::ParameterGroup`
- `AWS::DAX::SubnetGroup`
- `AWS::DirectoryService::MicrosoftAD`
- `AWS::DLM::LifecyclePolicy`
- `AWS::DMS::Certificate`
- `AWS::DMS::Endpoint`
- `AWS::DMS::EventSubscription`
- `AWS::DMS::ReplicationInstance`
- `AWS::DMS::ReplicationSubnetGroup`
- `AWS::DMS::ReplicationTask`

- AWS::DocDB::DBCluster
- AWS::DocDB::DBClusterParameterGroup
- AWS::DocDB::DBInstance
- AWS::DocDB::DBSubnetGroup
- AWS::DocDB::EventSubscription
- AWS::EC2::ClientVpnAuthorizationRule
- AWS::EC2::ClientVpnEndpoint
- AWS::EC2::ClientVpnRoute
- AWS::EC2::ClientVpnTargetNetworkAssociation
- AWS::EC2::NetworkInterfacePermission
- AWS::ElastiCache::CacheCluster
- AWS::ElastiCache::ReplicationGroup
- AWS::ElastiCache::SecurityGroup
- AWS::ElastiCache::SecurityGroupIngress
- AWS::ElasticLoadBalancing::LoadBalancer
- AWS::ElasticLoadBalancingV2::ListenerCertificate
- AWS::Elasticsearch::Domain
- AWS::EMR::Cluster
- AWS::EMR::InstanceFleetConfig
- AWS::EMR::InstanceGroupConfig
- AWS::FSx::FileSystem
- AWS::FSx::Snapshot
- AWS::FSx::StorageVirtualMachine
- AWS::FSx::Volume
- AWS::Glue::Classifier
- AWS::Glue::Connection
- AWS::Glue::CustomEntityType
- AWS::Glue::DataCatalogEncryptionSettings
- AWS::Glue::DataQualityRuleset

- `AWS::Glue::DevEndpoint`
- `AWS::Glue::MLTransform`
- `AWS::Glue::Partition`
- `AWS::Glue::SecurityConfiguration`
- `AWS::Glue::Table`
- `AWS::Glue::TableOptimizer`
- `AWS::Glue::Workflow`
- `AWS::Greengrass::ConnectorDefinition`
- `AWS::Greengrass::ConnectorDefinitionVersion`
- `AWS::Greengrass::CoreDefinition`
- `AWS::Greengrass::CoreDefinitionVersion`
- `AWS::Greengrass::DeviceDefinition`
- `AWS::Greengrass::DeviceDefinitionVersion`
- `AWS::Greengrass::FunctionDefinition`
- `AWS::Greengrass::FunctionDefinitionVersion`
- `AWS::Greengrass::Group`
- `AWS::Greengrass::GroupVersion`
- `AWS::Greengrass::LoggerDefinition`
- `AWS::Greengrass::LoggerDefinitionVersion`
- `AWS::Greengrass::ResourceDefinition`
- `AWS::Greengrass::ResourceDefinitionVersion`
- `AWS::Greengrass::SubscriptionDefinition`
- `AWS::Greengrass::SubscriptionDefinitionVersion`
- `AWS::IAM::AccessKey`
- `AWS::IAM::UserToGroupAddition`
- `AWS::IoT::PolicyPrincipalAttachment`
- `AWS::IoT::ThingPrincipalAttachment`
- `AWS::IoTThingsGraph::FlowTemplate`
- `AWS::KinesisAnalytics::Application`

- `AWS::KinesisAnalytics::ApplicationOutput`
- `AWS::KinesisAnalytics::ApplicationReferenceDataSource`
- `AWS::KinesisAnalyticsV2::ApplicationCloudWatchLoggingOption`
- `AWS::KinesisAnalyticsV2::ApplicationOutput`
- `AWS::KinesisAnalyticsV2::ApplicationReferenceDataSource`
- `AWS::LakeFormation::DataLakeSettings`
- `AWS::LakeFormation::Permissions`
- `AWS::LakeFormation::Resource`
- `AWS::ManagedBlockchain::Member`
- `AWS::ManagedBlockchain::Node`
- `AWS::MediaConvert::JobTemplate`
- `AWS::MediaConvert::Preset`
- `AWS::MediaConvert::Queue`
- `AWS::MediaLive::Channel`
- `AWS::MediaLive::Input`
- `AWS::MediaLive::InputSecurityGroup`
- `AWS::MediaStore::Container`
- `AWS::OpsWorks::App`
- `AWS::OpsWorks::ElasticLoadBalancerAttachment`
- `AWS::OpsWorks::Instance`
- `AWS::OpsWorks::Layer`
- `AWS::OpsWorks::Stack`
- `AWS::OpsWorks::UserProfile`
- `AWS::OpsWorks::Volume`
- `AWS::Pinpoint::ADMChannel`
- `AWS::Pinpoint::APNSChannel`
- `AWS::Pinpoint::APNSSandboxChannel`
- `AWS::Pinpoint::APNSVoipChannel`
- `AWS::Pinpoint::APNSVoipSandboxChannel`

- `AWS::Pinpoint::App`
- `AWS::Pinpoint::ApplicationSettings`
- `AWS::Pinpoint::BaiduChannel`
- `AWS::Pinpoint::Campaign`
- `AWS::Pinpoint::EmailChannel`
- `AWS::Pinpoint::EmailTemplate`
- `AWS::Pinpoint::EventStream`
- `AWS::Pinpoint::GCMChannel`
- `AWS::Pinpoint::PushTemplate`
- `AWS::Pinpoint::Segment`
- `AWS::Pinpoint::SMSChannel`
- `AWS::Pinpoint::SmsTemplate`
- `AWS::Pinpoint::VoiceChannel`
- `AWS::PinpointEmail::ConfigurationSet`
- `AWS::PinpointEmail::ConfigurationSetEventDestination`
- `AWS::PinpointEmail::DedicatedIpPool`
- `AWS::PinpointEmail::Identity`
- `AWS::QLDB::Ledger`
- `AWS::RDS::DBSecurityGroup`
- `AWS::RDS::DBSecurityGroupIngress`
- `AWS::Redshift::ClusterSecurityGroup`
- `AWS::Redshift::ClusterSecurityGroupIngress`
- `AWS::Route53::RecordSet`
- `AWS::Route53::RecordSetGroup`
- `AWS::SageMaker::CodeRepository`
- `AWS::SageMaker::EndpointConfig`
- `AWS::SageMaker::Model`
- `AWS::SageMaker::NotebookInstance`
- `AWS::SageMaker::NotebookInstanceLifecycleConfig`

- AWS::SageMaker::Workteam
- AWS::SDB::Domain
- AWS::ServiceCatalog::AcceptedPortfolioShare
- AWS::ServiceCatalog::LaunchRoleConstraint
- AWS::ServiceCatalog::Portfolio
- AWS::ServiceCatalog::StackSetConstraint
- AWS::ServiceDiscovery::HttpNamespace
- AWS::ServiceDiscovery::Instance
- AWS::ServiceDiscovery::PrivateDnsNamespace
- AWS::ServiceDiscovery::PublicDnsNamespace
- AWS::ServiceDiscovery::Service
- AWS::SES::ReceiptFilter
- AWS::SES::ReceiptRule
- AWS::SES::ReceiptRuleSet
- AWS::SSM::MaintenanceWindow
- AWS::SSM::MaintenanceWindowTarget
- AWS::SSM::MaintenanceWindowTask
- AWS::WAF::ByteMatchSet
- AWS::WAF::IPSet
- AWS::WAF::Rule
- AWS::WAF::SizeConstraintSet
- AWS::WAF::SqlInjectionMatchSet
- AWS::WAF::WebACL
- AWS::WAF::XssMatchSet
- AWS::WAFRegional::ByteMatchSet
- AWS::WAFRegional::GeoMatchSet
- AWS::WAFRegional::IPSet
- AWS::WAFRegional::RateBasedRule
- AWS::WAFRegional::RegexPatternSet

- `AWS::WAFRegional::Rule`
- `AWS::WAFRegional::SizeConstraintSet`
- `AWS::WAFRegional::SqlInjectionMatchSet`
- `AWS::WAFRegional::WebACL`
- `AWS::WAFRegional::WebACLAssociation`
- `AWS::WAFRegional::XssMatchSet`
- `AWS::WorkSpaces::Workspace`
- `AWS::AmazonMQ::ConfigurationAssociation`
- `AWS::ApiGateway::DomainNameAccessAssociation`
- `AWS::AppConfig::ExtensionAssociation`
- `AWS::AppStream::ApplicationEntitlementAssociation`
- `AWS::AppStream::ApplicationFleetAssociation`
- `AWS::AppSync::DomainNameApiAssociation`
- `AWS::AppSync::SourceApiAssociation`
- `AWS::CleanRooms::ConfiguredTableAssociation`
- `AWS::CleanRooms::IdNamespaceAssociation`
- `AWS::CodeGuruReviewer::RepositoryAssociation`
- `AWS::Cognito::IdentityPoolRoleAttachment`
- `AWS::Cognito::UserPoolRiskConfigurationAttachment`
- `AWS::Cognito::UserPoolUICustomizationAttachment`
- `AWS::Cognito::UserPoolUserToGroupAttachment`
- `AWS::Connect::IntegrationAssociation`
- `AWS::Deadline::QueueFleetAssociation`
- `AWS::Deadline::QueueLimitAssociation`
- `AWS::EC2::EIPAssociation`
- `AWS::EC2::EnclaveCertificateIamRoleAssociation`
- `AWS::EC2::GatewayRouteTableAssociation`
- `AWS::EC2::IPAMResourceDiscoveryAssociation`
- `AWS::EC2::IpPoolRouteTableAssociation`

- `AWS::EC2::LocalGatewayRouteTableVPCAssociation`
- `AWS::EC2::LocalGatewayRouteTableVirtualInterfaceGroupAssociation`
- `AWS::EC2::NetworkInterfaceAttachment`
- `AWS::EC2::RouteServerAssociation`
- `AWS::EC2::SecurityGroupVpcAssociation`
- `AWS::EC2::SubnetNetworkAclAssociation`
- `AWS::EC2::SubnetRouteTableAssociation`
- `AWS::EC2::TransitGatewayAttachment`
- `AWS::EC2::TransitGatewayMulticastDomainAssociation`
- `AWS::EC2::TransitGatewayPeeringAttachment`
- `AWS::EC2::TransitGatewayRouteTableAssociation`
- `AWS::EC2::TransitGatewayVpcAttachment`
- `AWS::EC2::VPCDHCPOptionsAssociation`
- `AWS::EC2::VPCGatewayAttachment`
- `AWS::EC2::VolumeAttachment`
- `AWS::ECS::ClusterCapacityProviderAssociations`
- `AWS::EKS::PodIdentityAssociation`
- `AWS::FSx::DataRepositoryAssociation`
- `AWS::FSx::S3AccessPointAttachment`
- `AWS::GlobalAccelerator::CrossAccountAttachment`
- `AWS::LakeFormation::TagAssociation`
- `AWS::NetworkFirewall::VpcEndpointAssociation`
- `AWS::NetworkManager::ConnectAttachment`
- `AWS::NetworkManager::CustomerGatewayAssociation`
- `AWS::NetworkManager::DirectConnectGatewayAttachment`
- `AWS::NetworkManager::LinkAssociation`
- `AWS::NetworkManager::SiteToSiteVpnAttachment`
- `AWS::NetworkManager::TransitGatewayRouteTableAttachment`
- `AWS::NetworkManager::VpcAttachment`

- `AWS::Notifications::ChannelAssociation`
- `AWS::Notifications::ManagedNotificationAccountContactAssociation`
- `AWS::Notifications::ManagedNotificationAdditionalChannelAssociation`
- `AWS::Notifications::OrganizationalUnitAssociation`
- `AWS::ResourceExplorer2::DefaultViewAssociation`
- `AWS::Route53Profiles::ProfileAssociation`
- `AWS::Route53Profiles::ProfileResourceAssociation`
- `AWS::Route53Resolver::FirewallRuleGroupAssociation`
- `AWS::Route53Resolver::ResolverQueryLoggingConfigAssociation`
- `AWS::Route53Resolver::ResolverRuleAssociation`
- `AWS::SSM::Association`
- `AWS::SecretsManager::SecretTargetAttachment`
- `AWS::SecurityHub::PolicyAssociation`
- `AWS::ServiceCatalog::PortfolioPrincipalAssociation`
- `AWS::ServiceCatalog::PortfolioProductAssociation`
- `AWS::ServiceCatalog::ServiceActionAssociation`
- `AWS::ServiceCatalog::TagOptionAssociation`
- `AWS::ServiceCatalogAppRegistry::AttributeGroupAssociation`
- `AWS::ServiceCatalogAppRegistry::ResourceAssociation`
- `AWS::VpcLattice::ServiceNetworkResourceAssociation`
- `AWS::VpcLattice::ServiceNetworkServiceAssociation`
- `AWS::VpcLattice::ServiceNetworkVpcAssociation`
- `AWS::WAFv2::WebACLAssociation`
- `AWS::Wisdom::AssistantAssociation`
- `AWS::WorkspacesInstances::VolumeAssociation`
- `AWS::IAM::Policy`
- `AWS::SNS::TopicPolicy`
- `AWS::SQS::QueuePolicy`
- `AWS::EC2::NetworkAclEntry`

- `AWS::EC2::VPNGatewayRoutePropagation`
- `AWS::CloudFormation::Stack`
- `AWS::CloudWatch::MetricStream`
- `AWS::WorkSpaces::ConnectionAlias`
- `AWS::IoT::ProvisioningTemplate`
- `AWS::MediaPackage::Channel`
- `AWS::CloudFront::OriginRequestPolicy`
- `AWS::Route53Resolver::ResolverQueryLoggingConfig`
- `AWS::NetworkManager::TransitGatewayRegistration`
- `AWS::ImageBuilder::Image`
- `AWS::Config::ConformancePack`
- `AWS::S3::AccessPoint`
- `AWS::CodeStarConnections::Connection`
- `AWS::CloudFront::CachePolicy`
- `AWS::FMS::NotificationChannel`
- `AWS::ImageBuilder::InfrastructureConfiguration`
- `AWS::Detective::Graph`
- `AWS::EC2::CarrierGateway`
- `AWS::CloudWatch::CompositeAlarm`
- `AWS::CodeArtifact::Repository`
- `AWS::GroundStation::DataflowEndpointGroup`
- `AWS::ElasticLoadBalancingV2::Listener`
- `AWS::ImageBuilder::ImageRecipe`
- `AWS::NetworkManager::Device`
- `AWS::Kendra::DataSource`
- `AWS::Timestream::Database`
- `AWS::CodeGuruProfiler::ProfilingGroup`
- `AWS::Lambda::EventSourceMapping`
- `AWS::ECR::Repository`

- AWS::WAFv2::IPSet
- AWS::GameLift::Alias
- AWS::IoTSiteWise::Asset
- AWS::OpsWorksCM::Server
- AWS::IoT::Authorizer
- AWS::WAFv2::RuleGroup
- AWS::NetworkManager::Site
- AWS::ResourceGroups::Group
- AWS::MediaPackage::PackagingConfiguration
- AWS::ImageBuilder::ImagePipeline
- AWS::ECS::TaskDefinition
- AWS::Macie::CustomDataIdentifier
- AWS::MediaPackage::OriginEndpoint
- AWS::Logs::LogGroup
- AWS::CodeArtifact::Domain
- AWS::Kendra::Faq
- AWS::ECS::TaskSet
- AWS::WAFv2::RegexPatternSet
- AWS::ECS::Cluster
- AWS::SSO::Assignment
- AWS::GlobalAccelerator::Listener
- AWS::ServiceCatalog::CloudFormationProvisionedProduct
- AWS::RDS::DBProxy
- AWS::EC2::FlowLog
- AWS::ImageBuilder::Component
- AWS::CloudFront::RealtimeLogConfig
- AWS::NetworkManager::GlobalNetwork
- AWS::RDS::DBProxyTargetGroup
- AWS::WAFv2::WebACL

- `AWS::IVS::StreamKey`
- `AWS::IVS::PlaybackKeyPair`
- `AWS::Macie::Session`
- `AWS::Route53::HealthCheck`
- `AWS::Synthetics::Canary`
- `AWS::Lambda::CodeSigningConfig`
- `AWS::EFS::AccessPoint`
- `AWS::Timestream::Table`
- `AWS::MediaPackage::PackagingGroup`
- `AWS::ECS::PrimaryTaskSet`
- `AWS::Config::ConfigurationAggregator`
- `AWS::GroundStation::Config`
- `AWS::IoTSiteWise::AssetModel`
- `AWS::SES::ConfigurationSet`
- `AWS::ImageBuilder::DistributionConfiguration`
- `AWS::Config::OrganizationConformancePack`
- `AWS::EC2::LocalGatewayRoute`
- `AWS::KMS::Key`
- `AWS::Detective::MemberInvitation`
- `AWS::EKS::FargateProfile`
- `AWS::MediaPackage::Asset`
- `AWS::GlobalAccelerator::EndpointGroup`
- `AWS::Macie::FindingsFilter`
- `AWS::IoT::Certificate`
- `AWS::SageMaker::MonitoringSchedule`
- `AWS::IVS::Channel`
- `AWS::Kendra::Index`
- `AWS::EventSchemas::RegistryPolicy`
- `AWS::KinesisFirehose::DeliveryStream`

- AWS::GlobalAccelerator::Accelerator
- AWS::EC2::PrefixList
- AWS::GameLift::GameServerGroup
- AWS::NetworkManager::Link
- AWS::EFS::FileSystem
- AWS::Route53::HostedZone
- AWS::GroundStation::MissionProfile
- AWS::KMS::Alias
- AWS::FMS::Policy
- AWS::SSO::PermissionSet
- AWS::StepFunctions::StateMachine
- AWS::QLDB::Stream
- AWS::IoTSiteWise::Gateway
- AWS::ECS::Service
- AWS::ECS::CapacityProvider
- AWS::EC2::SecurityGroup
- AWS::EC2::SecurityGroupIngress
- AWS::EC2::SecurityGroupEgress
- AWS::EC2::EC2Fleet
- AWS::IAM::Group
- AWS::IAM::Role
- AWS::IAM::User
- AWS::ApiGateway::GatewayResponse
- AWS::S3::BucketPolicy
- AWS::SNS::Topic
- AWS::SNS::Subscription
- AWS::RDS::DBInstance
- AWS::RDS::DBParameterGroup
- AWS::RDS::DBCluster

- AWS::RDS::DBClusterParameterGroup
- AWS::RDS::DBSubnetGroup
- AWS::RDS::EventSubscription
- AWS::RDS::GlobalCluster
- AWS::RDS::OptionGroup
- AWS::Neptune::DBInstance
- AWS::Neptune::DBParameterGroup
- AWS::Neptune::DBCluster
- AWS::Neptune::DBClusterParameterGroup
- AWS::Neptune::DBSubnetGroup
- AWS::Redshift::Cluster
- AWS::Redshift::ClusterParameterGroup
- AWS::Redshift::ClusterSubnetGroup
- AWS::Redshift::EndpointAccess
- AWS::Redshift::EndpointAuthorization
- AWS::Redshift::EventSubscription
- AWS::Redshift::ScheduledAction
- AWS::ElastiCache::SubnetGroup

직접 스택 업데이트

스택에 대한 업데이트를 빠르게 배포하려면 직접 업데이트를 수행합니다. 직접 업데이트를 통해 스택의 리소스에 대한 업데이트를 지정하는 템플릿 또는 입력 파라미터를 제출하고 CloudFormation에서는 이러한 템플릿 및 파라미터를 즉시 배포합니다. 템플릿을 사용하여 업데이트하려는 경우 현재 템플릿을 수정하고 로컬 또는 Amazon S3 버킷에 로컬로 저장할 수 있습니다.

업데이트를 지원하지 않는 리소스 속성의 경우 현재 값을 유지해야 합니다. 스택을 업데이트하기 전에 CloudFormation이 스택에 대해 변경할 사항을 미리 보려면 변경 세트를 사용합니다. 자세한 내용은 [변경 세트를 사용하여 CloudFormation 스택 업데이트](#) 섹션을 참조하세요.

스택을 업데이트하는 경우 CloudFormation에서는 업데이트하는 속성에 따라 리소스를 중단하거나 업데이트된 리소스를 대체할 수 있습니다. 리소스 업데이트 동작에 대한 자세한 내용은 [스택 리소스의 업데이트 동작 이해](#) 단원을 참조하십시오.

스택 업데이트(콘솔)

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 스택 페이지에서 업데이트하려는 실행 중인 스택을 선택합니다.
4. 스택 세부 정보 창에서 업데이트를 선택합니다.
5. 스택 템플릿을 수정하지 않은 경우, 기존 템플릿 사용을 선택하고 다음을 선택합니다.

템플릿을 수정한 경우 기존 템플릿 교체를 선택하고 템플릿 지정 섹션에서 업데이트된 템플릿의 위치를 지정합니다.

- 컴퓨터에 로컬로 저장된 템플릿의 경우 템플릿 파일 업로드를 선택합니다. [파일 선택(Choose file)]을 선택하고 해당 파일을 찾아 선택한 후 [다음(Next)]을 선택합니다.

Note

로컬 템플릿 파일을 업로드하면 CloudFormation에서 해당 파일을 AWS 계정의 Amazon Simple Storage Service(Amazon S3) 버킷에 업로드합니다. CloudFormation에서 생성한 S3 버킷이 아직 없는 경우 템플릿 파일을 업로드하는 각 리전에 대해 고유한 버킷이 생성됩니다. CloudFormation에서 생성한 S3 버킷이 AWS 계정에 이미 있는 경우 CloudFormation이 템플릿을 해당 버킷에 추가합니다.

CloudFormation에서 생성한 S3 버킷 관련 고려 사항

- AWS 계정의 Amazon S3 권한이 있는 모든 사용자가 버킷에 액세스할 수 있습니다.
- CloudFormation은 기본값으로 서버 측 암호화가 활성화된 버킷을 생성하여 버킷에 저장된 모든 객체를 암호화합니다.

CloudFormation에서 생성한 버킷에 대한 암호화 옵션을 직접 관리할 수 있습니다(예: <https://console.aws.amazon.com/s3/>의 Amazon S3 콘솔 또는 AWS CLI 사용). 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 [Amazon S3 버킷에 대한 기본 서버 측 암호화 동작 설정](#)을 참조하세요.

- 템플릿을 Amazon S3에 수동으로 업로드하여 자체 버킷을 사용하고 버킷의 권한을 관리할 수 있습니다. 스택을 생성하거나 업데이트할 때마다 템플릿 파일의 Amazon S3 URL을 지정합니다.
- Amazon S3 버킷에 저장된 템플릿의 경우 Amazon S3 URL을 선택합니다. 템플릿에 대한 URL을 입력하거나 붙여 넣고 다음을 선택합니다.

템플릿이 버전 관리가 활성화된 버킷에 있는 경우 URL에 ?versionId=*version-id*를 추가하여 특정 템플릿 버전을 지정할 수 있습니다. 자세한 내용을 알아보려면 Amazon Simple Storage Service 사용 설명서의 [버전 관리를 사용하는 버킷의 객체 관리](#)를 참조하세요.

구문 문제가 감지되면 콘솔은 템플릿을 수정하는 데 도움이 되는 오류 메시지를 제공합니다.

6. 템플릿에 파라미터가 포함되어 있는 경우 [스택 세부 정보 지정(Specify stack details)] 페이지에서 파라미터값을 입력하거나 수정하고 [다음(Next)]을 선택하면 됩니다.

CloudFormation에서는 NoEcho 속성으로 파라미터가 선언된 경우를 제외하고 현재 스택에서 설정된 값으로 각 파라미터를 채웁니다. 하지만 기존 값 사용을 선택하여 현재 값을 계속해서 사용할 수 있습니다.

동적 파라미터를 사용하여 보안 암호를 관리하는 방법과 NoEcho를 사용하여 민감한 정보를 마스킹 처리하는 방법에 대한 자세한 내용은 [템플릿에 자격 증명을 포함하지 않음](#) 모범 사례를 참조하세요.

7. Configure stack options(스택 옵션 구성) 페이지에서 스택에 적용된 태그 및 권한을 업데이트하고 스택 정책, 롤백 구성과 같은 고급 옵션을 수정하거나 Amazon SNS 알림 주제를 업데이트할 수 있습니다. 이러한 옵션에 대한 자세한 내용은 [스택 옵션 구성](#) 섹션을 참조하세요.
8. 템플릿에 IAM 리소스가 포함되어 있는 경우 기능에서 이 템플릿이 IAM 리소스를 생성할 수 있음을 확인합니다를 선택하여 템플릿에서 IAM 리소스를 사용하도록 지정합니다. 자세한 내용은 [CloudFormation 템플릿에서 IAM 리소스 승인](#) 섹션을 참조하세요.
9. 다음을 선택하여 계속 진행합니다.
10. 스택 정보와 제출한 변경 사항을 검토합니다.

올바른 정보(예: 올바른 파라미터값 또는 템플릿 URL)를 제출했는지 확인합니다.

변경 세트 미리 보기 섹션에서 CloudFormation에서 필요한 모든 변경을 수행하는지 확인합니다. 예를 들어, 추가, 제거 또는 수정하려는 리소스를 CloudFormation에서 추가, 제거 또는 수정하는지 확인할 수 있습니다. CloudFormation에서는 스택에 대한 변경 세트를 만들어 이 미리 보기를 생성합니다. 자세한 내용은 [변경 세트를 사용하여 CloudFormation 스택 업데이트](#) 섹션을 참조하세요.

11. 변경 사항이 만족스러우면 Update stack(스택 업데이트)을 선택합니다.

Note

이때, 제안한 업데이트를 보다 철저하게 검토하기 위해 변경 세트를 볼 수도 있습니다. 이렇게 하려면 Update stack(스택 업데이트) 대신 View change set(변경 세트 보기)을 선택

합니다. CloudFormation은 업데이트를 기반으로 생성된 변경 세트를 표시합니다. 스택 업데이트를 수행할 준비가 되면 [실행(Execute)]을 선택합니다.

이벤트 창이 선택된 상태로 CloudFormation이 스택에 대한 스택 세부 정보 페이지를 표시합니다. 이제 스택의 상태는 UPDATE_IN_PROGRESS입니다. CloudFormation이 스택 업데이트를 성공적으로 완료하면 스택 상태가 UPDATE_COMPLETE로 설정됩니다.

스택 업데이트가 실패하면 CloudFormation에서 변경 사항을 자동으로 롤백하고 스택 상태를 UPDATE_ROLLBACK_COMPLETE로 설정합니다.

 Note

UPDATE_IN_PROGRESS 상태인 경우에는 업데이트를 취소할 수 있습니다. 자세한 내용은 [스택 업데이트 취소](#) 섹션을 참조하세요.

명령줄을 사용하여 스택을 업데이트하려면

다음 명령 중 하나를 사용할 수 있습니다:

- [update-stack](#) (AWS CLI)
- [Update-CFNStack](#) (AWS Tools for Windows PowerShell)

명령줄을 사용하여 스택을 업데이트하는 예제는 [AWS CLI 및 PowerShell에 대한 CloudFormation 스택 작업 명령의 예](#)의 내용을 참조하세요.

스택 업데이트 취소

스택 업데이트 시작 후 스택이 UPDATE_IN_PROGRESS 상태이면 스택 업데이트를 취소할 수 있습니다. 업데이트가 완료된 후에는 업데이트를 취소할 수 없습니다. 그러나 이전 설정을 사용하여 스택을 다시 업데이트할 수 있습니다.

스택 업데이트를 취소하면 스택 업데이트를 시작하기 전에 존재하던 스택 구성으로 스택이 롤백됩니다.

주제

- [스택 업데이트를 취소하려면 다음을 수행하세요\(콘솔\).](#)

- [스택 업데이트를 취소하려면 다음을 수행하세요\(AWS CLI\).](#)

스택 업데이트를 취소하려면 다음을 수행하세요(콘솔).

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 스택이 위치한 AWS 리전을 선택합니다.
3. 스택 페이지에서 현재 업데이트 중인 스택을 선택합니다. 상태는 UPDATE_IN_PROGRESS여야 합니다.
4. Stack actions(스택 작업)를 선택한 다음 스택 업데이트 취소를 선택하십시오.
5. 업데이트 취소를 계속 진행하려면 [업데이트 취소(Cancel update)]를 선택합니다. 그렇지 않으면 [취소(Cancel)]를 선택해 업데이트를 재개합니다.

스택이 UPDATE_ROLLBACK_IN_PROGRESS 상태로 바뀝니다. 업데이트 취소가 완료되면 스택이 UPDATE_ROLLBACK_COMPLETE로 설정됩니다.

스택 업데이트를 취소하려면 다음을 수행하세요(AWS CLI).

[cancel-update-stack](#) 명령을 사용하여 업데이트를 취소합니다. 자세한 내용은 [스택 업데이트 취소](#) 섹션을 참조하세요.

CloudFormation 콘솔에서 스택 삭제

스택의 리소스가 더는 필요하지 않으면 스택 전체를 삭제할 수 있습니다.

스택을 삭제할 때 CloudFormation은 삭제 정책을 사용하여 특정 리소스를 보관하지 않는 한 해당 스택의 모든 리소스를 삭제합니다. 자세한 내용은 [DeletionPolicy 속성](#)을 참조하세요.

스택을 삭제하려면(콘솔)

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 스택이 위치한 AWS 리전을 선택합니다.
3. 스택 페이지에서 삭제하려는 스택을 선택합니다. 스택이 현재 실행 중이어야 합니다.
4. 삭제>Delete)를 선택합니다.

5. 확인 메시지가 나타나면 삭제(Delete)를 선택합니다.

Note

스택 삭제가 시작된 후에는 스택 삭제 작업을 중지할 수 없습니다. 스택이 DELETE_IN_PROGRESS 상태로 바뀝니다.

스택 삭제가 완료되면 스택의 상태가 DELETE_COMPLETE로 바뀝니다. 상태가 DELETE_COMPLETE인 스택은 기본적으로 CloudFormation 콘솔에 표시되지 않습니다. 삭제된 스택을 표시하려면 [CloudFormation 콘솔에서 삭제된 스택 보기](#)에서 설명하는 것처럼 스택 보기 필터를 변경해야 합니다.

스택을 강제 삭제하려면(콘솔)

스택의 리소스를 삭제하지 못해 스택 삭제에 실패할 수 있습니다. 예를 들어, CloudFormation은 다른 스택이 종속되어 있는 리소스를 삭제하지 못합니다. 삭제되지 않은 모든 리소스는 스택을 성공적으로 삭제할 수 있을 때까지 보존됩니다. 삭제에 실패하고 DELETE_FAILED 상태가 반환되는 경우 다음 두 방법 중 하나를 사용하여 재시도할 수 있습니다.

1. CloudFormation 콘솔의 스택 페이지에서 강제 삭제할 스택을 선택합니다.
2. 스택 세부 정보 창에서 삭제 재시도를 선택합니다.
3. 다음 옵션 중에서 선택:
 - 이 스택은 삭제하고 리소스는 유지함: 이 옵션을 사용하면 원래 삭제에 실패했지만 강제 스택 삭제 중에 유지하려는 특정 리소스를 선택할 수 있습니다.
 - 전체 스택 강제 삭제: 이 옵션은 삭제에 실패한 모든 리소스를 유지하고 해당 리소스의 종속성도 유지합니다.
4. 삭제를 선택하여 선택한 스택의 강제 삭제 프로세스를 시작합니다.

유지된 리소스를 검토하려면(콘솔)

스택을 삭제한 후 콘솔에서 유지된 리소스를 볼 수 있습니다.

1. 스택 목록에서 상태 필터링을 선택하고 삭제됨을 선택합니다.
2. 삭제된 스택을 선택합니다.
3. 리소스 탭을 선택합니다.
4. 유지된 모든 리소스에는 DELETE_SKIPPED 상태가 표시됩니다.

5. 검토하려는 유지된 리소스를 선택합니다.

명령줄을 사용하여 스택을 삭제하려면

다음 명령 중 하나를 사용할 수 있습니다:

- [delete-stack](#) (AWS CLI)
- [Remove-CFNStack](#) (AWS Tools for Windows PowerShell)

명령줄을 사용하여 스택을 삭제하는 예제는 [AWS CLI 및 PowerShell에 대한 CloudFormation 스택 작업 명령의 예](#) 단원을 참조하세요.

관련 리소스

스택 삭제 오류 문제 해결에 대한 도움을 받으려면 [스택 삭제 실패](#) 문제 해결 주제를 참조하세요.

스택이 실수로 삭제되는 것을 방지하는 방법에 대한 자세한 내용은 [CloudFormation 스택 삭제 방지](#) 단원을 참조하세요.

CloudFormation 콘솔에서 삭제된 스택 보기

기본적으로 CloudFormation 콘솔은 DELETE_COMPLETE 상태의 스택을 표시하지 않습니다. 삭제된 스택에 대한 정보를 표시하려면 스택 보기를 변경해야 합니다.

삭제된 스택을 보려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 삭제된 스택이 위치한 AWS 리전을 선택합니다.
3. 스택 페이지의 필터 상태 드롭다운에서 삭제됨을 선택합니다.

CloudFormation은 DELETE_COMPLETE 상태의 모든 스택을 나열합니다.

다음 사항도 참조하세요.

- [CloudFormation 콘솔에서 스택 삭제](#)
- [CloudFormation 콘솔에서 스택 정보 보기](#)

스택 진행 상황 모니터링

이 섹션에서는 현재 진행 중인 스택 배포를 모니터링하는 방법을 설명합니다. CloudFormation에서는 진행 상황과 배포 중에 발생한 문제를 보여주는 배포 이벤트의 자세한 시간순 목록을 제공합니다.

주제

- [CloudFormation 스택 이벤트 보기](#)
- [작업별 스택 이벤트 보기](#)
- [CloudFormation 스택 배포의 타임라인 보기](#)
- [CloudFormation 스택 생성 이벤트 이해](#)
- [스택 업데이트 진행 상황 모니터링](#)
- [업데이트 롤백 계속](#)
- [스택 실패 원인 파악](#)
- [리소스 프로비저닝 시 실패 처리 방법 선택](#)

CloudFormation 스택 이벤트 보기

스택 이벤트를 확인하여 스택 및 스택 내 리소스의 진행 상황과 상태를 모니터링할 수 있습니다. 스택 이벤트는 리소스가 생성, 업데이트 또는 삭제되는 시기와 스택 배포가 예상대로 진행되고 있는지 파악하는 데 도움이 됩니다.

주제

- [스택 이벤트 보기\(콘솔\)](#)
- [스택 이벤트 보기\(AWS CLI\)](#)
- [스택 상태 코드](#)

스택 이벤트 보기(콘솔)

스택 이벤트를 보는 방법

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 화면 상단의 탐색 모음에서 스택을 생성한 AWS 리전을 선택합니다.
3. CloudFormation 콘솔의 Stacks(스택) 페이지에서 스택 이름을 선택합니다. CloudFormation에 선택한 스택에 대한 스택 세부 정보가 표시됩니다.

4. 이벤트 탭을 선택하여 CloudFormation에서 스택에 대해 생성한 스택 이벤트를 봅니다.

CloudFormation은 매분 스택 이벤트를 자동으로 새로 고칩니다. 또한 CloudFormation에서는 새 스택 이벤트가 발생할 때 사용 가능한 새 이벤트 배지를 표시합니다. 이러한 이벤트를 목록에 로드하려면 새로 고침 아이콘을 선택합니다. 스택 생성 이벤트를 보면 스택 생성의 원인이 된 이벤트의 시퀀스(또는 스택 디버깅 중인 경우 실패)를 파악할 수 있습니다.

스택이 생성되는 동안 스택 페이지에 CREATE_IN_PROGRESS 상태로 나열됩니다. 스택이 생성된 후에는 상태가 CREATE_COMPLETE로 변경됩니다.

자세한 내용은 [CloudFormation 스택 생성 이벤트 이해](#) 및 [스택 업데이트 진행 상황 모니터링](#)(을)를 참조하세요.

스택 이벤트 보기(AWS CLI)

또는 스택이 생성되는 동안 [describe-stack-events](#) 명령을 사용하여 보고되는 이벤트를 볼 수도 있습니다.

다음 describe-stack-events 명령에서는 *my-stack* 스택 이벤트를 설명합니다.

```
aws cloudformation describe-stack-events --stack-name my-stack
```

다음은 응답의 예입니다.

```
{
  "StackEvents": [
    {
      "StackId": "arn:aws:cloudformation:aws-region:123456789012:stack/my-stack/64726230-7edf-11f0-8a36-06453a64f325",
      "EventId": "7b755820-7edf-11f0-ab15-0673b09f3847",
      "StackName": "my-stack",
      "LogicalResourceId": "my-stack",
      "PhysicalResourceId": "arn:aws:cloudformation:aws-region:123456789012:stack/my-stack/64726230-7edf-11f0-8a36-06453a64f325",
      "ResourceType": "AWS::CloudFormation::Stack",
      "Timestamp": "2025-08-21T22:37:56.243000+00:00",
      "ResourceStatus": "CREATE_COMPLETE",
      "ClientRequestToken": "token"
    },
    {
      "StackId": "arn:aws:cloudformation:aws-region:123456789012:stack/my-stack/64726230-7edf-11f0-8a36-06453a64f325",
```

```

        "EventId": "WebServer-CREATE_COMPLETE-2025-08-21T22:37:54.356Z",
        "StackName": "my-stack",
        "LogicalResourceId": "WebServer",
        "PhysicalResourceId": "i-099df76cb31b866a9",
        "ResourceType": "AWS::EC2::Instance",
        "Timestamp": "2025-08-21T22:37:54.356000+00:00",
        "ResourceStatus": "CREATE_COMPLETE",
        "ResourceProperties": "{\"UserData\":
\\\"IyEvYm1uL2Jhc2gKeXVtIGluc3RhbGwgLXkgYXdzLWNmbi1ib290c3RyYXAKL29wdC9hd3MvYm1uL2Nmbi1pbml0IC12I
\\\",\\\"ImageId\\\":\\\"ami-0bbc328167dee8f3c\\\",\\\"InstanceType\\\":\\\"t2.micro\\\",
\\\"SecurityGroupIds\\\":[\\\"my-stack-WebServerSecurityGroup-n8A43bQT1ty2\\\"],\\\"Tags\\\":
[{\\\"Value\\\":\\\"Bootstrap Tutorial Web Server\\\",\\\"Key\\\":\\\"Name\\\"}]}\",
        "ClientRequestToken": "token"
    },
    {
        "StackId": "arn:aws:cloudformation:aws-region:123456789012:stack/my-
stack/64726230-7edf-11f0-8a36-06453a64f325",
        "EventId": "WebServer-CREATE_IN_PROGRESS-2025-08-21T22:37:31.226Z",
        "StackName": "my-stack",
        "LogicalResourceId": "WebServer",
        "PhysicalResourceId": "i-099df76cb31b866a9",
        "ResourceType": "AWS::EC2::Instance",
        "Timestamp": "2025-08-21T22:37:31.226000+00:00",
        "ResourceStatus": "CREATE_IN_PROGRESS",
        "ResourceStatusReason": "Resource creation Initiated",
        "ResourceProperties": "{\"UserData\":
\\\"IyEvYm1uL2Jhc2gKeXVtIGluc3RhbGwgLXkgYXdzLWNmbi1ib290c3RyYXAKL29wdC9hd3MvYm1uL2Nmbi1pbml0IC12I
\\\",\\\"ImageId\\\":\\\"ami-0bbc328167dee8f3c\\\",\\\"InstanceType\\\":\\\"t2.micro\\\",
\\\"SecurityGroupIds\\\":[\\\"my-stack-WebServerSecurityGroup-n8A43bQT1ty2\\\"],\\\"Tags\\\":
[{\\\"Value\\\":\\\"Bootstrap Tutorial Web Server\\\",\\\"Key\\\":\\\"Name\\\"}]}\",
        "ClientRequestToken": "token"
    },
    {
        "StackId": "arn:aws:cloudformation:aws-region:123456789012:stack/my-
stack/64726230-7edf-11f0-8a36-06453a64f325",
        "EventId": "WebServer-CREATE_IN_PROGRESS-2025-08-21T22:37:29.210Z",
        "StackName": "my-stack",
        "LogicalResourceId": "WebServer",
        "PhysicalResourceId": "",
        "ResourceType": "AWS::EC2::Instance",
        "Timestamp": "2025-08-21T22:37:29.210000+00:00",
        "ResourceStatus": "CREATE_IN_PROGRESS",
        "ResourceProperties": "{\"UserData\":
\\\"IyEvYm1uL2Jhc2gKeXVtIGluc3RhbGwgLXkgYXdzLWNmbi1ib290c3RyYXAKL29wdC9hd3MvYm1uL2Nmbi1pbml0IC12I

```

```

\","ImageId\":"ami-0bbc328167dee8f3c\","InstanceType\":"t2.micro\","
"SecurityGroupIds\":[\"my-stack-WebServerSecurityGroup-n8A43bQT1ty2\"],\"Tags\":"
[{\\"Value\":"Bootstrap Tutorial Web Server\\",\\"Key\":"Name\\"}]]",
  "ClientRequestToken": "token"
},
{
  "StackId": "arn:aws:cloudformation:aws-region:123456789012:stack/my-
stack/64726230-7edf-11f0-8a36-06453a64f325",
  "EventId": "WebServerSecurityGroup-
CREATE_COMPLETE-2025-08-21T22:37:28.803Z",
  "StackName": "my-stack",
  "LogicalResourceId": "WebServerSecurityGroup",
  "PhysicalResourceId": "my-stack-WebServerSecurityGroup-n8A43bQT1ty2",
  "ResourceType": "AWS::EC2::SecurityGroup",
  "Timestamp": "2025-08-21T22:37:28.803000+00:00",
  "ResourceStatus": "CREATE_COMPLETE",
  "ResourceProperties": "{\\"GroupDescription\":"Allow HTTP access from my IP
address\\",\\"SecurityGroupIngress\":[{\\"CidrIp\":"0.0.0.0/0\\",\\"Description\":"HTTP
\\",\\"FromPort\":"80\\",\\"ToPort\":"80\\",\\"IpProtocol\":"tcp\\"}]]",
  "ClientRequestToken": "token"
},
{
  "StackId": "arn:aws:cloudformation:aws-region:123456789012:stack/my-
stack/64726230-7edf-11f0-8a36-06453a64f325",
  "EventId": "WebServerSecurityGroup-
CREATE_IN_PROGRESS-2025-08-21T22:37:22.626Z",
  "StackName": "my-stack",
  "LogicalResourceId": "WebServerSecurityGroup",
  "PhysicalResourceId": "my-stack-WebServerSecurityGroup-n8A43bQT1ty2",
  "ResourceType": "AWS::EC2::SecurityGroup",
  "Timestamp": "2025-08-21T22:37:22.626000+00:00",
  "ResourceStatus": "CREATE_IN_PROGRESS",
  "ResourceStatusReason": "Resource creation Initiated",
  "ResourceProperties": "{\\"GroupDescription\":"Allow HTTP access from my IP
address\\",\\"SecurityGroupIngress\":[{\\"CidrIp\":"0.0.0.0/0\\",\\"Description\":"HTTP
\\",\\"FromPort\":"80\\",\\"ToPort\":"80\\",\\"IpProtocol\":"tcp\\"}]]",
  "ClientRequestToken": "token"
},
{
  "StackId": "arn:aws:cloudformation:aws-region:123456789012:stack/my-
stack/64726230-7edf-11f0-8a36-06453a64f325",
  "EventId": "WebServerSecurityGroup-
CREATE_IN_PROGRESS-2025-08-21T22:37:20.186Z",
  "StackName": "my-stack",

```

```

    "LogicalResourceId": "WebServerSecurityGroup",
    "PhysicalResourceId": "",
    "ResourceType": "AWS::EC2::SecurityGroup",
    "Timestamp": "2025-08-21T22:37:20.186000+00:00",
    "ResourceStatus": "CREATE_IN_PROGRESS",
    "ResourceProperties": "{\"GroupDescription\": \"Allow HTTP access from my IP address\", \"SecurityGroupIngress\": [{\"CidrIp\": \"0.0.0.0/0\", \"Description\": \"HTTP\", \"FromPort\": \"80\", \"ToPort\": \"80\", \"IpProtocol\": \"tcp\"}]}",
    "ClientRequestToken": "token"
  },
  {
    "StackId": "arn:aws:cloudformation:aws-region:123456789012:stack/my-stack/64726230-7edf-11f0-8a36-06453a64f325",
    "EventId": "64740fe0-7edf-11f0-8a36-06453a64f325",
    "StackName": "my-stack",
    "LogicalResourceId": "my-stack",
    "PhysicalResourceId": "arn:aws:cloudformation:aws-region:123456789012:stack/my-stack/64726230-7edf-11f0-8a36-06453a64f325",
    "ResourceType": "AWS::CloudFormation::Stack",
    "Timestamp": "2025-08-21T22:37:17.819000+00:00",
    "ResourceStatus": "CREATE_IN_PROGRESS",
    "ResourceStatusReason": "User Initiated",
    "ClientRequestToken": "token"
  }
]
}

```

가장 최신 이벤트가 먼저 보고됩니다. 다음 표에서는 describe-stack-events 명령에서 반환되는 필드를 설명합니다.

필드	설명
EventId	이벤트 식별자입니다.
StackName	이벤트에 해당하는 스택의 이름입니다.
StackId	이벤트에 해당하는 스택의 식별자입니다.
LogicalResourceId	리소스의 논리적 식별자입니다.
PhysicalResourceId	리소스의 물리적 식별자입니다.

필드	설명
ResourceProperties	리소스의 속성입니다.
ResourceType	리소스의 유형입니다.
Timestamp	이벤트가 발생한 시간입니다.
ResourceStatus	CREATE_COMPLETE CREATE_FAILED CREATE_IN_PROGRESS DELETE_COMPLETE DELETE_FAILED DELETE_IN_PROGRESS DELETE_SKIPPED IMPORT_COMPLETE IMPORT_IN_PROGRESS IMPORT_ROLLBACK_COMPLETE IMPORT_ROLLBACK_FAILED IMPORT_ROLLBACK_IN_PROGRESS REVIEW_IN_PROGRESS ROLLBACK_COMPLETE ROLLBACK_FAILED ROLLBACK_IN_PROGRESS UPDATE_COMPLETE UPDATE_COMPLETE_CLEANUP_IN_PROGRESS UPDATE_FAILED UPDATE_IN_PROGRESS UPDATE_ROLLBACK_COMPLETE UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS UPDATE_ROLLBACK_FAILED UPDATE_ROLLBACK_IN_PROGRESS 상태 코드 중 하나일 수 있는 리소스의 상태입니다. DELETE_SKIPPED 상태는 삭제 정책 속성이 보관인 리소스에 적용됩니다.
DetailedStatus	스택의 세부 상태입니다. CONFIGURATION_COMPLETE 가 있는 경우 스택 리소스 구성 단계가 완료되었으며 리소스 안정화가 진행 중입니다.
ResourceStatusReason	상태에 대한 추가 정보입니다.

스택 상태 코드

다음 표에서는 스택 상태 코드를 설명합니다.

스택 상태 및 선택적 세부 상태	설명
CREATE_COMPLETE	하나 이상의 스택을 성공적으로 생성했습니다.
CREATE_IN_PROGRESS	하나 이상의 스택을 생성하고 있습니다.
CREATE_FAILED	하나 이상의 스택을 생성하지 못했습니다. 스택 이벤트를 참조하여 관련 오류 메시지를 확인하세요. 가능한 생성 실패 원인에는 스택 내의 모든 리소스로 작업할 수 있는 권한 부족, AWS 서비스에서 파라미터값 거부, 또는 리소스 생성 중 제한 시간 초과가 포함될 수 있습니다.
DELETE_COMPLETE	하나 이상의 스택을 성공적으로 삭제했습니다. 삭제된 스택은 90일 동안 유지되며 볼 수 있습니다.
DELETE_FAILED	하나 이상의 스택을 삭제하지 못했습니다. 삭제에 실패했으므로 일부 리소스가 아직 실행 중일 수 있습니다. 하지만 스택으로 작업하거나 스택을 업데이트할 수 없습니다. 스택을 다시 삭제하거나 스택 이벤트를 참조하여 관련 오류 메시지를 확인하세요.
DELETE_IN_PROGRESS	하나 이상의 스택을 제거하고 있습니다.
REVIEW_IN_PROGRESS	템플릿 또는 리소스 없이 예상 StackId를 사용하여 하나 이상의 스택을 생성하고 있습니다.
	 Important 이 상태 코드를 가진 스택은 가능한 최대 스택 수에 포함됩니다.
ROLLBACK_COMPLETE	스택 생성이 실패하거나 스택 생성을 명시적으로 취소한 이후에 하나 이상의 스택을 성공적으로 제거했습니다. 스택은 이전 작업 상태로 돌아갑니다. 스택 생성 작업 중에 생성된 모든 리소스가 삭제됩니다.

스택 상태 및 선택적 세부 상태	설명
	스택 생성이 실패한 후에만 이 상태가 존재합니다. 이 상태는 부분적으로 생성된 스택의 모든 작업이 적절하게 정리되었음을 나타냅니다. 이 상태에서는 삭제 작업만 수행할 수 있습니다.
ROLLBACK_FAILED	스택 생성이 실패하거나 스택 생성을 명시적으로 취소한 이후에 하나 이상의 스택을 제거하지 못했습니다. 스택을 삭제하십시오. 연결된 오류 메시지는 스택 이벤트를 참조하십시오.
ROLLBACK_IN_PROGRESS	스택 생성이 실패하거나 스택 생성을 명시적으로 취소한 이후에 하나 이상의 스택을 제거하고 있습니다.
UPDATE_COMPLETE	하나 이상의 스택을 성공적으로 업데이트했습니다.
UPDATE_COMPLETE_CLEANUP_IN_PROGRESS	스택을 업데이트한 후 하나 이상의 스택에 대한 이전 리소스를 제거하고 있습니다. 스택 업데이트 중에 리소스를 대체해야 하는 경우 CloudFormation에서는 새 리소스를 생성한 다음 이전 리소스를 삭제하여 스택 종단을 줄여줍니다. 이 상태에서는 스택이 업데이트되고 사용 가능하지만 CloudFormation에서는 이전 리소스를 계속 삭제하고 있습니다.
UPDATE_FAILED	하나 이상의 스택을 업데이트하지 못했습니다. 스택 이벤트를 참조하여 관련 오류 메시지를 확인하세요.
UPDATE_IN_PROGRESS	하나 이상의 스택을 업데이트하고 있습니다.
UPDATE_ROLLBACK_COMPLETE	스택 업데이트를 실패한 이후에 하나 이상의 스택을 이전 작업 상태로 되돌렸습니다.
UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS	스택 업데이트를 실패한 후 하나 이상의 스택에 대한 새 리소스를 제거하고 있습니다. 이 상태에서는 스택이 이전 작업 상태를 백되고 사용 가능하지만 CloudFormation에서는 스택 업데이트 중에 생성된 새 리소스를 계속 삭제하고 있습니다.

스택 상태 및 선택적 세부 상태	설명
UPDATE_ROLLBACK_FAILED	스택 업데이트를 실패한 이후에 하나 이상의 스택을 이전 작업 상태로 되돌리지 못했습니다. 이 상태에서는 스택 또는 continue rollback 을 삭제할 수 있습니다. 스택이 작동 상태로 돌아가기 전에 오류를 수정해야 할 수도 있습니다. 또는 지원에 문의하여 스택을 사용 가능한 상태로 복원할 수 있습니다.
UPDATE_ROLLBACK_IN_PROGRESS	스택 업데이트를 실패한 이후에 하나 이상의 스택을 이전 작업 상태로 되돌리고 있습니다.
IMPORT_IN_PROGRESS	가져오기 작업이 현재 진행 중입니다.
IMPORT_COMPLETE	resource import를 지원하는 스택의 모든 리소스에 대해 가져오기 작업이 성공적으로 완료되었습니다.
IMPORT_ROLLBACK_IN_PROGRESS	가져오기는 이전 템플릿 구성으로 롤백됩니다.
IMPORT_ROLLBACK_FAILED	스택에 있는 하나 이상의 리소스에 대해 가져오기 롤백 작업이 실패했습니다. CloudFormation이 성공적으로 가져온 리소스에 대해 결과를 사용할 수 있습니다.
IMPORT_ROLLBACK_COMPLETE	가져오기가 이전 템플릿 구성으로 성공적으로 롤백되었습니다.

작업별 스택 이벤트 보기

스택에 대한 변경의 순서와 범위를 더 잘 이해하기 위해 작업별로 그룹화된 스택 이벤트를 볼 수 있습니다. 작업 기반 그룹화를 통해 관련 이벤트를 함께 추적하여 진행 상황을 모니터링하고 문제가 발생할 때 문제를 더 쉽게 해결할 수 있습니다.

각 스택 작업(생성, 업데이트, 삭제, 롤백)에는 모든 관련 이벤트를 그룹화하는 고유한 작업 ID가 할당됩니다. 이를 통해 특정 작업에 집중하고 장애의 근본 원인을 신속하게 식별할 수 있습니다.

주제

- [사전 조건](#)
- [작업별 스택 이벤트 보기\(콘솔\)](#)

- [작업별 스택 이벤트 보기\(AWS CLI\)](#)
- [스택 상태 코드](#)

사전 조건

DescribeEvents API를 사용하려면 DescribeEvents에 필요한 IAM 권한이 있어야 합니다.

작업별 스택 이벤트 보기(콘솔)

작업별로 그룹화된 스택 이벤트를 보는 방법

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 스택을 생성한 AWS 리전을 선택합니다.
3. 스택 페이지에서 스택 이름을 선택하세요. CloudFormation에 선택한 스택에 대한 스택 세부 정보가 표시됩니다.
4. 이벤트 탭을 선택하여 CloudFormation에서 스택에 대해 생성한 스택 이벤트를 봅니다.
5. 이벤트는 작업 ID별로 자동 그룹화됩니다. 각 작업은 작업 유형, 상태, 타임스탬프를 보여주는 확장 가능한 섹션으로 표시됩니다.
6. 작업 ID를 클릭하여 해당 특정 작업과 관련된 이벤트만 보여주는 세부 보기를 여세요.
7. 작업 세부 정보 보기에서 실패만 표시 확인란을 선택하여 근본 원인 분석을 위해 실패한 이벤트만 표시하세요.

CloudFormation은 매분 스택 이벤트를 자동으로 새로 고칩니다. 새 스택 이벤트가 발생하면 사용 가능한 새 이벤트 배지가 나타납니다. 이러한 이벤트를 목록에 로드하려면 새로 고침 아이콘을 선택합니다.

작업별로 그룹화된 스택 이벤트를 보면 각 작업의 이벤트 순서를 이해하고 문제의 원인이 되는 특정 작업을 빠르게 식별할 수 있습니다(스택을 디버깅하는 경우).

스택 작업이 실행 중인 동안 CREATE_IN_PROGRESS, UPDATE_IN_PROGRESS 또는 DELETE_IN_PROGRESS 상태로 나열됩니다. 작업이 성공적으로 완료되면 상태가 CREATE_COMPLETE, UPDATE_COMPLETE 또는 DELETE_COMPLETE로 변경됩니다.

자세한 내용은 [CloudFormation 스택 생성 이벤트 이해](#) 및 [스택 업데이트 진행 상황 모니터링\(을\)](#)을 참조하세요.

작업별 스택 이벤트 보기(AWS CLI)

작업 ID 필터링과 함께 `describe-events` 명령을 사용하여 특정 작업에 대한 이벤트를 볼 수 있습니다.

마지막 작업 ID 가져오기

이제 `describe-stacks` API를 통해 사용할 수 있는 스택 설명에 최근 작업 ID와 해당 유형을 보여주는 `LastOperations` 정보가 포함됩니다. 이를 통해 이벤트 로그를 구문 분석하지 않고도 수행된 작업과 현재 상태를 빠르게 식별할 수 있습니다.

```
aws cloudformation describe-stacks --stack-name MyStack
```

다음은 마지막 작업이 업데이트 작업에 실패한 후 롤백된 작업임을 보여주는 응답 예제입니다.

```
{
  "Stacks": [
    {
      "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/MyStack/07580010-bb79-11f0-8f6c-0289bb5c804f",
      "StackName": "MyStack",
      "Description": "A simple CloudFormation template to create an S3 bucket.",
      "CreationTime": "2025-11-07T01:28:13.778000+00:00",
      "LastUpdatedTime": "2025-11-07T01:43:39.838000+00:00",
      "RollbackConfiguration": {},
      "StackStatus": "UPDATE_ROLLBACK_COMPLETE",
      "DisableRollback": false,
      "NotificationARNs": [],
      "Tags": [],
      "EnableTerminationProtection": false,
      "DriftInformation": {
        "StackDriftStatus": "NOT_CHECKED"
      },
      "LastOperations": [
        {
          "OperationType": "ROLLBACK",
          "OperationId": "d0f12313-7bdb-414d-a879-828a99b36f29"
        },
        {
          "OperationType": "UPDATE_STACK",
          "OperationId": "1c211b5a-4538-4dc9-bfed-e07734371e57"
        }
      ]
    }
  ]
}
```

```

    }
  ]
}
]
}
}

```

작업 ID별 이벤트 필터링

다음 `describe-events` 명령은 특정 작업 ID에 대한 이벤트를 설명합니다.

```
aws cloudformation describe-events \
  --operation-id 1c211b5a-4538-4dc9-bfed-e07734371e57
```

문제 해결을 위해 실패한 이벤트만 보려면 `--filter FailedEvents=true` 파라미터를 사용합니다.

```
aws cloudformation describe-events \
  --operation-id 1c211b5a-4538-4dc9-bfed-e07734371e57 \
  --filter FailedEvents=true
```

새로운 작업 ID 필터링 기능을 사용하면 특정 작업 및 관련 이벤트에 집중할 수 있습니다. 특히 다음과 같은 경우에 유용합니다.

- 특정 장애 문제 해결: 실패한 작업에서 이벤트를 격리하여 무엇이 잘못되었는지 파악합니다.
- 장기 실행 작업 모니터링: 복잡한 업데이트 또는 대규모 스택 배포의 진행 상황을 추적합니다.
- 변경 내용 감사: 특정 업데이트 작업과 연결된 모든 이벤트를 검토합니다.
- 근본 원인 분석: 장애 필터를 사용하여 배포 문제의 원인을 빠르게 식별합니다.

스택 상태 코드

다음 표에서는 작업 ID 필터링을 사용할 때 `describe-events` 명령에서 반환되는 필드를 설명합니다.

필드	설명
EventId	이벤트 식별자입니다.
OperationId	이 이벤트를 생성한 작업의 고유 식별자.

필드	설명
StackName	이벤트에 해당하는 스택의 이름입니다.
StackId	이벤트에 해당하는 스택의 식별자입니다.
LogicalResourceId	리소스의 논리적 식별자입니다.
PhysicalResourceId	리소스의 물리적 식별자입니다.
ResourceProperties	리소스의 속성입니다.
ResourceType	리소스의 유형입니다.
Timestamp	이벤트가 발생한 시간입니다.
ResourceStatus	리소스 상태(CREATE_COMPLETE , UPDATE_FAILED 등).
DetailedStatus	스택의 세부 상태입니다. CONFIGURATION_COMPLETE 가 있는 경우 스택 리소스 구성 단계가 완료되었으며 리소스 안정화가 진행 중입니다.
ResourceStatusReason	상태에 대한 추가 정보입니다.

CloudFormation 스택 배포의 타임라인 보기

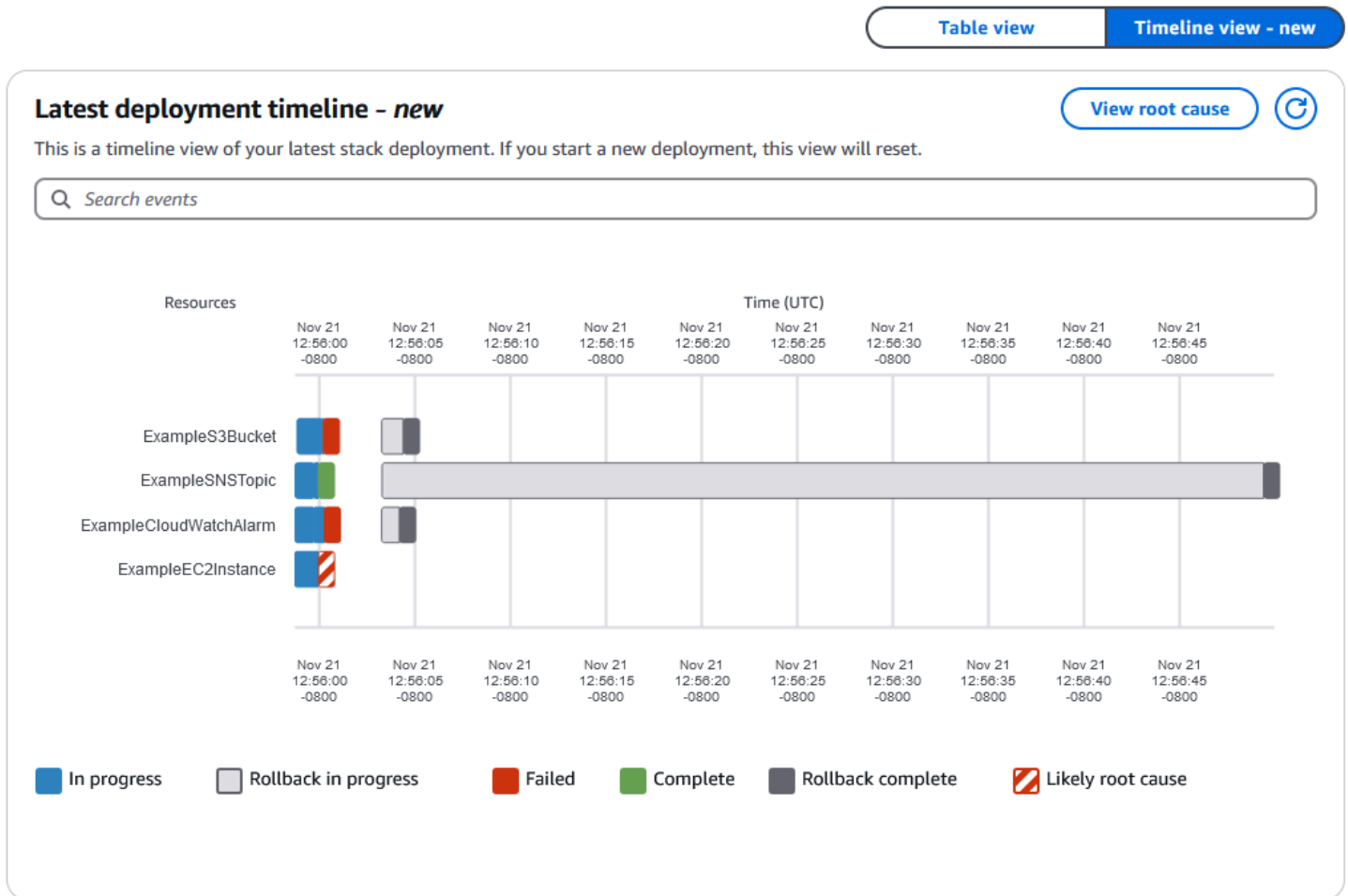
스택 배포 타임라인 그래프는 스택 배포 타임라인을 시각적으로 표현합니다. 이 뷰에는 스택 및 각 스택 리소스의 배포 상태와 각 상태가 변경된 시간이 표시됩니다. 스택 배포 상태는 해당 색상으로 표시됩니다.

주제

- [스택 배포 타임라인 그래프 이해](#)
- [스택 배포 타임라인 그래프 보기\(콘솔\)](#)

스택 배포 타임라인 그래프 이해

다음 이미지는 시작하지 못한 Amazon EC2 인스턴스 리소스로 인해 실패한 스택 배포의 타임라인 그래프를 보여줍니다.

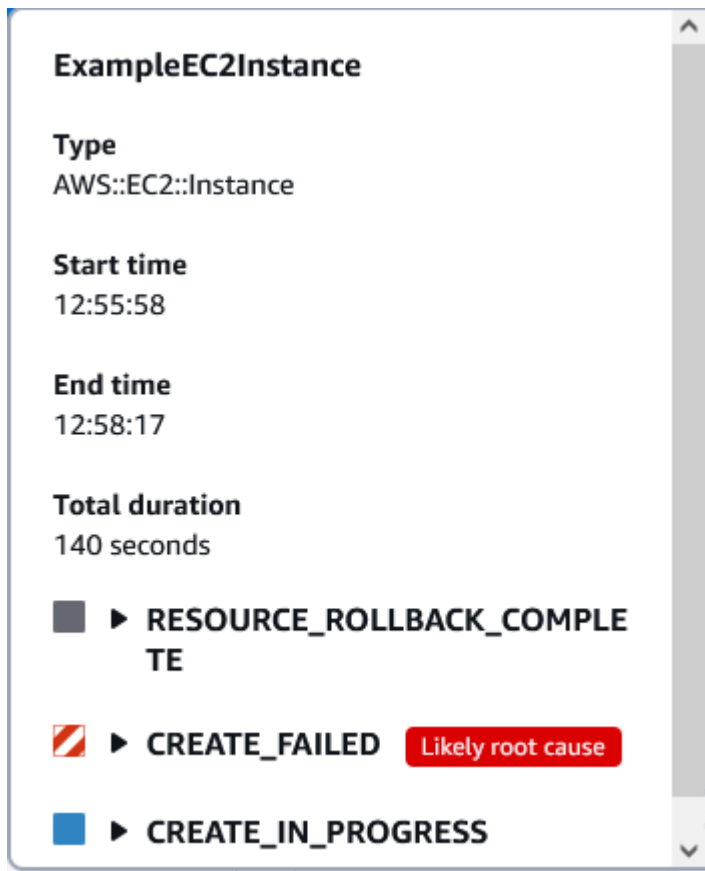


스택 리소스의 이름은 그래프의 왼쪽에 표시되고, 배포 시간을 기준으로 한 날짜와 시간은 그래프의 맨 위에 표시됩니다.

각 리소스는 진행 중 상태로 시작합니다. 배포가 성공할 때마다 상태 표시줄이 완료로 변경됩니다. 리소스가 배포되지 않으면 상태 표시줄이 실패로 변경됩니다. 리소스 배포에 실패하고 스택 배포도 실패하면 스택 배포 실패의 원인이 되는 리소스는 루트 장애 가능성 상태를 수신합니다.

스택 배포 작업이 실패하면 성공적으로 배포된 리소스가 롤백을 시작하고 롤백 진행 중 상태로 변경됩니다. 리소스 롤백이 완료되면 상태가 롤백 완료로 변경됩니다.

각 리소스를 선택하면 배포 타임라인에 대한 자세한 내용을 확인할 수 있습니다.



리소스를 선택하면 배포의 유형, 배포 시작 시간, 배포 종료 시간 및 배포의 총 기간이 표시됩니다. 아래 드롭다운 메뉴에서 각 배포 상태의 시작 시간, 종료 시간 및 기간도 확인할 수 있습니다. 리소스를 배포하지 못한 경우 실패 이유가 제공됩니다.

스택 상태에 대한 자세한 내용은 [스택 상태 코드](#) 섹션을 참조하세요.

스택 배포 타임라인 그래프 보기(콘솔)

스택 배포 타임라인 그래프를 보려면 다음을 수행하세요.

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 화면 상단의 탐색 모음에서 스택을 생성한 AWS 리전을 선택합니다.
3. CloudFormation 콘솔의 스택 페이지에서 스택 이름을 선택합니다. CloudFormation에 선택한 스택에 대한 스택 세부 정보가 표시됩니다.
4. 이벤트 탭을 선택하여 CloudFormation에서 스택에 대해 생성한 스택 이벤트를 봅니다.
5. 타임라인 그래프 버튼을 선택하여 스택에 대한 타임라인 그래프를 봅니다.

CloudFormation 스택 생성 이벤트 이해

스택 배포 중에 스택 템플릿에 정의된 리소스를 생성, 구성 및 검증하는 여러 이벤트가 발생합니다. 이러한 이벤트를 이해하면 스택 생성 프로세스를 최적화하고 배포를 간소화하는 데 도움이 될 수 있습니다.

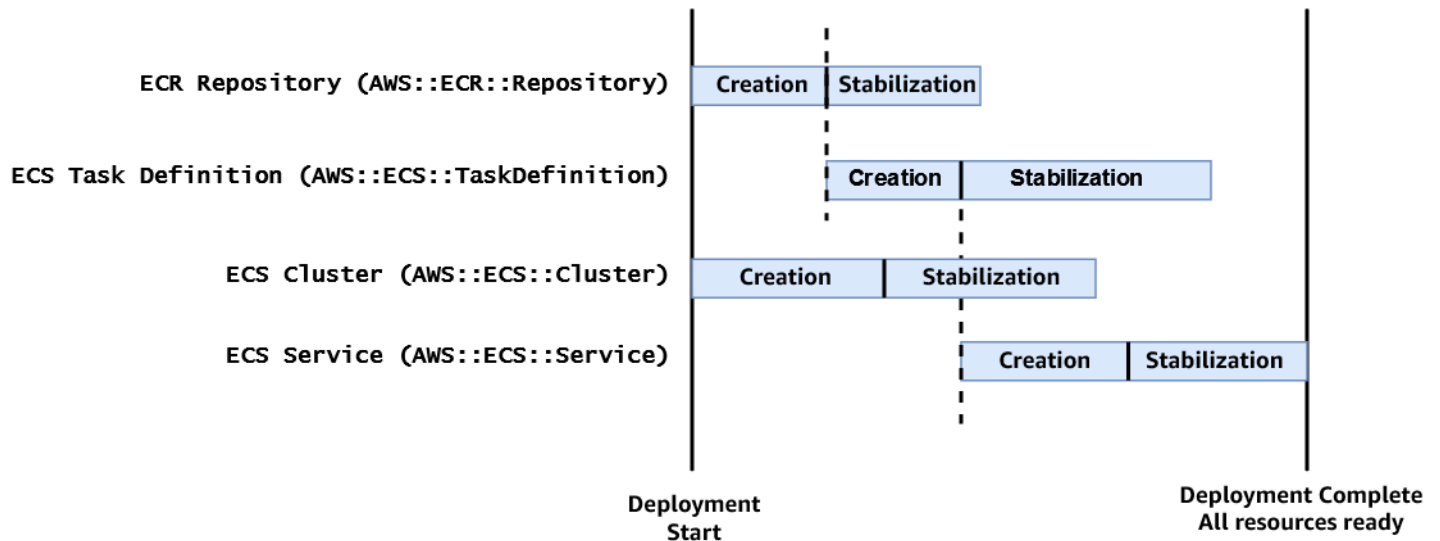
- 리소스 생성 프로세스 – 각 리소스에서 생성 프로세스가 시작되면 `CREATE_IN_PROGRESS` 이벤트의 상태가 설정됩니다. 이 이벤트는 리소스가 프로비저닝되고 있다는 것을 나타냅니다.
- 최종 일관성 검사 – 스택에서 생성된 리소스에 대한 최종 일관성 검사를 수행하는 데 스택 생성 시간의 상당 부분이 소요됩니다. 이 단계에서 서비스는 내부 일관성 검사를 수행하여 리소스가 완전히 작동하고 각 AWS 서비스에서 정의한 서비스 안정화 기준을 충족하는지 확인합니다.
- 구성 완료 이벤트 – 각 리소스가 프로비저닝의 최종 일관성 검사 단계를 마치면 `CONFIGURATION_COMPLETE` 이벤트의 세부 상태가 설정됩니다.
- 리소스 생성 완료 이벤트 – 지정된 대로 리소스가 생성 및 구성된 후 구성이 템플릿에 지정된 구성과 일치하면 `CREATE_COMPLETE` 이벤트의 상태가 설정됩니다.

사전 프로덕션 스택 구성 또는 스택 간 프로비저닝의 검증과 같이 리소스 최종 일관성 검사가 필요하지 않은 시나리오에서 `CONFIGURATION_COMPLETE` 이벤트를 활용하여 스택 생성 프로세스를 간소화할 수 있습니다. 이 이벤트는 여러 방법으로 사용할 수 있습니다. 예를 들어 리소스 또는 스택 일관성 검사가 완료될 때까지 기다리지 않아도 된다는 시각적 신호로 사용할 수 있습니다. 또는 지속적 통합 및 지속적 전달(CI/CD)을 사용하여 추가 작업을 트리거하는 자동 메커니즘을 생성할 수 있습니다.

Important

`CONFIGURATION_COMPLETE` 이벤트를 활용하면 스택 생성 시간이 가속화되지만 그에 따른 장단점을 염두에 두어야 합니다. 첫째, 드리프트 감지를 지원하는 리소스 유형의 하위 집합에 대해서만 지원됩니다. 드리프트 감지를 지원하는 리소스 유형 목록은 [리소스 유형 지원](#) 섹션을 참조하세요. 이 접근 방식은 일부 시나리오에 적합하지 않을 수 있습니다. 특히 클라우드 환경(예: 프로덕션 환경) 전반에서 완전한 운영 준비 상태를 보장하기 위해 리소스에 대한 철저한 최종 일관성 검사가 필요한 경우에 그렇습니다. 배포 요구 사항과 각 리소스에 대한 일관성 검사의 중요도를 신중하게 평가하는 것이 좋습니다. `CONFIGURATION_COMPLETE` 이벤트를 사용하여 인프라의 무결성과 신뢰성을 저해하지 않고 배포 속도를 최적화합니다.

`CONFIGURATION_COMPLETE` 이벤트 설정이 보장되지 않으므로 이벤트를 사용하는 모든 시나리오에서 `CONFIGURATION_COMPLETE` 이벤트가 설정되지 않았을 때 `CREATE_COMPLETE` 이벤트를 처리할 수 있도록 준비해야 합니다.



스택 배포가 시작되면 `AWS::ECR::Repository` 및 `AWS::ECS::Cluster` 리소스 모두에서 생성 프로세스가 시작됩니다(`ResourceStatus = CREATE_IN_PROGRESS`). `AWS::ECR::Repository` 리소스 유형이 최종 일관성 검사를 시작하면(`DetailedStatus = CONFIGURATION_COMPLETE`) `AWS::ECS::TaskDefinition` 리소스가 생성 프로세스를 시작할 수 있습니다. 마찬가지로 `AWS::ECS::TaskDefinition` 리소스가 최종 일관성 검사를 시작하면 `AWS::ECS::Service` 리소스에서 생성 프로세스가 시작됩니다.

CREATE_IN_PROGRESS 및 CREATE_COMPLETE 이벤트

- [스택]: `CREATE_IN_PROGRESS`
- [리소스]: ECR 리포지토리 `CREATE_IN_PROGRESS`
- [리소스]: ECS 클러스터 `CREATE_IN_PROGRESS`
- [리소스]: ECR 리포지토리 `CREATE_IN_PROGRESS`, `CONFIGURATION_COMPLETE`
- [리소스]: ECS 작업 정의 `CREATE_IN_PROGRESS`
- [리소스]: ECS 클러스터 `CREATE_IN_PROGRESS`, `CONFIGURATION_COMPLETE`
- [리소스]: ECS 작업 정의 `CREATE_IN_PROGRESS`, `CONFIGURATION_COMPLETE`
- [리소스]: ECS 서비스 `CREATE_IN_PROGRESS`
- [리소스]: ECR 리포지토리 `CREATE_COMPLETE`
- [리소스]: ECS 클러스터 `CREATE_COMPLETE`
- [리소스]: ECS 서비스 `CREATE_IN_PROGRESS`, `CONFIGURATION_COMPLETE`
- [스택]: `CREATE_IN_PROGRESS`, `CONFIGURATION_COMPLETE`
- [리소스]: ECS 작업 정의 `CREATE_COMPLETE`

- [리소스]: ECS 서비스 CREATE_COMPLETE
- [스택]: CREATE_COMPLETE

스택 업데이트 진행 상황 모니터링

스택의 이벤트를 보면서 스택 업데이트 진행률을 모니터링할 수 있습니다. 스택의 이벤트 탭에는 스택을 생성 및 업데이트하는 각 주요 단계가 각 이벤트의 시간순으로 정렬되어 표시됩니다. 즉, 최신 이벤트가 맨 위에 표시됩니다. 자세한 내용은 [스택 진행 상황 모니터링](#) 섹션을 참조하세요.

주제

- [성공적인 스택 업데이트 중 생성된 이벤트](#)
- [리소스 업데이트 실패 시 생성된 이벤트](#)

성공적인 스택 업데이트 중 생성된 이벤트

스택 업데이트 프로세스의 맨 처음에는 스택에 대해 UPDATE_IN_PROGRESS 이벤트가 표시됩니다.

```
2011-09-30 09:35 PDT AWS::CloudFormation::Stack MyStack UPDATE_IN_PROGRESS
```

그다음은 업데이트 템플릿에서 변경된 각 리소스의 업데이트 시작과 완료를 표시하는 이벤트가 표시됩니다. 예를 들어, MyDB라는 [AWS::RDS::DBInstance](#) 리소스를 업데이트하면 다음과 같은 항목이 표시됩니다.

```
2011-09-30 09:35 PDT AWS::RDS::DBInstance MyDB UPDATE_COMPLETE
2011-09-30 09:35 PDT AWS::RDS::DBInstance MyDB UPDATE_IN_PROGRESS
```

UPDATE_IN_PROGRESS 이벤트는 CloudFormation에서 리소스 업데이트가 시작되었다고 보고할 때 로깅됩니다. UPDATE_COMPLETE 이벤트는 리소스를 성공적으로 생성한 경우에 로깅됩니다.

CloudFormation에서 스택을 성공적으로 업데이트한 경우 다음과 같은 이벤트가 표시됩니다.

```
2011-09-30 09:35 PDT AWS::CloudFormation::Stack MyStack UPDATE_COMPLETE
```

⚠ Important

스택 업데이트 작업 중에 CloudFormation이 기존 리소스를 교체해야 할 경우, 먼저 새 리소스를 생성한 다음 이전 리소스를 삭제합니다. 그러나 CloudFormation이 이전 리소스를 삭제하지 못하는 경우가 있습니다(예: 사용자가 특정 유형의 리소스를 삭제할 권한이 없을 경우).

CloudFormation은 세 번에 걸쳐 이전 리소스를 삭제하려고 시도합니다. CloudFormation이 이전 리소스를 삭제하지 못하면 스택에서 이전 리소스를 제거하고 스택 업데이트를 진행합니다. 스택 업데이트가 완료되면 CloudFormation이 UPDATE_COMPLETE 스택 이벤트를 실행하지만 하나 이상의 리소스를 삭제할 수 없다고 되어 있는 StatusReason이(가) 포함됩니다. CloudFormation은 특정 리소스에 대한 DELETE_FAILED 이벤트를 실행하는데, CloudFormation이 리소스 삭제에 실패한 이유가 자세히 나와 있는 해당 StatusReason도 제공됩니다.

이전 리소스는 아직 남아 있고 계속 요금이 부과되지만, 더 이상 CloudFormation을 통해 액세스할 수 없습니다. 이전 리소스를 삭제하려면 콘솔 또는 API를 사용하여 기본 서비스에 대한 이전 리소스에 직접 액세스합니다.

이는 스택 템플릿에서 제거된 리소스에도 적용되며, 스택 업데이트 중에 스택에서 삭제됩니다.

리소스 업데이트 실패 시 생성된 이벤트

리소스 업데이트에 실패한 경우 CloudFormation에서는 실패 이유를 포함하는 UPDATE_FAILED 이벤트를 보고합니다. 예를 들어, 업데이트 템플릿에 리소스에서 지원되지 않는 속성 변경이 지정되어 있는 경우(예: [AWS::RDS::DBInstance](#) 리소스에 대한 AllocatedStorage의 크기 축소) 다음과 같은 이벤트가 표시됩니다.

```
2011-09-30 09:36 PDT AWS::RDS::DBInstance MyDB UPDATE_FAILED Size cannot be less than
current size; requested: 5; current: 10
2011-09-30 09:35 PDT AWS::RDS::DBInstance MyDB UPDATE_IN_PROGRESS
```

리소스 업데이트가 실패한 경우 CloudFormation에서는 업그레이드 중에 업데이트한 모든 리소스를 업데이트하기 이전의 구성으로 롤백합니다. 다음 예에서는 업데이트 롤백 중에 표시되는 이벤트를 보여줍니다.

```
2011-09-30 09:38 PDT AWS::CloudFormation::Stack MyStack UPDATE_ROLLBACK_COMPLETE
2011-09-30 09:38 PDT AWS::RDS::DBInstance MyDB UPDATE_COMPLETE
2011-09-30 09:37 PDT AWS::RDS::DBInstance MyDB UPDATE_IN_PROGRESS
```

2011-09-30 09:37 PDT AWS::CloudFormation::Stack *MyStack* UPDATE_ROLLBACK_IN_PROGRESS The following resource(s) failed to update: [MyDB]

업데이트 롤백 계속

CloudFormation에서 스택 업데이트를 롤백하려고 시도할 때 업데이트 프로세스 중에 변경한 내용을 모두 롤백하지 못하는 경우가 있습니다. 이를 UPDATE_ROLLBACK_FAILED 상태라고 합니다. 예를 들어, 스택이 CloudFormation 외부에서 삭제된 이전 데이터베이스 인스턴스로 롤백하기 시작합니다. CloudFormation에서는 데이터베이스가 삭제되었는지 알지 못하므로 해당 데이터베이스 인스턴스가 존재하는 것으로 간주하고 롤백을 시도합니다. 따라서 업데이트 롤백이 실패합니다.

UPDATE_ROLLBACK_FAILED 상태의 스택은 업데이트할 수 없지만 작업 상태 (UPDATE_ROLLBACK_COMPLETE)로 롤백할 수 있습니다. 스택을 원래 설정으로 되돌린 후 다시 업데이트를 시도할 수 있습니다.

대부분의 경우 스택을 계속해서 롤백하려면 업데이트 롤백 실패의 원인인 오류를 해결해야 합니다. 그 밖의 경우(예: 스택 작업이 시간 초과된 경우) 변경하지 않고 업데이트를 계속 롤백할 수 있습니다.

Note

중첩 스택을 사용하는 경우 상위 스택을 롤백하면 모든 하위 스택도 함께 롤백하려고 시도합니다.

업데이트 롤백을 계속하려면(콘솔)

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 스택이 위치한 AWS 리전을 선택합니다.
3. 스택 페이지에서 업데이트할 스택을 선택하고 스택 작업, 업데이트 롤백 계속을 차례로 선택합니다.

[오류 해결](#)의 해결 방법이 모두 효과가 없는 경우 고급 옵션을 사용하여 CloudFormation에서 롤백할 수 없는 리소스를 건너뛸 수 있습니다. 건너뛸 리소스의 논리적 ID를 [조회](#)하여 입력해야 합니다. 업데이트 전달 도중이 아니라 UpdateRollback 중에 UPDATE_FAILED 상태로 전환된 리소스만 지정합니다.

⚠ Warning

CloudFormation에서 지정된 리소스의 상태를 UPDATE_COMPLETE로 설정하고 스택을 계속 롤백합니다. 롤백이 완료된 이후에는 건너뛴 리소스의 상태가 스택 템플릿에 있는 리소스의 상태와 일치하지 않게 됩니다. 다른 스택을 업데이트하기 전에 스택 또는 리소스를 서로 일치하도록 업데이트해야 합니다. 그렇지 않으면 후속 스택 업데이트가 실패할 수 있습니다. 이 경우 스택을 복구할 수 없습니다.

스택을 롤백하는 데 필요한 최소 리소스 수를 지정합니다. 예를 들어, 리소스 업데이트가 실패하여 종속 리소스가 실패할 수 있습니다. 이 경우 종속 리소스를 건너뛴 필요가 없습니다.

중첩 스택에 포함된 리소스를 건너뛰려면 *NestedStackName.ResourceLogicalID* 형식을 사용합니다. Type: AWS::CloudFormation::Stack 목록에서 스택 리소스의 논리적 ID(ResourcesToSkip)를 지정하려면 포함된 해당 스택이 DELETE_IN_PROGRESS, DELETE_COMPLETE 또는 DELETE_FAILED 상태 중 하나여야 합니다.

업데이트 롤백을 계속하려면(AWS CLI)

- [continue-update-rollback](#) 명령을 --stack-name 옵션과 함께 사용하여 롤백을 계속할 스택의 ID를 지정합니다.

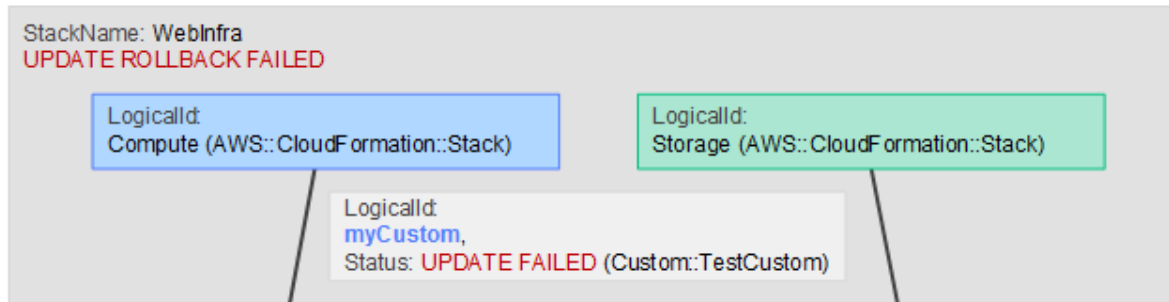
실패한 중첩 스택 업데이트에서 계속 롤백

여러 개의 스택이 서로 중첩된 경우 풀 스택 계층 구조를 작동 상태로 되돌리려면 여러 중첩 수준에서 리소스를 건너뛰어야 할 수도 있습니다.

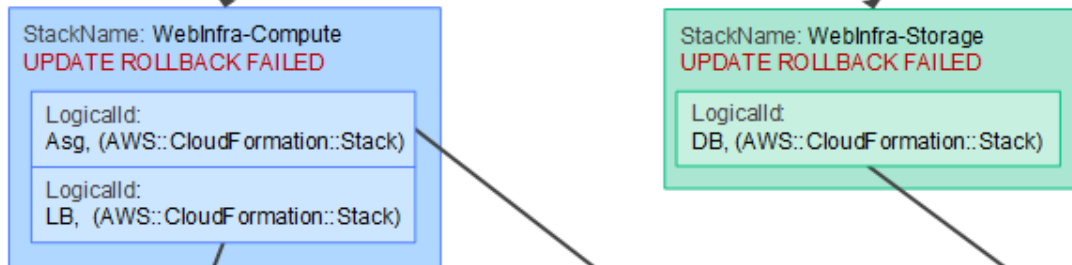
예를 들어, 내부에 WebInfra-Compute 및 WebInfra-Storage 스택이 포함된 WebInfra라는 루트 스택이 있습니다. 또한 이 두 스택 내부에는 자체 중첩 스택이 있습니다.

업데이트 중에 문제가 발생하여 업데이트 프로세스가 실패하면 전체 스택 계층 구조가 다음 다이어그램과 같이 UPDATE_ROLLBACK_FAILED 상태가 될 수 있습니다.

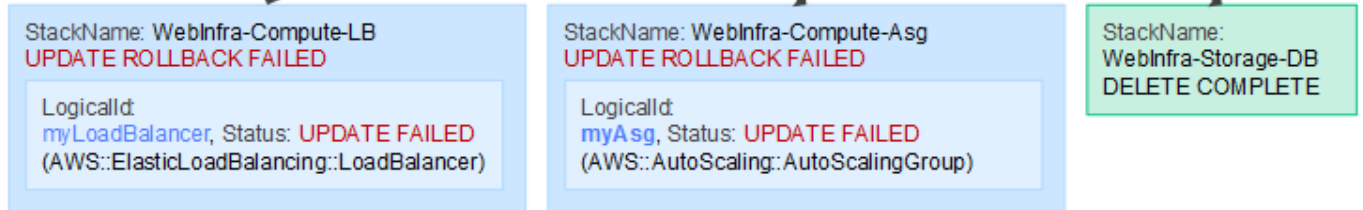
Root



Level 1



Level 2



Note

간소화하기 위해 이 예에서는 스택 이름이 잘려 있습니다. 하위 스택 이름은 일반적으로 CloudFormation에 의해 생성되고 고유한 임의의 문자열을 포함하므로 실제 이름은 사용자에게 친숙하지 않을 수 있습니다.

`continue-update-rollback` 명령을 사용하여 루트 스택을 실행 가능한 상태로 가져오려면 `--resources-to-skip` 옵션을 사용하여 롤백에 실패한 리소스를 건너뛰어야 합니다.

다음 `continue-update-rollback` 예제에서는 이전에 실패한 스택 업데이트에서 롤백 작업을 재개합니다. 이 예제에서 `--resources-to-skip` 옵션은 다음 항목을 포함합니다.

- *myCustom*

- *WebInfra-Compute-Asg.myAsg*
- *WebInfra-Compute-LB.myLoadBalancer*
- *WebInfra-Storage.DB*

루트 스택의 리소스의 경우 논리적 ID(예: *myCustom*)만 입력하면 됩니다. 하지만 중첩 스택에 포함된 리소스의 경우 중첩 스택 이름과 해당 논리적 ID를 마침표로 구분하여 입력해야 합니다. 예를 들어 *WebInfra-Compute-Asg.myAsg*입니다.

```
aws cloudformation continue-update-rollback --stack-name WebInfra \  
  --resources-to-skip myCustom WebInfra-Compute-Asg.myAsg WebInfra-Compute-  
  LB.myLoadBalancer WebInfra-Storage.DB
```

중첩 스택의 스택 이름 찾기

하위 스택의 스택 ID 또는 Amazon 리소스 이름(ARN)에서 찾을 수 있습니다.

다음 ARN 예제는 이름이 *WebInfra-Storage-Z2VKC706XKXT*인 스택을 참조합니다.

```
arn:aws:cloudformation:us-east-1:123456789012:stack/WebInfra-Storage-Z2VKC706XKXT/  
ea9e7f90-54f7-11e6-a032-028f3d2330bd
```

중첩 스택의 논리적 ID 찾기

상위 스택의 템플릿 정의에서 하위 스택의 논리적 ID를 찾을 수 있습니다. 다이어그램에서 *WebInfra-Storage-DB* 하위 스택의 *LogicalId*는 상위 *WebInfra-Storage*의 *DB*입니다.

CloudFormation 콘솔의 Resources(리소스) 또는 Events(이벤트) 탭에 있는 스택 리소스에 대한 Logical ID(논리적 ID) 열에서 논리적 ID를 찾을 수도 있습니다. 자세한 내용은 [CloudFormation 콘솔에서 스택 정보 보기](#) 섹션을 참조하세요.

스택 실패 원인 파악

스택 생성에 실패할 경우 CloudFormation은 스택 실패의 근본 원인일 가능성이 있는 이벤트를 파악하는 데 도움이 될 수 있습니다. 시나리오와 권한에 따라 이벤트에서 제공된 상태 이유가 명확하지 않은 경우 AWS CloudTrail 이벤트에서 근본 원인에 대한 추가 세부 정보를 제공할 수 있습니다.

스택 실패의 근본 원인 파악

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 스택 페이지에서 실패한 스택을 선택합니다.

3. 이벤트 탭을 선택합니다.
4. 근본 원인 감지를 선택합니다. CloudFormation은 실패를 분석하고 특정 이벤트 상태에 가능성 높은 근본 원인 라벨을 추가하여 실패의 원인일 가능성이 높은 이벤트를 보여줍니다. CloudFormation 콘솔의 상태에 대한 자세한 설명은 상태 이유를 참조하세요.
5. 실패 원인에 대해 자세히 알아보려면 가능성 높은 근본 원인 라벨이 있는 실패 상태를 선택합니다. 시나리오와 권한에 따라 구체적인 CloudTrail 이벤트를 검토할 수 있습니다. 상태를 선택할 때 발생할 수 있는 잠재적 결과는 다음과 같습니다.
 - 이 문제와 관련된 CloudTrail 이벤트가 제공되어 해결에 도움이 될 수 있습니다. CloudTrail 이벤트를 봅니다.
 - 해결에 도움이 될 수 있는 이 문제와 관련된 CloudTrail 이벤트를 찾을 수 없습니다.
 - 현재 권한으로는 CloudTrail 이벤트를 볼 수 없습니다. [자세히 알아보기](#)
 - 제공되는 CloudTrail 이벤트를 확인하는 중입니다. 몇 분 후에 다시 확인하세요.
 - CloudTrail 이벤트를 가져오는 도중 오류가 발생했습니다. 수동 검사는 CloudTrail 콘솔을 참조하세요.
6. 상태 이유에 제공된 이유가 명확하지 않고 근본 원인에 CloudTrail 콘솔로 연결되는 링크가 표시되는 경우 링크를 열어 이벤트를 보고 구체적인 근본 원인을 확인합니다.

CloudTrail 이벤트에 대한 자세한 내용은 [CloudTrail 이벤트 이해하기](#) 및 [CloudTrail 레코드 콘텐츠를 참조](#)하세요.

CloudTrail 이벤트 기록에 대한 자세한 내용은 [CloudTrail 이벤트 기록 작업](#)을 참조하세요.

Note

중첩 스택은 근본 원인 감지를 지원하지 않습니다.

리소스 프로비저닝 시 실패 처리 방법 선택

스택 작업에 실패하는 경우 이미 성공적으로 프로비저닝된 리소스를 롤백하고 매번 처음부터 다시 시작할 필요가 없습니다. 대신에 CREATE_FAILED 또는 UPDATE_FAILED 상태의 리소스 문제를 해결한 다음에 문제가 발생한 지점에서 프로비저닝을 재개할 수 있습니다.

이렇게 하려면 성공적으로 프로비저닝된 리소스 보존 옵션을 활성화해야 합니다. 이 옵션은 모든 스택 배포 및 변경 세트 작업에 사용할 수 있습니다.

- 스택 생성의 경우 성공적으로 프로비저닝된 리소스 보존 옵션을 선택하면 CloudFormation에서는 성공적으로 생성된 리소스의 상태를 보존하고 실패한 리소스는 다음 업데이트 작업이 수행될 때까지 실패 상태로 유지합니다.
- 업데이트 및 변경 세트 작업 중에 성공적으로 프로비저닝된 리소스 보존을 선택하면 성공한 리소스의 상태를 보존하면서 실패한 리소스를 마지막으로 알려진 안정 상태로 롤백합니다. 실패한 리소스는 UPDATE_FAILED 상태가 됩니다. 마지막으로 알려진 안정 상태가 없는 리소스는 다음 스택 작업 시 삭제됩니다.

주제

- [스택 실패 옵션 개요](#)
- [스택 롤백 일시 중지](#)에 필요한 조건
- [성공적으로 프로비저닝된 리소스 보존\(콘솔\)](#)
- [성공적으로 프로비저닝된 리소스 보존\(AWS CLI\)](#)

스택 실패 옵션 개요

CloudFormation 콘솔, API 또는 AWS CLI에서 작업을 실행하기 전에 프로비저닝된 리소스 실패에 대한 동작을 지정합니다. 그런 다음 다른 수정 없이 리소스 배포 프로세스를 진행합니다. 운영 실패가 발생하면 CloudFormation은 각 독립 프로비저닝 경로의 첫 번째 실패 시 중지합니다. CloudFormation은 리소스 간의 종속 항목을 식별하여 독립 프로비저닝 작업을 병렬화합니다. 그런 다음 실패가 발생할 때까지 각 독립 프로비저닝 경로에서 리소스 프로비저닝을 진행합니다. 한 경로의 실패는 다른 프로비저닝 경로에 영향을 주지 않습니다. CloudFormation은 완료될 때까지 리소스를 계속 프로비저닝하거나 다른 실패 시 중지됩니다.

배포 프로세스를 계속하려면 문제를 수정합니다. CloudFormation은 이전에 성공적으로 프로비저닝할 수 없었던 리소스에 대한 프로비저닝 작업을 재시도하기 전에 필요한 업데이트를 수행합니다. Retry(재시도), Update(업데이트) 또는 Roll back(롤백) 작업을 제출하여 문제를 해결합니다. 예를 들어 Amazon EC2 인스턴스를 프로비저닝하고 EC2 인스턴스가 생성 작업 중 실패하는 경우 실패한 리소스를 즉시 롤백하는 대신 오류를 조사할 수 있습니다. 시스템 상태 확인 및 인스턴스 상태 확인을 검토한 다음 문제가 해결되면 Retry(재시도) 작업을 선택할 수 있습니다.

스택 작업이 실패하고 Stack failure options(스택 실패 옵션) 메뉴에서 Preserve successfully provisioned resources(성공적으로 프로비저닝된 리소스 보존)를 지정한 경우 다음 옵션을 선택할 수 있습니다.

- **Retry(재시도)** – 실패한 리소스에 대한 프로비저닝 작업을 재시도하고 스택 작업이 성공적으로 완료되거나 다음 오류가 발생할 때까지 템플릿 프로비저닝을 계속합니다. AWS Identity and Access Management(IAM) 권한과 같이 템플릿 수정이 필요하지 않은 문제로 인해 리소스 프로비저닝에 실패한 경우 이 옵션을 선택합니다.
- **Update(업데이트)** – 프로비저닝된 리소스는 템플릿 업데이트 시 업데이트됩니다. 생성 또는 업데이트에 실패한 리소스는 재시도됩니다. 템플릿 오류로 인해 리소스 프로비저닝에 실패하고 템플릿을 수정한 경우 이 옵션을 선택합니다. FAILED 상태인 스택을 업데이트할 때 스택 업데이트를 계속하려면 Stack failure options(스택 실패 옵션)에 대해 Preserve successfully provisioned resources(성공적으로 프로비저닝된 리소스 보존)를 선택해야 합니다.
- **Roll back(롤백)** – CloudFormation은 스택을 마지막으로 알려진 안정 상태로 롤백합니다.

스택 롤백 일시 중지에는 필요한 조건

생성된 리소스를 CloudFormation에서 자동으로 롤백하거나 삭제하지 않도록 하려면 다음과 같은 조건이 충족되어야 합니다.

1. 스택을 생성하거나 업데이트할 때 성공적으로 프로비저닝된 리소스 보존 옵션을 선택해야 합니다. 그러면 전반적으로 스택 작업이 실패하더라도 성공적으로 생성된 리소스를 삭제하지 않도록 CloudFormation에 지시합니다.
2. 스택 작업이 실패하여 스택 상태가 CREATE_FAILED 또는 UPDATE_FAILED입니다.

Note

변경할 수 없는 업데이트 유형은 지원되지 않습니다.

성공적으로 프로비저닝된 리소스 보존(콘솔)

Create stack

스택 생성 작업 중 성공적으로 프로비저닝된 리소스 보존

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 스택 페이지의 오른쪽 상단에서 스택 생성을 선택하고 새 리소스 사용(표준)을 선택합니다.
3. 사전 조건 - 템플릿 준비에서 기존 템플릿 선택을 선택합니다.

4. 템플릿 지정, 아래에서 스택 템플릿을 포함하는 S3 버킷의 URL을 지정하거나 스택 템플릿 파일을 업로드하도록 선택합니다. 그리고 다음을 선택합니다.
5. Specify stack details(스택 세부 정보 지정) 페이지의 Stack name(스택 이름) 상자에 스택 이름을 입력합니다.
6. Parameters(파라미터) 섹션에서 스택 템플릿에 정의된 파라미터를 지정합니다.

기본값을 사용하여 파라미터를 사용하거나 변경할 수 있습니다.

7. 파라미터값에 만족하면 Next(다음)를 선택합니다.
8. Configure stack options(스택 옵션 구성) 페이지에서 스택에 대한 추가 옵션을 설정할 수 있습니다.
9. Stack failure options(스택 실패 옵션)에서 Preserve successfully provisioned resources(성공적으로 프로비저닝된 리소스 보존)를 선택합니다.
10. 스택 옵션에 만족하면 Next(다음)를 선택합니다.
11. Review(검토) 페이지에서 스택을 검토하고 Create stack(스택 생성)을 선택합니다.

결과: 생성에 실패한 리소스는 스택 작업이 실패할 때 스택이 롤백되지 않도록 스택 상태를 CREATE_FAILED로 전환합니다. 성공적으로 프로비저닝된 리소스는 CREATE_COMPLETE 상태입니다. Stack events(스택 이벤트) 탭에서 스택을 모니터링할 수 있습니다.

Update stack

스택 업데이트 작업 중 성공적으로 프로비저닝된 리소스 보존

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 업데이트할 스택을 선택하고 Update(업데이트)를 선택합니다.
3. Update stack(스택 지정) 페이지에서 다음 옵션 중 하나를 사용하여 스택 템플릿을 선택합니다.
 - 기존 템플릿 사용
 - Replace current template(현재 템플릿 교체)
 - Infrastructure Composer에서 템플릿 편집

설정을 수락하고 Next(다음)를 선택합니다.

4. Specify stack details(스택 세부 정보 지정) 페이지에서 스택 템플릿에 정의된 파라미터를 지정합니다.

기본값을 사용하여 파라미터를 사용하거나 변경할 수 있습니다.

5. 파라미터값에 만족하면 Next(다음)를 선택합니다.
6. Configure stack options(스택 옵션 구성) 페이지에서 스택에 대한 추가 옵션을 설정할 수 있습니다.
7. Behavior on provisioning failure(프로비저닝 실패에 대한 동작)에서 Preserve successfully provisioned resources(성공적으로 프로비저닝된 리소스 보존)를 선택합니다.
8. 스택 옵션에 만족하면 Next(다음)를 선택합니다.
9. Review(검토) 페이지에서 스택을 검토하고 Update stack(스택 업데이트)을 선택합니다.

결과: 업데이트에 실패한 리소스는 스택 상태를 UPDATE_FAILED로 전환하고 마지막으로 알려진 안정 상태로 롤백합니다. 마지막으로 알려진 안정 상태가 없는 리소스는 다음 스택 작업 시 CloudFormation에 의해 삭제됩니다. 성공적으로 프로비저닝된 리소스는 CREATE_COMPLETE 또는 UPDATE_COMPLETE 상태입니다. Stack events(스택 이벤트) 탭에서 스택을 모니터링할 수 있습니다.

Change set

Note

CREATE_FAILED 또는 UPDATE_FAILED 상태의 스택에 대해 변경 세트를 시작할 수 있지만 UPDATE_ROLLBACK_FAILED 상태의 스택에 대해 변경 세트를 시작할 수 없습니다.

변경 세트 작업 중 성공적으로 프로비저닝된 리소스 보존

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 시작하려는 변경 세트가 포함된 스택을 선택한 다음 Change sets(변경 세트) 탭을 선택합니다.
3. 변경 세트를 선택한 다음 Execute(실행)를 선택합니다.
4. Execute change set(변경 세트 실행)에서 Preserve successfully provisioned resources(성공적으로 프로비저닝된 리소스 보존) 옵션을 선택합니다.
5. Execute change set(변경 세트 실행)를 선택합니다.

결과: 업데이트에 실패한 리소스는 스택 상태를 UPDATE_FAILED로 전환하고 마지막으로 알려진 안정 상태로 롤백합니다. 마지막으로 알려진 안정 상태가 없는 리소스는 다음 스택 작업 시

CloudFormation에 의해 삭제됩니다. 성공적으로 프로비저닝된 리소스는 CREATE_COMPLETE 또는 UPDATE_COMPLETE 상태입니다. Stack events(스택 이벤트) 탭에서 스택을 모니터링할 수 있습니다.

성공적으로 프로비저닝된 리소스 보존(AWS CLI)

Create stack

스택 생성 작업 중 성공적으로 프로비저닝된 리소스 보존

[create-stack](#) 작업 중 `--disable-rollback` 옵션이나 `on-failure DO_NOTHING` 열거를 선택합니다.

1. `--disable-rollback` 옵션을 사용하여 `create-stack` 명령에 스택 이름과 템플릿을 제공합니다.

```
aws cloudformation create-stack --stack-name myteststack \  
  --template-body file://template.yaml \  
  --disable-rollback
```

명령은 다음 출력을 반환합니다.

```
{  
  "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/  
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896"  
}
```

2. `describe-stacks` 명령을 사용하여 스택의 상태를 설명합니다.

```
aws cloudformation describe-stacks --stack-name myteststack
```

명령은 다음 출력을 반환합니다.

```
{  
  "Stacks": [  
    {  
      "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/  
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",  
      "Description": "AWS CloudFormation Sample Template",  
      "Tags": [],
```

```

        "Outputs": [],
        "StackStatusReason": "The following resource(s) failed to create:
[MyBucket]",
        "CreationTime": "2013-08-23T01:02:15.422Z",
        "Capabilities": [],
        "StackName": "myteststack",
        "StackStatus": "CREATE_FAILED",
        "DisableRollback": true
    }
]
}

```

Update stack

스택 업데이트 작업 중 성공적으로 프로비저닝된 리소스 보존

1. `--disable-rollback` 옵션을 사용하여 `update-stack` 명령에 기존 스택 이름과 템플릿을 제공합니다.

```

aws cloudformation update-stack --stack-name myteststack \
    --template-url https://s3.amazonaws.com/amzn-s3-demo-bucket/updated.template
    --disable-rollback

```

명령은 다음 출력을 반환합니다.

```

{
    "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896"
}

```

2. `describe-stacks` 또는 `describe-stack-events` 명령을 사용하여 스택의 상태를 설명합니다.

```

aws cloudformation describe-stacks --stack-name myteststack

```

명령은 다음 출력을 반환합니다.

```

{
    "Stacks": [
        {

```

```

        "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",
        "Description": "AWS CloudFormation Sample Template",
        "Tags": [],
        "Outputs": [],
        "CreationTime": "2013-08-23T01:02:15.422Z",
        "Capabilities": [],
        "StackName": "myteststack",
        "StackStatus": "UPDATE_COMPLETE",
        "DisableRollback": true
    }
]
}

```

Change set

Note

CREATE_FAILED 또는 UPDATE_FAILED 상태의 스택에 대해 변경 세트를 시작할 수 있지만 UPDATE_ROLLBACK_FAILED 상태의 스택에 대해 변경 세트를 시작할 수 없습니다.

변경 세트 작업 중 성공적으로 프로비저닝된 리소스 보존

[execute-change-set](#) 작업 중 `--disable-rollback` 옵션을 지정합니다.

1. `--disable-rollback` 옵션을 사용하여 `execute-change-set` 명령에 스택 이름과 템플릿을 제공합니다.

```

aws cloudformation execute-change-set --stack-name myteststack \
    --change-set-name my-change-set --template-body file://template.yaml

```

명령은 다음 출력을 반환합니다.

```

{
  "Id": "arn:aws:cloudformation:us-east-1:123456789012:changeSet/my-change-set/
bc9555ba-a949-xmpl-bfb8-f41d04ec5784",
  "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896"
}

```

2. `--disable-rollback` 옵션으로 변경 세트를 시작합니다.

```
aws cloudformation execute-change-set --stack-name myteststack \  
  --change-set-name my-change-set --disable-rollback
```

3. `describe-stacks` 또는 `describe-stack-events` 명령을 사용하여 스택의 상태를 결정합니다.

```
aws cloudformation describe-stack-events --stack-name myteststack
```

명령은 다음 출력을 반환합니다.

```
{  
  "StackEvents": [  
    {  
      "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/  
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",  
      "EventId": "49c966a0-7b74-11ea-8071-024244bb0672",  
      "StackName": "myteststack",  
      "LogicalResourceId": " MyBucket",  
      "PhysicalResourceId": "myteststack-MyBucket-abcdefghijkl1",  
      "ResourceType": "AWS::S3::Bucket",  
      "Timestamp": "2020-04-10T21:43:17.015Z",  
      "ResourceStatus": "UPDATE_FAILED"  
      "ResourceStatusReason": "User XYZ is not allowed to perform  
S3::UpdateBucket on MyBucket"  
    }  
  ]  
}
```

4. 권한 오류를 수정하고 작업을 다시 시도합니다.

```
aws cloudformation update-stack --stack-name myteststack \  
  --use-previous-template --disable-rollback
```

명령은 다음 출력을 반환합니다.

```
{  
  "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/  
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896"  
}
```

5. `describe-stacks` 또는 `describe-stack-events` 명령을 사용하여 스택의 상태를 설명합니다.

```
aws cloudformation describe-stacks --stack-name myteststack
```

명령은 다음 출력을 반환합니다.

```
{
  "Stacks": [
    {
      "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",
      "Description": "AWS CloudFormation Sample Template",
      "Tags": [],
      "Outputs": [],
      "CreationTime": "2013-08-23T01:02:15.422Z",
      "Capabilities": [],
      "StackName": "myteststack",
      "StackStatus": "UPDATE_COMPLETE",
      "DisableRollback": true
    }
  ]
}
```

스택 롤백

[rollback-stack](#) 명령을 사용하여 스택 상태가 CREATE_FAILED 또는 UPDATE_FAILED인 스택을 마지막 안정 상태로 롤백할 수 있습니다.

다음 rollback-stack 명령에서는 지정된 스택을 롤백합니다.

```
aws cloudformation rollback-stack --stack-name myteststack
```

명령은 다음 출력을 반환합니다.

```
{
  "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896"
}
```


 Note

rollback-stack 작업은 마지막으로 알려진 안정 상태가 없는 스택을 삭제합니다.

롤백 트리거를 사용하여 경보 위반 시 CloudFormation 스택을 롤백합니다.

롤백 트리거를 사용하면 스택 생성 및 업데이트 중에 CloudFormation에서 애플리케이션의 상태를 모니터링할 수 있으며, 지정된 경보의 임계값을 애플리케이션에서 위반하는 경우 해당 작업을 롤백할 수 있습니다. 생성한 각 롤백 트리거에 대해 CloudFormation이 모니터링해야 하는 CloudWatch 경보를 지정합니다. CloudFormation은 스택 생성 또는 업데이트 작업 도중, 그리고 모든 리소스가 배포된 이후 지정된 시간 동안 지정된 경보를 모니터링합니다. 스택 작업 도중 또는 모니터링 기간 동안 ALARM 상태로 전환되는 경보가 하나라도 있을 경우 CloudFormation이 전체 스택 작업을 롤백합니다.

모니터링 시간을 0~180분 사이로 설정할 수 있습니다. CloudFormation은 스택 생성 또는 업데이트 작업이 필요한 모든 리소스를 배포한 후 이 시간 동안 모든 롤백 트리거를 모니터링합니다. 스택 작업 도중 또는 이 모니터링 기간 동안 ALARM 상태로 전환되는 경보가 하나라도 있을 경우 CloudFormation이 전체 스택 작업을 롤백합니다. 그런 다음, 업데이트 작업을 위해, 아무 경보도 ALARM 상태로 전환하지 않고 모니터링 기간이 경과한 경우 CloudFormation이 평소와 같이 기존 리소스를 폐기합니다. 모니터링 시간은 설정하지만 롤백 트리거는 지정하지 않는 경우에도 CloudFormation은 업데이트를 작업을 위해 기존 리소스를 제거하기 전에 지정된 기간 동안 대기합니다. 이 모니터링 기간을 사용하여 원하는 수동 스택 검증을 수행하고 필요에 따라 수동으로 스택 생성 또는 업데이트를 취소할 수 있습니다. 모니터링 시간을 0분으로 설정할 경우에도 CloudFormation이 스택 생성 및 업데이트 작업 도중 롤백 트리거를 모니터링하고 경보가 하나라도 ALARM 상태로 전환되면 작업을 롤백합니다. 그런 다음, 경보 위반이 없을 경우 업데이트 작업을 위해 작업이 완료되는 즉시 기존 리소스를 폐기하기 시작합니다.

기본적으로, CloudFormation은 경보가 INSUFFICIENT_DATA 상태가 아니라 ALARM 상태로 전환되는 경우에만 스택 작업을 롤백합니다. 경보가 INSUFFICIENT_DATA 상태로 전환되는 경우에도 CloudFormation이 스택 작업을 롤백하도록 하려면 누락 데이터를 breaching으로 처리하도록 CloudWatch 경보를 수정합니다. 자세한 설명은 Amazon CloudWatch 사용 설명서의 [CloudWatch 경보가 누락된 데이터를 처리하는 방법 구성](#)을 참조하세요.

CloudFormation은 업데이트 작업 도중 스택을 롤백할 때 롤백 트리거를 모니터링하지 않습니다.

최대 5개의 롤백 트리거를 추가할 수 있습니다. 롤백 트리거를 추가하려면 CloudWatch 경보의 Amazon 리소스 이름(ARN)을 지정합니다. 현재 AWS::CloudWatch::Alarm 및

AWS::CloudWatch::CompositeAlarm 유형은 롤백 트리거로 사용할 수 있습니다. CloudWatch 경보에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 [Amazon CloudWatch 경보 사용](#)을 참조하세요.

지정된 Cloudwatch 경보가 누락되는 경우 전체 스택 작업이 실패하고 롤백됩니다.

CloudWatch에 액세스하려면 자격 증명이 필요합니다. 이러한 자격 증명에는 리소스에 대한 CloudWatch 지표 데이터를 검색하는 것 같이 AWS 리소스에 액세스할 수 있는 권한이 있어야 합니다. 자세한 내용은 Amazon CloudWatch 사용 설명서의 [CloudWatch에 대한 인증 및 액세스 제어](#)를 참조하세요.

스택 생성 또는 업데이트 중에 롤백 트리거 추가

스택 생성 또는 업데이트 도중 롤백 트리거를 추가하려면(콘솔)

1. 스택 생성 또는 업데이트 시 스택 옵션 구성 페이지의 고급 옵션에서 롤백 구성 섹션을 확장합니다.
2. 0~180분 사이의 모니터링 시간을 입력합니다. 기본값은 0입니다.
3. 롤백 트리거로 사용할 CloudWatch 경보 또는 복합 경보의 ARN을 지정하고 CloudWatch 경보 ARN 추가를 선택합니다.

예를 들어 다음은 CloudWatch 경보 또는 복합 경보 `arn:aws:cloudwatch:us-east-1:123456789012:alarm:MyAlarmName`에 대한 ARN입니다.

4. 다음을 선택하고 스택의 세부 정보를 검토합니다.
5. 준비가 되면 제출을 선택하여 스택을 생성하거나 업데이트합니다.

스택 생성 또는 업데이트 도중 롤백 트리거를 추가하려면(AWS CLI)

[create-stack](#) 또는 [update-stack](#) 명령을 `--rollback-configuration` 옵션과 함께 사용합니다.

예를 들어 다음 `update-stack` 명령은 `MyCompositeAlarm`을 모니터링 기간이 5분인 롤백 트리거로 설정합니다.

```
aws cloudformation update-stack --stack-name MyStack \
  --use-previous-template \
  --rollback-configuration \
  "RollbackTriggers=[{Arn=arn:aws:cloudwatch:us-east-1:123456789012:alarm:MyCompositeAlarm,Type=AWS::CloudWatch::CompositeAlarm}],MonitoringTime=5"
```

변경 세트에 롤백 트리거 추가

변경 세트에 롤백 트리거를 추가하려면(콘솔)

1. 변경 세트 생성 시 스택 옵션 구성 페이지의 고급 옵션에서 롤백 구성 섹션을 확장합니다.
2. 0~180분 사이의 모니터링 시간을 입력합니다. 기본값은 5입니다.
3. 롤백 트리거로 사용할 CloudWatch 경보 또는 복합 경보의 ARN을 지정하고 CloudWatch 경보 ARN 추가를 선택합니다.

예를 들어 다음은 CloudWatch 경보 또는 복합 경보 `arn:aws:cloudwatch:us-east-1:123456789012:alarm:MyAlarmName`에 대한 ARN입니다.

4. 다음을 선택하고 변경 세트의 세부 정보를 검토합니다.
5. 준비가 되면 변경 세트 생성을 선택하여 변경 세트를 생성합니다.

변경 세트에 롤백 트리거를 추가하려면(AWS CLI)

--rollback-configuration 옵션과 함께 [create-change-set](#) 명령을 사용합니다.

스택에 대한 롤백 트리거 보기

스택에 대한 롤백 트리거를 보려면 Rollback configuration(롤백 구성) 섹션을 봅니다.

1. 스택 페이지의 왼쪽 목록에서 보려는 스택을 선택합니다.
2. 스택 정보 탭에서 롤백 구성 섹션을 확장하여 롤백 트리거를 봅니다.

변경 세트에 대한 롤백 트리거 보기

변경 세트에 대한 롤백 트리거를 보려면 Rollback configuration(롤백 구성) 섹션을 봅니다.

1. 스택 페이지의 왼쪽 목록에서 보려는 스택을 선택합니다.
2. 변경 세트 탭을 선택하고 보려는 변경 세트를 선택합니다.
3. 입력 탭을 선택하고 롤백 구성 섹션을 봅니다.

드리프트 감지를 사용하여 스택 및 리소스에 대한 비관리형 구성 변경 감지

CloudFormation을 통해 리소스를 관리하는 경우에도 사용자는 CloudFormation 외부의 리소스를 변경할 수 있습니다. 사용자는 리소스를 생성하는 기본 서비스를 사용하여 리소스를 직접 편집할 수 있습니다. 예를 들어 Amazon EC2 콘솔을 사용하여 CloudFormation 스택의 일부로 생성된 서버 인스턴스를 업데이트할 수 있습니다. 일부 변경 사항은 실수일 수 있으며, 일부는 시간에 민감한 작업 이벤트에 의도적으로 응답하는 것일 수 있습니다. 그와 무관하게 CloudFormation 외부에서의 변경 사항은 스택 업데이트 또는 삭제 작업을 복잡하게 만들 수 있습니다. 드리프트 감지를 사용하여 구성 변경이 CloudFormation 관리 외부에서 이루어진 스택 리소스를 식별할 수 있습니다. 교정 작업을 수행하여 스택 리소스가 스택 템플릿의 정의와 동기화될 수 있습니다. 예를 들어 드리프트된 리소스를 직접 업데이트하여 템플릿 정의가 일치할 수 있습니다. 드리프트 확인은 구성 일관성과 성공적인 스택 작업에 도움이 됩니다.

주제

- [드리프트란 무엇인가요?](#)
- [드리프트 감지 상태 코드](#)
- [드리프트 감지 시 고려 사항](#)
- [전체 CloudFormation 스택의 드리프트 감지](#)
- [개별 스택 리소스의 드리프트 감지](#)
- [가져오기 작업을 통한 드리프트 해결](#)

드리프트란 무엇인가요?

드리프트 감지 기능을 사용하면 스택의 실제 구성이 예상 구성과 다르거나 드리프트했는지 여부를 감지할 수 있습니다. CloudFormation을 사용하여 전체 스택 또는 스택 내 개별 리소스의 드리프트를 감지합니다. 리소스는 실제 속성값이 예상 속성값과 다를 경우 드리프트된 것으로 간주됩니다. 여기에는 속성 또는 리소스가 삭제된 경우가 포함됩니다. 스택은 하나 이상의 리소스가 드리프트된 경우 드리프트된 것으로 간주됩니다.

리소스가 드리프트되었는지 확인하려면 CloudFormation에서 템플릿 파라미터로 지정된 스택 템플릿 및 모든 값에서 정의된 예상 속성값을 확인합니다. 그런 다음 CloudFormation은 이러한 예상값을 스택에 현재 존재하는 리소스 속성의 실제 값과 비교합니다. 리소스는 하나 이상의 속성이 삭제되었거나 값이 변경된 경우 드리프트된 것으로 간주됩니다.

CloudFormation은 드리프트된 스택의 각 리소스에서 세부 정보를 생성합니다.

CloudFormation에서 드리프트 감지를 지원하는 AWS 리소스의 드리프트를 감지합니다. 드리프트 감지를 지원하지 않는 리소스는 드리프트 상태가 NOT_CHECKED로 할당됩니다. 드리프트 감지를 지원하는 AWS 리소스 목록은 [리소스 유형 지원](#) 섹션을 참조하세요.

또한 CloudFormation에서는 프로비저닝 가능한 프라이빗 리소스 유형, 즉 프로비저닝 유형이 FULLY_MUTABLE 또는 IMMUTABLE인 프라이빗 리소스 유형에 대한 드리프트 감지를 지원합니다. 프라이빗 리소스 유형의 리소스에 대한 드리프트 감지를 수행하려면 계정에 등록된 리소스 유형의 기본 버전이 프로비저닝 가능해야 합니다. 리소스 프로비저닝 유형에 대한 자세한 내용은 AWS CloudFormation API 참조의 [DescribeType](#) 작업과 AWS CLI Command Reference의 [DescribeType](#) 명령의 ProvisioningType 파라미터를 참조하세요. 프라이빗 리소스에 대한 자세한 내용은 [CloudFormation 레지스트리를 통해 익스텐션 관리](#) 섹션을 참조하세요.

상태가 CREATE_COMPLETE, UPDATE_COMPLETE, UPDATE_ROLLBACK_COMPLETE 및 UPDATE_ROLLBACK_FAILED인 스택에서 드리프트 감지를 수행할 수 있습니다.

스택의 드리프트를 감지할 때 CloudFormation에서 해당 스택에 속하는 중첩 스택의 드리프트를 감지하지 않습니다. 자세한 내용은 [중첩 스택을 사용하여 템플릿을 재사용 가능한 조각으로 분할](#) 섹션을 참조하세요. 대신 중첩 스택에서 직접 드리프트 감지 작업을 개시할 수 있습니다.

Note

CloudFormation에서 스택 템플릿을 통해 또는 템플릿 파라미터 지정을 통해 명시적으로 설정된 속성값에 대한 드리프트만을 감지합니다. 여기에는 리소스 속성에 대한 기본값이 포함되지 않습니다. CloudFormation에서 드리프트 확인을 목적으로 리소스 속성을 추적하도록 하려면 기본값으로 설정하는 경우에도 속성값을 명시적으로 설정합니다.

드리프트 감지 상태 코드

이 단원의 표에서는 드리프트 감지에 사용되는 다양한 상태 유형을 설명합니다.

- 드리프트 감지 작업 상태는 드리프트 작업의 현재 상태를 설명합니다.
- 드리프트 상태

스택 세트의 경우, 스택 세트에 속하는 스택 인스턴스의 드리프트 상태를 기준으로 스택 세트의 드리프트 상태를 전체적으로 설명합니다.

스택 인스턴스의 경우, 연결된 스택의 드리프트 상태를 기준으로 스택 인스턴스의 드리프트 상태를 설명합니다.

스택의 경우, 리소스의 드리프트 상태를 기준으로 스택의 드리프트 상태를 전체적으로 설명합니다.

- 리소스 드리프트 상태는 개별 리소스의 드리프트 상태를 설명합니다.

다음 표에서는 CloudFormation에서 스택 드리프트 감지 작업에 할당하는 상태 코드를 나열합니다.

드리프트 감지 작업 상태	설명
DETECTION_COMPLETE	드리프트 감지를 지원하는 스택의 모든 리소스에 대해 스택 드리프트 감지 작업이 완료되었습니다.
DETECTION_FAILED	스택에 있는 하나 이상의 리소스에 대해 스택 드리프트 감지 작업이 실패했습니다. 결과는 CloudFormation에서 드리프트 감지가 완료된 리소스에 대해 제공됩니다.
DETECTION_IN_PROGRESS	스택 드리프트 감지 작업이 현재 진행되고 있습니다.

다음 표에서는 CloudFormation에서 스택에 할당하는 드리프트 상태 코드를 나열합니다.

드리프트 상태	설명
DRIFTED	스택의 경우: 스택이 예상 템플릿 구성과 다르거나 드리프트되었습니다. 스택은 하나 이상의 리소스가 드리프트된 경우 드리프트된 것으로 간주됩니다. 스택 인스턴스의 경우: 스택 인스턴스는 연결된 스택이 드리프트되면 드리프트된 것으로 간주됩니다. 스택 세트의 경우: 하나 이상의 스택 인스턴스가 드리프트되면 스택 세트가 드리프트된 것으로 간주됩니다.
NOT_CHECKED	CloudFormation에서는 스택, 스택 세트 또는 스택 인스턴스가 예상 템플릿 구성과 다른지 여부를 확인하지 않았습니다.
IN_SYNC	각 지원되는 리소스의 현재 구성이 예상 템플릿 구성과 일치합니다. 드리프트 감지를 지원하는 리소스가 없는 스택, 스택 세트 또는 스택 인스턴스도 IN_SYNC 상태가 됩니다.

다음 표에서는 CloudFormation에서 스택 리소스에 할당하는 드리프트 상태 코드를 나열합니다.

리소스 드리프트 상태	설명
DELETED	리소스가 삭제되었기 때문에 리소스가 예상 템플릿 구성과 다릅니다.
MODIFIED	리소스가 예상 템플릿 구성과 다릅니다.
NOT_CHECKED	CloudFormation에서 리소스가 예상 템플릿 구성과 다른지 여부를 확인하지 않았습니다.
IN_SYNC	리소스의 현재 구성이 예상 템플릿 구성과 일치합니다.

다음 표에서는 CloudFormation에서 예상 템플릿 구성과 다른 리소스 속성에 할당하는 차이 유형 상태 코드를 나열합니다.

속성 차이 유형	설명
ADD	값이 배열 또는 목록 데이터 유형인 리소스 속성에 추가되었습니다.
REMOVE	속성이 현재 리소스 구성에서 제거되었습니다.
NOT_EQUAL	현재 속성값이 스택 템플릿에서 정의된 예상값과 다릅니다.

드리프트 감지 시 고려 사항

스택의 드리프트 감지를 수행하려면 사용자가 다음 권한을 보유해야 합니다.

- 스택에 포함된 드리프트 감지를 지원하는 각 리소스에 대한 읽기 권한. 예를 들어 스택에 `AWS::EC2::Instance` 리소스가 포함된 경우 `ec2:DescribeInstances` 권한을 보유하고 스택에서 드리프트 감지를 수행해야 합니다.
- `cloudformation:DetectStackDrift`
- `cloudformation:DetectStackResourceDrift`
- `cloudformation:BatchDescribeTypeConfigurations`

CloudFormation에 권한을 설정하는 방법에 대한 자세한 내용은 [AWS Identity and Access Management를 사용하여 CloudFormation 액세스 제어](#) 섹션을 참조하세요.

특정 엣지 케이스에서 CloudFormation은 정확한 드리프트 결과를 항상 반환할 수는 없습니다. 드리프트 감지 결과를 적절하게 해석하려면 이러한 엣지 케이스를 인식해야 합니다.

- 실제로 리소스에 대한 책임이 있는 기본 서비스로부터 속성에 공급되는 기본값인 경우 특정 케이스에서 속성 어레이에 포함된 객체는 드리프트로 보고됩니다.
- 특정 리소스에는 관련 리소스와 연결 관계가 있습니다. 이를 통해 리소스에서 동일하거나 다른 템플릿에 정의된 또 다른 리소스에 대한 속성값을 연결 또는 제거할 수 있습니다. 예를 들어 `AWS::EC2::SecurityGroupIngress` 및 `AWS::EC2::SecurityGroupEgress` 리소스는 `AWS::EC2::SecurityGroup` 리소스에서 값을 연결 또는 제거하는 데 사용될 수 있습니다. 이러한 경우 CloudFormation에서 드리프트 비교 수행 전에 스택 템플릿을 분석합니다. 하지만 CloudFormation에서 스택에 걸쳐 이 분석을 수행할 수 없고, 정확한 드리프트 결과를 반환할 수 없습니다. 이 경우 연결된 리소스는 다른 스택에 위치합니다.

드리프트 감지를 지원하고 다른 리소스로부터의 연결을 허용 또는 필요로 하는 리소스는 다음과 같습니다.

리소스 유형	연결 리소스 유형
<code>AWS::SNS::Topic</code>	<code>AWS::SNS::Subscription</code>
<code>AWS::IAM::User</code>	<code>AWS::IAM::UserToGroupAddition</code>
<code>AWS::IAM::Group</code>	<code>AWS::IAM::ManagedPolicy</code>
<code>AWS::IAM::Role</code>	<code>AWS::IAM::Policy</code>
<code>AWS::IAM::User</code>	<code>AWS::IAM::RolePolicy</code>
<code>AWS::ElasticLoadBalancingV2::Listener</code>	<code>AWS::ElasticLoadBalancingV2::ListenerCertificate</code>
<code>AWS::EC2::SecurityGroup</code>	<code>AWS::EC2::SecurityGroupEgress</code> <code>AWS::EC2::SecurityGroupIngress</code>

- CloudFormation에서는 모든 리소스의 KMSKeyId 속성에 대해 드리프트 감지를 수행하지 않습니다. 여러 별칭에서 AWS KMS 키를 참조할 수 있기 때문에 CloudFormation에서 이 속성에 대한 정확한 드리프트 결과를 일관적으로 보장할 수 없습니다.
- 특성에 따라 스택 템플릿에서 지정할 수 있는 특정 리소스 속성이 있습니다. CloudFormation에서는 결과로 나온 스택 리소스의 속성을 비교할 수 없습니다. 따라서 이러한 속성은 드리프트 감지 결과에 포함될 수 없습니다. 이러한 속성은 두 가지 넓은 범주로 나뉩니다.
 - CloudFormation에서 스택 템플릿의 최초 리소스 속성값으로 매핑할 수 없는 속성값입니다.

예를 들어 CloudFormation에서 Lambda 함수의 소스 코드를 [AWS::Lambda::Function](#) 리소스의 [Code](#) 속성 유형으로 매핑할 수 없으므로 CloudFormation에서 이를 드리프트 감지 결과에 포함할 수 없습니다.

- 리소스에 대해 책임이 있는 서비스에서 반환하지 않는 속성값입니다.

설계 면에서 리소스가 속하는 서비스에 의해 반환되지 않는 특정 속성값이 있습니다. 암호 또는 노출되어서는 안 되는 기타 민감한 데이터와 같은 기밀 정보가 포함되는 경우가 있습니다. 예를 들어, IAM 서비스에서 [AWS::IAM::User LoginProfile](#) 속성 유형의 Password 속성값을 반환하지 않으며, 따라서 CloudFormation은 드리프트 감지 결과에 포함될 수 없습니다.

- 어레이에 있는 객체는 실제로 서비스 기본값이 될 수 있지만 수동으로 추가된 드리프트는 아닙니다.
- 오답지가 발생하는 경우 CloudFormation 콘솔의 피드백 링크를 사용하여 의견을 보내거나 [AWS re:Post](#)을 통해 문의하세요.
- 일부 속성에는 같지만 동일하지는 않은 입력 값이 있을 수 있습니다. 오답지를 방지하려면 예상 구성이 실제 구성과 일치하는지 확인해야 합니다.
- 예를 들어, 리소스 속성의 예상 구성은 1024MB이고 동일한 리소스 속성의 실제 구성은 1GB일 수 있습니다. 1024MB와 1GB는 같지만 동일하지는 않습니다.

이 리소스 속성에서 드리프트 감지를 실행하면 드리프트된 결과를 알립니다.

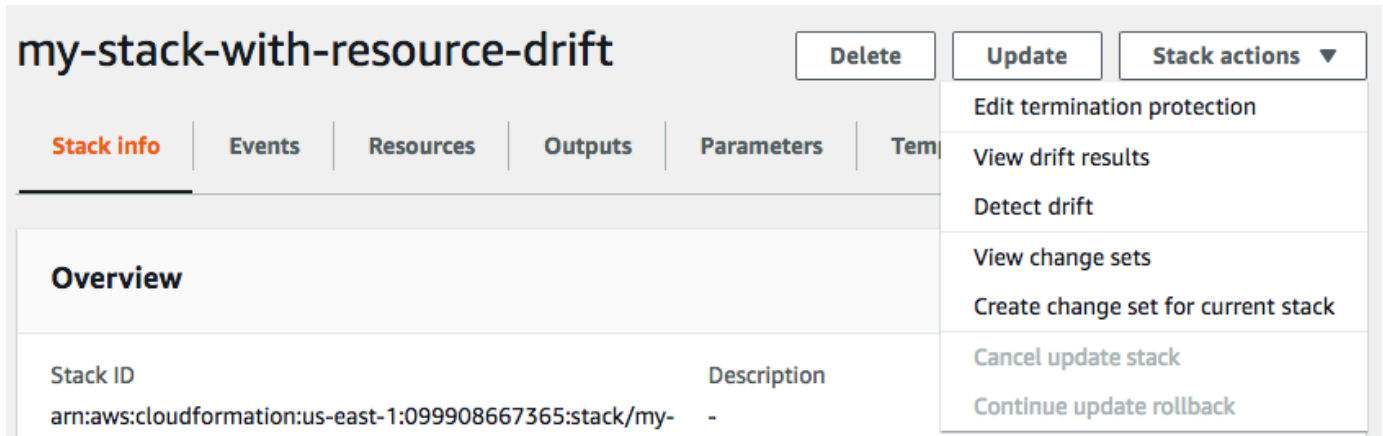
이러한 오답지를 방지하려면 리소스 속성의 예상 구성을 1024MB로 변경한 다음 드리프트 감지를 실행하세요.

전체 CloudFormation 스택의 드리프트 감지

스택에서 드리프트 감지 작업을 수행하면 스택이 예정 템플릿 구성에서 드리프트되었는지 확인하고, 드리프트 감지를 지원하는 스택의 각 리소스에 대한 드리프트 상태 관련 세부 정보를 반환합니다.

AWS Management Console을 사용하여 전체 스택의 드리프트를 감지하려면

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 스택 목록에서 드리프트 감지를 수행할 스택을 선택하십시오. 스택 세부 정보 창에서 Stack actions(스택 작업)를 선택한 후 Detect drift(드리프트 감지)를 선택합니다.



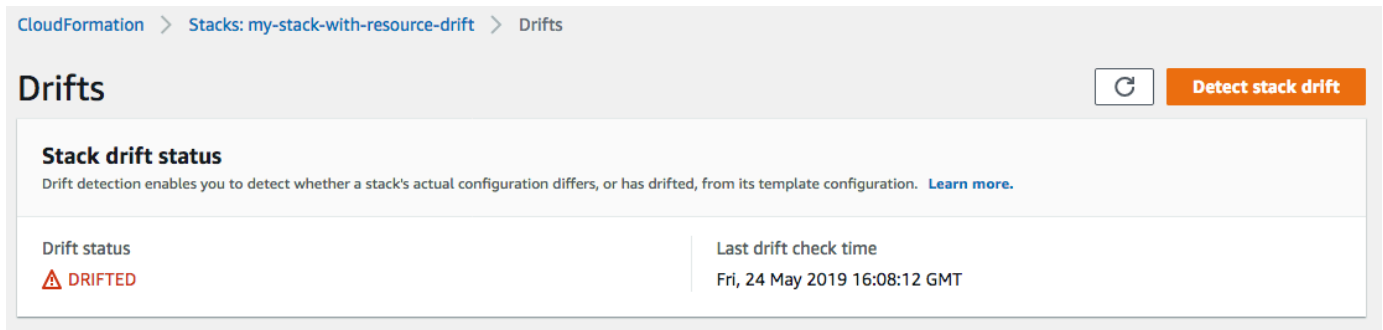
CloudFormation은 선택한 스택에 대해 드리프트 감지가 시작되었음을 나타내는 정보 표시줄을 표시합니다.

3. CloudFormation에서 드리프트 감지 작업을 완료할 때까지 기다립니다. 드리프트 감지 작업이 완료되면 CloudFormation은 스택의 [드리프트 상태(Drift status)] 및 [마지막 드리프트 확인 시간 (Last drift check time)]을 업데이트합니다. 이 필드는 스택 세부 정보 페이지에서 Stack info(스택 정보) 창의 Overview(개요) 섹션에 나열됩니다.

드리프트 감지 작업은 몇 분이 걸릴 수 있으며, 스택에 포함된 리소스의 수에 따라 다릅니다. 동시에 해당 스택에서 단일 드리프트 감지 작업만을 실행할 수 있습니다. CloudFormation은 정보 표시줄을 닫은 후에도 드리프트 감지 작업을 계속합니다.

4. 스택 및 리소스에 대한 드리프트 감지 결과를 검토합니다. 스택을 선택한 상태에서 Stack actions(스택 작업) 메뉴에서 View drift results(드리프트 결과 보기)를 선택하십시오.

CloudFormation은 스택 또는 개별 리소스에서 드리프트 감지가 개시된 마지막 시간에 더해 스택의 전체 드리프트 상태를 나열합니다. 스택은 하나 이상의 리소스가 드리프트 된 경우 드리프트 된 것으로 간주됩니다.



[리소스 드리프트 상태(Resource drift status)] 섹션에서 CloudFormation은 각 스택 리소스, 드리프트 상태 및 리소스에서 드리프트 감지가 개시된 마지막 시간을 나열합니다. 각 리소스의 논리적 ID 및 물리적 ID가 표시되어 식별에 도움이 됩니다. 추가로 상태가 MODIFIED인 리소스의 경우 CloudFormation에서 리소스 드리프트 세부 정보를 표시합니다.

드리프트 상태 열을 사용하여 드리프트 상태에 따라 리소스를 정렬할 수 있습니다.

- 수정된 리소스의 세부 정보를 보려면
 - 수정된 리소스를 선택한 상태에서 View drift details(드리프트 세부 정보 보기)를 선택하십시오.

CloudFormation은 해당 리소스에 대한 드리프트 세부 정보 페이지를 표시합니다. 이 페이지는 리소스의 예상 및 현재 속성 값, 그리고 둘 사이의 차이를 나열합니다.

차이를 강조 표시하려면 차이 섹션에서 속성 이름을 선택하십시오.

- 추가된 속성은 Details(세부 정보) 섹션의 Current(현재) 열에 녹색으로 강조 표시됩니다.
- 삭제된 속성은 Details(세부 정보) 섹션의 Expected(예상) 열에 붉은색으로 강조 표시됩니다.
- 값이 변경된 속성의 경우 Expected 및 현재 열 모두에서 노란색으로 강조 표시됩니다.

Differences (4)
< 1 >

☒	Property	Change	Expected value	Current value
☒	AttributeDefinitions.1	ADD	-	{"AttributeName":"stuff","AttributeType":"S"}
☒	ProvisionedThroughput.ReadCapacityUnits	NOT_EQUAL	5	6
☒	ProvisionedThroughput.WriteCapacityUnits	NOT_EQUAL	5	4
☒	Tags	REMOVE	[{"Key":"test","Value":"test"}]	-

Details

Expected

```

{
  "AttributeDefinitions": [
    {
      "AttributeName": "Name",
      "AttributeType": "S"
    },
  ],
  "KeySchema": [
    {
      "AttributeName": "Name",
      "KeyType": "HASH"
    }
  ],
  "ProvisionedThroughput": {
    "ReadCapacityUnits": 5,
    "WriteCapacityUnits": 5
  },
  "Tags": [
    {
      "Key": "test",
      "Value": "test"
    }
  ]
}

```

Actual

```

{
  "AttributeDefinitions": [
    {
      "AttributeName": "Name",
      "AttributeType": "S"
    },
    {
      "AttributeName": "stuff",
      "AttributeType": "S"
    }
  ],
  "KeySchema": [
    {
      "AttributeName": "Name",
      "KeyType": "HASH"
    }
  ],
  "ProvisionedThroughput": {
    "ReadCapacityUnits": 6,
    "WriteCapacityUnits": 4
  }
}

```

AWS CLI를 사용하여 전체 스택의 드리프트를 감지하려면

Important

스택의 마지막 드리프트 확인 시간을 검토하고 오래된 데이터를 사용하지 않도록 리소스 드리프트 결과에 표시된 타임스탬프보다 이전인지 확인합니다.

AWS CLI를 사용하여 전체 스택의 드리프트를 감지하려면 AWS CLI 명령을 사용합니다.

- detect-stack-drift는 스택의 드리프트 감지 작업을 개시합니다.
- describe-stack-drift-detection-status는 스택 드리프트 감지 작업의 상태를 모니터링합니다.
- describe-stack-resource-drifts는 스택 드리프트 감지 작업의 세부 정보를 검토합니다.

1. detect-stack-drift를 사용하여 전체 스택의 드리프트를 감지합니다. 스택 이름 또는 ARN을 지정합니다. 또한 이 드리프트 감지 작업에 대한 필터로 사용하고자 하는 특정 리소스의 논리적 ID를 지정할 수도 있습니다.

```
aws cloudformation detect-stack-drift --stack-name my-stack-with-resource-drift
```

출력:

```
{
  "StackDriftDetectionId": "624af370-311a-11e8-b6b7-500cexample"
}
```

2. 스택 드리프트 감지 작업은 오래 걸릴 수 있기 때문에 describe-stack-drift-detection-status를 사용하여 스택 드리프트 감지 작업의 상태를 모니터링합니다. 이 명령은 detect-stack-drift 명령에서 반환된 스택 드리프트 감지 ID를 가져옵니다.

아래 예제에서 위 예제의 detect-stack-drift에서 반환된 스택 드리프트 감지 ID를 가져와 이를 파라미터로 describe-stack-drift-detection-status에 전달했습니다. 파라미터는 드리프트 감지 작업이 완료되었고, 단일 스택 리소스가 드리프트되었으며, 전체 스택이 결과적으로 드리프트된 것으로 간주되는 것을 보여주는 작업 세부 정보를 반환합니다.

```
aws cloudformation describe-stack-drift-detection-status --stack-drift-detection-id 624af370-311a-11e8-b6b7-500cexample
```

출력:

```
{
  "StackId": "arn:aws:cloudformation:us-east-1:099908667365:stack/my-stack-with-resource-drift/489e5570-df85-11e7-a7d9-50example",
  "StackDriftDetectionId": "624af370-311a-11e8-b6b7-500cexample",
  "StackDriftStatus": "DRIFTED",
  "Timestamp": "2018-03-26T17:23:22.279Z",
  "DetectionStatus": "DETECTION_COMPLETE",
  "DriftedStackResourceCount": 1
}
```

```
}

```

3. 스택 드리프트 감지 작업이 완료될 때 `describe-stack-resource-drifts` 명령을 사용하여 결과를 검토합니다. 결과에는 드리프트된 리소스에 대한 실제 및 예상 속성 값이 포함됩니다.

아래 예에서는 `--stack-resource-drift-status-filters` 옵션을 사용하여 수정 또는 삭제된 리소스에 대한 스택 드리프트 정보를 요청합니다. 요청을 통해 수정된 리소스에 대한 정보가 반환되며, 여기에는 값이 변경된 속성에 대한 세부 정보가 포함됩니다. 삭제된 리소스가 없었습니다.

```
aws cloudformation describe-stack-resource-drifts --stack-name my-stack-with-resource-drift --stack-resource-drift-status-filters MODIFIED DELETED
```

출력:

```
{
  "StackResourceDrifts": [
    {
      "StackId": "arn:aws:cloudformation:us-east-1:099908667365:stack/my-stack-with-resource-drift/489e5570-df85-11e7-a7d9-50example",
      "ActualProperties": "{\"ReceiveMessageWaitTimeSeconds\":0, \"DelaySeconds\":120, \"RedrivePolicy\":{\"deadLetterTargetArn\":\"arn:aws:sqs:us-east-1:099908667365:my-stack-with-resource-drift-DLQ-1BCY7HHD5QIM3\", \"maxReceiveCount\":12}, \"MessageRetentionPeriod\":345600, \"MaximumMessageSize\":262144, \"VisibilityTimeout\":60, \"QueueName\":\"my-stack-with-resource-drift-Queue-494PBHC076H4\"}",
      "ResourceType": "AWS::SQS::Queue",
      "Timestamp": "2018-03-26T17:23:34.489Z",
      "PhysicalResourceId": "https://sqs.us-east-1.amazonaws.com/099908667365/my-stack-with-resource-drift-Queue-494PBHC076H4",
      "StackResourceDriftStatus": "MODIFIED",
      "ExpectedProperties": "{\"ReceiveMessageWaitTimeSeconds\":0, \"DelaySeconds\":20, \"RedrivePolicy\":{\"deadLetterTargetArn\":\"arn:aws:sqs:us-east-1:099908667365:my-stack-with-resource-drift-DLQ-1BCY7HHD5QIM3\", \"maxReceiveCount\":10}, \"MessageRetentionPeriod\":345600, \"MaximumMessageSize\":262144, \"VisibilityTimeout\":60, \"QueueName\":\"my-stack-with-resource-drift-Queue-494PBHC076H4\"}",
      "PropertyDifferences": [
        {
          "PropertyPath": "/DelaySeconds",
          "ActualValue": "120",

```

```

        "ExpectedValue": "20",
        "DifferenceType": "NOT_EQUAL"
    },
    {
        "PropertyPath": "/RedrivePolicy/maxReceiveCount",
        "ActualValue": "12",
        "ExpectedValue": "10",
        "DifferenceType": "NOT_EQUAL"
    }
],
"LogicalResourceId": "Queue"
}
]
}

```

개별 스택 리소스의 드리프트 감지

전체 스택이 아닌 하나의 스택 내 특정 리소스의 드리프트를 감지할 수 있습니다. 이는 특히 특정 리소스가 예상 템플릿 구성과 일치하는지 다시 확인해야 하는 경우 유용합니다.

리소스의 드리프트 감지를 수행할 때 CloudFormation에서는 전체 스택 드리프트 상태와 해당되는 경우 [마지막 드리프트 확인 시간(Last drift check time)]을 업데이트합니다. 예를 들어 드리프트 상태가 IN_SYNC인 스택이 있다고 가정합니다. CloudFormation에서 스택에 포함된 하나 이상의 리소스에 드리프트 감지를 수행하도록 하고, CloudFormation에서 하나 이상의 리소스가 드리프트되었음을 감지합니다. CloudFormation은 스택 드리프트 상태를 DRIFTED로 업데이트합니다. 반대로 드리프트된 단일 리소스로 인해 드리프트 상태가 DRIFTED인 스택이 있다고 가정합니다. 리소스를 예상 속성 값으로 설정한 다음 다시 리소스의 드리프트를 탐지하는 경우 CloudFormation에서 전체 스택에 대한 드리프트를 다시 탐지할 필요 없이 리소스 드리프트 상태 및 스택 드리프트 상태 모두를 IN_SYNC로 업데이트합니다.

AWS Management Console을 사용하여 개별 리소스의 드리프트를 감지하려면

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 스택 목록에서 리소스가 포함된 스택을 선택합니다. CloudFormation이 해당 스택에 대한 스택 세부 정보를 표시합니다.
3. 왼쪽 탐색 창의 스택에서 스택 동작을 선택한 다음 드리프트 감지를 선택합니다.
4. 리소스 드리프트 상태에서 리소스를 선택한 다음 리소스 드리프트 상태를 선택합니다.

CloudFormation에서는 선택한 리소스의 드리프트 감지를 수행합니다. 성공하면 CloudFormation은 리소스의 드리프트 상태를 업데이트하고 필요한 경우 전체 스택 드리프트 상태를 업데이트합니다. CloudFormation은 또한 드리프트 감지가 리소스 및 스택 전체에서 마지막으로 수행된 시간에 대한 타임스탬프를 업데이트합니다. 리소스가 수정된 경우 CloudFormation에서 리소스의 예상 및 현재 속성 값에 대한 세부 드리프트 정보를 표시합니다.

5. 리소스에 대한 드리프트 감지 결과를 검토합니다.

- 수정된 리소스의 세부 정보를 보려면
 - 수정된 리소스를 선택한 상태에서 View drift details(드리프트 세부 정보 보기)를 선택하십시오.

CloudFormation에서 리소스의 예상 및 현재 속성 값, 그리고 둘 사이의 차이를 비롯한 해당 리소스의 드리프트 세부 정보를 표시합니다.

차이를 강조 표시하려면 차이 섹션에서 속성 이름을 선택하십시오.

- 추가된 속성은 Details(세부 정보) 섹션의 Current(현재) 열에 녹색으로 강조 표시됩니다.
- 삭제된 속성은 Details(세부 정보) 섹션의 Expected(예상) 열에 붉은색으로 강조 표시됩니다.
- 값이 변경된 속성의 경우 Expected 및 현재 열 모두에서 노란색으로 강조 표시됩니다.

Differences (4)
< 1 >

☒	Property	Change	Expected value	Current value
☒	AttributeDefinitions.1	ADD	-	{"AttributeName":"stuff","AttributeType":"S"}
☒	ProvisionedThroughput.ReadCapacityUnits	NOT_EQUAL	5	6
☒	ProvisionedThroughput.WriteCapacityUnits	NOT_EQUAL	5	4
☒	Tags	REMOVE	[{"Key":"test","Value":"test"}]	-

Details

Expected

Actual

```

{
  "AttributeDefinitions": [
    {
      "AttributeName": "Name",
      "AttributeType": "S"
    },
  ],
  "KeySchema": [
    {
      "AttributeName": "Name",
      "KeyType": "HASH"
    }
  ],
  "ProvisionedThroughput": {
    "ReadCapacityUnits": 5,
    "WriteCapacityUnits": 5
  },
  "Tags": [
    {
      "Key": "test",
      "Value": "test"
    }
  ]
}

```

```

{
  "AttributeDefinitions": [
    {
      "AttributeName": "Name",
      "AttributeType": "S"
    },
    {
      "AttributeName": "stuff",
      "AttributeType": "S"
    }
  ],
  "KeySchema": [
    {
      "AttributeName": "Name",
      "KeyType": "HASH"
    }
  ],
  "ProvisionedThroughput": {
    "ReadCapacityUnits": 6,
    "WriteCapacityUnits": 4
  }
}

```

AWS CLI를 사용하여 개별 리소스의 드리프트를 감지하려면

•

⚠ Important

스택 리소스의 마지막 드리프트 확인 시간을 검토하고 오래된 데이터를 사용하지 않도록 리소스 드리프트 결과에 표시된 타임스탬프보다 이전인지 확인합니다.

AWS CLI를 사용하여 개별 리소스의 드리프트를 감지하려면 `detect-stack-resource-drift` 명령을 사용합니다. 리소스와 리소스가 포함된 스택의 논리적 ID를 지정합니다.

다음 예에서는 특정 스택 리소스인 `my-drifted-resource`에서 드리프트 탐지 작업을 실행합니다. 응답을 통해 리소스가 수정되었음을 확인하는 정보가 반환되며, 여기에는 값이 변경된 속성에 대한 세부 정보가 포함됩니다.

```
aws cloudformation detect-stack-resource-drift \
  --stack-name my-stack-with-resource-drift \
  --logical-resource-id my-drifted-resource
```

출력:

```
{
  "StackResourceDrift": {
    "StackId": "arn:aws:cloudformation:us-east-1:099908667365:stack/my-stack-with-resource-drift/489e5570-df85-11e7-a7d9-50example",
    "ActualProperties": "{\"ReceiveMessageWaitTimeSeconds\":0,\"DelaySeconds\":120,\"RedrivePolicy\":{\"deadLetterTargetArn\":\"arn:aws:sqs:us-east-1:099908667365:my-stack-with-resource-drift-DLQ-1BCY7HHD5QIM3\",\"maxReceiveCount\":12},\"MessageRetentionPeriod\":345600,\"MaximumMessageSize\":262144,\"VisibilityTimeout\":60,\"QueueName\":\"my-stack-with-resource-drift-Queue-494PBHC076H4\"}\",
    "ResourceType": "AWS::SQS::Queue",
    "Timestamp": "2018-03-26T18:54:28.462Z",
    "PhysicalResourceId": "https://sqs.us-east-1.amazonaws.com/099908667365/my-stack-with-resource-drift-Queue-494PBHC076H4",
    "StackResourceDriftStatus": "MODIFIED",
    "ExpectedProperties": "{\"ReceiveMessageWaitTimeSeconds\":0,\"DelaySeconds\":20,\"RedrivePolicy\":{\"deadLetterTargetArn\":\"arn:aws:sqs:us-east-1:099908667365:my-stack-with-resource-drift-DLQ-1BCY7HHD5QIM3\",\"maxReceiveCount\":10},\"MessageRetentionPeriod\":345600,\"MaximumMessageSize\":262144,\"VisibilityTimeout\":60,\"QueueName\":\"my-stack-with-resource-drift-Queue-494PBHC076H4\"}\",
    "PropertyDifferences": [
      {
        "PropertyPath": "/DelaySeconds",
        "ActualValue": "120",
        "ExpectedValue": "20",
        "DifferenceType": "NOT_EQUAL"
```

```

    },
    {
      "PropertyPath": "/RedrivePolicy/maxReceiveCount",
      "ActualValue": "12",
      "ExpectedValue": "10",
      "DifferenceType": "NOT_EQUAL"
    }
  ],
  "LogicalResourceId": "my-drifted-resource"
}

```

가져오기 작업을 통한 드리프트 해결

리소스의 구성이 의도된 구성에서 드리프트되어 새 구성을 의도된 구성으로 적용하려는 경우가 있을 수 있습니다. 대부분의 경우 스택 템플릿의 리소스 정의를 새 구성으로 업데이트하여 드리프트 결과를 해결한 다음 스택 업데이트를 수행합니다. 그러나 새 구성에서 교체가 필요한 리소스 속성을 업데이트 하면 스택 업데이트 중에 리소스가 다시 생성됩니다. 기존 리소스를 유지하려면 리소스 가져오기 기능을 사용하여 리소스를 업데이트하고 리소스를 교체하지 않으면서 드리프트 결과를 해결할 수 있습니다.

가져오기 작업을 통해 리소스에 대한 드리프트를 해결하는 기본 단계는 다음과 같습니다.

- [Retain으로 설정된 DeletionPolicy 속성을 리소스에 추가합니다](#). 이렇게 하면 기존 리소스가 스택에서 제거될 때 삭제되지 않고 보존됩니다.
- [템플릿에서 리소스를 제거하고 스택 업데이트 작업을 실행합니다](#). 이렇게 하면 스택에서 리소스가 제거되지만 삭제되지 않습니다.
- [스택 템플릿에서 리소스의 실제 상태를 설명한 다음 기존 리소스를 다시 스택으로 가져옵니다](#). 이렇게 하면 리소스가 스택에 다시 추가되고 드리프트 결과를 유발하는 속성 차이가 해결됩니다.

리소스 가져오기에 대한 자세한 내용은 [수동으로 CloudFormation 스택에 AWS 리소스 가져오기](#) 섹션을 참조하세요. 가져오기를 지원하는 리소스의 목록은 [리소스 유형 지원](#) 섹션을 참조하세요.

다음 예제에서는 `templateToImport.json`이라는 템플릿을 사용합니다.

Example JSON

```

{
  "AWSTemplateFormatVersion": "2010-09-09",

```

```
"Description": "Import test",
"Resources": {
  "ServiceTable":{
    "Type":"AWS::DynamoDB::Table",
    "Properties":{
      "TableName":"Service",
      "AttributeDefinitions":[
        {
          "AttributeName":"key",
          "AttributeType":"S"
        }
      ],
      "KeySchema":[
        {
          "AttributeName":"key",
          "KeyType":"HASH"
        }
      ],
      "BillingMode": "PROVISIONED",
      "ProvisionedThroughput":{
        "ReadCapacityUnits":5,
        "WriteCapacityUnits":1
      }
    }
  },
  "GamesTable": {
    "Type": "AWS::DynamoDB::Table",
    "Properties": {
      "TableName": "Games",
      "AttributeDefinitions": [
        {
          "AttributeName": "key",
          "AttributeType": "S"
        }
      ],
      "KeySchema": [
        {
          "AttributeName": "key",
          "KeyType": "HASH"
        }
      ],
      "BillingMode": "PROVISIONED",
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 5,
```

```

        "WriteCapacityUnits": 1
      }
    }
  }
}

```

Example YAML

```

AWSTemplateFormatVersion: 2010-09-09
Description: Import test
Resources:
  ServiceTable:
    Type: AWS::DynamoDB::Table
    Properties:
      TableName: Service
      AttributeDefinitions:
        - AttributeName: key
          AttributeType: S
      KeySchema:
        - AttributeName: key
          KeyType: HASH
      BillingMode: PROVISIONED
      ProvisionedThroughput:
        ReadCapacityUnits: 5
        WriteCapacityUnits: 1
  GamesTable:
    Type: AWS::DynamoDB::Table
    Properties:
      TableName: Games
      AttributeDefinitions:
        - AttributeName: key
          AttributeType: S
      KeySchema:
        - AttributeName: key
          KeyType: HASH
      BillingMode: PROVISIONED
      ProvisionedThroughput:
        ReadCapacityUnits: 5
        WriteCapacityUnits: 1

```

이 예제에서는 사용자가 CloudFormation 외부의 리소스를 변경했다고 가정해 보겠습니다. 드리프트 감지를 실행한 후 GamesTable의 BillingMode가 PAY_PER_REQUEST로 수정된 것을 발견했습니다. 드리프트 탐지에 대한 자세한 내용은 [드리프트 감지를 사용하여 스택 및 리소스에 대한 비관리형 구성 변경 감지](#) 섹션을 참조하세요.

Differences (3)					< 1 >
☒	Property	Change	Expected value	Current value	
☒	BillingMode	NOT_EQUAL	PROVISIONED	PAY_PER_REQUEST	
☒	ProvisionedThroughput.ReadCapacityUnits	NOT_EQUAL	5	0	
☒	ProvisionedThroughput.WriteCapacityUnits	NOT_EQUAL	1	0	

Expected	Actual
<pre>{ "AttributeDefinitions": [{ "AttributeName": "key", "AttributeType": "S" }], "BillingMode": "PROVISIONED", "KeySchema": [{ "AttributeName": "key", "KeyType": "HASH" }], "ProvisionedThroughput": { "ReadCapacityUnits": 5, "WriteCapacityUnits": 1 }, "TableName": "Games" }</pre>	<pre>{ "AttributeDefinitions": [{ "AttributeName": "key", "AttributeType": "S" }], "BillingMode": "PAY_PER_REQUEST", "KeySchema": [{ "AttributeName": "key", "KeyType": "HASH" }], "ProvisionedThroughput": { "ReadCapacityUnits": 0, "WriteCapacityUnits": 0 }, "TableName": "Games" }</pre>

스택이 오래되었고 리소스는 라이브 상태이지만 의도된 리소스 구성을 보존하려고 합니다. 가져오기 작업을 통해 서비스 중단 없이 드리프트를 해결하면 이렇게 할 수 있습니다.

CloudFormation 콘솔을 사용하여 가져오기 작업으로 드리프트 해결

1단계. 보존 삭제 정책을 사용하여 스택 업데이트

Retain 옵션과 함께 **DeletionPolicy** 속성을 사용하여 스택을 업데이트하려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 스택(Stacks) 페이지에서 드리프트된 스택을 선택합니다.
3. 업데이트(Update)를 선택한 다음 스택 세부 정보(Stack details) 창에서 현재 템플릿 교체(Replace current template)를 선택합니다.
4. 템플릿 지정(Specify template) 페이지에서 다음 방법 중 하나를 사용하여 Retain 옵션과 함께 DeletionPolicy 속성이 포함되어 있는 업데이트된 템플릿을 제공합니다.
 - Amazon S3 URL을 선택한 다음, 텍스트 상자에 템플릿의 URL을 지정합니다.
 - 템플릿 파일 업로드를 선택한 다음 템플릿을 찾습니다.

그리고 다음을 선택합니다.

5. 스택 세부 정보 지정(Specify stack details) 페이지를 검토하고 다음(Next)을 선택합니다.

6. 스택 옵션 구성(Configure stack options) 페이지를 검토하고 다음(Next)을 선택합니다.
7. **stack-name** 검토(Review stack-name) 페이지에서 스택 업데이트(Update stack)를 선택합니다.

결과: 스택의 이벤트(Events) 페이지에서 상태가 UPDATE_COMPLETE입니다.

서비스를 중단하지 않고 가져오기 작업을 통해 드리프트를 해결하려면 스택에서 제거할 리소스에 대해 Retain [DeletionPolicy](#)를 지정합니다. 다음 예에서는 Retain으로 설정된 [DeletionPolicy](#) 속성을 GameTable 리소스에 추가했습니다.

Example JSON

```
"GameTable": {
  "Type": "AWS::DynamoDB::Table",
  "DeletionPolicy": "Retain",
  "Properties": {
    "TableName": "Games",
```

Example YAML

```
GameTable:
  Type: AWS::DynamoDB::Table
  DeletionPolicy: Retain
  Properties:
    TableName: Games
```

2단계. 드리프트된 리소스, 관련 파라미터 및 출력 제거

드리프트된 리소스, 관련 파라미터 및 출력을 제거하려면

1. 업데이트(Update)를 선택한 다음 스택 세부 정보(Stack details) 창에서 현재 템플릿 교체(Replace current template)를 선택합니다.
2. 템플릿 지정(Specify template) 페이지에서 다음 방법 중 하나를 사용하여 스택 템플릿에서 제거된 리소스, 관련 파라미터 및 출력을 업데이트된 템플릿에 제공합니다.
 - Amazon S3 URL을 선택한 다음, 텍스트 상자에 템플릿의 URL을 지정합니다.
 - 템플릿 파일 업로드를 선택한 다음 템플릿을 찾습니다.

그리고 다음을 선택합니다.

3. 스택 세부 정보 지정(Specify stack details) 페이지를 검토하고 다음(Next)을 선택합니다.
4. 스택 옵션 구성(Configure stack options) 페이지를 검토하고 다음(Next)을 선택합니다.
5. **stack-name** 검토(Review stack-name) 페이지에서 스택 업데이트(Update stack)를 선택합니다.

결과: 스택의 Events(이벤트) 페이지에서 Logical ID(논리적 ID) GamesTable이 DELETE_SKIPPED 상태입니다.

CloudFormation이 스택 업데이트 작업을 완료할 때까지 기다립니다. 스택 업데이트 작업이 완료되면 스택 템플릿에서 리소스, 관련 파라미터 및 출력을 제거합니다. 그런 다음 업데이트된 템플릿을 가져옵니다. 이러한 작업을 완료한 후 예제 템플릿은 다음과 같습니다.

Example JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Import test",
  "Resources": {
    "ServiceTable": {
      "Type": "AWS::DynamoDB::Table",
      "Properties": {
        "TableName": "Service",
        "AttributeDefinitions": [
          {
            "AttributeName": "key",
            "AttributeType": "S"
          }
        ],
        "KeySchema": [
          {
            "AttributeName": "key",
            "KeyType": "HASH"
          }
        ],
        "BillingMode": "PROVISIONED",
        "ProvisionedThroughput": {
          "ReadCapacityUnits": 5,
          "WriteCapacityUnits": 1
        }
      }
    }
  }
}
```


}

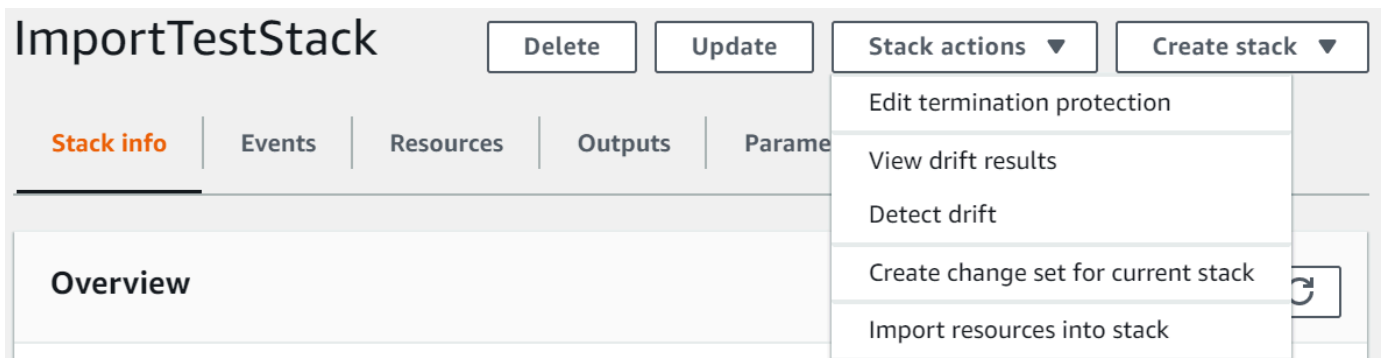
Example YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: Import test
Resources:
  ServiceTable:
    Type: AWS::DynamoDB::Table
    Properties:
      TableName: Service
      AttributeDefinitions:
        - AttributeName: key
          AttributeType: S
      KeySchema:
        - AttributeName: key
          KeyType: HASH
      BillingMode: PROVISIONED
      ProvisionedThroughput:
        ReadCapacityUnits: 5
        WriteCapacityUnits: 1
```

3단계. 리소스의 라이브 상태와 일치하도록 템플릿 업데이트

리소스의 라이브 상태와 일치하도록 템플릿을 업데이트하려면

- 업데이트된 템플릿을 가져오려면 스택 작업(Stack actions)을 선택한 다음 리소스를 스택으로 가져오기(Import resources into stack)를 선택합니다.



- 가져오기 개요(Import overview) 페이지에서 이 작업 중에 제공해야 하는 항목 목록을 검토한 후 다음(Next)을 선택합니다.

3. Specify template(템플릿 지정) 페이지에서 다음 방법 중 하나를 사용하여 업데이트된 템플릿을 제 공합니다.
 - Amazon S3 URL을 선택한 다음, 텍스트 상자에 템플릿의 URL을 지정합니다.
 - 템플릿 파일 업로드를 선택한 다음 템플릿을 찾습니다.

그리고 다음을 선택합니다.
4. Identify resources(리소스 식별) 페이지에서 각 대상 리소스를 식별합니다. 자세한 내용은 [리소스 식별자](#) 섹션을 참조하세요.
 - a. Identifier property(식별자 속성)에서 리소스 식별자의 유형을 선택합니다. 예를 들어, TableName 속성은 AWS::DynamoDB::Table 리소스를 식별합니다.
 - b. 식별자 값(Identifier value)에서 실제 속성 값을 입력합니다. 예제 템플릿에서 GamesTable 리 소스의 TableName은 Games입니다.
 - c. 다음을 선택합니다.
5. 스택 세부 정보 지정(Specify stack details) 페이지를 검토하고 다음(Next)을 선택합니다.
6. 가져오기 개요(Import overview) 페이지에서 가져올 리소스를 검토한 다음 리소스 가져오기 (Import resources)를 선택합니다. 그러면 AWS::DynamoDB::Table 리소스 유형을 다시 스택으 로 가져오게 됩니다.

결과: 이 예제에서는 서비스를 중단하지 않고 가져오기 작업을 통해 리소스 드리프트를 해결했습니다. CloudFormation 콘솔의 [이벤트(Events)] 탭에서 가져오기 작업의 진행 상황을 확인할 수 있습니다. 가 젤온 리소스가 IMPORT_COMPLETE 상태가 된 다음 CREATE_COMPLETE 상태가 되고 상태 이유로 리 소스 가져오기 완료(Resource import complete)가 표시됩니다.

CloudFormation이 스택 업데이트 작업을 완료할 때까지 기다립니다. 스택 업데이트 작업이 완료되면 리소스의 실제 드리프트된 상태와 일치하도록 템플릿을 업데이트합니다. 예를 들어 BillingMode는 PAY_PER_REQUEST 및 ReadCapacityUnits로 설정되고 WriteCapacityUnits는 0으로 설정됩 니다.

Example JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Import test",
  "Resources": {
    "ServiceTable":{
```

```
    "Type": "AWS::DynamoDB::Table",
    "Properties": {
      "TableName": "Service",
      "AttributeDefinitions": [
        {
          "AttributeName": "key",
          "AttributeType": "S"
        }
      ],
      "KeySchema": [
        {
          "AttributeName": "key",
          "KeyType": "HASH"
        }
      ],
      "BillingMode": "PROVISIONED",
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 5,
        "WriteCapacityUnits": 1
      }
    }
  },
  "GamesTable": {
    "Type": "AWS::DynamoDB::Table",
    "DeletionPolicy": "Retain",
    "Properties": {
      "TableName": "Games",
      "AttributeDefinitions": [
        {
          "AttributeName": "key",
          "AttributeType": "S"
        }
      ],
      "KeySchema": [
        {
          "AttributeName": "key",
          "KeyType": "HASH"
        }
      ],
      "BillingMode": "PAY_PER_REQUEST",
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 0,
        "WriteCapacityUnits": 0
      }
    }
  }
}
```

```

    }
  }
}

```

Example YAML

```

AWSTemplateFormatVersion: 2010-09-09
Description: Import test
Resources:
  ServiceTable:
    Type: AWS::DynamoDB::Table
    Properties:
      TableName: Service
      AttributeDefinitions:
        - AttributeName: key
          AttributeType: S
      KeySchema:
        - AttributeName: key
          KeyType: HASH
      BillingMode: PROVISIONED
      ProvisionedThroughput:
        ReadCapacityUnits: 5
        WriteCapacityUnits: 1
  GamesTable:
    Type: AWS::DynamoDB::Table
    DeletionPolicy: Retain
    Properties:
      TableName: Games
      AttributeDefinitions:
        - AttributeName: key
          AttributeType: S
      KeySchema:
        - AttributeName: key
          KeyType: HASH
      BillingMode: PAY_PER_REQUEST
      ProvisionedThroughput:
        ReadCapacityUnits: 0
        WriteCapacityUnits: 0

```

CloudFormation 스택에 AWS 리소스 가져오기

CloudFormation 스택에 기존 리소스를 가져올 수 있습니다. 이 기능은 CloudFormation 외부에서 생성된 리소스를 삭제하고 다시 생성할 필요 없이 CloudFormation을 사용하여 해당 리소스를 관리하려는 경우에 매우 유용합니다.

CloudFormation에서는 기존 리소스를 스택에 가져오기 위해 다음 옵션을 제공합니다.

- [IaC 생성기](#)는 기존 리소스를 자동으로 스캔하고 현재 상태를 기반으로 CloudFormation 템플릿을 생성하는 도구입니다. 그런 다음에 이 템플릿을 사용하여 해당 리소스를 스택으로 가져올 수 있습니다.
- [리소스 가져오기](#)는 CloudFormation 템플릿의 기존 리소스를 설명한 다음에 스택으로 가져오는 수동 프로세스입니다. 이 접근 방식을 사용하려면 템플릿의 리소스 속성과 구성을 수동으로 지정해야 합니다.
- [자동 가져오기](#)는 사용자가 CloudFormation 템플릿의 기존 리소스를 설명하고 CloudFormation이 일치하는 사용자 지정 이름의 리소스를 스택에 가져오는 자동 프로세스입니다.
- [스택 리팩터링](#)은 기존 리소스 속성과 데이터를 유지하면서 CloudFormation 스택에서 리소스 재구성을 간소화하는 기능입니다. 스택 리팩터링을 사용하면 스택 사이에서 리소스를 이동하거나 모놀리식 스택을 더 작은 구성 요소로 분할하거나 여러 스택을 하나로 통합할 수 있습니다.

리소스 가져오기 특성은 CloudFormation에서 관리하도록 기존 리소스를 가져오는 외에 다음과 같은 시나리오에서 유용할 수 있습니다.

- 스택 간 리소스 이동 – 필요한 대로 인프라를 재구성할 수 있도록 하나의 스택에서 다른 스택으로 리소스를 가져올 수 있습니다.
- 기존 스택 중첩 – 재사용 가능한 모듈식 인프라를 설계할 수 있도록 기존 스택을 다른 스택 내에 중첩되는 스택으로 가져올 수 있습니다.

CloudFormation에서는 다양한 리소스의 가져오기를 지원합니다. 자세한 내용은 [리소스 유형 지원](#) 섹션을 참조하세요.

주제

- [수동으로 CloudFormation 스택에 AWS 리소스 가져오기](#)
- [자동으로 CloudFormation 스택에 AWS 리소스 가져오기](#)
- [가져오기 작업 되돌리기](#)

수동으로 CloudFormation 스택에 AWS 리소스 가져오기

리소스 가져오기를 사용하면 기존 AWS 리소스를 신규 또는 기존 CloudFormation 스택에 가져올 수 있습니다. 가져오기 작업 중에 기존 리소스를 스택으로 가져오거나 기존 리소스에서 새 스택을 생성하는 변경 세트를 생성합니다. 가져오기 작업 동안 다음을 제공합니다.

- 원본 스택 리소스와 가져오기 중인 리소스를 모두 포함하여 전체 스택을 설명하는 템플릿. 가져올 각 리소스에는 [DeletionPolicy 속성](#)이 있어야 합니다.
- CloudFormation이 템플릿의 논리적 ID를 기존 리소스와 매핑하는 데 사용할 수 있도록 가져오는 리소스의 식별자.

Note

CloudFormation에서는 리소스 가져오기를 사용하여 한 수준의 중첩만 지원합니다. 즉, 스택을 하위 스택으로 가져오거나 하위 요소가 있는 스택을 가져올 수 없습니다.

주제

- [리소스 식별자](#)
- [리소스 가져오기 확인](#)
- [리소스 가져오기 상태 코드](#)
- [가져오기 작업 중 고려 사항](#)
- [추가적인 리소스](#)
- [기존 리소스에서 스택 생성](#)
- [기존 리소스를 스택으로 가져오기](#)
- [스택 간 리소스 이동](#)
- [기존 스택 중첩](#)

리소스 식별자

가져오는 각 리소스를 식별하기 위해 두 개의 값을 제공합니다.

- 식별자 속성. 각 리소스 유형을 식별하는 데 사용할 수 있는 리소스 속성입니다. 예를 들어, `AWS::S3::Bucket` 리소스는 `BucketName`를 사용해 식별할 수 있습니다.

가져오는 리소스를 식별하는 데 사용하는 리소스 속성은 리소스 유형에 따라 다릅니다.

CloudFormation 콘솔에서 리소스 속성을 찾을 수 있습니다. 가져올 리소스가 포함된 템플릿을 만든 후에는 가져오기 프로세스를 시작하고, 여기에서 가져오려는 리소스의 식별자 속성을 찾을 수 있습니다. 일부 리소스 유형의 경우 여러 가지 방법으로 식별할 수 있으며, 드롭다운 목록에서 사용할 속성을 선택할 수 있습니다.

아니면 [get-template-summary](#) CLI 명령을 호출하고 스택 템플릿의 S3 URL을 `--template-url` 옵션 값으로 지정하여 가져오려는 리소스의 식별자 속성을 가져올 수 있습니다.

- 식별자 값. 리소스의 실제 속성값입니다. 예를 들어 BucketName 속성의 실제 값은 MyS3Bucket일 수 있습니다.

서비스 콘솔에서 리소스의 식별자 속성값을 가져올 수 있습니다.

리소스 가져오기 확인

가져오기 작업 중에 CloudFormation은 다음과 같은 유효성 검사를 수행합니다.

- 가져오기를 수행할 자원이 존재합니다.
- 가져오기를 수행할 각 리소스의 속성 및 구성 값은 허용되는 속성, 필수 속성 및 지원되는 속성값을 정의하는 리소스 유형 스키마를 따릅니다.
- 필수 속성은 템플릿에서 지정됩니다. 각 리소스 유형에 대한 필수 속성은 [AWS 리소스 및 속성 참조](#)에 설명되어 있습니다.
- 가져오기를 수행할 리소스는 동일한 리전의 다른 스택에 속하지 않습니다.

CloudFormation은 템플릿 구성이 리소스 속성의 실제 구성과 일치하는지 확인하지 않습니다.

Important

템플릿에 정의된 리소스 및 해당 속성이 예기치 않은 변경을 방지하기 위해 리소스 가져오기의 의도된 구성과 일치하는지 확인합니다.

리소스 가져오기 상태 코드

이 표에서는 리소스 가져오기 특성에서 사용되는 다양한 상태 유형에 대해 설명합니다.

가져오기 작업 상태	설명
IMPORT_IN_PROGRESS	가져오기 작업이 진행 중입니다.
IMPORT_COMPLETE	를 스택의 모든 리소스에 대해 가져오기 작업이 완료되었습니다.
IMPORT_ROLLBACK_IN_PROGRESS	가져오기 롤백 작업이 이전 템플릿 구성을 롤백하는 중입니다.
IMPORT_ROLLBACK_FAILED	가져오기 롤백 작업이 실패했습니다.
IMPORT_ROLLBACK_COMPLETE	가져오기는 이전 템플릿 구성으로 롤백됩니다.

가져오기 작업 중 고려 사항

- 가져오기가 완료된 후 후속 스택 작업을 수행하기 전에 가져오기를 수행한 리소스에 대해 드리프트 감지를 실행하는 것이 좋습니다. 드리프트 감지는 템플릿 구성이 실제 구성과 일치하는지 확인합니다. 자세한 내용은 [전체 CloudFormation 스택의 드리프트 감지](#) 섹션을 참조하세요.
- 가져오기 작업에서는 새 리소스 생성, 리소스 삭제 또는 속성 구성 변경을 허용하지 않습니다.
- 가져오기를 수행할 각 리소스에서 가져오기 작업이 성공하려면 DeletionPolicy 속성이 있어야 합니다. DeletionPolicy는 가능한 어떤 값으로든 설정할 수 있습니다. 가져오는 리소스에만 DeletionPolicy가 필요합니다. 이미 스택의 일부인 리소스에서는 DeletionPolicy가 필요하지 않습니다.
- 동일한 리소스를 여러 스택으로 가져올 수 없습니다.
- cloudformation:ImportResourceTypes IAM 정책 조건을 사용하여 가져오기 작업 중에 사용자가 작업할 수 있는 리소스 유형을 제어할 수 있습니다. 자세한 내용은 [CloudFormation에 대한 정책 조건 키](#) 섹션을 참조하세요.
- CloudFormation 스택 한도는 리소스를 가져올 때 적용됩니다. 제한에 대한 자세한 내용은 [CloudFormation 할당량 이해](#) 섹션을 참조하세요.

추가적인 리소스

리소스 가져오기로 스택 드리프트를 해결하는 방법은 [가져오기 작업을 통한 드리프트 해결](#) 섹션을 참조하세요.

기존 리소스에서 스택 생성

이 주제에서는 템플릿에서 기존 AWS 리소스를 설명하여 해당 리소스에서 스택을 생성하는 방법을 보여줍니다. 이렇게 하는 대신, 기존 리소스를 스캔하고 해당 리소스를 CloudFormation으로 가져오거나 새 계정에 해당 리소스를 복제하는 데 사용할 수 있는 템플릿을 자동으로 생성하려면 [IaC 생성기를 사용하여 기존 리소스에서 템플릿 생성](#) 섹션을 참조하세요.

사전 조건

시작하기 전에 다음을 완비해야 합니다.

- 새 스택에 필요한 모든 리소스를 설명하는 템플릿. 템플릿을 로컬 또는 Amazon S3 버킷에 저장합니다.
- 가져오려는 각 리소스에 대해 다음을 포함하세요.
 - 리소스의 현재 구성을 정의하는 속성 및 속성값
 - 리소스의 고유 식별자(예: 리소스 이름) 자세한 내용은 [리소스 식별자](#) 섹션을 참조하세요.
 - [DeletionPolicy](#) 속성

주제

- [예제 템플릿](#)
- [AWS Management Console를 사용하여 기존 리소스에서 스택 생성](#)
- [AWS CLI를 사용하여 기존 리소스에서 스택 생성](#)

예제 템플릿

이 연습에서는 다음과 같이 CloudFormation 외부에서 생성된 두 개의 DynamoDB 테이블을 지정하는 TemplateToImport.json이라는 예제 템플릿을 사용한다고 가정합니다. ServiceTable 및 GameTable은 가져오기의 대상입니다.

Note

이 템플릿은 예로만 사용됩니다. 이 템플릿을 자체 테스트 목적으로 사용하려면 샘플 리소스를 계정의 리소스로 대체하세요.

```
{  
  "AWSTemplateFormatVersion": "2010-09-09",
```

```
"Description": "Import test",
"Resources": {
  "ServiceTable": {
    "Type": "AWS::DynamoDB::Table",
    "DeletionPolicy": "Retain",
    "Properties": {
      "TableName": "Service",
      "AttributeDefinitions": [
        {
          "AttributeName": "key",
          "AttributeType": "S"
        }
      ],
      "KeySchema": [
        {
          "AttributeName": "key",
          "KeyType": "HASH"
        }
      ],
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 5,
        "WriteCapacityUnits": 1
      }
    }
  },
  "GamesTable": {
    "Type": "AWS::DynamoDB::Table",
    "DeletionPolicy": "Retain",
    "Properties": {
      "TableName": "Games",
      "AttributeDefinitions": [
        {
          "AttributeName": "key",
          "AttributeType": "S"
        }
      ],
      "KeySchema": [
        {
          "AttributeName": "key",
          "KeyType": "HASH"
        }
      ],
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 5,
```

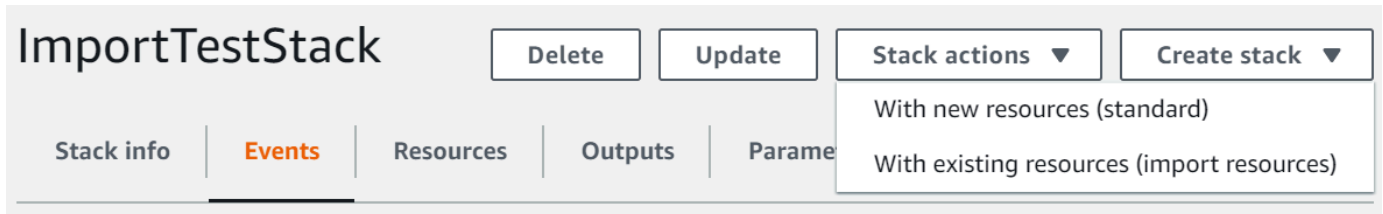
```

        "WriteCapacityUnits": 1
    }
}
}
}
}

```

AWS Management Console를 사용하여 기존 리소스에서 스택 생성

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 스택 페이지에서 스택 생성을 선택한 다음 With existing resources(import resources)(기존 리소스 사용(리소스 가져오기))를 선택합니다.



3. 이 작업 중에 제공해야 하는 항목 목록은 Import overview(가져오기 개요) 페이지를 참조하십시오. 그리고 다음을 선택합니다.
4. Specify template(템플릿 지정) 페이지에서 다음 방법 중 하나를 사용하여 템플릿을 제공하고 다음을 선택합니다.
 - Amazon S3 URL을 선택한 다음, 텍스트 상자에 템플릿의 URL을 지정합니다.
 - 템플릿 파일 업로드를 선택한 다음 템플릿을 찾습니다.
5. Identify resources(리소스 식별) 페이지에서 각 대상 리소스를 식별합니다. 자세한 내용은 [리소스 식별자](#) 섹션을 참조하세요.
 - a. Identifier property(식별자 속성)에서 리소스 식별자의 유형을 선택합니다. 예를 들어 TableName 속성을 사용하여 AWS::DynamoDB::Table 리소스를 식별할 수 있습니다.
 - b. Identifier value(식별자 값)에서 실제 속성값을 입력합니다. 예를 들어 예제 템플릿의 GamesTable 리소스에 대한 TableName는 *Games*입니다.
 - c. 다음을 선택합니다.
6. Specify stack details(스택 세부 정보 지정) 페이지에서 어떤 파라미터든 수정하고 다음을 선택합니다. 이렇게 하면 변경 세트가 자동으로 생성됩니다.

⚠ Important

생성, 업데이트 또는 삭제 작업을 시작하는 기존 파라미터를 수정하면 가져오기 작업이 실패합니다.

7. **stack-name** 검토 페이지에서 올바른 리소스를 가져오고 있는지 확인한 다음, Import resources(리소스 가져오기)를 선택합니다. 이렇게 하면 마지막 단계에서 생성된 변경 세트가 자동으로 실행됩니다.

새로운 스택에 대한 Stack details(스택 세부 정보) 페이지의 Events(이벤트) 창이 표시됩니다.

Timestamp	Logical ID	Status	Status reason
2019-10-24 15:52:49 UTC-0700	ImportTestStack	IMPORT_COMPLETE	-
2019-10-24 15:52:47 UTC-0700	SampleIamUser	UPDATE_COMPLETE	-
2019-10-24 15:52:47 UTC-0700	SampleIamUser	UPDATE_IN_PROGRESS	Apply stack-level tags to imported resource if applicable.
2019-10-24 15:52:46 UTC-0700	SampleIamUser	IMPORT_COMPLETE	Resource import completed.
2019-10-24 15:52:45 UTC-0700	SampleIamUser	IMPORT_IN_PROGRESS	Resource import started.
2019-10-24 15:52:43 UTC-0700	ImportTestStack	IMPORT_IN_PROGRESS	User Initiated
2019-10-24 15:52:01 UTC-0700	ImportTestStack	REVIEW_IN_PROGRESS	User Initiated

8. (선택 사항) 스택에서 드리프트 감지를 실행하여 가져온 리소스의 템플릿과 실제 구성이 일치하는지 확인합니다. 드리프트 감지에 대한 자세한 내용은 [전체 CloudFormation 스택의 드리프트 감지](#) 섹션을 참조하세요.
9. (선택 사항) 가져온 리소스가 예상 템플릿 구성과 일치하지 않으면 템플릿 구성을 수정하거나 리소스를 직접 업데이트합니다. 이 연습에서는 템플릿 구성을 실제 구성과 일치하도록 수정합니다.
 - a. 영향을 받는 리소스에 대해 [가져오기 작업을 되돌립니다](#).
 - b. 템플릿 구성이 실제 구성과 일치하는지 확인하여 템플릿에 가져오기 대상을 다시 추가합니다.
 - c. 수정된 템플릿을 사용해 2~8단계를 반복하여 리소스를 다시 가져옵니다.

AWS CLI를 사용하여 기존 리소스에서 스택 생성

1. 템플릿의 각 리소스 유형을 식별하는 속성을 알아보고, 템플릿의 S3 URL을 지정하여 `get-template-summary` 명령을 실행합니다. 예를 들어 `TableName` 속성을 사용하여 `AWS::DynamoDB::Table` 리소스를 식별할 수 있습니다. 예제 템플릿의 `GamesTable` 리소스의 `TableName`의 값은 `Games`입니다. 다음 단계에는 이 정보가 필요합니다.

```
aws cloudformation get-template-summary \
  --template-url https://amzn-s3-demo-bucket.s3.us-west-2.amazonaws.com/
  TemplateToImport.json
```

자세한 내용은 [리소스 식별자](#) 섹션을 참조하세요.

2. 템플릿의 실제 리소스 목록과 고유한 식별자를 다음 JSON 문자열 형식으로 작성합니다.

```
[{"ResourceType":"AWS::DynamoDB::Table","LogicalResourceId":"GamesTable","ResourceIdentifier":{
  "TableName":"Games"}},
{"ResourceType":"AWS::DynamoDB::Table","LogicalResourceId":"ServiceTable","ResourceIdentifier":{
  "TableName":"Service"}}]
```

또는 구성 파일에서 JSON 형식 파라미터를 지정할 수 있습니다.

예를 들어 `ServiceTable` 및 `GamesTable`을 가져오려면 다음 구성이 포함된 `ResourcesToImport.txt` 파일을 만들 수 있습니다.

```
[
  {
    "ResourceType":"AWS::DynamoDB::Table",
    "LogicalResourceId":"GamesTable",
    "ResourceIdentifier":{
      "TableName":"Games"
    }
  },
  {
    "ResourceType":"AWS::DynamoDB::Table",
    "LogicalResourceId":"ServiceTable",
    "ResourceIdentifier":{
      "TableName":"Service"
    }
  }
]
```

3. 변경 세트를 생성하려면 다음 `create-change-set` 명령을 사용하고 자리 표시자 텍스트를 변경합니다. `--change-set-type` 옵션의 경우, **IMPORT**의 값을 지정합니다. `--resources-to-import` 옵션의 경우 샘플 JSON 문자열을 방금 생성한 실제 JSON 문자열로 바꿉니다.

```
aws cloudformation create-change-set \
  --stack-name TargetStack --change-set-name ImportChangeSet \
  --change-set-type IMPORT \
  --template-url https://amzn-s3-demo-bucket.s3.us-west-2.amazonaws.com/
TemplateToImport.json \
  --resources-to-
import '[{"ResourceType":"AWS::DynamoDB::Table","LogicalResourceId":"GamesTable","ResourceIdentifi
{"TableName":"Games"}},
{"ResourceType":"AWS::DynamoDB::Table","LogicalResourceId":"ServiceTable","ResourceIdentifi
{"TableName":"Service"}}]'
```

Note

`--resources-to-import`에서는 인라인 YAML을 지원하지 않습니다. JSON 문자열의 인용 부호 요구 사항은 터미널에 따라 다릅니다. 자세한 내용은 AWS Command Line Interface 사용 설명서의 [문자열 안에 따옴표 사용](#)을 참조하세요.

또는 다음 예제와 같이 `--resources-to-import` 옵션에 대한 입력으로 파일 URL을 사용할 수 있습니다.

```
--resources-to-import file:///ResourcesToImport.txt
```

4. 변경 세트를 검토하여 올바른 리소스를 가져올 것인지 확인합니다.

```
aws cloudformation describe-change-set \
  --change-set-name ImportChangeSet --stack-name TargetStack
```

5. 변경 세트를 시작하고 리소스를 가져오려면 다음 `execute-change-set` 명령을 사용하고 자리 표시자 텍스트를 변경합니다. 작업 (IMPORT_COMPLETE)이 성공적으로 완료되면 리소스가 성공적으로 가져오기 됩니다.

```
aws cloudformation execute-change-set \
  --change-set-name ImportChangeSet --stack-name TargetStack
```

6. (선택 사항) `IMPORT_COMPLETE` 스택에서 드리프트 감지를 실행하여 가져온 리소스의 템플릿과 실제 구성이 일치하는지 확인합니다. 드리프트 감지의 자세한 내용은 [개별 스택 리소스의 드리프트 감지](#) 섹션을 참조하세요.

- a. 지정된 스택에서 드리프트 감지를 실행합니다.

```
aws cloudformation detect-stack-drift --stack-name TargetStack
```

성공하면 이 명령에서 다음 샘플 출력이 반환됩니다.

```
{ "Stack-Drift-Detection-Id" : "624af370-311a-11e8-b6b7-500cexample" }
```

- b. 지정된 스택 드리프트 감지 ID에 대한 드리프트 감지 작업의 진행 상황을 확인합니다.

```
aws cloudformation describe-stack-drift-detection-status \  
--stack-drift-detection-id 624af370-311a-11e8-b6b7-500cexample
```

- c. 지정된 스택에서 드리프트 여부가 확인된 리소스에 대한 드리프트 정보를 확인합니다.

```
aws cloudformation describe-stack-resource-drifts --stack-name TargetStack
```

7. (선택 사항) 가져온 리소스가 예상 템플릿 구성과 일치하지 않으면 템플릿 구성을 수정하거나 리소스를 직접 업데이트합니다. 이 연습에서는 템플릿 구성을 실제 구성과 일치하도록 수정합니다.

- a. 영향을 받는 리소스에 대해 [가져오기 작업을 되돌립니다](#).
- b. 템플릿 구성이 실제 구성과 일치하는지 확인하여 템플릿에 가져오기 대상을 다시 추가합니다.
- c. 수정된 템플릿을 사용해 3~6단계를 반복하여 리소스를 다시 가져옵니다.

기존 리소스를 스택으로 가져오기

이 주제에서는 템플릿에서 기존 AWS 리소스를 설명하여 기존 스택으로 해당 리소스를 가져오는 방법을 보여줍니다. 이렇게 하는 대신, 기존 리소스를 스캔하고 해당 리소스를 CloudFormation으로 가져오거나 새 계정에 해당 리소스를 복제하는 데 사용할 수 있는 템플릿을 자동으로 생성하려면 [IaC 생성기를 사용하여 기존 리소스에서 템플릿 생성](#) 섹션을 참조하세요.

사전 조건

시작하기 전에 다음을 완비해야 합니다.

- 스택에 이미 포함되어 있는 리소스와 가져올 리소스를 모두 포함한 전체 스택을 설명하는 템플릿. 템플릿을 로컬 또는 Amazon S3 버킷에 저장합니다.

실행 중인 스택의 템플릿 사본을 가져오려면

1. <https://console.aws.amazon.com/cloudformation/>에서 CloudFormation 콘솔을 엽니다.
 2. 스택 목록에서 템플릿을 검색할 스택을 선택합니다.
 3. 스택 세부 정보 창에서 템플릿 탭을 선택한 다음 클립보드에 복사를 선택합니다.
 4. 코드를 텍스트 편집기에 붙여넣어 템플릿에 다른 리소스를 추가하기 시작합니다.
- 가져오려는 각 리소스에 대해 다음을 포함하세요.
 - 리소스의 현재 구성을 정의하는 속성 및 속성값
 - 리소스의 고유 식별자(예: 리소스 이름) 자세한 내용은 [리소스 식별자](#) 섹션을 참조하세요.
 - [DeletionPolicy 속성](#)

주제

- [예제 템플릿](#)
- [AWS Management Console를 사용하여 기존 리소스를 스택으로 가져오기](#)
- [AWS CLI를 사용하여 기존 리소스를 스택으로 가져오기](#)

예제 템플릿

이 연습에서는 다음과 같이 두 개의 DynamoDB 테이블을 지정하는 TemplateToImport.json이라는 예제 템플릿을 사용한다고 가정합니다. ServiceTable은 현재 스택의 일부이며 GamesTable은 가져오려는 테이블입니다.

Note

이 템플릿은 예로만 사용됩니다. 이 템플릿을 자체 테스트 목적으로 사용하려면 샘플 리소스를 계정의 리소스로 대체하세요.

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Import test",
  "Resources": {
    "ServiceTable": {
```



```
    "Type": "AWS::DynamoDB::Table",
    "Properties": {
      "TableName": "Service",
      "AttributeDefinitions": [
        {
          "AttributeName": "key",
          "AttributeType": "S"
        }
      ],
      "KeySchema": [
        {
          "AttributeName": "key",
          "KeyType": "HASH"
        }
      ],
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 5,
        "WriteCapacityUnits": 1
      }
    },
    "GamesTable": {
      "Type": "AWS::DynamoDB::Table",
      "DeletionPolicy": "Retain",
      "Properties": {
        "TableName": "Games",
        "AttributeDefinitions": [
          {
            "AttributeName": "key",
            "AttributeType": "S"
          }
        ],
        "KeySchema": [
          {
            "AttributeName": "key",
            "KeyType": "HASH"
          }
        ],
        "ProvisionedThroughput": {
          "ReadCapacityUnits": 5,
          "WriteCapacityUnits": 1
        }
      }
    }
  }
```

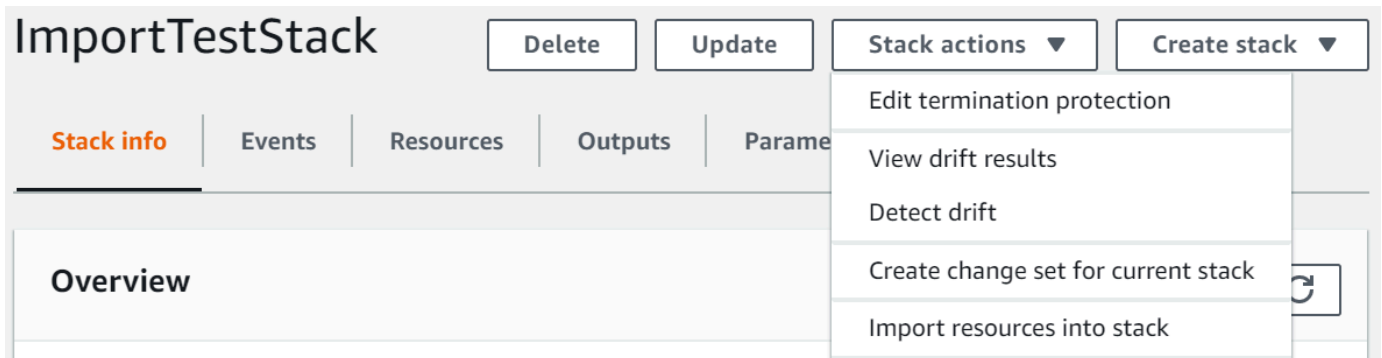
```
}
}
```

AWS Management Console를 사용하여 기존 리소스를 스택으로 가져오기

Note

CloudFormation 콘솔에서는 리소스를 가져올 때 내장 함수 `Fn::Transform` 사용을 지원하지 않습니다. AWS CLI를 사용하여 `Fn::Transform` 함수를 사용하는 리소스를 가져올 수 있습니다.

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 스택 페이지에서 리소스를 가져올 스택을 선택합니다.
3. Stack actions(스택 작업)를 선택한 다음 Import resources into stack(스택으로 리소스 가져오기)을 선택합니다.



4. Import overview(가져오기 개요) 페이지를 검토한 후 다음을 선택합니다.
5. Specify template(템플릿 지정) 페이지에서 다음 방법 중 하나를 사용하여 업데이트된 템플릿을 제공하고 다음을 선택합니다.
 - Amazon S3 URL을 선택한 다음, 텍스트 상자에 템플릿의 URL을 지정합니다.
 - 템플릿 파일 업로드를 선택한 다음 템플릿을 찾습니다.
6. Identify resources(리소스 식별) 페이지에서 각 대상 리소스를 식별합니다. 자세한 내용은 [리소스 식별자](#) 섹션을 참조하세요.
 - a. Identifier property(식별자 속성)에서 리소스 식별자의 유형을 선택합니다. 예를 들어 TableName 속성을 사용하여 `AWS::DynamoDB::Table` 리소스를 식별할 수 있습니다.

- b. Identifier value(식별자 값)에서 실제 속성값을 입력합니다. 예를 들어 예제 템플릿의 GamesTable 리소스에 대한 TableName은 **Games**입니다.
 - c. 다음을 선택합니다.
7. Specify stack details(스택 세부 정보 지정) 페이지에서 어떤 파라미터든 업데이트하고 다음을 선택합니다. 이렇게 하면 변경 세트가 자동으로 생성됩니다.

Note

생성, 업데이트 또는 삭제 작업을 시작하는 기존 파라미터를 수정하면 가져오기 작업이 실패합니다.

8. **stack-name** 검토 페이지에서 가져올 리소스를 검토한 다음, Import resources(리소스 가져오기)를 선택합니다. 이렇게 하면 마지막 단계에서 생성된 변경 세트가 자동으로 실행됩니다. 이때 어떤 스택 레벨 태그든 가져온 리소스에 적용됩니다. 자세한 내용은 [스택 옵션 구성](#) 섹션을 참조하세요.

스택에 대한 이벤트 페이지가 표시됩니다.

ImportTestStack

Delete

Update

Stack actions

Create stack

Stack info

Events

Resources

Outputs

Parameters

Template

Change sets

Events

Q Search events

Timestamp	Logical ID	Status	Status reason
2019-10-24 15:52:49 UTC-0700	ImportTestStack	IMPORT_COMPLETE	-
2019-10-24 15:52:47 UTC-0700	SampleIamUser	UPDATE_COMPLETE	-
2019-10-24 15:52:47 UTC-0700	SampleIamUser	UPDATE_IN_PROGRESS	Apply stack-level tags to imported resource if applicable.
2019-10-24 15:52:46 UTC-0700	SampleIamUser	IMPORT_COMPLETE	Resource import completed.
2019-10-24 15:52:45 UTC-0700	SampleIamUser	IMPORT_IN_PROGRESS	Resource import started.
2019-10-24 15:52:43 UTC-0700	ImportTestStack	IMPORT_IN_PROGRESS	User Initiated
2019-10-24 15:52:01 UTC-0700	ImportTestStack	REVIEW_IN_PROGRESS	User Initiated

9. (선택 사항) 스택에서 드리프트 감지를 실행하여 가져온 리소스의 템플릿과 실제 구성이 일치하는지 확인합니다. 드리프트 감지에 대한 자세한 내용은 [전체 CloudFormation 스택의 드리프트 감지](#) 섹션을 참조하세요.
10. (선택 사항) 가져온 리소스가 예상 템플릿 구성과 일치하지 않으면 템플릿 구성을 수정하거나 리소스를 직접 업데이트합니다. 드리프트된 리소스 가져오기에 대한 자세한 내용은 [가져오기 작업을 통한 드리프트 해결](#) 섹션을 참조하세요.

AWS CLI를 사용하여 기존 리소스를 스택으로 가져오기

1. 템플릿의 각 리소스 유형을 식별하는 속성을 알아보고, 템플릿의 S3 URL을 지정하여 `get-template-summary` 명령을 실행합니다. 예를 들어 `TableName` 속성을 사용하여 `AWS::DynamoDB::Table` 리소스를 식별할 수 있습니다. 예제 템플릿의 `GamesTable` 리소스의 `TableName`의 값은 `Games`입니다. 다음 단계에는 이 정보가 필요합니다.

```
aws cloudformation get-template-summary \  
  --template-url https://amzn-s3-demo-bucket.s3.us-west-2.amazonaws.com/  
TemplateToImport.json
```

자세한 내용은 [리소스 식별자](#) 섹션을 참조하세요.

2. 가져올 실제 리소스 목록과 고유 식별자를 다음 JSON 문자열 형식으로 작성합니다.

```
[{"ResourceType":"AWS::DynamoDB::Table","LogicalResourceId":"GamesTable","ResourceIdentifier":{"TableName":"Games"}}]
```

또는 구성 파일에서 JSON 형식 파라미터를 지정할 수 있습니다.

예를 들어 `GamesTable`을 가져오려면 다음 구성이 포함된 *ResourcesToImport.txt* 파일을 만들 수 있습니다.

```
[  
  {  
    "ResourceType":"AWS::DynamoDB::Table",  
    "LogicalResourceId":"GamesTable",  
    "ResourceIdentifier": {  
      "TableName":"Games"  
    }  
  }  
]
```

3. 변경 세트를 생성하려면 다음 `create-change-set` 명령을 사용하고 자리 표시자 텍스트를 변경합니다. `--change-set-type` 옵션의 경우, **IMPORT**의 값을 지정합니다. `--resources-to-import` 옵션의 경우 샘플 JSON 문자열을 방금 생성한 실제 JSON 문자열로 바꿉니다.

```
aws cloudformation create-change-set \  
  --stack-name TargetStack --change-set-name ImportChangeSet \  
  --change-set-type IMPORT \  
  --resources-to-import [JSON 문자열]
```

```
--template-url https://amzn-s3-demo-bucket.s3.us-west-2.amazonaws.com/
TemplateToImport.json \
--resources-to-
import '["ResourceType":"AWS::DynamoDB::Table","LogicalResourceId":"GamesTable","ResourceI
{"TableName":"Games"}]']
```

Note

--resources-to-import에서는 인라인 YAML을 지원하지 않습니다. JSON 문자열의 인용 부호 요구 사항은 터미널에 따라 다릅니다. 자세한 내용은 AWS Command Line Interface 사용 설명서의 [문자열 안에 따옴표 사용](#)을 참조하세요.

또는 다음 예제와 같이 --resources-to-import 옵션에 대한 입력으로 파일 URL을 사용할 수 있습니다.

```
--resources-to-import file://ResourcesToImport.txt
```

4. 변경 세트를 검토하여 올바른 리소스를 가져올 것인지 확인합니다.

```
aws cloudformation describe-change-set \
--change-set-name ImportChangeSet --stack-name TargetStack
```

5. 변경 세트를 시작하고 리소스를 가져오려면 다음 execute-change-set 명령을 사용하고 자리 표시자 텍스트를 변경합니다. 이때 어떤 스택 레벨 태그든 가져온 리소스에 적용됩니다. 자세한 내용은 [스택 옵션 구성](#) 섹션을 참조하세요. 작업 (IMPORT_COMPLETE)이 성공적으로 완료되면 리소스가 성공적으로 가져오기 됩니다.

```
aws cloudformation execute-change-set \
--change-set-name ImportChangeSet --stack-name TargetStack
```

6. (선택 사항) IMPORT_COMPLETE 스택에서 드리프트 감지를 실행하여 가져온 리소스의 템플릿과 실제 구성이 일치하는지 확인합니다. 드리프트 감지에 대한 자세한 내용은 [전체 CloudFormation 스택의 드리프트 감지](#) 섹션을 참조하세요.
 - a. 지정된 스택에서 드리프트 감지를 실행합니다.

```
aws cloudformation detect-stack-drift --stack-name TargetStack
```

성공하면 이 명령에서 다음 샘플 출력이 반환됩니다.

```
{ "Stack-Drift-Detection-Id" : "624af370-311a-11e8-b6b7-500cexample" }
```

- b. 지정된 스택 드리프트 감지 ID에 대한 드리프트 감지 작업의 진행 상황을 확인합니다.

```
aws cloudformation describe-stack-drift-detection-status \
  --stack-drift-detection-id 624af370-311a-11e8-b6b7-500cexample
```

- c. 지정된 스택에서 드리프트 여부가 확인된 리소스에 대한 드리프트 정보를 확인합니다.

```
aws cloudformation describe-stack-resource-drifts --stack-name TargetStack
```

7. (선택 사항) 가져온 리소스가 예상 템플릿 구성과 일치하지 않으면 템플릿 구성을 수정하거나 리소스를 직접 업데이트합니다. 드리프트된 리소스 가져오기에 대한 자세한 내용은 [가져오기 작업을 통한 드리프트 해결](#) 섹션을 참조하세요.

스택 간 리소스 이동

`resource import` 기능을 사용하여 스택간에 리소스를 이동하거나 스택을 리팩터링할 수 있습니다. 소스 스택에서 제거한 후 대상 스택으로 가져오기를 할 때 리소스가 보존되도록 하려면 먼저 이동시키려는 리소스에 Retain 삭제 정책을 추가해야 합니다.

가져오기를 처음 사용하는 경우 먼저 [CloudFormation 스택에 AWS 리소스 가져오기](#) 주제의 소개 정보를 검토하는 것이 좋습니다.

Important

모든 리소스가 가져오기 작업을 지원하는 것은 아닙니다. 스택에서 리소스를 제거하기 전에 [가져오기 작업을 지원하는 리소스](#)를 참조하세요. 스택에서 가져오기 작업을 지원하지 않는 리소스를 제거하면 리소스를 다른 스택으로 가져오거나 원본 스택으로 다시 가져올 수 없습니다.

AWS Management Console을 사용하여 스택 리팩터링

1. 소스 템플릿에서 이동시킬 리소스에 대한 Retain [DeletionPolicy](#)를 지정합니다.

다음 예제 소스 템플릿에서 Games은 이 리팩터의 대상입니다.

Example JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Import test",
  "Resources": {
    "ServiceTable": {
      "Type": "AWS::DynamoDB::Table",
      "Properties": {
        "TableName": "Service",
        "AttributeDefinitions": [
          {
            "AttributeName": "key",
            "AttributeType": "S"
          }
        ],
        "KeySchema": [
          {
            "AttributeName": "key",
            "KeyType": "HASH"
          }
        ],
        "ProvisionedThroughput": {
          "ReadCapacityUnits": 5,
          "WriteCapacityUnits": 1
        }
      }
    },
    "GamesTable": {
      "Type": "AWS::DynamoDB::Table",
      "DeletionPolicy": "Retain",
      "Properties": {
        "TableName": "Games",
        "AttributeDefinitions": [
          {
            "AttributeName": "key",
            "AttributeType": "S"
          }
        ],
        "KeySchema": [
          {
            "AttributeName": "key",
            "KeyType": "HASH"
          }
        ]
      }
    }
  }
}
```

```

        }
      ],
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 5,
        "WriteCapacityUnits": 1
      }
    }
  }
}

```

2. CloudFormation 콘솔을 열어 스택 업데이트를 수행하여 삭제 정책을 적용합니다.
 - a. 스택(Stacks) 페이지에서 스택을 선택하고 업데이트(Update)를 선택합니다.
 - b. Prepare template(템플릿 준비)에서 Replace current template(현재 템플릿 바꾸기)을 선택합니다.
 - c. Specify template(템플릿 지정)에서 GamesTable에 DeletionPolicy 속성이 설정된 업데이트된 소스 템플릿을 제공한 후, 다음을 선택합니다.
 - Amazon S3 URL을 선택한 다음 텍스트 상자에 업데이트된 소스 템플릿의 URL을 지정합니다.
 - 템플릿 파일 업로드를 선택한 다음 업데이트된 소스 템플릿 파일을 찾습니다.
 - d. Specify stack details(스택 세부 정보 지정) 페이지에서 어떤 변경도 필요하지 않습니다. 다음을 선택합니다.
 - e. Configure stack options(스택 옵션 구성) 페이지에서 어떤 변경도 필요하지 않습니다. 다음을 선택합니다.
 - f. **SourceStackName** 검토 페이지에서 변경 사항을 검토합니다. 템플릿에 IAM 리소스가 포함되어 있는 경우 이 템플릿이 IAM 리소스를 생성할 수 있음을 확인합니다(I acknowledge that this template may create IAM resources)를 선택하여 템플릿에서 IAM 리소스를 사용하도록 지정합니다. 템플릿에서 IAM 리소스를 사용하는 방법에 대한 자세한 내용은 [AWS Identity and Access Management를 사용하여 CloudFormation 액세스 제어](#) 단원을 참조하십시오. 그런 다음 변경 세트를 생성하여 소스 스택을 업데이트하거나 소스 스택을 직접 업데이트하십시오.
3. 소스 템플릿에서 리소스, 관련 파라미터 및 출력을 제거한 다음, 이들을 대상 템플릿에 추가합니다.

이제 소스 템플릿이 다음과 같이 보입니다.

Example JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Import test",
  "Resources": {
    "ServiceTable": {
      "Type": "AWS::DynamoDB::Table",
      "Properties": {
        "TableName": "Service",
        "AttributeDefinitions": [
          {
            "AttributeName": "key",
            "AttributeType": "S"
          }
        ],
        "KeySchema": [
          {
            "AttributeName": "key",
            "KeyType": "HASH"
          }
        ],
        "ProvisionedThroughput": {
          "ReadCapacityUnits": 5,
          "WriteCapacityUnits": 1
        }
      }
    }
  }
}
```

현재 다음과 같은 예제 대상 템플릿에는 PlayersTable 리소스가 있으며 GamesTable도 포함되어 있습니다.

Example JSON

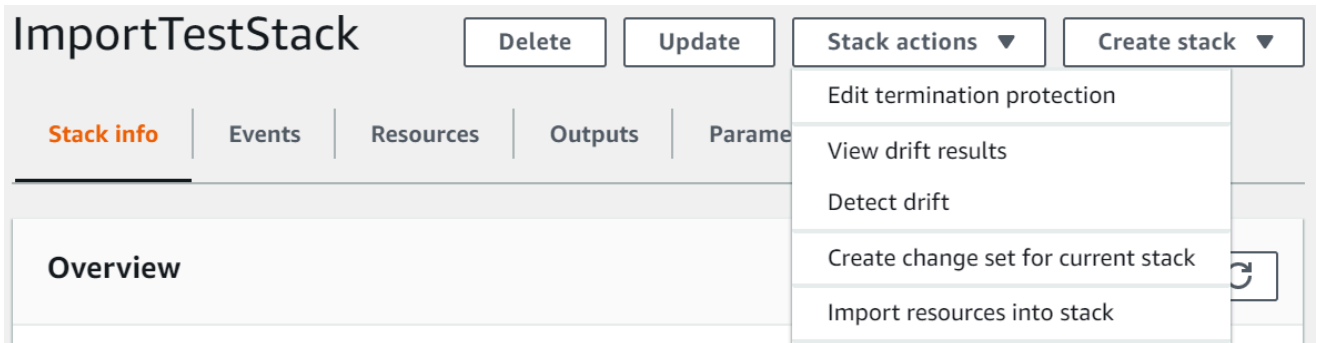
```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Import test",
  "Resources": {
    "PlayersTable": {
      "Type": "AWS::DynamoDB::Table",
```

```
        "Properties": {
            "TableName": "Players",
            "AttributeDefinitions": [
                {
                    "AttributeName": "key",
                    "AttributeType": "S"
                }
            ],
            "KeySchema": [
                {
                    "AttributeName": "key",
                    "KeyType": "HASH"
                }
            ],
            "ProvisionedThroughput": {
                "ReadCapacityUnits": 5,
                "WriteCapacityUnits": 1
            }
        },
        "GamesTable": {
            "Type": "AWS::DynamoDB::Table",
            "DeletionPolicy": "Retain",
            "Properties": {
                "TableName": "Games",
                "AttributeDefinitions": [
                    {
                        "AttributeName": "key",
                        "AttributeType": "S"
                    }
                ],
                "KeySchema": [
                    {
                        "AttributeName": "key",
                        "KeyType": "HASH"
                    }
                ],
                "ProvisionedThroughput": {
                    "ReadCapacityUnits": 5,
                    "WriteCapacityUnits": 1
                }
            }
        }
    }
}
```

}

4. 2~3단계를 반복하여 소스 스택을 다시 업데이트합니다. 이번에는 대상 리소스를 스택에서 삭제합니다.
5. 가져오기 작업을 수행하여 대상 스택에 GamesTable을 추가합니다.

- a. 스택 페이지에서 상위 스택을 선택한 상태에서 Stack actions(스택 작업)를 선택한 다음 Import resources into stack(스택으로 리소스 가져오기)을 선택합니다.



- b. 이 작업 중에 제공해야 하는 항목 목록은 Import overview(가져오기 개요) 페이지를 참조하십시오. 그리고 다음을 선택합니다.
- c. Specify template(템플릿 지정) 페이지에서 다음 중 하나를 완료한 후 Next(다음)를 선택합니다.
 - Amazon S3 URL을 선택한 다음, 텍스트 상자에 URL을 지정합니다.
 - 템플릿 파일 업로드를 선택한 다음 업로드할 파일을 찾습니다.
- d. Identify resources(리소스 식별) 페이지에서 이동 중인 리소스를 식별합니다(이 예제에서는 GamesTable). 자세한 내용은 [리소스 식별자](#) 섹션을 참조하세요.
 1. Identifier property(식별자 속성)에서 리소스 식별자의 유형을 선택합니다. 예를 들어 AWS::DynamoDB::Table 리소스는 TableName 속성을 사용하여 식별할 수 있습니다.
 2. Identifier value(식별자 값)에서 실제 속성값을 입력합니다. 예를 들어 **GamesTables**입니다.
 3. 다음을 선택합니다.
- e. Specify stack details(스택 세부 정보 지정) 페이지에서 어떤 파라미터든 수정하고 다음을 선택합니다. 이렇게 하면 변경 세트가 자동으로 생성됩니다.

⚠ Important

생성, 업데이트 또는 삭제 작업을 시작하는 기존 파라미터를 수정하면 가져오기 작업이 실패합니다.

- f. **TargetStackName** 검토 페이지에서 올바른 리소스를 가져오고 있는지 확인한 다음 리소스 가져오기를 선택합니다. 그러면 마지막 단계에서 생성된 변경 세트가 자동으로 시작됩니다. 이때 어떤 스택 레벨 태그든 가져온 리소스에 적용됩니다.
- g. 상위 스택에 대한 Stack details(스택 세부 정보) 페이지의 Events(이벤트) 창이 표시됩니다.

ImportTestStack					Delete	Update	Stack actions ▼	Create stack ▼
Stack info Events Resources Outputs Parameters Template Change sets								
Events								
Search events								
Timestamp	Logical ID	Status	Status reason					
2019-10-24 15:52:49 UTC-0700	ImportTestStack	✔ IMPORT_COMPLETE	-					
2019-10-24 15:52:47 UTC-0700	SampleIamUser	✔ UPDATE_COMPLETE	-					
2019-10-24 15:52:47 UTC-0700	SampleIamUser	⌚ UPDATE_IN_PROGRESS	Apply stack-level tags to imported resource if applicable.					
2019-10-24 15:52:46 UTC-0700	SampleIamUser	✔ IMPORT_COMPLETE	Resource import completed.					
2019-10-24 15:52:45 UTC-0700	SampleIamUser	⌚ IMPORT_IN_PROGRESS	Resource import started.					
2019-10-24 15:52:43 UTC-0700	ImportTestStack	⌚ IMPORT_IN_PROGRESS	User Initiated					
2019-10-24 15:52:01 UTC-0700	ImportTestStack	⌚ REVIEW_IN_PROGRESS	User Initiated					

📌 Note

AWS::CloudFormation::Stack 리소스가 이미 CloudFormation에서 관리되기 때문에 이러한 가져오기 작업을 수행한 후에 상위 스택에서 드리프트 감지를 실행할 필요가 없습니다.

AWS CLI를 사용하여 스택 리팩터링

1. 소스 템플릿에서 이동시킬 리소스에 대한 Retain DeletionPolicy를 지정합니다.

다음 예제 소스 템플릿에서 GamesTable은 이 리팩터의 대상입니다.

Example JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
```

```
"Description": "Import test",
"Resources": {
  "ServiceTable":{
    "Type":"AWS::DynamoDB::Table",
    "Properties":{
      "TableName":"Service",
      "AttributeDefinitions":[
        {
          "AttributeName":"key",
          "AttributeType":"S"
        }
      ],
      "KeySchema":[
        {
          "AttributeName":"key",
          "KeyType":"HASH"
        }
      ],
      "ProvisionedThroughput":{
        "ReadCapacityUnits":5,
        "WriteCapacityUnits":1
      }
    }
  },
  "GamesTable": {
    "Type": "AWS::DynamoDB::Table",
    "DeletionPolicy": "Retain",
    "Properties": {
      "TableName": "Games",
      "AttributeDefinitions": [
        {
          "AttributeName": "key",
          "AttributeType": "S"
        }
      ],
      "KeySchema": [
        {
          "AttributeName": "key",
          "KeyType": "HASH"
        }
      ],
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 5,
        "WriteCapacityUnits": 1
      }
    }
  }
}
```

```

    }
  }
}
}
}

```

2. 소스 스택을 업데이트하여 삭제 정책을 리소스에 적용합니다.

```
aws cloudformation update-stack --stack-name SourceStackName
```

3. 소스 템플릿에서 리소스, 관련 파라미터 및 출력을 제거한 다음, 이들을 대상 템플릿에 추가합니다.

이제 소스 템플릿이 다음과 같이 보입니다.

Example JSON

```

{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Import test",
  "Resources": {
    "ServiceTable": {
      "Type": "AWS::DynamoDB::Table",
      "Properties": {
        "TableName": "Service",
        "AttributeDefinitions": [
          {
            "AttributeName": "key",
            "AttributeType": "S"
          }
        ],
        "KeySchema": [
          {
            "AttributeName": "key",
            "KeyType": "HASH"
          }
        ],
        "ProvisionedThroughput": {
          "ReadCapacityUnits": 5,
          "WriteCapacityUnits": 1
        }
      }
    }
  }
}

```

```
}  
}
```

현재 다음과 같은 예제 대상 템플릿에는 PlayersTable 리소스가 있으며 GamesTable도 포함되어 있습니다.

Example JSON

```
{  
  "AWSTemplateFormatVersion": "2010-09-09",  
  "Description": "Import test",  
  "Resources": {  
    "PlayersTable": {  
      "Type": "AWS::DynamoDB::Table",  
      "Properties": {  
        "TableName": "Players",  
        "AttributeDefinitions": [  
          {  
            "AttributeName": "key",  
            "AttributeType": "S"  
          }  
        ],  
        "KeySchema": [  
          {  
            "AttributeName": "key",  
            "KeyType": "HASH"  
          }  
        ],  
        "ProvisionedThroughput": {  
          "ReadCapacityUnits": 5,  
          "WriteCapacityUnits": 1  
        }  
      }  
    },  
    "GamesTable": {  
      "Type": "AWS::DynamoDB::Table",  
      "DeletionPolicy": "Retain",  
      "Properties": {  
        "TableName": "Games",  
        "AttributeDefinitions": [  
          {  
            "AttributeName": "key",  
            "AttributeType": "S"  
          }  
        ]  
      }  
    }  
  }  
}
```

```

        }
      ],
      "KeySchema": [
        {
          "AttributeName": "key",
          "KeyType": "HASH"
        }
      ],
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 5,
        "WriteCapacityUnits": 1
      }
    }
  }
}

```

4. 소스 스택을 업데이트하여 GamesTable 리소스 및 관련 파라미터와 출력을 스택에서 삭제합니다.

```
aws cloudformation update-stack --stack-name SourceStackName
```

5. 가져올 실제 리소스 목록과 고유 식별자를 다음 JSON 문자열 형식으로 작성합니다. 자세한 내용은 [리소스 식별자](#) 섹션을 참조하세요.

```
[{"ResourceType":"AWS::DynamoDB::Table","LogicalResourceId":"GamesTable","ResourceIdentifier":
{"TableName":"Games"}]}
```

또는 구성 파일에서 JSON 형식 파라미터를 지정할 수 있습니다.

예를 들어 GamesTable을 가져오려면 다음 구성이 포함된 *ResourcesToImport.txt* 파일을 만들 수 있습니다.

```
[
  {
    "ResourceType":"AWS::DynamoDB::Table",
    "LogicalResourceId":"GamesTable",
    "ResourceIdentifier": {
      "TableName":"Games"
    }
  }
]
```


6. 변경 세트를 생성하려면 다음 `create-change-set` 명령을 사용하고 자리 표시자 텍스트를 변경합니다. `--change-set-type` 옵션의 경우, **IMPORT**의 값을 지정합니다. `--resources-to-import` 옵션의 경우 샘플 JSON 문자열을 방금 생성한 실제 JSON 문자열로 바꿉니다.

```
aws cloudformation create-change-set \
  --stack-name TargetStackName --change-set-name ImportChangeSet \
  --change-set-type IMPORT \
  --template-body file://TemplateToImport.json \
  --resources-to-import
  "'[{"ResourceType": "AWS::DynamoDB::Table", "LogicalResourceId": "GamesTable", "ResourceIdentifiers": [{"TableName": "Games"}]}]'"
```

Note

`--resources-to-import`에서는 인라인 YAML을 지원하지 않습니다. JSON 문자열의 인용 부호 요구 사항은 터미널에 따라 다릅니다. 자세한 내용은 AWS Command Line Interface 사용 설명서의 [문자열 안에 따옴표 사용](#)을 참조하세요.

또는 다음 예제와 같이 `--resources-to-import` 옵션에 대한 입력으로 파일 URL을 사용할 수 있습니다.

```
--resources-to-import file://ResourcesToImport.txt
```

7. 변경 세트를 검토하여 올바른 리소스를 대상 스택에 가져오고 있는지 확인합니다.

```
aws cloudformation describe-change-set \
  --change-set-name ImportChangeSet
```

8. 변경 세트를 시작하고 리소스를 가져오려면 다음 `execute-change-set` 명령을 사용하고 자리 표시자 텍스트를 변경합니다. 이때 어떤 스택 레벨 태그든 가져온 리소스에 적용됩니다. 작업 (IMPORT_COMPLETE)가 성공적으로 완료되면 리소스가 성공적으로 가져오기 됩니다.

```
aws cloudformation execute-change-set \
  --change-set-name ImportChangeSet --stack-name TargetStackName
```

Note

리소스가 이미 CloudFormation에서 관리되기 때문에 이러한 가져오기 작업을 수행한 후에 대상 스택에서 드리프트 감지를 실행할 필요가 없습니다.

기존 스택 중첩

resource import 기능을 사용하여 기존 스택을 다른 기존 스택에 중첩합니다. 중첩된 스택은 다른 템플릿 내에서 선언하고 참조하는 공통 구성 요소입니다. 따라서 동일한 구성을 복사하여 템플릿에 붙여 넣을 필요가 없고 스택 업데이트를 간소화할 수 있습니다. 공통 구성 요소에 대한 템플릿이 있는 경우 `AWS::CloudFormation::Stack` 리소스를 사용하여 다른 템플릿 내에서 이 템플릿을 참조할 수 있습니다. 중첩된 스택에 대한 자세한 내용은 [중첩 스택을 사용하여 템플릿을 재사용 가능한 조각으로 분할](#) 단원을 참조하십시오.

CloudFormation는 resource import를 사용하여 한 수준의 중첩만 지원합니다. 즉, 스택을 하위 스택으로 가져오거나 하위 요소가 있는 스택을 가져올 수 없습니다.

가져오기를 처음 사용하는 경우 먼저 [수동으로 CloudFormation 스택에 AWS 리소스 가져오기](#) 주제의 소개 정보를 검토하는 것이 좋습니다.

중첩 스택 가져오기 확인

중첩 스택 가져오기 작업 중에 CloudFormation에서는 다음과 같은 유효성 검사를 수행합니다.

- 상위 스택 템플릿의 중첩 `AWS::CloudFormation::Stack` 정의는 실제 중첩 스택의 템플릿과 일치합니다.
- 상위 스택 템플릿의 중첩 `AWS::CloudFormation::Stack` 정의에 대한 태그는 실제 중첩 스택 리소스의 태그와 일치합니다.

AWS Management Console을 사용하여 기존 스택 중첩

1. Retain [DeletionPolicy](#)를 사용하여 상위 스택 템플릿에 `AWS::CloudFormation::Stack` 리소스를 추가합니다. 다음 예제 상위 스택 템플릿에서 `MyNestedStack`은 가져오기 작업의 대상입니다.

JSON

```
{
```

```

"AWSTemplateFormatVersion" : "2010-09-09",
"Resources" : {
  "ServiceTable":{
    "Type":"AWS::DynamoDB::Table",
    "Properties":{
      "TableName":"Service",
      "AttributeDefinitions":[
        {
          "AttributeName":"key",
          "AttributeType":"S"
        }
      ],
      "KeySchema":[
        {
          "AttributeName":"key",
          "KeyType":"HASH"
        }
      ],
      "ProvisionedThroughput":{
        "ReadCapacityUnits":5,
        "WriteCapacityUnits":1
      }
    }
  },
  "MyNestedStack" : {
    "Type" : "AWS::CloudFormation::Stack",
    "DeletionPolicy": "Retain",
    "Properties" : {
      "TemplateURL" : "https://s3.amazonaws.com/cloudformation-templates-us-east-2/EC2ChooseAMI.template",
      "Parameters" : {
        "InstanceType" : "t1.micro",
        "KeyName" : "mykey"
      }
    }
  }
}

```

YAML

```

AWSTemplateFormatVersion: 2010-09-09
Resources:

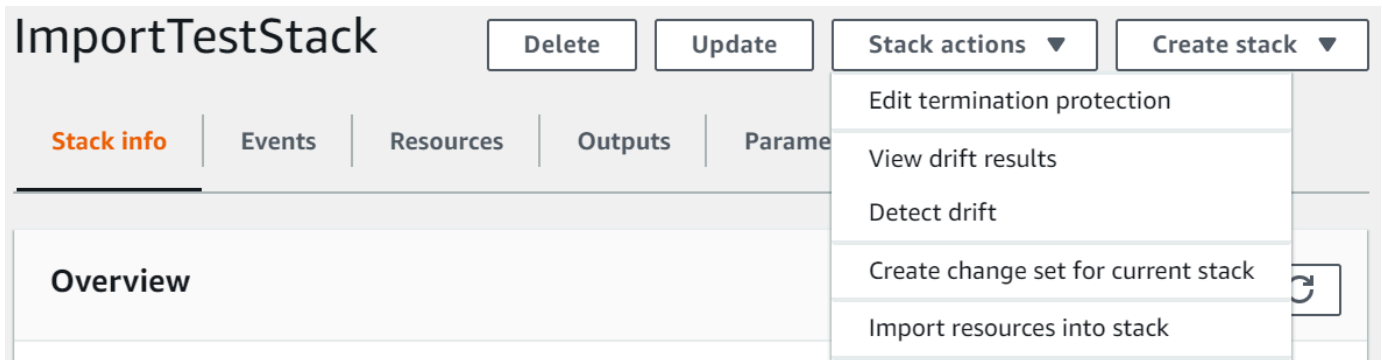
```

```

ServiceTable:
  Type: AWS::DynamoDB::Table
  Properties:
    TableName: Service
    AttributeDefinitions:
      - AttributeName: key
        AttributeType: S
    KeySchema:
      - AttributeName: key
        KeyType: HASH
    ProvisionedThroughput:
      ReadCapacityUnits: 5
      WriteCapacityUnits: 1
MyNestedStack:
  Type: AWS::CloudFormation::Stack
  DeletionPolicy: Retain
  Properties:
    TemplateURL: >-
      https://s3.amazonaws.com/cloudformation-templates-us-east-2/
      EC2ChooseAMI.template
    Parameters:
      InstanceType: t1.micro
      KeyName: mykey

```

2. CloudFormation 콘솔을 엽니다.
3. 스택 페이지에서 상위 스택을 선택한 상태에서 Stack actions(스택 작업)를 선택한 다음 Import resources into stack(스택으로 리소스 가져오기)을 선택합니다.



4. 이 작업 중에 제공해야 하는 항목 목록은 Import overview(가져오기 개요) 페이지를 참조하십시오. 그리고 다음을 선택합니다.
5. Specify template(템플릿 지정) 페이지에서 다음 방법 중 하나를 사용하여 업데이트된 템플릿을 제공하고 다음을 선택합니다.

- Amazon S3 URL을 선택한 다음, 텍스트 상자에 템플릿의 URL을 지정합니다.
 - 템플릿 파일 업로드를 선택한 다음 템플릿을 찾습니다.
6. Identify resources(리소스 식별) 페이지에서 `AWS::CloudFormation::Stack` 리소스를 식별합니다.
- Identifier property(식별자 속성)에서 리소스 식별자의 유형을 선택합니다. 예를 들어 `AWS::CloudFormation::Stack` 리소스는 `StackId` 속성을 사용하여 식별할 수 있습니다.
 - 식별자 값에 가져오려는 스택의 ARN을 입력합니다. 예를 들어 `arn:aws:cloudformation:us-west-2:12345678910:stack/mystack/5b918d10-cd98-11ea-90d5-0a9cd3354c10`입니다.

Resources to import (1)

The following logical IDs are new to your template. Provide the identifier values for the resource you are importing.

NestedStack

`AWS::CloudFormation::Stack`

Identifier property

StackId

Identifier value

Enter StackId

- 다음을 선택합니다.
7. Specify stack details(스택 세부 정보 지정) 페이지에서 어떤 파라미터든 수정하고 다음을 선택합니다. 이렇게 하면 변경 세트가 자동으로 생성됩니다.

Important

생성, 업데이트 또는 삭제 작업을 시작하는 기존 파라미터를 수정하면 가져오기 작업이 실패합니다.

- MyParentStack** 검토 페이지에서 올바른 리소스를 가져오고 있는지 확인하고, 리소스 가져오기를 선택합니다. 이렇게 하면 마지막 단계에서 생성된 변경 세트가 자동으로 실행됩니다. 이때 어떤 스택 레벨 태그든 가져온 리소스에 적용됩니다.
- 상위 스택에 대한 Stack details(스택 세부 정보) 페이지의 Events(이벤트) 창이 표시됩니다.

ImportTestStack

Delete

Update

Stack actions

Create stack

Stack info

Events

Resources

Outputs

Parameters

Template

Change sets

Events

Q Search events

Timestamp	Logical ID	Status	Status reason
2019-10-24 15:52:49 UTC-0700	ImportTestStack	IMPORT_COMPLETE	-
2019-10-24 15:52:47 UTC-0700	SampleIamUser	UPDATE_COMPLETE	-
2019-10-24 15:52:47 UTC-0700	SampleIamUser	UPDATE_IN_PROGRESS	Apply stack-level tags to imported resource if applicable.
2019-10-24 15:52:46 UTC-0700	SampleIamUser	IMPORT_COMPLETE	Resource import completed.
2019-10-24 15:52:45 UTC-0700	SampleIamUser	IMPORT_IN_PROGRESS	Resource import started.
2019-10-24 15:52:43 UTC-0700	ImportTestStack	IMPORT_IN_PROGRESS	User Initiated
2019-10-24 15:52:01 UTC-0700	ImportTestStack	REVIEW_IN_PROGRESS	User Initiated

Note

AWS::CloudFormation::Stack 리소스가 이미 CloudFormation에서 관리되기 때문에 이러한 가져오기 작업을 수행한 후에 상위 스택에서 드리프트 감지를 실행할 필요가 없습니다.

AWS CLI를 사용하여 기존 스택 중첩

1. Retain [DeletionPolicy](#)를 사용하여 상위 스택 템플릿에 AWS::CloudFormation::Stack 리소스를 추가합니다. 다음 예제 상위 템플릿에서 MyNestedStack은 가져오기 작업의 대상입니다.

JSON

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Resources" : {
    "ServiceTable":{
      "Type":"AWS::DynamoDB::Table",
      "Properties":{
        "TableName":"Service",
        "AttributeDefinitions":[
          {
            "AttributeName":"key",
            "AttributeType":"S"
          }
        ],

```

```

        "KeySchema":[
            {
                "AttributeName":"key",
                "KeyType":"HASH"
            }
        ],
        "ProvisionedThroughput":{
            "ReadCapacityUnits":5,
            "WriteCapacityUnits":1
        }
    },
    "MyNestedStack" : {
        "Type" : "AWS::CloudFormation::Stack",
        "DeletionPolicy": "Retain",
        "Properties" : {
            "TemplateURL" : "https://s3.amazonaws.com/cloudformation-templates-us-east-2/EC2ChooseAMI.template",
            "Parameters" : {
                "InstanceType" : "t1.micro",
                "KeyName" : "mykey"
            }
        }
    }
}

```

YAML

```

AWSTemplateFormatVersion: 2010-09-09
Resources:
  ServiceTable:
    Type: AWS::DynamoDB::Table
    Properties:
      TableName: Service
      AttributeDefinitions:
        - AttributeName: key
          AttributeType: S
      KeySchema:
        - AttributeName: key
          KeyType: HASH
      ProvisionedThroughput:
        ReadCapacityUnits: 5

```

```

    WriteCapacityUnits: 1
  MyNestedStack:
    Type: AWS::CloudFormation::Stack
    DeletionPolicy: Retain
    Properties:
      TemplateURL: >-
        https://s3.amazonaws.com/cloudformation-templates-us-east-2/
        EC2ChooseAMI.template
      Parameters:
        InstanceType: t1.micro
        KeyName: mykey

```

2. 다음 예제와 같이 JSON 문자열을 작성하고, 다음을 변경합니다.

- *MyNestedStack*을 템플릿에 지정된 대상 리소스의 논리적 ID로 바꿉니다.
- *arn:aws:cloudformation:us-west-2:12345678910:stack/mystack/5b918d10-cd98-11ea-90d5-0a9cd3354c10*을 가져오려는 스택의 ARN으로 바꿉니다.

```

[{"ResourceType":"AWS::CloudFormation::Stack","LogicalResourceId":"MyNestedStack","Resource
{"StackId":"arn:aws:cloudformation:us-east-2:123456789012:stack/mystack/5b918d10-
cd98-11ea-90d5-0a9cd3354c10"}]}]

```

또는 구성 파일에서 파라미터를 지정할 수 있습니다.

예를 들어 *MyNestedStack*을 가져오려면 다음 구성이 포함된 *ResourcesToImport.txt* 파일을 만들 수 있습니다.

JSON

```

[
  {
    "ResourceType":"AWS::CloudFormation::Stack",
    "LogicalResourceId":"MyNestedStack",
    "ResourceIdentifier": {
      "StackId":"arn:aws:cloudformation:us-west-2:12345678910:stack/
mystack/5b918d10-cd98-11ea-90d5-0a9cd3354c10"
    }
  }
]

```

YAML


```

ResourceType: AWS::CloudFormation::Stack
LogicalResourceId: MyNestedStack
ResourceIdentifier:
  StackId: >-
    arn:aws:cloudformation:us-west-2:12345678910:stack/mystack/5b918d10-
cd98-11ea-90d5-0a9cd3354c10

```

3. 변경 세트를 생성하려면 다음 `create-change-set` 명령을 사용하고 자리 표시자 텍스트를 변경합니다. `--change-set-type` 옵션의 경우, **IMPORT**의 값을 지정합니다. `--resources-to-import` 옵션의 경우 샘플 JSON 문자열을 방금 생성한 실제 JSON 문자열로 바꿉니다.

```

aws cloudformation create-change-set \
  --stack-name MyParentStack --change-set-name ImportChangeSet \
  --change-set-type IMPORT \
  --template-body file://TemplateToImport.json \
  --resources-to-
import '[{"ResourceType":"AWS::CloudFormation::Stack","LogicalResourceId":"MyNestedStack",
{"StackId":"arn:aws:cloudformation:us-west-2:12345678910:stack/mystack/5b918d10-
cd98-11ea-90d5-0a9cd3354c10"}}]'

```

Note

`--resources-to-import`에서는 인라인 YAML을 지원하지 않습니다. JSON 문자열의 인용 부호 요구 사항은 터미널에 따라 다릅니다. 자세한 내용은 AWS Command Line Interface 사용 설명서의 [문자열 안에 따옴표 사용](#)을 참조하세요.

또는 다음 예제와 같이 `--resources-to-import` 옵션에 대한 입력으로 파일 URL을 사용할 수 있습니다.

```

--resources-to-import file://ResourcesToImport.txt

```

성공하면 이 명령에서 다음 샘플 출력이 반환됩니다.

```

{
  "Id": "arn:aws:cloudformation:us-west-2:12345678910:changeSet/
ImportChangeSet/8ad75b3f-665f-46f6-a200-0b4727a9442e",
  "StackId": "arn:aws:cloudformation:us-west-2:12345678910:stack/
MyParentStack/4e345b70-1281-11ef-b027-027366d8e82b"
}

```

```
}
```

4. 변경 세트를 검토하여 올바른 리소스를 가져오고 있는지 확인합니다.

```
aws cloudformation describe-change-set --change-set-name ImportChangeSet
```

5. 변경 세트를 시작하고 스택을 소스 상위 스택으로 가져오려면 다음 `execute-change-set` 명령을 사용하고 자리 표시자 텍스트를 변경합니다. 이때 어떤 [스택 레벨 태그](#)든 가져온 리소스에 적용됩니다. 가져오기 작업 (IMPORT_COMPLETE)이 성공적으로 완료되면 스택이 성공적으로 중첩됩니다.

```
aws cloudformation execute-change-set --change-set-name ImportChangeSet
```

Note

AWS::CloudFormation::Stack 리소스가 이미 CloudFormation에서 관리되기 때문에 이러한 가져오기 작업을 수행한 후에 상위 스택에서 드리프트 감지를 실행할 필요가 없습니다.

자동으로 CloudFormation 스택에 AWS 리소스 가져오기

이제 CloudFormation 스택을 생성하거나 업데이트할 때 명명된 리소스를 자동으로 가져올 수 있습니다. 명명된 리소스란, 사용자 지정 이름이 지정된 리소스입니다. 자세한 내용은 CloudFormation 템플릿 참조의 [이름 유형](#)을 참조하세요.

자동 가져오기를 시작하면 CloudFormation은 템플릿과 일치하는 기존 리소스가 있는지 확인하고 배포 중에 이를 가져옵니다. 중첩 스택의 경우 루트 스택에서 변경 세트를 생성합니다.

가져오기가 완료된 후 후속 스택 작업을 수행하기 전에 가져오기를 수행한 리소스에 대해 드리프트 감지를 실행하는 것이 좋습니다. 드리프트 감지는 템플릿 구성이 실제 구성과 일치하는지 확인합니다. 자세한 내용은 [전체 CloudFormation 스택의 드리프트 감지](#) 섹션을 참조하세요.

리소스를 가져오려면 다음 요구 사항을 충족해야 합니다.

- 리소스의 템플릿에 정적 사용자 지정 이름이 정의되어 있어야 합니다. 동적 이름(!Ref 또는 기타 함수 사용)은 현재 지원되지 않습니다.
- 리소스의 DeletionPolicy가 Retain 또는 RetainExceptOnCreate이어야 합니다.
- 리소스가 미리 다른 CloudFormation 스택에 속하지 않아야 합니다.

- 리소스 유형이 CloudFormation 가져오기 작업을 지원해야 합니다. 자세한 내용은 [리소스 유형 지원](#) 섹션을 참조하세요.
- 리소스 유형의 기본 ID는 템플릿에 있어야 합니다. 읽기 전용 속성이 있는 기본 ID는 지원되지 않습니다. 유형의 기본 ID가 무엇인지 확인하려면 리소스 스키마에서 `primaryIdentifier` 속성을 찾습니다. 속성에 대한 자세한 내용은 [primaryIdentifier](#)를 참조하세요.

Example자동 가져오기 예제

다음 예제에서는 변경 세트(CreateChangeSet)를 사용하여 템플릿 파일(template.yaml)을 기반으로 스택(my-stack)을 생성하고 일치하는 리소스를 자동으로 가져옵니다.

```
aws cloudformation create-change-set \  
  --stack-name my-stack \  
  --change-set-name CreateChangeSet \  
  --change-set-type CREATE \  
  --template-body file://template.yaml \  
  --import-existing-resources
```

문제 해결

자동 가져오기에 실패하면 문제 해결을 위해 다음을 수행하세요.

- 템플릿의 리소스 이름이 리소스 이름과 정확히 일치하는지 확인
- 리소스가 다른 스택에서 아직 관리되지 않는지 확인
- 리소스 유형이 가져오기 작업을 지원하는지 확인
- 리소스 유형에 필요한 모든 속성이 템플릿에 포함되어 있는지 확인

가져오기 작업 되돌리기

가져오기 작업을 되돌리려면 템플릿에서 제거할 리소스에 대한 Retain 삭제 정책을 지정하여 스택에서 삭제할 때 해당 리소스가 보존되도록 합니다.

AWS Management Console을 사용하여 가져오기 작업 되돌리기

1. 스택에서 제거할 리소스에 대한 Retain [DeletionPolicy](#)를 지정합니다. 다음 예제 템플릿에서 GamesTable는 이러한 되돌리기 작업의 대상입니다.

Example JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Import test",
  "Resources": {
    "ServiceTable": {
      "Type": "AWS::DynamoDB::Table",
      "Properties": {
        "TableName": "Service",
        "AttributeDefinitions": [
          {
            "AttributeName": "key",
            "AttributeType": "S"
          }
        ],
        "KeySchema": [
          {
            "AttributeName": "key",
            "KeyType": "HASH"
          }
        ],
        "ProvisionedThroughput": {
          "ReadCapacityUnits": 5,
          "WriteCapacityUnits": 1
        }
      }
    },
    "GamesTable": {
      "Type": "AWS::DynamoDB::Table",
      "DeletionPolicy": "Retain",
      "Properties": {
        "TableName": "Games",
        "AttributeDefinitions": [
          {
            "AttributeName": "key",
            "AttributeType": "S"
          }
        ],
        "KeySchema": [
          {
            "AttributeName": "key",
            "KeyType": "HASH"
          }
        ]
      }
    }
  }
}
```

```
}  
    ],  
    "ProvisionedThroughput": {  
        "ReadCapacityUnits": 5,  
        "WriteCapacityUnits": 1  
    }  
}  
}  
}
```

2. CloudFormation 콘솔을 열어 스택 업데이트를 수행하여 삭제 정책을 적용합니다.
 - a. 스택 페이지에서 스택을 선택한 상태에서 업데이트를 선택한 다음, Update stack(standard) (스택 업데이트(표준))를 선택합니다.
 - b. Prepare template(템플릿 준비)에서 Replace current template(현재 템플릿 바꾸기)을 선택합니다.
 - c. Specify template(템플릿 지정)에서 GamesTable에 DeletionPolicy 속성이 설정된 업데이트된 소스 템플릿을 제공한 후, 다음을 선택합니다.
 - Amazon S3 URL을 선택한 다음 텍스트 상자에 업데이트된 소스 템플릿의 URL을 지정합니다.
 - 템플릿 파일 업로드를 선택한 다음 업데이트된 소스 템플릿 파일을 찾습니다.
 - d. Specify stack details(스택 세부 정보 지정) 페이지에서 어떤 변경도 필요하지 않습니다. 다음을 선택합니다.
 - e. Configure stack options(스택 옵션 구성) 페이지에서 어떤 변경도 필요하지 않습니다. 다음을 선택합니다.
 - f. **MyStack** 검토 페이지에서 변경 사항을 검토합니다. 템플릿에 IAM 리소스가 포함되어 있는 경우 이 템플릿이 IAM 리소스를 생성할 수 있음을 확인합니다(I acknowledge that this template may create IAM resources)를 선택하여 템플릿에서 IAM 리소스를 사용하도록 지정합니다. 자세한 내용은 [CloudFormation 템플릿에서 IAM 리소스 승인](#) 섹션을 참조하세요. 그런 다음 변경 세트를 생성하여 소스 스택을 업데이트하거나 소스 스택을 직접 업데이트하십시오.
3. 스택 템플릿에서 리소스, 관련 파라미터 및 출력을 제거합니다. 이 예제에서는 이제 소스 템플릿이 다음과 같이 보입니다.

Example JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Import test",
  "Resources": {
    "ServiceTable": {
      "Type": "AWS::DynamoDB::Table",
      "Properties": {
        "TableName": "Service",
        "AttributeDefinitions": [
          {
            "AttributeName": "key",
            "AttributeType": "S"
          }
        ],
        "KeySchema": [
          {
            "AttributeName": "key",
            "KeyType": "HASH"
          }
        ],
        "ProvisionedThroughput": {
          "ReadCapacityUnits": 5,
          "WriteCapacityUnits": 1
        }
      }
    }
  }
}
```

4. 2단계를 반복하여 리소스(GameTable)와 관련 파라미터 및 출력을 스택에서 삭제합니다.

AWS CLI을 사용하여 가져오기 작업 되돌리기

1. 스택에서 제거할 리소스에 대한 Retain [DeletionPolicy](#)를 지정합니다. 다음 예제 템플릿에서 GameTable는 이러한 되돌리기 작업의 대상입니다.

Example JSON

```
{
```

```
"AWSTemplateFormatVersion": "2010-09-09",
"Description": "Import test",
"Resources": {
  "ServiceTable": {
    "Type": "AWS::DynamoDB::Table",
    "Properties": {
      "TableName": "Service",
      "AttributeDefinitions": [
        {
          "AttributeName": "key",
          "AttributeType": "S"
        }
      ],
      "KeySchema": [
        {
          "AttributeName": "key",
          "KeyType": "HASH"
        }
      ],
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 5,
        "WriteCapacityUnits": 1
      }
    }
  },
  "GamesTable": {
    "Type": "AWS::DynamoDB::Table",
    "DeletionPolicy": "Retain",
    "Properties": {
      "TableName": "Games",
      "AttributeDefinitions": [
        {
          "AttributeName": "key",
          "AttributeType": "S"
        }
      ],
      "KeySchema": [
        {
          "AttributeName": "key",
          "KeyType": "HASH"
        }
      ],
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 5,
```

```

        "WriteCapacityUnits": 1
      }
    }
  }
}

```

2. 스택을 업데이트하여 삭제 정책을 리소스에 적용합니다.

```
aws cloudformation update-stack --stack-name MyStack
```

3. 스택 템플릿에서 리소스, 관련 파라미터 및 출력을 제거합니다. 이 예제에서는 이제 소스 템플릿이 다음과 같이 보입니다.

Example JSON

```

{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Import test",
  "Resources": {
    "ServiceTable": {
      "Type": "AWS::DynamoDB::Table",
      "Properties": {
        "TableName": "Service",
        "AttributeDefinitions": [
          {
            "AttributeName": "key",
            "AttributeType": "S"
          }
        ],
        "KeySchema": [
          {
            "AttributeName": "key",
            "KeyType": "HASH"
          }
        ],
        "ProvisionedThroughput": {
          "ReadCapacityUnits": 5,
          "WriteCapacityUnits": 1
        }
      }
    }
  }
}

```



```
}
```

4. 스택을 업데이트하여 리소스(GameTable)와 관련 파라미터 및 출력을 스택에서 삭제합니다.

```
aws cloudformation update-stack --stack-name MyStack
```

스택 리팩터링

스택 리팩터링을 사용하면 기존 리소스 속성 및 데이터를 유지하면서 CloudFormation 스택에서 리소스를 재구성할 수 있습니다. 스택 간에 리소스를 이동하거나 대형 스택을 더 작은 스택으로 분할하거나 여러 스택을 하나로 결합할 수 있습니다.

주제

- [스택 리팩터링 작동 방식](#)
- [스택 리팩터링 고려 사항](#)
- [사전 조건](#)
- [스택 리팩터링\(콘솔\)](#)
- [스택 리팩터링\(AWS CLI\)](#)
- [리소스 제한 사항](#)

스택 리팩터링 작동 방식

스택 리팩터링에는 다음 단계가 포함됩니다.

1. 현재 인프라 평가 - 기존 CloudFormation 스택 및 리소스를 검토하여 스택 리팩터링 기회를 식별합니다.
2. 리팩터링 계획 - 리소스를 구성하는 방법을 정의합니다. 종속 항목, 이름 지정 규칙 및 운영 제한을 고려합니다. 나중에 CloudFormation 검증에 영향을 미칠 수 있습니다.
3. 대상 스택 결정 - 리소스를 리팩터링할 스택을 결정합니다. 최소 2개 스택에서 최대 5개 스택(AWS CLI 사용) 사이로 리소스를 이동할 수 있습니다. 중첩 스택 사이에서 리소스를 이동할 수 있습니다.
4. 템플릿 업데이트 - 템플릿 사이에서 리소스 정의를 이동하는 등 계획된 변경 사항을 반영하도록 CloudFormation 템플릿을 변경합니다. 이 프로세스 중에 논리적 ID의 이름을 바꿀 수 있습니다.
5. 스택 리팩터링 생성 - 리팩터링하려는 스택 이름 및 템플릿의 목록을 제공합니다.

6. 리팩터링 영향 검토 및 충돌 해결 - CloudFormation은 사용자가 제공하는 템플릿을 검증하고 스택 간 종속성, 태그 업데이트 문제가 있는 리소스 유형 및 리소스 논리적 ID 충돌을 확인합니다.

검증이 성공하면 CloudFormation은 실행 중에 발생할 리팩터링 작업의 미리 보기를 생성합니다.

검증이 실패하면 식별된 문제를 해결하고 다시 시도합니다. 충돌이 있는 경우 충돌하는 리소스의 소스 및 대상을 보여주는 리소스 논리적 ID 매핑을 제공합니다.

7. 리팩터링 실행 - 변경 사항이 리팩터링 목표에 부합하는지 확인한 후 스택 리팩터링을 완료합니다.
8. 모니터링 - 실행 상태를 추적하여 작업이 성공적으로 완료되는지 확인합니다.

스택 리팩터링 고려 사항

스택을 리팩터링할 때 다음 사항에 유의하세요.

- 스택 리팩터링은 기존 리소스를 재구성하는 작업으로 제한됩니다. 리팩터링 중에는 리소스를 생성 또는 삭제하거나 리소스 구성을 수정하거나 새 파라미터, 조건 또는 매핑을 변경하거나 추가할 수 없습니다. 이러한 변경을 수행하려면 먼저 스택을 업데이트한 다음 스택 리팩터링을 수행합니다.
- 동일한 리소스를 여러 스택으로 리팩터링할 수 없습니다.
- `AWS::StackName`과 같이 소스 스택과 대상 스택 간에 값이 다른 가상 파라미터를 참조하는 리소스는 리팩터링할 수 없습니다.
- CloudFormation은 빈 스택을 지원하지 않습니다. 리팩터링으로 인해 스택에 리소스가 없게 되는 경우 [create-stack-refactor](#)를 실행하기 전에 먼저 해당 스택에 리소스를 하나 이상 추가해야 합니다. 이는 `AWS::SNS::Topic` 또는 `AWS::CloudFormation::WaitCondition`과 같은 간단한 리소스일 수 있습니다. 예제:

```
Resources:
  MySimpleSNSTopic:
    Type: AWS::SNS::Topic
    Properties:
      DisplayName: MySimpleTopic
```

- 스택 리팩터링은 정책이 허용 또는 거부하는 대상과 관계없이 스택 정책이 연결된 스택을 지원하지 않습니다.

사전 조건

스택을 리팩터링하려면 수정된 템플릿을 이미 생성한 상태여야 합니다.

[get-template](#) 명령을 사용하여 리팩터링하려는 스택의 CloudFormation 템플릿을 검색합니다.

```
aws cloudformation get-template --stack-name Stack1
```

템플릿이 있으면 선택한 통합 개발 환경(IDE)을 사용하여 원하는 구조 및 리소스 구성을 사용하도록 템플릿을 업데이트합니다.

스택 리팩터링(콘솔)

다음 절차를 사용하여 콘솔에서 스택을 리팩터링합니다.

스택을 리팩터링하는 방법

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 스택이 위치한 AWS 리전을 선택하세요.
3. 왼쪽에 있는 탐색 창에서 스택 리팩터링을 선택하세요.
4. 스택 리팩터링 페이지에서 스택 리팩터링 시작을 선택하세요.
5. 설명에 스택 리팩터링을 식별하는 데 도움이 되는 설명을 입력하세요. 그리고 다음을 선택합니다.
6. 스택 1의 경우 다음을 수행하세요.

- a. 기존 스택의 템플릿 업데이트 또는 새 스택 생성을 선택하세요.

기존 스택의 템플릿 업데이트를 선택한 경우 목록에서 기존 스택을 선택하세요. 또는 스택 ARN 입력을 선택하여 기존 스택의 ARN을 입력하세요.

새 스택 생성을 선택한 경우 스택 이름에 새 스택의 이름을 입력하세요.

- b. 기존 템플릿을 리팩터링된 템플릿으로 교체에서 Amazon S3 URL 또는 템플릿 파일 업로드를 선택하여 스택 1에 원하는 템플릿을 업로드하세요.
 - c. 다음을 선택합니다.
7. 스택 2의 경우 다음을 수행하세요.

- a. 기존 스택의 템플릿 업데이트 또는 새 스택 생성을 선택하세요.

기존 스택의 템플릿 업데이트를 선택한 경우 목록에서 기존 스택을 선택하세요. 또는 스택 ARN 입력을 선택하여 기존 스택의 ARN을 입력하세요.

새 스택 생성을 선택한 경우 스택 이름에 새 스택의 이름을 입력하세요.

- b. 기존 템플릿을 리팩터링된 템플릿으로 교체에서 Amazon S3 URL 또는 템플릿 파일 업로드를 선택하여 스택 2에 원하는 템플릿을 업로드하세요.
 - c. 다음을 선택합니다.
8. 논리적 리소스 ID 이름 바꾸기 지정 페이지에서 CloudFormation이 표시된 리소스를 올바른 논리적 ID에 매핑하여 스택을 리팩터링하는 방법을 알고 있는지 확인하세요. 스택 리팩터링의 일부로 리소스의 논리적 ID가 변경된 경우 소스 스택 이름, 원래 논리적 ID, 대상 스택 이름 및 이름이 바뀐 논리적 ID를 제공하여 이름이 변경된 방법을 지정해야 합니다. 경우에 따라 CloudFormation 콘솔이 리소스 매핑을 자동으로 감지할 수 있으며, 진행하기 전에 미리 채워진 리소스 매핑이 올바른지 간단히 확인할 수 있습니다.
9. 다음을 선택합니다.
10. 검토 및 실행 페이지에서 이전 단계의 모든 선택 사항을 검토하고 모든 항목이 올바르게 설정되었는지 확인하세요.
11. 스택을 리팩터링할 준비가 되면 스택 리팩터링 실행을 선택하세요.

스택 리팩터링(AWS CLI)

스택 리팩터링을 위한 AWS CLI 명령은 다음과 같습니다.

- [create-stack-refactor](#): 계획된 변경 사항을 검증하고 미리 보기를 생성합니다.
- [describe-stack-refactor](#): 스택 리팩터링 작업의 상태 및 세부 정보를 검색합니다.
- [execute-stack-refactor](#): 검증된 스택 리팩터링 작업을 완료합니다.
- [list-stack-refactors](#): 계정의 모든 스택 리팩터링 작업을 현재 상태 및 기본 정보와 함께 나열합니다.
- [list-stack-refactor-actions](#): CloudFormation이 리팩터링 실행 중에 각 스택 및 리소스에서 수행할 특정 작업에 대한 미리 보기를 표시합니다.

AWS CLI를 사용하여 스택을 생성하려면 다음 절차를 따릅니다.

스택을 리팩터링하는 방법

1. [create-stack-refactor](#) 명령을 사용하고 리팩터링할 스택의 스택 이름과 업데이트된 템플릿을 제공합니다. 새 스택이 아직 없는 경우 CloudFormation이 새 스택을 생성하도록 허용하는 `--enable-stack-creation` 옵션을 포함합니다.

```
aws cloudformation create-stack-refactor \
  --stack-definitions \
```

```
StackName=Stack1,TemplateBody@=file://template1-updated.yaml \
StackName=Stack2,TemplateBody@=file://template2-updated.yaml \
--enable-stack-creation
```

이 명령은 이후 단계에서 사용할 StackRefactorId를 반환합니다.

```
{
  "StackRefactorId": "9c384f70-4e07-4ed7-a65d-fee5eb430841"
}
```

템플릿 검증 중에 충돌이 감지되면(다음 단계에서 확인할 수 있음) [create-stack-refactor](#) 명령을 --resource-mappings 옵션과 함께 사용합니다.

```
aws cloudformation create-stack-refactor \
  --stack-definitions \
    StackName=Stack1,TemplateBody@=file://template1-updated.yaml \
    StackName=Stack2,TemplateBody@=file://template2-updated.yaml \
  --enable-stack-creation \
  --resource-mappings file://resource-mapping.json
```

다음은 예 resource-mapping.json 파일입니다.

```
[
  {
    "Source": {
      "StackName": "Stack1",
      "LogicalResourceId": "MySNSTopic"
    },
    "Destination": {
      "StackName": "Stack2",
      "LogicalResourceId": "MyLambdaSNSTopic"
    }
  }
]
```

2. [describe-stack-refactor](#) 명령을 사용하여 Status가 CREATE_COMPLETE인지 확인합니다. 이렇게 하면 검증이 완료됩니다.

```
aws cloudformation describe-stack-refactor \
  --stack-refactor-id 9c384f70-4e07-4ed7-a65d-fee5eb430841
```

출력 예시:

```
{
  "StackRefactorId": "9c384f70-4e07-4ed7-a65d-fee5eb430841",
  "StackIds": [
    "arn:aws:cloudformation:us-east-1:123456789012:stack/Stack1/3e6a1ff0-94b1-11f0-aa6f-0a88d2e03acf",
    "arn:aws:cloudformation:us-east-1:123456789012:stack/Stack2/5da91650-94b1-11f0-81cf-0a23500e151b"
  ],
  "ExecutionStatus": "AVAILABLE",
  "Status": "CREATE_COMPLETE"
}
```

3. [list-stack-refactor-actions](#) 명령을 사용하여 실행할 특정 작업을 미리 봅니다.

```
aws cloudformation list-stack-refactor-actions \
  --stack-refactor-id 9c384f70-4e07-4ed7-a65d-fee5eb430841
```

출력 예시:

```
{
  "StackRefactorActions": [
    {
      "Action": "MOVE",
      "Entity": "RESOURCE",
      "PhysicalResourceId": "MyTestLambdaRole",
      "Description": "No configuration changes detected.",
      "Detection": "AUTO",
      "TagResources": [],
      "UntagResources": [],
      "ResourceMapping": {
        "Source": {
          "StackName": "arn:aws:cloudformation:us-east-1:123456789012:stack/Stack1/3e6a1ff0-94b1-11f0-aa6f-0a88d2e03acf",
          "LogicalResourceId": "MyLambdaRole"
        },
        "Destination": {
          "StackName": "arn:aws:cloudformation:us-east-1:123456789012:stack/Stack2/5da91650-94b1-11f0-81cf-0a23500e151b",
          "LogicalResourceId": "MyLambdaRole"
        }
      }
    }
  ]
}
```

```

    }
  },
  {
    "Action": "MOVE",
    "Entity": "RESOURCE",
    "PhysicalResourceId": "MyTestFunction",
    "Description": "Resource configuration changes will be validated during
refactor execution.",
    "Detection": "AUTO",
    "TagResources": [
      {
        "Key": "aws:cloudformation:stack-name",
        "Value": "Stack2"
      },
      {
        "Key": "aws:cloudformation:logical-id",
        "Value": "MyFunction"
      },
      {
        "Key": "aws:cloudformation:stack-id",
        "Value": "arn:aws:cloudformation:us-east-1:123456789012:stack/
Stack2/5da91650-94b1-11f0-81cf-0a23500e151b"
      }
    ],
    "UntagResources": [
      "aws:cloudformation:stack-name",
      "aws:cloudformation:logical-id",
      "aws:cloudformation:stack-id"
    ],
    "ResourceMapping": {
      "Source": {
        "StackName": "arn:aws:cloudformation:us-
east-1:123456789012:stack/Stack1/3e6a1ff0-94b1-11f0-aa6f-0a88d2e03acf",
        "LogicalResourceId": "MyFunction"
      },
      "Destination": {
        "StackName": "arn:aws:cloudformation:us-
east-1:123456789012:stack/Stack2/5da91650-94b1-11f0-81cf-0a23500e151b",
        "LogicalResourceId": "MyFunction"
      }
    }
  }
]

```

```
}
```

4. 변경 사항을 검토하고 확인한 후 [execute-stack-refactor](#) 명령을 사용하여 스택 리팩터링 작업을 완료합니다.

```
aws cloudformation execute-stack-refactor \
  --stack-refactor-id 9c384f70-4e07-4ed7-a65d-fee5eb430841
```

5. [describe-stack-refactor](#) 명령을 사용하여 실행 상태를 모니터링합니다.

```
aws cloudformation describe-stack-refactor \
  --stack-refactor-id 9c384f70-4e07-4ed7-a65d-fee5eb430841
```

출력 예시:

```
{
  "StackRefactorId": "9c384f70-4e07-4ed7-a65d-fee5eb430841",
  "StackIds": [
    "arn:aws:cloudformation:us-east-1:123456789012:stack/Stack1/3e6a1ff0-94b1-11f0-aa6f-0a88d2e03acf",
    "arn:aws:cloudformation:us-east-1:123456789012:stack/Stack2/5da91650-94b1-11f0-81cf-0a23500e151b"
  ],
  "ExecutionStatus": "SUCCEEDED",
  "Status": "COMPLETE"
}
```

리소스 제한 사항

- 스택 리팩터링은 provisioningType가 FULLY_MUTABLE인 리소스 유형만 지원하며, [describe-type](#) 명령을 사용하여 이를 확인할 수 있습니다.
- CloudFormation은 리팩터링 생성 중에 리소스 자격을 검증하고 [describe-stack-refactor](#) 명령의 출력에서 지원되지 않는 모든 리소스를 보고합니다.
- 다음 리소스는 스택 리팩터링에서 지원되지 않습니다.
 - AWS::ACMPCA::Certificate
 - AWS::ACMPCA::CertificateAuthority
 - AWS::ACMPCA::CertificateAuthorityActivation
 - AWS::ApiGateway::BasePathMapping

- AWS::ApiGateway::Method
- AWS::AppConfig::ConfigurationProfile
- AWS::AppConfig::Deployment
- AWS::AppConfig::Environment
- AWS::AppConfig::Extension
- AWS::AppConfig::ExtensionAssociation
- AWS::AppStream::DirectoryConfig
- AWS::AppStream::StackFleetAssociation
- AWS::AppStream::StackUserAssociation
- AWS::AppStream::User
- AWS::BackupGateway::Hypervisor
- AWS::CertificateManager::Certificate
- AWS::CloudFormation::CustomResource
- AWS::CloudFormation::Macro
- AWS::CloudFormation::WaitCondition
- AWS::CloudFormation::WaitConditionHandle
- AWS::CodeDeploy::DeploymentGroup
- AWS::CodePipeline::CustomActionType
- AWS::Cognito::UserPoolRiskConfigurationAttachment
- AWS::Cognito::UserPoolUICustomizationAttachment
- AWS::Cognito::UserPoolUserToGroupAttachment
- AWS::Config::ConfigRule
- AWS::Config::ConfigurationRecorder
- AWS::Config::DeliveryChannel
- AWS::DataBrew::Dataset
- AWS::DataBrew::Job
- AWS::DataBrew::Project
- AWS::DataBrew::Recipe
- AWS::DataBrew::Ruleset
- AWS::DataBrew::Schedule

- AWS::DataZone::DataSource
- AWS::DataZone::Environment
- AWS::DataZone::EnvironmentBlueprintConfiguration
- AWS::DataZone::EnvironmentProfile
- AWS::DataZone::Project
- AWS::DataZone::SubscriptionTarget
- AWS::DirectoryService::MicrosoftAD
- AWS::DynamoDB::GlobalTable
- AWS::EC2::CustomerGateway
- AWS::EC2::EIP
- AWS::EC2::LaunchTemplate
- AWS::EC2::NetworkInterfacePermission
- AWS::EC2::PlacementGroup
- AWS::EC2::SpotFleet
- AWS::EC2::VPCDHCPOptionsAssociation
- AWS::EC2::VolumeAttachment
- AWS::EMR::Cluster
- AWS::EMR::InstanceFleetConfig
- AWS::EMR::InstanceGroupConfig
- AWS::ElastiCache::CacheCluster
- AWS::ElastiCache::ReplicationGroup
- AWS::ElastiCache::SecurityGroup
- AWS::ElastiCache::SecurityGroupIngress
- AWS::ElasticBeanstalk::ConfigurationTemplate
- AWS::ElasticLoadBalancing::LoadBalancer
- AWS::ElasticLoadBalancingV2::ListenerCertificate
- AWS::Elasticsearch::Domain
- AWS::FIS::ExperimentTemplate
- AWS::Glue::Schema
- AWS::GuardDuty::IPSet

- AWS::GuardDuty::PublishingDestination
- AWS::GuardDuty::ThreatIntelSet
- AWS::IAM::AccessKey
- AWS::IAM::UserToGroupAddition
- AWS::ImageBuilder::Component
- AWS::IoT::PolicyPrincipalAttachment
- AWS::IoT::ThingPrincipalAttachment
- AWS::IoTFleetWise::Campaign
- AWS::IoTWireless::WirelessDeviceImportTask
- AWS::Lambda::EventInvokeConfig
- AWS::Lex::BotVersion
- AWS::M2::Application
- AWS::MSK::Configuration
- AWS::MSK::ServerlessCluster
- AWS::Maester::DocumentType
- AWS::MediaTailor::Channel
- AWS::NeptuneGraph::PrivateGraphEndpoint
- AWS::Omics::AnnotationStore
- AWS::Omics::ReferenceStore
- AWS::Omics::SequenceStore
- AWS::OpenSearchServerless::Collection
- AWS::OpsWorks::App
- AWS::OpsWorks::ElasticLoadBalancerAttachment
- AWS::OpsWorks::Instance
- AWS::OpsWorks::Layer
- AWS::OpsWorks::Stack
- AWS::OpsWorks::UserProfile
- AWS::OpsWorks::Volume
- AWS::PCAConnectorAD::Connector
- AWS::PCAConnectorAD::DirectoryRegistration

- AWS::PCAConnectorAD::Template
- AWS::PCAConnectorAD::TemplateGroupAccessControlEntry
- AWS::Panorama::PackageVersion
- AWS::QuickSight::Theme
- AWS::RDS::DBSecurityGroup
- AWS::RDS::DBSecurityGroupIngress
- AWS::Redshift::ClusterSecurityGroup
- AWS::Redshift::ClusterSecurityGroupIngress
- AWS::RefactorSpaces::Environment
- AWS::RefactorSpaces::Route
- AWS::RefactorSpaces::Service
- AWS::RoboMaker::RobotApplication
- AWS::RoboMaker::SimulationApplication
- AWS::Route53::RecordSet
- AWS::Route53::RecordSetGroup
- AWS::SDB::Domain
- AWS::SageMaker::InferenceComponent
- AWS::ServiceCatalog::PortfolioPrincipalAssociation
- AWS::ServiceCatalog::PortfolioProductAssociation
- AWS::ServiceCatalog::PortfolioShare
- AWS::ServiceCatalog::TagOptionAssociation
- AWS::ServiceCatalogAppRegistry::AttributeGroupAssociation
- AWS::ServiceCatalogAppRegistry::ResourceAssociation
- AWS::StepFunctions::StateMachineVersion
- AWS::Synthetics::Canary
- AWS::VoiceID::Domain
- AWS::WAF::ByteMatchSet
- AWS::WAF::IPSet
- AWS::WAF::Rule
- AWS::WAF::SizeConstraintSet

- `AWS::WAF::SqlInjectionMatchSet`
- `AWS::WAF::WebACL`
- `AWS::WAF::XssMatchSet`
- `AWS::WAFv2::IPSet`
- `AWS::WAFv2::RegexPatternSet`
- `AWS::WAFv2::RuleGroup`
- `AWS::WAFv2::WebACL`
- `AWS::WorkSpaces::Workspace`

리소스 유형 지원

다음 테이블에서는 현재 가져오기, 드리프트 감지, 코드형 인프라(IaC) 생성기 작업을 지원하는 AWS 리소스 유형을 나열합니다. 각 리소스 유형 이름은 [CloudFormation 템플릿 참조 가이드](#)의 해당 참조 주제에 연결됩니다.

리소스 가져오기를 지원하는 리소스는 자동 가져오기도 지원할 수 있습니다. 자세한 내용은 [CloudFormation 스택에 AWS 리소스 가져오기](#) 섹션을 참조하세요.

이 목록은 AWS 리소스의 전체 목록이 아닙니다. 특정 리소스 유형이 나열되지 않은 경우 AWS Cloud Control API에서 액세스하지 못할 수 있습니다. 자세한 내용은 Cloud Control API 사용 설명서의 [Cloud Control API를 지원하는 리소스 유형](#)을 참조하세요. 각 AWS 서비스는 Cloud Control API를 통해 액세스할 수 있도록 설정할 리소스 유형을 결정합니다.

또한 CloudFormation에서는 프로비저닝 가능한 프라이빗 리소스 유형(FULLY_MUTABLE 또는 IMMUTABLE의 프로비저닝 유형)에 대한 가져오기 및 드리프트 감지 작업을 지원합니다. 프라이빗 리소스 유형에서 가져오기 및 드리프트 감지를 수행하려면 먼저 계정에 해당 리소스 유형의 기본 버전을 등록하고 프로비저닝 가능한지 확인해야 합니다. 자세한 내용은 [사용자와 공유된 서드 파티 비공개 확장 프로그램 사용](#) 섹션을 참조하세요.

IaC 생성기는 사용자 리전의 Cloud Control API와 호환되는 AWS 리소스만 지원합니다.

퍼블릭 및 프라이빗의 프로비저닝 가능한 리소스 유형에 대한 정보에 프로그래밍 방식으로 액세스하려는 경우 AWS Cloud Control API를 사용할 수 있습니다. 자세한 내용은 Cloud Control API 사용 설명서의 [리소스 유형이 Cloud Control API를 지원하는지 여부](#)를 참조하세요.

가져오기, 드리프트 감지 또는 IaC 생성기를 시작하려는 경우 다음과 같은 몇 가지 유용한 주제를 검토하세요.

- [CloudFormation 스택에 AWS 리소스 가져오기](#)
- [드리프트 감지를 사용하여 스택 및 리소스에 대한 비관리형 구성 변경 감지](#)
- [IaC 생성기를 사용하여 기존 리소스에서 템플릿 생성](#)

리소스	가져오기	드리프트 감지	IaC 생성기
AWS::ACMPCA::Certificate	 예	 아니요	 아니요
AWS::ACMPCA::CertificateAuthority	 예	 예	 예
AWS::ACMPCA::CertificateAuthorityActivation	 예	 아니요	 아니요
AWS::ACMPCA::Permission	 예	 예	 아니요
AWS::AIOps::InvestigationGroup	 예	 예	 아니요
AWS::APS::AnomalyDetector	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::APS::ResourcePolicy</u>	 예	 예	 아니요
<u>AWS::APS::RuleGroupsNamespace</u>	 예	 예	 예
<u>AWS::APS::Scraper</u>	 예	 예	 아니요
<u>AWS::APS::Workspace</u>	 예	 예	 예
<u>AWS::ARCRegionSwitch::Plan</u>	 예	 예	 아니요
<u>AWS::ARCZonalShift::AutoshiftObserve rNotificationStatus</u>	 예	 예	 아니요
<u>AWS::ARCZonalShift::ZonalAutoshiftCo nfiguration</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::AccessAnalyzer::Analyzer</u>	 예	 예	 예
<u>AWS::AmazonMQ::Broker</u>	 예	 예	 아니요
<u>AWS::AmazonMQ::Configuration</u>	 예	 예	 아니요
<u>AWS::Amplify::App</u>	 예	 예	 예
<u>AWS::Amplify::Branch</u>	 예	 예	 예
<u>AWS::Amplify::Domain</u>	 예	 예	 예
<u>AWS::AmplifyUIBuilder::Component</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::AmplifyUIBuilder::Form</u>	 예	 예	 예
<u>AWS::AmplifyUIBuilder::Theme</u>	 예	 예	 예
<u>AWS::ApiGateway::Account</u>	 예	 예	 아니요
<u>AWS::ApiGateway::ApiKey</u>	 예	 예	 예
<u>AWS::ApiGateway::Authorizer</u>	 예	 예	 아니요
<u>AWS::ApiGateway::BasePathMapping</u>	 예	 예	 아니요
<u>AWS::ApiGateway::BasePathMappingV2</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ApiGateway::ClientCertificate</u>	 예	 예	 예
<u>AWS::ApiGateway::Deployment</u>	 예	 예	 예
<u>AWS::ApiGateway::DocumentationPart</u>	 예	 예	 아니요
<u>AWS::ApiGateway::DocumentationVersion</u>	 예	 예	 예
<u>AWS::ApiGateway::DomainName</u>	 예	 예	 아니요
<u>AWS::ApiGateway::DomainNameAccessAssociation</u>	 예	 예	 아니요
<u>AWS::ApiGateway::DomainNameV2</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ApiGateway::GatewayResponse</u>	 예	 예	 아니요
<u>AWS::ApiGateway::Method</u>	 예	 예	 아니요
<u>AWS::ApiGateway::Model</u>	 예	 예	 아니요
<u>AWS::ApiGateway::RequestValidator</u>	 예	 예	 아니요
<u>AWS::ApiGateway::Resource</u>	 예	 예	 아니요
<u>AWS::ApiGateway::RestApi</u>	 예	 예	 예
<u>AWS::ApiGateway::Stage</u>	 예	 아니요	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ApiGateway::UsagePlan</u>	 예	 예	 예
<u>AWS::ApiGateway::UsagePlanKey</u>	 예	 아니요	 예
<u>AWS::ApiGateway::VpcLink</u>	 예	 예	 아니요
<u>AWS::ApiGatewayV2::Api</u>	 예	 예	 예
<u>AWS::ApiGatewayV2::ApiMapping</u>	 예	 예	 예
<u>AWS::ApiGatewayV2::Authorizer</u>	 예	 예	 아니요
<u>AWS::ApiGatewayV2::Deployment</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ApiGatewayV2::DomainName</u>	 예	 예	 예
<u>AWS::ApiGatewayV2::Integration</u>	 예	 예	 아니요
<u>AWS::ApiGatewayV2::IntegrationResponse</u>	 예	 예	 아니요
<u>AWS::ApiGatewayV2::Model</u>	 예	 예	 아니요
<u>AWS::ApiGatewayV2::Route</u>	 예	 예	 아니요
<u>AWS::ApiGatewayV2::RouteResponse</u>	 예	 예	 아니요
<u>AWS::ApiGatewayV2::RoutingRule</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ApiGatewayV2::VpcLink</u>	 예	 예	 예
<u>AWS::AppConfig::Application</u>	 예	 예	 예
<u>AWS::AppConfig::ConfigurationProfile</u>	 예	 예	 아니요
<u>AWS::AppConfig::Deployment</u>	 예	 예	 아니요
<u>AWS::AppConfig::DeploymentStrategy</u>	 예	 예	 아니요
<u>AWS::AppConfig::Environment</u>	 예	 예	 아니요
<u>AWS::AppConfig::Extension</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::AppConfig::ExtensionAssociation</u>	 예	 예	 아니요
<u>AWS::AppConfig::HostedConfigurationVersion</u>	 예	 예	 아니요
<u>AWS::AppFlow::Connector</u>	 예	 예	 예
<u>AWS::AppFlow::ConnectorProfile</u>	 예	 예	 예
<u>AWS::AppFlow::Flow</u>	 예	 예	 예
<u>AWS::AppIntegrations::Application</u>	 예	 예	 아니요
<u>AWS::AppIntegrations::DataIntegration</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::AppIntegrations::EventIntegration</u>	 예	 예	 예
<u>AWS::AppRunner::AutoScalingConfiguration</u>	 예	 예	 아니요
<u>AWS::AppRunner::ObservabilityConfiguration</u>	 예	 예	 아니요
<u>AWS::AppRunner::Service</u>	 예	 예	 아니요
<u>AWS::AppRunner::VpcConnector</u>	 예	 예	 아니요
<u>AWS::AppRunner::VpcIngressConnection</u>	 예	 예	 아니요
<u>AWS::AppStream::AppBlock</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::AppStream::AppBlockBuilder</u>	 예	 예	 아니요
<u>AWS::AppStream::Application</u>	 예	 예	 아니요
<u>AWS::AppStream::ApplicationEntitlementAssociation</u>	 예	 예	 아니요
<u>AWS::AppStream::ApplicationFleetAssociation</u>	 예	 예	 아니요
<u>AWS::AppStream::DirectoryConfig</u>	 예	 예	 예
<u>AWS::AppStream::Entitlement</u>	 예	 예	 아니요
<u>AWS::AppStream::ImageBuilder</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::AppSync::Api</u>	 예	 예	 아니요
<u>AWS::AppSync::ChannelNamespace</u>	 예	 예	 아니요
<u>AWS::AppSync::DataSource</u>	 예	 예	 아니요
<u>AWS::AppSync::DomainName</u>	 예	 예	 예
<u>AWS::AppSync::DomainNameApiAssociation</u>	 예	 예	 아니요
<u>AWS::AppSync::FunctionConfiguration</u>	 예	 예	 아니요
<u>AWS::AppSync::GraphQLApi</u>	 아니요	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::AppSync::Resolver</u>	 예	 예	 아니요
<u>AWS::AppSync::SourceApiAssociation</u>	 예	 예	 예
<u>AWS::AppTest::TestCase</u>	 예	 예	 아니요
<u>AWS::ApplicationAutoScaling::ScalableTarget</u>	 예	 예	 예
<u>AWS::ApplicationAutoScaling::ScalingPolicy</u>	 예	 아니요	 아니요
<u>AWS::ApplicationInsights::Application</u>	 예	 예	 아니요
<u>AWS::ApplicationSignals::Discovery</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ApplicationSignals::GroupingConfiguration</u>	 예	 예	 아니요
<u>AWS::ApplicationSignals::ServiceLevelObjective</u>	 예	 예	 아니요
<u>AWS::Athena::CapacityReservation</u>	 예	 예	 예
<u>AWS::Athena::DataCatalog</u>	 예	 예	 예
<u>AWS::Athena::NamedQuery</u>	 예	 예	 예
<u>AWS::Athena::PreparedStatement</u>	 예	 예	 예
<u>AWS::Athena::WorkGroup</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::AuditManager::Assessment</u>	 예	 아니요	 아니요
<u>AWS::AutoScaling::AutoScalingGroup</u>	 예	 아니요	 예
<u>AWS::AutoScaling::LaunchConfiguration</u>	 예	 예	 예
<u>AWS::AutoScaling::LifecycleHook</u>	 예	 예	 예
<u>AWS::AutoScaling::ScalingPolicy</u>	 예	 예	 예
<u>AWS::AutoScaling::ScheduledAction</u>	 예	 예	 예
<u>AWS::AutoScaling::WarmPool</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::B2BI::Capability</u>	 예	 예	 아니요
<u>AWS::B2BI::Partnership</u>	 예	 예	 아니요
<u>AWS::B2BI::Profile</u>	 예	 예	 아니요
<u>AWS::B2BI::Transformer</u>	 예	 예	 아니요
<u>AWS::BCMDDataExports::Export</u>	 예	 예	 아니요
<u>AWS::Backup::BackupPlan</u>	 예	 예	 아니요
<u>AWS::Backup::BackupSelection</u>	 예	 아니요	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Backup::BackupVault</u>	 예	 예	 아니요
<u>AWS::Backup::Framework</u>	 예	 예	 예
<u>AWS::Backup::LogicallyAirGappedBackupVault</u>	 예	 예	 아니요
<u>AWS::Backup::ReportPlan</u>	 예	 예	 예
<u>AWS::Backup::RestoreTestingPlan</u>	 예	 예	 아니요
<u>AWS::Backup::RestoreTestingSelection</u>	 예	 예	 아니요
<u>AWS::BackupGateway::Hypervisor</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Batch::ComputeEnvironment</u>	 예	 예	 예
<u>AWS::Batch::ConsumableResource</u>	 예	 예	 아니요
<u>AWS::Batch::JobDefinition</u>	 예	 예	 아니요
<u>AWS::Batch::JobQueue</u>	 예	 예	 예
<u>AWS::Batch::SchedulingPolicy</u>	 예	 예	 예
<u>AWS::Batch::ServiceEnvironment</u>	 예	 예	 아니요
<u>AWS::Bedrock::Agent</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Bedrock::AgentAlias</u>	 예	 예	 예
<u>AWS::Bedrock::ApplicationInferenceProfile</u>	 예	 예	 예
<u>AWS::Bedrock::AutomatedReasoningPolicy</u>	 예	 예	 아니요
<u>AWS::Bedrock::AutomatedReasoningPolicyVersion</u>	 예	 예	 아니요
<u>AWS::Bedrock::Blueprint</u>	 예	 예	 예
<u>AWS::Bedrock::DataAutomationProject</u>	 예	 예	 예
<u>AWS::Bedrock::DataSource</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Bedrock::Flow</u>	 예	 예	 예
<u>AWS::Bedrock::FlowAlias</u>	 예	 예	 예
<u>AWS::Bedrock::FlowVersion</u>	 예	 예	 예
<u>AWS::Bedrock::Guardrail</u>	 예	 예	 예
<u>AWS::Bedrock::GuardrailVersion</u>	 예	 예	 아니요
<u>AWS::Bedrock::IntelligentPromptRoute</u>	 예	 예	 예
<u>AWS::Bedrock::KnowledgeBase</u>	 예	 아니요	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Bedrock::Prompt</u>	 예	 예	 예
<u>AWS::Bedrock::PromptVersion</u>	 예	 예	 예
<u>AWS::BedrockAgentCore::BrowserCustom</u>	 예	 예	 아니요
<u>AWS::BedrockAgentCore::CodeInterpreterCustom</u>	 예	 예	 아니요
<u>AWS::BedrockAgentCore::Gateway</u>	 예	 예	 아니요
<u>AWS::BedrockAgentCore::GatewayTarget</u>	 예	 예	 아니요
<u>AWS::BedrockAgentCore::Memory</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::BedrockAgentCore::Runtime</u>	 예	 예	 아니요
<u>AWS::BedrockAgentCore::RuntimeEndpoint</u>	 예	 예	 아니요
<u>AWS::BedrockAgentCore::WorkloadIdentity</u>	 예	 예	 아니요
<u>AWS::Billing::BillingView</u>	 예	 예	 아니요
<u>AWS::BillingConductor::BillingGroup</u>	 예	 예	 예
<u>AWS::BillingConductor::CustomLineItem</u>	 예	 예	 예
<u>AWS::BillingConductor::PricingPlan</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::BillingConductor::PricingRule</u>	 예	 예	 예
<u>AWS::Budgets::BudgetsAction</u>	 예	 예	 예
<u>AWS::CE::AnomalyMonitor</u>	 예	 예	 아니요
<u>AWS::CE::AnomalySubscription</u>	 예	 예	 아니요
<u>AWS::CE::CostCategory</u>	 예	 예	 예
<u>AWS::CUR::ReportDefinition</u>	 예	 예	 아니요
<u>AWS::Cases::CaseRule</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Cases::Domain</u>	 예	 예	 아니요
<u>AWS::Cases::Field</u>	 예	 예	 아니요
<u>AWS::Cases::Layout</u>	 예	 예	 아니요
<u>AWS::Cases::Template</u>	 예	 예	 아니요
<u>AWS::Cassandra::Keyspace</u>	 예	 예	 아니요
<u>AWS::Cassandra::Table</u>	 예	 예	 예
<u>AWS::Cassandra::Type</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::CertificateManager::Account</u>	 예	 예	 아니요
<u>AWS::Chatbot::CustomAction</u>	 예	 예	 아니요
<u>AWS::Chatbot::MicrosoftTeamsChannelConfiguration</u>	 예	 예	 예
<u>AWS::Chatbot::SlackChannelConfiguration</u>	 예	 예	 예
<u>AWS::CleanRooms::AnalysisTemplate</u>	 예	 예	 예
<u>AWS::CleanRooms::Collaboration</u>	 예	 예	 예
<u>AWS::CleanRooms::ConfiguredTable</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::CleanRooms::ConfiguredTableAssociation</u>	 예	 예	 예
<u>AWS::CleanRooms::IdMappingTable</u>	 예	 예	 아니요
<u>AWS::CleanRooms::IdNamespaceAssociation</u>	 예	 예	 아니요
<u>AWS::CleanRooms::Membership</u>	 예	 예	 예
<u>AWS::CleanRooms::PrivacyBudgetTemplate</u>	 예	 예	 아니요
<u>AWS::CleanRoomsML::TrainingDataset</u>	 예	 예	 아니요
<u>AWS::CloudFormation::GuardHook</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::CloudFormation::HookDefaultVersion</u>	 예	 예	 예
<u>AWS::CloudFormation::HookTypeConfig</u>	 예	 예	 예
<u>AWS::CloudFormation::HookVersion</u>	 예	 예	 예
<u>AWS::CloudFormation::LambdaHook</u>	 예	 예	 아니요
<u>AWS::CloudFormation::ModuleDefaultVersion</u>	 예	 예	 아니요
<u>AWS::CloudFormation::ModuleVersion</u>	 예	 예	 아니요
<u>AWS::CloudFormation::PublicTypeVersion</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::CloudFormation::Publisher</u>	 예	 예	 아니요
<u>AWS::CloudFormation::ResourceDefaultVersion</u>	 예	 예	 아니요
<u>AWS::CloudFormation::ResourceVersion</u>	 예	 예	 아니요
<u>AWS::CloudFormation::Stack</u>	 예	 예	 아니요
<u>AWS::CloudFormation::StackSet</u>	 예	 예	 아니요
<u>AWS::CloudFormation::TypeActivation</u>	 예	 예	 아니요
<u>AWS::CloudFront::AnycastIpList</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::CloudFront::CachePolicy</u>	 예	 예	 예
<u>AWS::CloudFront::CloudFrontOriginAccessIdentity</u>	 예	 예	 예
<u>AWS::CloudFront::ConnectionFunction</u>	 예	 예	 아니요
<u>AWS::CloudFront::ConnectionGroup</u>	 예	 예	 아니요
<u>AWS::CloudFront::ContinuousDeploymentPolicy</u>	 예	 예	 예
<u>AWS::CloudFront::Distribution</u>	 예	 예	 예
<u>AWS::CloudFront::DistributionTenant</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::CloudFront::Function</u>	 예	 예	 아니요
<u>AWS::CloudFront::KeyGroup</u>	 예	 예	 예
<u>AWS::CloudFront::KeyValueStore</u>	 예	 예	 아니요
<u>AWS::CloudFront::MonitoringSubscription</u>	 예	 예	 아니요
<u>AWS::CloudFront::OriginAccessControl</u>	 예	 예	 예
<u>AWS::CloudFront::OriginRequestPolicy</u>	 예	 예	 예
<u>AWS::CloudFront::PublicKey</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::CloudFront::RealtimeLogConfig</u>	 예	 예	 예
<u>AWS::CloudFront::ResponseHeadersPolicy</u>	 예	 예	 아니요
<u>AWS::CloudFront::TrustStore</u>	 예	 예	 아니요
<u>AWS::CloudFront::VpcOrigin</u>	 예	 예	 아니요
<u>AWS::CloudTrail::Channel</u>	 예	 예	 아니요
<u>AWS::CloudTrail::Dashboard</u>	 예	 예	 아니요
<u>AWS::CloudTrail::EventDataStore</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::CloudTrail::ResourcePolicy</u>	 예	 예	 아니요
<u>AWS::CloudTrail::Trail</u>	 예	 예	 예
<u>AWS::CloudWatch::Alarm</u>	 예	 예	 예
<u>AWS::CloudWatch::CompositeAlarm</u>	 예	 예	 예
<u>AWS::CloudWatch::Dashboard</u>	 예	 예	 아니요
<u>AWS::CloudWatch::MetricStream</u>	 예	 예	 아니요
<u>AWS::CodeArtifact::Domain</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::CodeArtifact::PackageGroup</u>	 예	 예	 아니요
<u>AWS::CodeArtifact::Repository</u>	 예	 예	 예
<u>AWS::CodeBuild::Fleet</u>	 예	 예	 아니요
<u>AWS::CodeConnections::Connection</u>	 예	 예	 아니요
<u>AWS::CodeDeploy::Application</u>	 예	 예	 예
<u>AWS::CodeDeploy::DeploymentConfig</u>	 예	 예	 아니요
<u>AWS::CodeGuruProfiler::ProfilingGroup</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::CodeGuruReviewer::RepositoryAssociation</u>	 예	 예	 아니요
<u>AWS::CodePipeline::CustomActionType</u>	 예	 예	 아니요
<u>AWS::CodePipeline::Pipeline</u>	 예	 예	 아니요
<u>AWS::CodePipeline::Webhook</u>	 예	 예	 아니요
<u>AWS::CodeStarConnections::Connection</u>	 예	 예	 예
<u>AWS::CodeStarConnections::RepositoryLink</u>	 예	 예	 아니요
<u>AWS::CodeStarConnections::SyncConfiguration</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::CodeStarNotifications::NotificationRule</u>	 예	 예	 예
<u>AWS::Cognito::IdentityPool</u>	 예	 예	 아니요
<u>AWS::Cognito::IdentityPoolPrincipalTag</u>	 예	 예	 예
<u>AWS::Cognito::IdentityPoolRoleAttachment</u>	 예	 예	 아니요
<u>AWS::Cognito::LogDeliveryConfiguration</u>	 예	 아니요	 아니요
<u>AWS::Cognito::ManagedLoginBranding</u>	 예	 예	 아니요
<u>AWS::Cognito::Terms</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Cognito::UserPool</u>	 예	 아니요	 아니요
<u>AWS::Cognito::UserPoolClient</u>	 예	 예	 아니요
<u>AWS::Cognito::UserPoolDomain</u>	 예	 예	 아니요
<u>AWS::Cognito::UserPoolGroup</u>	 예	 예	 예
<u>AWS::Cognito::UserPoolIdentityProvider</u>	 예	 예	 아니요
<u>AWS::Cognito::UserPoolResourceServer</u>	 예	 예	 아니요
<u>AWS::Cognito::UserPoolRiskConfigurationAttachment</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Cognito::UserPoolUICustomizationAttachment</u>	 예	 예	 아니요
<u>AWS::Cognito::UserPoolUser</u>	 예	 아니요	 아니요
<u>AWS::Cognito::UserPoolUserToGroupAttachment</u>	 예	 예	 아니요
<u>AWS::Comprehend::DocumentClassifier</u>	 예	 예	 예
<u>AWS::Comprehend::Flywheel</u>	 예	 예	 예
<u>AWS::Config::AggregationAuthorization</u>	 예	 예	 예
<u>AWS::Config::ConfigRule</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Config::ConfigurationAggregator</u>	 예	 예	 예
<u>AWS::Config::ConformancePack</u>	 예	 예	 예
<u>AWS::Config::OrganizationConformance Pack</u>	 예	 예	 예
<u>AWS::Config::StoredQuery</u>	 예	 예	 예
<u>AWS::Connect::AgentStatus</u>	 예	 예	 예
<u>AWS::Connect::ApprovedOrigin</u>	 예	 예	 예
<u>AWS::Connect::ContactFlow</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Connect::ContactFlowModule</u>	 예	 예	 예
<u>AWS::Connect::ContactFlowVersion</u>	 예	 예	 예
<u>AWS::Connect::DataTable</u>	 예	 예	 아니요
<u>AWS::Connect::DataTableAttribute</u>	 예	 예	 아니요
<u>AWS::Connect::DataTableRecord</u>	 예	 예	 아니요
<u>AWS::Connect::EmailAddress</u>	 예	 예	 예
<u>AWS::Connect::EvaluationForm</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Connect::HoursOfOperation</u>	 예	 예	 예
<u>AWS::Connect::Instance</u>	 예	 예	 예
<u>AWS::Connect::InstanceStorageConfig</u>	 예	 예	 예
<u>AWS::Connect::IntegrationAssociation</u>	 예	 예	 예
<u>AWS::Connect::PhoneNumber</u>	 예	 예	 예
<u>AWS::Connect::PredefinedAttribute</u>	 예	 예	 예
<u>AWS::Connect::Prompt</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Connect::Queue</u>	 예	 예	 예
<u>AWS::Connect::QuickConnect</u>	 예	 예	 예
<u>AWS::Connect::RoutingProfile</u>	 예	 예	 예
<u>AWS::Connect::Rule</u>	 예	 예	 아니요
<u>AWS::Connect::SecurityKey</u>	 예	 예	 예
<u>AWS::Connect::SecurityProfile</u>	 예	 예	 예
<u>AWS::Connect::TaskTemplate</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Connect::TrafficDistributionGroup</u>	 예	 예	 예
<u>AWS::Connect::User</u>	 예	 예	 예
<u>AWS::Connect::UserHierarchyGroup</u>	 예	 예	 예
<u>AWS::Connect::UserHierarchyStructure</u>	 예	 예	 아니요
<u>AWS::Connect::View</u>	 예	 예	 예
<u>AWS::Connect::ViewVersion</u>	 예	 예	 예
<u>AWS::Connect::Workspace</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ConnectCampaigns::Campaign</u>	 예	 예	 예
<u>AWS::ConnectCampaignsV2::Campaign</u>	 예	 예	 예
<u>AWS::ControlTower::EnabledBaseline</u>	 예	 예	 아니요
<u>AWS::ControlTower::EnabledControl</u>	 예	 예	 예
<u>AWS::ControlTower::LandingZone</u>	 예	 예	 아니요
<u>AWS::CustomerProfiles::CalculatedAttributeDefinition</u>	 예	 예	 예
<u>AWS::CustomerProfiles::Domain</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::CustomerProfiles::EventStream</u>	 예	 예	 예
<u>AWS::CustomerProfiles::EventTrigger</u>	 예	 예	 아니요
<u>AWS::CustomerProfiles::Integration</u>	 예	 예	 예
<u>AWS::CustomerProfiles::ObjectType</u>	 예	 예	 예
<u>AWS::CustomerProfiles::SegmentDefinition</u>	 예	 예	 아니요
<u>AWS::DMS::DataMigration</u>	 예	 예	 아니요
<u>AWS::DMS::DataProvider</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::DMS::InstanceProfile</u>	 예	 예	 아니요
<u>AWS::DMS::MigrationProject</u>	 예	 예	 아니요
<u>AWS::DMS::ReplicationConfig</u>	 예	 예	 예
<u>AWS::DSQL::Cluster</u>	 예	 예	 아니요
<u>AWS::DataBrew::Dataset</u>	 예	 예	 예
<u>AWS::DataBrew::Job</u>	 예	 예	 예
<u>AWS::DataBrew::Project</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::DataBrew::Recipe</u>	 예	 예	 예
<u>AWS::DataBrew::Ruleset</u>	 예	 예	 예
<u>AWS::DataBrew::Schedule</u>	 예	 예	 예
<u>AWS::DataPipeline::Pipeline</u>	 예	 예	 예
<u>AWS::DataSync::Agent</u>	 예	 예	 예
<u>AWS::DataSync::LocationAzureBlob</u>	 예	 예	 예
<u>AWS::DataSync::LocationEFS</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::DataSync::LocationFSxLustre</u>	 예	 예	 예
<u>AWS::DataSync::LocationFSxONTAP</u>	 예	 예	 예
<u>AWS::DataSync::LocationFSxOpenZFS</u>	 예	 예	 예
<u>AWS::DataSync::LocationFSxWindows</u>	 예	 예	 예
<u>AWS::DataSync::LocationHDFS</u>	 예	 예	 예
<u>AWS::DataSync::LocationNFS</u>	 예	 예	 예
<u>AWS::DataSync::LocationObjectStorage</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::DataSync::LocationS3</u>	 예	 예	 예
<u>AWS::DataSync::LocationSMB</u>	 예	 예	 예
<u>AWS::DataSync::Task</u>	 예	 예	 예
<u>AWS::DataZone::Connection</u>	 예	 예	 아니요
<u>AWS::DataZone::DataSource</u>	 예	 예	 아니요
<u>AWS::DataZone::Domain</u>	 예	 예	 아니요
<u>AWS::DataZone::DomainUnit</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::DataZone::Environment</u>	 예	 예	 아니요
<u>AWS::DataZone::EnvironmentActions</u>	 예	 예	 아니요
<u>AWS::DataZone::EnvironmentBlueprintConfiguration</u>	 예	 예	 아니요
<u>AWS::DataZone::EnvironmentProfile</u>	 예	 예	 아니요
<u>AWS::DataZone::FormType</u>	 예	 예	 아니요
<u>AWS::DataZone::GroupProfile</u>	 예	 예	 아니요
<u>AWS::DataZone::Owner</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::DataZone::PolicyGrant</u>	 예	 예	 아니요
<u>AWS::DataZone::Project</u>	 예	 예	 아니요
<u>AWS::DataZone::ProjectMembership</u>	 예	 예	 아니요
<u>AWS::DataZone::ProjectProfile</u>	 예	 예	 아니요
<u>AWS::DataZone::SubscriptionTarget</u>	 예	 예	 아니요
<u>AWS::DataZone::UserProfile</u>	 예	 예	 아니요
<u>AWS::Deadline::Farm</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Deadline::Fleet</u>	 예	 예	 아니요
<u>AWS::Deadline::LicenseEndpoint</u>	 예	 예	 아니요
<u>AWS::Deadline::Limit</u>	 예	 예	 아니요
<u>AWS::Deadline::MeteredProduct</u>	 예	 예	 아니요
<u>AWS::Deadline::Monitor</u>	 예	 예	 아니요
<u>AWS::Deadline::Queue</u>	 예	 예	 아니요
<u>AWS::Deadline::QueueEnvironment</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Deadline::QueueFleetAssociation</u>	 예	 예	 아니요
<u>AWS::Deadline::QueueLimitAssociation</u>	 예	 예	 아니요
<u>AWS::Deadline::StorageProfile</u>	 예	 예	 아니요
<u>AWS::Detective::Graph</u>	 예	 예	 예
<u>AWS::Detective::MemberInvitation</u>	 예	 예	 예
<u>AWS::Detective::OrganizationAdmin</u>	 예	 예	 예
<u>AWS::DevOpsAgent::AgentSpace</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::DevOpsAgent::Association</u>	 예	 예	 아니요
<u>AWS::DevOpsGuru::LogAnomalyDetectionIntegration</u>	 예	 예	 예
<u>AWS::DevOpsGuru::NotificationChannel</u>	 예	 예	 예
<u>AWS::DevOpsGuru::ResourceCollection</u>	 예	 예	 예
<u>AWS::DeviceFarm::DevicePool</u>	 예	 예	 아니요
<u>AWS::DeviceFarm::InstanceProfile</u>	 예	 예	 아니요
<u>AWS::DeviceFarm::NetworkProfile</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::DeviceFarm::Project</u>	 예	 예	 아니요
<u>AWS::DeviceFarm::TestGridProject</u>	 예	 예	 아니요
<u>AWS::DeviceFarm::VPCEConfiguration</u>	 예	 예	 아니요
<u>AWS::DirectoryService::SimpleAD</u>	 예	 예	 예
<u>AWS::DocDBElastic::Cluster</u>	 예	 예	 예
<u>AWS::DynamoDB::GlobalTable</u>	 예	 예	 아니요
<u>AWS::DynamoDB::Table</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EC2::CapacityManagerDataExport</u>	 예	 예	 아니요
<u>AWS::EC2::CapacityReservation</u>	 예	 예	 예
<u>AWS::EC2::CapacityReservationFleet</u>	 예	 예	 예
<u>AWS::EC2::CarrierGateway</u>	 예	 예	 예
<u>AWS::EC2::CustomerGateway</u>	 예	 예	 예
<u>AWS::EC2::DHCPOptions</u>	 예	 예	 예
<u>AWS::EC2::EC2Fleet</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EC2::EIP</u>	 예	 예	 예
<u>AWS::EC2::EIPAssociation</u>	 예	 아니요	 예
<u>AWS::EC2::EgressOnlyInternetGateway</u>	 예	 예	 예
<u>AWS::EC2::EnclaveCertificateIamRoleAssociation</u>	 예	 예	 예
<u>AWS::EC2::FlowLog</u>	 예	 예	 예
<u>AWS::EC2::GatewayRouteTableAssociation</u>	 예	 예	 아니요
<u>AWS::EC2::Host</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EC2::IPAM</u>	 예	 예	 예
<u>AWS::EC2::IPAMAllocation</u>	 예	 예	 예
<u>AWS::EC2::IPAMPool</u>	 예	 예	 예
<u>AWS::EC2::IPAMPoolCidr</u>	 예	 예	 예
<u>AWS::EC2::IPAMResourceDiscovery</u>	 예	 예	 예
<u>AWS::EC2::IPAMResourceDiscoveryAssociation</u>	 예	 예	 예
<u>AWS::EC2::IPAMScope</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EC2::Instance</u>	 예	 예	 예
<u>AWS::EC2::InstanceConnectEndpoint</u>	 예	 예	 예
<u>AWS::EC2::InternetGateway</u>	 예	 예	 예
<u>AWS::EC2::IpPoolRouteTableAssociation</u>	 예	 예	 아니요
<u>AWS::EC2::KeyPair</u>	 예	 예	 예
<u>AWS::EC2::LaunchTemplate</u>	 예	 예	 아니요
<u>AWS::EC2::LocalGatewayRoute</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EC2::LocalGatewayRouteTable</u>	 예	 예	 예
<u>AWS::EC2::LocalGatewayRouteTableVPCAssociation</u>	 예	 예	 예
<u>AWS::EC2::LocalGatewayRouteTableVirtualInterfaceGroupAssociation</u>	 예	 예	 예
<u>AWS::EC2::LocalGatewayVirtualInterface</u>	 예	 예	 아니요
<u>AWS::EC2::LocalGatewayVirtualInterfaceGroup</u>	 예	 예	 아니요
<u>AWS::EC2::NatGateway</u>	 예	 예	 예
<u>AWS::EC2::NetworkAcl</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EC2::NetworkInsightsAccessScope</u>	 예	 예	 아니요
<u>AWS::EC2::NetworkInsightsAccessScopeAnalysis</u>	 예	 예	 예
<u>AWS::EC2::NetworkInsightsAnalysis</u>	 예	 예	 아니요
<u>AWS::EC2::NetworkInsightsPath</u>	 예	 예	 예
<u>AWS::EC2::NetworkInterface</u>	 예	 예	 예
<u>AWS::EC2::NetworkInterfaceAttachment</u>	 예	 예	 예
<u>AWS::EC2::NetworkPerformanceMetricSubscription</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EC2::PlacementGroup</u>	 예	 예	 예
<u>AWS::EC2::PrefixList</u>	 예	 예	 아니요
<u>AWS::EC2::Route</u>	 예	 예	 예
<u>AWS::EC2::RouteServer</u>	 예	 예	 아니요
<u>AWS::EC2::RouteServerAssociation</u>	 예	 예	 아니요
<u>AWS::EC2::RouteServerEndpoint</u>	 예	 예	 아니요
<u>AWS::EC2::RouteServerPeer</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EC2::RouteServerPropagation</u>	 예	 예	 아니요
<u>AWS::EC2::RouteTable</u>	 예	 예	 예
<u>AWS::EC2::SecurityGroup</u>	 예	 예	 예
<u>AWS::EC2::SecurityGroupEgress</u>	 예	 예	 아니요
<u>AWS::EC2::SecurityGroupIngress</u>	 예	 예	 아니요
<u>AWS::EC2::SecurityGroupVpcAssociation</u>	 예	 예	 아니요
<u>AWS::EC2::SnapshotBlockPublicAccess</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EC2::SpotFleet</u>	 예	 예	 아니요
<u>AWS::EC2::Subnet</u>	 예	 예	 예
<u>AWS::EC2::SubnetCidrBlock</u>	 예	 예	 예
<u>AWS::EC2::SubnetNetworkAclAssociation</u>	 예	 예	 예
<u>AWS::EC2::SubnetRouteTableAssociation</u>	 예	 아니요	 예
<u>AWS::EC2::TrafficMirrorFilter</u>	 예	 예	 아니요
<u>AWS::EC2::TrafficMirrorFilterRule</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EC2::TrafficMirrorSession</u>	 예	 예	 아니요
<u>AWS::EC2::TrafficMirrorTarget</u>	 예	 예	 아니요
<u>AWS::EC2::TransitGateway</u>	 예	 예	 예
<u>AWS::EC2::TransitGatewayAttachment</u>	 예	 예	 예
<u>AWS::EC2::TransitGatewayConnect</u>	 예	 예	 예
<u>AWS::EC2::TransitGatewayConnectPeer</u>	 예	 예	 아니요
<u>AWS::EC2::TransitGatewayMeteringPolicy</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EC2::TransitGatewayMeteringPolicyEntry</u>	 예	 예	 아니요
<u>AWS::EC2::TransitGatewayMulticastDomain</u>	 예	 예	 예
<u>AWS::EC2::TransitGatewayMulticastDomainAssociation</u>	 예	 예	 예
<u>AWS::EC2::TransitGatewayMulticastGroupMember</u>	 예	 예	 예
<u>AWS::EC2::TransitGatewayMulticastGroupSource</u>	 예	 예	 예
<u>AWS::EC2::TransitGatewayPeeringAttachment</u>	 예	 예	 예
<u>AWS::EC2::TransitGatewayRoute</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EC2::TransitGatewayRouteTable</u>	 예	 예	 예
<u>AWS::EC2::TransitGatewayRouteTableAssociation</u>	 예	 예	 아니요
<u>AWS::EC2::TransitGatewayRouteTablePropagation</u>	 예	 예	 아니요
<u>AWS::EC2::TransitGatewayVpcAttachment</u>	 예	 예	 예
<u>AWS::EC2::VPC</u>	 예	 예	 예
<u>AWS::EC2::VPCBlockPublicAccessExclusion</u>	 예	 예	 아니요
<u>AWS::EC2::VPCBlockPublicAccessOptions</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EC2::VPCCidrBlock</u>	 예	 예	 아니요
<u>AWS::EC2::VPCDHCPOptionsAssociation</u>	 예	 예	 예
<u>AWS::EC2::VPCEncryptionControl</u>	 예	 예	 아니요
<u>AWS::EC2::VPCEndpoint</u>	 예	 예	 예
<u>AWS::EC2::VPCEndpointConnectionNotification</u>	 예	 예	 예
<u>AWS::EC2::VPCEndpointService</u>	 예	 예	 예
<u>AWS::EC2::VPCEndpointServicePermissions</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EC2::VPCGatewayAttachment</u>	 예	 예	 아니요
<u>AWS::EC2::VPCPeeringConnection</u>	 예	 예	 예
<u>AWS::EC2::VPNConcentrator</u>	 예	 예	 아니요
<u>AWS::EC2::VPNConnection</u>	 예	 예	 예
<u>AWS::EC2::VPNConnectionRoute</u>	 예	 예	 예
<u>AWS::EC2::VPNGateway</u>	 예	 예	 예
<u>AWS::EC2::VerifiedAccessEndpoint</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EC2::VerifiedAccessGroup</u>	 예	 예	 예
<u>AWS::EC2::VerifiedAccessInstance</u>	 예	 예	 예
<u>AWS::EC2::VerifiedAccessTrustProvider</u>	 예	 예	 예
<u>AWS::EC2::Volume</u>	 예	 예	 예
<u>AWS::EC2::VolumeAttachment</u>	 예	 예	 예
<u>AWS::ECR::PublicRepository</u>	 예	 예	 예
<u>AWS::ECR::PullThroughCacheRule</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ECR::PullTimeUpdateExclusion</u>	 예	 예	 아니요
<u>AWS::ECR::RegistryPolicy</u>	 예	 예	 예
<u>AWS::ECR::RegistryScanningConfiguration</u>	 예	 예	 아니요
<u>AWS::ECR::ReplicationConfiguration</u>	 예	 예	 예
<u>AWS::ECR::Repository</u>	 예	 예	 예
<u>AWS::ECR::RepositoryCreationTemplate</u>	 예	 예	 아니요
<u>AWS::ECR::SigningConfiguration</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ECS::CapacityProvider</u>	 예	 예	 아니요
<u>AWS::ECS::Cluster</u>	 예	 아니요	 예
<u>AWS::ECS::ClusterCapacityProviderAssociations</u>	 예	 아니요	 예
<u>AWS::ECS::ExpressGatewayService</u>	 예	 예	 아니요
<u>AWS::ECS::PrimaryTaskSet</u>	 예	 예	 아니요
<u>AWS::ECS::Service</u>	 예	 예	 예
<u>AWS::ECS::TaskDefinition</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
AWS::ECS::TaskSet	 예	 예	 아니요
AWS::EFS::AccessPoint	 예	 예	 예
AWS::EFS::FileSystem	 예	 예	 예
AWS::EFS::MountTarget	 예	 예	 예
AWS::EKS::AccessEntry	 예	 예	 아니요
AWS::EKS::Addon	 예	 예	 예
AWS::EKS::Capability	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EKS::Cluster</u>	 예	 아니요	 예
<u>AWS::EKS::FargateProfile</u>	 예	 예	 예
<u>AWS::EKS::IdentityProviderConfig</u>	 예	 예	 예
<u>AWS::EKS::Nodegroup</u>	 예	 아니요	 예
<u>AWS::EKS::PodIdentityAssociation</u>	 예	 예	 아니요
<u>AWS::EMR::SecurityConfiguration</u>	 예	 예	 예
<u>AWS::EMR::Step</u>	 예	 아니요	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EMR::Studio</u>	 예	 예	 예
<u>AWS::EMR::StudioSessionMapping</u>	 예	 예	 예
<u>AWS::EMR::WALWorkspace</u>	 예	 예	 아니요
<u>AWS::EMRContainers::VirtualCluster</u>	 예	 예	 아니요
<u>AWS::EMRServerless::Application</u>	 예	 예	 예
<u>AWS::EVS::Environment</u>	 예	 예	 아니요
<u>AWS::ElastiCache::GlobalReplicationGroup</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ElastiCache::ParameterGroup</u>	 예	 예	 아니요
<u>AWS::ElastiCache::ServerlessCache</u>	 예	 예	 아니요
<u>AWS::ElastiCache::SubnetGroup</u>	 예	 예	 예
<u>AWS::ElastiCache::User</u>	 예	 예	 예
<u>AWS::ElastiCache::UserGroup</u>	 예	 예	 예
<u>AWS::ElasticBeanstalk::Application</u>	 예	 예	 예
<u>AWS::ElasticBeanstalk::ApplicationVersion</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
AWS::ElasticBeanstalk::ConfigurationTemplate	 예	 예	 예
AWS::ElasticBeanstalk::Environment	 예	 아니요	 아니요
AWS::ElasticLoadBalancingV2::Listener	 예	 예	 예
AWS::ElasticLoadBalancingV2::ListenerRule	 예	 아니요	 예
AWS::ElasticLoadBalancingV2::LoadBalancer	 예	 예	 예
AWS::ElasticLoadBalancingV2::TargetGroup	 예	 예	 예
AWS::ElasticLoadBalancingV2::TrustStore	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ElasticLoadBalancingV2::TrustStoreRevocation</u>	 예	 예	 아니요
<u>AWS::EntityResolution::IdMappingWorkflow</u>	 예	 예	 예
<u>AWS::EntityResolution::IdNamespace</u>	 예	 예	 아니요
<u>AWS::EntityResolution::MatchingWorkflow</u>	 예	 예	 예
<u>AWS::EntityResolution::PolicyStatement</u>	 예	 예	 아니요
<u>AWS::EntityResolution::SchemaMapping</u>	 예	 예	 예
<u>AWS::EventSchemas::Discoverer</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::EventSchemas::Registry</u>	 예	 예	 아니요
<u>AWS::EventSchemas::RegistryPolicy</u>	 예	 예	 아니요
<u>AWS::EventSchemas::Schema</u>	 예	 예	 아니요
<u>AWS::Events::ApiDestination</u>	 예	 예	 예
<u>AWS::Events::Archive</u>	 예	 예	 예
<u>AWS::Events::Connection</u>	 예	 예	 아니요
<u>AWS::Events::Endpoint</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Events::EventBus</u>	 예	 예	 예
<u>AWS::Events::EventBusPolicy</u>	 예	 예	 아니요
<u>AWS::Events::Rule</u>	 예	 예	 예
<u>AWS::Evidently::Experiment</u>	 예	 예	 아니요
<u>AWS::Evidently::Feature</u>	 예	 예	 아니요
<u>AWS::Evidently::Launch</u>	 예	 예	 아니요
<u>AWS::Evidently::Project</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Evidently::Segment</u>	 예	 예	 아니요
<u>AWS::FIS::ExperimentTemplate</u>	 예	 예	 예
<u>AWS::FIS::TargetAccountConfiguration</u>	 예	 예	 아니요
<u>AWS::FMS::NotificationChannel</u>	 예	 예	 아니요
<u>AWS::FMS::Policy</u>	 예	 아니요	 아니요
<u>AWS::FMS::ResourceSet</u>	 예	 예	 아니요
<u>AWS::FSx::DataRepositoryAssociation</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::FSx::S3AccessPointAttachment</u>	 예	 예	 아니요
<u>AWS::FinSpace::Environment</u>	 예	 예	 예
<u>AWS::Forecast::Dataset</u>	 예	 예	 아니요
<u>AWS::Forecast::DatasetGroup</u>	 예	 예	 아니요
<u>AWS::FraudDetector::Detector</u>	 예	 예	 예
<u>AWS::FraudDetector::EntityType</u>	 예	 예	 예
<u>AWS::FraudDetector::EventType</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::FraudDetector::Label</u>	 예	 예	 예
<u>AWS::FraudDetector::List</u>	 예	 예	 예
<u>AWS::FraudDetector::Outcome</u>	 예	 예	 예
<u>AWS::FraudDetector::Variable</u>	 예	 예	 예
<u>AWS::GameLift::Alias</u>	 예	 예	 예
<u>AWS::GameLift::Build</u>	 예	 예	 예
<u>AWS::GameLift::ContainerFleet</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::GameLift::ContainerGroupDefinition</u>	 예	 예	 아니요
<u>AWS::GameLift::Fleet</u>	 예	 예	 예
<u>AWS::GameLift::GameServerGroup</u>	 예	 예	 아니요
<u>AWS::GameLift::GameSessionQueue</u>	 예	 예	 아니요
<u>AWS::GameLift::Location</u>	 예	 예	 아니요
<u>AWS::GameLift::MatchmakingConfiguration</u>	 예	 예	 아니요
<u>AWS::GameLift::MatchmakingRuleSet</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::GameLift::Script</u>	 예	 예	 아니요
<u>AWS::GameLiftStreams::Application</u>	 예	 예	 아니요
<u>AWS::GameLiftStreams::StreamGroup</u>	 예	 예	 아니요
<u>AWS::GlobalAccelerator::Accelerator</u>	 예	 예	 예
<u>AWS::GlobalAccelerator::CrossAccount Attachment</u>	 예	 예	 아니요
<u>AWS::GlobalAccelerator::EndpointGroup</u>	 예	 예	 예
<u>AWS::GlobalAccelerator::Listener</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Glue::Crawler</u>	 예	 예	 아니요
<u>AWS::Glue::Database</u>	 예	 예	 아니요
<u>AWS::Glue::IdentityCenterConfiguration</u>	 예	 예	 아니요
<u>AWS::Glue::Integration</u>	 예	 예	 아니요
<u>AWS::Glue::IntegrationResourceProperty</u>	 예	 예	 아니요
<u>AWS::Glue::Job</u>	 예	 예	 아니요
<u>AWS::Glue::Registry</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Glue::Schema</u>	 예	 예	 예
<u>AWS::Glue::SchemaVersion</u>	 예	 예	 예
<u>AWS::Glue::SchemaVersionMetadata</u>	 예	 예	 예
<u>AWS::Glue::Trigger</u>	 예	 예	 아니요
<u>AWS::Glue::UsageProfile</u>	 예	 예	 아니요
<u>AWS::Grafana::Workspace</u>	 예	 예	 예
<u>AWS::GreengrassV2::ComponentVersion</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::GreengrassV2::Deployment</u>	 예	 예	 예
<u>AWS::GroundStation::Config</u>	 예	 예	 예
<u>AWS::GroundStation::DataflowEndpointGroup</u>	 예	 예	 예
<u>AWS::GroundStation::DataflowEndpointGroupV2</u>	 예	 예	 아니요
<u>AWS::GroundStation::MissionProfile</u>	 예	 예	 예
<u>AWS::GuardDuty::Detector</u>	 예	 예	 예
<u>AWS::GuardDuty::Filter</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::GuardDuty::IPSet</u>	 예	 예	 아니요
<u>AWS::GuardDuty::MalwareProtectionPlan</u>	 예	 예	 아니요
<u>AWS::GuardDuty::Master</u>	 예	 예	 아니요
<u>AWS::GuardDuty::Member</u>	 예	 예	 아니요
<u>AWS::GuardDuty::PublishingDestination</u>	 예	 예	 아니요
<u>AWS::GuardDuty::ThreatEntitySet</u>	 예	 예	 아니요
<u>AWS::GuardDuty::ThreatIntelSet</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::GuardDuty::TrustedEntitySet</u>	 예	 예	 아니요
<u>AWS::HealthImaging::Datastore</u>	 예	 예	 예
<u>AWS::HealthLake::FHIRDatastore</u>	 예	 예	 예
<u>AWS::IAM::Group</u>	 예	 예	 예
<u>AWS::IAM::GroupPolicy</u>	 예	 예	 아니요
<u>AWS::IAM::InstanceProfile</u>	 예	 예	 예
<u>AWS::IAM::ManagedPolicy</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::IAM::OIDCProvider</u>	 예	 예	 예
<u>AWS::IAM::Role</u>	 예	 예	 예
<u>AWS::IAM::RolePolicy</u>	 예	 예	 아니요
<u>AWS::IAM::SAMLProvider</u>	 예	 예	 예
<u>AWS::IAM::ServerCertificate</u>	 예	 예	 예
<u>AWS::IAM::ServiceLinkedRole</u>	 예	 예	 아니요
<u>AWS::IAM::User</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::IAM::UserPolicy</u>	 예	 예	 아니요
<u>AWS::IAM::VirtualMFADevice</u>	 예	 예	 아니요
<u>AWS::IVS::Channel</u>	 예	 예	 예
<u>AWS::IVS::EncoderConfiguration</u>	 예	 예	 아니요
<u>AWS::IVS::IngestConfiguration</u>	 예	 예	 아니요
<u>AWS::IVS::PlaybackKeyPair</u>	 예	 예	 예
<u>AWS::IVS::PlaybackRestrictionPolicy</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::IVS::PublicKey</u>	 예	 예	 아니요
<u>AWS::IVS::RecordingConfiguration</u>	 예	 예	 예
<u>AWS::IVS::Stage</u>	 예	 예	 아니요
<u>AWS::IVS::StorageConfiguration</u>	 예	 예	 아니요
<u>AWS::IVS::StreamKey</u>	 예	 예	 아니요
<u>AWS::IVSChat::LoggingConfiguration</u>	 예	 예	 예
<u>AWS::IVSChat::Room</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::IdentityStore::Group</u>	 예	 예	 예
<u>AWS::IdentityStore::GroupMembership</u>	 예	 예	 예
<u>AWS::ImageBuilder::Component</u>	 예	 예	 예
<u>AWS::ImageBuilder::ContainerRecipe</u>	 예	 예	 아니요
<u>AWS::ImageBuilder::DistributionConfiguration</u>	 예	 예	 예
<u>AWS::ImageBuilder::Image</u>	 예	 예	 예
<u>AWS::ImageBuilder::ImagePipeline</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ImageBuilder::ImageRecipe</u>	 예	 예	 예
<u>AWS::ImageBuilder::InfrastructureConfiguration</u>	 예	 예	 예
<u>AWS::ImageBuilder::LifecyclePolicy</u>	 예	 예	 아니요
<u>AWS::ImageBuilder::Workflow</u>	 예	 예	 아니요
<u>AWS::Inspector::AssessmentTarget</u>	 예	 예	 예
<u>AWS::Inspector::AssessmentTemplate</u>	 예	 예	 예
<u>AWS::Inspector::ResourceGroup</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::InspectorV2::CisScanConfiguration</u>	 예	 예	 아니요
<u>AWS::InspectorV2::CodeSecurityIntegration</u>	 예	 예	 아니요
<u>AWS::InspectorV2::CodeSecurityScanConfiguration</u>	 예	 예	 아니요
<u>AWS::InspectorV2::Filter</u>	 예	 예	 아니요
<u>AWS::InternetMonitor::Monitor</u>	 예	 예	 예
<u>AWS::Invoicing::InvoiceUnit</u>	 예	 예	 아니요
<u>AWS::IoT::AccountAuditConfiguration</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::IoT::Authorizer</u>	 예	 예	 예
<u>AWS::IoT::BillingGroup</u>	 예	 예	 예
<u>AWS::IoT::CACertificate</u>	 예	 예	 예
<u>AWS::IoT::Certificate</u>	 예	 예	 예
<u>AWS::IoT::CertificateProvider</u>	 예	 예	 아니요
<u>AWS::IoT::Command</u>	 예	 예	 아니요
<u>AWS::IoT::CustomMetric</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::IoT::Dimension</u>	 예	 예	 예
<u>AWS::IoT::DomainConfiguration</u>	 예	 예	 아니요
<u>AWS::IoT::EncryptionConfiguration</u>	 예	 예	 아니요
<u>AWS::IoT::FleetMetric</u>	 예	 예	 아니요
<u>AWS::IoT::JobTemplate</u>	 예	 예	 아니요
<u>AWS::IoT::Logging</u>	 예	 예	 예
<u>AWS::IoT::MitigationAction</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::IoT::Policy</u>	 예	 예	 예
<u>AWS::IoT::ProvisioningTemplate</u>	 예	 예	 예
<u>AWS::IoT::ResourceSpecificLogging</u>	 예	 예	 예
<u>AWS::IoT::RoleAlias</u>	 예	 예	 예
<u>AWS::IoT::ScheduledAudit</u>	 예	 예	 예
<u>AWS::IoT::SecurityProfile</u>	 예	 예	 예
<u>AWS::IoT::SoftwarePackage</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::IoT::SoftwarePackageVersion</u>	 예	 예	 아니요
<u>AWS::IoT::Thing</u>	 예	 예	 예
<u>AWS::IoT::ThingGroup</u>	 예	 예	 예
<u>AWS::IoT::ThingType</u>	 예	 예	 아니요
<u>AWS::IoT::TopicRule</u>	 예	 예	 예
<u>AWS::IoT::TopicRuleDestination</u>	 예	 예	 예
<u>AWS::IoTAnalytics::Channel</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::IoTAnalytics::Dataset</u>	 예	 예	 예
<u>AWS::IoTAnalytics::Datastore</u>	 예	 예	 예
<u>AWS::IoTAnalytics::Pipeline</u>	 예	 예	 예
<u>AWS::IoTCoreDeviceAdvisor::SuiteDefinition</u>	 예	 예	 예
<u>AWS::IoTEvents::AlarmModel</u>	 예	 예	 예
<u>AWS::IoTEvents::DetectorModel</u>	 예	 예	 예
<u>AWS::IoTEvents::Input</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::IoT FleetWise::Campaign</u>	 예	 예	 예
<u>AWS::IoT FleetWise::DecoderManifest</u>	 예	 예	 예
<u>AWS::IoT FleetWise::Fleet</u>	 예	 예	 예
<u>AWS::IoT FleetWise::ModelManifest</u>	 예	 예	 예
<u>AWS::IoT FleetWise::SignalCatalog</u>	 예	 예	 예
<u>AWS::IoT FleetWise::StateTemplate</u>	 예	 예	 아니요
<u>AWS::IoT FleetWise::Vehicle</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::IoTSiteWise::AccessPolicy</u>	 예	 예	 아니요
<u>AWS::IoTSiteWise::Asset</u>	 예	 예	 예
<u>AWS::IoTSiteWise::AssetModel</u>	 예	 예	 예
<u>AWS::IoTSiteWise::ComputationModel</u>	 예	 예	 아니요
<u>AWS::IoTSiteWise::Dashboard</u>	 예	 예	 아니요
<u>AWS::IoTSiteWise::Dataset</u>	 예	 예	 아니요
<u>AWS::IoTSiteWise::Gateway</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::IoTSiteWise::Portal</u>	 예	 예	 아니요
<u>AWS::IoTSiteWise::Project</u>	 예	 예	 아니요
<u>AWS::IoTTwinMaker::ComponentType</u>	 예	 예	 예
<u>AWS::IoTTwinMaker::Entity</u>	 예	 예	 예
<u>AWS::IoTTwinMaker::Scene</u>	 예	 예	 예
<u>AWS::IoTTwinMaker::SyncJob</u>	 예	 예	 예
<u>AWS::IoTTwinMaker::Workspace</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::IoTWireless::Destination</u>	 예	 예	 예
<u>AWS::IoTWireless::DeviceProfile</u>	 예	 예	 예
<u>AWS::IoTWireless::FirmwareTask</u>	 예	 예	 예
<u>AWS::IoTWireless::MulticastGroup</u>	 예	 예	 예
<u>AWS::IoTWireless::NetworkAnalyzerConfiguration</u>	 예	 예	 예
<u>AWS::IoTWireless::PartnerAccount</u>	 예	 예	 예
<u>AWS::IoTWireless::ServiceProfile</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::IoTWireless::TaskDefinition</u>	 예	 예	 예
<u>AWS::IoTWireless::WirelessDevice</u>	 예	 예	 예
<u>AWS::IoTWireless::WirelessDeviceImportTask</u>	 예	 예	 예
<u>AWS::IoTWireless::WirelessGateway</u>	 예	 예	 예
<u>AWS::KMS::Alias</u>	 예	 아니요	 예
<u>AWS::KMS::Key</u>	 예	 아니요	 예
<u>AWS::KMS::ReplicaKey</u>	 예	 아니요	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::KafkaConnect::Connector</u>	 예	 아니요	 아니요
<u>AWS::KafkaConnect::CustomPlugin</u>	 예	 예	 아니요
<u>AWS::KafkaConnect::WorkerConfiguration</u>	 예	 예	 아니요
<u>AWS::Kendra::DataSource</u>	 예	 예	 예
<u>AWS::Kendra::FAQ</u>	 예	 예	 예
<u>AWS::Kendra::Index</u>	 예	 예	 예
<u>AWS::KendraRanking::ExecutionPlan</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Kinesis::ResourcePolicy</u>	 예	 예	 아니요
<u>AWS::Kinesis::Stream</u>	 예	 예	 예
<u>AWS::Kinesis::StreamConsumer</u>	 예	 예	 아니요
<u>AWS::KinesisAnalyticsV2::Application</u>	 예	 예	 아니요
<u>AWS::KinesisFirehose::DeliveryStream</u>	 예	 아니요	 아니요
<u>AWS::KinesisVideo::SignalingChannel</u>	 예	 예	 아니요
<u>AWS::KinesisVideo::Stream</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::LakeFormation::DataCellsFilter</u>	 예	 예	 아니요
<u>AWS::LakeFormation::PrincipalPermissions</u>	 예	 예	 아니요
<u>AWS::LakeFormation::Tag</u>	 예	 예	 아니요
<u>AWS::LakeFormation::TagAssociation</u>	 예	 예	 아니요
<u>AWS::Lambda::Alias</u>	 예	 예	 아니요
<u>AWS::Lambda::CapacityProvider</u>	 예	 예	 아니요
<u>AWS::Lambda::CodeSigningConfig</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Lambda::EventInvokeConfig</u>	 예	 예	 아니요
<u>AWS::Lambda::EventSourceMapping</u>	 예	 예	 예
<u>AWS::Lambda::Function</u>	 예	 예	 예
<u>AWS::Lambda::LayerVersion</u>	 예	 예	 아니요
<u>AWS::Lambda::LayerVersionPermission</u>	 예	 예	 예
<u>AWS::Lambda::Permission</u>	 예	 예	 예
<u>AWS::Lambda::Url</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Lambda::Version</u>	 예	 예	 예
<u>AWS::LaunchWizard::Deployment</u>	 예	 예	 아니요
<u>AWS::Lex::Bot</u>	 예	 예	 아니요
<u>AWS::Lex::BotAlias</u>	 예	 예	 아니요
<u>AWS::Lex::BotVersion</u>	 예	 예	 아니요
<u>AWS::Lex::ResourcePolicy</u>	 예	 예	 아니요
<u>AWS::LicenseManager::Grant</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::LicenseManager::License</u>	 예	 예	 아니요
<u>AWS::Lightsail::Alarm</u>	 예	 예	 예
<u>AWS::Lightsail::Bucket</u>	 예	 예	 예
<u>AWS::Lightsail::Certificate</u>	 예	 예	 예
<u>AWS::Lightsail::Container</u>	 예	 예	 예
<u>AWS::Lightsail::Database</u>	 예	 예	 아니요
<u>AWS::Lightsail::Disk</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Lightsail::DiskSnapshot</u>	 예	 예	 아니요
<u>AWS::Lightsail::Distribution</u>	 예	 예	 예
<u>AWS::Lightsail::Domain</u>	 예	 예	 아니요
<u>AWS::Lightsail::Instance</u>	 예	 예	 예
<u>AWS::Lightsail::InstanceSnapshot</u>	 예	 예	 아니요
<u>AWS::Lightsail::LoadBalancer</u>	 예	 예	 예
<u>AWS::Lightsail::LoadBalancerTlsCertificate</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Lightsail::StaticIp</u>	 예	 예	 예
<u>AWS::Location::APIKey</u>	 예	 예	 아니요
<u>AWS::Location::GeofenceCollection</u>	 예	 예	 예
<u>AWS::Location::Map</u>	 예	 예	 예
<u>AWS::Location::PlaceIndex</u>	 예	 예	 예
<u>AWS::Location::RouteCalculator</u>	 예	 예	 예
<u>AWS::Location::Tracker</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Location::TrackerConsumer</u>	 예	 예	 예
<u>AWS::Logs::AccountPolicy</u>	 예	 예	 예
<u>AWS::Logs::Delivery</u>	 예	 예	 아니요
<u>AWS::Logs::DeliveryDestination</u>	 예	 예	 아니요
<u>AWS::Logs::DeliverySource</u>	 예	 예	 아니요
<u>AWS::Logs::Destination</u>	 예	 예	 예
<u>AWS::Logs::Integration</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Logs::LogAnomalyDetector</u>	 예	 예	 아니요
<u>AWS::Logs::LogGroup</u>	 예	 예	 예
<u>AWS::Logs::LogStream</u>	 예	 예	 예
<u>AWS::Logs::MetricFilter</u>	 예	 예	 예
<u>AWS::Logs::QueryDefinition</u>	 예	 예	 아니요
<u>AWS::Logs::ResourcePolicy</u>	 예	 예	 예
<u>AWS::Logs::SubscriptionFilter</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Logs::Transformer</u>	 예	 예	 아니요
<u>AWS::LookoutEquipment::InferenceScheduler</u>	 예	 예	 아니요
<u>AWS::LookoutVision::Project</u>	 예	 예	 예
<u>AWS::M2::Application</u>	 예	 예	 아니요
<u>AWS::M2::Deployment</u>	 예	 예	 아니요
<u>AWS::M2::Environment</u>	 예	 예	 예
<u>AWS::MPA::ApprovalTeam</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::MPA::IdentitySource</u>	 예	 예	 아니요
<u>AWS::MSK::BatchScramSecret</u>	 예	 예	 예
<u>AWS::MSK::Cluster</u>	 예	 예	 예
<u>AWS::MSK::ClusterPolicy</u>	 예	 예	 예
<u>AWS::MSK::Configuration</u>	 예	 예	 예
<u>AWS::MSK::Replicator</u>	 예	 예	 예
<u>AWS::MSK::ServerlessCluster</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::MSK::VpcConnection</u>	 예	 예	 예
<u>AWS::MWAA::Environment</u>	 예	 예	 아니요
<u>AWS::Macie::AllowList</u>	 예	 예	 아니요
<u>AWS::Macie::CustomDataIdentifier</u>	 예	 예	 아니요
<u>AWS::Macie::FindingsFilter</u>	 예	 예	 아니요
<u>AWS::Macie::Session</u>	 예	 예	 예
<u>AWS::ManagedBlockchain::Accessor</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::MediaConnect::Bridge</u>	 예	 예	 예
<u>AWS::MediaConnect::BridgeOutput</u>	 예	 예	 아니요
<u>AWS::MediaConnect::BridgeSource</u>	 예	 예	 아니요
<u>AWS::MediaConnect::Flow</u>	 예	 예	 아니요
<u>AWS::MediaConnect::FlowEntitlement</u>	 예	 예	 아니요
<u>AWS::MediaConnect::FlowOutput</u>	 예	 예	 아니요
<u>AWS::MediaConnect::FlowSource</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::MediaConnect::FlowVpcInterface</u>	 예	 예	 예
<u>AWS::MediaConnect::Gateway</u>	 예	 예	 예
<u>AWS::MediaConnect::RouterInput</u>	 예	 예	 아니요
<u>AWS::MediaConnect::RouterNetworkInterface</u>	 예	 예	 아니요
<u>AWS::MediaConnect::RouterOutput</u>	 예	 예	 아니요
<u>AWS::MediaLive::ChannelPlacementGroup</u>	 예	 예	 아니요
<u>AWS::MediaLive::CloudWatchAlarmTemplate</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::MediaLive::CloudWatchAlarmTemplateGroup</u>	 예	 예	 아니요
<u>AWS::MediaLive::Cluster</u>	 예	 예	 아니요
<u>AWS::MediaLive::EventBridgeRuleTemplate</u>	 예	 예	 아니요
<u>AWS::MediaLive::EventBridgeRuleTemplateGroup</u>	 예	 예	 아니요
<u>AWS::MediaLive::Multiplex</u>	 예	 예	 아니요
<u>AWS::MediaLive::Multiplexprogram</u>	 예	 예	 아니요
<u>AWS::MediaLive::Network</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::MediaLive::SdiSource</u>	 예	 예	 아니요
<u>AWS::MediaLive::SignalMap</u>	 예	 예	 아니요
<u>AWS::MediaPackage::Asset</u>	 예	 예	 예
<u>AWS::MediaPackage::Channel</u>	 예	 예	 예
<u>AWS::MediaPackage::OriginEndpoint</u>	 예	 예	 예
<u>AWS::MediaPackage::PackagingConfiguration</u>	 예	 예	 예
<u>AWS::MediaPackage::PackagingGroup</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::MediaPackageV2::Channel</u>	 예	 예	 예
<u>AWS::MediaPackageV2::ChannelGroup</u>	 예	 예	 예
<u>AWS::MediaPackageV2::ChannelPolicy</u>	 예	 예	 아니요
<u>AWS::MediaPackageV2::OriginEndpoint</u>	 예	 예	 예
<u>AWS::MediaPackageV2::OriginEndpointPolicy</u>	 예	 예	 아니요
<u>AWS::MediaTailor::Channel</u>	 예	 예	 아니요
<u>AWS::MediaTailor::ChannelPolicy</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::MediaTailor::LiveSource</u>	 예	 예	 예
<u>AWS::MediaTailor::PlaybackConfiguration</u>	 예	 예	 아니요
<u>AWS::MediaTailor::SourceLocation</u>	 예	 예	 예
<u>AWS::MediaTailor::VodSource</u>	 예	 예	 예
<u>AWS::MemoryDB::ACL</u>	 예	 예	 예
<u>AWS::MemoryDB::Cluster</u>	 예	 예	 예
<u>AWS::MemoryDB::MultiRegionCluster</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::MemoryDB::ParameterGroup</u>	 예	 예	 예
<u>AWS::MemoryDB::SubnetGroup</u>	 예	 예	 예
<u>AWS::MemoryDB::User</u>	 예	 예	 예
<u>AWS::Neptune::DBCluster</u>	 예	 예	 예
<u>AWS::Neptune::DBClusterParameterGroup</u>	 예	 예	 아니요
<u>AWS::Neptune::DBInstance</u>	 예	 예	 아니요
<u>AWS::Neptune::DBParameterGroup</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Neptune::DBSubnetGroup</u>	 예	 예	 아니요
<u>AWS::Neptune::EventSubscription</u>	 예	 예	 아니요
<u>AWS::NeptuneGraph::Graph</u>	 예	 예	 아니요
<u>AWS::NeptuneGraph::PrivateGraphEndpoint</u>	 예	 예	 아니요
<u>AWS::NetworkFirewall::Firewall</u>	 예	 예	 예
<u>AWS::NetworkFirewall::FirewallPolicy</u>	 예	 예	 예
<u>AWS::NetworkFirewall::LoggingConfiguration</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::NetworkFirewall::RuleGroup</u>	 예	 예	 예
<u>AWS::NetworkFirewall::TLSInspectionConfiguration</u>	 예	 예	 아니요
<u>AWS::NetworkFirewall::VpcEndpointAssociation</u>	 예	 예	 아니요
<u>AWS::NetworkManager::ConnectAttachment</u>	 예	 예	 예
<u>AWS::NetworkManager::ConnectPeer</u>	 예	 예	 예
<u>AWS::NetworkManager::CoreNetwork</u>	 예	 예	 예
<u>AWS::NetworkManager::CoreNetworkPrefixListAssociation</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::NetworkManager::CustomerGatewayAssociation</u>	 예	 예	 예
<u>AWS::NetworkManager::Device</u>	 예	 예	 예
<u>AWS::NetworkManager::DirectConnectGatewayAttachment</u>	 예	 예	 아니요
<u>AWS::NetworkManager::GlobalNetwork</u>	 예	 예	 예
<u>AWS::NetworkManager::Link</u>	 예	 예	 예
<u>AWS::NetworkManager::LinkAssociation</u>	 예	 예	 예
<u>AWS::NetworkManager::Site</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::NetworkManager::SiteToSiteVpnAttachment</u>	 예	 예	 예
<u>AWS::NetworkManager::TransitGatewayPeering</u>	 예	 예	 예
<u>AWS::NetworkManager::TransitGatewayRegistration</u>	 예	 예	 예
<u>AWS::NetworkManager::TransitGatewayRouteTableAttachment</u>	 예	 예	 예
<u>AWS::NetworkManager::VpcAttachment</u>	 예	 예	 예
<u>AWS::Notifications::ChannelAssociation</u>	 예	 예	 아니요
<u>AWS::Notifications::EventRule</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Notifications::ManagedNotificationAccountContactAssociation</u>	 예	 예	 아니요
<u>AWS::Notifications::ManagedNotificationAdditionalChannelAssociation</u>	 예	 예	 아니요
<u>AWS::Notifications::NotificationConfiguration</u>	 예	 예	 아니요
<u>AWS::Notifications::NotificationHub</u>	 예	 예	 아니요
<u>AWS::Notifications::OrganizationalUnitAssociation</u>	 예	 예	 아니요
<u>AWS::NotificationsContacts::EmailContact</u>	 예	 예	 아니요
<u>AWS::ODB::CloudAutonomousVmCluster</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ODB::CloudExadataInfrastructure</u>	 예	 예	 아니요
<u>AWS::ODB::CloudVmCluster</u>	 예	 예	 아니요
<u>AWS::ODB::OdbNetwork</u>	 예	 예	 아니요
<u>AWS::ODB::OdbPeeringConnection</u>	 예	 예	 아니요
<u>AWS::OSIS::Pipeline</u>	 예	 예	 예
<u>AWS::Oam::Link</u>	 예	 예	 예
<u>AWS::Oam::Sink</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ObservabilityAdmin::OrganizationCentralizationRule</u>	 예	 예	 아니요
<u>AWS::ObservabilityAdmin::OrganizationTelemetryRule</u>	 예	 예	 아니요
<u>AWS::ObservabilityAdmin::S3TableIntegration</u>	 예	 예	 아니요
<u>AWS::ObservabilityAdmin::TelemetryPipelines</u>	 예	 예	 아니요
<u>AWS::ObservabilityAdmin::TelemetryRule</u>	 예	 예	 아니요
<u>AWS::Omics::AnnotationStore</u>	 예	 예	 예
<u>AWS::Omics::ReferenceStore</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Omics::RunGroup</u>	 예	 예	 예
<u>AWS::Omics::SequenceStore</u>	 예	 예	 예
<u>AWS::Omics::VariantStore</u>	 예	 예	 예
<u>AWS::Omics::Workflow</u>	 예	 예	 예
<u>AWS::Omics::WorkflowVersion</u>	 예	 예	 아니요
<u>AWS::OpenSearchServerless::AccessPolicy</u>	 예	 예	 예
<u>AWS::OpenSearchServerless::Collection</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::OpenSearchServerless::CollectionGroup</u>	 예	 예	 아니요
<u>AWS::OpenSearchServerless::Index</u>	 예	 예	 아니요
<u>AWS::OpenSearchServerless::LifecyclePolicy</u>	 예	 예	 아니요
<u>AWS::OpenSearchServerless::SecurityConfig</u>	 예	 예	 예
<u>AWS::OpenSearchServerless::SecurityPolicy</u>	 예	 예	 예
<u>AWS::OpenSearchServerless::VpcEndpoint</u>	 예	 예	 예
<u>AWS::OpenSearchService::Application</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::OpenSearchService::Domain</u>	 예	 예	 아니요
<u>AWS::Organizations::Account</u>	 예	 예	 예
<u>AWS::Organizations::Organization</u>	 예	 예	 예
<u>AWS::Organizations::OrganizationalUnit</u>	 예	 예	 예
<u>AWS::Organizations::Policy</u>	 예	 예	 예
<u>AWS::Organizations::ResourcePolicy</u>	 예	 예	 예
<u>AWS::PCACConnectorAD::Connector</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::PCAConnectorAD::DirectoryRegistration</u>	 예	 예	 예
<u>AWS::PCAConnectorAD::ServicePrincipalName</u>	 예	 예	 예
<u>AWS::PCAConnectorAD::Template</u>	 예	 예	 예
<u>AWS::PCAConnectorAD::TemplateGroupAccessControlEntry</u>	 예	 예	 예
<u>AWS::PCAConnectorSCEP::Challenge</u>	 예	 예	 아니요
<u>AWS::PCAConnectorSCEP::Connector</u>	 예	 예	 아니요
<u>AWS::PCS::Cluster</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::PCS::ComputeNodeGroup</u>	 예	 예	 아니요
<u>AWS::PCS::Queue</u>	 예	 예	 아니요
<u>AWS::Panorama::ApplicationInstance</u>	 예	 예	 예
<u>AWS::Panorama::Package</u>	 예	 예	 예
<u>AWS::Panorama::PackageVersion</u>	 예	 예	 아니요
<u>AWS::PaymentCryptography::Alias</u>	 예	 예	 아니요
<u>AWS::PaymentCryptography::Key</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Personalize::Dataset</u>	 예	 예	 예
<u>AWS::Personalize::DatasetGroup</u>	 예	 예	 예
<u>AWS::Personalize::Schema</u>	 예	 예	 예
<u>AWS::Personalize::Solution</u>	 예	 예	 아니요
<u>AWS::Pinpoint::InAppTemplate</u>	 예	 예	 예
<u>AWS::Pipes::Pipe</u>	 예	 예	 예
<u>AWS::Proton::EnvironmentAccountConnection</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Proton::EnvironmentTemplate</u>	 예	 예	 예
<u>AWS::Proton::ServiceTemplate</u>	 예	 예	 예
<u>AWS::QBusiness::Application</u>	 예	 예	 아니요
<u>AWS::QBusiness::DataAccessor</u>	 예	 예	 아니요
<u>AWS::QBusiness::DataSource</u>	 예	 예	 아니요
<u>AWS::QBusiness::Index</u>	 예	 예	 아니요
<u>AWS::QBusiness::Permission</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::QBusiness::Plugin</u>	 예	 예	 아니요
<u>AWS::QBusiness::Retriever</u>	 예	 예	 아니요
<u>AWS::QBusiness::WebExperience</u>	 예	 예	 아니요
<u>AWS::QLDB::Stream</u>	 예	 예	 예
<u>AWS::QuickSight::Analysis</u>	 예	 예	 아니요
<u>AWS::QuickSight::CustomPermissions</u>	 예	 예	 아니요
<u>AWS::QuickSight::Dashboard</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::QuickSight::DataSet</u>	 예	 예	 예
<u>AWS::QuickSight::DataSource</u>	 예	 예	 예
<u>AWS::QuickSight::Folder</u>	 예	 예	 아니요
<u>AWS::QuickSight::RefreshSchedule</u>	 예	 예	 예
<u>AWS::QuickSight::Template</u>	 예	 예	 아니요
<u>AWS::QuickSight::Theme</u>	 예	 예	 아니요
<u>AWS::QuickSight::Topic</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::QuickSight::VPCConnection</u>	 예	 예	 예
<u>AWS::RAM::Permission</u>	 예	 예	 아니요
<u>AWS::RAM::ResourceShare</u>	 예	 예	 아니요
<u>AWS::RDS::CustomDBEngineVersion</u>	 예	 예	 아니요
<u>AWS::RDS::DBCluster</u>	 예	 예	 예
<u>AWS::RDS::DBClusterParameterGroup</u>	 예	 예	 아니요
<u>AWS::RDS::DBInstance</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::RDS::DBParameterGroup</u>	 예	 예	 아니요
<u>AWS::RDS::DBProxy</u>	 예	 예	 예
<u>AWS::RDS::DBProxyEndpoint</u>	 예	 예	 예
<u>AWS::RDS::DBProxyTargetGroup</u>	 예	 예	 예
<u>AWS::RDS::DBShardGroup</u>	 예	 예	 아니요
<u>AWS::RDS::DBSubnetGroup</u>	 예	 예	 예
<u>AWS::RDS::EventSubscription</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::RDS::GlobalCluster</u>	 예	 예	 예
<u>AWS::RDS::Integration</u>	 예	 예	 아니요
<u>AWS::RDS::OptionGroup</u>	 예	 예	 예
<u>AWS::RTBFabric::InboundExternalLink</u>	 예	 예	 아니요
<u>AWS::RTBFabric::Link</u>	 예	 예	 아니요
<u>AWS::RTBFabric::OutboundExternalLink</u>	 예	 예	 아니요
<u>AWS::RTBFabric::RequesterGateway</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::RTBFabric::ResponderGateway</u>	 예	 예	 아니요
<u>AWS::RUM::AppMonitor</u>	 예	 예	 예
<u>AWS::Rbin::Rule</u>	 예	 예	 아니요
<u>AWS::Redshift::Cluster</u>	 예	 예	 예
<u>AWS::Redshift::ClusterParameterGroup</u>	 예	 예	 아니요
<u>AWS::Redshift::ClusterSubnetGroup</u>	 예	 예	 아니요
<u>AWS::Redshift::EndpointAccess</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Redshift::EndpointAuthorization</u>	 예	 예	 예
<u>AWS::Redshift::EventSubscription</u>	 예	 예	 아니요
<u>AWS::Redshift::Integration</u>	 예	 예	 아니요
<u>AWS::Redshift::ScheduledAction</u>	 예	 예	 예
<u>AWS::RedshiftServerless::Namespace</u>	 예	 예	 아니요
<u>AWS::RedshiftServerless::Snapshot</u>	 예	 예	 아니요
<u>AWS::RedshiftServerless::Workgroup</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::RefactorSpaces::Application</u>	 예	 예	 예
<u>AWS::RefactorSpaces::Environment</u>	 예	 예	 예
<u>AWS::RefactorSpaces::Route</u>	 예	 예	 예
<u>AWS::RefactorSpaces::Service</u>	 예	 예	 예
<u>AWS::Rekognition::Collection</u>	 예	 예	 예
<u>AWS::Rekognition::Project</u>	 예	 예	 예
<u>AWS::Rekognition::StreamProcessor</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ResilienceHub::App</u>	 예	 예	 예
<u>AWS::ResilienceHub::ResiliencyPolicy</u>	 예	 예	 예
<u>AWS::ResourceExplorer2::DefaultViewAssociation</u>	 예	 예	 아니요
<u>AWS::ResourceExplorer2::Index</u>	 예	 예	 예
<u>AWS::ResourceExplorer2::View</u>	 예	 예	 예
<u>AWS::ResourceGroups::Group</u>	 예	 예	 예
<u>AWS::ResourceGroups::TagSyncTask</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::RolesAnywhere::CRL</u>	 예	 예	 예
<u>AWS::RolesAnywhere::Profile</u>	 예	 예	 예
<u>AWS::RolesAnywhere::TrustAnchor</u>	 예	 예	 아니요
<u>AWS::Route53::CidrCollection</u>	 예	 예	 예
<u>AWS::Route53::DNSSEC</u>	 예	 예	 예
<u>AWS::Route53::HealthCheck</u>	 예	 예	 예
<u>AWS::Route53::HostedZone</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Route53::KeySigningKey</u>	 예	 예	 예
<u>AWS::Route53Profiles::Profile</u>	 예	 예	 아니요
<u>AWS::Route53Profiles::ProfileAssociation</u>	 예	 예	 아니요
<u>AWS::Route53Profiles::ProfileResourceAssociation</u>	 예	 예	 아니요
<u>AWS::Route53RecoveryControl::Cluster</u>	 예	 예	 예
<u>AWS::Route53RecoveryControl::ControlPanel</u>	 예	 예	 예
<u>AWS::Route53RecoveryControl::RoutingControl</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Route53RecoveryControl::SafetyRule</u>	 예	 예	 예
<u>AWS::Route53RecoveryReadiness::Cell</u>	 예	 예	 예
<u>AWS::Route53RecoveryReadiness::ReadinessCheck</u>	 예	 예	 예
<u>AWS::Route53RecoveryReadiness::RecoveryGroup</u>	 예	 예	 예
<u>AWS::Route53RecoveryReadiness::ResourceSet</u>	 예	 예	 예
<u>AWS::Route53Resolver::FirewallDomainList</u>	 예	 예	 아니요
<u>AWS::Route53Resolver::FirewallRuleGroup</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Route53Resolver::FirewallRuleGroupAssociation</u>	 예	 예	 예
<u>AWS::Route53Resolver::OutpostResolver</u>	 예	 예	 예
<u>AWS::Route53Resolver::ResolverConfig</u>	 예	 예	 예
<u>AWS::Route53Resolver::ResolverDNSSECConfig</u>	 예	 예	 예
<u>AWS::Route53Resolver::ResolverEndpoint</u>	 예	 예	 아니요
<u>AWS::Route53Resolver::ResolverQueryLoggingConfig</u>	 예	 예	 예
<u>AWS::Route53Resolver::ResolverQueryLoggingConfigAssociation</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Route53Resolver::ResolverRule</u>	 예	 예	 아니요
<u>AWS::Route53Resolver::ResolverRuleAssociation</u>	 예	 예	 예
<u>AWS::S3::AccessGrant</u>	 예	 예	 아니요
<u>AWS::S3::AccessGrantsInstance</u>	 예	 예	 아니요
<u>AWS::S3::AccessGrantsLocation</u>	 예	 예	 아니요
<u>AWS::S3::AccessPoint</u>	 예	 예	 예
<u>AWS::S3::Bucket</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::S3::BucketPolicy</u>	 예	 아니요	 예
<u>AWS::S3::MultiRegionAccessPoint</u>	 예	 예	 예
<u>AWS::S3::MultiRegionAccessPointPolicy</u>	 예	 아니요	 아니요
<u>AWS::S3::StorageLens</u>	 예	 예	 예
<u>AWS::S3::StorageLensGroup</u>	 예	 예	 아니요
<u>AWS::S3Express::AccessPoint</u>	 예	 예	 아니요
<u>AWS::S3Express::BucketPolicy</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::S3Express::DirectoryBucket</u>	 예	 예	 아니요
<u>AWS::S3ObjectLambda::AccessPoint</u>	 예	 예	 예
<u>AWS::S3ObjectLambda::AccessPointPolicy</u>	 예	 예	 아니요
<u>AWS::S3Outposts::AccessPoint</u>	 예	 예	 예
<u>AWS::S3Outposts::Bucket</u>	 예	 예	 예
<u>AWS::S3Outposts::BucketPolicy</u>	 예	 예	 아니요
<u>AWS::S3Outposts::Endpoint</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::S3Tables::Namespace</u>	 예	 예	 아니요
<u>AWS::S3Tables::Table</u>	 예	 예	 아니요
<u>AWS::S3Tables::TableBucket</u>	 예	 예	 아니요
<u>AWS::S3Tables::TableBucketPolicy</u>	 예	 예	 아니요
<u>AWS::S3Tables::TablePolicy</u>	 예	 예	 아니요
<u>AWS::S3Vectors::Index</u>	 예	 예	 아니요
<u>AWS::S3Vectors::VectorBucket</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::S3Vectors::VectorBucketPolicy</u>	 예	 예	 아니요
<u>AWS::SES::ConfigurationSet</u>	 예	 예	 아니요
<u>AWS::SES::ConfigurationSetEventDestination</u>	 아니요	 예	 아니요
<u>AWS::SES::ContactList</u>	 예	 예	 아니요
<u>AWS::SES::DedicatedIpPool</u>	 예	 예	 예
<u>AWS::SES::EmailIdentity</u>	 예	 예	 예
<u>AWS::SES::MailManagerAddonInstance</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SES::MailManagerAddonSubscription</u>	 예	 예	 아니요
<u>AWS::SES::MailManagerAddressList</u>	 예	 예	 아니요
<u>AWS::SES::MailManagerArchive</u>	 예	 예	 아니요
<u>AWS::SES::MailManagerIngressPoint</u>	 예	 예	 아니요
<u>AWS::SES::MailManagerRelay</u>	 예	 예	 아니요
<u>AWS::SES::MailManagerRuleSet</u>	 예	 예	 아니요
<u>AWS::SES::MailManagerTrafficPolicy</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SES::MultiRegionEndpoint</u>	 예	 예	 아니요
<u>AWS::SES::Template</u>	 예	 예	 아니요
<u>AWS::SES::Tenant</u>	 예	 예	 아니요
<u>AWS::SES::VdmAttributes</u>	 예	 예	 아니요
<u>AWS::SMSVOICE::ConfigurationSet</u>	 예	 예	 아니요
<u>AWS::SMSVOICE::OptOutList</u>	 예	 예	 아니요
<u>AWS::SMSVOICE::PhoneNumber</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SMSVOICE::Pool</u>	 예	 예	 아니요
<u>AWS::SMSVOICE::ProtectConfiguration</u>	 예	 예	 아니요
<u>AWS::SMSVOICE::ResourcePolicy</u>	 예	 예	 아니요
<u>AWS::SMSVOICE::SenderId</u>	 예	 예	 아니요
<u>AWS::SNS::Subscription</u>	 예	 예	 아니요
<u>AWS::SNS::Topic</u>	 예	 예	 예
<u>AWS::SNS::TopicInlinePolicy</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SQS::Queue</u>	 예	 예	 예
<u>AWS::SQS::QueueInlinePolicy</u>	 예	 예	 아니요
<u>AWS::SSM::Association</u>	 예	 예	 예
<u>AWS::SSM::Document</u>	 예	 아니요	 아니요
<u>AWS::SSM::MaintenanceWindowTarget</u>	 예	 예	 아니요
<u>AWS::SSM::MaintenanceWindowTask</u>	 예	 예	 아니요
<u>AWS::SSM::Parameter</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SSM::PatchBaseline</u>	 예	 예	 아니요
<u>AWS::SSM::ResourceDataSync</u>	 예	 예	 아니요
<u>AWS::SSM::ResourcePolicy</u>	 예	 예	 예
<u>AWS::SSMContacts::Contact</u>	 예	 예	 예
<u>AWS::SSMContacts::ContactChannel</u>	 예	 예	 예
<u>AWS::SSMContacts::Plan</u>	 예	 예	 아니요
<u>AWS::SSMContacts::Rotation</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SSMGuiConnect::Preferences</u>	 예	 예	 아니요
<u>AWS::SSMIncidents::ReplicationSet</u>	 예	 예	 예
<u>AWS::SSMIncidents::ResponsePlan</u>	 예	 예	 예
<u>AWS::SSMQuickSetup::ConfigurationManager</u>	 예	 예	 아니요
<u>AWS::SSMQuickSetup::LifecycleAutomation</u>	 예	 예	 아니요
<u>AWS::SSO::Application</u>	 예	 예	 아니요
<u>AWS::SSO::ApplicationAssignment</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SSO::Assignment</u>	 예	 예	 아니요
<u>AWS::SSO::Instance</u>	 예	 예	 아니요
<u>AWS::SSO::InstanceAccessControlAttributeConfiguration</u>	 예	 예	 예
<u>AWS::SageMaker::App</u>	 예	 예	 아니요
<u>AWS::SageMaker::AppImageConfig</u>	 예	 예	 아니요
<u>AWS::SageMaker::Cluster</u>	 예	 예	 아니요
<u>AWS::SageMaker::DataQualityJobDefinition</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SageMaker::Device</u>	 예	 예	 아니요
<u>AWS::SageMaker::DeviceFleet</u>	 예	 예	 아니요
<u>AWS::SageMaker::Domain</u>	 예	 예	 아니요
<u>AWS::SageMaker::Endpoint</u>	 아니요	 예	 아니요
<u>AWS::SageMaker::FeatureGroup</u>	 예	 예	 예
<u>AWS::SageMaker::Image</u>	 예	 예	 예
<u>AWS::SageMaker::ImageVersion</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SageMaker::InferenceComponent</u>	 예	 예	 아니요
<u>AWS::SageMaker::InferenceExperiment</u>	 예	 예	 예
<u>AWS::SageMaker::MlflowTrackingServer</u>	 예	 예	 아니요
<u>AWS::SageMaker::ModelBiasJobDefinition</u>	 예	 예	 아니요
<u>AWS::SageMaker::ModelCard</u>	 예	 예	 예
<u>AWS::SageMaker::ModelExplainabilityJobDefinition</u>	 예	 예	 아니요
<u>AWS::SageMaker::ModelPackage</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SageMaker::ModelPackageGroup</u>	 예	 예	 예
<u>AWS::SageMaker::ModelQualityJobDefinition</u>	 예	 예	 아니요
<u>AWS::SageMaker::MonitoringSchedule</u>	 예	 예	 아니요
<u>AWS::SageMaker::PartnerApp</u>	 예	 예	 아니요
<u>AWS::SageMaker::Pipeline</u>	 예	 예	 예
<u>AWS::SageMaker::ProcessingJob</u>	 예	 예	 아니요
<u>AWS::SageMaker::Project</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SageMaker::Space</u>	 예	 예	 아니요
<u>AWS::SageMaker::StudioLifecycleConfig</u>	 예	 예	 아니요
<u>AWS::SageMaker::UserProfile</u>	 예	 예	 아니요
<u>AWS::Scheduler::Schedule</u>	 예	 예	 예
<u>AWS::Scheduler::ScheduleGroup</u>	 예	 예	 아니요
<u>AWS::SecretsManager::ResourcePolicy</u>	 예	 아니요	 아니요
<u>AWS::SecretsManager::RotationSchedule</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SecretsManager::Secret</u>	 예	 예	 예
<u>AWS::SecretsManager::SecretTargetAttachment</u>	 예	 예	 아니요
<u>AWS::SecurityHub::AggregatorV2</u>	 예	 예	 아니요
<u>AWS::SecurityHub::AutomationRule</u>	 예	 예	 아니요
<u>AWS::SecurityHub::AutomationRuleV2</u>	 예	 예	 아니요
<u>AWS::SecurityHub::ConfigurationPolicy</u>	 예	 예	 아니요
<u>AWS::SecurityHub::ConnectorV2</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SecurityHub::DelegatedAdmin</u>	 예	 예	 아니요
<u>AWS::SecurityHub::FindingAggregator</u>	 예	 예	 아니요
<u>AWS::SecurityHub::Hub</u>	 예	 예	 아니요
<u>AWS::SecurityHub::HubV2</u>	 예	 예	 아니요
<u>AWS::SecurityHub::Insight</u>	 예	 예	 아니요
<u>AWS::SecurityHub::OrganizationConfiguration</u>	 예	 예	 아니요
<u>AWS::SecurityHub::PolicyAssociation</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SecurityHub::ProductSubscription</u>	 예	 예	 아니요
<u>AWS::SecurityHub::SecurityControl</u>	 예	 예	 아니요
<u>AWS::SecurityHub::Standard</u>	 예	 예	 아니요
<u>AWS::SecurityLake::AwsLogSource</u>	 예	 예	 아니요
<u>AWS::SecurityLake::DataLake</u>	 예	 예	 아니요
<u>AWS::SecurityLake::Subscriber</u>	 예	 예	 아니요
<u>AWS::SecurityLake::SubscriberNotification</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ServiceCatalog::CloudFormationProvisionedProduct</u>	 예	 아니요	 아니요
<u>AWS::ServiceCatalog::LaunchNotificationConstraint</u>	 예	 예	 아니요
<u>AWS::ServiceCatalog::LaunchTemplateConstraint</u>	 예	 예	 아니요
<u>AWS::ServiceCatalog::PortfolioPrincipalAssociation</u>	 예	 예	 아니요
<u>AWS::ServiceCatalog::PortfolioProductAssociation</u>	 예	 예	 아니요
<u>AWS::ServiceCatalog::PortfolioShare</u>	 예	 예	 아니요
<u>AWS::ServiceCatalog::ResourceUpdateConstraint</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ServiceCatalog::ServiceAction</u>	 예	 예	 예
<u>AWS::ServiceCatalog::ServiceActionAssociation</u>	 예	 예	 아니요
<u>AWS::ServiceCatalog::TagOption</u>	 예	 예	 아니요
<u>AWS::ServiceCatalog::TagOptionAssociation</u>	 예	 예	 아니요
<u>AWS::ServiceCatalogAppRegistry::Application</u>	 예	 예	 예
<u>AWS::ServiceCatalogAppRegistry::AttributeGroup</u>	 예	 예	 예
<u>AWS::ServiceCatalogAppRegistry::AttributeGroupAssociation</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::ServiceCatalogAppRegistry::ResourceAssociation</u>	 예	 아니요	 아니요
<u>AWS::Shield::DRTAccess</u>	 예	 예	 아니요
<u>AWS::Shield::ProactiveEngagement</u>	 예	 예	 아니요
<u>AWS::Shield::Protection</u>	 예	 예	 아니요
<u>AWS::Shield::ProtectionGroup</u>	 예	 예	 아니요
<u>AWS::Signer::ProfilePermission</u>	 예	 예	 아니요
<u>AWS::Signer::SigningProfile</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SimSpaceWeaver::Simulation</u>	 예	 예	 예
<u>AWS::StepFunctions::Activity</u>	 예	 예	 아니요
<u>AWS::StepFunctions::StateMachine</u>	 예	 예	 예
<u>AWS::StepFunctions::StateMachineAlias</u>	 예	 예	 예
<u>AWS::StepFunctions::StateMachineVersion</u>	 예	 예	 예
<u>AWS::SupportApp::AccountAlias</u>	 예	 예	 예
<u>AWS::SupportApp::SlackChannelConfiguration</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::SupportApp::SlackWorkspaceConfiguration</u>	 예	 예	 예
<u>AWS::Synthetics::Canary</u>	 예	 예	 예
<u>AWS::Synthetics::Group</u>	 예	 예	 예
<u>AWS::SystemsManagerSAP::Application</u>	 예	 예	 아니요
<u>AWS::Timestream::Database</u>	 예	 예	 예
<u>AWS::Timestream::InfluxDBInstance</u>	 예	 예	 아니요
<u>AWS::Timestream::ScheduledQuery</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Timestream::Table</u>	 예	 예	 예
<u>AWS::Transfer::Agreement</u>	 예	 예	 예
<u>AWS::Transfer::Certificate</u>	 예	 예	 예
<u>AWS::Transfer::Connector</u>	 예	 예	 예
<u>AWS::Transfer::Profile</u>	 예	 예	 예
<u>AWS::Transfer::Server</u>	 예	 예	 아니요
<u>AWS::Transfer::User</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
AWS::Transfer::WebApp	 예	 예	 아니요
AWS::Transfer::Workflow	 예	 예	 예
AWS::VerifiedPermissions::IdentitySource	 예	 예	 예
AWS::VerifiedPermissions::Policy	 예	 예	 예
AWS::VerifiedPermissions::PolicyStore	 예	 예	 예
AWS::VerifiedPermissions::PolicyTemplate	 예	 예	 예
AWS::VoiceID::Domain	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::VpcLattice::AccessLogSubscription</u>	 예	 예	 예
<u>AWS::VpcLattice::AuthPolicy</u>	 예	 예	 아니요
<u>AWS::VpcLattice::DomainVerification</u>	 예	 예	 아니요
<u>AWS::VpcLattice::Listener</u>	 예	 예	 예
<u>AWS::VpcLattice::ResourceConfiguration</u>	 예	 예	 아니요
<u>AWS::VpcLattice::ResourceGateway</u>	 예	 예	 아니요
<u>AWS::VpcLattice::ResourcePolicy</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::VpcLattice::Rule</u>	 예	 예	 예
<u>AWS::VpcLattice::Service</u>	 예	 예	 예
<u>AWS::VpcLattice::ServiceNetwork</u>	 예	 예	 예
<u>AWS::VpcLattice::ServiceNetworkResourceAssociation</u>	 예	 예	 아니요
<u>AWS::VpcLattice::ServiceNetworkServiceAssociation</u>	 예	 예	 예
<u>AWS::VpcLattice::ServiceNetworkVpcAssociation</u>	 예	 예	 예
<u>AWS::VpcLattice::TargetGroup</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::WAFv2::IPSet</u>	 예	 예	 예
<u>AWS::WAFv2::LoggingConfiguration</u>	 예	 예	 예
<u>AWS::WAFv2::RegexPatternSet</u>	 예	 예	 예
<u>AWS::WAFv2::RuleGroup</u>	 예	 예	 예
<u>AWS::WAFv2::WebACL</u>	 예	 예	 예
<u>AWS::WAFv2::WebACLAssociation</u>	 예	 예	 아니요
<u>AWS::Wisdom::AIAgent</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Wisdom::AIAgentVersion</u>	 예	 예	 아니요
<u>AWS::Wisdom::AIGuardrail</u>	 예	 예	 아니요
<u>AWS::Wisdom::AIGuardrailVersion</u>	 예	 예	 아니요
<u>AWS::Wisdom::AIPrompt</u>	 예	 예	 아니요
<u>AWS::Wisdom::AIPromptVersion</u>	 예	 예	 아니요
<u>AWS::Wisdom::Assistant</u>	 예	 예	 예
<u>AWS::Wisdom::AssistantAssociation</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::Wisdom::KnowledgeBase</u>	 예	 예	 예
<u>AWS::Wisdom::MessageTemplate</u>	 예	 예	 아니요
<u>AWS::Wisdom::MessageTemplateVersion</u>	 예	 예	 아니요
<u>AWS::Wisdom::QuickResponse</u>	 예	 예	 아니요
<u>AWS::WorkSpaces::ConnectionAlias</u>	 예	 예	 아니요
<u>AWS::WorkSpaces::WorkspacesPool</u>	 예	 예	 아니요
<u>AWS::WorkSpacesThinClient::Environment</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::WorkSpacesWeb::BrowserSettings</u>	 예	 예	 예
<u>AWS::WorkSpacesWeb::DataProtectionSettings</u>	 예	 예	 아니요
<u>AWS::WorkSpacesWeb::IdentityProvider</u>	 예	 예	 예
<u>AWS::WorkSpacesWeb::IpAccessSettings</u>	 예	 예	 예
<u>AWS::WorkSpacesWeb::NetworkSettings</u>	 예	 예	 예
<u>AWS::WorkSpacesWeb::Portal</u>	 예	 예	 예
<u>AWS::WorkSpacesWeb::SessionLogger</u>	 예	 예	 아니요

리소스	가져오기	드리프트 감지	IaC 생성기
<u>AWS::WorkSpacesWeb::TrustStore</u>	 예	 예	 예
<u>AWS::WorkSpacesWeb::UserAccessLoggingSettings</u>	 예	 예	 예
<u>AWS::WorkSpacesWeb::UserSettings</u>	 예	 예	 예
<u>AWS::WorkspacesInstances::Volume</u>	 예	 예	 아니요
<u>AWS::WorkspacesInstances::VolumeAssociation</u>	 예	 예	 아니요
<u>AWS::WorkspacesInstances::WorkspaceInstance</u>	 예	 예	 아니요
<u>AWS::XRay::Group</u>	 예	 예	 예

리소스	가져오기	드리프트 감지	IaC 생성기
AWS::XRay::ResourcePolicy	 예	 예	 예
AWS::XRay::SamplingRule	 예	 예	 아니요
AWS::XRay::TransactionSearchConfig	 예	 예	 아니요

빠른 생성 링크를 사용하여 CloudFormation 스택 생성

빠른 생성 링크는 CloudFormation 콘솔의 URL에서 직접 CloudFormation 스택을 시작하는 간소화된 방법을 제공합니다. URL 쿼리 파라미터와 같이 템플릿 URL, 스택 이름, 템플릿 파라미터를 지정하여 단일 스택 생성 페이지를 미리 채워 스택 생성 속도를 높일 수 있습니다. 이렇게 하면 마법사 페이지 수와 필수 사항인 사용자 입력 양이 모두 줄어 스택 생성 프로세스가 간소화됩니다. 또한 동일한 템플릿에 대해 다른 값을 지정하는 여러 URL을 생성할 수 있으므로 템플릿 재사용이 최적화됩니다.

URL 형식

빠른 생성 링크는 다음 URL 형식을 따릅니다.

```
https://region-code.console.aws.amazon.com/cloudformation/home?region=region-code#/stacks/create/review?templateURL=TemplateURL&stackName=StackName&param_parameterName=parameterValue
```

CloudFormation에서는 다음 URL 쿼리 파라미터를 지원합니다.

템플릿 URL

필수 사항입니다. `templateURL` 파라미터는 Amazon S3 버킷에 있는 스택 템플릿의 URL을 지정합니다. 미리 서명된 S3 URL을 사용하여 액세스 문제를 방지하려면 이 S3 URL을 URL로 인코딩해야 합니다.

지원되는 S3 URL 형식:

- `https://s3.region-code.amazonaws.com/bucket-name/template-name`
- `https://bucket-name.s3.region-code.amazonaws.com/template-name`
- `https://s3-region-code.amazonaws.com/bucket-name/template-name`(레거시 형식)

스택 이름

선택 사항. `stackName` 파라미터를 사용하여 생성할 CloudFormation 스택의 이름을 지정합니다. 스택 이름에는 영숫자(대소문자 구분)와 하이픈만 사용할 수 있습니다. 영문자로 시작해야 하고 128자 이하여야 합니다.

템플릿 파라미터

선택 사항. `NoEcho` 파라미터 유형이 아닌 스택 템플릿의 파라미터의 경우 URL 쿼리 문자열의 `param_parameterName` 형식을 사용합니다. URL 파라미터에는 `param_` 접두사가 있어야 하며, 파라미터 이름 세그먼트가 템플릿의 파라미터 이름과 일치해야 합니다. 예를 들면 `param_DBName`입니다.

CloudFormation은 템플릿에 없는 파라미터와 `NoEcho` 속성이 `true` 유형으로 설정되어 정의된 모든 파라미터(일반적으로 사용자 이름 및 암호)를 무시합니다. URL 파라미터는 템플릿에 지정된 기본값을 재정의합니다. 필요한 만큼 파라미터를 포함할 수 있습니다.

Important

AWS Systems Manager Parameter Store 또는 AWS Secrets Manager와 같이 CloudFormation 외부에서 저장 및 관리되는 중요한 정보를 참조하려면 CloudFormation 템플릿에 직접 중요한 정보를 포함하는 대신 스택 템플릿에 있는 동적 파라미터를 사용하는 것이 좋습니다.

자세한 내용은 [템플릿에 자격 증명을 포함하지 않음](#)을 참조하십시오.

모든 쿼리 파라미터 이름은 대/소문자를 구분합니다. 사용자는 스택을 생성하기 전에 콘솔에서 이러한 값을 덮어쓸 수 있습니다.

예제

다음 예는 [WordPress 기본 단일 인스턴스](#) 샘플 템플릿을 기반으로 합니다. 쿼리 문자열에는 필수 `templateURL` 파라미터와 `stackName`, `DBName`, `InstanceType` 및 `KeyName` 파라미터가 포함되어 있습니다.

다음 URL에는 명확성을 위해 줄 바꿈이 추가되었습니다.

```
https://us-east-2.console.aws.amazon.com/cloudformation/home?region=us-east-2#/stacks/
create/review
    ?templateURL=https://s3.us-east-2.amazonaws.com/cloudformation-templates-us-east-2/
WordPress_Single_Instance.template
    &stackName=MyWPBlog
    &param_DBName=mywpblog
    &param_InstanceType=t2.medium
```

다음 URL에는 이전 예와 동일한 파라미터가 포함되어 있지만 줄 바꿈이 삭제되었습니다. 이것이 실제 URL 형식입니다.

```
https://us-east-2.console.aws.amazon.com/cloudformation/home?
region=us-east-2#/stacks/create/review?templateURL=https://
s3.us-east-2.amazonaws.com/cloudformation-templates-us-east-2/
WordPress_Single_Instance.template&stackName=MyWPBlog&param_DBName=mywpblog&param_InstanceType=
```

빠른 생성 링크를 사용하여 스택 생성

빠른 생성 링크를 열면 CloudFormation 콘솔로 이동합니다. 콘솔이 빠른 스택 생성 페이지로 바로 열리고 제공된 값이 파라미터에 자동으로 사용됩니다.

빠른 생성 링크를 사용하여 스택을 생성하려면(콘솔)

1. 빠른 스택 생성 페이지의 템플릿, 템플릿 URL에서 템플릿 URL이 올바른지 확인합니다.
2. 템플릿 보기 섹션을 확장하여 템플릿을 확인합니다.
3. 스택 이름에서 미리 채워진 스택 이름을 확인합니다.
4. 파라미터 섹션을 검토합니다. 미리 채워진 파라미터 값이 올바른지 확인합니다. URL에 지정되지 않은 필수 파라미터를 입력합니다. 필요한 경우 미리 채워진 값을 수정합니다.
5. 그다음 단계에서 다음과 같은 설정을 구성할 수 있습니다.
 - 태그 - 키-값 페어로 리소스를 구성합니다.

- 권한 - 스택 작업에 대한 IAM 서비스 역할을 선택합니다.
- 스택 실패 옵션 - 리소스 롤백(기본값) 또는 리소스 보존을 선택합니다.
- 스택 정책 - 리소스 업데이트 권한을 제어합니다.
- 롤백 구성 - CloudWatch 경보 기반 롤백을 구성합니다.
- 알림 옵션 - 스택 이벤트에 대한 Amazon SNS 알림을 설정합니다.
- 스택 생성 옵션 - 최대 스택 생성 시간을 정의하고 종료 방지 기능을 활성화하여 실수로 삭제되지 않게 합니다.

자세한 내용은 [스택 옵션 구성](#) 섹션을 참조하세요.

6. 기능에서 필요한 승인을 완료합니다. 예를 들어 템플릿에 IAM 리소스가 포함된 경우 이 템플릿이 IAM 리소스를 생성할 수 있음을 확인합니다를 선택하여 템플릿에서 IAM 리소스를 사용하도록 지정합니다. 자세한 내용은 [CloudFormation 템플릿에서 IAM 리소스 승인](#) 섹션을 참조하세요.
7. (선택 사항) 변경 세트를 생성하여 스택을 생성하기 전에 해당 구성을 미리 볼 수 있습니다. 변경 세트 생성을 선택하고 지침을 따릅니다. 자세한 내용은 [스택 구성 미리 보기](#) 섹션을 참조하세요.
8. 준비가 되면 스택 생성을 선택하여 스택을 시작한 다음 이벤트 탭에서 스택 생성 진행 상황을 모니터링합니다. 자세한 내용은 [스택 진행 상황 모니터링](#) 섹션을 참조하세요.

AWS CLI 및 PowerShell에 대한 CloudFormation 스택 작업 명령의 예

다음 명령줄 예제는 AWS CLI 및 PowerShell을 사용하여 개별 CloudFormation 작업을 수행하는 방법을 보여줍니다. 이러한 예제에는 가장 일반적으로 사용되는 작업만 포함되어 있습니다. 전체 옵션 목록은 AWS CLI 명령 참조의 [cloudformation](#) 섹션을 참조하세요.

이 설명서의 예제에서는 백슬래시(\) 규칙을 사용하여 긴 명령줄이 다음 줄에 계속됨을 나타냅니다.

주제

- [스택 업데이트 취소](#)
- [업데이트 롤백 계속](#)
- [스택을 만듭니다](#)
- [변환이 포함된 스택 생성](#)
- [스택 삭제](#)
- [스택 이벤트 설명](#)

- [스택 리소스 설명](#)
- [스택 리소스 설명](#)
- [스택 설명](#)
- [템플릿 가져오기](#)
- [스택 리소스 나열](#)
- [스택 나열](#)
- [스택 업데이트](#)
- [템플릿 검증](#)
- [AWS CLI를 사용하여 S3 버킷에 로컬 아티팩트 업로드](#)

스택 업데이트 취소

[cancel-update-stack](#) 명령을 사용하여 스택 업데이트를 취소합니다. 자세한 내용은 [스택 업데이트 취소](#) 섹션을 참조하세요.

CLI

AWS CLI

진행 중인 스택 업데이트를 취소하는 방법

다음 `cancel-update-stack` 명령은 `myteststack` 스택의 스택 업데이트를 취소합니다.

```
aws cloudformation cancel-update-stack --stack-name myteststack
```

- API 세부 정보는 AWS CLI 명령 참조의 [CancelUpdateStack](#) 섹션을 참조하세요.

PowerShell

Tools for PowerShell V4

예 1: 지정된 스택에 대한 업데이트를 취소합니다.

```
Stop-CFNUdateStack -StackName "myStack"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 [CancelUpdateStack](#)을 참조하세요.

Tools for PowerShell V5

예 1: 지정된 스택에 대한 업데이트를 취소합니다.

```
Stop-CFNUpdateStack -StackName "myStack"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 [CancelUpdateStack](#)을 참조하세요.

업데이트 롤백 계속

[continue-update-rollback](#) 명령을 사용하여 업데이트 롤백을 계속합니다. 자세한 내용은 [업데이트 롤백 계속](#) 섹션을 참조하세요.

CLI

AWS CLI

업데이트 롤백을 재시도하는 방법

다음 `continue-update-rollback` 예제에서는 이전에 실패한 스택 업데이트에서 롤백 작업을 재개합니다.

```
aws cloudformation continue-update-rollback \  
  --stack-name my-stack
```

이 명령은 출력을 생성하지 않습니다.

- API 세부 정보는 AWS CLI 명령 참조의 [ContinueUpdateRollback](#) 섹션을 참조하세요.

PowerShell

Tools for PowerShell V4

예 1: 이름이 지정된 스택의 롤백을 계속합니다. 스택은 'UPDATE_ROLLBACK_FAILED' 상태여야 합니다. 계속된 롤백이 성공하면 스택은 'UPDATE_ROLLBACK_COMPLETE' 상태가 됩니다.

```
Resume-CFNUpdateRollback -StackName "myStack"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 [ContinueUpdateRollback](#)을 참조하세요.

Tools for PowerShell V5

예 1: 이름이 지정된 스택의 롤백을 계속합니다. 스택은 'UPDATE_ROLLBACK_FAILED' 상태여야 합니다. 계속된 롤백이 성공하면 스택은 'UPDATE_ROLLBACK_COMPLETE' 상태가 됩니다.

```
Resume-CFNUpdateRollback -StackName "myStack"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 [ContinueUpdateRollback](#)을 참조하세요.

스택을 만듭니다

[create-stack](#) 명령을 사용하여 스택을 생성합니다. 스택 이름, 유효한 템플릿의 위치 및 입력 파라미터를 제공해야 합니다. 파라미터 키 이름은 대/소문자를 구분합니다. 파라미터 키 이름을 잘못 입력할 경우 CloudFormation에서는 스택을 생성하지 않고 템플릿에 해당 파라미터가 없다고 보고합니다.

다음과 같은 예제에서는 지정된 이름, 템플릿 및 입력 파라미터를 사용하여 새 스택을 생성하는 방법을 보여줍니다.

CLI

AWS CLI

AWS CloudFormation 스택을 생성하는 방법

다음 `create-stacks` 명령에서는 `sampletemplate.json` 템플릿을 사용하여 이름이 `myteststack`인 스택을 생성합니다.

```
aws cloudformation create-stack --stack-name myteststack --template-body file://sampletemplate.json --parameters ParameterKey=KeyPairName,ParameterValue=TestKey  
ParameterKey=SubnetIDs,ParameterValue=SubnetID1\\,SubnetID2
```

출력:

```
{  
  "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/  
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896"
```

```
}
```

자세한 내용을 알아보려면 AWS CloudFormation 사용 설명서의 스택을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 [CreateStack](#) 섹션을 참조하세요.

PowerShell

Tools for PowerShell V4

예 1: 지정된 이름으로 새 스택을 생성합니다. 템플릿은 사용자 지정 파라미터를 사용하여 제공된 콘텐츠에서 구문 분석됩니다. 여기서, 'PK1' 및 'PK2'는 템플릿 콘텐츠에 선언된 파라미터의 이름을 나타내고, 'PV1' 및 'PV2'는 해당 파라미터의 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다. 스택 생성이 실패하면 롤백되지 않습니다.

```
New-CFNStack -StackName "myStack" `
    -TemplateBody "{TEMPLATE CONTENT HERE}" `
    -Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
    @{ ParameterKey="PK2"; ParameterValue="PV2" } ) `
    -DisableRollback $true
```

예 2: 지정된 이름으로 새 스택을 생성합니다. 템플릿은 사용자 지정 파라미터를 사용하여 제공된 콘텐츠에서 구문 분석됩니다. 여기서, 'PK1' 및 'PK2'는 템플릿 콘텐츠에 선언된 파라미터의 이름을 나타내고, 'PV1' 및 'PV2'는 해당 파라미터의 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다. 스택 생성이 실패하면 롤백됩니다.

```
$p1 = New-Object -Type Amazon.CloudFormation.Model.Parameter
$p1.ParameterKey = "PK1"
$p1.ParameterValue = "PV1"

$p2 = New-Object -Type Amazon.CloudFormation.Model.Parameter
$p2.ParameterKey = "PK2"
$p2.ParameterValue = "PV2"

New-CFNStack -StackName "myStack" `
    -TemplateBody "{TEMPLATE CONTENT HERE}" `
    -Parameter @( $p1, $p2 ) `
    -OnFailure "ROLLBACK"
```

예 3: 지정된 이름으로 새 스택을 생성합니다. 템플릿은 사용자 지정 파라미터를 사용하여 Amazon S3 URL에서 가져옵니다. 여기서, 'PK1'은 템플릿 콘텐츠에 선언된 파라미터의 이름을 나타내고, 'PV1'은 해당 파라미터의 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다. 스택 생성이 실패하면 롤백됩니다(-DisableRollback \$false를 지정하는 것과 동일).

```
New-CFNStack -StackName "myStack" `
              -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
              -Parameter @{ ParameterKey="PK1"; ParameterValue="PV1" }
```

예 4: 지정된 이름으로 새 스택을 생성합니다. 템플릿은 사용자 지정 파라미터를 사용하여 Amazon S3 URL에서 가져옵니다. 여기서, 'PK1'은 템플릿 콘텐츠에 선언된 파라미터의 이름을 나타내고, 'PV1'은 해당 파라미터의 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다. 스택 생성이 실패하면 롤백됩니다(-DisableRollback \$false를 지정하는 것과 동일). 지정된 알림 AEN은 게시된 스택 관련 이벤트를 수신합니다.

```
New-CFNStack -StackName "myStack" `
              -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
              -Parameter @{ ParameterKey="PK1"; ParameterValue="PV1" } `
              -NotificationARN @( "arn1", "arn2" )
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 [CreateStack](#)을 참조하세요.

Tools for PowerShell V5

예 1: 지정된 이름으로 새 스택을 생성합니다. 템플릿은 사용자 지정 파라미터를 사용하여 제공된 콘텐츠에서 구문 분석됩니다. 여기서, 'PK1' 및 'PK2'는 템플릿 콘텐츠에 선언된 파라미터의 이름을 나타내고, 'PV1' 및 'PV2'는 해당 파라미터의 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다. 스택 생성이 실패하면 롤백되지 않습니다.

```
New-CFNStack -StackName "myStack" `
              -TemplateBody "{TEMPLATE CONTENT HERE}" `
              -Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
                              @{ ParameterKey="PK2"; ParameterValue="PV2" } ) `
              -DisableRollback $true
```


예 2: 지정된 이름으로 새 스택을 생성합니다. 템플릿은 사용자 지정 파라미터를 사용하여 제공된 콘텐츠에서 구문 분석됩니다. 여기서, 'PK1' 및 'PK2'는 템플릿 콘텐츠에 선언된 파라미터의 이름을 나타내고, 'PV1' 및 'PV2'는 해당 파라미터의 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다. 스택 생성이 실패하면 롤백됩니다.

```
$p1 = New-Object -Type Amazon.CloudFormation.Model.Parameter
$p1.ParameterKey = "PK1"
$p1.ParameterValue = "PV1"

$p2 = New-Object -Type Amazon.CloudFormation.Model.Parameter
$p2.ParameterKey = "PK2"
$p2.ParameterValue = "PV2"

New-CFNStack -StackName "myStack" `
              -TemplateBody "{TEMPLATE CONTENT HERE}" `
              -Parameter @( $p1, $p2 ) `
              -OnFailure "ROLLBACK"
```

예 3: 지정된 이름으로 새 스택을 생성합니다. 템플릿은 사용자 지정 파라미터를 사용하여 Amazon S3 URL에서 가져옵니다. 여기서, 'PK1'은 템플릿 콘텐츠에 선언된 파라미터의 이름을 나타내고, 'PV1'은 해당 파라미터의 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다. 스택 생성이 실패하면 롤백됩니다(-DisableRollback \$false를 지정하는 것과 동일).

```
New-CFNStack -StackName "myStack" `
              -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
              -Parameter @{ ParameterKey="PK1"; ParameterValue="PV1" }
```

예 4: 지정된 이름으로 새 스택을 생성합니다. 템플릿은 사용자 지정 파라미터를 사용하여 Amazon S3 URL에서 가져옵니다. 여기서, 'PK1'은 템플릿 콘텐츠에 선언된 파라미터의 이름을 나타내고, 'PV1'은 해당 파라미터의 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다. 스택 생성이 실패하면 롤백됩니다(-DisableRollback \$false를 지정하는 것과 동일). 지정된 알림 AEN은 게시된 스택 관련 이벤트를 수신합니다.

```
New-CFNStack -StackName "myStack" `
              -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
```

```
-Parameter @{ ParameterKey="PK1"; ParameterValue="PV1" } `
-NotificationARN @( "arn1", "arn2" )
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 [CreateStack](#)을 참조하세요.

Note

AWS CLI `--template-url` 옵션을 사용하여 Amazon S3 또는 AWS Systems Manager에서 템플릿 파일 위치를 지정할 수 있습니다.

Amazon S3의 경우 URL은 `https://`로 시작해야 합니다. S3 정적 웹 사이트 URL은 지원되지 않습니다.

```
--template-url https://s3.region-code.amazonaws.com/bucket-name/template-name
```

AWS Systems Manager의 경우 다음 형식을 사용합니다.

```
--template-url "ssm-doc://arn:aws:ssm:region-code:account-id:document/document-name"
```

변환이 포함된 스택 생성

[deploy](#) 명령을 사용하여 변환이 포함된 스택을 생성합니다. 변환이 포함된 템플릿에서 스택을 생성하는 경우 변경 세트를 사용해야 합니다. `deploy` 명령은 두 단계(변경 세트 생성 및 이를 실행)를 단일 명령으로 결합합니다.

AWS CLI

다음 `deploy` 명령에서는 지정된 이름, 템플릿 및 입력 파라미터를 사용하여 스택을 생성 합니다.

```
aws cloudformation deploy --stack-name myteststack \
  --template /path_to_template/my-template.json \
  --parameter-overrides Key1=Value1 Key2=Value2
```

스택 삭제

[delete-stack](#) 명령을 사용하여 스택을 삭제합니다. 자세한 내용은 [스택 삭제](#) 섹션을 참조하세요.

CLI

AWS CLI

스택을 삭제하는 방법

다음 `delete-stack` 예제에서는 지정된 스택을 삭제합니다.

```
aws cloudformation delete-stack \  
  --stack-name my-stack
```

이 명령은 출력을 생성하지 않습니다.

- API 세부 정보는 AWS CLI 명령 참조의 [DeleteStack](#)을 참조하세요.

PowerShell

Tools for PowerShell V4

예 1: 지정된 스택을 삭제합니다.

```
Remove-CFNStack -StackName "myStack"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 [DeleteStack](#)을 참조하세요.

Tools for PowerShell V5

예 1: 지정된 스택을 삭제합니다.

```
Remove-CFNStack -StackName "myStack"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 [DeleteStack](#)을 참조하세요.

삭제에 실패하고 `DELETE_FAILED` 상태가 반환되는 경우 `--deletion-mode` 옵션을 사용하여 스택 강제 삭제를 선택할 수 있습니다. `deletion-mode`에서 다음 값을 사용할 수 있습니다.

- `STANDARD`: 스택을 정상적으로 삭제합니다. 기본 삭제 모드입니다.
- `FORCE_DELETE_STACK`: 스택을 삭제하고 삭제에 실패한 모든 리소스를 건너뛵니다.

AWS CLI

다음 `delete-stack` 명령은 `deletion-mode` 파라미터가 포함된 `FORCE_DELETE_STACK` 값을 사용하여 *myteststack* 스택을 강제 삭제합니다.

```
aws cloudformation delete-stack --stack-name myteststack \
  --deletion-mode FORCE_DELETE_STACK
```

이 명령은 출력을 생성하지 않습니다.

`FORCE_DELETE_STACK` 사용 후 `list-stack-resources` 명령을 사용하여 스택 삭제 프로세스 중에 건너뛰었던 리소스를 나열할 수 있습니다. 유지된 리소스에는 `DELETE_SKIPPED` 상태가 표시됩니다. 자세한 내용은 [스택 리소스 나열](#) 섹션을 참조하세요.

스택 이벤트 설명

[describe-stack-events](#) 명령을 사용하여 스택 이벤트를 설명합니다. 자세한 내용은 [스택 진행 상황 모니터링](#) 섹션을 참조하세요.

CLI

AWS CLI

스택 이벤트를 설명하려면

다음 `describe-stack-events` 예제에서는 지정된 스택의 가장 최근 이벤트 2개를 표시합니다.

```
aws cloudformation describe-stack-events \
  --stack-name my-stack \
  --max-items 2

{
  "StackEvents": [
    {
      "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
      "EventId": "4e1516d0-e4d6-xmpl-b94f-0a51958a168c",
      "StackName": "my-stack",
      "LogicalResourceId": "my-stack",
      "PhysicalResourceId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
```

```

        "ResourceType": "AWS::CloudFormation::Stack",
        "Timestamp": "2019-10-02T05:34:29.556Z",
        "ResourceStatus": "UPDATE_COMPLETE"
    },
    {
        "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-
stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
        "EventId": "4dd3c810-e4d6-xmpl-bade-0aaf8b31ab7a",
        "StackName": "my-stack",
        "LogicalResourceId": "my-stack",
        "PhysicalResourceId": "arn:aws:cloudformation:us-
west-2:123456789012:stack/my-stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
        "ResourceType": "AWS::CloudFormation::Stack",
        "Timestamp": "2019-10-02T05:34:29.127Z",
        "ResourceStatus": "UPDATE_COMPLETE_CLEANUP_IN_PROGRESS"
    }
],
"NextToken": "eyJ0ZXh0VG9XMPLiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ=="
}

```

- API 세부 정보는 AWS CLI 명령 참조의 [DescribeStackEvents](#) 섹션을 참조하세요.

PowerShell

Tools for PowerShell V4

예 1: 지정된 스택에 대한 모든 스택 관련 이벤트를 반환합니다.

```
Get-CFNStackEvent -StackName "myStack"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 [DescribeStackEvents](#)를 참조하세요.

Tools for PowerShell V5

예 1: 지정된 스택에 대한 모든 스택 관련 이벤트를 반환합니다.

```
Get-CFNStackEvent -StackName "myStack"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 [DescribeStackEvents](#)를 참조하세요.

스택 리소스 설명

CLI

AWS CLI

스택 리소스에 대한 정보를 가져오려면

다음 `describe-stack-resource` 예제에서는 지정된 스택의 이름이 `MyFunction`인 리소스에 대한 세부 정보를 표시합니다.

```
aws cloudformation describe-stack-resource \
  --stack-name MyStack \
  --logical-resource-id MyFunction
```

출력:

```
{
  "StackResourceDetail": {
    "StackName": "MyStack",
    "StackId": "arn:aws:cloudformation:us-east-2:123456789012:stack/MyStack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
    "LogicalResourceId": "MyFunction",
    "PhysicalResourceId": "my-function-SEZV4XMPL4S5",
    "ResourceType": "AWS::Lambda::Function",
    "LastUpdatedTimestamp": "2019-10-02T05:34:27.989Z",
    "ResourceStatus": "UPDATE_COMPLETE",
    "Metadata": "{}",
    "DriftInformation": {
      "StackResourceDriftStatus": "IN_SYNC"
    }
  }
}
```

- API 세부 정보는 AWS CLI 명령 참조의 [DescribeStackResource](#) 섹션을 참조하세요.

PowerShell

Tools for PowerShell V4

예 1: 논리적 ID "MyDBInstance"로 지정된 스택과 연결된 템플릿에서 식별된 리소스의 설명을 반환합니다.

```
Get-CFNStackResource -StackName "myStack" -LogicalResourceId "MyDBInstance"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 [DescribeStackResource](#)를 참조하세요.

Tools for PowerShell V5

예 1: 논리적 ID "MyDBInstance"로 지정된 스택과 연결된 템플릿에서 식별된 리소스의 설명을 반환합니다.

```
Get-CFNStackResource -StackName "myStack" -LogicalResourceId "MyDBInstance"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 [DescribeStackResource](#)를 참조하세요.

스택 리소스 설명

CLI

AWS CLI

스택 리소스에 대한 정보를 가져오려면

다음 `describe-stack-resources` 예제에서는 지정된 스택의 리소스에 대한 세부 정보를 표시합니다.

```
aws cloudformation describe-stack-resources \
  --stack-name my-stack
```

출력:

```
{
  "StackResources": [
    {
      "StackName": "my-stack",
      "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
      "LogicalResourceId": "bucket",
      "PhysicalResourceId": "my-stack-bucket-1vc62xmplgguf",
      "ResourceType": "AWS::S3::Bucket",
      "Timestamp": "2019-10-02T04:34:11.345Z",
    }
  ]
}
```

```

        "ResourceStatus": "CREATE_COMPLETE",
        "DriftInformation": {
            "StackResourceDriftStatus": "IN_SYNC"
        }
    },
    {
        "StackName": "my-stack",
        "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
        "LogicalResourceId": "function",
        "PhysicalResourceId": "my-function-SEZV4XMPL4S5",
        "ResourceType": "AWS::Lambda::Function",
        "Timestamp": "2019-10-02T05:34:27.989Z",
        "ResourceStatus": "UPDATE_COMPLETE",
        "DriftInformation": {
            "StackResourceDriftStatus": "IN_SYNC"
        }
    },
    {
        "StackName": "my-stack",
        "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-stack/d0a825a0-e4cd-xmpl-b9fb-061c69e99204",
        "LogicalResourceId": "functionRole",
        "PhysicalResourceId": "my-functionRole-HIZXMPLE0M9E",
        "ResourceType": "AWS::IAM::Role",
        "Timestamp": "2019-10-02T04:34:06.350Z",
        "ResourceStatus": "CREATE_COMPLETE",
        "DriftInformation": {
            "StackResourceDriftStatus": "IN_SYNC"
        }
    }
]
}

```

- API 세부 정보는 AWS CLI 명령 참조의 [DescribeStackResources](#) 섹션을 참조하세요.

PowerShell

Tools for PowerShell V4

예 1: 지정된 스택과 연결된 최대 100개의 리소스에 대한 AWS 리소스 설명을 반환합니다. 스택과 연결된 모든 리소스의 세부 정보를 얻으려면 결과의 수동 페이징도 지원하는 Get-CFNStackResourceSummary를 사용하세요.


```
Get-CFNStackResourceList -StackName "myStack"
```

예 2: 논리적 ID "Ec2Instance"로 지정된 스택과 연결된 템플릿에서 식별된 Amazon EC2 인스턴스의 설명을 반환합니다.

```
Get-CFNStackResourceList -StackName "myStack" -LogicalResourceId "Ec2Instance"
```

예 3: 인스턴스 ID "i-123456"으로 식별되는 Amazon EC2 인스턴스를 포함하는 스택과 연결된 최대 100개의 리소스에 대한 설명을 반환합니다. 스택과 연결된 모든 리소스의 세부 정보를 얻으려면 결과의 수동 페이징도 지원하는 Get-CFNStackResourceSummary를 사용하세요.

```
Get-CFNStackResourceList -PhysicalResourceId "i-123456"
```

예 4: 논리적 ID "Ec2Instance"로 지정된 스택의 템플릿에서 식별된 Amazon EC2 인스턴스의 설명을 반환합니다. 스택은 포함된 리소스의 물리적 리소스 ID를 사용하여 식별됩니다. 이 경우에는 인스턴스 ID가 "i-123456"인 Amazon EC2 인스턴스입니다. 템플릿 콘텐츠에 따라 다른 물리적 리소스를 사용하여 스택을 식별할 수도 있습니다(예: Amazon S3 버킷).

```
Get-CFNStackResourceList -PhysicalResourceId "i-123456" -LogicalResourceId  
"Ec2Instance"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 [DescribeStackResources](#)를 참조하세요.

Tools for PowerShell V5

예 1: 지정된 스택과 연결된 최대 100개의 리소스에 대한 AWS 리소스 설명을 반환합니다. 스택과 연결된 모든 리소스의 세부 정보를 얻으려면 결과의 수동 페이징도 지원하는 Get-CFNStackResourceSummary를 사용하세요.

```
Get-CFNStackResourceList -StackName "myStack"
```

예 2: 논리적 ID "Ec2Instance"로 지정된 스택과 연결된 템플릿에서 식별된 Amazon EC2 인스턴스의 설명을 반환합니다.

```
Get-CFNStackResourceList -StackName "myStack" -LogicalResourceId "Ec2Instance"
```

예 3: 인스턴스 ID "i-123456"으로 식별되는 Amazon EC2 인스턴스를 포함하는 스택과 연결된 최대 100개의 리소스에 대한 설명을 반환합니다. 스택과 연결된 모든 리소스의 세부 정보를 얻으려면 결과의 수동 페이지도 지원하는 `Get-CFNStackResourceSummary`를 사용하세요.

```
Get-CFNStackResourceList -PhysicalResourceId "i-123456"
```

예 4: 논리적 ID "Ec2Instance"로 지정된 스택의 템플릿에서 식별된 Amazon EC2 인스턴스의 설명을 반환합니다. 스택은 포함된 리소스의 물리적 리소스 ID를 사용하여 식별됩니다. 이 경우에는 인스턴스 ID가 "i-123456"인 Amazon EC2 인스턴스입니다. 템플릿 콘텐츠에 따라 다른 물리적 리소스를 사용하여 스택을 식별할 수도 있습니다(예: Amazon S3 버킷).

```
Get-CFNStackResourceList -PhysicalResourceId "i-123456" -LogicalResourceId "Ec2Instance"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 [DescribeStackResources](#)를 참조하세요.

스택 설명

CLI

AWS CLI

AWS CloudFormation 스택을 설명하려면

다음 `describe-stacks` 명령에서는 `myteststack` 스택에 대한 요약 정보를 보여줍니다.

```
aws cloudformation describe-stacks --stack-name myteststack
```

출력:

```
{
  "Stacks": [
    {
      "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",
      "Description": "AWS CloudFormation Sample Template S3_Bucket: Sample template showing how to create a publicly accessible S3 bucket. **WARNING** This template creates an S3 bucket. You will be billed for the AWS resources used if you create a stack from this template.",
    }
  ]
}
```

```

        "Tags": [],
        "Outputs": [
            {
                "Description": "Name of S3 bucket to hold website content",
                "OutputKey": "BucketName",
                "OutputValue": "myteststack-s3bucket-jssofi1zie2w"
            }
        ],
        "StackStatusReason": null,
        "CreationTime": "2013-08-23T01:02:15.422Z",
        "Capabilities": [],
        "StackName": "myteststack",
        "StackStatus": "CREATE_COMPLETE",
        "DisableRollback": false
    }
}

```

자세한 내용을 알아보려면 AWS CloudFormation 사용 설명서의 스택을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 [DescribeStacks](#)를 참조하세요.

Go

SDK for Go V2

Note

GitHub에 더 많은 내용이 있습니다. [AWS 코드 예제 리포지토리](#)에서 전체 예제를 확인하고 설정 및 실행하는 방법을 알아보세요.

```

import (
    "context"
    "log"

    "github.com/aws/aws-sdk-go-v2/aws"
    "github.com/aws/aws-sdk-go-v2/service/cloudformation"
)

// StackOutputs defines a map of outputs from a specific stack.

```

```

type StackOutputs map[string]string

type CloudFormationActions struct {
    CfnClient *cloudformation.Client
}

// GetOutputs gets the outputs from a CloudFormation stack and puts them into a
// structured format.
func (actor CloudFormationActions) GetOutputs(ctx context.Context, stackName
    string) StackOutputs {
    output, err := actor.CfnClient.DescribeStacks(ctx,
        &cloudformation.DescribeStacksInput{
            StackName: aws.String(stackName),
        })
    if err != nil || len(output.Stacks) == 0 {
        log.Panicf("Couldn't find a CloudFormation stack named %v. Here's why: %v\n",
            stackName, err)
    }
    stackOutputs := StackOutputs{}
    for _, out := range output.Stacks[0].Outputs {
        stackOutputs[*out.OutputKey] = *out.OutputValue
    }
    return stackOutputs
}

```

- API 세부 정보는 AWS SDK for Go API 참조의 [DescribeStacks](#)를 참조하세요.

PowerShell

Tools for PowerShell V4

예 1: 모든 사용자 스택을 설명하는 스택 인스턴스 컬렉션을 반환합니다.

```
Get-CFNStack
```

예 2: 지정된 스택을 설명하는 스택 인스턴스를 반환합니다.

```
Get-CFNStack -StackName "myStack"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 [DescribeStacks](#)를 참조하세요.

Tools for PowerShell V5

예 1: 모든 사용자 스택을 설명하는 스택 인스턴스 컬렉션을 반환합니다.

```
Get-CFNStack
```

예 2: 지정된 스택을 설명하는 스택 인스턴스를 반환합니다.

```
Get-CFNStack -StackName "myStack"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 [DescribeStacks](#)를 참조하세요.

기본적으로 `describe-stacks` 명령은 파라미터값을 반환합니다. 암호와 같은 민감한 파라미터값이 반환되지 않도록 하려면 TRUE로 설정된 `NoEcho` 속성을 CloudFormation 템플릿에 포함합니다.

Important

`NoEcho` 속성을 사용해도 다음에 저장된 정보는 마스킹되지 않습니다.

- Metadata 템플릿 섹션. CloudFormation은 Metadata 섹션에 포함된 정보를 변환, 수정 또는 삭제하지 않습니다. 자세한 내용은 [메타데이터](#)를 참조하십시오.
- Outputs 템플릿 섹션. 자세한 내용은 [출력](#)을 참조하십시오.
- 리소스 정의의 Metadata 속성입니다. 자세한 내용은 [Metadata 속성](#)을 참조하세요.

이러한 메커니즘을 사용하여 암호나 보안 정보와 같은 중요한 정보를 포함하지 않는 것이 좋습니다.

Important

AWS Systems Manager Parameter Store 또는 AWS Secrets Manager와 같이 CloudFormation 외부에서 저장 및 관리되는 중요한 정보를 참조하려면 CloudFormation 템플릿에 직접 중요한 정보를 포함하는 대신 스택 템플릿에 있는 동적 파라미터를 사용하는 것이 좋습니다.

자세한 내용은 [템플릿에 자격 증명을 포함하지 않음](#)을 참조하십시오.

템플릿 가져오기

CLI

AWS CLI

AWS CloudFormation 스택의 템플릿 본문을 보려면

다음 `get-template` 명령에서는 `myteststack` 스택에 대한 템플릿을 보여줍니다.

```
aws cloudformation get-template --stack-name myteststack
```

출력:

```
{
  "TemplateBody": {
    "AWSTemplateFormatVersion": "2010-09-09",
    "Outputs": {
      "BucketName": {
        "Description": "Name of S3 bucket to hold website content",
        "Value": {
          "Ref": "S3Bucket"
        }
      }
    },
    "Description": "AWS CloudFormation Sample Template S3_Bucket: Sample
template showing how to create a publicly accessible S3 bucket. **WARNING** This
template creates an S3 bucket. You will be billed for the AWS resources used if
you create a stack from this template.",
    "Resources": {
      "S3Bucket": {
        "Type": "AWS::S3::Bucket",
        "Properties": {
          "AccessControl": "PublicRead"
        }
      }
    }
  }
}
```

- API 세부 정보는 AWS CLI 명령 참조의 [GetTemplate](#)을 참조하세요.

PowerShell

Tools for PowerShell V4

예 1: 지정된 스택과 연결된 템플릿을 반환합니다.

```
Get-CFNTemplate -StackName "myStack"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 [GetTemplate](#)을 참조하세요.

Tools for PowerShell V5

예 1: 지정된 스택과 연결된 템플릿을 반환합니다.

```
Get-CFNTemplate -StackName "myStack"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 [GetTemplate](#)을 참조하세요.

스택 리소스 나열

CLI

AWS CLI

스택의 리소스를 나열하려면

다음 명령은 지정된 스택의 리소스 목록을 표시합니다.

```
aws cloudformation list-stack-resources \  
  --stack-name my-stack
```

출력:

```
{  
  "StackResourceSummaries": [  
    {  
      "LogicalResourceId": "bucket",  
      "PhysicalResourceId": "my-stack-bucket-1vc62xmplgguf",
```

```

        "ResourceType": "AWS::S3::Bucket",
        "LastUpdatedTimestamp": "2019-10-02T04:34:11.345Z",
        "ResourceStatus": "CREATE_COMPLETE",
        "DriftInformation": {
            "StackResourceDriftStatus": "IN_SYNC"
        }
    },
    {
        "LogicalResourceId": "function",
        "PhysicalResourceId": "my-function-SEZV4XMPL4S5",
        "ResourceType": "AWS::Lambda::Function",
        "LastUpdatedTimestamp": "2019-10-02T05:34:27.989Z",
        "ResourceStatus": "UPDATE_COMPLETE",
        "DriftInformation": {
            "StackResourceDriftStatus": "IN_SYNC"
        }
    },
    {
        "LogicalResourceId": "functionRole",
        "PhysicalResourceId": "my-functionRole-HIZXMPLE0M9E",
        "ResourceType": "AWS::IAM::Role",
        "LastUpdatedTimestamp": "2019-10-02T04:34:06.350Z",
        "ResourceStatus": "CREATE_COMPLETE",
        "DriftInformation": {
            "StackResourceDriftStatus": "IN_SYNC"
        }
    }
]
}

```

- API 세부 정보는 AWS CLI 명령 참조의 [ListStackResources](#) 섹션을 참조하세요.

PowerShell

Tools for PowerShell V4

예 1: 지정된 스택과 연결된 모든 리소스에 대한 설명을 반환합니다.

```
Get-CFNStackResourceSummary -StackName "myStack"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 [ListStackResources](#)를 참조하세요.

Tools for PowerShell V5

예 1: 지정된 스택과 연결된 모든 리소스에 대한 설명을 반환합니다.

```
Get-CFNStackResourceSummary -StackName "myStack"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 [ListStackResources](#)를 참조하세요.

스택 나열

[list-stacks](#) 명령을 사용하여 스택을 나열합니다. 지정된 상태 코드가 있는 스택만 나열하려면 `--stack-status-filter` 옵션을 포함합니다. `--stack-status-filter` 옵션에 대해 하나 이상의 스택 상태 코드를 지정할 수 있습니다. 자세한 내용은 [스택 상태 코드](#) 섹션을 참조하세요.

CLI

AWS CLI

AWS CloudFormation 스택을 나열하려면

다음 `list-stacks` 명령에서는 상태가 `CREATE_COMPLETE`인 모든 스택에 대한 요약 내용을 보여줍니다.

```
aws cloudformation list-stacks --stack-status-filter CREATE_COMPLETE
```

출력:

```
[
  {
    "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
myteststack/466df9e0-0dff-08e3-8e2f-5088487c4896",
    "TemplateDescription": "AWS CloudFormation Sample Template S3_Bucket:
Sample template showing how to create a publicly accessible S3 bucket.
**WARNING** This template creates an S3 bucket. You will be billed for the AWS
resources used if you create a stack from this template.",
    "StackStatusReason": null,
    "CreationTime": "2013-08-26T03:27:10.190Z",
    "StackName": "myteststack",
    "StackStatus": "CREATE_COMPLETE"
```

```
}  
]
```

- API 세부 정보는 AWS CLI 명령 참조의 [ListStacks](#) 섹션을 참조하세요.

PowerShell

Tools for PowerShell V4

예 1: 모든 스택에 대한 요약 정보를 반환합니다.

```
Get-CFNStackSummary
```

예 2: 현재 생성 중인 모든 스택에 대한 요약 정보를 반환합니다.

```
Get-CFNStackSummary -StackStatusFilter "CREATE_IN_PROGRESS"
```

예 3: 현재 생성 중이거나 업데이트 중인 모든 스택에 대한 요약 정보를 반환합니다.

```
Get-CFNStackSummary -StackStatusFilter @("CREATE_IN_PROGRESS",  
"UPDATE_IN_PROGRESS")
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 [ListStacks](#)를 참조하세요.

Tools for PowerShell V5

예 1: 모든 스택에 대한 요약 정보를 반환합니다.

```
Get-CFNStackSummary
```

예 2: 현재 생성 중인 모든 스택에 대한 요약 정보를 반환합니다.

```
Get-CFNStackSummary -StackStatusFilter "CREATE_IN_PROGRESS"
```

예 3: 현재 생성 중이거나 업데이트 중인 모든 스택에 대한 요약 정보를 반환합니다.

```
Get-CFNStackSummary -StackStatusFilter @("CREATE_IN_PROGRESS",  
"UPDATE_IN_PROGRESS")
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 [ListStacks](#)를 참조하세요.

스택 업데이트

[update-stack](#) 명령을 사용하여 스택을 직접 업데이트합니다. 업데이트하려는 스택, 파라미터값 및 기능을 지정하고 업데이트된 템플릿을 사용하려는 경우 템플릿 이름을 지정합니다. 자세한 내용은 [직접 스택 업데이트](#) 섹션을 참조하세요.

CLI

AWS CLI

AWS CloudFormation 스택을 업데이트하려면

다음 update-stack 명령에서는 mystack 스택의 템플릿 및 입력 파라미터를 업데이트합니다.

```
aws cloudformation update-stack --stack-name mystack --  
template-url https://s3.amazonaws.com/sample/updated.template --  
parameters ParameterKey=KeyPairName,ParameterValue=SampleKeyPair  
ParameterKey=SubnetIDs,ParameterValue=SampleSubnetID1\\,SampleSubnetID2
```

다음 update-stack 명령에서는 mystack 스택의 SubnetIDs 파라미터값만 업데이트합니다. 파라미터값을 지정하지 않으면 템플릿에 지정된 기본값이 사용됩니다.

```
aws cloudformation update-stack --stack-name mystack --  
template-url https://s3.amazonaws.com/sample/updated.template  
--parameters ParameterKey=KeyPairName,UsePreviousValue=true  
ParameterKey=SubnetIDs,ParameterValue=SampleSubnetID1\\,UpdatedSampleSubnetID2
```

다음 update-stack 명령에서는 mystack 스택에 스택 알림 주제 2개를 추가합니다.

```
aws cloudformation update-stack --stack-name mystack --use-previous-template --  
notification-arns "arn:aws:sns:us-east-1:123456789012:mytopic1" "arn:aws:sns:us-east-1:123456789012:mytopic2"
```

자세한 내용을 알아보려면 AWS CloudFormation 사용 설명서의 [AWS CloudFormation 스택 업데이트](#)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 [UpdateStack](#)을 참조하세요.

PowerShell

Tools for PowerShell V4

예 1: 지정된 템플릿 및 사용자 지정 파라미터를 사용하여 'myStack' 스택을 업데이트합니다. 'PK1'은 템플릿에 선언된 파라미터의 이름을 나타내고 'PV1'은 해당 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다.

```
Update-CFNStack -StackName "myStack" `
    -TemplateBody "{Template Content Here}" `
    -Parameter @{ ParameterKey="PK1"; ParameterValue="PV1" }
```

예 2: 지정된 템플릿 및 사용자 지정 파라미터를 사용하여 'myStack' 스택을 업데이트합니다. 'PK1' 및 'PK2'는 템플릿에 선언된 파라미터의 이름을 나타내고, 'PV1' 및 'PV2'는 요청된 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다.

```
Update-CFNStack -StackName "myStack" `
    -TemplateBody "{Template Content Here}" `
    -Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
    @{ ParameterKey="PK2"; ParameterValue="PV2" } )
```

예 3: 지정된 템플릿 및 사용자 지정 파라미터를 사용하여 'myStack' 스택을 업데이트합니다. 'PK1'은 템플릿에 선언된 파라미터의 이름을 나타내고 'PV2'는 해당 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다.

```
Update-CFNStack -StackName "myStack" -TemplateBody "{Template Content Here}" -
Parameters @{ ParameterKey="PK1"; ParameterValue="PV1" }
```

예 4: Amazon S3에서 가져온 지정된 템플릿 및 사용자 지정 파라미터를 사용하여 'myStack' 스택을 업데이트합니다. 'PK1' 및 'PK2'는 템플릿에 선언된 파라미터의 이름을 나타내고, 'PV1' 및 'PV2'는 요청된 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다.

```
Update-CFNStack -StackName "myStack" `
    -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
    templatefile.template `
```

```
-Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
@{ ParameterKey="PK2"; ParameterValue="PV2" } )
```

예 5: Amazon S3에서 가져온 지정된 템플릿 및 사용자 지정 파라미터를 사용하여 이 예제에서 IAM 리소스를 포함하는 것으로 가정되는 'myStack' 스택을 업데이트합니다. 'PK1' 및 'PK2'는 템플릿에 선언된 파라미터의 이름을 나타내고, 'PV1' 및 'PV2'는 요청된 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다. IAM 리소스가 포함된 스택에서는 -Capabilities "CAPABILITY_IAM" 파라미터를 지정해야 합니다. 그렇지 않으면 업데이트가 실패하고 'InsufficientCapabilities' 오류가 발생합니다.

```
Update-CFNStack -StackName "myStack" `
-TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
-Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
@{ ParameterKey="PK2"; ParameterValue="PV2" } ) `
-Capabilities "CAPABILITY_IAM"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 [UpdateStack](#)을 참조하세요.

Tools for PowerShell V5

예 1: 지정된 템플릿 및 사용자 지정 파라미터를 사용하여 'myStack' 스택을 업데이트합니다. 'PK1'은 템플릿에 선언된 파라미터의 이름을 나타내고 'PV1'은 해당 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다.

```
Update-CFNStack -StackName "myStack" `
-TemplateBody "{Template Content Here}" `
-Parameter @{ ParameterKey="PK1"; ParameterValue="PV1" }
```

예 2: 지정된 템플릿 및 사용자 지정 파라미터를 사용하여 'myStack' 스택을 업데이트합니다. 'PK1' 및 'PK2'는 템플릿에 선언된 파라미터의 이름을 나타내고, 'PV1' 및 'PV2'는 요청된 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다.

```
Update-CFNStack -StackName "myStack" `
-TemplateBody "{Template Content Here}" `
-Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
@{ ParameterKey="PK2"; ParameterValue="PV2" } )
```

예 3: 지정된 템플릿 및 사용자 지정 파라미터를 사용하여 'myStack' 스택을 업데이트합니다. 'PK1'은 템플릿에 선언된 파라미터의 이름을 나타내고 'PV2'는 해당 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다.

```
Update-CFNStack -StackName "myStack" -TemplateBody "{Template Content Here}" -
Parameters @{ ParameterKey="PK1"; ParameterValue="PV1" }
```

예 4: Amazon S3에서 가져온 지정된 템플릿 및 사용자 지정 파라미터를 사용하여 'myStack' 스택을 업데이트합니다. 'PK1' 및 'PK2'는 템플릿에 선언된 파라미터의 이름을 나타내고, 'PV1' 및 'PV2'는 요청된 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다.

```
Update-CFNStack -StackName "myStack" `
                -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
                -Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
                              @{ ParameterKey="PK2"; ParameterValue="PV2" } )
```

예 5: Amazon S3에서 가져온 지정된 템플릿 및 사용자 지정 파라미터를 사용하여 이 예제에서 IAM 리소스를 포함하는 것으로 가정되는 'myStack' 스택을 업데이트합니다. 'PK1' 및 'PK2'는 템플릿에 선언된 파라미터의 이름을 나타내고, 'PV1' 및 'PV2'는 요청된 값을 나타냅니다. 사용자 지정 파라미터는 'ParameterKey' 및 'ParameterValue' 대신 'Key' 및 'Value'를 사용하여 지정할 수도 있습니다. IAM 리소스가 포함된 스택에서는 -Capabilities "CAPABILITY_IAM" 파라미터를 지정해야 합니다. 그렇지 않으면 업데이트가 실패하고 'InsufficientCapabilities' 오류가 발생합니다.

```
Update-CFNStack -StackName "myStack" `
                -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/
templatefile.template `
                -Parameter @( @{ ParameterKey="PK1"; ParameterValue="PV1" },
                              @{ ParameterKey="PK2"; ParameterValue="PV2" } ) `
                -Capabilities "CAPABILITY_IAM"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 [UpdateStack](#)을 참조하세요.

Note

모든 알림을 제거하려면 `--notification-arns` 옵션에 대해 `[]` 기호를 지정합니다.

템플릿 검증

[validate-template](#) 명령을 사용하여 템플릿 파일의 구문 오류를 점검합니다.

확인 중에 CloudFormation에서는 먼저 템플릿이 유효한 JSON인지를 확인합니다. 유효한 JSON이 아니면 CloudFormation은 템플릿이 유효한 YAML인지를 확인합니다. 두 확인이 모두 실패한 경우 CloudFormation에서는 템플릿 확인 오류를 반환합니다.

CLI

AWS CLI

AWS CloudFormation 템플릿의 유효성을 확인하려면

다음 `validate-template` 명령은 `sampletemplate.json` 템플릿의 유효성을 확인합니다.

```
aws cloudformation validate-template --template-body file://sampletemplate.json
```

출력:

```
{
  "Description": "AWS CloudFormation Sample Template S3_Bucket: Sample template
showing how to create a publicly accessible S3 bucket. **WARNING** This template
creates an S3 bucket. You will be billed for the AWS resources used if you
create a stack from this template.",
  "Parameters": [],
  "Capabilities": []
}
```

자세한 내용은 AWS CloudFormation 사용 설명서의 AWS CloudFormation 템플릿 사용을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 [ValidateTemplate](#)을 참조하세요.

PowerShell

Tools for PowerShell V4

예 1: 지정된 템플릿 콘텐츠의 유효성을 확인합니다. 출력에는 템플릿의 기능, 설명 및 파라미터가 자세히 설명되어 있습니다.

```
Test-CFNTemplate -TemplateBody "{TEMPLATE CONTENT HERE}"
```

예 2: Amazon S3 URL을 통해 액세스한 지정된 템플릿의 유효성을 확인합니다. 출력에는 템플릿의 기능, 설명 및 파라미터가 자세히 설명되어 있습니다.

```
Test-CFNTemplate -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/templatefile.template
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V4)의 [ValidateTemplate](#)을 참조하세요.

Tools for PowerShell V5

예 1: 지정된 템플릿 콘텐츠의 유효성을 확인합니다. 출력에는 템플릿의 기능, 설명 및 파라미터가 자세히 설명되어 있습니다.

```
Test-CFNTemplate -TemplateBody "{TEMPLATE CONTENT HERE}"
```

예 2: Amazon S3 URL을 통해 액세스한 지정된 템플릿의 유효성을 확인합니다. 출력에는 템플릿의 기능, 설명 및 파라미터가 자세히 설명되어 있습니다.

```
Test-CFNTemplate -TemplateURL https://s3.amazonaws.com/amzn-s3-demo-bucket/templatefile.template
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조(V5)의 [ValidateTemplate](#)을 참조하세요.

다음은 검증 오류가 발생하는 응답의 예입니다.

```
{
  "ResponseMetadata": {
    "RequestId": "4ae33ec0-1988-11e3-818b-e15a6df955cd"
  },

```



```

    "Errors": [
      {
        "Message": "Template format error: JSON not well-formed. (line 11, column
8)",
        "Code": "ValidationError",
        "Type": "Sender"
      }
    ],
    "Capabilities": [],
    "Parameters": []
  }
}
A client error (ValidationError) occurred: Template format error: JSON not well-formed.
(line 11, column 8)

```

Note

validate-template 명령은 템플릿의 구문만 점검하도록 설계되어 있습니다. 리소스에 대해 지정한 속성값이 해당 리소스에 유효한지 확인하지는 않습니다. 생성된 스택에 존재하게 될 리소스 수를 결정하지도 않습니다.

운영상 유효성을 점검하려면 스택을 만들어 보아야 합니다. CloudFormation 스택에는 샌드박스나 테스트 영역이 없기 때문에 테스트 중 만든 리소스는 본인이 담당해야 합니다.

AWS CLI를 사용하여 S3 버킷에 로컬 아티팩트 업로드

AWS CLI를 사용하여 CloudFormation 템플릿에서 참조되는 로컬 아티팩트를 Amazon S3 버킷에 업로드할 수 있습니다. 로컬 아티팩트는 템플릿에서 참조하는 파일입니다. 파일을 S3 버킷에 수동으로 업로드한 다음 템플릿에 위치를 추가하는 대신, 템플릿에서 로컬 아티팩트를 지정하고 [package](#) 명령을 사용하여 빠르게 업로드할 수 있습니다.

로컬 아티팩트는 package 명령을 통해 Amazon S3에 업로드하는 파일 또는 폴더의 경로입니다. 예를 들어 아티팩트는 AWS Lambda 함수의 소스 코드 또는 Amazon API Gateway REST API의 OpenAPI 파일에 대한 경로일 수 있습니다.

package 명령을 사용하는 경우:

- 파일을 지정하면 이 명령은 해당 파일을 S3 버킷에 직접 업로드합니다.
- 폴더를 지정하면 이 명령은 폴더에 대한 .zip 파일을 생성한 다음 .zip 파일을 업로드합니다.
- 경로를 지정하지 않으면 이 명령은 작업 디렉터리에 대한 .zip 파일을 생성하고 이 파일을 업로드합니다.

절대 경로 또는 상대 경로를 지정할 수 있습니다. 여기서 상대 경로는 템플릿 위치에 상대적입니다.

아티팩트를 업로드한 후 이 명령은 로컬 아티팩트에 대한 참조를 아티팩트를 업로드한 S3 위치로 바꾸어 템플릿의 사본을 반환합니다. 그런 다음 반환된 템플릿을 사용하여 스택을 생성하거나 업데이트할 수 있습니다.

Note

`package` 명령이 지원하는 리소스 속성에만 로컬 아티팩트를 사용할 수 있습니다. 이 명령 및 지원되는 리소스 속성 목록에 대한 자세한 내용을 알아보려면 [AWS CLI 명령 레퍼런스의 `package` 설명서](#)를 참조하세요.

사전 조건

시작하기 전에 기존 Amazon S3 버킷이 있어야 합니다.

로컬 아티팩트를 사용하여 템플릿 패키징 및 배포

다음 템플릿은 Lambda 함수의 소스 코드에 대한 로컬 아티팩트를 지정합니다. 소스 코드는 `/home/user/code/lambdafunction` 폴더에 저장됩니다.

원본 템플릿

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Transform": "AWS::Serverless-2016-10-31",
  "Resources": {
    "MyFunction": {
      "Type": "AWS::Serverless::Function",
      "Properties": {
        "Handler": "index.handler",
        "Runtime": "nodejs18.x",
        "CodeUri": "/home/user/code/lambdafunction"
      }
    }
  }
}
```

다음 [package](#) 명령은 함수의 소스 코드 폴더 `.zip` 파일을 지정된 버킷의 루트에 생성 및 업로드합니다.

```
aws cloudformation package \
  --s3-bucket amzn-s3-demo-bucket \
  --template /path_to_template/template.json \
  --output-template-file packaged-template.json \
  --output json
```

명령은 --output-template-file에 지정된 경로에서 새 템플릿을 생성합니다. 이는 아래와 같이 아티팩트 참조를 Amazon S3 위치로 바꿉니다. .zip 파일의 이름은 폴더 이름 자체를 사용하기보다는 폴더 콘텐츠의 MD5 체크섬을 사용하여 지정됩니다.

결과 템플릿

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Transform": "AWS::Serverless-2016-10-31",
  "Resources": {
    "MyFunction": {
      "Type": "AWS::Serverless::Function",
      "Properties": {
        "Handler": "index.handler",
        "Runtime": "nodejs18.x",
        "CodeUri": "s3://amzn-s3-demo-bucket/md5 checksum"
      }
    }
  }
}
```

템플릿의 아티팩트를 패키징한 후 [deploy](#) 명령을 사용하여 처리된 템플릿을 배포합니다.

```
aws cloudformation deploy \
  --template-file packaged-template.json \
  --stack-name stack-name
```

51,200바이트를 초과하는 템플릿을 배포할 때는 다음 예제와 같이 deploy 명령을 --s3-bucket 옵션과 함께 사용하여 템플릿을 S3에 업로드합니다.

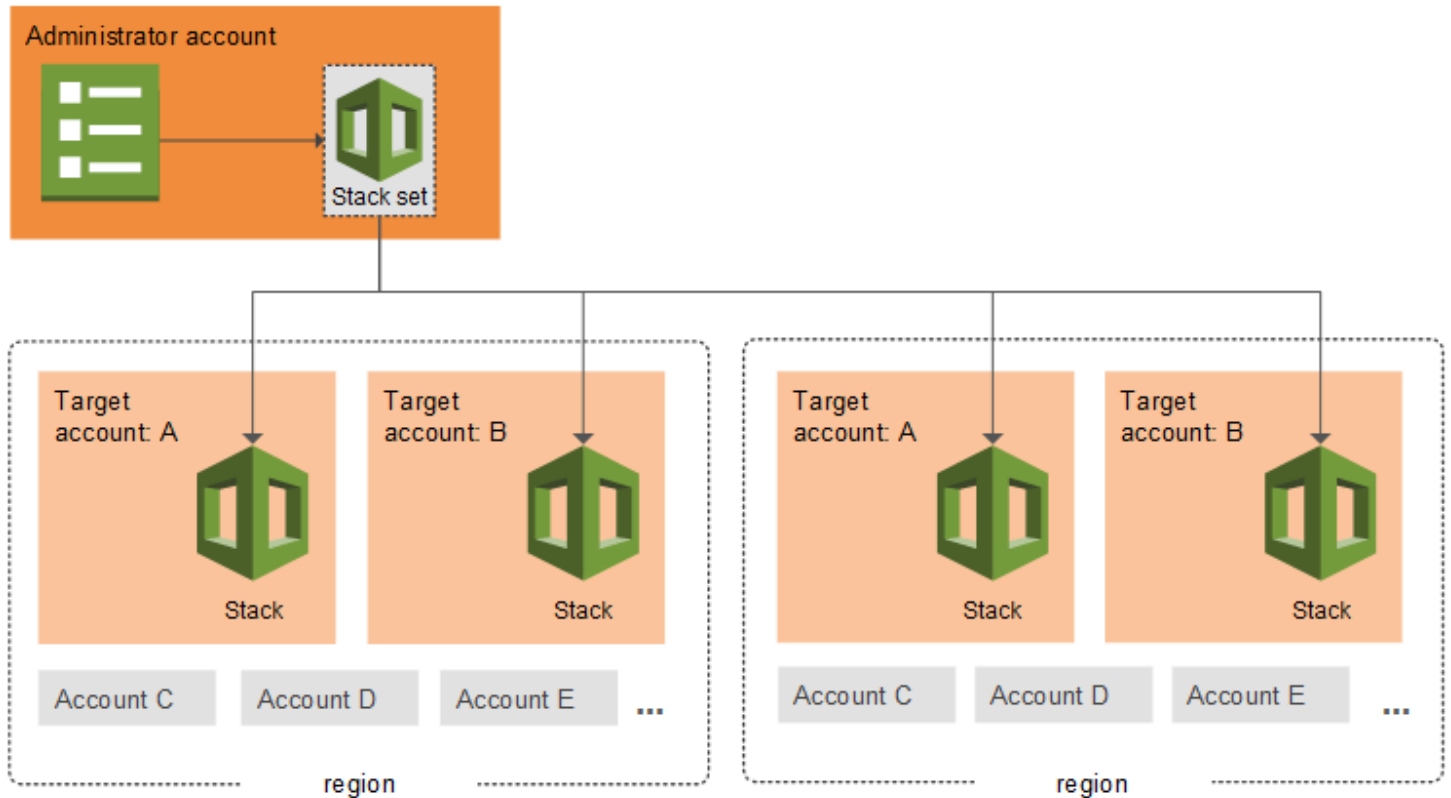
```
aws cloudformation deploy \
  --template-file packaged-template.json \
  --stack-name stack-name \
  --s3-bucket amzn-s3-demo-bucket
```

 Note

package 명령을 사용하여 로컬 아티팩트를 업로드하는 또 다른 예는 [중첩 스택을 사용하여 템플릿을 재사용 가능한 조각으로 분할](#)을 참조하세요.

StackSets를 사용하여 여러 계정 및 리전의 스택 관리

CloudFormation StackSets에서는 단일 작업으로 여러 계정과 AWS 리전에 대해 스택을 생성, 업데이트 또는 삭제할 수 있도록 하여 스택의 기능을 확장합니다. 관리자 계정을 사용하여 CloudFormation 템플릿을 정의 및 관리하고, 템플릿을 지정된 AWS 리전에 대해 선택한 대상 계정으로 스택을 프로비저닝하기 위한 기본으로 사용합니다.



이 섹션은 StackSet 사용을 시작하는 데 도움이 되며, StackSet 생성, 업데이트 및 삭제 작업을 수행하는 방법 및 문제 해결에 대한 일반적인 질문에 대한 답변을 제공합니다.

주제

- [StackSets 개념](#)
- [CloudFormation StackSet를 사용하기 위한 사전 조건](#)
- [샘플 템플릿을 사용하여 StackSet 시작하기](#)
- [자체 관리형 권한으로 CloudFormation StackSet 생성](#)
- [서비스 관리형 권한으로 CloudFormation StackSet 생성](#)
- [AWS Organizations에서 StackSets에 대한 자동 배포 활성화 또는 비활성화](#)
- [CloudFormation StackSet 업데이트](#)

- [CloudFormation StackSets에 스택 추가](#)
- [CloudFormation StackSet 내 스택에서 파라미터 값 재정의](#)
- [CloudFormation StackSet에서 스택 삭제](#)
- [CloudFormation StackSet 삭제](#)
- [대상 계정 게이트를 사용하여 StackSet 배포 실패 방지](#)
- [CloudFormation StackSet에 대한 동시성 모드 선택](#)
- [CloudFormation StackSets에서 드리프트 감지 수행](#)
- [CloudFormation StackSet로 스택 가져오기](#)
- [CloudFormation StackSet 사용 모범 사례](#)
- [CloudFormation StackSet 샘플 템플릿](#)
- [CloudFormation StackSets 문제 해결](#)

StackSets 개념

다음 용어와 개념은 StackSet의 이해와 사용에 매우 중요합니다.

주제

- [관리자 및 대상 계정](#)
- [CloudFormation StackSets](#)
- [StackSet에 대한 권한 모델](#)
- [스택 인스턴스](#)
- [StackSet 작업](#)
- [StackSet 작업 옵션](#)
- [Tags](#)
- [StackSets 상태 코드](#)
- [스택 인스턴스 상태 코드](#)

관리자 및 대상 계정

관리자 계정은 StackSet를 생성하는 AWS 계정입니다. 서비스 관리형 권한이 있는 StackSet의 경우 관리자 계정은 조직의 관리 계정 또는 위임된 관리자 계정입니다. StackSet를 생성한 AWS 관리자 계정으로 로그인하여 StackSet를 관리할 수 있습니다.

대상 계정은 StackSet에서 하나 이상의 스택을 생성, 업데이트 또는 삭제할 계정입니다. StackSet를 사용하여 대상 계정에 스택을 생성하기 전에 관리자 계정과 대상 계정 간에 신뢰 관계를 설정합니다.

CloudFormation StackSets

StackSet는 지정된 AWS 계정 및 리전에 배포되는 여러 스택의 컨테이너 역할을 합니다. 각 스택은 동일한 CloudFormation 템플릿을 기반으로 하지만 파라미터를 사용하여 개별 스택을 사용자 지정할 수 있습니다.

StackSet를 정의하면 지정한 대상 계정과 AWS 리전에서 스택을 생성, 업데이트 또는 삭제할 수 있습니다. 스택을 생성, 업데이트 또는 삭제할 때 작업 기본 설정을 지정할 수도 있습니다. 예를 들어 작업을 수행하려는 리전의 순서, 스택 작업이 중지되기 전의 내결함성 임계값, 스택 작업을 동시에 수행하는 계정 수를 포함시킵니다.

StackSet는 리전 리소스입니다. 한 AWS 리전에서 StackSet를 생성하면 해당 리전을 볼 때만 보거나 StackSet를 변경할 수 있습니다.

StackSet에 대한 권한 모델

자체 관리형 권한 또는 서비스 관리형 권한을 사용하여 StackSet를 생성할 수 있습니다.

자체 관리형 권한을 사용하면 StackSets가 계정 및 리전 전체에 배포하는 데 필요한 IAM 역할을 생성할 수 있습니다. 이러한 역할은 StackSet를 관리하는 계정과 스택 인스턴스를 배포하는 계정 간에 신뢰 관계를 설정하는 데 필요합니다. 이 권한 모델을 사용하여 StackSets에서는 IAM 역할을 생성할 수 있는 권한이 있는 모든 AWS 계정에 배포할 수 있습니다.

서비스 관리형 권한을 사용하면 AWS Organizations에서 관리하는 계정에 스택 인스턴스를 배포할 수 있습니다. 이 권한 모델을 사용하면 필요한 IAM 역할을 생성할 필요가 없습니다. StackSets가 사용자를 대신하여 IAM 역할을 생성합니다. 이 모델을 사용하면 나중에 조직에 추가되는 계정에 자동 배포를 설정할 수도 있습니다.

AWS Organizations는 CloudFormation과 통합되며 AWS 리소스 조정 및 확장 시 중앙에서 환경을 관리하고 통제하는 데 도움이 됩니다.

- 관리 계정은 조직을 생성할 때 사용하는 계정입니다. 자세한 내용은 [AWS Organizations 용어 및 개념](#)을 참조하세요.
- 위임된 관리자 - 호환되는 AWS 서비스는 조직의 AWS 멤버 계정을, 해당 서비스에서 조직 계정의 관리자로 등록할 수 있습니다. 자세한 내용을 알아보려면 [AWS Organizations로 작업하는 AWS 서비스](#)를 참조하세요.

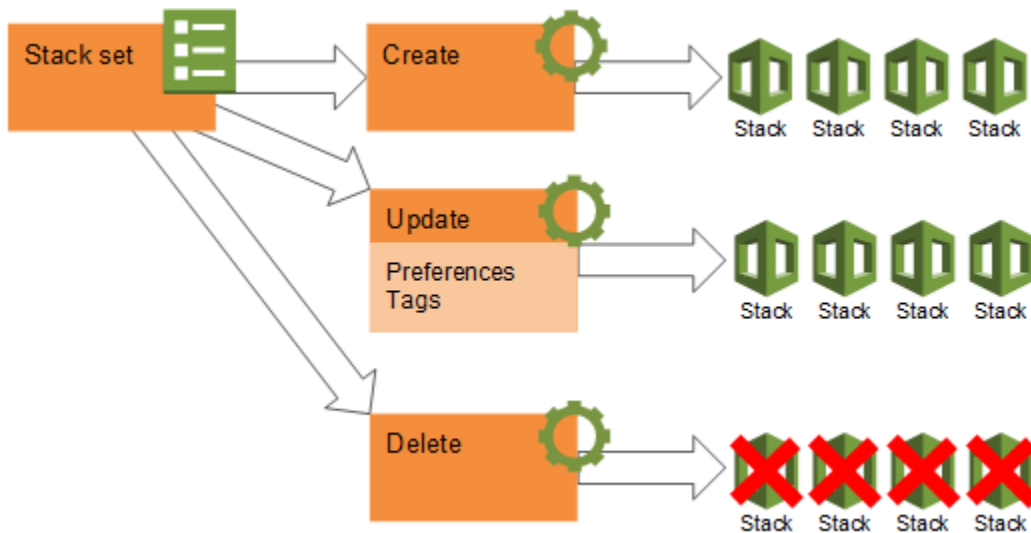
서비스 관리형 권한으로 StackSet을 생성하고 관리하는 방법에 대한 자세한 내용은 다음 주제를 참조하세요.

- [AWS Organizations를 사용하여 StackSet에 대한 신뢰할 수 있는 액세스 활성화](#)
- [위임된 관리자 멤버 계정 등록](#)
- [서비스 관리형 권한으로 CloudFormation StackSet 생성](#)

스택 인스턴스

스택 인스턴스는 리전 내 대상 계정에서 스택에 대한 참조입니다. 스택 인스턴스는 스택 없이 존재할 수 있습니다. 예를 들어 몇 가지 이유로 인해 스택을 생성할 수 없는 경우 스택 인스턴스에 스택 생성 실패에 대한 이유가 표시됩니다. 스택 인스턴스는 하나의 StackSet에만 연결됩니다.

다음 그림에서는 StackSet, 스택 작업 및 스택 간의 논리적 관계를 보여줍니다. StackSet를 업데이트하면 연결된 모든 스택 인스턴스가 모든 계정 및 리전에서 업데이트됩니다.



StackSet 작업

StackSet에 대해 다음 작업을 수행할 수 있습니다.

StackSet 생성

새 StackSet 생성에는 스택을 생성하는 데 사용할 CloudFormation 템플릿 지정, 스택을 생성할 대상 계정 지정 및 대상 계정의 스택을 배포할 AWS 리전 식별이 포함됩니다. StackSet를 사용하면 동일한 스택 리소스를 동일한 설정으로 선택한 리전 내의 지정된 모든 대상 계정에 일관되게 배포할 수 있습니다.

StackSet 업데이트

StackSet를 업데이트하면 StackSet의 스택으로 변경 사항을 푸시합니다. 다음 방법 중 하나로 StackSet를 업데이트할 수 있습니다. 템플릿 업데이트는 항상 모든 스택에 영향을 줍니다. StackSet의 일부 스택에 대한 템플릿을 선택적으로 업데이트할 수 없지만 그 이외에는 가능합니다.

- 템플릿의 기존 설정을 변경하거나 특정 서비스에 대한 파라미터 설정 업데이트 또는 새 Amazon EC2 인스턴스 추가 등의 새 리소스를 추가합니다.
- 템플릿을 다른 템플릿으로 바꿉니다.
- 기존 또는 추가 리전의 기존 또는 추가 대상 계정에 스택을 추가합니다.

스택 삭제

스택을 삭제하면 지정한 리전 내의 지정한 대상 계정에서 스택과 이와 관련된 모든 리소스가 제거됩니다. 스택을 삭제하는 방법은 다음과 같습니다.

- 일부 대상 계정에서 스택을 삭제하고 다른 대상 계정에서 다른 스택을 계속 실행합니다.
- 일부 리전에서 스택을 삭제하고 다른 리전에서 스택을 계속 실행합니다.
- StackSet에서 스택을 삭제하지만, 스택 보관 옵션을 선택하여 StackSet와 상관없이 스택이 계속 실행되도록 해당 스택을 저장합니다. 그런 다음 의 StackSet 외부의 CloudFormation에 보관된 스택을 관리할 수 있습니다.
- 전체 StackSet를 삭제하기 위해 StackSet에서 모든 스택을 삭제합니다.

StackSet 삭제

StackSet에 스택 인스턴스가 없는 경우에만 StackSet를 삭제할 수 있습니다.

StackSet 작업 옵션

이 섹션에 설명된 옵션은 성공적인 StackSet 작업을 수행하는 데 허용된 실패의 수와 시간을 제어하고 스택 리소스의 손실을 방지하는 데 도움이 됩니다.

최대 동시 계정

생성, 업데이트 및 삭제 워크플로에 사용 가능한 이 설정은 작업이 한 번에 수행되는 대상 계정의 최대 수 또는 비율을 지정할 수 있습니다. 낮은 수 또는 비율은 한 번에 더 적은 대상 계정에서 작업이 수행됨을 의미합니다. 작업은 Deployment order(배포 순서) 상자에 지정된 순서에 따라 한 번에 한 리전에서 수행됩니다. 예를 들어, 두 리전 내의 대상 계정 10개에 스택을 배포할 경우 Maximum concurrent accounts(최대 동시 계정)를 50 및 By percentage(백분율 기준)로 설정하면 첫 번째 리전의 계정 5개에 스택을 배포한 다음 첫 번째 계정 내의 나머지 계정 5개에 배포한 후, 다음 리전으로 이동하여 첫 번째 5개 대상 계정에 배포를 시작합니다.

[백분율 기준(By percentage)]을 선택할 때 지정된 비율이 지정된 계정의 전체 수를 나타내지 않으면 CloudFormation에서는 내림합니다. 예를 들어, 대상 계정 10개에 스택을 배포하고 [최대 동시 계정(Maximum concurrent accounts)]을 [25] 및 [백분율 기준(By percentage)]으로 설정하면 CloudFormation에서는 2.5개 스택을 동시에 배포하는 것(가능하지 않음)에서 2개 스택을 동시에 배포하는 것까지 내림합니다.

이 설정을 사용하면 작업에 대한 최대값을 지정할 수 있습니다. 대규모 배포의 경우 특정 상황에서 동시에 실행된 실제 계정 수는 서비스 조절로 인해 낮아질 수 있습니다.

최대 동시 계정은 동시성 모드에 따른 내결함성 값에 따라 달라질 수 있습니다. 동시성 모드가 엄격한 내결함성으로 설정된 경우 최대 동시 계정은 내결함성 설정보다 최대 하나 더 많을 수 있습니다.

동시성 모드

워크플로 생성, 업데이트 및 삭제 시 사용할 수 있는 이 설정을 사용하면 StackSet 작업 중 동시성 수준이 작동하는 방식을 선택할 수 있습니다. 자세한 내용은 [CloudFormation StackSet에 대한 동시성 모드 선택](#) 섹션을 참조하세요.

내결함성

생성, 업데이트 및 삭제 워크플로우에 사용 가능한 이 설정은 CloudFormation이 작업을 자동으로 중지하는 리전당 발생할 수 있는 스택 작업 실패의 최대 수 또는 비율을 지정할 수 있습니다. 낮은 수 또는 비율은 작업이 더 적은 스택에서 수행되지만 실패 작업의 문제 해결을 더 빠르게 시작할 수 있다는 의미입니다. 예를 들어, 리전 3개 내의 대상 계정 10개에서 스택 10개를 업데이트할 경우 내결함성을 20 및 백분율 기준(By percentage)으로 설정하면 작업을 계속하기 위해 리전에서 최대 2개의 스택 업데이트가 실패할 수 있습니다. 동일한 리전의 세 번째 스택이 실패할 경우 CloudFormation은 작업을 중지합니다. 첫 번째 리전에서 스택을 업데이트할 수 없는 경우 해당 리전에서 업데이트 작업이 계속되며 다음 리전으로 이동합니다. 두 번째 리전에서 스택 2개를 업데이트할 수 없는 경우 내결함성은 20%에 도달합니다. 리전에서 세 번째 스택이 실패하면 CloudFormation은 업데이트 작업을 중지하며, 후속 리전으로 이동하지 않습니다.

[백분율 기준(By percentage)]을 선택할 때 지정된 비율이 각 리전 내에서 스택의 전체 수를 나타내지 않으면 CloudFormation에서는 내림합니다. 예를 들어 3개 리전에서 대상 계정 10개에 스택을 배포하고 [내결함성(Failure tolerance)]을 [25] 및 [백분율 기준(By percentage)]으로 설정하면 CloudFormation에서는 2.5 스택의 내결함성(가능하지 않음)에서 리전당 2개 스택의 내결함성까지 내림합니다.

스택 보관

스택 삭제 워크플로에 사용 가능한 이 설정은 StackSet에서 스택을 제거한 후에도 스택 및 해당 리소스가 계속 실행되도록 할 수 있습니다. 스택을 보관할 경우 CloudFormation은 개별 계정 및 리전

의 스택을 그대로 유지합니다. 스택은 StackSet에서 연결 해제되지만 스택과 해당 리소스를 저장합니다. 스택 삭제 작업이 완료된 후 스택을 생성한 대상 계정(관리자 계정 아님)의 CloudFormation에서 보관된 스택을 관리합니다.

리전 동시성

워크플로 생성, 업데이트 및 삭제 시 사용할 수 있는 이 설정으로 StackSets가 리전에 배포되는 방식을 선택할 수 있습니다.

순차 - 리전의 배포 실패가 지정된 내결함성을 초과하지 않는 한 리전 배포 순서 상자에 지정된 대로 한 번에 한 리전에 StackSets 작업을 배포합니다. 기본 선택 항목은 [순차(Sequential)] 배포입니다.

병렬 - 리전의 배포 실패가 지정된 내결함성을 초과하지 않는 한 지정된 모든 리전에 StackSets 작업을 동시에 배포합니다.

Tags

키 및 값 페어를 지정하여 StackSet 생성 및 업데이트 작업 중 태그를 추가할 수 있습니다. 태그는 결제 및 비용 할당을 위해 StackSet 리소스를 정렬하고 필터링하는 데 유용합니다. AWS에서 태그를 사용하는 방법에 대한 자세한 내용은 AWS 결제 및 비용 관리 사용 설명서의 [Organizing and tracking costs using AWS cost allocation tags](#)를 참조하세요. 키-값 페어를 지정한 후에 +를 선택하여 태그를 저장합니다. 태그 오른쪽에 있는 빨간색 X를 선택하여 더 이상 사용하지 않는 태그를 삭제할 수 있습니다.

StackSet에 적용하는 태그는 모든 스택과 스택에서 생성하는 리소스에 적용됩니다. CloudFormation의 스택 전용 수준에서 태그를 추가할 수도 있지만 해당 태그는 StackSets에 표시되지 않을 수 있습니다.

StackSets는 시스템 정의 태그를 추가하지 않지만 태그의 키 이름을 문자열 aws:로 시작하면 안 됩니다.

StackSets 상태 코드

CloudFormation StackSet는 StackSet 작업에 대한 상태 코드를 생성합니다.

다음 표에서는 StackSet 작업에 대한 상태 코드를 설명합니다.

RUNNING

작업이 현재 진행되고 있습니다.

SUCCEEDED

작업이 작업에 대한 내결함성을 초과하지 않고 완료되었습니다.

FAILED

작업을 완료할 수 없는 스택의 수가 사용자 정의 내결함성을 초과했습니다. 작업에 대해 설정한 내결함성 값이 스택 생성 및 업데이트 작업 중 각 리전에 적용됩니다. 리전 내에서 실패한 스택의 수가 내결함성을 초과하는 경우 리전의 작업 상태가 FAILED로 변경됩니다. 작업의 전체 상태도 FAILED로 설정되며, CloudFormation은 나머지 리전에서 작업을 취소합니다.

QUEUED

[Service-managed permissions] 작업 시퀀스가 필요한 자동 배포의 경우 작업이 수행을 위해 대기열에 들어갑니다. 예제:

- 한 조직 단위(OU) OU1에서 다른 OU2로 계정을 이동하면 자동 배포가 트리거됩니다. StackSets는 삭제 작업을 실행하여 대상 리전의 대상 OU1 계정에서 스택 인스턴스를 제거하고, 대상 리전의 대상 OU2 계정에 스택 인스턴스를 추가하는 생성 작업을 대기열에 넣습니다.
- OU에 계정 AccountA를 추가하면 자동 배포가 트리거됩니다. StackSets는 대상 리전의 AccountA에 스택 인스턴스를 추가하는 생성 작업을 실행합니다. 이 생성 작업이 실행되는 동안 OU에 다른 계정 AccountB를 추가하면 StackSets가 두 번째 생성 작업을 대기열에 넣습니다. 첫 번째 생성 작업이 완료되면 StackSets가 두 번째 생성 작업을 실행하여 대상 리전의 AccountB에 스택 인스턴스를 추가합니다.

STOPPING

사용자 요청에 따라 작업이 중지되고 있습니다.

STOPPED

사용자 요청에 따라 작업이 중지되었습니다.

스택 인스턴스 상태 코드

CloudFormation StackSets는 스택 인스턴스에 대한 상태 코드를 생성합니다.

다음 표에서는 StackSets 내의 스택 인스턴스에 대한 상태 코드를 설명합니다.

CURRENT

스택이 StackSet와 함께 최신 상태입니다.

OUTDATED

다음 이유 중 하나로 인해 스택이 StackSet와 함께 최신 상태가 아닙니다.

- 연결된 스택에 대한 [CreateStackSet](#) 또는 [UpdateStackSet](#) 작업이 실패했습니다.
- 스택이 실패하거나 스택을 생성 또는 업데이트하기 전에 중지된 [CreateStackSet](#) 또는 [UpdateStackSet](#) 작업의 일부였습니다.

INOPERABLE

[DeleteStackInstances](#) 작업이 실패했으며 스택이 불안정한 상태로 남습니다. 이 상태의 스택은 향후 [UpdateStackSet](#) 작업에서 제외됩니다. RetainStacks가 true로 설정된 [DeleteStackInstances](#) 작업을 수행하여 스택 인스턴스를 삭제한 다음 스택을 수동으로 삭제해야 할 수 있습니다.

CANCELLED

지정된 계정 및 리전의 작업이 취소되었습니다. 이는 사용자가 StackSet 작업을 중지했거나 StackSet 작업이 내결함성을 초과했기 때문에 발생합니다.

FAILED

지정된 계정 및 리전의 작업이 실패했습니다. 한 리전 내의 충분한 계정에서 StackSet 작업이 실패하면 StackSet 작업 전체에 대한 내결함성을 초과될 수 있습니다.

FAILED_IMPORT

지정된 계정 및 리전에서 스택 인스턴스 가져오기가 실패하여 스택이 불안정한 상태가 되었습니다. 실패를 유발하는 문제가 해결되면 가져오기 작업을 다시 시도할 수 있습니다. 리전 내의 충분한 계정에서 충분한 StackSet 작업이 실패하면 StackSet 작업 전체에 대한 내결함성을 초과될 수 있습니다.

PENDING

지정된 계정 및 리전의 작업이 아직 시작되지 않았습니다.

RUNNING

지정된 계정 및 리전의 작업이 현재 진행되고 있습니다.

SKIPPED_SUSPENDED_ACCOUNT

작업할 때 계정이 일시 중지되었으므로 지정된 계정 및 리전의 작업을 건너뛰었습니다.

SUCCEEDED

지정된 계정 및 리전의 작업이 성공적으로 완료되었습니다.

CloudFormation StackSet를 사용하기 위한 사전 조건

StackSets는 단일 작업으로 여러 계정 및 리전에 대해 스택을 생성, 업데이트 또는 삭제할 수 있도록 하여 스택의 기능을 확장합니다.

StackSet는 여러 계정에서 스택 작업을 수행하기 때문에 첫 번째 StackSet를 생성하려면 먼저 AWS 계정에 필요한 권한을 정의해야 합니다.

자체 관리형 또는 서비스 관리형 권한을 사용하여 StackSets를 관리할 수 있습니다.

- 자체 관리형 StackSets의 경우 각 대상 계정과 AWS 리전에서 IAM 역할을 생성하고 관리해야 합니다. 자세한 내용은 [자체 관리형 권한 부여](#) 섹션을 참조하세요.
- 서비스 관리형 StackSets의 경우 각 계정에서 IAM 역할을 수동으로 생성하고 관리할 필요가 없습니다. AWS가 사용자를 대신하여 역할 생성 및 권한을 처리합니다. 자세한 내용은 [신뢰할 수 있는 액세스 스 활성화](#) 섹션을 참조하세요.

주제

- [기본적으로 비활성화된 AWS 리전에서 StackSet 작업 수행 준비하기](#)
- [자체 관리형 권한 부여](#)
- [AWS Organizations를 사용하여 StackSet에 대한 신뢰할 수 있는 액세스 활성화](#)

기본적으로 비활성화된 AWS 리전에서 StackSet 작업 수행 준비하기

아시아 태평양(홍콩)과 같이 2019년 3월 20일 이후에 도입된 AWS 리전은 기본적으로 비활성 상태입니다. 계정을 사용하려면 먼저 계정에 대해 이러한 리전을 활성화해야 합니다. 자세한 내용은 AWS Account Management 참조 안내서의 [계정에서 AWS 리전 활성화 또는 비활성화](#)를 참조하세요.

기본적으로 비활성화된 리전에서 StackSet의 관리자 계정(자체 관리형 권한을 사용할 경우) 또는 조직의 관리 계정(서비스 관리형 권한을 사용할 경우)에서 StackSet를 생성하려면 먼저 관리자 또는 관리 계정에 대해 해당 리전을 활성화해야 합니다.

CloudFormation이 스택 인스턴스를 성공적으로 생성/업데이트하려면:

- 대상 계정이 현재 해당 대상 계정에 활성화된 리전에 있어야 합니다.
- StackSet 관리자 계정 또는 조직 관리 계정은 동일한 리전을 대상 계정으로 활성화해야 합니다.

⚠ Important

StackSet 작업 중에 관리자 계정과 대상 계정은 관련 StackSet 및 StackSet 인스턴스에 더해 계정 자체에 대한 메타데이터도 교환합니다.

또한 StackSet 인스턴스가 있는 계정이 포함된 리전을 비활성화하는 경우 필요에 따라 해당 인스턴스 또는 리소스를 삭제해야 합니다. 아울러 비활성화된 리전의 대상 계정과 관련된 메타데이터는 관리자 계정에 보존됩니다.

자체 관리형 권한 부여

이 주제에서는 StackSets에서 자체 관리형 권한을 사용하여 계정 및 AWS 리전 전체에 배포하는 경우 필요한 IAM 서비스 역할을 생성하는 방법에 대한 지침을 제공합니다. 이러한 역할은 StackSet을 관리하는 계정과 스택 인스턴스를 배포하는 계정 간에 신뢰 관계를 설정하는 데 필요합니다. 이 권한 모델을 사용하여 StackSets에서는 IAM 역할을 생성할 수 있는 권한이 있는 모든 AWS 계정에 배포할 수 있습니다.

서비스 관리형 권한을 사용하려면 [신뢰할 수 있는 액세스 활성화](#) 섹션을 대신 참조하세요.

주제

- [자체 관리형 권한 개요](#)
- [관리자 계정의 모든 사용자에게 모든 대상 계정의 스택을 관리할 권한 제공](#)
- [StackSet 작업에 대한 고급 권한 옵션 설정](#)
- [혼란스러운 대리자 문제를 완화하기 위해 전역 키 설정](#)

자체 관리형 권한 개요

자체 관리형 권한으로 StackSet을 생성하기 전에 각 계정에 IAM 서비스 역할을 생성해야 합니다.

기본 단계는 다음과 같습니다.

1. 어느 AWS 계정이 관리자 계정인지 결정합니다.

이 관리자 계정에 StackSet가 생성됩니다. 대상 계정은 StackSet에 속하는 개별 스택이 생성되는 계정입니다.

2. StackSet에 대한 권한을 어떻게 구조화할지 결정합니다.

가장 단순한 (그리고 가장 허용적인) 권한 구성은 관리자 계정의 모든 사용자 및 그룹에게 해당 계정을 통해 관리하는 모든 StackSet를 생성하고 업데이트하는 기능을 제공하는 것입니다. 더욱 세분화된 제어가 필요한 경우 다음을 지정할 권한을 설정할 수 있습니다.

- 대상 계정의 StackSet 작업을 수행할 수 있는 사용자 및 그룹.
- StackSet에 포함될 수 있는 리소스 사용자 및 그룹.
- 특정 사용자 및 그룹에서 수행할 수 있는 StackSet 작업.

3. 관리자 및 대상 계정에 필요한 IAM 서비스 역할을 생성하여 원하는 권한을 정의합니다.

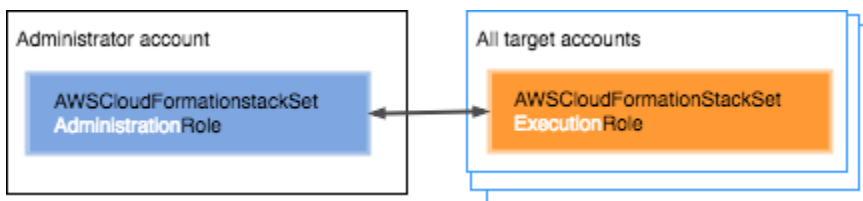
구체적으로 필요한 역할 2개는 다음과 같습니다.

- `AWSCloudFormationStackSetAdministrationRole` - 이 역할은 관리자 계정에 배포됩니다.
- `AWSCloudFormationStackSetExecutionRole` - 이 역할은 스택 인스턴스를 생성하는 모든 계정에 배포됩니다.

관리자 계정의 모든 사용자에게 모든 대상 계정의 스택을 관리할 권한 제공

이 섹션에서는 관리자 계정의 모든 사용자와 그룹이 모든 대상 계정에서 StackSet 작업을 허용하는 권한을 설정하는 방법을 보여줍니다. 관리자 및 대상 계정에 필요한 IAM 서비스 역할을 생성하는 과정을 안내합니다. 관리자 계정에 속한 누구나 모든 대상 계정에서 스택을 생성, 업데이트 또는 삭제할 수 있습니다.

이런 방식으로 권한을 구조화하면 사용자가 StackSet를 생성 또는 업데이트할 때 관리자 역할을 전달하지 않습니다.



Administrator account

관리자 계정에서 `AWSCloudFormationStackSetAdministrationRole`이라는 이름의 IAM 역할을 생성합니다.

<https://s3.amazonaws.com/cloudformation-stackset-sample-templates-us-east-1/>

[AWSCloudFormationStackSetAdministrationRole.yml](#)에서 사용할 수 있는 CloudFormation 템플릿에서 스택을 생성하면 가능합니다.

Example권한 정책 예

이전 템플릿에서 생성한 관리 역할에는 다음 권한 정책이 포함됩니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "sts:AssumeRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWSCloudFormationStackSetExecutionRole"
      ],
      "Effect": "Allow"
    }
  ]
}
```

Example신뢰 정책 예제 1

이전 템플릿에는 서비스에 관리자 역할을 사용할 수 있는 권한과 역할에 연결된 권한을 부여하는 다음 신뢰 정책도 포함되어 있습니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudformation.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Example 신뢰 정책 예제 2

기본적으로 비활성화되어 있는 리전에 상주하는 대상 계정에 스택 인스턴스를 배포하려면 해당 리전의 리전 서비스 보안 주체도 포함해야 합니다. 기본적으로 비활성화된 각 리전에는 자체 리전 서비스 보안 주체가 있습니다.

다음 신뢰 정책 예제는 기본적으로 비활성화되어 있는 리전인 아시아 태평양(홍콩) 리전(ap-east-1)의 관리자 역할을 사용할 수 있는 권한을 서비스에 부여합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "cloudformation.amazonaws.com",
          "cloudformation.ap-east-1.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

자세한 내용은 [기본적으로 비활성화된 AWS 리전에서 StackSet 작업 수행 준비하기](#) 섹션을 참조하세요. 리전 코드 목록은 AWS 일반 참조 안내서의 [리전 엔드포인트](#)를 참조하세요.

Target accounts

각 대상 계정에서 관리자 계정을 신뢰하는 `AWSCloudFormationStackSetExecutionRole01`라는 이름의 서비스 역할을 생성합니다. 역할에는 정확한 이름이 있어야 합니다.

<https://s3.amazonaws.com/cloudformation-stackset-sample-templates-us-east-1/AWSCloudFormationStackSetExecutionRole.yml>에서 사용할 수 있는 CloudFormation 템플릿에서 스택을 생성하면 가능합니다. 이 템플릿을 사용하면 대상 계정과 신뢰 관계가 있어야 하는 관리자 계정의 계정 ID를 제공하라는 메시지가 표시됩니다.

⚠ Important

이 템플릿은 관리자 액세스 권한을 부여하므로 유의하십시오. 템플릿을 사용하여 대상 계정 실행 역할을 생성한 후에는 정책 설명에 있는 권한의 범위를 StackSets를 사용하여 생성하는 리소스의 유형으로 지정해야 합니다.

대상 계정 서비스 역할에는 CloudFormation 템플릿에서 지정된 작업을 수행할 수 있는 권한이 필요합니다. 예를 들어, 템플릿이 S3 버킷을 생성할 경우 S3에 대한 새 객체를 생성할 수 있는 권한이 필요합니다. 대상 계정에는 항상 스택의 생성, 업데이트, 삭제, 설명을 비롯한 전체 CloudFormation 권한이 필요합니다.

Example 권한 정책 예 1

이 템플릿에 의해 생성된 역할을 사용하면 대상 계정에 대해 다음 정책을 사용할 수 있습니다.

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

Example 권한 정책 예 2

다음 예는 스택 세트가 작동하기 위한 최소 권한이 포함된 정책 설명을 보여줍니다. 대상 계정에서 CloudFormation 이외의 서비스에서 리소스를 사용하는 스택을 생성하려면 각 대상 계정에 대한 `AWSCloudFormationStackSetExecutionRole` 정책 설명에 해당 서비스 작업 및 리소스를 추가해야 합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```

    "Effect": "Allow",
    "Action": [
        "cloudformation:*"
    ],
    "Resource": "*"
  }
]
}

```

Example 신뢰 정책 예제

다음 신뢰 관계는 템플릿에 의해 생성됩니다. 관리자 계정의 ID는 *admin_account_id*로 표시됩니다.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}

```

관리자 계정의 특정 역할을 신뢰하도록 기존 대상 계정 실행 역할의 신뢰 관계를 구성할 수 있습니다. 관리자 계정에서 역할을 삭제하고 대체할 새 역할을 생성하는 경우, 이전 예에서 *admin_account_id*로 표현된 새 관리자 계정 역할과의 대상 계정 신뢰 관계를 구성해야 합니다.

StackSet 작업에 대한 고급 권한 옵션 설정

사용자 및 그룹이 단일 관리자 계정을 통해 생성한 StackSet에 대한 더욱 세분화된 제어가 필요한 경우 IAM 역할을 사용하여 다음을 지정할 수 있습니다.

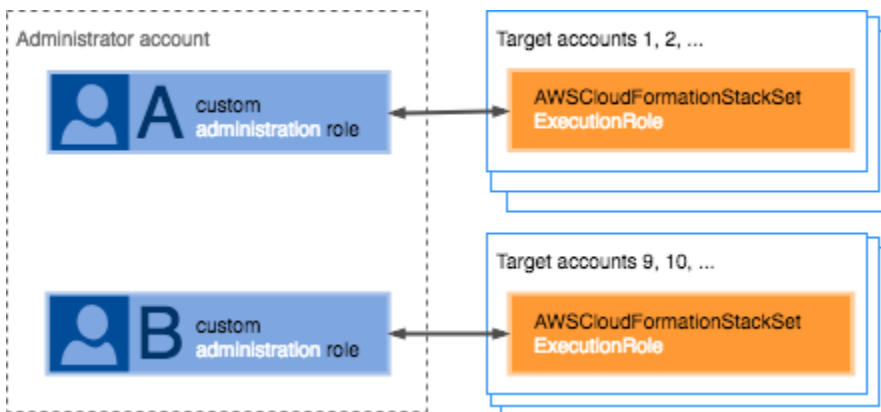
- 대상 계정의 StackSet 작업을 수행할 수 있는 사용자 및 그룹.
- StackSet에 포함될 수 있는 리소스 사용자 및 그룹.

- 특정 사용자 및 그룹에서 수행할 수 있는 StackSet 작업.

특정 대상 계정의 StackSet 작업을 수행할 수 있는 사용자 제어

사용자 지정 관리 역할을 사용하여 어떤 대상 계정에서 어떤 사용자 및 그룹이 StackSet 작업을 수행할 수 있는지 제어합니다. 어떤 대상 계정에서 어떤 관리자 계정의 사용자가 StackSet 작업을 수행할 수 있는지를 제어하고 싶을 수 있습니다. 이를 위해 관리자 계정 자체 내에 `AWSCloudFormationStackSetAdministrationRole` 서비스 역할을 생성하는 대신 각 대상 계정과 특정 사용자 지정 관리자 역할(관리자 계정 자체가 아닌) 간에 신뢰 관계를 생성합니다. 그런 다음 특정 대상 계정에서 StackSet 작업을 수행할 때 사용자 지정 관리 역할을 사용하도록 특정 사용자 및 그룹을 활성화합니다.

예를 들어 관리자 계정 내에 역할 A와 역할 B를 생성할 수 있습니다. 역할 A에 계정 8을 통해 대상 계정 1에 액세스할 수 있는 권한을 부여할 수 있습니다. 또한 역할 B에 계정 16을 통해 대상 계정 9에 액세스할 수 있는 권한을 부여할 수 있습니다.



필요한 권한을 설정하려면 사용자 지정 관리 역할을 정의하고, 대상 계정에 대한 서비스 역할을 생성하며, StackSet 작업을 수행할 때 사용자 지정 관리 역할을 전달할 수 있는 권한을 사용자에게 부여해야 합니다.

일반적으로 필요한 권한을 확보했으면 다음과 같이 진행됩니다. 사용자는 StackSet를 생성할 때 사용자 지정 관리 역할을 지정해야 합니다. 사용자는 CloudFormation에 역할을 전달할 수 있는 권한을 가지고 있어야 합니다. 뿐만 아니라 사용자 지정 관리 역할은 StackSet에 지정된 대상 계정과 신뢰 관계를 맺고 있어야 합니다. CloudFormation은 StackSet를 생성하고 사용자 지정 관리 역할을 여기에 연결합니다. StackSet를 업데이트할 때 사용자는 사용자 지정 관리 역할을 명시적으로 지정해야 합니다. 이전에 이 StackSet에 사용된 것과 동일한 사용자 지정 관리 역할인 경우에도 마찬가지입니다. CloudFormation은 위의 요구 사항에 따라 해당 역할을 사용하여 스택을 업데이트합니다.

Administrator account

Example 권한 정책 예

각 StackSet에 대해 대상 계정 실행 역할을 수임할 수 있는 권한을 가진 사용자 지정 관리 역할을 생성합니다.

대상 계정 실행 역할 이름은 모든 대상 계정에서 동일해야 합니다. 역할 이름이 `AWSCloudFormationStackSetExecutionRole`인 경우 StackSet는 StackSet를 생성할 때 이 역할을 자동으로 사용합니다. 사용자 지정 역할 이름을 지정하는 경우 사용자는 StackSet를 생성할 때 실행 역할 이름을 제공해야 합니다.

사용자 지정 이름과 다음 권한 정책을 가진 [IAM 서비스 역할](#)을 생성합니다. 아래 예제에서 `custom_execution_role`은 대상 계정의 실행 역할을 의미합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "sts:AssumeRole"
      ],
      "Resource": [
        "arn:aws:iam::111122223333:role/custom_execution_role"
      ],
      "Effect": "Allow"
    }
  ]
}
```

단일 문에서 여러 계정을 지정하려면 다음과 같이 쉼표로 구분합니다.

```
"Resource": [
  "arn:aws:iam::target_account_id_1:role/custom_execution_role",
  "arn:aws:iam::target_account_id_2:role/custom_execution_role"
]
```

계정 ID 대신 와일드카드(*)를 사용하여 모든 대상 계정을 지정할 수 있습니다.

```
"Resource": [
```

```

    "arn:aws:iam::*:role/custom_execution_role"
  ]

```

Example 신뢰 정책 예제 1

역할을 수임할 수 있는 IAM 보안 주체를 정의하려면 서비스 역할에 대한 신뢰 정책을 제공해야 합니다.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudformation.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}

```

Example 신뢰 정책 예제 2

기본적으로 비활성화되어 있는 리전에 상주하는 대상 계정에 스택 인스턴스를 배포하려면 해당 리전의 리전 서비스 보안 주체도 포함해야 합니다. 기본적으로 비활성화된 각 리전에는 자체 리전 서비스 보안 주체가 있습니다.

다음 신뢰 정책 예제는 기본적으로 비활성화되어 있는 리전인 아시아 태평양(홍콩) 리전(ap-east-1)의 관리자 역할을 사용할 수 있는 권한을 서비스에 부여합니다.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "cloudformation.amazonaws.com",
          "cloudformation.ap-east-1.amazonaws.com"
        ]
      }
    }
  ]
}

```

```

    },
    "Action": "sts:AssumeRole"
  }
]
}

```

자세한 내용은 [기본적으로 비활성화된 AWS 리전에서 StackSet 작업 수행 준비하기](#) 섹션을 참조하세요. 리전 코드 목록은 AWS 일반 참조 안내서의 [리전 엔드포인트](#)를 참조하세요.

Example역할 전달 정책 예제

사용자가 StackSet 작업을 수행할 때 사용자 지정 관리 역할 전달을 허용하는 IAM 사용자에게 대한 IAM 권한 정책도 필요합니다. 자세한 내용은 [사용자에게 AWS 서비스 역할을 전달할 수 있는 권한 부여](#)를 참조하세요.

아래 예제에서 *customized_admin_role*은 사용자가 전달해야 하는 관리 역할을 의미합니다.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:GetRole",
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::*:role/customized_admin_role"
    }
  ]
}

```

Target accounts

각 대상 계정에서 서비스 역할을 생성합니다. 이 서비스 역할은 이 계정에서 사용하고 싶은 사용자 지정 관리자 역할과 신뢰 관계를 맺고 있습니다.

대상 계정 역할에는 CloudFormation 템플릿에서 지정된 작업을 수행할 수 있는 권한이 필요합니다. 예를 들어, 템플릿이 S3 버킷을 생성할 경우 S3에서 새 객체를 생성할 수 있는 권한이 필요합니다. 대상 계정에는 항상 스택의 생성, 업데이트, 삭제, 설명을 비롯한 전체 CloudFormation 권한이 필요합니다.

대상 계정 역할 이름은 모든 대상 계정에서 동일해야 합니다. 역할 이름이 `AWSCloudFormationStackSetExecutionRole`인 경우 `StackSet`는 `StackSet`를 생성할 때 이 역할을 자동으로 사용합니다. 사용자 지정 역할 이름을 지정하는 경우 사용자는 `StackSet`를 생성할 때 실행 역할 이름을 제공해야 합니다.

Example권한 정책 예

다음 예는 스택 세트가 작동하기 위한 최소 권한이 포함된 정책 설명을 보여줍니다. 대상 계정에서 CloudFormation 이외의 서비스에서 리소스를 사용하는 스택을 생성하려면 권한 정책에 해당 서비스 작업 및 리소스를 추가해야 합니다.

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

Example신뢰 정책 예제

신뢰 관계를 정의하기 위해서는 역할을 생성할 때 다음과 같은 신뢰 정책을 제공해야 합니다.

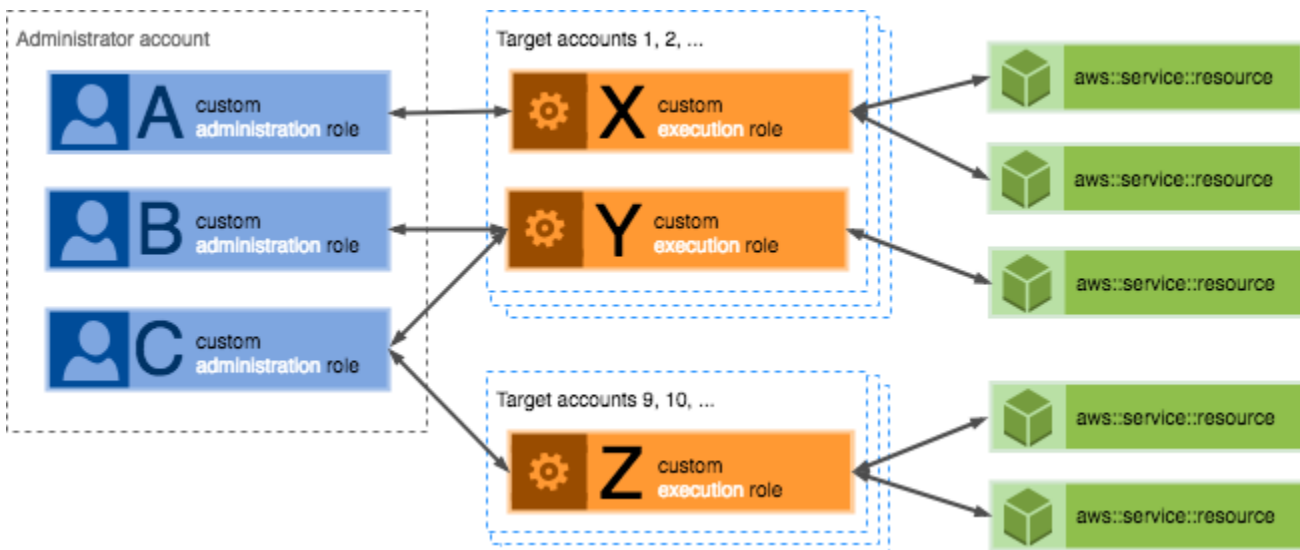
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/customized_admin_role"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

}

사용자가 특정 StackSet에 포함할 수 있는 리소스 제어

사용자 지정 실행 역할을 사용하여 사용자 및 그룹이 StackSet에 어떤 스택 리소스를 포함할 수 있는지를 제어합니다. 예를 들어 생성하는 StackSet에 Amazon S3 관련 리소스만 포함할 수 있도록 그룹을 설정하는 경우, 다른 팀에서는 DynamoDB 리소스만 포함하고자 할 수 있습니다. 이렇게 하려면 각 그룹에 대한 사용자 지정 관리 역할과 각 리소스 세트에 대한 사용자 지정 실행 역할 간의 신뢰 관계를 생성합니다. 사용자 지정 실행 역할은 StackSet에 어떤 스택 리소스를 포함할 수 있는지를 정의합니다. 사용자 지정 관리 역할은 관리자 계정 내에 상주하고, 사용자 지정 실행 역할은 정의된 리소스를 사용하여 StackSet를 생성하고자 하는 각 대상 계정 내에 상주합니다. 그런 다음 StackSet 작업을 수행할 때 사용자 지정 관리 역할을 사용하도록 특정 사용자 및 그룹을 활성화합니다.

예를 들어 관리자 계정 내에 사용자 지정 관리 역할 A, B, C를 생성할 수 있습니다. 역할 A를 사용할 수 있는 권한을 보유한 사용자 및 그룹은 사용자 지정 실행 역할 X에 나열된 스택 리소스를 포함하는 StackSet를 생성할 수 있지만 여기에 역할 Y나 Z에 나열된 스택 리소스 또는 모든 실행 역할에 포함되지 않은 리소스를 포함할 수는 없습니다.



StackSet를 업데이트할 때 사용자는 사용자 지정 관리 역할을 명시적으로 지정해야 합니다. 이전에 이 StackSet에 사용된 것과 동일한 사용자 지정 관리 역할인 경우에도 마찬가지입니다. CloudFormation은 사용자에게 해당 StackSet에 대한 작업을 수행할 수 있는 권한이 있는 한 지정된 사용자 지정 관리 역할을 사용하여 업데이트를 수행합니다.

마찬가지로 사용자는 사용자 지정 실행 역할 또한 지정할 수 있습니다. 사용자 지정 실행 역할을 지정하는 경우 CloudFormation은 위의 요구 사항에 따라 해당 역할을 사용하여 스택을 업데이트합니다. 사용자가 사용자 지정 실행 역할을 지정하지 않은 경우 CloudFormation은 이전에 StackSet에 연결된 사

용자 지정 실행 역할을 사용하여 업데이트를 수행합니다(단, 사용자가 StackSet에서 작업을 수행할 수 있는 권한을 가지고 있는 경우).

Administrator account

[특정 대상 계정의 StackSet 작업을 수행할 수 있는 사용자 제어](#)에서 자세히 설명하는 것과 같이 관리자 계정에 사용자 지정 관리 역할을 생성합니다. 사용자 지정 관리 역할과 사용하고자 하는 사용자 지정 실행 역할 간의 신뢰 관계를 포함합니다.

Example권한 정책 예

다음 예제는 사용자 지정 실행 역할은 물론 대상 계정에 대해 정의된 `AWSCloudFormationStackSetExecutionRole` 모두에 대한 권한 정책입니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1487980684000",
      "Effect": "Allow",
      "Action": [
        "sts:AssumeRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWSCloudFormationStackSetExecutionRole",
        "arn:aws:iam::*:role/custom_execution_role"
      ]
    }
  ]
}
```

Target accounts

StackSet를 생성하고자 하는 대상 계정에서 사용자 및 그룹에서 StackSet에 포함할 수 있는 서비스 및 리소스에 대한 권한을 부여하는 사용자 지정 실행 역할을 생성합니다.

Example권한 정책 예

다음 예는 Amazon DynamoDB 테이블 생성 권한이 포함된 StackSet에 대한 최소 권한을 제공합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:createTable"
      ],
      "Resource": "*"
    }
  ]
}
```

Example 신뢰 정책 예제

신뢰 관계를 정의하기 위해서는 역할을 생성할 때 다음과 같은 신뢰 정책을 제공해야 합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/customized_admin_role"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

특정 StackSet 작업에 대한 권한 설정

추가로 사용자 및 그룹에서 특정 StackSet 작업을 수행할 수 있는 권한을 설정할 수 있습니다(예: StackSet 또는 스택 인스턴스 생성, 업데이트 또는 삭제). 자세한 내용은 서비스 권한 부여 참조의 [CloudFormation에 사용되는 작업, 리소스 및 조건 키](#)를 참조하세요.

혼란스러운 대리자 문제를 완화하기 위해 전역 키 설정

혼동된 대리자 문제는 작업을 수행할 권한이 없는 엔터티가 권한이 더 많은 엔터티에게 작업을 수행하도록 강요할 수 있는 보안 문제입니다. AWS에서는 교차 서비스 가장으로 인해 혼동된 대리자 문제가 발생할 수 있습니다. 교차 서비스 가장은 한 서비스(직접적으로 호출하는 서비스)가 다른 서비스(직접적으로 호출되는 서비스)를 직접적으로 호출할 때 발생할 수 있습니다. 호출하는 서비스는 다른 고객의 리소스에 대해 액세스 권한이 없는 방식으로 작동하게 권한을 사용하도록 조작될 수 있습니다. 이를 방지하기 위해 AWS에서는 계정의 리소스에 대한 액세스 권한이 부여된 서비스 위탁자를 사용하여 모든 서비스에 대한 데이터를 보호하는 데 도움이 되는 도구를 제공합니다.

CloudFormation StackSet가 다른 서비스에 제공하는 리소스에 대한 권한을 제한하려면 리소스 정책에서 [aws:SourceArn](#) 및 [aws:SourceAccount](#) 글로벌 조건 컨텍스트 키를 사용하는 것이 좋습니다. 두 글로벌 조건 컨텍스트 키를 모두 사용하는 경우 `aws:SourceAccount`값과 `aws:SourceArn`값의 계정은 동일한 정책 문에서 사용할 경우 동일한 계정 ID를 사용해야 합니다.

혼동된 대리인 문제로부터 보호하는 가장 효과적인 방법은 리소스의 전체 ARN이 포함된 `aws:SourceArn`글로벌 조건 컨텍스트 키를 사용하는 것입니다. 리소스의 전체 ARN을 모를 경우 또는 여러 리소스를 지정하는 경우, ARN의 알 수 없는 부분에 대해 와일드카드(*)를 포함한 `aws:SourceArn`전역 조건 컨텍스트 키를 사용합니다. 예제: `arn:aws:cloudformation::123456789012:*`. 가능하면 더 구체적인 `aws:SourceArn`을 사용합니다. 올바른 ARN 또는 ARN 패턴을 결정할 수 없는 경우에만 `aws:SourceAccount`를 사용합니다.

StackSets가 administrator(관리자) 계정에서 Administration(관리) 역할을 맡으면 StackSets는 administrator(관리자) 계정 ID와 StackSets Amazon 리소스 이름(ARN)을 채웁니다. 따라서 [혼란스러운 대리자 문제](#)를 방지하기 위해 신뢰 관계에서 [전역 키](#) `aws:SourceAccount` 및 `aws:SourceArn`에 대한 조건을 정의할 수 있습니다. 다음 예제에서는 StackSets에서 `aws:SourceArn` 및 `aws:SourceAccount` 전역 조건 컨텍스트 키를 사용하여 혼동된 대리자 문제를 방지하는 방법을 보여줍니다.

Administrator account

Example **aws:SourceAccount** 및 **aws:SourceArn**에 대한 전역 키

StackSets를 사용할 때 AWSCloudFormationStackSetAdministrationRole 신뢰 정책에 전역 키 **aws:SourceAccount** 및 **aws:SourceArn**을 정의하여 혼동된 대리자 문제를 방지합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudformation.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "111122223333"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:cloudformation:*:111122223333:stackset/*"
        }
      }
    }
  ]
}
```

Example StackSets ARN

더 세밀하게 제어할 수 있도록 연결된 StackSets ARN을 지정합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
```

```

        "Service": "cloudformation.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
        "StringEquals": {
            "aws:SourceAccount": "111122223333",
            "aws:SourceArn": [
                "arn:aws:cloudformation:STACKSETS-  
REGION:111122223333:stackset/STACK-SET-ID-1",
                "arn:aws:cloudformation:STACKSETS-  
REGION:111122223333:stackset/STACK-SET-ID-2"
            ]
        }
    }
}

```

AWS Organizations를 사용하여 StackSet에 대한 신뢰할 수 있는 액세스 활성화

이 주제에서는 AWS Organizations에서 신뢰할 수 있는 액세스를 활성화하는 방법에 대한 지침을 제공하며, 이는 StackSets에서 서비스 관리형 권한을 사용하여 계정 및 AWS 리전 전체에서 배포하는 경우 필요합니다. 자체 관리형 권한을 사용하려면 [자체 관리형 권한 부여](#) 섹션을 대신 참조하세요.

서비스 관리형 권한으로 StackSet를 생성하기 전에 먼저 다음 작업을 완료해야 합니다.

- AWS Organizations의 [모든 기능을 활성화](#)합니다. 통합 결제 기능만 활성화하면 서비스 관리형 권한으로 StackSet를 생성할 수 없습니다.
- AWS Organizations를 사용하여 신뢰할 수 있는 액세스를 활성화합니다. 이 작업을 통해 CloudFormation은 관리 계정에서 서비스 연결 역할을 생성할 수 있습니다. 신뢰할 수 있는 액세스가 활성화된 후 서비스 관리형 권한으로 StackSet를 생성하면 CloudFormation은 대상 멤버 계정에서 이름이 `stacksets-exec-*`인 서비스 역할 및 필요한 서비스 연결 역할을 모두 생성합니다.

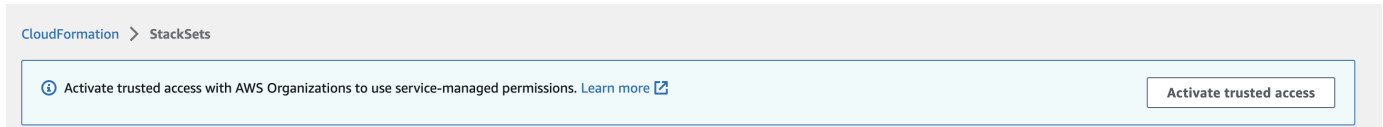
신뢰할 수 있는 액세스를 활성화하면 관리 계정 및 위임된 관리자 계정으로 조직의 서비스 관리형 StackSet를 생성하고 관리할 수 있습니다.

신뢰할 수 있는 액세스를 활성화하려면 관리 계정의 관리자 사용자여야 합니다. 관리자 사용자는 AWS 계정에 대한 모든 권한을 가진 사용자입니다. 자세한 내용은 AWS Account Management 참조 안내

서의 [관리자 사용자 생성하기](#)를 참조하세요. 관리 계정의 보안을 보호하기 위한 권장 사항은 AWS Organizations 사용 설명서의 [관리 계정의 모범 사례](#)를 참조하세요.

신뢰할 수 있는 액세스를 활성화하려면 다음을 수행하세요.

1. 관리 계정의 관리자로 AWS에 로그인하고 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 탐색 창에서 스택 세트를 선택합니다. 신뢰할 수 있는 액세스가 비활성화된 경우 신뢰할 수 있는 액세스를 활성화하라는 메시지가 배너에 표시됩니다.



3. 신뢰할 수 있는 액세스 활성화를 선택합니다.

다음 배너가 표시되면 신뢰할 수 있는 액세스가 성공적으로 활성화된 것입니다.



Note

조직 액세스 활성화는 조직 액세스 사용 설정과 동일하고 조직 액세스 비활성화는 조직 액세스 사용 중지와 동일합니다. 이 약관은 마케팅 지침에 따라 업데이트되었습니다.

신뢰할 수 있는 액세스를 비활성화하려면

AWS Organizations 사용 설명서의 [CloudFormation StackSets and AWS Organizations](#)를 참조하세요.

AWS Organizations에서 신뢰할 수 있는 액세스를 비활성화하기 전에 모든 위임된 관리자를 등록 취소해야 합니다. 자세한 내용은 [위임된 관리자 등록](#) 섹션을 참조하세요.

Note

콘솔 대신 API 작업을 사용하여 신뢰할 수 있는 액세스를 활성화하거나 비활성화하는 방법에 대한 자세한 내용은 다음을 참조하세요.

- [ActivateOrganizationsAccess](#)
- [DeactivateOrganizationsAccess](#)

- [DescribeOrganizationsAccess](#)

서비스 연결 역할

관리 계정은 `AWSServiceRoleForCloudFormationStackSetsOrgAdmin` 서비스 연결 역할을 사용합니다. AWS Organizations에서 신뢰할 수 있는 액세스가 비활성화된 경우에만 이 역할을 수정하거나 삭제할 수 있습니다.

각 대상 계정은 `AWSServiceRoleForCloudFormationStackSetsOrgMember` 서비스 연결 역할을 사용합니다. 이 역할은 두 가지 조건에서만 수정하거나 삭제할 수 있습니다. AWS Organizations에서 신뢰할 수 있는 액세스가 비활성화된 경우 또는 대상 조직이나 조직 단위(OU)에서 계정이 제거된 경우입니다.

자세한 내용은 AWS Organizations 사용 설명서의 [CloudFormation StackSets 및 AWS Organizations](#)를 참조하세요.

위임된 관리자 멤버 계정 등록

조직의 관리 계정에 더해 위임된 관리자 권한이 있는 멤버 계정으로 조직의 서비스 관리형 권한을 사용하여 StackSet을 생성하고 관리할 수 있습니다. 위임된 관리자가 생성한 StackSet을 포함하여 서비스 관리형 권한이 있는 StackSet가 관리 계정에 생성됩니다. 조직의 위임된 관리자로 등록하려면 해당 멤버 계정이 조직에 있어야 합니다. 조직 가입에 대한 자세한 내용은 [조직에 가입할 AWS 계정 초대](#)를 참조하세요.

조직에서 한 번에 최대 5명의 위임된 관리자를 등록할 수 있습니다. 위임된 관리자는 조직의 모든 계정 또는 특정 OU에 배포하도록 선택할 수 있습니다. 위임된 관리자가 Organizations에서 관리하는 계정에 배포하려면 먼저 AWS Organizations에서 신뢰할 수 있는 액세스를 활성화해야 합니다. 자세한 내용은 [AWS Organizations를 사용하여 StackSet에 대한 신뢰할 수 있는 액세스 활성화](#) 섹션을 참조하세요.

Important

다음에 유의하세요.

- 위임된 관리자는 조직의 계정에 배포할 수 있는 모든 권한을 가집니다. 관리 계정은 위임된 관리자 권한을 특정 OU에 배포하는 권한 또는 특정 StackSet 작업을 수행하는 권한으로 제한할 수 없습니다.
- 잠재적 오류를 방지하기 위해 위임된 관리자에게 `organizations:ListDelegatedAdministrators` 권한이 있는지 확인합니다.

미국 동부(오하이오), 미국 동부(버지니아 북부), 미국 서부(캘리포니아 북부), 미국 서부(오리건), 아시아 태평양(뭄바이), 아시아 태평양(서울), 아시아 태평양(싱가포르), 아시아 태평양(시드니), 아시아 태평양(도쿄), 캐나다(중부), 유럽(프랑크푸르트), 유럽(아일랜드), 유럽(런던), 유럽(파리), 유럽(스톡홀름), 이스라엘(텔아비브), 남아메리카(상파울루), AWS GovCloud(미국 동부) 및 AWS GovCloud(미국 서부) 리전에서 조직의 위임된 관리자를 등록할 수 있습니다.

[CloudFormation 콘솔](#), [AWS CLI](#) 또는 [AWS SDK](#)를 사용하여 위임된 관리자를 등록하고 등록 취소할 수 있습니다.

위임된 관리자를 추가하려면(콘솔)

1. 관리 계정의 관리자로 AWS에 로그인하고 <https://console.aws.amazon.com/cloudformation/>에서 CloudFormation 콘솔을 엽니다.
2. 탐색 창에서 스택 세트를 선택합니다.
3. [위임된 관리자(Delegated administrators)]에서 [위임된 관리자 등록(Register delegated administrator)]을 선택합니다.
4. [위임된 관리자 등록(Register delegated administrator)] 대화 상자에서 [위임된 관리자 등록(Register delegated administrator)]을 선택합니다.

성공 메시지는 멤버 계정이 위임된 관리자로 성공적으로 등록되었음을 나타냅니다.

위임된 관리자를 등록 취소하려면(콘솔)

1. 관리 계정의 관리자로 AWS에 로그인하고 <https://console.aws.amazon.com/>에서 CloudFormation 콘솔을 엽니다.
2. 탐색 창에서 스택 세트를 선택합니다.
3. [위임된 관리자(Delegated administrators)]에서 등록 취소할 계정을 선택한 다음 [등록 취소(Deregister)]를 선택합니다.

성공 메시지는 멤버 계정이 위임된 관리자로 성공적으로 등록 취소되었음을 나타냅니다.

언제든지 이 계정을 다시 등록할 수 있습니다.

위임된 관리자 등록 방법(AWS CLI)

1. AWS CLI를 엽니다.
2. `register-delegated-administrator` 명령을 실행합니다.

```
$ aws organizations register-delegated-administrator \  
  --service-principal=member.org.stacksets.cloudformation.amazonaws.com \  
  --account-id="memberAccountId"
```

3. `list-delegated-administrators` 명령을 실행하여 지정된 멤버 계정이 위임된 관리자로 성공적으로 등록되었는지 확인합니다.

```
$ aws organizations list-delegated-administrators \  
  --service-principal=member.org.stacksets.cloudformation.amazonaws.com
```

위임된 관리자를 등록 취소 방법(AWS CLI)

1. AWS CLI를 엽니다.
2. `deregister-delegated-administrator` 명령을 실행합니다.

```
$ aws organizations deregister-delegated-administrator \  
  --service-principal=member.org.stacksets.cloudformation.amazonaws.com \  
  --account-id="memberAccountId"
```

3. `list-delegated-administrators` 명령을 실행하여 지정된 멤버 계정이 위임된 관리자로 성공적으로 등록 취소되었는지 확인합니다.

```
$ aws organizations list-delegated-administrators \  
  --service-principal=member.org.stacksets.cloudformation.amazonaws.com
```

언제든지 이 계정을 다시 등록할 수 있습니다.

샘플 템플릿을 사용하여 StackSet 시작하기

이 자습서는 AWS Management Console을 사용하여 StackSet를 시작하는 데 도움이 됩니다. 샘플 템플릿을 사용하여 StackSet를 생성하는 프로세스를 안내합니다. 여러 리전에 스택을 배포하고, StackSet 작업을 모니터링하고, 결과를 확인하는 방법을 알아봅니다.

이 자습서에서는 미국 서부(오리건) 리전(us-west-2) 및 미국 동부(버지니아 북부) 리전(us-east-1) 내의 AWS 계정에서 AWS Config를 활성화하는 StackSet를 생성해 봅니다. 미국 동부(버지니아 북부) StackSet를 사용하면 단일 작업으로 여러 계정 및 리전에서 스택을 생성, 업데이트 또는 삭제할 수 있

으므로 대규모 인프라 관리에 이상적인 솔루션입니다. 이 자습서에서는 설명의 편의를 위해 단일 계정을 사용하지만, StackSet의 다중 리전 기능을 효과적으로 보여줍니다.

샘플 템플릿은 S3 버킷(<https://s3.amazonaws.com/cloudformation-stackset-sample-templates-us-east-1/EnableAWSConfig.yml>)에서 사용할 수 있습니다.

Note

StackSet 자체는 무료이지만 이 자습서의 AWS Config와 같이 StackSet로 생성한 AWS 리소스에 대해서는 요금이 부과됩니다. 자세한 내용은 [AWS Config 요금](#)을 참조하십시오.

주제

- [사전 조건](#)
- [콘솔에서 샘플 템플릿을 사용하여 StackSet 생성](#)
- [StackSet 생성 모니터링](#)
- [StackSet 결과 보기](#)
- [StackSet 업데이트](#)
- [StackSet에 스택 추가](#)
- [정리](#)
- [다음 단계](#)

사전 조건

자습서를 시작하기 전에 다음 사전 조건을 완료했는지 확인합니다.

- 자체 관리형 권한에 필요한 IAM 역할을 설정해야 합니다. StackSet을 생성하고 단일 계정 내에 스택을 배포하려면 계정에 다음 역할이 있어야 합니다.
 - `AWSCloudFormationStackSetAdministrationRole`
 - `AWSCloudFormationStackSetExecutionRole`

이러한 역할 설정에 대한 자세한 지침은 [자체 관리형 권한 부여](#) 섹션을 참조하세요.

콘솔에서 샘플 템플릿을 사용하여 StackSet 생성

AWS Config를 활성화하는 StackSet를 생성하려면

1. [CloudFormation 콘솔](#)을 엽니다.
2. 화면 상단의 탐색 모음에서 StackSet를 관리할 AWS 리전을 선택합니다.

StackSet를 지원하는 모든 리전을 선택할 수 있습니다. 선택한 리전은 StackSet를 사용하여 배포할 수 있는 리전에 영향을 주지 않습니다.
3. 탐색 창에서 스택 세트를 선택합니다.
4. StackSet 페이지의 상단에서 StackSet 만들기를 선택합니다.
5. 권한에서 셀프 서비스 권한을 선택하고 사전 작업에서 생성한 IAM 역할을 선택합니다.
 - IAM 관리자 역할에서 `AWSCloudFormationStackSetAdministrationRole`을 선택합니다.
 - IAM 실행 역할 이름에서 `AWSCloudFormationStackSetExecutionRole`을 선택합니다.
6. Prerequisite - Prepare template(사전 조건 - 템플릿 준비)에서 Use a sample template(샘플 템플릿 사용)을 선택하십시오.
7. 샘플 템플릿 선택 아래에서 AWS Config 활성화 템플릿을 선택합니다. 그리고 다음을 선택합니다.

이 템플릿은 구성 레코더, 전송 채널 등, 계정에서 AWS Config를 활성화하는 데 필요한 리소스를 생성합니다.
8. StackSet 세부 정보 지정 페이지의 StackSet 이름에 **my-awsconfig-stackset**를 입력합니다.
9. StackSet 설명에 **A StackSet that enables Config across multiple Regions**를 입력합니다.
10. 파라미터에서 다음과 같이 AWS Config 설정을 구성합니다.
 - a. 모든 리소스 유형 지원에서 지원되는 모든 리소스 유형을 기록하려면 기본값(true)을 유지하세요.
 - b. 글로벌 리소스 유형 포함에서 IAM 역할과 같은 글로벌 리소스를 제외하려면 기본값(false)를 유지하세요.
 - c. 모두 지원되지 않는 경우 리소스 유형 나열을 <All>로 설정된 상태로 두세요.
 - d. Config 서비스 연결 역할 리소스를 포함하는 리전의 경우 <DeployToAnyRegion>을 **us-west-2**로 바꾸세요.

즉, `AWSServiceRoleForConfig` 서비스 연결 역할은 스택이 미국 서부(오리건) 리전에 배포된 경우에만 생성됩니다. 이 절차의 뒷부분에서 배포 리전을 선택합니다.

- e. 구성 레코더 기록 빈도에서 매일 기록을 선택합니다.
11. 다음을 선택하여 계속 진행합니다.
 12. StackSet 옵션 구성 페이지에서 새 태그 추가를 선택하고 키 및 값 페어를 지정하여 태그를 추가합니다.
 - a. 키(Key)에 **Stage**를 입력합니다.
 - b. 값에 **Test**을(를) 입력합니다.

StackSet에 적용한 태그는 스택에 의해 생성된 리소스에 적용됩니다.

13. 실행 구성에서 활성을 선택하여 CloudFormation의 최적화된 작업 처리를 활성화합니다.
 - 배포 시간을 단축하기 위해, 서로 충돌하지 않는 작업이 동시에 실행됩니다.
 - 서로 충돌하는 작업은 요청된 순서대로 자동으로 대기열에 추가되고 처리됩니다.

실행 중이거나 대기 중인 작업이 있는 경우 CloudFormation은 충돌하지 않더라도 모든 수신 작업을 대기열에 넣습니다. 이 시간 동안에는 실행 설정을 변경할 수 없습니다.

14. 다음을 선택합니다.
15. 배포 옵션 설정 페이지의 StackSet에 스택 추가에서 새 스택 배포를 선택합니다.
16. 계정의 경우 Deploy stacks in accounts(계정에 스택 배포)를 선택하십시오.
17. 텍스트 상자에 AWS 계정 ID를 입력합니다.
18. 리전 지정에서 다음 리전을 다음 순서로 선택합니다.
 1. 미국 서부(오리건) 리전(us-west-2)
 2. 미국 동부(버지니아 북부) 리전(us-east-1)

필요한 경우, 미국 서부(오리건) 리전 옆에 있는 위쪽 화살표를 사용하여 목록의 첫 번째 항목으로 이동합니다. 리전 순서에 따라 배포 순서가 결정됩니다.

19. 배포 옵션에서 다음 설정을 구성합니다.
 - a. Maximum concurrent accounts(최대 동시 계정)의 경우 Number(번호) 및 1의 기본값을 유지합니다.

다중 계정 배포의 경우 이 설정을 사용하면 CloudFormation이 한 번에 한 계정에서만 스택을 배포합니다.

- b. Failure tolerance(내결함성)의 경우, 번호 및 0의 기본값을 유지하십시오.

즉, CloudFormation이 현재 리전에서 배포를 중단하고 나머지 리전에서 배포를 취소하기 전에 지정된 리전 중 하나에서 최대 0개의 스택 배포가 실패할 수 있습니다.

- c. 리전 동시성에서 순차적(기본값)을 선택합니다.

이 설정을 선택하면 CloudFormation이 한 리전에서 배포를 완료한 후에 다음 리전에서 배포를 시작합니다.

- d. 동시성 모드에서는 기본값인 엄격한 내결함성을 그대로 둡니다.

다중 계정 배포의 경우 실패 시 계정 동시성 수준이 감소하여 내결함성 +1 이내로 유지됩니다.

20. 다음을 선택합니다.

21. [검토] 페이지에서 선택 사항을 검토합니다. 변경하려면 관련 섹션에서 편집을 선택합니다.

22. StackSet를 생성할 준비가 되면 제출을 선택합니다.

StackSet 생성 모니터링

제출을 선택하면 CloudFormation이 StackSet를 생성하고 계정의 지정된 리전에 스택을 배포하기 시작합니다. 작업 진행 상황을 모니터링할 수 있는 StackSet 세부 정보 페이지가 자동으로 열립니다.

StackSet 생성을 모니터링하려면

1. StackSet 세부 정보 페이지에는 진행 중인 현재 작업을 보여주는 작업 탭이 기본적으로 표시됩니다.
2. 처음에 작업 상태는 RUNNING입니다. 사용자가 구성한 배포 옵션에 따라 CloudFormation이 지정된 리전에 스택을 생성합니다.
3. 작업에 대한 세부 정보를 보려면 목록에서 작업 ID를 선택합니다.
4. 작업 세부 정보 페이지에서 각 리전에서 생성 중인 스택 인스턴스의 상태를 볼 수 있습니다.
5. 작업 상태가 SUCCEEDED로 변경될 때까지 기다립니다. 이는 StackSet 및 모든 스택 인스턴스가 성공적으로 생성되었음을 나타냅니다.

StackSet 결과 보기

StackSet 생성이 완료되고 나면, 배포된 스택 인스턴스를 보고 지정한 리전의 계정에서 AWS Config가 활성화되었는지 확인할 수 있습니다.

StackSet 결과를 보려면

1. StackSet 세부 정보 페이지에서 스택 인스턴스 탭을 선택합니다.
2. 지정한 리전의 계정에 생성된 스택 인스턴스 목록이 표시되어야 합니다. 각 스택 인스턴스의 상태는 SUCCEEDED여야 하며, 이는 성공적으로 배포되었음을 나타냅니다.
3. 계정에서 AWS Config가 활성화되어 있는지 확인하려면 배포된 각 리전에서 AWS Config 콘솔을 확인하면 됩니다.

StackSet 업데이트

StackSet를 생성한 후 파라미터 값을 수정하거나 리전을 더 추가하여 업데이트할 수 있습니다. 이 섹션에서는 AWS Config 기록 빈도 파라미터를 업데이트하는 방법을 살펴보겠습니다.

StackSet를 업데이트하려면

1. StackSet 페이지에서 **my-awsconfig-stackset**를 선택합니다.
2. StackSet를 선택한 상태로 작업 메뉴에서 StackSet 세부 정보 편집을 선택합니다.
3. 템플릿 선택 페이지의 사전 조건 - 템플릿 준비에서 현재 템플릿 사용을 선택합니다.
4. 다음을 선택합니다.
5. StackSet 세부 정보 지정 페이지의 파라미터에서 구성 레코더 기록 빈도를 찾아 매일에서 계속으로 변경합니다.
6. 다음을 선택합니다.
7. StackSet 옵션 구성 페이지에서 설정을 그대로 두고 다음을 선택합니다.
8. 배포 옵션 설정 페이지에서 StackSet를 생성할 때 사용한 것과 동일한 리전과 계정 ID를 지정합니다.
9. 배포 옵션의 경우 이전과 동일한 설정을 유지합니다.
10. 다음을 선택합니다.
11. 검토 페이지에서 변경 내용을 검토하고 제출을 선택합니다.
12. CloudFormation이 StackSet 업데이트를 시작합니다. StackSet 세부 정보 페이지의 작업 탭에서 진행 상황을 모니터링할 수 있습니다.

StackSet에 스택 추가

새 리전에 추가로 배포하여 StackSet에 스택을 더 추가할 수 있습니다. 이 섹션에서는 새 리전에 스택을 추가하는 방법을 보여줍니다.

StackSet에 스택을 추가하려면

1. StackSet 페이지에서 **my-awsconfig-stackset**를 선택합니다.
2. StackSet를 선택한 상태로 작업 메뉴에서 StackSet에 스택 추가를 선택합니다.
3. 배포 옵션 설정 페이지의 StackSet에 스택 추가에서 새 스택 배포를 선택합니다.
4. 계정에서 계정에 스택 배포를 선택하고 계정 ID를 입력합니다.
5. 리전 지정에서 유럽(아일랜드)(eu-west-1)와 같은 새 리전을 선택합니다.
6. 배포 옵션의 경우 이전과 동일한 설정을 유지합니다.
7. 다음을 선택합니다.
8. 재정의 지정 페이지의 속성 값을 지정한 값으로 남겨두고 다음을 선택합니다.
9. 검토 페이지에서 선택 사항을 검토하고 제출을 선택하십시오.
10. CloudFormation이 지정한 리전에 새 스택을 생성하기 시작합니다. StackSet 세부 정보 페이지의 작업 탭에서 진행 상황을 모니터링할 수 있습니다.

정리

원치 않는 AWS Config 리소스에 대한 요금이 발생하지 않도록 하려면, StackSet에서 스택을 삭제하고, StackSet 자체를 삭제하고, 이 자습서에서 생성한 IAM 역할을 제거하여 정리해야 합니다. 모든 리소스는 사용자의 계정 내에서 배포되므로 간단히 정리할 수 있습니다.

StackSet에서 스택을 삭제하려면

1. StackSet 페이지에서 **my-awsconfig-stackset**를 선택합니다.
2. StackSet를 선택한 상태로 작업 메뉴에서 StackSet에서 스택 삭제를 선택합니다.
3. 배포 옵션 설정 페이지의 계정에서 계정에 스택 배포를 선택하고 계정 ID를 입력합니다.
4. 리전 지정에서 스택을 배포한 리전을 모두 선택합니다.
5. 배포 옵션의 경우 기본 설정을 유지합니다.
6. 스택과 해당 리소스가 삭제되도록, 스택 유지가 켜져 있지 않은지 확인합니다.
7. 다음을 선택합니다.

8. 검토 페이지에서 선택 사항을 검토하고 제출을 선택하십시오.
9. CloudFormation이 StackSet에서 스택을 삭제하기 시작합니다. StackSet 세부 정보 페이지의 작업 탭에서 진행 상황을 모니터링할 수 있습니다.

StackSet를 삭제하려면

1. 스택이 모두 삭제된 후 StackSet 페이지에서 **my-awsconfig-stackset**를 선택합니다.
2. StackSet를 선택한 상태로 작업 메뉴에서 StackSet 삭제를 선택합니다.
3. 확인하라는 메시지가 나타나면 삭제를 선택합니다.

IAM 서비스 역할을 삭제하려면

사용자의 계정에만 배포했으므로, 이 단일 계정에서 IAM 역할만 삭제하면 되고 다중 계정 배포보다 정리가 훨씬 간단합니다.

1. [IAM 콘솔](#)을 엽니다.
2. 탐색 창에서 역할을 선택합니다.
3. 검색 상자에 **AWSCloudFormationStackSet**를 입력하여 이 자습서에서 생성한 역할을 찾습니다.
4. AWSCloudFormationStackSetAdministrationRole 옆의 확인란을 선택합니다.
5. 페이지 상단에서 삭제를 선택합니다.
6. 확인 대화 상자에 **delete**를 입력한 다음 삭제를 선택합니다.
7. 동일한 프로세스를 반복하여 AWSCloudFormationStackSetExecutionRole을 삭제합니다.

StackSet를 삭제하고 나면, AWS::S3::Bucket 리소스의 DeletionPolicy 속성으로 인해 각 AWS 리전에 Amazon S3 버킷이 남아 있습니다. 따라서 AWS Config 기록 데이터가 보존됩니다. 이 데이터가 더 이상 필요하지 않은 경우 버킷을 수동으로 안전하게 삭제할 수 있습니다. 버킷을 삭제하려면 먼저 버킷을 비워야 합니다. 버킷을 비우면 버킷 내 모든 객체가 삭제됩니다.

Amazon S3 버킷을 비우고 삭제하려면

1. [Amazon S3 콘솔](#)을 엽니다.
2. 콘솔 왼쪽의 탐색 창에서 버킷을 선택합니다.
3. 배포한 각 리전에서 이 StackSet에 대해 생성된 버킷이 버킷 목록에 표시됩니다. 이 StackSet에 대해 생성된 버킷의 이름 옆에 있는 옵션을 선택한 다음 비우기를 선택합니다.

4. 버킷 비우기 페이지에서 텍스트 필드에 **permanently delete**를 입력하여 버킷을 비우려는 의사를 확인한 후 비우기를 선택합니다.
5. [버킷 비우기: 상태(Empty bucket: Status)] 페이지에서 버킷 비우기 프로세스의 진행 상황을 모니터링합니다.
6. 버킷 목록으로 돌아가려면 종료를 선택합니다.
7. 버킷의 이름 옆에 있는 옵션을 선택한 다음 삭제를 선택합니다.
8. 확인 메시지가 나타나면 버킷의 이름을 입력한 다음 버킷 삭제를 선택합니다.
9. 버킷 목록에서 버킷 삭제 프로세스의 진행 상황을 모니터링합니다. Amazon S3가 버킷 삭제를 완료하면 목록에서 해당 버킷을 제거합니다.
10. 다른 리전의 StackSet에 의해 생성된 각 버킷마다 이 프로세스를 반복합니다.

다음 단계

축하합니다! 샘플 템플릿을 사용하여 StackSet을 생성하고, 계정 내의 여러 리전에 스택을 배포하고, StackSet을 업데이트하고, 스택을 더 추가하고, 리소스를 정리했습니다. 단일 계정 배포에 집중함으로써, StackSet의 핵심 다중 리전 기능을 학습하기에 편리하도록 정리 프로세스를 간소화했습니다.

StackSet에 대해 자세히 알아보려면 다음 주제를 살펴보세요.

- [CloudFormation StackSet 내 스택에서 파라미터 값 재정의](#) - 특정 계정 및 리전의 파라미터 값을 재정의하는 방법을 알아봅니다.
- [서비스 관리형 권한으로 CloudFormation StackSet 생성](#) - AWS Organizations를 사용하여 다중 계정 배포를 위한 StackSet을 생성하는 방법을 살펴봅니다.

자체 관리형 권한으로 CloudFormation StackSet 생성

이 주제에서는 AWS 계정 및 리전에 스택을 배포할 자체 관리형 권한이 있는 StackSet을 생성하는 방법을 설명합니다.

Note

계속하려면 먼저 StackSet을 관리하는 계정과 스택을 배포할 계정 간에 신뢰할 수 있는 관계를 설정하기 위해 StackSet에 필요한 IAM 서비스 역할을 생성해야 합니다. 자세한 내용은 [자체 관리형 권한 부여](#) 섹션을 참조하세요.

주제

- [자체 관리형 권한으로 StackSet 생성\(콘솔\)](#)
- [자체 관리형 권한으로 StackSet 생성\(AWS CLI\)](#)

자체 관리형 권한으로 StackSet 생성(콘솔)

StackSet를 생성하려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
 2. 화면 상단의 탐색 모음에서 StackSet를 관리할 AWS 리전을 선택합니다.
 3. 탐색 창에서 스택 세트를 선택합니다.
 4. StackSet 페이지의 상단에서 StackSet 만들기를 선택합니다.
 5. 권한에서 셀프 서비스 권한을 선택하고 생성한 IAM 역할을 선택합니다.
 6. 사전 조건 - 템플릿 준비에서 템플릿 준비 완료를 선택합니다.
 7. 템플릿 지정, 아래에서 스택 템플릿을 포함하는 S3 버킷의 URL을 지정하거나 스택 템플릿 파일을 업로드하도록 선택합니다. 그리고 다음을 선택합니다.
 8. StackSet 세부 정보 지정 페이지에서 StackSet의 이름을 제공하고 파라미터를 지정한 후 다음을 선택합니다.
 9. 다음을 선택하여 계속 진행합니다.
 10. StackSet 옵션 구성 페이지의 태그 아래에서 스택의 리소스에 적용할 태그를 지정합니다. AWS에서 태그를 사용하는 방법에 대한 자세한 내용은 AWS 결제 및 비용 관리 사용 설명서의 [Organizing and tracking costs using AWS cost allocation tags](#)를 참조하세요.
 11. 실행 구성에서 활성을 선택하여 CloudFormation의 최적화된 작업 처리를 활성화합니다.
 - 배포 시간을 단축하기 위해, 서로 충돌하지 않는 작업이 동시에 실행됩니다.
 - 서로 충돌하는 작업은 요청된 순서대로 자동으로 대기열에 추가되고 처리됩니다.
- 실행 중이거나 대기 중인 작업이 있는 경우 CloudFormation은 충돌하지 않더라도 모든 수신 작업을 대기열에 넣습니다. 이 시간 동안에는 실행 설정을 변경할 수 없습니다.
12. 템플릿에 IAM 리소스가 포함되어 있는 경우 기능에서 이 템플릿이 IAM 리소스를 생성할 수 있음을 확인합니다를 선택하여 템플릿에서 IAM 리소스를 사용하도록 지정합니다. 자세한 내용은 [CloudFormation 템플릿에서 IAM 리소스 승인](#) 섹션을 참조하세요.

13. 다음을 선택합니다.
14. 배포 옵션 설정 페이지의 StackSet에 스택 추가에서 새 스택 배포를 선택합니다.
15. 계정의 경우 Deploy stacks in accounts(계정에 스택 배포)를 선택하십시오. 숫자 여러 개를 쉼표로 구분하여 대상 AWS 계정 번호를 텍스트 상자에 붙여 넣습니다.

 Note

관리자 계정도 스택을 배포하려는 경우 해당 계정 ID를 포함하면 됩니다.

16. 리전 지정에서 스택을 배포하려는 리전을 선택하세요.
17. 배포 옵션에서 다음을 수행하세요.
 - 최대 동시 계정에서 동시에 처리되는 계정 수를 지정하세요.
 - 내결함성에서 리전당 허용되는 최대 계정 실패 횟수를 지정합니다. 이 한도에 도달하면 작업이 중지되고 다른 리전으로 진행되지 않습니다.
 - 리전 동시성에서 리전을 처리하는 방법(순차: 한 번에 하나의 리전 또는 병렬: 동시에 여러 리전)을 선택하세요.
 - 동시성 모드에서 작업 실행 중에 동시성이 작동하는 방식을 선택하세요.
 - 엄격한 내결함성 - 내결함성 +1의 범위를 유지하면서 장애가 발생하는 경우 계정 동시성 수준을 줄입니다.
 - 소프트한 내결함성 - 장애에 관계없이 지정된 동시성 수준(최대 동시 계정의 값)을 유지합니다.
18. 다음을 선택합니다.
19. [검토] 페이지에서 선택 사항을 검토합니다. 변경하려면 관련 섹션에서 편집을 선택합니다.
20. StackSet를 생성할 준비가 되면 제출을 선택합니다.

CloudFormation이 StackSet 생성을 시작합니다. 제출을 선택하면 열리는 StackSet 세부 정보 페이지에서 StackSet의 스택 생성 진행률 및 상태를 봅니다.

자체 관리형 권한으로 StackSet 생성(AWS CLI)

이 섹션의 다음 단계를 따라 AWS CLI를 사용합니다.

- StackSet 컨테이너를 생성합니다.
- 스택 인스턴스를 배포합니다.

StackSet를 생성하려면

1. [create-stack-set](#) 명령을 사용하여 *my-stackset*라는 이름의 새 StackSet를 생성합니다. 다음 예에서는 S3 버킷에 저장된 템플릿을 사용하고, *KeyPairName* 값을 *TestKey*으로 설정하는 파라미터를 포함합니다.

```
aws cloudformation create-stack-set \
  --stack-set-name my-stackset \
  --template-url https://s3.region-code.amazonaws.com/amzn-s3-demo-bucket/MyApp.template \
  --parameters ParameterKey=KeyPairName,ParameterValue=TestKey
```

2. `create-stack-set` 명령이 완료되면 [list-stack-sets](#) 명령을 실행하여 StackSet가 생성되었는지 확인합니다. 결과에 새 StackSet가 표시됩니다.

```
aws cloudformation list-stack-sets
```

3. [create-stack-instances](#) 명령을 사용하여 StackSet 내에 스택을 배포합니다. 다음 예에서는 두 리전(*us-west-2* 및 *us-east-1*)의 두 AWS 계정(*account_ID_1* 및 *account_ID_2*)에 스택을 배포합니다.

--operation-preferences 옵션을 사용하여 동시 계정 처리 및 기타 배포 기본 설정을 지정하세요. 이 예제에서는 개수 기반 설정을 사용합니다. MaxConcurrentCount는 FailureToleranceCount + 1을 초과해서는 안 됩니다. 백분율 기반 설정의 경우 대신 FailureTolerancePercentage 또는 MaxConcurrentPercentage를 사용하세요.

```
aws cloudformation create-stack-instances \
  --stack-set-name my-stackset \
  --accounts account_ID_1 account_ID_2 \
  --regions us-west-2 us-east-1 \
  --operation-preferences MaxConcurrentCount=1,FailureToleranceCount=0
```

자세한 내용은 AWS CloudFormation API 참조의 [CreateStackInstances](#)를 참조하세요.

4. [describe-stack-set-operation](#) 명령을 사용하여, 스택이 성공적으로 생성되었는지 확인합니다. --operation-id 옵션에서, create-stack-instances 출력의 일부로 반환된 작업 ID를 지정합니다.

```
aws cloudformation describe-stack-set-operation \
  --stack-set-name my-stackset \
  --operation-id operation_ID
```

서비스 관리형 권한으로 CloudFormation StackSet 생성

서비스 관리형 권한을 사용하면 특정 리전에서 AWS Organizations가 관리하는 계정에 스택을 배포할 수 있습니다. 이 모델을 사용하면 각 대상 계정 및 AWS 리전에서 IAM 역할을 생성할 필요가 없습니다. CloudFormation이 사용자를 대신하여 IAM 역할을 생성합니다. 자세한 내용은 [신뢰할 수 있는 액세스 활성화](#) 섹션을 참조하세요.

주제

- [고려 사항](#)
- [서비스 관리형 권한으로 StackSet 생성\(콘솔\)](#)
- [서비스 관리형 권한으로 StackSet 생성\(AWS CLI\)](#)

고려 사항

서비스 관리형 권한으로 StackSet를 생성하기 전에 다음을 고려합니다.

- 서비스 관리형 권한이 있는 StackSet는 조직의 관리 계정 또는 위임된 관리자 계정에서 시작할 수 있지만 모든 작업은 관리 계정에서 수행합니다.
- CloudFormation은 해당 계정이 조직의 일부이거나 조직 단위(OU)에 속하더라도 관리 계정에 스택을 배포하지 않습니다.
- StackSet는 전체 조직(모든 계정 포함) 또는 지정된 OU를 대상으로 지정할 수 있습니다. StackSet가 상위 OU를 대상으로 지정하면 모든 하위 OU는 자동으로 포함됩니다. StackSet가 특정 OU 대상으로 지정하는 경우 해당 OU 내의 모든 계정이 기본적으로 포함됩니다. 단, 계정 필터 옵션을 사용하여 특정 계정을 대상으로 지정할 수 있습니다.
- 여러 StackSets가 동일한 조직 또는 OU를 대상으로 할 수 있습니다.
- StackSet는 조직 외부의 계정을 대상으로 지정할 수 없습니다.
- StackSet를 배포할 수 있는 권한은 관리 계정에 로그인하는 데 사용하는 IAM 위탁자(사용자, 역할 또는 그룹)의 권한에 따라 달라집니다. 조직에 배포할 권한을 부여하는 IAM 정책 예제는 [리전 및 리소스 유형을 기반으로 스택 세트 작업 제한](#) 단원을 참조하십시오.
- 위임된 관리자는 조직의 어떤 계정도 배포할 수 있는 전체 권한을 가집니다. 관리 계정은 위임된 관리자 권한을 특정 OU 또는 StackSet 작업에 배포하는 권한으로 제한할 수 없습니다.
- 자동 배포 설정은 StackSet 수준에서 적용됩니다. OU, 계정 또는 리전의 경우 자동 배포를 선택적으로 조정할 수 없습니다.
- 서비스 관리형 권한을 사용하는 StackSet는 매크로 또는 변환이 포함된 중첩 스택 또는 템플릿을 지원하지 않습니다.

서비스 관리형 권한으로 StackSet 생성(콘솔)

StackSet를 생성하려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 StackSet를 관리할 AWS 리전을 선택합니다.
3. 탐색 창에서 스택 세트를 선택합니다.
4. StackSet 페이지의 상단에서 StackSet 만들기를 선택합니다.
5. 권한에서 서비스 관리형 권한을 선택합니다.

Note

AWS Organizations에서 신뢰할 수 있는 액세스가 비활성화되면 배너가 표시됩니다. 서비스 관리형 권한으로 StackSet를 생성하거나 업데이트하려면 신뢰할 수 있는 액세스가 필요합니다. 조직 관리 계정의 관리자만 [AWS Organizations를 사용하여 StackSet에 대한 신뢰할 수 있는 액세스 활성화](#)에 대한 권한이 있습니다.

6. 사전 조건 - 템플릿 준비에서 템플릿 준비 완료를 선택합니다.
7. 템플릿 지정, 아래에서 스택 템플릿을 포함하는 S3 버킷의 URL을 지정하거나 스택 템플릿 파일을 업로드하도록 선택합니다. 그리고 다음을 선택합니다.
8. StackSet 세부 정보 지정 페이지에서 StackSet의 이름을 제공하고 파라미터를 지정한 후 다음을 선택합니다.
9. StackSet 옵션 구성 페이지의 태그 아래에서 스택의 리소스에 적용할 태그를 지정합니다. AWS에서 태그를 사용하는 방법에 대한 자세한 내용은 AWS 결제 및 비용 관리 사용 설명서의 [Organizing and tracking costs using AWS cost allocation tags](#)를 참조하세요.
10. 실행 구성에서 활성을 선택하여 CloudFormation의 최적화된 작업 처리를 활성화합니다.
 - 배포 시간을 단축하기 위해, 서로 충돌하지 않는 작업이 동시에 실행됩니다.
 - 서로 충돌하는 작업은 요청된 순서대로 자동으로 대기열에 추가되고 처리됩니다.

실행 중이거나 대기 중인 작업이 있는 경우 CloudFormation은 충돌하지 않더라도 모든 수신 작업을 대기열에 넣습니다. 이 시간 동안에는 실행 설정을 변경할 수 없습니다.

11. 템플릿에 IAM 리소스가 포함되어 있는 경우 기능에서 이 템플릿이 IAM 리소스를 생성할 수 있음을 확인합니다를 선택하여 템플릿에서 IAM 리소스를 사용하도록 지정합니다. 자세한 내용은 [CloudFormation 템플릿에서 IAM 리소스 승인](#) 섹션을 참조하세요.
12. 아직 활성화하지 않은 경우 계속해서 신뢰할 수 있는 액세스를 활성화하려면 다음을 선택합니다.
13. 배포 옵션 설정 페이지의 배포 대상에서 다음 중 하나를 수행합니다.

- 조직의 모든 계정에 배포하려면 조직에 배포를 선택합니다.
- 특정 OU의 모든 계정에 배포하려면 OU(조직 단위)에 배포를 선택합니다. OU 추가를 선택한 다음, 텍스트 상자에 대상 OU ID를 붙여 넣습니다. 각각의 새로운 대상 OU에 대해 반복합니다.

계정 필터 유형에서 조직 단위(OU)에 배포를 선택한 경우 다음 옵션 중 하나를 선택하고 계정 번호를 제공하여 배포 대상을 특정 개별 계정으로 설정할 수 있습니다.

- 없음(기본값) - 지정된 OU에 있는 모든 계정에 스택을 배포합니다.
 - 교집합 - 선택한 OU 내 특정 개별 계정에 스택을 배포합니다.
 - 차집합 - 특정 계정을 제외하고 선택한 OU의 모든 계정에 스택을 배포합니다.
 - 합집합 - 지정된 OU 및 추가 개별 계정에 스택을 배포합니다.
14. 자동 배포에서 나중에 대상 조직 또는 OU에 추가되는 계정에 자동으로 배포할지 여부를 선택합니다. 자세한 내용은 [AWS Organizations에서 StackSets에 대한 자동 배포 활성화 또는 비활성화](#) 섹션을 참조하세요.
 15. 자동 배포를 활성화한 경우 계정 제거 동작에서 대상 조직 또는 OU에서 계정이 제거될 때 스택 리소스를 유지할지 또는 삭제할지 여부를 선택합니다.

Note

스택 유지를 선택하면 스택이 StackSet에서 제거되지만 스택 및 관련 리소스는 유지됩니다. 리소스는 현재 상태로 유지되지만 더 이상 StackSet에 포함되지 않습니다.

16. 리전 지정에서 스택을 배포하려는 리전을 선택하세요.
17. 배포 옵션에서 다음을 수행하세요.
 - 최대 동시 계정에서 동시에 처리되는 계정 수를 지정하세요.
 - 내결함성에서 리전당 허용되는 최대 계정 실패 횟수를 지정합니다. 이 한도에 도달하면 작업이 중지되고 다른 리전으로 진행되지 않습니다.

- 리전 동시성에서 리전을 처리하는 방법(순차: 한 번에 하나의 리전 또는 병렬: 동시에 여러 리전)을 선택하세요.
- 동시성 모드에서 작업 실행 중에 동시성이 작동하는 방식을 선택하세요.
 - 엄격한 내결함성 - 내결함성 +1의 범위를 유지하면서 장애가 발생하는 경우 계정 동시성 수준을 줄입니다.
 - 소프트한 내결함성 - 장애에 관계없이 지정된 동시성 수준(최대 동시 계정의 값)을 유지합니다.
- StackSet 종속성의 경우 종속성을 최대 10개 이내로 유지하면서 종속 StackSet ARN을 추가합니다. 자세한 내용은 [AWS Organizations에서 StackSets에 대한 자동 배포 활성화 또는 비활성화](#) 섹션을 참조하세요.

18. 다음을 선택하여 계속 진행합니다.

19. 검토 페이지에서 StackSet가 올바른 리전의 올바른 계정에 배포하는지 확인한 다음 StackSet 생성을 선택합니다.

StackSet 세부 정보 페이지가 열립니다. StackSet의 스택 생성 진행률과 상태를 볼 수 있습니다.

서비스 관리형 권한으로 StackSet 생성(AWS CLI)

이 섹션의 다음 단계를 따라 AWS CLI를 사용합니다.

- StackSet 컨테이너를 생성합니다.
- 스택 인스턴스를 배포합니다.

Note

위임된 관리자 역할로 작업하는 경우 명령에 `--call-as DELEGATED_ADMIN`을 포함해야 합니다.

Deploy to your organization

StackSet를 생성하려면

1. [create-stack-set](#) 명령을 사용하여 *my-stackset*라는 이름의 새 StackSet를 생성합니다. 다음 예에서는 S3 버킷에 저장된 템플릿을 사용하고, 자동 배포를 활성화하고, 계정이 제거될 때 스

택을 보존합니다. 자세한 내용은 [AWS Organizations에서 StackSets에 대한 자동 배포 활성화 또는 비활성화](#) 섹션을 참조하세요.

```
aws cloudformation create-stack-set \
  --stack-set-name my-stackset \
  --template-url https://s3.region-code.amazonaws.com/amzn-s3-demo-bucket/MyApp.template \
  --permission-model SERVICE_MANAGED \
  --auto-deployment
Enabled=true,RetainStacksOnAccountRemoval=true,DependsOn=ARN1,ARN2
```

2. [list-stack-sets](#) 명령을 사용하여 StackSet가 생성되었는지 확인합니다. 새 StackSet가 결과에 나열됩니다.

```
aws cloudformation list-stack-sets
```

- 멤버 계정에 로그인한 상태에서 --call-as 옵션을 DELEGATED_ADMIN으로 설정하는 경우 list-stack-sets는 조직의 관리 계정에서 서비스 관리형 권한이 있는 모든 StackSets를 반환합니다.
- AWS 계정에 로그인한 상태에서 --call-as 옵션을 SELF로 설정하는 경우 list-stack-sets는 AWS 계정에 있는 모든 자체 관리형 스택 세트를 반환합니다.
- 조직의 관리 계정에 로그인한 상태에서 --call-as 옵션을 SELF로 설정하는 경우 list-stack-sets는 조직의 관리 계정에 있는 모든 StackSets를 반환합니다.

3. [create-stack-instances](#) 명령을 사용하여 StackSet에 스택을 추가합니다. --deployment-targets 옵션에서 조직의 모든 계정에 배포할 조직 루트 ID를 지정합니다.

--operation-preferences 옵션을 사용하여 동시 계정 처리 및 기타 배포 기본 설정을 지정하세요. 이 예제에서는 개수 기반 설정을 사용합니다. MaxConcurrentCount는 FailureToleranceCount + 1을 초과해서는 안 됩니다. 백분율 기반 설정의 경우 대신 FailureTolerancePercentage 또는 MaxConcurrentPercentage를 사용하세요.

```
aws cloudformation create-stack-instances --stack-set-name my-stackset \
  --deployment-targets OrganizationalUnitIds=r-a1b2c3d4e5 \
  --regions us-west-2 us-east-1 \
  --operation-preferences MaxConcurrentCount=1,FailureToleranceCount=0
```

자세한 내용은 AWS CloudFormation API 참조의 [CreateStackInstances](#)를 참조하세요.

4. `create-stack-instances` 출력의 일부로 반환된 `operation-id`를 사용하여 다음 [describe-stack-set-operation](#)을 통해 스택이 생성되었는지 확인하세요.

```
aws cloudformation describe-stack-set-operation \  
  --stack-set-name my-stackset \  
  --operation-id operation_ID
```

Deploy to organizational units (OUs)

StackSet를 생성하려면

1. [create-stack-set](#) 명령을 사용하여 *my-stackset*라는 이름의 새 StackSet를 생성합니다. 다음 예에서는 S3 버킷에 저장된 템플릿을 사용하고, *KeyPairName* 값을 *TestKey*으로 설정하는 파라미터를 포함합니다.

```
aws cloudformation create-stack-set \  
  --stack-set-name my-stackset \  
  --template-url https://s3.region-code.amazonaws.com/amzn-s3-demo-bucket/MyApp.template \  
  --permission-model SERVICE_MANAGED \  
  --parameters ParameterKey=KeyPairName,ParameterValue=TestKey
```

2. [list-stack-sets](#) 명령을 사용하여 StackSet가 생성되었는지 확인합니다. 새 StackSet가 결과에 나열됩니다.

```
aws cloudformation list-stack-sets
```

- 멤버 계정에 로그인한 상태에서 `--call-as` 옵션을 `DELEGATED_ADMIN`으로 설정하는 경우 `list-stack-sets`는 조직의 관리 계정에서 서비스 관리형 권한이 있는 모든 StackSets를 반환합니다.
 - AWS 계정에 로그인한 상태에서 `--call-as` 옵션을 `SELF`로 설정하는 경우 `list-stack-sets`는 AWS 계정에 있는 모든 자체 관리형 스택 세트를 반환합니다.
 - 조직의 관리 계정에 로그인한 상태에서 `--call-as` 옵션을 `SELF`로 설정하는 경우 `list-stack-sets`는 조직의 관리 계정에 있는 모든 StackSets를 반환합니다.
3. [create-stack-instances](#) 명령을 사용하여 StackSet에 스택을 추가합니다. `--deployment-targets` 옵션에서 스택을 배포할 OU ID 지정합니다.

--operation-preferences 옵션을 사용하여 동시 계정 처리 및 기타 배포 기본 설정을 지정하세요. 이 예제에서는 개수 기반 설정을 사용합니다. MaxConcurrentCount는 FailureToleranceCount + 1을 초과해서는 안 됩니다. 백분율 기반 설정의 경우 대신 FailureTolerancePercentage 또는 MaxConcurrentPercentage를 사용하세요.

```
aws cloudformation create-stack-instances --stack-set-name my-stackset \
  --deployment-targets OrganizationalUnitIds=ou-rcuk-1x5j1lwo,ou-rcuk-slr5lh0a \
  --regions us-west-2 us-east-1 \
  --operation-preferences MaxConcurrentCount=1,FailureToleranceCount=0
```

자세한 내용은 AWS CloudFormation API 참조의 [CreateStackInstances](#)를 참조하세요.

4. create-stack-instances 출력의 일부로 반환된 operation-id를 사용하여 다음 [describe-stack-set-operation](#)을 통해 스택이 생성되었는지 확인하세요.

```
aws cloudformation describe-stack-set-operation \
  --stack-set-name my-stackset \
  --operation-id operation_ID
```

Deploy to specific accounts in OUs

특정 조직 단위(OU)를 대상으로 하고 계정 필터링을 사용하여, 스택을 배포할 계정을 정확하게 제어할 수 있습니다. 기본적으로, 계정 필터링이 지정되지 않은 경우 스택은 지정된 OU 내의 모든 계정에 배포됩니다.

AWS CLI에서 --deployment-targets 옵션을 사용하여 계정 필터링을 지정합니다. 자세한 내용은 [DeploymentTargets](#)를 참조하세요.

create-stack-set 명령을 사용하여 StackSet 컨테이너를 생성한 후, 다음 예 중 하나를 사용하여 특정 계정에 스택을 배포합니다.

OU의 특정 계정을 대상으로 지정

다음 예에서는 OU1의 계정 A1 및 A2에만 스택을 배포합니다.

```
aws cloudformation create-stack-instances --stack-set-name my-stackset \
  --deployment-targets
  OrganizationalUnitIds=OU1,Accounts=A1,A2,AccountFilterType=INTERSECTION \
  --regions us-west-2 us-east-1
```

OU에서 계정 제외

다음 예에서는 계정 A1과 A2를 제외한 OU1의 모든 계정에 스택을 배포합니다.

```
aws cloudformation create-stack-instances --stack-set-name my-stackset \  
--deployment-targets  
OrganizationalUnitIds=OU1,Accounts=A1,A2,AccountFilterType=DIFFERENCE \  
--regions us-west-2 us-east-1
```

AWS Organizations에서 StackSets에 대한 자동 배포 활성화 또는 비활성화

CloudFormation은 대상 조직 또는 조직 단위(OU)에 추가될 때 새 AWS Organizations 계정에 추가 스택을 자동으로 배포할 수 있습니다. 자동 배포를 활성화하고 대상 OU에서 계정이 제거될 때 스택 및 연결된 리소스를 삭제할 것인지 또는 유지할 것인지 선택할 수 있습니다. 이러한 설정은 서비스 관리형 권한을 사용하는 StackSets에 대해 언제든지 수정할 수 있습니다.

자동 배포 작동 방식

자동 배포가 활성화되면 계정이 대상 조직 또는 OU에 추가되거나 대상 조직 또는 OU에서 제거되거나 대상 OU 간에 이동될 때 이 기능이 트리거됩니다.

예를 들어 us-east-1 리전의 OU1을 대상으로 하는 StackSet1 및 us-east-1 리전의 OU2를 대상으로 하는 StackSet2를 고려합니다. OU1에는 AccountA가 포함되어 있습니다.

자동 배포가 활성화된 상태에서 AccountA를 OU1에서 OU2로 이동하면 CloudFormation이 자동으로 삭제 작업을 실행하여 AccountA에서 StackSet1 스택을 제거하고, AccountA에 StackSet2 스택을 추가하는 생성 작업을 대기열에 추가합니다.

고려 사항

다음은 자동 배포를 사용할 때 고려 사항입니다.

- 자동 배포 기능은 StackSet 수준에서 활성화됩니다. OU, 계정 또는 리전의 경우 자동 배포를 선택적으로 조정할 수 없습니다.
- 재정의된 파라미터값은 현재 대상 OU와 해당 하위 OU에 있는 계정에만 적용됩니다. 나중에 대상 OU와 해당 하위 OU에 추가되는 계정은 재정의된 값이 아닌 StackSet 기본값을 사용합니다.

- 자동 배포 시 계정 수준 대상 지정 필터를 고려하지 않습니다. 특정 계정을 대상으로 하고 자동 배포를 활성화하는 경우 StackSet는 배포된 조직 내 새로 추가된 계정에도 계속 배포합니다. 새로 추가된 계정에 대한 배포를 방지하려면 자동 배포를 비활성화합니다.
- 종속성 관리: StackSet당 최대 10개의 종속성을 정의하고 이때 계정당 최대 100개의 종속성을 포함합니다. 예를 들어 종속성이 각각 5개인 StackSet가 5개 있는 경우 25개의 종속성이 있으며 종속성 한도를 100개로 계산됩니다. [Service Quotas 콘솔](#)을 통해 한도 증가를 요청할 수 있습니다. 종속성은 StackSet가 삭제되거나 조직이 비활성화될 때 제거됩니다.
- 자동 배포를 사용할 경우 관리형 실행을 활성화하는 것이 좋습니다. 관리형 실행을 사용하면 여러 대상 계정의 자동 배포 작업을 StackSet 내에서 동시에 실행할 수 있으므로 특히 규모가 더 큰 조직의 경우 처리 속도가 향상됩니다.

자동 배포 활성화 또는 비활성화(콘솔)

자동 배포를 활성화 또는 비활성화하려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 StackSet를 생성한 AWS 리전을 선택합니다.
3. 탐색 창에서 스택 세트를 선택합니다.
4. StackSet 페이지에서 업데이트할 StackSet 이름 옆의 옵션을 선택합니다.
5. 오른쪽 상단의 작업 메뉴에서 자동 배포 편집을 선택하세요.
6. 열리는 대화 상자에서 다음을 수행하세요.
 - a. 자동 배포에서 활성화됨 또는 비활성화됨을 선택하세요.
 - b. 계정 제거 동작에서 스택 삭제 또는 스택 유지를 선택하세요. 유지되는 리소스는 현재 상태로 남아 있지만 더 이상 StackSet에 포함되지 않습니다.
 - c. StackSet 종속성의 경우 종속성을 최대 10개 이내로 유지하면서 종속 StackSet ARN을 추가합니다.
7. 저장을 선택합니다.

자동 배포 활성화 또는 비활성화(AWS CLI)

자동 배포를 활성화 또는 비활성화하려면

1. `--auto-deployment` 옵션과 함께 [update-stack-set](#) 명령을 사용합니다.

다음 명령은 자동 배포를 활성화합니다.

```
aws cloudformation update-stack-set --stack-set-name my-stackset \  
  --use-previous-template --auto-deployment  
  Enabled=true,RetainStacksOnAccountRemoval=true,DependsOn=ARN1,ARN2
```

또는 자동 배포를 비활성화하려면 다음 예제와 같이 --auto-deployment 옵션의 값으로 Enabled=false를 지정하세요.

```
aws cloudformation update-stack-set --stack-set-name my-stackset \  
  --use-previous-template --auto-deployment Enabled=false
```

2. update-stack-set 출력의 일부로 반환된 작업 ID를 사용해 [describe-stack-set-operation](#)을 실행하여 StackSet가 업데이트되었는지 확인합니다.

```
aws cloudformation describe-stack-set-operation --operation-id operation_ID
```

CloudFormation StackSet 업데이트

CloudFormation 콘솔이나 AWS CLI를 사용하여 StackSet를 업데이트할 수 있습니다.

StackSet에서 계정과 리전을 추가하고 제거하려면 [StackSets에 스택 추가](#) 및 [StackSets에서 스택 삭제](#) 섹션을 참조하세요. 스택에 대한 파라미터 값을 재정의하려면 [스택에서 파라미터 재정의](#) 섹션을 참조하세요.

주제

- [StackSet 업데이트\(콘솔\)](#)
- [StackSet 업데이트\(AWS CLI\)](#)

StackSet 업데이트(콘솔)

StackSet를 업데이트하려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 StackSet를 생성한 AWS 리전을 선택합니다.

3. 탐색 창에서 스택 세트를 선택합니다.
4. StackSet 페이지에서 업데이트하려는 StackSet를 선택합니다.
5. StackSet를 선택한 상태로 작업 메뉴에서 StackSet 세부 정보 편집을 선택합니다.
6. 템플릿 선택 페이지에서 필요에 따라 권한 섹션을 업데이트하거나 다음 단계로 건너뜁니다.
7. 사전 조건 - 템플릿 준비에서 현재 템플릿 사용을 선택하여 현재 템플릿을 사용하거나, 현재 템플릿 교체를 선택하여 다른 템플릿의 S3 URL을 지정하거나 새 템플릿을 업로드합니다.
8. 다음을 선택합니다.
9. StackSet 세부 정보 지정 페이지의 StackSet 설명에서 필요에 따라 StackSet에 대한 설명을 업데이트합니다.
10. 파라미터에서 필요에 따라 파라미터 값을 업데이트합니다.
11. 다음을 선택합니다.
12. StackSet 옵션 구성 페이지의 태그에서 필요에 따라 태그를 수정합니다. 태그를 추가, 업데이트 또는 삭제합니다. AWS에서 태그를 사용하는 방법에 대한 자세한 내용은 AWS 결제 및 비용 관리 사용 설명서의 [Organizing and tracking costs using AWS cost allocation tags](#)를 참조하세요.
13. 실행 구성에서 필요에 따라 실행 구성을 업데이트할 수 있습니다.

 Note

작업이 실행 중이거나 대기열에 있는 경우 실행 설정을 변경할 수 없습니다.

14. 템플릿에 IAM 리소스가 포함되어 있는 경우 기능에서 이 템플릿이 IAM 리소스를 생성할 수 있음을 확인합니다를 선택하여 템플릿에서 IAM 리소스를 사용하도록 지정합니다. 자세한 내용은 [CloudFormation 템플릿에서 IAM 리소스 승인](#) 섹션을 참조하세요.
15. 다음을 선택합니다.
16. 배포 옵션 설정 페이지에서 업데이트를 위한 계정과 리전을 제공합니다.

리전의 배포 실패가 지정된 내결함성을 초과하지 않는 한 CloudFormation은 첫 번째 리전의 지정된 계정에 스택 업데이트를 배포한 후 다음으로 이동하는 식의 과정을 계속합니다.

- a. [자체 관리형 권한] 계정, 배포 위치에서 계정에 스택 배포를 선택합니다. StackSet를 생성하는 데 사용한 대상 계정 ID를 텍스트 상자에 붙여 넣습니다. 숫자가 여러 개인 경우 각 숫자를 쉼표로 구분합니다.

[서비스 관리형 권한] 다음 중 하나를 수행합니다.

- 조직 단위(OU)에 배포를 선택합니다. StackSet을 생성하는 데 사용한 대상 OU를 입력합니다.
 - 계정에 배포를 선택합니다. StackSet을 생성하는 데 사용한 대상 OU ID 또는 계정 ID를 붙여 넣습니다.
- b. 리전 지정에서 CloudFormation에서 업데이트를 배포할 순서를 지정합니다.
- c. 배포 옵션에서 다음을 수행하세요.
- 최대 동시 계정에서 동시에 처리되는 계정 수를 지정하세요.
 - 내결함성에서 리전당 허용되는 최대 계정 실패 횟수를 지정합니다. 이 한도에 도달하면 작업이 중지되고 다른 리전으로 진행되지 않습니다.
 - 리전 동시성에서 리전을 처리하는 방법(순차: 한 번에 하나의 리전 또는 병렬: 동시에 여러 리전)을 선택하세요.
 - 동시성 모드에서 작업 실행 중에 동시성이 작동하는 방식을 선택하세요.
 - 엄격한 내결함성 - 내결함성 +1의 범위를 유지하면서 장애가 발생하는 경우 계정 동시성 수준을 줄입니다.
 - 소프트한 내결함성 - 장애에 관계없이 지정된 동시성 수준(최대 동시 계정의 값)을 유지합니다.
 - [서비스 관리형 권한] StackSet 종속성의 경우 종속성을 최대 10개 이내로 유지하면서 종속 StackSet ARN을 추가합니다. 자세한 내용은 [AWS Organizations에서 StackSets에 대한 자동 배포 활성화 또는 비활성화](#) 섹션을 참조하세요.
- d. 다음을 선택하여 계속 진행합니다.
17. [검토] 페이지에서 선택 사항을 검토합니다. 변경하려면 관련 섹션에서 편집을 선택합니다.
18. 계속 진행할 준비가 되면 제출을 선택합니다.

CloudFormation은 StackSet에 업데이트 적용을 시작하고 StackSet 세부 정보 페이지의 작업 탭을 표시합니다. 작업 탭에서 업데이트 작업의 진행률 및 상태를 볼 수 있습니다.

StackSet 업데이트(AWS CLI)

Note

위임된 관리자 역할로 작업하는 경우 명령에 `--call-as DELEGATED_ADMIN`을 포함해야 합니다.

1. StackSet를 업데이트하려면

[update-stack-set](#) 명령을 사용하여 StackSet를 변경합니다.

다음 예에서는 `--parameters` 옵션을 사용하여 StackSet를 업데이트합니다. 구체적으로, 전송 채널 구성의 기본 스냅샷 전송 빈도를 `TwentyFour_Hours`에서 `Twelve_Hours`로 변경합니다. 아직 현재 템플릿을 사용하고 있기 때문에 `--use-previous-template` 옵션을 추가합니다.

`--operation-preferences` 옵션을 사용하여 동시 계정 처리 및 기타 배포 기본 설정을 지정하세요. 이 예제에서는 개수 기반 설정을 사용합니다. `MaxConcurrentCount`는 `FailureToleranceCount + 1`을 초과해서는 안 됩니다. 백분율 기반 설정의 경우 대신 `FailureTolerancePercentage` 또는 `MaxConcurrentPercentage`를 사용하세요.

[자체 관리형 권한] `--accounts` 옵션에 업데이트 대상으로 지정할 계정 ID를 입력합니다.

```
aws cloudformation update-stack-set --stack-set-name my-stackset \
  --use-previous-template \
  --parameters ParameterKey=MaximumExecutionFrequency,ParameterValue=Twelve_Hours \
  --accounts account_ID_1 account_ID_2 \
  --regions us-west-2 us-east-1 \
  --operation-preferences MaxConcurrentCount=1,FailureToleranceCount=0
```

[서비스 관리형 권한] `--deployment-targets` 옵션에 업데이트 대상으로 지정할 조직 루트 ID 또는 조직 단위(OU) ID를 제공하세요.

```
aws cloudformation update-stack-set --stack-set-name my-stackset \
  --use-previous-template \
  --parameters ParameterKey=MaximumExecutionFrequency,ParameterValue=Twelve_Hours \
  --deployment-targets OrganizationalUnitIds=ou-rcuk-1x5j1lwo,ou-rcuk-slr5lh0a \
  --regions us-west-2 us-east-1 \
  --operation-preferences MaxConcurrentCount=1,FailureToleranceCount=0
```

자세한 내용을 알아보려면 AWS CloudFormation API 참조의 [UpdateStackSet](#)를 참조하세요.

- 업데이트 작업 상태 및 결과를 표시하려면 `describe-stack-set-operation` 명령을 실행하여 StackSet가 성공적으로 업데이트되었는지 확인합니다. `--operation-id`의 경우 `update-stack-set` 명령에서 반환한 작업 ID를 사용합니다.

```
aws cloudformation describe-stack-set-operation \
  --operation-id operation_ID
```

CloudFormation StackSets에 스택 추가

StackSet를 생성할 때 해당 StackSet에 대한 스택을 생성할 수 있습니다. 또한 CloudFormation을 사용하면 StackSet가 생성된 후 언제든지 추가 계정과 리전에 대해 스택을 추가할 수 있습니다. CloudFormation 콘솔이나 AWS CLI를 사용하여 스택을 추가할 수 있습니다.

주제

- [StackSet에 스택 추가\(콘솔\)](#)
- [StackSet에 스택 추가\(AWS CLI\)](#)

StackSet에 스택 추가(콘솔)

StackSet에 스택을 추가하려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 StackSet를 생성한 AWS 리전을 선택합니다.
3. 탐색 창에서 스택 세트를 선택합니다. StackSet 페이지에서 생성한 StackSet를 선택합니다.
4. StackSet를 선택한 상태로 작업 메뉴에서 StackSet에 스택 추가를 선택합니다.
5. 배포 옵션 설정 페이지에서 다음을 수행합니다.
 - a. StackSet에 스택 추가에서 새 스택 배포를 선택합니다.
 - b. 다음으로 StackSet의 권한 구성에 따라 다음을 수행합니다.
 - [자체 관리형 권한] 계정, 배포 위치에서 계정에 스택 배포를 선택합니다. 숫자 여러 개를 쉼표로 구분하여 대상 계정 번호를 텍스트 상자에 붙여 넣습니다.
 - [서비스 관리형 권한] 배포 대상에 대해 다음 중 하나를 수행합니다.
 - 조직에 배포를 선택하여 조직의 모든 계정에 배포합니다.
 - 조직 단위(OU)에 배포를 선택하여 특정 OU의 모든 계정에 배포합니다. 다른 OU 추가를 선택한 다음, 텍스트 상자에 대상 OU ID를 붙여 넣습니다. 각각의 새로운 대상 OU에 대해 반복합니다. CloudFormation은 선택한 대상의 하위 OU도 대상으로 합니다.

Note

StackSet가 이미 대상으로 하는 OU를 추가하는 경우 CloudFormation은 StackSet의 스택이 아직 없는 OU의 모든 계정(예: StackSet가 생성되고 자동 배포가 비활성화된 후 OU에 추가된 계정)에 새 스택을 생성합니다.

- c. 리전 지정에서 이전 단계에서 지정한 대상 계정에 배포할 AWS 리전을 지정합니다. 리전의 배포 실패가 지정된 내결함성을 초과하지 않는 한 기본적으로 CloudFormation은 첫 번째 리전의 지정된 계정에 스택을 배포한 후 다음으로 이동하는 식의 과정을 계속합니다.
 - d. 배포 옵션에서 다음을 수행하세요.
 - 최대 동시 계정에서 동시에 처리되는 계정 수를 지정하세요.
 - 내결함성에서 리전당 허용되는 최대 계정 실패 횟수를 지정합니다. 이 한도에 도달하면 작업이 중지되고 다른 리전으로 진행되지 않습니다.
 - 리전 동시성에서 리전을 처리하는 방법(순차: 한 번에 하나의 리전 또는 병렬: 동시에 여러 리전)을 선택하세요.
 - 동시성 모드에서 작업 실행 중에 동시성이 작동하는 방식을 선택하세요.
 - 엄격한 내결함성 - 내결함성 +1의 범위를 유지하면서 장애가 발생하는 경우 계정 동시성 수준을 줄입니다.
 - 소프트한 내결함성 - 장애에 관계없이 지정된 동시성 수준(최대 동시 계정의 값)을 유지합니다.
 - e. 다음을 선택합니다.
6. Specify Overrides(재정의 설정) 페이지의 속성값을 지정한 값으로 남겨둡니다. 생성할 스택의 속성값 모두를 무시할 수 없습니다. 다음을 선택합니다.
 7. [검토] 페이지에서 선택 사항을 검토합니다. 변경하려면 관련 섹션에서 편집을 선택합니다.
 8. 계속 진행할 준비가 되면 제출을 선택합니다.

CloudFormation이 스택 생성을 시작합니다. 제출을 선택하면 열리는 StackSet 세부 정보 페이지에서 StackSet의 스택 생성 진행률 및 상태를 봅니다. 완료되면 새 스택이 스택 인스턴스 탭에 나열됩니다.

StackSet에 스택 추가(AWS CLI)

Note

위임된 관리자 역할로 작업하는 경우 명령에 `--call-as DELEGATED_ADMIN`을 포함해야 합니다.

자체 관리형 권한을 사용하여 스택을 StackSet에 추가하려면

`create-stack-instances` CLI 명령을 사용합니다. `--accounts` 옵션에서 스택을 생성하려는 계정 ID를 제공합니다.

```
aws cloudformation create-stack-instances --stack-set-name my-stackset \
  --accounts account_id --regions eu-west-1 us-west-2
```

서비스 관리형 권한을 사용하여 스택을 StackSet에 추가하려면

`create-stack-instances` CLI 명령을 사용합니다. `--deployment-targets` 옵션에서 스택을 생성하려는 조직(루트) ID 또는 OU ID를 제공합니다. 특정 계정을 대상으로 하는 명령의 예는 [서비스 관리형 권한으로 StackSet 생성\(AWS CLI\)](#) 섹션을 참조하세요.

```
aws cloudformation create-stack-instances --stack-set-name my-stackset \
  --deployment-targets OrganizationalUnitIds=ou-rcuk-r1qi0wl7 --regions eu-west-1 us-west-2
```

Note

StackSet가 이미 대상으로 하는 OU를 추가하는 경우 CloudFormation은 StackSet의 스택이 아직 없는 OU의 모든 계정(예: StackSet가 생성되고 자동 배포가 비활성화된 후 OU에 추가된 계정)에 새 스택을 생성합니다.

CloudFormation StackSet 내 스택에서 파라미터 값 재정의

경우에 따라 StackSet 자체에 지정한 속성 값과 다른 값을 갖는 스택을 특정 리전이나 계정에 적용하고자 할 수 있습니다. 예를 들어, 계정을 개발이나 생산에 사용할지 여부에 따라 특정 파라미터의 값을 다르게 지정하고 싶을 수 있습니다. 이 경우 CloudFormation을 통해 계정과 리전별로 스택의 파라미터 값

을 재정의할 수 있습니다. 스택을 처음 생성할 때 템플릿 파라미터 값을 재정의할 수 있고, 기존 스택의 파라미터 값을 재정의할 수 있습니다. 이전에 스택에서 재정의한 파라미터만 StackSet에 지정된 값으로 다시 설정할 수 있습니다.

선택한 계정과 리전의 스택에 파라미터 값 재정의의 적용합니다. StackSet를 업데이트하는 동안, 스택에 대해 재정의된 파라미터 값은 업데이트되지 않고, 재정의된 값이 유지됩니다.

StackSet에 지정한 파라미터 값만 재정의할 수 있습니다. 파라미터 자체를 추가 또는 삭제하려면 StackSet 템플릿을 업데이트해야 합니다. StackSet 템플릿에 파라미터를 추가하는 경우 StackSet에 지정한 새로운 파라미터와 값으로 모든 스택을 먼저 업데이트해야 스택의 해당 파라미터 값을 재정의할 수 있습니다. 새 파라미터로 모든 스택을 업데이트한 후에는 원하는 대로 개별 스택에서 파라미터 값을 재정의할 수 있습니다.

스택을 생성하는 경우 StackSet 파라미터 값을 재정의하는 방법은 [StackSets에 스택 추가](#) 섹션을 참조하세요.

주제

- [스택에서 파라미터 재정의\(콘솔\)](#)
- [스택에서 파라미터 재정의\(AWS CLI\)](#)

스택에서 파라미터 재정의(콘솔)

특정 스택의 파라미터를 재정의하려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 StackSet를 생성한 AWS 리전을 선택합니다.
3. 탐색 창에서 스택 세트를 선택합니다. StackSet 페이지에서 StackSet를 선택합니다.
4. StackSet를 선택한 상태로 작업 메뉴에서 StackSet 파라미터 재정의의 선택합니다.
5. 배포 옵션 설정 페이지에서 재정의의 생성할 스택의 계정과 리전을 제공하세요.

리전의 배포 실패가 지정된 내결합성을 초과하지 않는 경우에 한해 기본적으로 CloudFormation은 첫 번째 리전의 지정된 계정에 스택을 배포한 후 다음으로 이동하는 식의 과정을 계속합니다.

- a. [자체 관리형 권한] 배포 위치에서 계정에 스택 배포를 선택합니다. StackSet를 생성하는 데 사용한 대상 계정 ID를 모두 또는 일부 붙여 넣습니다.

[서비스 관리형 권한] 다음 중 하나를 수행합니다.

- 조직 단위(OU)에 배포를 선택합니다. StackSet을 생성하는 데 사용한 대상 OU를 하나 이상 입력합니다. 재정의된 파라미터값은 현재 대상 OU와 해당 하위 OU에 있는 계정에만 적용됩니다. 나중에 대상 OU와 해당 하위 OU에 추가되는 계정은 재정의된 값이 아닌 StackSet 기본값을 사용합니다.
 - 계정에 배포를 선택합니다. StackSet을 생성하는 데 사용한 대상 OU ID 또는 계정 ID를 모두 또는 일부 붙여 넣습니다.
- b. 리전 지정에서 이 StackSet에 대해 스택을 배포한 리전을 하나 이상 추가합니다.
- 여러 리전을 추가하면 Specify regions(리전 지정) 아래의 리전 순서에 따라 배포 순서가 결정됩니다.
- c. 배포 옵션에서 다음을 수행하세요.
- 최대 동시 계정에서 동시에 처리되는 계정 수를 지정하세요.
 - 내결함성에서 리전당 허용되는 최대 계정 실패 횟수를 지정합니다. 이 한도에 도달하면 작업이 중지되고 다른 리전으로 진행되지 않습니다.
 - 리전 동시성에서 리전을 처리하는 방법(순차: 한 번에 하나의 리전 또는 병렬: 동시에 여러 리전)을 선택하세요.
 - 동시성 모드에서 작업 실행 중에 동시성이 작동하는 방식을 선택하세요.
 - 엄격한 내결함성 - 내결함성 +1의 범위를 유지하면서 장애가 발생하는 경우 계정 동시성 수준을 줄입니다.
 - 소프트한 내결함성 - 장애에 관계없이 지정된 동시성 수준(최대 동시 계정의 값)을 유지합니다.
- d. 다음을 선택합니다.
6. 재정의 지정 페이지의 재정의 값 편집 메뉴에서 StackSet 값 재정의의를 선택합니다.
7. StackSet 파라미터 값 재정 페이지에서 변경하고 변경 사항 저장을 선택합니다.

Note

재정의한 파라미터에 StackSet에서 지정한 값을 사용하려면 모든 파라미터를 선택하고 재정의 값 편집 메뉴에서 StackSet 값으로 설정을 선택합니다. 이렇게 하면 스택을 업데이트한 후 모든 재정의 값이 제거됩니다.

8. [검토] 페이지에서 선택 사항을 검토합니다. 변경하려면 관련 섹션에서 편집을 선택합니다.
9. 계속 진행할 준비가 되면 제출을 선택합니다.

CloudFormation이 스택 업데이트를 시작합니다. 제출을 선택하면 열리는 StackSet 세부 정보 페이지에서 스택의 진행률 및 상태를 확인합니다.

스택에서 파라미터 재정의(AWS CLI)

Note

위임된 관리자 역할로 작업하는 경우 명령에 `--call-as DELEGATED_ADMIN`을 포함해야 합니다.

특정 스택의 파라미터를 재정의하려면

1. [update-stack-instances](#) AWS CLI 명령을 사용하고 `--parameter-overrides` 옵션을 지정합니다.

[자체 관리형 권한] `--accounts` 옵션에 스택에서 파라미터 값을 재정의하려는 계정 ID를 제공하세요.

```
aws cloudformation update-stack-instances --stack-set-name my-stackset \  
  --parameter-overrides ParameterKey=Subnets,ParameterValue=subnet-1baa3351 \  
  \, subnet-27b86940 \  
  --accounts account_id --regions us-east-1
```

[서비스 관리형 권한] `--deployment-targets` 옵션에 스택에서 파라미터를 재정의하려는 조직 루트 ID, OU ID 또는 AWS Organizations 계정 ID를 제공하세요. 이 예에서는 OU에 있는 모든 계정의 스택에 대한 파라미터 값을 *ou-rcuk-1x5j1lwo* ID로 재정의합니다.

재정의된 파라미터값은 현재 대상 OU와 해당 하위 OU에 있는 계정에만 적용됩니다. 나중에 대상 OU와 해당 하위 OU에 추가되는 계정은 재정의된 값이 아닌 StackSet 기본값을 사용합니다.

```
aws cloudformation update-stack-instances --stack-set-name my-stackset \  
  --parameter-overrides ParameterKey=Subnets,ParameterValue=subnet-1baa3351 \  
  \, subnet-27b86940 \  
  --deployment-targets OrganizationalUnitIds=ou-rcuk-1x5j1lwo \  
  --regions us-east-1
```

- 업데이트 작업의 상태와 결과를 표시하는 `describe-stack-set-operation` 명령을 실행하여 스택에서 파라미터 값이 재정의되었는지 확인하세요. `--operation-id`의 경우 `update-stack-instances` 명령에서 반환한 작업 ID를 사용합니다.

```
aws cloudformation describe-stack-set-operation --operation-id operation_ID
```

CloudFormation StackSet에서 스택 삭제

CloudFormation 콘솔이나 AWS CLI를 사용하여 StackSets에서 스택을 삭제할 수 있습니다.

주제

- [StackSet에서 스택 삭제\(콘솔\)](#)
- [StackSet에서 스택 삭제\(AWS CLI\)](#)

Note

최상위 조직 단위(OU)에서 스택을 삭제하면 해당 OU가 StackSet 대상에서 해제됩니다.

StackSet에서 스택 삭제(콘솔)

스택을 삭제하는 방법

- AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
- 화면 상단의 탐색 모음에서 StackSet를 생성한 AWS 리전을 선택합니다.
- 탐색 창에서 스택 세트를 선택합니다. StackSet 페이지에서 StackSet를 선택합니다.
- StackSet를 선택한 상태로 작업 메뉴에서 StackSet에서 스택 삭제를 선택합니다.
- 배포 옵션 설정 페이지에서 먼저 스택을 삭제할 계정 및 리전을 선택하세요.
 - [자체 관리형 권한] 계정에서 계정에 스택 배포 또는 조직 단위에 스택 배포를 선택하세요.

계정에 스택 배포를 선택하는 경우 계정 번호 텍스트 상자에 대상 계정 번호를 붙여넣으세요 (여러 숫자는 쉼표로 구분).

조직 단위에 스택 배포를 선택하는 경우 조직 번호 텍스트 상자에 대상 OU ID를 붙여넣어 지정된 조직의 일부인 모든 계정을 대상으로 지정하세요.

- b. [서비스 관리형 권한] OU(조직 단위)에서 대상 OU ID를 지정합니다.

⚠ Important

CloudFormation이 지정된 대상 OU 및 하위 OU에서 스택을 삭제합니다.

계정 필터 유형에서는 다음 옵션 중 하나를 선택하고 계정 번호를 제공하여 스택을 삭제하려는 계정을 세분화할 수 있습니다.

- 없음(기본값) - 지정된 OU에 있는 모든 계정에서 스택을 삭제합니다.
- 교집합 - 선택한 OU 내 특정 개별 계정에서만 스택을 삭제합니다.
- 차집합 - 특정 계정을 제외하고 선택한 OU의 모든 계정에서 스택을 삭제합니다.
- 합집합 - 지정된 OU 및 추가 개별 계정에서 스택을 삭제합니다.

- c. 리전 지정에서는 대상 계정 내에서 스택을 삭제하려는 리전을 선택하세요.

6. 배포 옵션에서 다음을 수행하세요.

- 최대 동시 계정에서 동시에 처리되는 계정 수를 지정하세요.
- 내결함성에서 리전당 허용되는 최대 계정 실패 횟수를 지정합니다. 이 한도에 도달하면 작업이 중지되고 다른 리전으로 진행되지 않습니다.
- 스택 유지에서, 이 옵션을 활성화하여 StackSet에서 스택을 제거할 때 스택 및 관련 리소스를 저장할지 여부를 선택합니다. 리소스는 현재 상태로 유지되지만 더 이상 StackSet의 일부가 아닙니다.
- 리전 동시성에서 리전을 처리하는 방법(순차: 한 번에 하나의 리전 또는 병렬: 동시에 여러 리전)을 선택하세요.
- 동시성 모드에서 작업 실행 중에 동시성이 작동하는 방식을 선택하세요.
 - 엄격한 내결함성 - 내결함성 +1의 범위를 유지하면서 장애가 발생하는 경우 계정 동시성 수준을 줄입니다.
 - 소프트한 내결함성 - 장애에 관계없이 지정된 동시성 수준(최대 동시 계정의 값)을 유지합니다.

7. 다음을 선택합니다.

8. [검토] 페이지에서 선택 사항을 검토합니다. 변경하려면 관련 섹션에서 편집을 선택합니다.

9. StackSet에서 스택을 제거할 준비가 되면 제출을 선택합니다.

스택 삭제가 완료되면 스택 인스턴스 탭의 StackSet 세부 정보 페이지에서 스택이 StackSet에서 삭제되었는지 확인할 수 있습니다.

StackSet에서 스택 삭제(AWS CLI)

Note

위임된 관리자 역할로 작업하는 경우 명령에 `--call-as DELEGATED_ADMIN`을 포함해야 합니다.

StackSet 이름과 함께 `delete-stack-instances` 명령을 사용합니다.

이 예제에서는 스택을 유지하지 않기 때문에 `--no-retain-stacks` 옵션을 사용합니다. 스택과 해당 리소스를 유지하려면 `--no-retain-stacks` 대신 `--retain-stacks`를 사용하세요.

`--regions`에는 스택을 삭제하려는 AWS 리전(예: `us-west-2` 및 `us-east-1`)을 지정하세요.

`--operation-preferences` 옵션을 사용하여 동시 계정 처리 및 기타 기본 설정을 지정하세요. 이 예제에서는 개수 기반 설정을 사용합니다. `MaxConcurrentCount`는 `FailureToleranceCount` + 1을 초과해서는 안 됩니다. 백분율 기반 설정의 경우 대신 `FailureTolerancePercentage` 또는 `MaxConcurrentPercentage`를 사용하세요.

스택을 삭제하려면(자체 관리형 권한)

`--accounts` 옵션에서 스택을 삭제할 계정의 ID를 지정합니다.

```
aws cloudformation delete-stack-instances --stack-set-name my-stackset \  
  --accounts account_ID_1 account_ID_2 \  
  --regions us-west-2 us-east-1 \  
  --no-retain-stacks \  
  --operation-preferences MaxConcurrentCount=1,FailureToleranceCount=0
```

스택을 삭제하려면(서비스 관리형 권한)

`--deployment-targets`에서 스택을 삭제할 조직 루트 ID 또는 조직 단위(OU) ID를 지정합니다.

⚠ Important

CloudFormation이 지정된 대상 OU 및 하위 OU에서 스택을 삭제합니다.

```
aws cloudformation delete-stack-instances --stack-set-name my-stackset \  
--deployment-targets OrganizationalUnitIds=ou-rcuk-1x5jlwo,ou-rcuk-slrl5lh0a \  
--regions us-west-2 us-east-1 \  
--no-retain-stacks \  
--operation-preferences MaxConcurrentCount=1,FailureToleranceCount=0
```

자세한 내용은 AWS CloudFormation API 참조의 [DeleteStackInstances](#)를 참조하세요.

필요한 경우, 스택 삭제가 완료되면 `describe-stack-set-operation` 명령을 실행하여 스택 삭제 작업의 상태 및 결과를 표시해 스택이 StackSet에서 삭제되었는지 확인합니다. `--operation-id`의 경우 `delete-stack-instances` 명령에서 반환한 작업 ID를 사용합니다.

```
aws cloudformation describe-stack-set-operation --stack-set-name my-stackset \  
--operation-id ddf16f54-ad62-4d9b-b0ab-3ed8e9example
```

CloudFormation StackSet 삭제

StackSet를 삭제하려면 먼저 StackSet에 있는 스택을 모두 삭제해야 합니다. 모든 스택을 삭제하는 방법에 대한 자세한 내용은 [StackSets에서 스택 삭제](#) 섹션을 참조하세요.

주제

- [StackSet 삭제\(콘솔\)](#)
- [StackSet 삭제\(AWS CLI\)](#)
- [서비스 역할 삭제\(선택 사항\)](#)

StackSet 삭제(콘솔)

StackSet를 삭제하려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.

2. 화면 상단의 탐색 모음에서 StackSet를 생성한 AWS 리전을 선택합니다.
3. StackSet 페이지에서 StackSet를 선택합니다.
4. StackSet를 선택한 상태로 작업 메뉴에서 StackSet 삭제를 선택합니다.
5. StackSet를 삭제할지 확인하는 메시지가 표시되면 StackSet 삭제를 선택합니다.

StackSet 삭제(AWS CLI)

Note

위임된 관리자 역할로 작업하는 경우 명령에 `--call-as DELEGATED_ADMIN`을 포함해야 합니다.

StackSet를 삭제하려면

1. 다음 `delete-stack-set` 명령을 사용합니다. 확인하라는 메시지가 표시되면 **y**를 입력한 다음 Enter를 누릅니다.

```
aws cloudformation delete-stack-set --stack-set-name my-stackset
```

2. `list-stack-sets` 명령을 실행하여 StackSet가 삭제되었는지 확인합니다. `list-stack-sets` 명령 결과에 스택 상태가 DELETED로 표시되어야 합니다.

```
aws cloudformation list-stack-sets
```

서비스 역할 삭제(선택 사항)

CloudFormation에서 StackSet 작업을 수행하는 데 필요한 IAM 서비스 역할이 더 이상 필요하지 않은 경우 해당 역할을 삭제하는 것이 좋습니다.

자체 관리형 StackSets의 경우 사용자가 생성한 역할. 이 역할에 대한 자세한 내용은 [자체 관리형 권한 부여](#) 섹션을 참조하세요.

서비스 관리형 StackSet의 경우 StackSet용으로 자동으로 생성된 역할로, 조직 관리 계정에서는 CloudFormationStackSetsOrgAdmin 접미사를 사용하고 각 대상 계정에서는 CloudFormationStackSetsOrgMember 접미사를 사용합니다. 자세한 내용은 [서비스 연결 역할](#) 섹션을 참조하세요.

서비스 역할을 삭제하려면 다음을 수행하세요(콘솔).

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/iam/>에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택하고 삭제할 역할 옆에 있는 확인란을 선택합니다.
3. 페이지 상단의 역할 작업 메뉴에서 역할 삭제를 선택합니다.
4. 확인 대화 상자에서 [Yes, Delete]를 선택합니다. 확신한다면, 서비스 마지막 액세스 데이터가 로드되고 있을 때에도 삭제를 진행할 수 있습니다.

서비스 역할을 삭제하려면 다음을 수행하세요(AWS CLI).

- 다음 delete-role 명령을 사용합니다. 확인하라는 메시지가 표시되면 **y**를 입력하고 Enter 키를 누릅니다.

```
aws iam delete-role --role-name role name
```

역할 삭제에 대한 자세한 내용은 IAM 사용 설명서의 [역할 또는 인스턴스 프로파일 삭제](#)를 참조하세요.

대상 계정 게이트를 사용하여 StackSet 배포 실패 방지

계정 게이트는 옵션 기능으로, CloudFormation이 해당 계정에서 StackSet 작업을 시작하기 전에 대상 계정이 특정 요구 사항을 충족하는지 검증할 수 있게 해줍니다. 이 검증은 사전 조건 확인 역할을 하는 AWS Lambda 함수를 통해 수행됩니다.

계정 게이트의 일반적인 예제는 대상 계정에 활성 또는 해결되지 않은 CloudWatch 경보가 없는지 확인하는 것입니다. CloudFormation은 대상 계정에서 스택 작업을 시작할 때마다 Lambda 함수를 간접적으로 호출하며, 함수가 SUCCEEDED 코드를 반환하는 경우에만 계속됩니다. Lambda 함수가 FAILED 상태를 반환하면 CloudFormation은 요청된 작업을 계속하지 않습니다. Lambda 함수가 구성된 계정 게이트가 없는 경우 CloudFormation은 확인을 건너뛰고 작업을 계속합니다.

대상 계정에서 계정 게이트 확인에 실패하면 실패한 작업은 지정된 내결함성 수 또는 스택 비율에 반영됩니다. 내결함성에 대한 자세한 내용은 [StackSet 작업 옵션](#)을 확인하십시오.

계정 게이트는 StackSets 작업에 대해서만 사용할 수 있습니다. 이 기능은 StackSet 외부의 다른 CloudFormation 작업에는 사용할 수 없습니다.

요구 사항

계정 게이트 확인을 위해서는 다음 요구 사항을 충족해야 합니다.

- 이 기능을 사용하려면 Lambda 함수의 이름이 `AWSCloudFormationStackSetAccountGate`여야 합니다.
- `AWSCloudFormationStackSetExecutionRole`에는 Lambda 함수를 간접적으로 호출할 권한이 필요합니다. 이 권한이 없으면 CloudFormation이 계정 게이트 확인을 건너뛰고 스택 작업을 계속 진행합니다.
- 계정 게이트가 작동하려면 대상 계정에 Lambda `InvokeFunction` 권한이 추가되어야 합니다. 대상 계정 신뢰 정책에 관리자 계정과의 신뢰 관계가 있어야 합니다. 다음은 Lambda `InvokeFunction` 권한을 부여하는 정책 설명의 예입니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "lambda:InvokeFunction"
      ],
      "Resource": "*"
    }
  ]
}
```

Lambda 함수 생성을 위한 CloudFormation 템플릿

다음 샘플 템플릿을 사용하여 Lambda `AWSCloudFormationStackSetAccountGate` 함수를 생성합니다. 이러한 템플릿 중 하나를 사용하여 새 스택을 생성하려면 [CloudFormation 콘솔에서 스택 생성](#) 섹션을 참조하세요.

템플릿 위치	설명
https://s3.amazonaws.com/cloudformation-templates-us-east-1/AccountGateSucceeded.yml	SUCCEEDED 상태를 반환하는 Lambda 계정 게이트 함수를 구현하는 스택을 생성합니다.
https://s3.amazonaws.com/cloudformation-templates-us-east-1/AccountGateFailed.yml	FAILED 상태를 반환하는 Lambda 계정 게이트 함수를 구현하는 스택을 생성합니다.

CloudFormation StackSet에 대한 동시성 모드 선택

동시성 모드는 StackSet 작업 중에 동시성 수준이 작동하는 방식을 선택할 수 있는 [StackSetOperationPreferences](#)의 파라미터입니다. 다음 모드 중에서 선택할 수 있습니다.

- 엄격한 내결함성: 이 옵션은 실패한 계정 수가 내결함성 +1 값을 초과하지 않도록 동시성 수준을 동적으로 낮춥니다. 초기 실제 동시성은 최대 동시 계정 값이나 내결함성 +1 값 중 낮은 값으로 설정됩니다. 그러면 실제 동시성은 실패 횟수에 비례하여 감소합니다. 이는 기본 설정 동작입니다.
- 소프트 내결함성: 이 옵션은 내결함성을 실제 동시성에서 분리합니다. 이렇게 하면 실패 횟수에 관계없이 최대 동시 계정 값으로 설정된 동시성 수준에서 StackSet 작업을 실행할 수 있습니다.

엄격한 내결함성을 사용하면 StackSet 작업이 실패할 때마다 동시성이 감소하므로 배포 속도가 느려집니다. 소프트 내결함성을 사용하면 CloudFormation 안전 기능을 계속 활용하면서 배포 속도에 우선순위를 둡니다. 이를 통해 기존 리소스, 서비스 할당량 및 권한과 관련된 문제와 같은 일반적인 문제에 대한 StackSet 작업 실패를 검토하고 해결할 수 있습니다.

StackSets 스택 작업 실패에 대한 자세한 내용은 [일반적인 스택 작업 실패 이유](#)(를) 참조하세요.

최대 동시 계정 및 내결함성에 대한 자세한 내용은 [StackSet 작업 옵션](#)(를) 참조하세요.

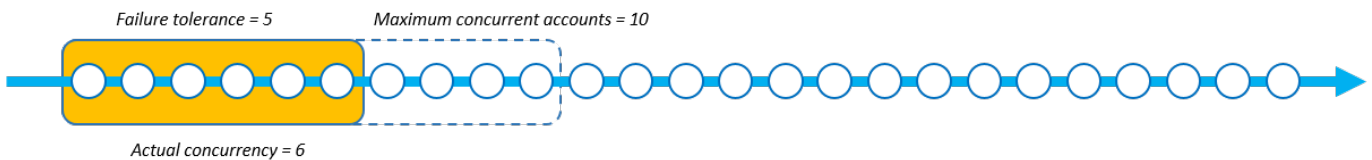
각 동시성 모드의 작동 방식

아래 이미지는 StackSet 작업 중에 각 동시성 모드가 작동하는 방식을 시각적으로 보여 줍니다. 노드 문자열은 단일 AWS 리전에 대한 배포를 나타내며 각 노드는 대상 AWS 계정입니다.

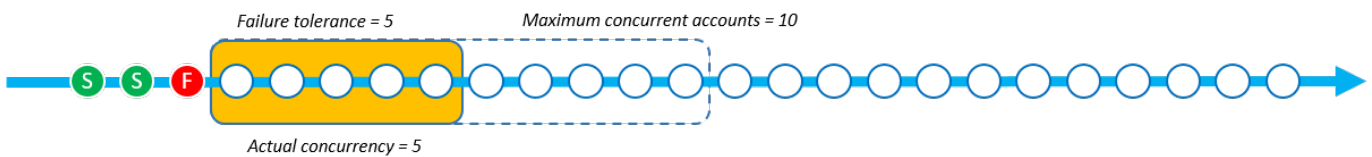
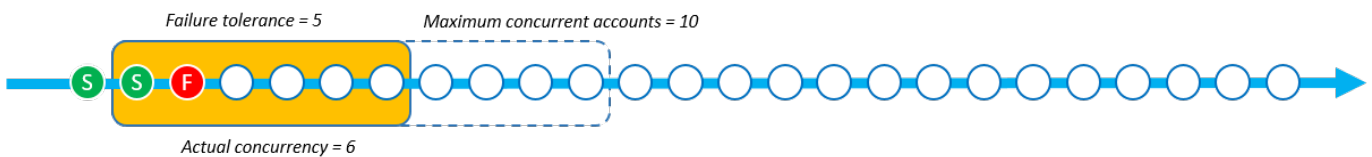
엄격한 내결함성

엄격한 내결함성을 사용하는 StackSet 작업의 내결함성 값이 5로 설정되고 최대 동시 계정 값이 10으로 설정된 경우 실제 동시성은 6입니다. 내결함성 값인 5 + 1이 최대 동시 계정 값보다 낮기 때문에 실제 동시성은 6입니다.

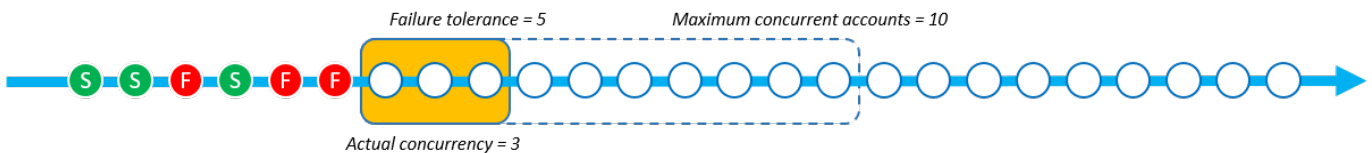
다음 이미지는 내결함성 값이 최대 동시 계정 값에 미치는 영향과 이 두 값이 StackSet 작업의 실제 동시성에 미치는 영향을 보여 줍니다.



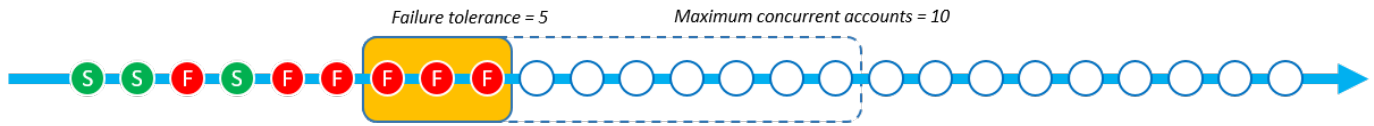
배포가 시작되고 실패한 스택 인스턴스가 있으면 안전한 배포 환경을 제공하기 위해 실제 동시성이 감소합니다. StackSets가 스택 인스턴스 1개를 배포하지 못하면 실제 동시성이 6에서 5로 감소합니다.



엄격한 내결함성은 실패한 스택 인스턴스 수에 비례하여 실제 동시성을 감소시킵니다. 다음 예에서 StackSets가 스택 인스턴스 2개를 더 배포하지 못하여 실패한 스택 인스턴스의 총 개수가 3개가 되면 실제 동시성이 5에서 3으로 감소합니다.



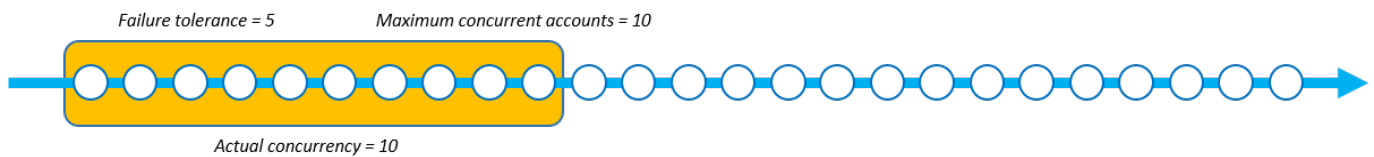
실패한 스택 인스턴스의 수가 정의된 내결함성 값 + 1과 같으면 StackSet 작업이 실패합니다. 다음 예에서 StackSets는 실패한 스택 인스턴스가 6개 있으며 내결함성 값이 5인 경우 작업에 실패합니다.



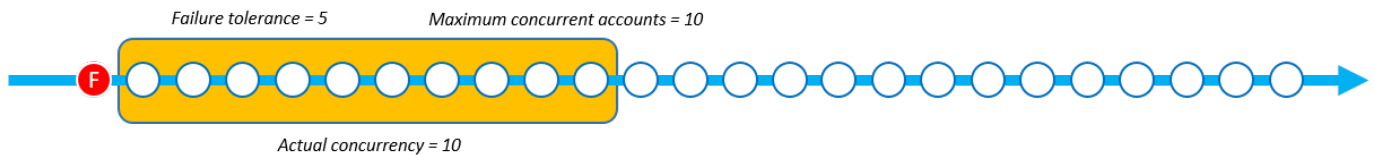
이 예에서 CloudFormation은 StackSet 작업을 중지하기 전에 스택 인스턴스 9개(성공 3개, 실패 6개)를 배포했습니다.

소프트 내결함성

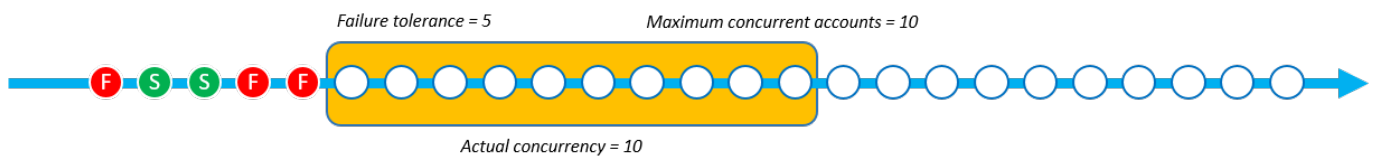
소프트 내결함성을 사용하는 StackSet 작업의 내결함성 값이 5로 설정되고 최대 동시 계정 값이 10으로 설정된 경우 실제 동시성은 10입니다.



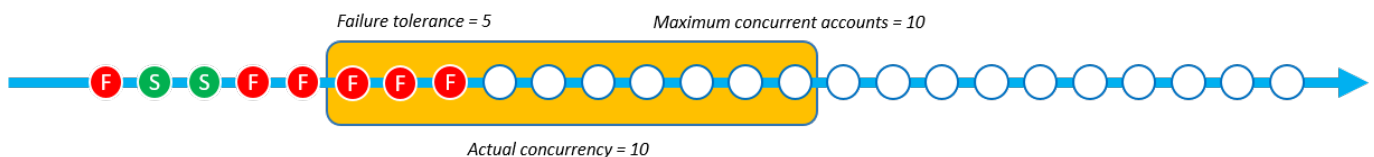
배포가 시작되고 실패한 스택 인스턴스가 있더라도 실제 동시성은 변경되지 않습니다. 다음 예에서 스택 작업은 1번 실패했지만 실제 동시성은 10으로 유지됩니다.



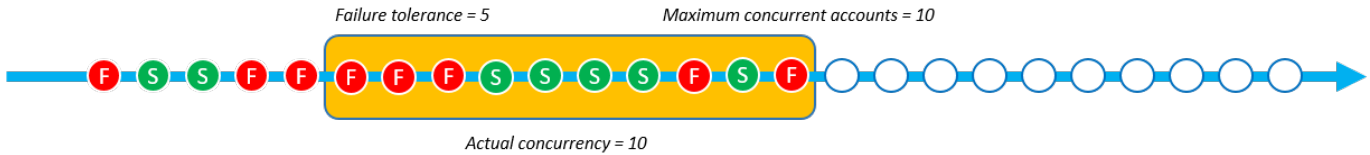
스택 인스턴스가 2번 더 실패한 후에도 실제 동시성은 10으로 유지됩니다.



실패한 스택 인스턴스가 내결함성 값을 초과하면 StackSet 작업이 실패합니다. 다음 예에서 StackSets는 실패한 스택 인스턴스가 6개 있으며 내결함성 수가 5인 경우 작업에 실패합니다. 하지만 동시성 대기열의 나머지 작업이 완료될 때까지 작업은 종료되지 않습니다.



StackSets는 이미 동시성 대기열에 있는 스택 인스턴스를 계속 배포합니다. 즉, 실패한 스택 인스턴스의 수가 내결함성보다 클 수 있습니다. 다음 예에서 StackSet 작업이 내결함성인 5에 도달했는데도 동시성 대기열에 아직 수행할 작업이 7개 남아 있으므로 실패한 스택 인스턴스가 8개 있습니다.



이 예에서 StackSets는 스택 작업을 중지하기 전에 스택 인스턴스 15개(성공 7개, 실패 8개)를 배포했습니다.

배포 속도에 따라 엄격한 내결함성과 소프트 내결함성 중에서 선택

엄격한 내결함성과 소프트 내결함성 모드 중에서 선택하는 것은 선호하는 StackSet 배포 속도와 허용되는 배포 실패 수에 따라 달라집니다.

다음 표는 총 1,000개의 스택 인스턴스를 배포하려고 시도하는 동안 각 동시성 모드가 실패한 StackSet 작업을 처리하는 방식을 보여 줍니다. 각 시나리오에서 내결함성 값은 스택 인스턴스 100개로 설정되고 최대 동시 계정 값은 스택 인스턴스 250개로 설정됩니다.

StackSets는 실제로는 계정을 슬라이딩 윈도우로 대기열에 추가하지만([각 동시성 모드의 작동 방식](#) 참조) 이 예에서는 각 모드의 속도를 보여 주기 위해 작업을 배치로 보여 줍니다.

엄격한 내결함성

엄격한 내결함성 모드를 사용하는 이 예에서는 각 이전 배치에서 발생하는 실패 수에 비례하여 실제 동시성을 낮춥니다. 각 배치에는 20개의 실패한 인스턴스가 있으며, 이로 인해 StackSet 작업이 내결함성 값인 100에 도달할 때까지 다음 배치의 실제 동시성이 20만큼 낮아집니다.

다음 표에서 첫 번째 배치의 초기 실제 동시성은 스택 인스턴스 101개입니다. 실제 동시성은 최대 동시 계정(250)과 내결함성(100) + 1 중 낮은 값이므로 101입니다. 각 배치에는 20개의 실패한 스택 인스턴스 배포가 포함되며, 이로 인해 각 다음 배치의 실제 동시성이 스택 인스턴스 20개만큼 낮아집니다.

엄격한 내결함성	배치 1	배치 2	배치 3	배치 4	배치 5	배치 6
실제 동시성 수	101	81	61	41	21	-

엄격한 내결함성	배치 1	배치 2	배치 3	배치 4	배치 5	배치 6
실패한 인스턴스 수	20	20	20	20	20	-
성공한 스택 인스턴스 수	81	61	41	21	1	-

엄격한 내결함성을 사용한 작업은 StackSet 작업이 스택 인스턴스 100개인 내결함성에 도달할 때까지 5개의 배치에서 305개의 스택 인스턴스 배포를 완료했습니다. StackSet 작업이 실패하기 전에 205개의 스택 인스턴스를 성공적으로 배포합니다.

소프트 내결함성

소프트 내결함성을 사용하는 이 예에서는 실패한 인스턴스 수에 관계없이 최대 동시 계정 값인 스택 인스턴스 250개로 정의된 실제 동시성 수를 동일하게 유지합니다. StackSet 작업이 내결함성 값인 인스턴스 100개에 도달할 때까지 동일한 실제 동시성을 유지합니다.

다음 표에서 첫 번째 배치의 초기 실제 동시성은 스택 인스턴스 250개입니다. 최대 동시 계정 값이 250으로 설정되어 있고 소프트 내결함성 모드에서는 StackSets가 실패 횟수에 관계없이 이 값을 실제 동시성으로 사용할 수 있기 때문에 실제 동시성은 250입니다. 이 예에서는 각 배치에서 실패가 50번 발생하더라도 실제 동시성은 영향을 받지 않습니다.

소프트 내결함성	배치 1	배치 2	배치 3	배치 4	배치 5	배치 6
실제 동시성 수	250	250	-	-	-	-
실패한 인스턴스 수	50	50	-	-	-	-
성공한 스택 인스턴스 수	200	200	-	-	-	-

동일한 최대 동시 계정 값과 내결함성 값을 사용하여 소프트 내결함성 모드를 사용한 작업은 2개의 배치에서 500개의 스택 인스턴스 배포를 완료했습니다. StackSet 작업이 실패하기 전에 400개의 스택 인스턴스를 성공적으로 배포합니다.

동시성 모드 선택(콘솔)

StackSet를 생성하거나 업데이트할 때 배포 옵션 설정 페이지의 동시성 모드에서 엄격한 내결함성 또는 소프트한 내결함성을 선택합니다.

동시성 모드 선택(AWS CLI)

다음 StackSets 명령에서 ConcurrencyMode 파라미터를 사용할 수 있습니다.

- [create-stack-instances](#)
- [delete-stack-instances](#)
- [detect-stack-set-drift](#)
- [import-stacks-to-stack-set](#)
- [update-stack-instances](#)
- [update-stack-set](#)

이러한 명령에는 ConcurrencyMode 설정을 사용할 수 있는 기존 파라미터 `--operation-preferences`이(가) 있습니다. ConcurrencyMode은(는) 다음 값 중 하나로 설정할 수 있습니다.

- STRICT_FAILURE_TOLERANCE
- SOFT_FAILURE_TOLERANCE

다음 예제에서는 FailureToleranceCount를 10으로 설정하고 MaxConcurrentCount를 5로 설정한 STRICT_FAILURE_TOLERANCE ConcurrencyMode를 사용하여 스택 인스턴스를 생성합니다.

```
aws cloudformation create-stack-instances \
  --stack-set-name example-stackset \
  --accounts 123456789012 \
  --regions eu-west-1 \
  --operation-preferences
  ConcurrencyMode=STRICT_FAILURE_TOLERANCE,FailureToleranceCount=10,MaxConcurrentCount=5
```

Note

StackSet을 생성하고 업데이트하는 자세한 절차는 다음 주제를 참조하세요.

- [StackSets 생성\(자체 관리형 권한\)](#)
- [StackSets 생성\(서비스 관리형 권한\)](#)
- [StackSets 업데이트](#)

CloudFormation StackSets에서 드리프트 감지 수행

CloudFormation을 통해 스택과 스택에 포함된 리소스를 관리하는 경우에도 사용자는 CloudFormation 외부에서 이러한 리소스를 변경할 수 있습니다. 사용자는 리소스를 생성하는 기본 서비스를 사용하여 리소스를 직접 편집할 수 있습니다. StackSet에 대해 드리프트 감지를 수행하여 해당 StackSet에 속하는 스택 인스턴스가 예상 구성과 다른지, 아니면 트리프트 되었는지 확인할 수 있습니다.

주제

- [CloudFormation이 StackSet에서 드리프트 감지를 수행하는 방법](#)
- [StackSet에서 드리프트 감지\(콘솔\)](#)
- [StackSet에서 드리프트 감지\(AWS CLI\)](#)
- [StackSet에서 드리프트 감지 중지](#)

CloudFormation이 StackSet에서 드리프트 감지를 수행하는 방법

CloudFormation은 StackSet에서 드리프트 감지를 수행할 때 StackSet의 각 스택 인스턴스와 연결되는 스택에 대해 드리프트 감지를 수행합니다. 이를 위해 CloudFormation에서는 스택의 각 리소스의 현재 상태를 스택의 템플릿 및 지정된 입력 파라미터에 정의된 대로 해당 리소스의 예상 상태와 비교합니다. 리소스의 현재 상태가 예상 상태와 다를 경우 해당 리소스가 드리프트된 것으로 간주됩니다. 스택에서 하나 이상의 리소스가 드리프트되면 스택 자체가 드리프트된 것으로 간주되고 스택과 연결된 스택 인스턴스도 드리프트된 것으로 간주됩니다. StackSet에서 하나 이상의 스택 인스턴스가 드리프트되면 StackSet 자체가 드리프트된 것으로 간주됩니다.

드리프트 감지는 비관리형 변경, 즉 CloudFormation 외부의 스택에 대한 변경을 식별합니다. StackSet 수준에서가 아니라 직접 CloudFormation을 통해 스택을 변경하는 것은 드리프트로 간주되지 않습니다. 예를 들어 StackSet의 스택 인스턴스와 연결된 스택이 있다고 가정합니다. CloudFormation을 사용하여 다른 템플릿을 사용하도록 해당 스택을 업데이트하는 경우에는 해당 스택에 StackSet에 속하는

다른 스택과 다른 템플릿이 있더라도 드리프트로 간주되지 않습니다. 스택이 CloudFormation의 예상 템플릿 및 파라미터 구성과 여전히 일치하기 때문입니다.

CloudFormation이 스택에서 드리프트 감지를 수행하는 방법에 대한 자세한 내용은 [드리프트 감지를 사용하여 스택 및 리소스에 대한 비관리형 구성 변경 감지](#) 섹션을 참조하세요.

CloudFormation은 각 스택에 대해 개별적으로 드리프트 감지를 수행하므로 스택이 드리프트되었는지 여부를 결정할 때 재정의된 파라미터값을 고려합니다. 스택 인스턴스의 템플릿 파라미터 재정의에 대한 자세한 내용은 [CloudFormation StackSet 내 스택에서 파라미터 값 재정의](#) 단원을 참조하십시오.

스택 인스턴스와 연결된 [스택에서 직접](#) 드리프트 감지를 수행하는 경우 StackSets 콘솔 페이지에서 이러한 드리프트 결과를 사용할 수 없습니다.

StackSet에서 드리프트 감지(콘솔)

StackSet에서 드리프트를 감지하려면

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. StackSet 페이지에서 드리프트 감지를 수행할 StackSet를 선택합니다.
3. 작업 메뉴에서 Detect drifts(드리프트 감지)를 선택합니다.

CloudFormation은 선택한 StackSet에 대해 드리프트 감지가 시작되었음을 나타내는 정보 표시줄을 표시합니다.

4. 선택 사항: 드리프트 감지 작업의 진행 상황을 모니터링하려면 다음을 수행합니다.
 - a. StackSet 이름을 선택하여 StackSet 세부 정보 페이지를 표시합니다.
 - b. 작업 탭을 선택하고 드리프트 감지 작업을 선택한 다음, View drift details(드리프트 세부 정보 보기)를 선택합니다.

CloudFormation에 [작업 세부 정보(Operation details)] 대화 상자가 표시됩니다.

5. CloudFormation에서 드리프트 감지 작업을 완료할 때까지 기다립니다. 드리프트 감지 작업이 완료되면 CloudFormation은 StackSet의 드리프트 상태 및 마지막 드리프트 확인 시간을 업데이트합니다. 이러한 필드는 선택한 StackSet에 대한 StackSet 세부 정보 페이지의 개요 탭에 나열됩니다.

드리프트 감지 작업은 StackSet에 포함된 스택 인스턴스의 수와 StackSet에 포함된 리소스 수에 따라 다소 시간이 걸릴 수 있습니다. 지정된 StackSet에서 한 번에 하나의 드리프트 감지 작업만 실행할 수 있습니다. CloudFormation은 정보 표시줄을 닫은 후에도 드리프트 감지 작업을 계속합니다.

6. StackSet의 스택 인스턴스에 대한 드리프트 감지 결과를 검토하려면 스택 인스턴스 탭을 선택합니다.

스택 이름 옆에는 각 스택 인스턴스와 연결된 스택의 이름이 나열되고 드리프트 상태 옆에는 해당 스택의 드리프트 상태가 나열됩니다. 스택은 하나 이상의 리소스가 드리프트 된 경우 드리프트 된 것으로 간주됩니다.

7. 특정 스택 인스턴스와 연결된 스택에 대한 드리프트 감지 결과를 검토하려면 다음을 수행합니다.
 - a. 작업 탭을 선택합니다.
 - b. 드리프트 감지 결과를 보려는 드리프트 작업을 선택합니다. 분할 패널에는 선택한 작업에 대한 스택 인스턴스 상태와 이유가 표시됩니다. 드리프트 작업의 경우 상태 이유 옆에 스택 인스턴스의 드리프트 상태가 표시됩니다.
 - c. 드리프트 세부 정보를 보려는 스택 인스턴스를 선택하고 View resource drift를 선택합니다. 리소스 드리프트 페이지의 리소스 드리프트 상태 테이블에서 각 스택 리소스는 드리프트 상태 및 리소스에서 드리프트 감지가 개시된 마지막 시간과 함께 나열됩니다. 각 리소스의 논리적 ID 및 물리적 ID가 표시되어 식별에 도움이 됩니다.
8. 드리프트 상태 옆을 사용하여 드리프트 상태에 따라 리소스를 정렬할 수 있습니다.

수정된 리소스의 세부 정보를 보려면

- 리소스를 선택한 상태에서 드리프트 세부 정보 보기를 선택합니다.

CloudFormation은 해당 리소스에 대한 드리프트 세부 정보 페이지를 표시합니다. 이 페이지에는 리소스의 차이점이 나열됩니다. 또한 리소스의 예상 및 현재 속성값이 나열됩니다.

Note

스택이 현재 로그인한 것과 다른 리전 및 계정에 속하는 경우 드리프트 감지 버튼이 비활성화되고 세부 정보를 볼 수 없습니다.

StackSet에서 드리프트 감지(AWS CLI)

AWS CLI를 사용하여 전체 스택에서 드리프트를 감지하려면 다음 절차를 사용하세요.

StackSet에서 드리프트를 감지하려면

1. [detect-stack-set-drift](#) 명령을 사용하여 전체 StackSet와 관련 스택 인스턴스에서 드리프트를 감지합니다.

다음 예는 StackSet `stack-set-drift-example`에서 드리프트 감지를 시작합니다.

```
aws cloudformation detect-stack-set-drift \  
  --stack-set-name stack-set-drift-example
```

출력:

```
{  
  "OperationId": "c36e44aa-3a83-411a-b503-cb611example"  
}
```

2. StackSet 드리프트 감지 작업은 오래 걸릴 수 있기 때문에 [describe-stack-set-operation](#) 명령을 사용하여 드리프트 감지 작업의 상태를 모니터링합니다. 이 명령은 `detect-stack-set-drift` 명령에서 반환된 StackSet 작업 ID를 가져옵니다.

다음 예에서는 이전 예의 작업 ID를 사용하여 StackSet 드리프트 감지 작업에 대한 정보를 반환합니다. 이 예제에서는 이러한 작업이 계속 실행 중입니다. 이 StackSet와 연결된 7개의 스택 인스턴스 중에서 스택 인스턴스 1개는 이미 드리프트된 것으로 확인되었고, 2개의 인스턴스는 동기화 중이며, 나머지 4개의 스택 인스턴스에 대한 드리프트 감지는 아직 진행 중입니다. 하나의 인스턴스가 드리프트되었으므로 StackSet 자체의 드리프트 상태는 이제 DRIFTED입니다.

```
aws cloudformation describe-stack-set-operation \  
  --stack-set-name stack-set-drift-example \  
  --operation-id c36e44aa-3a83-411a-b503-cb611example
```

출력:

```
{  
  "StackSetOperation": {  
    "Status": "RUNNING",  
    "AdministrationRoleARN": "arn:aws:iam::123456789012:role/  
AWSCloudFormationStackSetAdministrationRole",  
    "OperationPreferences": {  
      "RegionOrder": []  
    },  
  },  
}
```

```

    "ExecutionRoleName": "AWSCloudFormationStackSetExecutionRole",
    "StackSetDriftDetectionDetails": {
      "DriftedStackInstancesCount": 1,
      "TotalStackInstancesCount": 7,
      "LastDriftCheckTimestamp": "2019-12-04T20:34:28.543Z",
      "InSyncStackInstancesCount": 2,
      "InProgressStackInstancesCount": 4,
      "DriftStatus": "DRIFTED",
      "FailedStackInstancesCount": 0
    },
    "Action": "DETECT_DRIFT",
    "CreationTimestamp": "2019-12-04T20:33:13.673Z",
    "StackSetId": "stack-set-drift-example:bd1f4017-d4f9-432e-
a73f-8c22example",
    "OperationId": "c36e44aa-3a83-411a-b503-cb611example"
  }
}

```

이 예제에서는 나중에 동일한 명령을 수행하여 일단 드리프트 감지 작업이 완료되고 나면 반환되는 정보를 보여줍니다. 이 StackSet과 연결된 총 7개의 스택 인스턴스 중 2개가 드리프트되어 StackSet 자체의 드리프트 상태를 DRIFTED로 렌더링합니다.

```

aws cloudformation describe-stack-set-operation \
  --stack-set-name stack-set-drift-example \
  --operation-id c36e44aa-3a83-411a-b503-cb611example

```

출력:

```

{
  "StackSetOperation": {
    "Status": "SUCCEEDED",
    "AdministrationRoleARN": "arn:aws:iam::123456789012:role/
AWSCloudFormationStackSetAdministrationRole",
    "OperationPreferences": {
      "RegionOrder": []
    }
  },
  "ExecutionRoleName": "AWSCloudFormationStackSetExecutionRole",
  "EndTimestamp": "2019-12-04T20:37:32.829Z",
  "StackSetDriftDetectionDetails": {
    "DriftedStackInstancesCount": 2,
    "TotalStackInstancesCount": 7,
    "LastDriftCheckTimestamp": "2019-12-04T20:36:55.612Z",

```

```

        "InSyncStackInstancesCount": 5,
        "InProgressStackInstancesCount": 0,
        "DriftStatus": "DRIFTED",
        "FailedStackInstancesCount": 0
    },
    "Action": "DETECT_DRIFT",
    "CreationTimestamp": "2019-12-04T20:33:13.673Z",
    "StackSetId": "stack-set-drift-example:bd1f4017-d4f9-432e-
a73f-8c22example",
    "OperationId": "c36e44aa-3a83-411a-b503-cb611example"
}
}

```

3. StackSet 드리프트 감지 작업이 완료되면 `describe-stack-set`, `list-stack-instances`, `describe-stack-instance` 및 `list-stack-instance-resource-drifts` 명령을 사용하여 결과를 검토합니다.

[describe-stack-set](#) 명령에는 `describe-stack-set-operation` 명령에서 반환된 것과 동일한 상세 드리프트 정보가 포함되어 있습니다.

```

aws cloudformation describe-stack-set \
    --stack-set-name stack-set-drift-example

```

출력:

```

{
  "StackSet": {
    "Status": "ACTIVE",
    "Description": "Demonstration of drift detection on stack sets.",
    "Parameters": [],
    "Tags": [
      {
        "Value": "Drift detection",
        "Key": "Feature"
      }
    ],
    "ExecutionRoleName": "AWSCloudFormationStackSetExecutionRole",
    "Capabilities": [],
    "AdministrationRoleARN": "arn:aws:iam::123456789012:role/
AWSCloudFormationStackSetAdministrationRole",
    "StackSetDriftDetectionDetails": {
      "DriftedStackInstancesCount": 2,
      "TotalStackInstancesCount": 7,

```

```

        "LastDriftCheckTimestamp": "2019-12-04T20:36:55.612Z",
        "InProgressStackInstancesCount": 0,
        "DriftStatus": "DRIFTED",
        "DriftDetectionStatus": "COMPLETED",
        "InSyncStackInstancesCount": 5,
        "FailedStackInstancesCount": 0
    },
    "StackSetARN": "arn:aws:cloudformation:us-east-1:123456789012:stackset/stack-set-drift-example:bd1f4017-d4f9-432e-a73f-8c22example",
    "TemplateBody": [details omitted],
    "StackSetId": "stack-set-drift-example:bd1f4017-d4f9-432e-a73f-8c22ebexample",
    "StackSetName": "stack-set-drift-example"
}
}

```

[list-stack-instances](#) 명령을 사용하여 각 스택 인스턴스의 드리프트 상태를 포함하여 StackSet과 연결된 스택 인스턴스에 대한 요약 정보를 반환할 수 있습니다.

이 예의 예시 StackSet에서 드리프트 상태 필터를 DRIFTED로 설정하여 list-stack-instances를 실행하면 드리프트 상태가 DRIFTED인 두 스택 인스턴스를 식별할 수 있습니다.

```

aws cloudformation list-stack-instances \
  --stack-set-name stack-set-drift-example \
  --filters Name=DRIFT_STATUS,Values=DRIFTED

```

출력:

```

{
  "Summaries": [
    {
      "StackId": "arn:aws:cloudformation:eu-west-1:123456789012:stack/StackSet-stack-set-drift-example-b0fb6083-60c0-4e39-af15-2f071e0db90c/0e4f0940-16d4-11ea-93d8-0641cexample",
      "Status": "CURRENT",
      "Account": "012345678910",
      "Region": "eu-west-1",
      "LastDriftCheckTimestamp": "2019-12-04T20:37:32.687Z",
      "DriftStatus": "DRIFTED",
      "StackSetId": "stack-set-drift-example:bd1f4017-d4f9-432e-a73f-8c22eexample",
      "LastOperationId": "c36e44aa-3a83-411a-b503-cb611example"
    }
  ]
}

```

```

    },
    {
      "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
StackSet-stack-set-drift-example-b7fde68e-e541-44c2-b33d-
ef2e2988071a/008e6030-16d4-11ea-8090-12f89example",
      "Status": "CURRENT",
      "Account": "123456789012",
      "Region": "us-east-1",
      "LastDriftCheckTimestamp": "2019-12-04T20:34:28.275Z",
      "DriftStatus": "DRIFTED",
      "StackSetId": "stack-set-drift-example:bd1f4017-d4f9-432e-
a73f-8c22eexample",
      "LastOperationId": "c36e44aa-3a83-411a-b503-cb611example"
    },
    [additional stack instances omitted]
  ]
}

```

또한 [describe-stack-instance](#) 명령은 아래 예제와 같이 단일 스택 인스턴스에 대해서도 이 정보를 반환합니다.

```

aws cloudformation describe-stack-instance \
  --stack-set-name stack-set-drift-example \
  --stack-instance-account 012345678910 --stack-instance-region us-east-1

```

출력:

```

{
  "StackInstance": {
    "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/
StackSet-stack-set-drift-example-b7fde68e-e541-44c2-b33d-
ef2e2988071a/008e6030-16d4-11ea-8090-12f89example",
    "Status": "CURRENT",
    "Account": "123456789012",
    "Region": "us-east-1",
    "ParameterOverrides": [],
    "DriftStatus": "DRIFTED",
    "LastDriftCheckTimestamp": "2019-12-04T20:34:28.275Z",
    "StackSetId": "stack-set-drift-example:bd1f4017-d4f9-432e-
a73f-8c22eexample",

```

```

    "LastOperationId": "c36e44aa-3a83-411a-b503-cb611example"
  }
}

```

4. 어떤 스택 인스턴스가 드리프트되었는지 확인한 후에는 `list-stack-instances` 또는 `describe-stack-instance` 명령에 의해 반환되는 스택 인스턴스에 대한 정보를 사용하여 [list-stack-instance-resource-drifts](#) 명령을 실행할 수 있습니다. 이 명령은 특정 드리프트 작업에서 스택의 어떤 리소스가 드리프트되었는지에 대한 자세한 정보를 반환합니다.

다음 예제에서는 `--stack-instance-resource-drift-statuses` 파라미터를 사용하여 이전 드리프트 작업 예제에서 수정 또는 삭제된 리소스에 대한 스택 드리프트 정보를 요청합니다. 요청을 통해 수정된 리소스에 대한 정보가 반환되며, 여기에는 두 속성과 변경된 값에 대한 세부 정보가 포함됩니다. 삭제된 리소스가 없었습니다.

```

aws cloudformation list-stack-instance-resource-drifts \
  --stack-set-name my-stack-set-with-resource-drift \
  --stack-instance-account 123456789012 \
  --stack-instance-region us-east-1 \
  --operation-id c36e44aa-3a83-411a-b503-cb611example \
  --stack-instance-resource-drift-statuses MODIFIED DELETED

```

출력:

```

{
  "Summaries": [
    {
      "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/my-stack-set-with-resource-drift/489e5570-df85-11e7-a7d9-50example",
      "ResourceType": "AWS::SQS::Queue",
      "Timestamp": "2018-03-26T17:23:34.489Z",
      "PhysicalResourceId": "https://sqs.us-east-1.amazonaws.com/123456789012/my-stack-with-resource-drift-Queue-494PBHC076H4",
      "StackResourceDriftStatus": "MODIFIED",
      "PropertyDifferences": [
        {
          "PropertyPath": "/DelaySeconds",
          "ActualValue": "120",
          "ExpectedValue": "20",
          "DifferenceType": "NOT_EQUAL"
        },
        {
          "PropertyPath": "/RedrivePolicy/maxReceiveCount",

```

```

        "ActualValue": "12",
        "ExpectedValue": "10",
        "DifferenceType": "NOT_EQUAL"
      }
    ],
    "LogicalResourceId": "Queue"
  }
]
}

```

StackSet에서 드리프트 감지 중지

StackSet에서의 드리프트 감지는 시간이 오래 걸릴 수 있으므로 StackSet에서 현재 실행 중인 드리프트 감지 작업을 중지하려는 경우가 있을 수 있습니다.

StackSet에서 드리프트 감지를 중지하려면(콘솔)

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. StackSet 페이지에서 StackSet의 이름을 선택합니다.

CloudFormation에는 선택한 StackSet에 대한 StackSet 세부 정보 페이지가 표시됩니다.

3. StackSets details(스택 세트 세부 정보) 페이지에서 작업 탭을 선택한 다음, 드리프트 감지 작업을 선택합니다.
4. Stop operation(중지 작업)을 선택합니다.

StackSet에서 드리프트 감지를 중지하려면(AWS CLI)

- [stop-stack-set-operation](#) 명령을 사용합니다. 드리프트 감지 StackSet 작업의 StackSet 이름과 작업 ID를 모두 제공해야 합니다.

```

aws cloudformation stop-stack-set-operation \
  --stack-set-name stack-set-drift-example \
  --operation-id 624af370-311a-11e8-b6b7-500cexample

```


CloudFormation StackSet로 스택 가져오기

스택 가져오기 작업은 기존 스택을 새 스택 또는 기존 StackSet로 가져올 수 있으므로 한 번의 작업으로 기존 스택을 StackSet로 마이그레이션할 수 있습니다.

자체 관리형 StackSets의 경우 가져오기 작업은 관리자 계정 또는 다른 대상 계정 및 AWS 리전의 스택을 가져올 수 있습니다. 서비스 관리형 StackSets의 경우 가져오기 작업은 관리 계정과 동일한 AWS Organizations의 모든 스택을 가져올 수 있습니다.

다음은 StackSets로 스택을 가져올 때 고려 사항과 제한 사항입니다.

- 가져오기 작업은 인라인 스택 ID를 사용하여 최대 10개의 스택을 가져오거나 Amazon S3 객체를 사용하여 최대 200개의 스택을 가져올 수 있습니다.
- NoEcho 속성이 지원되지 않습니다. NoEcho가 포함된 스택은 StackSet 가져오기를 통해 새 StackSet로 가져오지 못합니다.
- 스택은 한 StackSet에만 속할 수 있습니다.
- 스택 가져오기 작업에서 태그를 파라미터로 명시적으로 지정하여 StackSet에 스택 태그를 구현할 수 있습니다.
- 스택의 사용자 지정 파라미터 재정의는 가져오기 작업 중 영향을 받지 않습니다.
- 가져온 스택 인스턴스의 파라미터는 StackSet 세부 정보 편집 옵션을 사용하여 업데이트할 수 없습니다. StackSet 파라미터 재정의의 사용해야 합니다. 파라미터 재정의에 대한 자세한 내용은 [스택에서 파라미터 재정의](#) 섹션을 참조하세요.
- 스택을 가져올 때 StackSets 할당량 및 스택 인스턴스가 적용됩니다. 할당량에 대한 자세한 내용은 [CloudFormation 할당량 이해](#) 섹션을 참조하세요.

주제

- [CloudFormation StackSets에 대한 자체 관리형 스택 가져오기](#)
- [CloudFormation StackSets에 대한 서비스 관리형 스택 가져오기](#)
- [CloudFormation StackSet로 스택 가져오기 되돌리기](#)

CloudFormation StackSets에 대한 자체 관리형 스택 가져오기

CloudFormation 스택 가져오기 작업은 기존 스택을 새 스택 또는 기존 StackSet로 가져올 수 있으므로 한 번의 작업으로 기존 스택을 StackSet로 마이그레이션할 수 있습니다. 스택 가져오기를 사용하

면 해당 리소스를 삭제하고 다시 생성하지 않고도 가동 중지 및 종단을 방지할 수 있습니다. 스택을 StackSet로 가져오면 원래 스택이 지정된 StackSet의 스택 인스턴스가 됩니다.

자체 관리형 스택 가져오기에 대한 고려 사항

- 스택 가져오기 작업에는 StackSet를 생성하는 관리자 계정과 스택이 포함된 대상 계정이 필요합니다.
- 대상 계정에는 스택 ID 또는 ARN 입력으로 GetTemplate 작업을 사용할 수 있는 권한이 있어야 합니다. 따라서 관리자 계정에 AWSCloudFormationStackSetAdministrationRole 또는 AWSCloudFormationStackSetsExecutionRole 권한이 부여되어야 합니다.

주제

- [기존 StackSet를 새 StackSet로 가져오기\(콘솔\)](#)
- [기존 StackSet를 기존 StackSet로 가져오기\(콘솔\)](#)
- [스택을 StackSet로 가져오기\(AWS CLI\)](#)

기존 StackSet를 새 StackSet로 가져오기(콘솔)

시작하기 전에 가져올 스택을 식별합니다.

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 탐색 창에서 스택 세트를 선택합니다.
3. 스택 세트 페이지의 상단에서 스택 세트 만들기를 선택하세요.
4. 템플릿 지정 페이지에서 다음 옵션 중 하나로 템플릿을 지정하고 다음을 선택합니다.
 - Amazon S3 URL을 선택하고 텍스트 상자에 템플릿의 URL을 지정합니다.
 - Upload a template file(템플릿 파일 업로드)을 선택하고 템플릿을 찾습니다.
 - From stack ID(스택 ID에서)를 선택하고 스택 ID를 입력합니다.
5. StackSet 세부 정보 지정 페이지에서 생성하려는 StackSet의 이름을 입력하고 다음을 선택합니다.
(선택 사항) StackSet에 대한 설명을 입력합니다.
6. StackSet 옵션 구성 페이지에서 선택 사항을 검토하고 다음을 선택합니다.
7. 배포 옵션 설정 페이지에서 스택 세트로 스택 가져오기를 선택합니다.

8. Stacks to import(가져올 스택) 필드에 가져올 스택의 스택 ID를 입력합니다. 예를 들어 `arn:aws:cloudformation:us-east-1:123456789012:stack/StackToImport/f449b250-b969-11e0-a185-5081d0136786`입니다.

(선택 사항) 다른 스택 ID 추가를 선택하고 가져올 다른 스택의 스택 ID를 입력합니다. 스택 가져오기 작업당 최대 10개의 스택을 추가할 수 있습니다.

9. 배포 옵션을 검토하고 다음을 선택합니다.

10. 검토 페이지에서 선택 내용과 StackSet 속성을 검토합니다. 스택을 StackSet로 가져올 준비가 되면 제출을 선택합니다.

결과: 가져온 스택은 이제 지정된 StackSet의 스택 인스턴스입니다. 스택 가져오기 상태에 대해 자세히 알아보려면 [StackSets 상태 코드](#) 섹션을 참조하세요.

기존 StackSet를 기존 StackSet로 가져오기(콘솔)

시작하기 전에 가져올 스택을 식별합니다.

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 탐색 창에서 스택 세트를 선택합니다.
3. StackSet 페이지에서 스택을 가져올 StackSet를 선택합니다.
4. StackSet를 선택한 상태로 작업 메뉴에서 StackSet에 스택 추가를 선택합니다.
5. 배포 옵션 설정 페이지에서 스택 세트로 스택 가져오기를 선택하고 가져올 스택 필드에 가져올 스택의 스택 ID를 입력합니다. 예를 들어 `arn:aws:cloudformation:us-east-1:123456789012:stack/StackToImport/f449b250-b969-11e0-a185-5081d0136786`입니다.

(선택 사항) 다른 스택 ID 추가를 선택하고 가져올 다른 스택의 스택 ID를 입력합니다. 스택 가져오기 작업당 최대 10개의 스택을 추가할 수 있습니다.

6. 다음을 선택합니다.
7. 재정의 지정 페이지에서 선택 사항을 검토하고 다음을 선택합니다.
8. 검토 페이지에서 선택 내용과 StackSet 속성을 검토합니다. StackSet를 생성할 준비가 되면 제출을 선택합니다.

결과: 가져온 스택은 이제 지정된 StackSet의 스택 인스턴스입니다. 스택 가져오기 상태에 대해 자세히 알아보려면 [StackSets 상태 코드](#) 섹션을 참조하세요.

스택을 StackSet로 가져오기(AWS CLI)

기존 스택을 새 StackSet로 가져오려면

다음 `create-stack-set` 명령은 StackSet를 생성하고 지정된 스택을 가져옵니다. 가져올 스택은 ARN으로 식별됩니다. 각 자리 표시자 텍스트를 자신의 정보로 바꿉니다.

```
aws cloudformation create-stack-set \  
  --stack-set-name MyStackSet \  
  --stack-id arn:aws:cloudformation:us-east-1:123456789012:stack/  
StackToImport/466df9e0-0dff-08e3-8e2f-5088487c4896 \  
  --administration-role-arn arn:aws:iam::123456789012:role/  
AWSCloudFormationStackSetAdministrationRole \  
  --execution-role-name AWSCloudFormationStackSetExecutionRole
```

기존 StackSet를 기존 StackSet로 가져오려면

다음 `import-stacks-to-stack-sets` 명령은 지정된 스택을 *MyStackSet* StackSet로 가져옵니다. 가져올 스택은 ARN으로 식별됩니다. 각 자리 표시자 텍스트를 자신의 정보로 바꿉니다.

```
aws cloudformation import-stacks-to-stack-set \  
  --stack-set MyStackSet \  
  --stack-ids arn:aws:cloudformation:us-east-1:123456789012:stack/StackToImport/  
f449b250-b969-11e0-a185-5081d0136786
```

스택을 2개 이상 지정하려면 `--stack-ids` 옵션의 값에 다음 형식을 사용합니다.

```
--stack-ids "arn_1" "arn_2"
```

가져온 스택을 다른 리전과 계정으로 복제

`create-stack-instances` 명령을 실행하여 스택 인스턴스를 StackSet에 추가합니다. 각 자리 표시자 텍스트를 자신의 정보로 바꿉니다.

```
aws cloudformation create-stack-instances \  
  --stack-set-name MyStackSet \  
  --accounts '["account_ID_1", "account_ID_2"]' \  
  --regions '["region_1", "region_2"]'
```

CloudFormation StackSets에 대한 서비스 관리형 스택 가져오기

CloudFormation 스택 가져오기 작업은 기존 스택을 새 스택 또는 기존 StackSet로 가져올 수 있으므로 한 번의 작업으로 기존 스택을 StackSet로 마이그레이션할 수 있습니다. StackSets는 단일 작업으로 여러 계정 및 리전에 대해 스택을 생성, 업데이트 또는 삭제할 수 있도록 하여 스택의 기능을 확장합니다.

서비스 관리형 스택 가져오기에 대한 고려 사항

- 스택 가져오기 작업에는 StackSets와의 신뢰 액세스 활성화와 같이 연결된 AWS Organizations를 관리할 수 있는 관리 계정 또는 위임된 관리자 계정이 필요합니다.
- 대상 계정은 관리 계정 또는 위임된 관리자 계정에서 관리하는 AWS Organizations의 멤버여야 합니다.
- 대상 스택은 대상 OU 중 하나에 있습니다.
- 대상 계정은 AWS Organizations의 멤버여야 합니다.
- AWS Organizations 액세스는 Organizations에 대해 ACTIVATED 상태여야 합니다.
- 가져오는 스택은 관리 계정이 아닌 모든 멤버 계정에 있어야 합니다.

주제

- [새 StackSet로 서비스 관리형 스택 가져오기\(콘솔\)](#)
- [서비스 관리형 스택 생성 및 기존 StackSet로 가져오기\(콘솔\)](#)
- [기존 StackSet로 서비스 관리형 스택 가져오기\(콘솔\)](#)
- [StackSet로 서비스 관리형 스택 가져오기\(AWS CLI\)](#)

새 StackSet로 서비스 관리형 스택 가져오기(콘솔)

AWS Management Console을 사용하여 새 StackSet로 스택 가져오기

StackSet로 새 스택을 가져오려면 가져올 리소스가 포함된 스택을 식별합니다.

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 탐색 창에서 스택 세트를 선택합니다.
3. 스택 세트 페이지의 상단에서 스택 세트 만들기를 선택하세요.
4. 템플릿 선택 페이지에서 다음을 수행합니다.

- a. StackSet 권한 모델에서 서비스 관리형 권한을 선택합니다.
- b. 사전 조건 - 템플릿 준비에서 템플릿 준비 완료를 선택하고 다음 옵션 중 하나를 사용하여 템플릿을 선택합니다.
 - Amazon S3 URL의 경우 Amazon S3 URL 필드에 Amazon S3 URL을 입력합니다.
 - 템플릿 파일 업로드에서 로컬 컴퓨터의 CloudFormation 템플릿을 선택합니다.

설정을 수락하고 Next(다음)를 선택합니다.

5. StackSet 세부 정보 지정 페이지에서 다음 작업을 수행합니다.
 - a. StackSet 이름 상자에 StackSet 이름을 입력합니다.
 - b. (옵션) StackSet description(스택 세트 설명)에 설명을 입력합니다.

StackSet 옵션 구성 페이지에서 선택 사항을 검토하고 다음을 선택합니다.

6. 배포 옵션 설정 페이지에서 다음을 수행합니다.
 - a. 스택 세트에 스택 추가에서 스택 세트로 스택 가져오기를 선택합니다.
 - b. Stacks to import(가져올 스택)에서 스택 가져오기 방법을 선택합니다.
 - i. Stack ID(스택 ID)에서 스택 ID를 입력합니다.
 - ii. 스택 URL에 Amazon S3 URL을 입력합니다.
7. 조직 단위 연결에서 다음을 수행합니다.
 - a. 조직과 연결을 선택하여 루트 OU를 사용합니다.
 - b. 조직 단위(OU)와 연결을 선택하여 가져올 스택의 상위 OU ID를 입력합니다. 예를 들어 Stack 1과 Stack 2가 OU1 아래에 있고 Stack 3이 OU2 아래에 있는 경우 OU1과 OU2를 입력합니다.

설정을 수락하고 Next(다음)를 선택합니다.

8. Review(검토) 페이지에서 설정을 검토하고 Submit(제출)을 선택합니다.

서비스 관리형 스택 생성 및 기존 StackSet로 가져오기(콘솔)

새 StackSet로 기존 스택을 가져오려면 가져올 리소스가 포함된 스택을 식별합니다.

StackSet를 생성하고 스택을 가져오려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 탐색 창에서 스택 세트를 선택합니다.
3. 스택 세트 페이지의 상단에서 스택 세트 만들기를 선택하세요.
4. 템플릿 선택 페이지에서 다음을 수행합니다.
 - a. StackSet 권한 모델에서 서비스 관리형 권한을 선택합니다.
 - b. 사전 조건 - 템플릿 준비에서 템플릿 준비 완료를 선택하고 다음 옵션 중 하나를 사용하여 템플릿을 선택합니다.
 - Amazon S3 URL의 경우 Amazon S3 URL 필드에 Amazon S3 URL을 입력합니다.
 - 템플릿 파일 업로드에서 로컬 컴퓨터의 CloudFormation 템플릿을 선택합니다.

설정을 수락하고 Next(다음)를 선택합니다.

5. StackSet 세부 정보 지정 페이지에서 다음 작업을 수행합니다.
 - a. StackSet 이름 상자에 StackSet 이름을 입력합니다.
 - b. (옵션) StackSet description(스택 세트 설명)에 설명을 입력합니다.

StackSet 옵션 구성 페이지에서 선택 사항을 검토하고 다음을 선택합니다.

6. 배포 옵션 설정 페이지에서 다음을 수행합니다.
 - Add stacks to stack set(스택 세트에 스택 추가)에서 Deploy new stacks(새 스택 배포)를 선택합니다.
7. 조직 단위 연결 섹션에서 다음을 수행합니다.
 - a. 조직과 연결을 선택하여 루트 OU를 사용합니다.
 - b. 조직 단위(OU)와 연결을 선택하여 가져올 스택의 상위 OU ID를 입력합니다. 예를 들어 Stack 1과 Stack 2가 OU1 아래에 있고 Stack 3이 OU2 아래에 있는 경우 OU1과 OU2를 입력합니다.
8. Specify regions(리전 지정)와 Deployment options(배포 옵션)에서 선택 사항을 검토합니다.

설정을 수락하고 Next(다음)를 선택합니다.

9. Review(검토) 페이지에서 설정을 검토하고 Submit(제출)을 선택합니다.

기존 StackSet로 서비스 관리형 스택 가져오기(콘솔)

StackSet를 선택하고 가져올 스택을 식별합니다.

기존 StackSet로 스택 가져오려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
 2. 탐색 창에서 스택 세트를 선택합니다.
 3. 스택을 가져올 StackSet를 선택한 다음 작업 드롭다운에서 StackSet에 스택 추가를 선택합니다.
 4. 배포 옵션 설정 페이지에서 다음을 수행합니다.
 - a. 스택 세트에 스택 추가에서 스택 세트로 스택 가져오기를 선택합니다.
 - b. 가져올 스택에서 다음을 수행합니다.
 - i. Stack ID(스택 ID)에서 스택 ID를 입력합니다.
 - ii. 스택 URL에 Amazon S3 URL을 입력합니다.
 - c. 조직 단위 연결에서 다음을 수행합니다.
 - i. 조직과 연결을 선택하여 루트 OU를 사용합니다.
 - ii. 조직 단위(OU)와 연결을 선택하여 가져올 스택의 상위 OU ID를 입력합니다. 예를 들어 Stack 1과 Stack 2가 OU1 아래에 있고 Stack 3이 OU2 아래에 있는 경우 OU1과 OU2를 입력합니다.
- 설정을 수락하고 Next(다음)를 선택합니다.
5. Specify overrides(재정의 지정) 페이지를 검토하고 Next(다음)를 선택합니다.
 6. Review(검토) 페이지를 확인 및 검토하고 Submit(제출)을 선택합니다.

StackSet로 서비스 관리형 스택 가져오기(AWS CLI)

StackSet가 생성되면 가져올 스택의 스택 ID를 전달하여 스택을 가져올 수 있습니다. 매핑하려는 OU ID 목록을 전달할 수도 있습니다.

CloudFormation은 해당 OU 내에서 사용자 제공 스택을 가져오고 해당 OU를 StackSet의 배포 대상으로 사용합니다. 입력에 표시된 스택 ID는 내부적으로 입력된 OU ID 목록에서 가장 가까운 OU에 매핑됩니다. 스택이 입력 목록의 기존 OU ID에 속하지 않으면 AWS CLI는 StackNotFoundException 오류를 반환합니다.

import-stacks-to-stack-set 작업은 OU ID 입력의 스택에 대한 스택 인스턴스를 생성합니다. 다음 AWS CLI 예에서는 import-stacks-to-stack-set 작업을 사용하여 스택을 StackSet로 가져옵니다.

- import-stacks-to-stack-sets 작업을 사용하려면 스택 세트로 가져오려는 stack-ids 또는 stack-ids-url을 지정합니다.

```
aws cloudformation import-stacks-to-stack-set \
  --stack-set-name ServiceManagedStackSet \
  --stack-ids "arn:123456789012:us-east-1:Stack1" \
  --organizational-unit-ids ou-examplerootid111-exampleouid111
```

```
aws cloudformation import-stacks-to-stack-set \
  --stack-set-name ServiceManagedStackSet \
  --stack-ids-url https://amzn-s3-demo-bucket.s3.us-west-2.amazonaws.com/file-name.json \
  --organizational-unit-ids ou-examplerootid111-exampleouid111
```

Note

import-stacks-to-stack-sets 작업을 수행하려면 가져올 스택을 해당 특정 OU에 연결할 수 있도록 최소한 하나의 조직 구성 단위 ID(OU ID)를 지정해야 합니다. 이 작업은 연결된 OU의 다른 멤버 계정에 대한 스택 인스턴스를 생성하지 않습니다. 연결된 OU의 멤버 계정을 업데이트하려면 create-stack-instances 또는 update-stack-instances를 사용합니다.

create-stack-set는 사용자가 직접 업로드하거나 Amazon S3에서 제공한 템플릿을 사용하여 OU 아래의 모든 계정에 대한 스택 인스턴스를 생성합니다. 다음 AWS CLI 예에서는 create-stack-set 작업을 사용하여 스택을 새 StackSet로 가져옵니다.

- create-stack-set 작업을 사용하려면 StackSet 이름을 지정하고 스택을 새로 생성된 StackSet로 가져옵니다.

```
aws cloudformation create-stack-set \
  --template-url https://amzn-s3-demo-bucket.s3.us-west-2.amazonaws.com/file-name.json \
  --permission-model SERVICE_MANAGED \
```

```
--auto-deployment Enabled=true
```

CloudFormation StackSet로 스택 가져오기 되돌리기

스택 인스턴스에 원하지 않는 변경 사항이 있는 경우 스택 인스턴스 가져오기를 되돌릴 수 있습니다.

스택 인스턴스 가져오기를 되돌리면 CloudFormation은 StackSet에서 스택 인스턴스를 삭제하지만 스택의 리소스는 유지합니다.

스택 가져오기 태스크를 되돌리려면 다음 절차를 수행합니다.

1. 스택 인스턴스가 삭제된 후 유지하려는 각 리소스에 대해 Retain의 DeletionPolicy 속성을 지정합니다. 자세한 내용은 [가져오기 작업 되돌리기](#) 섹션을 참조하세요.
2. StackSet에서 스택 인스턴스를 삭제합니다. 자세한 내용은 [CloudFormation StackSet에서 스택 삭제](#) 섹션을 참조하세요.
3. StackSet를 삭제합니다. 자세한 내용은 [CloudFormation StackSet 삭제](#) 섹션을 참조하세요.

CloudFormation StackSet 사용 모범 사례

이 섹션에서는 StackSet 템플릿 정의, 스택 생성 또는 StackSet에 추가, StackSet 업데이트의 모범 사례를 설명합니다.

CloudFormation을 처음 사용하는 경우 [CloudFormation 모범 사례](#) 주제를 검토하여 CloudFormation을 더 효과적이고 안전하게 사용하는 데 도움이 되는 추가 권장 사항을 확인하세요.

주제

- [템플릿 정의](#)
- [스택 생성 또는 StackSet에 추가](#)
- [StackSet의 스택 업데이트](#)

템플릿 정의

- 여러 리전 내의 여러 계정에 걸쳐 표준화하려는 템플릿을 정의합니다.
- 템플릿을 생성할 때 동일한 계정의 둘 이상의 리전에서 글로벌 리소스(예: IAM 역할 및 Amazon S3 버킷)가 생성될 경우 해당 리소스에 이름 충돌이 없는지 확인합니다.

- StackSet에 단일 템플릿 및 파라미터 세트가 있습니다. StackSet과 연결된 모든 계정에 동일한 스택이 생성됩니다. 템플릿을 작성할 때 제어와 표준화가 적절히 균형을 이룰 수 있도록 충분히 세분화합니다.
- 템플릿을 Amazon S3 버킷에 저장하는 것이 좋습니다.

스택 생성 또는 StackSet에 추가

- 더 많은 수의 스택 인스턴스를 StackSet에 추가하기 전에 초기 StackSet에 스택 인스턴스를 추가할 수 있는지 확인합니다.
- 사용 사례에 맞는 배포(롤아웃) 옵션을 선택하십시오.
 - 더 보수적인 배포의 경우 최대 동시 계정을 1로 설정하고, 내결함성을 0으로 설정합니다. 영향력이 가장 낮은 리전이 리전 순서 목록의 첫 번째가 되도록 설정합니다. 리전 하나에서 시작합니다.
 - 더 빠르게 배포하려면, 필요에 따라 최대 동시 계정 및 내결함성의 값을 늘립니다.
- StackSet의 작업은 관련된 스택 인스턴스의 수에 따라 달라지며, 많은 시간이 걸릴 수 있습니다.

StackSet의 스택 업데이트

- 기본적으로 StackSet 업데이트 시 모든 스택 인스턴스가 업데이트됩니다. 두 리전 각각에 20개의 계정이 있는 경우 40개의 스택 인스턴스가 있으며, StackSet를 업데이트하면 모든 스택 인스턴스가 업데이트됩니다.

다량의 스택 인스턴스가 포함된 StackSet의 경우 템플릿의 업데이트된 버전을 테스트하는 것이 좋으며, 모든 스택 인스턴스 업데이트 전에 몇 개의 테스트 계정에 있는 스택 인스턴스를 선택적으로 업데이트합니다.

- StackSet 내에서 개별 스택 업데이트에 대해 세부적으로 제어하려면 여러 StackSet의 생성을 계획합니다.
- 다량의 스택이 있는 StackSet를 업데이트할 경우, 많은 시간이 걸릴 수 있습니다. 이 릴리스에서는 StackSet에 대해 한 번에 작업 하나만 허용됩니다. StackSet에 대해 다른 작업의 수행이 차단되지 않도록 업데이트를 계획합니다.

CloudFormation StackSet 샘플 템플릿

이 섹션에는 기업에서 CloudFormation StackSet를 사용하는 데 도움이 될 수 있는 몇 가지 샘플 CloudFormation 템플릿에 대한 링크가 포함되어 있습니다. 이 단원에 나열된 템플릿은 [AWS CloudTrail](#) 또는 [AWS Config](#) 및 내부의 규칙을 활성화합니다.

Important

Amazon S3 버킷에 대한 AWS Config 액세스를 허용할 때 보안 모범 사례로서 AWS:SourceAccount 조건을 사용하여 버킷 정책에서 액세스를 제한할 것을 강력히 권장합니다. 새 템플릿은 AWS:SourceAccount을(를) 포함하도록 업데이트되었습니다. 기존 버킷 정책이 이 보안 모범 사례를 따르지 않는 경우 이 보호를 포함하도록 해당 버킷 정책을 편집할 것을 강력히 권장합니다. 이렇게 하면 예상 사용자만을 대신하여 AWS Config에 액세스 권한이 부여됩니다.

설명	S3 링크
AWS CloudTrail 활성화	https://s3.amazonaws.com/cloudformation-stackset-sample-templates-us-east-1/EnabledAWSCloudtrail.yml
AWS Config 활성화	https://s3.amazonaws.com/cloudformation-stackset-sample-templates-us-east-1/EnabledAWSConfig.yml
중앙 로깅으로 AWS Config 활성화	https://s3.amazonaws.com/cloudformation-stackset-sample-templates-us-east-1/EnabledAWSConfigForOrganizations.yml
AWS 조직 전체 또는 특정 AWS 계정 전체에서 Amazon Data Lifecycle Manager 기본 정책 활성화	- EBS 스냅샷의 기본 정책 - https://s3.amazonaws.com/cloudformation-stackset-sample-templates-us-east-1/DataLifecycleManagerEBSSnapshotDefaultPolicy.yml - EBS 지원 AMI에 대한 기본 정책 - https://s3.amazonaws.com/cloudformation-stackset-

설명	S3 링크
CloudTrail이 활성화되었는지 확인하기 위해 AWS Config 규칙 구성	sample-templates-us-east-1/DataLifecycleManagerAMIDefaultPolicy.yaml https://s3.amazonaws.com/cloudformation-templates-us-east-1/ConfigRuleEncryptedVolumes.yml https://s3.amazonaws.com/cloudformation-templates-us-east-1/ConfigRuleCloudTrailEnabled.yml
루트 MFA가 활성화되었는지 확인하기 위해 AWS Config 규칙 구성	https://s3.amazonaws.com/cloudformation-templates-us-east-1/ConfigRuleRootAccountMFAEnabled.yml
EIP가 연결되었는지 확인하기 위해 AWS Config 규칙 구성	https://s3.amazonaws.com/cloudformation-templates-us-east-1/ConfigRuleEipAttached.yml
EBS 볼륨이 암호화되었는지 확인하기 위해 AWS Config 규칙 구성	https://s3.amazonaws.com/cloudformation-templates-us-east-1/ConfigRuleEncryptedVolumes.yml

CloudFormation StackSets 문제 해결

이 주제에는 몇 가지 일반적인 StackSets 문제와 이러한 문제의 해결 방법이 제시되어 있습니다.

주제

- [일반적인 스택 작업 실패 이유](#)
- [실패한 스택 생성 또는 업데이트 작업 재시도](#)
- [스택 인스턴스 삭제 실패](#)
- [스택 가져오기 작업 실패](#)
- [StackSets 작업에 대한 스택 인스턴스 실패 횟수](#)

일반적인 스택 작업 실패 이유

문제: 스택 작업이 실패하고 스택 인스턴스 상태가 OUTDATED입니다.

원인: 스택 작업이 실패하는 여러 가지 일반적인 원인이 있습니다.

- 템플릿에 지정된 리소스를 만들기 위한 대상 계정의 권한이 충분하지 않습니다.
- CloudFormation 템플릿에 오류가 있을 수 있습니다. CloudFormation에서 템플릿을 확인하고 StackSet를 생성하기 전에 오류를 수정합니다.
- 템플릿은 S3 버킷과 같이 고유해야 하지만 그렇지 않은 글로벌 리소스를 생성하려고 시도할 수 있습니다.
- 지정된 대상 계정 번호가 존재하지 않습니다. 마법사의 배포 옵션 설정(Set deployment options) 페이지에서 지정한 대상 계정 번호를 확인합니다.
- 관리자 계정과 대상 계정의 신뢰 관계가 없습니다.
- 템플릿에 지정된 최대 리소스 수가 이미 대상 계정에 있습니다. 예를 들어, 대상 계정에서 허용된 IAM 역할 한도에 도달해도 템플릿이 IAM 역할을 더 만들 수 있습니다.
- StackSet에 허용되는 최대 스택 수에 도달했습니다. StackSet당 최대 스택 수는 [CloudFormation 할당량 이해](#) 섹션을 참조하세요.

해결 방법: StackSet를 생성하기 전에 대상 및 관리자 계정에 필요한 권한에 대한 자세한 내용은 [관리자 계정의 모든 사용자에게 모든 대상 계정의 스택을 관리할 권한 제공](#) 섹션을 참조하세요.

실패한 스택 생성 또는 업데이트 작업 재시도

문제: 스택 생성 또는 업데이트 작업이 실패하고 스택 인스턴스 상태가 OUTDATED입니다. 스택 생성 또는 업데이트가 실패한 이유를 해결하려면 CloudFormation 콘솔을 열고 DELETED(생성 작업 실패의 경우) 또는 FAILED(업데이트 작업 실패의 경우) 상태의 스택 이벤트를 봅니다. 스택 이벤트를 찾아보고 상태 사유 열을 찾습니다. 상태 사유 값이 스택 작업이 실패한 이유를 설명합니다.

스택 생성 실패의 근본적인 원인을 해결하고 스택을 생성할 준비가 되면 다음 단계를 수행하십시오.

해결 방법: 스택 작업을 재시도하려면 다음 단계를 수행합니다.

1. 콘솔에서 작업이 실패한 스택이 포함된 StackSet를 선택합니다.
2. 작업 메뉴에서 Edit StackSet details(스택 세트 세부 정보 편집)를 선택하여 스택 생성 또는 업데이트를 다시 시도하십시오.

3. 템플릿 지정 페이지에서 동일한 CloudFormation 템플릿을 사용하려면 기본 옵션인 현재 템플릿 사용을 유지합니다. 템플릿이 변경되어 스택 작업이 실패하고 수정된 템플릿을 업로드하려는 경우 Amazon S3에 템플릿 업로드를 선택한 다음 찾아보기를 선택하여 업데이트된 템플릿을 선택합니다. 수정된 템플릿 업로드가 완료되면 다음을 선택합니다.
4. 스택 세부 정보 지정 페이지에서 템플릿 관련 파라미터를 변경하지 않으려면 다음을 선택합니다.
5. 원한다면 배포 설정 옵션(Set deployment options) 페이지에서 최대 동시 계정 및 내결함성의 기본 값을 변경합니다. 이러한 설정에 대한 자세한 내용은 [StackSet 작업 옵션](#) 섹션을 참조하세요.
6. 검토 페이지에서 선택 사항을 검토하고 확인란을 선택하여 필요한 IAM 기능을 승인합니다. 제출을 선택합니다.
7. 스택이 성공적으로 업데이트되지 않은 경우 스택 생성에 방해가 되는 근본적인 문제를 해결한 후에 이 절차를 반복합니다.

스택 인스턴스 삭제 실패

문제: 스택 삭제가 실패했습니다.

원인: 종료 방지 기능이 활성화된 모든 스택에 대해 스택 삭제가 실패합니다.

솔루션: 스택에 종료 방지 기능이 활성화되어 있는지 확인합니다. 종료 방지 기능이 활성화되어 있으면 비활성화한 다음 스택 인스턴스 삭제를 다시 수행하십시오.

스택 가져오기 작업 실패

문제: 스택 가져오기 작업이 기존 스택을 새 StackSet나 기존 StackSet로 가져오지 못합니다. 스택 인스턴스가 INOPERABLE 상태입니다.

해결 방법: 다음 태스크를 완료하여 스택 가져오기 태스크를 되돌립니다.

1. StackSet에서 스택 삭제 옵션을 사용하고 구성 중에 RetainStacks를 활성화한 다음 StackSet에서 스택 인스턴스 삭제를 계속합니다. 자세한 내용은 [CloudFormation StackSet에서 스택 삭제](#) 섹션을 참조하세요.
2. StackSet의 스택 인스턴스가 업데이트되어 INOPERABLE 스택 인스턴스가 제거된 것을 확인할 수 있습니다.
3. 가져오기 실패 오류에 따라 스택 인스턴스를 수정하고 스택 가져오기 작업을 다시 시도합니다.

StackSets 작업에 대한 스택 인스턴스 실패 횟수

스택 인스턴스 실패 횟수는 스택 인스턴스가 프로비저닝 또는 업데이트에 실패할 경우 알립니다. 이러한 스택 인스턴스는 다음 중 하나 이상의 이유로 배포되지 않았습니다.

- 비슷한 구성의 기존 리소스
- AWS Identity and Access Management(IAM) 역할 등의 누락된 종속 항목
- 기타 충돌 요인

최대 동시성으로 배포하려는 경우 최대 동시성 개수는 내결함성 개수보다 하나 더 많습니다. 예를 들어, 내결함성 개수가 9일 경우 최대 동시성 개수는 10 이하여야 합니다. 이렇게 하면 일부 스택 인스턴스가 업데이트에 실패해도 작업이 SUCCEEDED를 반환합니다. 내결함성 개수가 모든 실패를 허용하도록 설정되었으므로 새 스택 인스턴스 실패 횟수를 통해 작업이 조건부로만 성공했는지 확인할 수 있습니다.

AWS Management Console, AWS SDK 또는 AWS CLI 를 사용하여 실패 횟수 및 필터 스택 인스턴스를 가져와서 재배포되어야 하는 인스턴스를 결정할 수 있습니다.

콘솔 사용

실패한 스택 인스턴스 수를 보려면 다음을 수행하세요.

1. [CloudFormation 콘솔](#)을 열고 StackSet를 선택합니다.
2. StackSet를 선택한 다음 Operations(작업)를 선택합니다.
3. Status(상태) 열에서 상태를 선택하여 상태 세부 정보를 봅니다. 상태 세부 정보에서 특정 작업에 대해 실패한 스택 인스턴스 수를 확인할 수 있습니다.

작업에 대한 스택 인스턴스의 계정, 리전 및 상태를 보려면 다음을 수행하세요.

1. 상태 세부 정보에서 실패한 스택 인스턴스 수를 선택합니다. 예: Stack instances(스택 인스턴스): **<number of failed stack instances>**.
2. 패널 헤더를 선택하여 측면 패널을 확장합니다. 측면 패널의 결과는 선택한 작업이 완료된 후의 스택 인스턴스 상태입니다.

작업에 대한 현재 스택 인스턴스 세부 정보를 보려면 다음을 수행하세요.

1. Stack Instances(스택 인스턴스) 탭을 선택합니다.

2. Last operation ID(마지막 작업 ID)로 필터링합니다. 결과는 마지막 인스턴스 수정 작업의 현재 상태와 상태 이유입니다. 이 필터를 AWS account, AWS region, Detailed Status(세부 상태) 및 Drift status(드리프트 상태)와 함께 사용하여 검색 결과를 더 세분화할 수 있습니다.

AWS CLI 사용

실패한 스택 인스턴스 수를 확인하려면 `describe-stack-set-operation` 또는 `list-stack-set-operations`를 호출하고 `StatusDetails`를 봅니다.

```
aws cloudformation describe-stack-set-operation --stack-set-name ss1 \
  --operation-id 5550e62f-c822-4331-88fa-21c1d7bafc60
```

```
{
  "StackSetOperation": {
    "OperationId": "5550e62f-c822-4331-88fa-21c1d7bafc60",
    "StackSetId": "ss1:9101ca57-49fc-4a61-a5a6-4c97b8adb08f",
    "Action": "CREATE",
    "Status": "SUCCEEDED",
    "OperationPreferences": {
      "RegionOrder": [],
      "FailureToleranceCount": 10,
      "MaxConcurrentCount": 10
    },
    "AdministrationRoleARN": "arn:aws:iam::123456789012:role/AWSCloudFormationStackSetAdministrationRole",
    "ExecutionRoleName": "AWSCloudFormationStackSetExecutionRole",
    "CreationTimestamp": "2022-10-26T17:18:53.947000+00:00",
    "EndTimestamp": "2022-10-26T17:19:35.304000+00:00",
    "StatusDetails": {
      "FailedStackInstancesCount": 3
    }
  }
}
```

```
aws cloudformation list-stack-set-operations --stack-set-name ss1
```

```
{
  "Summaries": [
    {
      "OperationId": "5550e62f-c822-4331-88fa-21c1d7bafc60",
```

```

    "Action": "CREATE",
    "Status": "SUCCEEDED",
    "CreationTimestamp": "2022-10-26T17:18:53.947000+00:00",
    "EndTimestamp": "2022-10-26T17:19:35.304000+00:00",
    "StatusDetails": {
      "FailedStackInstancesCount": 3
    },
    "OperationPreferences": {
      "RegionOrder": [],
      "FailureToleranceCount": 10,
      "MaxConcurrentCount": 10
    }
  }
]
}

```

특정 작업에 대한 기록 개요를 보려면 `list-stack-set-operation-results`를 사용하여 작업이 완료된 후 각 스택 인스턴스의 상태와 상태 이유를 확인합니다. 다음 예제를 보고 `Status`와 `StatusReason`을 찾습니다.

```

aws cloudformation list-stack-set-operation-results --stack-set-name ss1 \
  --operation-id 5550e62f-c822-4331-88fa-21c1d7bafc60 --
  filters Name=OPERATION_RESULT_STATUS,Values=FAILED

```

```

{
  "Summaries": [
    {
      "Account": "123456789012",
      "Region": "us-west-2",
      "Status": "FAILED",
      "StatusReason": "Account 123456789012 should have
'AWSCloudFormationStackSetExecutionRole' role with trust relationship to Role
'AWSCloudFormationStackSetAdministrationRole'.",
      "AccountGateResult": {
        "Status": "SKIPPED",
        "StatusReason": "Account 123456789012 should have
'AWSCloudFormationStackSetExecutionRole' role with trust relationship to Role
'AWSCloudFormationStackSetAdministrationRole'."
      },
      "OrganizationalUnitId": ""
    },
    {

```

```

        "Account": "123456789012",
        "Region": "us-west-1",
        "Status": "FAILED",
        "StatusReason": "Account 123456789012 should have
'AWSCloudFormationStackSetExecutionRole' role with trust relationship to Role
'AWSCloudFormationStackSetAdministrationRole'.",
        "AccountGateResult": {
            "Status": "SKIPPED",
            "StatusReason": "Account 123456789012 should have
'AWSCloudFormationStackSetExecutionRole' role with trust relationship to Role
'AWSCloudFormationStackSetAdministrationRole'."
        },
        "OrganizationalUnitId": ""
    },
    {
        "Account": "123456789012",
        "Region": "us-east-1",
        "Status": "FAILED",
        "StatusReason": "Account 123456789012 should have
'AWSCloudFormationStackSetExecutionRole' role with trust relationship to Role
'AWSCloudFormationStackSetAdministrationRole'.",
        "AccountGateResult": {
            "Status": "SKIPPED",
            "StatusReason": "Account 123456789012 should have
'AWSCloudFormationStackSetExecutionRole' role with trust relationship to Role
'AWSCloudFormationStackSetAdministrationRole'."
        },
        "OrganizationalUnitId": ""
    }
]
}

```

DETAILED_STATUS 및 LAST_OPERATION_ID 필터와 함께 `list-stack-instances`를 사용하여 스택 인스턴스 배포를 시도한 마지막 작업에서 실패한 스택 인스턴스 목록을 가져옵니다. DETAILED_STATUS 및 LAST_OPERATION_ID가 있는 예제에서 `--filters` 플래그를 참조하세요.

```

aws cloudformation list-stack-instances --stack-set-name ss1 \
  --filters Name=DETAILED_STATUS,Values=FAILED Name=LAST_OPERATION_ID,Values=5550e62f-
c822-4331-88fa-21c1d7bafc60

```

```

{
  "Summaries": [

```

```

{
  "StackSetId": "ss1:9101ca57-49fc-4a61-a5a6-4c97b8adb08f",
  "Region": "us-east-1",
  "Account": "123456789012",
  "Status": "OUTDATED",
  "StatusReason": "Account 123456789012 should have
'AWSCloudFormationStackSetExecutionRole' role with trust relationship to Role
'AWSCloudFormationStackSetAdministrationRole'.",
  "StackInstanceStatus": {
    "DetailedStatus": "FAILED"
  },
  "OrganizationalUnitId": "",
  "DriftStatus": "NOT_CHECKED",
  "LastOperationId": "5550e62f-c822-4331-88fa-21c1d7bafc60"
},
{
  "StackSetId": "ss1:9101ca57-49fc-4a61-a5a6-4c97b8adb08f",
  "Region": "us-west-1",
  "Account": "123456789012",
  "Status": "OUTDATED",
  "StatusReason": "Account 123456789012 should have
'AWSCloudFormationStackSetExecutionRole' role with trust relationship to Role
'AWSCloudFormationStackSetAdministrationRole'.",
  "StackInstanceStatus": {
    "DetailedStatus": "FAILED"
  },
  "OrganizationalUnitId": "",
  "DriftStatus": "NOT_CHECKED",
  "LastOperationId": "5550e62f-c822-4331-88fa-21c1d7bafc60"
},
{
  "StackSetId": "ss1:9101ca57-49fc-4a61-a5a6-4c97b8adb08f",
  "Region": "us-west-2",
  "Account": "123456789012",
  "Status": "OUTDATED",
  "StatusReason": "Account 123456789012 should have
'AWSCloudFormationStackSetExecutionRole' role with trust relationship to Role
'AWSCloudFormationStackSetAdministrationRole'.",
  "StackInstanceStatus": {
    "DetailedStatus": "FAILED"
  },
  "OrganizationalUnitId": "",
  "DriftStatus": "NOT_CHECKED",
  "LastOperationId": "5550e62f-c822-4331-88fa-21c1d7bafc60"
}

```

```
    }
  ]
}
```

스택 인스턴스를 수정한 마지막 작업 ID를 찾으려면 `list-stack-instances` 또는 `describe-stack-instance`를 사용하여 `LastOperationId`를 가져옵니다.

```
aws cloudformation describe-stack-instance --stack-set-name ss1 \
  --stack-instance-account 123456789012 --stack-instance-region us-east-2
```

```
{
  "StackInstance": {
    "StackSetId": "ss1:9101ca57-49fc-4a61-a5a6-4c97b8adb08f",
    "Region": "us-west-2",
    "Account": "123456789012",
    "ParameterOverrides": [],
    "Status": "OUTDATED",
    "StackInstanceStatus": {
      "DetailedStatus": "FAILED"
    },
    "StatusReason": "Account 123456789012 should have
'AWSCloudFormationStackSetExecutionRole' role with trust relationship to Role
'AWSCloudFormationStackSetAdministrationRole'.",
    "OrganizationalUnitId": "",
    "DriftStatus": "NOT_CHECKED",
    "LastOperationId": "5550e62f-c822-4331-88fa-21c1d7bafc60"
  }
}
```

Git 동기화를 사용하여 Git 리포지토리에 저장된 소스 코드와 스택 동기화

Git 동기화를 사용하면 소스 제어를 통해 CloudFormation 스택을 관리할 수 있습니다. 이렇게 하려면 Git 리포지토리를 모니터링하도록 CloudFormation을 구성합니다. 리포지토리에서 다음 두 파일의 변경 사항을 모니터링합니다.

- 스택을 정의하는 CloudFormation 템플릿 파일
- 스택을 구성하는 파라미터가 포함된 스택 배포 파일

Git 동기화를 사용하면 풀 요청과 버전 추적을 사용하여 중앙 위치에서 CloudFormation 스택을 구성, 배포 및 업데이트할 수 있습니다. 변경 사항을 템플릿 또는 배포 파일에 커밋하면 CloudFormation이 스택을 자동으로 업데이트합니다. 풀 요청을 사용하는 경우 CloudFormation은 실제로 스택을 업데이트하기 전에 스택에 어떤 변경 사항이 적용될지 설명하는 코멘트를 풀 요청에 남길 수 있습니다. 그러나 먼저 이 기능을 활성화해야 합니다.

Git 동기화에서 제공하는 콘솔 인터페이스를 통해 리포지토리로 연결되는 링크를 사용하고, 스택 배포 파일을 생성하고, CloudFormation 템플릿을 업데이트하고, 리포지토리에 풀 요청을 제출할 수 있습니다. 또한 Git 동기화는 활성 Git 동기화 스택 배포의 모니터링, 편집 및 문제 해결에 사용할 수 있는 상태 대시보드도 제공합니다. Git 동기화는 [스택을 생성할 때 CloudFormation 콘솔](#)을 통해 액세스할 수 있습니다. CodeConnections를 사용하여 Git 동기화에 액세스할 수도 있습니다. 자세한 내용은 개발자 도구 콘솔 사용 설명서에서 [연결된 리포지토리에 대한 동기화 구성 작업](#)을 참조하세요.

Git 동기화는 [GitHub](#), [GitHub Enterprise](#), [GitLab](#), [Bitbucket](#) 및 [GitLab 자체 관리](#) 리포지토리를 지원합니다.

Note

Git 동기화는 미국 동부(버지니아 북부), 미국 동부(오하이오), 미국 서부(캘리포니아 북부), 미국 서부(오레곤), 캐나다(중부), 아시아 태평양(뭄바이), 아시아 태평양(도쿄), 아시아 태평양(서울), 아시아 태평양(싱가포르), 아시아 태평양(시드니), 유럽(아일랜드), 유럽(런던), 유럽(파리), 유럽(스톡홀름), 유럽(프랑크푸르트), 유럽(밀라노) 및 남아메리카(상파울루) 리전에서 사용 가능합니다.

다중 계정 전략으로 Git 동기화 사용에 대한 자세한 내용은 다음 블로그 게시물 [Use CloudFormation Git sync to configure resources in customer accounts](#)를 참조하세요.

주제

- [CloudFormation에서 Git 동기화 작동 방식](#)
- [Git 동기화를 사용하여 스택을 Git 리포지토리에 동기화하기 위한 사전 요구 사항](#)
- [Git 동기화를 사용하여 리포지토리 소스 코드에서 스택 생성](#)
- [CloudFormation의 풀 요청에서 스택 변경 요약 게시 활성화](#)
- [Git 동기화 상태 대시보드](#)

CloudFormation에서 Git 동기화 작동 방식

이 주제에서는 Git 동기화의 작동 방식을 설명하고 Git 동기화를 사용하는 데 필요한 주요 개념을 소개합니다.

주제

- [Git 동기화 작동 방식](#)
- [풀 요청에 대한 의견](#)
- [스택 배포 파일](#)
- [CloudFormation 템플릿 파일](#)
- [템플릿 정의 리포지토리](#)

Git 동기화 작동 방식

Git 동기화를 사용하려면 먼저 [CodeConnections](#) 서비스를 사용하여 Git 공급자를 CloudFormation에 연결해야 합니다. 이 안내서의 절차에서는 CodeConnections 콘솔을 통해 연결을 생성합니다. 또는 AWS CLI를 사용하여 연결을 생성할 수도 있습니다. 다음과 같은 Git 공급자를 사용할 수 있습니다.

- [GitHub](#)
- [GitHub Enterprise](#)
- [GitLab](#)
- [Bitbucket](#)
- [GitLab 자체 관리형](#)

다음으로 스택을 정의하는 CloudFormation 템플릿을 생성하고 리포지토리에 추가합니다. 이 템플릿 파일은 모니터링됩니다. CloudFormation은 스택에 변경 사항이 커밋되면 스택을 자동으로 업데이트합니다.

CloudFormation 콘솔에서 새 스택을 생성하고 Git에서 동기화를 선택하여 CloudFormation에 Git 동기화를 사용하도록 지시합니다. CloudFormation에서 모니터링할 리포지토리 및 브랜치를 지정하고 스택을 정의하는 리포지토리의 CloudFormation 템플릿을 지정합니다.

구성 중에 리포지토리에서 자체 스택 배포 파일을 제공하거나 Git 동기화에서 자동으로 파일을 생성하도록 할 수 있습니다. 스택 배포 파일에는 스택의 리소스를 구성하는 파라미터 및 값이 포함됩니다. 이 스택 배포 파일은 모니터링됩니다. CloudFormation은 스택에 변경 사항이 커밋되면 스택을 자동으로 업데이트합니다.

Git 동기화는 리포지토리에 풀 요청을 생성하여 스택을 CloudFormation 템플릿 파일 및 스택 배포 파일과 동기화합니다. Git 동기화에서 스택 배포 파일을 생성하는 경우 Git 동기화에서 해당 파일을 리포지토리에 제출합니다.

그리고 나서 CloudFormation에서 스택을 프로비저닝하고, 배포 파라미터로 스택을 구성하고, 리포지토리의 변경 사항 모니터링을 시작하도록 풀 요청을 리포지토리에 병합합니다.

그때부터 템플릿 파일 또는 스택 배포 파일을 변경하여 리포지토리에 커밋할 때마다 CloudFormation은 자동으로 변경 사항을 탐지합니다. 팀에서 풀 요청을 사용하는 경우 팀원이 변경 사항을 배포하기 전에 검토하고 승인할 수 있습니다. 풀 요청이 수락되면 CloudFormation은 변경 사항을 배포합니다.

CloudFormation 콘솔에서 스택의 Git 동기화 구성 상태와 더불어 스택에 적용된 커밋 기록을 볼 수 있습니다. 또한 콘솔에서는 Git 동기화를 재구성하고 문제를 해결하기 위한 도구도 제공합니다.

풀 요청에 대한 의견

콘솔에서 풀 요청에서 주석 활성화 옵션을 켜서 CloudFormation이 CodeConnections 서비스를 통해 풀 요청의 코드 변경 요약 생성하도록 할 수 있습니다. 풀 요청의 변경 요약을 제공하면 팀원이 풀 요청을 병합하기 전에 제안된 수정이 미치는 영향을 쉽게 검토하고 이해할 수 있습니다. 자세한 내용은 [CloudFormation의 풀 요청에서 스택 변경 요약 게시 활성화](#) 섹션을 참조하세요.

스택 배포 파일

스택 배포 파일은 CloudFormation 스택을 관리하는 파라미터 및 값이 포함된 JSON(JavaScript Object Notation) 또는 YAML 표준 형식의 파일입니다. 이 파일의 변경 사항이 모니터링됩니다. 파일의 변경 사항이 리포지토리에 커밋되면 관련 스택이 자동으로 업데이트됩니다.

스택 배포 파일에는 다음과 같이 하나의 키-값 페어와 두 개의 사전이 포함되어 있습니다.

- `template-file-path`

CloudFormation 템플릿 파일의 전체 리포지토리 경로입니다. 템플릿 파일은 이 배포 파일과 연결된 CloudFormation 스택의 리소스를 선언합니다.

- `parameters`

`parameters` 사전에는 스택의 리소스를 구성하는 키-값 페어가 포함됩니다. 스택 배포 파일에는 최대 50개의 파라미터가 포함될 수 있습니다.

- `tags`

`tags` 사전에는 스택의 리소스를 식별하고 분류하는 데 사용할 수 있는 선택적 키-값 페어가 포함됩니다. 스택 배포 파일에는 최대 50개의 태그가 포함될 수 있습니다.

직접 스택 배포 파일을 제공하거나 Git 동기화에서 스택 배포 파일을 생성하고 리포지토리에 풀 요청을 자동으로 제출하도록 할 수 있습니다. 스택 배포 파일을 편집하고 변경 사항을 리포지토리에 커밋하여 파라미터 및 태그를 관리할 수 있습니다.

다음은 Git 동기화 스택 배포 파일의 예시입니다.

```
template-file-path: fargate-srvs/my-stack-template.yaml

parameters:
  image: public.ecr.aws/lts/nginx:latest
  task_size: x-small
  max_capacity: 5
  port: 8080
  env: production
tags:
  cost-center: '123456'
  org: 'AWS'
```

CloudFormation 템플릿 파일

템플릿 파일에는 CloudFormation 스택을 구성하는 AWS 리소스의 선언이 포함됩니다. Git 동기화를 사용하면 템플릿 파일이 Git 리포지토리에 저장되고 스택 배포 파일에서 참조됩니다. 템플릿 파일을 편집하고 변경 사항을 리포지토리에 커밋하여 스택을 관리할 수 있습니다.

자세한 내용은 [CloudFormation 템플릿 작업](#) 섹션을 참조하세요.

템플릿 정의 리포지토리

템플릿 정의 리포지토리는 Git 동기화를 통해 CloudFormation에 연결되는 Git 리포지토리입니다. 이 리포지토리에서 CloudFormation 템플릿 및 스택 배포 파일의 변경 사항이 모니터링됩니다. 파일의 변경 사항을 커밋하면 연결된 스택이 자동으로 업데이트됩니다.

Important

Git 동기화 콘솔에서 템플릿 정의 리포지토리를 구성할 때 Git 연결에서 올바른 리포지토리 및 브랜치를 선택해야 합니다. Git 동기화는 구성된 리포지토리 및 브랜치에서만 CloudFormation 템플릿 및 스택 배포 파일의 변경 사항을 모니터링합니다.

Git 동기화를 사용하여 스택을 Git 리포지토리에 동기화하기 위한 사전 요구 사항

CloudFormation 스택을 Git 리포지토리에 동기화하기 전에 다음 요구 사항이 충족되는지 확인하세요.

주제

- [Git 리포지토리](#)
- [CloudFormation 템플릿](#)
- [Git 동기화 서비스 역할](#)
- [콘솔 사용자의 IAM 권한](#)

Git 리포지토리

다음 플랫폼 중 하나에 호스팅되는 Git 리포지토리가 있어야 합니다.

- [GitHub](#)
- [GitHub Enterprise](#)
- [GitLab](#)
- [Bitbucket](#)
- [GitLab 자체 관리형](#)

리포지토리는 퍼블릭 또는 프라이빗일 수 있습니다. [Connections 콘솔](#)을 통해 Git 리포지토리를 CloudFormation에 연결해야 합니다.

CloudFormation 템플릿

Git 동기화와 연결하려는 브랜치에 체크인된 [CloudFormation 템플릿 파일](#)이 Git 리포지토리에 포함되어 있어야 합니다. 이 템플릿은 [스택 배포 파일](#)에서 참조합니다.

Git 동기화 서비스 역할

Git 동기화에는 IAM 역할이 필요합니다. Git 동기화를 구성할 때 스택에 대해 IAM 역할을 생성하도록 선택하거나 기존 역할을 사용할 수 있습니다.

Note

자동으로 생성된 IAM 역할은 역할이 생성된 스택에만 권한을 적용합니다. 자동으로 생성된 IAM 역할을 재사용하려면 새 스택용으로 역할을 편집해야 합니다.

Git 동기화 서비스 역할에 필요한 권한

Git 동기화에 제공하는 IAM 역할에는 다음 권한이 필요합니다.

- `cloudformation:CreateChangeSet`
- `cloudformation>DeleteChangeSet`
- `cloudformation:DescribeChangeSet`
- `cloudformation:DescribeStackEvents`
- `cloudformation:DescribeStacks`
- `cloudformation:ExecuteChangeSet`
- `cloudformation:ListChangeSets`
- `cloudformation:ValidateTemplate`
- `events:PutRule`
- `events:PutTargets`

Note

위의 필수 권한은 Git 동기화에서 생성하는 IAM 역할에 자동으로 추가됩니다.

다음 IAM 역할 예시에는 Git 동기화에 대한 사전 필수 권한이 포함되어 있습니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "SyncToCloudFormation",
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateChangeSet",
        "cloudformation:DeleteChangeSet",
        "cloudformation:DescribeChangeSet",
        "cloudformation:DescribeStackEvents",
        "cloudformation:DescribeStacks",
        "cloudformation:ExecuteChangeSet",
        "cloudformation:GetTemplate",
        "cloudformation:ListChangeSets",
        "cloudformation:ListStacks",
        "cloudformation:ValidateTemplate"
      ],
      "Resource": "*"
    },
    {
      "Sid": "PolicyForManagedRules",
      "Effect": "Allow",
      "Action": [
        "events:PutRule",
        "events:PutTargets"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "events:ManagedBy":
            ["cloudformation.sync.codeconnections.amazonaws.com"]
        }
      }
    }
  ]
}
```

```

        }
    },
    {
        "Sid": "PolicyForDescribingRule",
        "Effect": "Allow",
        "Action": "events:DescribeRule",
        "Resource": "*"
    }
]
}

```

신뢰 정책

신뢰 관계를 정의하기 위해서는 역할을 생성할 때 다음과 같은 신뢰 정책을 제공해야 합니다.

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CfnGitSyncTrustPolicy",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudformation.sync.codeconnections.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}

```

대리인 혼동 문제로부터 스스로를 보호하려면 `aws:SourceArn` 및 `aws:SourceAccount` 조건 키를 사용할 것을 권장합니다. 소스 계정은 계정 ID이고 소스 ARN은 CloudFormation이 Git 리포지토리에 연결할 수 있도록 허용하는 [CodeConnections](#) 서비스 연결의 ARN입니다.

JSON

```

{
  "Version": "2012-10-17",

```

```

"Statement": [
  {
    "Sid": "CfnGitSyncTrustPolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": "cloudformation.sync.codeconnections.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      },
      "ArnLike": {
        "aws:SourceArn": "arn:aws:codeconnections:us-east-1:123456789012:connection/EXAMPLE64-8aad-4d5d-8878-dfcab0bc441f"
      }
    }
  }
]
}

```

혼동된 대리자 문제에 대한 자세한 내용은 [교차 서비스 혼동된 대리인 방지](#)를 참조하세요.

콘솔 사용자의 IAM 권한

CloudFormation 콘솔을 통해 Git 동기화를 성공적으로 설정하려면 최종 사용자에게 IAM을 통한 권한도 부여해야 합니다.

Git 리포지토리에 대한 연결을 생성하고 관리하려면 다음 codeconnections 권한이 필요합니다.

- codeconnections:CreateRepositoryLink
- codeconnections:CreateSyncConfiguration
- codeconnections>DeleteRepositoryLink
- codeconnections>DeleteSyncConfiguration
- codeconnections:GetRepositoryLink
- codeconnections:GetSyncConfiguration
- codeconnections:ListRepositoryLinks
- codeconnections:ListSyncConfigurations

- `codeconnections:ListTagsForResource`
- `codeconnections:TagResource`
- `codeconnections:UntagResource`
- `codeconnections:UpdateRepositoryLink`
- `codeconnections:UpdateSyncBlocker`
- `codeconnections:UpdateSyncConfiguration`
- `codeconnections:UseConnection`

또한 콘솔 사용자는 Git 동기화 설정 프로세스 중에 스택을 보고 관리할 수 있는 다음과 같은 `cloudformation` 권한이 있어야 합니다.

- `cloudformation:CreateChangeSet`
- `cloudformation>DeleteChangeSet`
- `cloudformation:DescribeChangeSet`
- `cloudformation:DescribeStackEvents`
- `cloudformation:DescribeStacks`
- `cloudformation:ExecuteChangeSet`
- `cloudformation:GetTemplate`
- `cloudformation:ListChangeSets`
- `cloudformation:ListStacks`
- `cloudformation:ValidateTemplate`

Note

콘솔 전용 사용에 변경 세트 권한(`cloudformation:CreateChangeSet`, `cloudformation>DeleteChangeSet`, `cloudformation:DescribeChangeSet`, `cloudformation:ExecuteChangeSet`)이 반드시 필요한 것은 아니지만 전체 스택 검사 및 관리 기능을 활성화하는 것이 좋습니다.

다음 예제 IAM 정책에는 콘솔을 통해 Git 동기화를 설정하는 데 필요한 사용자 권한이 포함되어 있습니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CodeConnectionsPermissions",
      "Effect": "Allow",
      "Action": [
        "codeconnections:CreateRepositoryLink",
        "codeconnections:CreateSyncConfiguration",
        "codeconnections>DeleteRepositoryLink",
        "codeconnections>DeleteSyncConfiguration",
        "codeconnections:GetRepositoryLink",
        "codeconnections:GetSyncConfiguration",
        "codeconnections:ListRepositoryLinks",
        "codeconnections:ListSyncConfigurations",
        "codeconnections:ListTagsForResource",
        "codeconnections:TagResource",
        "codeconnections:UntagResource",
        "codeconnections:UpdateRepositoryLink",
        "codeconnections:UpdateSyncBlocker",
        "codeconnections:UpdateSyncConfiguration",
        "codeconnections:UseConnection",
        "codeconnections:CreateForcedTargetSync",
        "codeconnections:CreatePullRequestForResource"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CloudFormationConsolePermissions",
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateChangeSet",
        "cloudformation>DeleteChangeSet",
        "cloudformation:DescribeChangeSet",
        "cloudformation:DescribeStackEvents",
        "cloudformation:DescribeStacks",
        "cloudformation:ExecuteChangeSet",
        "cloudformation:GetTemplate",
        "cloudformation:ListChangeSets",
        "cloudformation:ListStacks",
        "cloudformation:ValidateTemplate"
      ]
    }
  ]
}
```



```

    ],
    "Resource": "*"
  }
]
}

```

Note

codeconnections:CreateForcedTargetSync 및 codeconnections:CreatePullRequestForResource 권한이 포함된 IAM 정책을 생성할 때 IAM 콘솔에 이러한 작업이 존재하지 않는다는 경고가 표시될 수 있습니다. 이 경고를 무시해도 정책이 성공적으로 생성됩니다. 이러한 권한은 IAM 콘솔에서 인식되지 않더라도 특정 Git 동기화 작업에 필요합니다.

Git 동기화를 사용하여 리포지토리 소스 코드에서 스택 생성

이 주제에서는 Git 동기화를 사용하여 Git 리포지토리에 동기화되는 CloudFormation 스택을 생성하는 방법을 설명합니다.

Important

계속하기 전에 이전 섹션의 모든 [사전 요구 사항](#)을 완료하세요.

리포지토리 소스 코드에서 스택 생성

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 스택을 생성할 AWS 리전을 선택합니다.
3. 스택 페이지에서 스택 생성을 선택하고 새 리소스 사용(표준)을 선택합니다.
4. 스택 생성 페이지에서 다음을 수행합니다.
 - a. 사전 조건 - 템플릿 준비의 경우 기존 템플릿 선택을 선택한 상태로 유지합니다.
 - b. 템플릿 지정에서 Git에서 동기화를 선택하고 다음을 선택합니다.
5. 스택 세부 정보 지정 페이지에서 스택 이름에 스택의 이름을 입력합니다. 스택 이름에는 문자(A~Z 및 a~z), 숫자(0~9), 대시(-)를 포함할 수 있습니다.

6. 스택 배포 파일의 배포 파일 생성:

- 스택 배포 파일을 생성하고 리포지토리에 추가하지 않았다면 다음 파라미터를 사용하여 파일을 생성하고 내 리포지토리에 배치를 선택합니다.
- 리포지토리에서 스택 배포 파일을 생성한 경우 리포지토리의 자체 파일 제공을 선택합니다.

7. 템플릿 정의 리포지토리의 경우 연결된 Git 리포지토리 선택을 선택하여 CloudFormation에 이미 연결된 Git 리포지토리를 선택하거나 Git 리포지토리 연결을 선택하여 새 리포지토리를 연결합니다. Git 리포지토리 연결을 선택한 경우 다음을 수행하세요.

a. 리포지토리 공급자 선택에서 다음 중 하나를 선택합니다.

- GitHub
- GitHub Enterprise Server
- GitLab
- Bitbucket
- GitLab 자체 관리형

b. 연결의 목록에서 연결을 선택합니다. 연결 목록에 옵션이 표시되지 않는 경우 새 연결 추가를 선택하면 [연결 콘솔](#)로 이동하여 리포지토리에 대한 연결을 생성할 수 있습니다.

8. 리포지토리 목록에서 스택 템플릿 파일이 포함된 Git 리포지토리를 선택합니다.

9. 브랜치 목록에서 Git 동기화에서 모니터링하려는 브랜치를 선택합니다.

Note

Git 동기화는 선택한 브랜치에서 CloudFormation 템플릿 및 스택 배포 파일의 변경 사항만 모니터링합니다. 스택에 적용하려는 모든 변경 사항은 이 브랜치에 커밋해야 합니다.

10. 배포 파일 경로에 리포지토리 브랜치의 루트를 기준으로 스택 배포 파일 이름을 포함한 전체 경로를 지정합니다.

CloudFormation에서 파일을 생성하는 경우 이 경로는 리포지토리에서 파일이 커밋되는 위치입니다. 파일을 직접 제공하는 경우 이 경로는 리포지토리에서 파일의 위치입니다.

11. IAM 역할을 추가합니다. IAM 역할에는 CloudFormation에서 Git 리포지토리의 스택을 동기화하는데 필요한 권한이 포함됩니다. 새 IAM 역할을 선택하여 새 역할을 생성하거나 기존 IAM 역할을 선택하여 AWS 계정에서 기존 역할을 선택할 수 있습니다. 새 역할을 생성하도록 선택하면 필요한 권한이 역할에 포함됩니다.

12. 폴 요청에 대한 주석 활성화 또는 비활성화:

- CloudFormation에서 스택 업데이트에 대한 풀 요청에 변경 세트를 게시하도록 하려면 풀 요청에서 주석 활성화 토글을 켜 상태로 유지합니다.
 - 이 토글을 끄면 CloudFormation은 리포지토리 파일이 업데이트될 때 현재 스택 구성과 풀 요청에서 제안된 변경 사항 간의 차이를 설명하지 않습니다.
13. 템플릿 파일 경로에 스택 템플릿 파일에 대한 리포지토리 루트를 기준으로 전체 경로를 지정합니다.
 14. (선택 사항) 스택 파라미터를 지정하려면 파라미터 추가를 선택하고 각 파라미터에 대한 키와 값을 입력한 후 다음을 선택합니다. 자세한 내용은 [스택 배포 파일](#) 섹션을 참조하세요.

예를 들어, 스택 배포 파일에서 **port=8080** 파라미터를 지정하려면 다음 작업을 수행합니다.

- a. Add parameter(파라미터 추가)를 선택합니다.
 - b. 키(Key)에 **port**를 입력합니다.
 - c. 값에 **8080**을(를) 입력합니다.
15. (선택 사항) 스택 태그를 지정하려면 새 태그 추가를 선택하고 각 태그에 태그 키와 값을 제공한 후 다음을 선택합니다. 자세한 내용은 [스택 배포 파일](#) 섹션을 참조하세요.
 16. 다음을 선택하여 스택 옵션 구성을 계속 진행합니다. 스택 옵션의 구성에 대한 자세한 내용은 [스택 옵션 구성](#)을 참조하세요.

스택 구성을 완료했으면 다음을 선택하여 계속 진행합니다.

17. 스택 설정을 검토하고 다음을 확인합니다.
 - 스택 템플릿이 올바르게 구성되고 Git에서 동기화로 설정되어 있습니다.
 - 배포 파일이 올바르게 구성되어 있습니다.
 - 템플릿 정의 리포지토리가 올바르게 구성되어 있고, 특히 리포지토리와 브랜치 이름이 올바르게 선택되어 있습니다.
 - 배포 파일의 미리 보기가 정확하며 필요한 파라미터와 값을 포함하고 있습니다.
18. 제출을 선택하여 스택을 생성합니다.

제출을 선택하면 Git 리포지토리에서 풀 요청이 자동으로 생성됩니다. 이 풀 요청을 Git 리포지토리에 병합하여 스택을 생성해야 합니다. 스택이 생성되면 CloudFormation에서 Git 리포지토리의 변경 사항을 모니터링합니다.

Git 리포지토리에서 스택 업데이트

스택을 업데이트하려면 템플릿 파일 또는 Git 리포지토리의 스택 배포 파일을 직접 변경합니다. 모니터링되는 브랜치에 변경 사항을 커밋하면 CloudFormation에서 자동으로 스택을 업데이트합니다. 풀 요청을 사용하는 경우 스택이 업데이트되기 전에 Git 리포지토리에 풀 요청이 자동으로 생성됩니다. 이 풀 요청을 Git 리포지토리에 병합하여 스택을 업데이트해야 합니다.

CloudFormation 콘솔에서 스택을 선택하고 Git 동기화 탭을 선택하여 스택 및 동기화 이벤트의 상태에 대한 정보를 볼 수 있습니다. 자세한 내용은 [Git 동기화 상태 대시보드](#) 섹션을 참조하세요.

CloudFormation의 풀 요청에서 스택 변경 요약 게시 활성화

이 주제에서는 CloudFormation이 Git 리포지토리의 풀 요청에 스택 변경 요약을 게시할 수 있도록 하는 방법을 보여줍니다.

풀 요청에서 주석을 활성화하면 CloudFormation은 리포지토리 파일이 업데이트될 때 현재 스택 구성과 제안된 변경 사항 간의 차이를 설명하는 주석을 게시할 수 있습니다. 이 주석은 추가, 수정 또는 삭제될 리소스의 요약을 제공하므로, 풀 요청을 병합하기 전에 철저하게 코드를 검토할 수 있습니다.

새 스택에 대한 풀 요청에서 주석 활성화(콘솔)

스택을 생성할 때 스택 세부 정보 지정 페이지의 템플릿 정의 리포지토리에서 풀 요청에서 주석 활성화 토글이 켜져 있는지 확인합니다. 이는 새 스택에서 기본으로 설정되어 있습니다.

기존 스택에 대한 풀 요청에서 주석 활성화(콘솔)

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 스택을 생성한 AWS 리전을 선택합니다.
3. 스택 페이지에서 업데이트하려는 실행 중인 스택을 선택합니다.
4. Git 동기화 탭을 선택하고, 편집을 선택합니다.
5. Git 동기화 설정 편집 페이지의 템플릿 정의 리포지토리에서 풀 요청에서 주석 활성화 토글을 켭니다.
6. Update configuration(구성 업데이트)을 선택합니다.

Git 동기화 상태 대시보드

AWS CloudFormation Git 동기화 배포의 상태를 보려면 CloudFormation 콘솔에서 스택을 선택하고 Git 동기화 탭을 선택합니다.

Git 동기화 탭은 Git 동기화 상태와 최신 동기화 이벤트라는 두 패널로 구분됩니다.

Git 동기화 상태

상단 패널에서는 스택의 Git 동기화 구성에 대한 다음 정보를 제공합니다.

리포지토리

Git 동기화에 연결된 리포지토리에 대한 링크

리포지토리 공급자

리포지토리 공급자의 이름

브랜치

Git 동기화가 모니터링 중인 브랜치의 이름

배포 파일 경로

이 스택에 대한 스택 배포 파일의 전체 경로

리포지토리 동기화 상태

가장 최근 동기화 작업의 상태

리포지토리 동기화 상태 메시지

가장 최근 동기화 작업의 메시지

Git 동기화 상태

이 스택에 대한 Git 동기화의 상태

프로비저닝 상태

프로비저닝 작업의 상태

패널 오른쪽 상단에서 다음 버튼을 사용하여 Git 동기화를 수정하거나 업데이트합니다.

- 편집 - Git 동기화 구성을 편집합니다.
- 최근 커밋 재시도 - 리포지토리에의 최근 커밋에 따라 스택을 업데이트합니다.

- 연결 해제 - 스택에서 Git 동기화 연결을 해제합니다.
- 새로 고침 - Git 동기화 상태 패널을 새로 고칩니다.

최신 동기화 이벤트

최신 동기화 이벤트 패널에는 스택에 적용된 커밋의 테이블이 표시됩니다.

각 열의 헤더에 있는 화살표를 사용하여 테이블을 정렬할 수 있습니다. 다음에 따라 테이블을 오름차순 또는 내림차순으로 정렬할 수 있습니다.

- 날짜
- 커밋 ID
- Event
- 날짜
- 이벤트 유형

지원되는 스택 상태

스택이 다음 지원되는 상태 중 하나인 경우에만 스택에 Git 동기화를 구성할 수 있습니다.

- CREATE_COMPLETE
- UPDATE_COMPLETE
- UPDATE_ROLLBACK_COMPLETE
- IMPORT_COMPLETE
- IMPORT_ROLLBACK_COMPLETE

다음 테이블에는 스택 상태 코드의 전체 목록과 설명이 나와 있습니다.

스택 상태 및 선택적 세부 상태	설명
CREATE_COMPLETE	하나 이상의 스택을 성공적으로 생성했습니다.
CREATE_IN_PROGRESS	하나 이상의 스택을 생성하고 있습니다.
CREATE_FAILED	하나 이상의 스택을 생성하지 못했습니다. 스택 이벤트를 참조하여 관련 오류 메시지를 확인하세요. 가능한 생성 실패 원인에

스택 상태 및 선택적 세부 상태	설명
	는 스택 내의 모든 리소스로 작업할 수 있는 권한 부족, AWS 서비스에서 파라미터값 거부, 또는 리소스 생성 중 제한 시간 초과가 포함될 수 있습니다.
DELETE_COMPLETE	하나 이상의 스택을 성공적으로 삭제했습니다. 삭제된 스택은 90일 동안 유지되며 볼 수 있습니다.
DELETE_FAILED	하나 이상의 스택을 삭제하지 못했습니다. 삭제에 실패했으므로 일부 리소스가 아직 실행 중일 수 있습니다. 하지만 스택으로 작업하거나 스택을 업데이트할 수 없습니다. 스택을 다시 삭제하거나 스택 이벤트를 참조하여 관련 오류 메시지를 확인하세요.
DELETE_IN_PROGRESS	하나 이상의 스택을 제거하고 있습니다.
REVIEW_IN_PROGRESS	템플릿 또는 리소스 없이 예상 StackId를 사용하여 하나 이상의 스택을 생성하고 있습니다.
	 Important 이 상태 코드를 가진 스택은 가능한 최대 스택 수에 포함됩니다.
ROLLBACK_COMPLETE	스택 생성이 실패하거나 스택 생성을 명시적으로 취소한 이후에 하나 이상의 스택을 성공적으로 제거했습니다. 스택은 이전 작업 상태로 돌아갑니다. 스택 생성 작업 중에 생성된 모든 리소스가 삭제됩니다. 스택 생성이 실패한 후에만 이 상태가 존재합니다. 이 상태는 부분적으로 생성된 스택의 모든 작업이 적절하게 정리되었음을 나타냅니다. 이 상태에서는 삭제 작업만 수행할 수 있습니다.
ROLLBACK_FAILED	스택 생성이 실패하거나 스택 생성을 명시적으로 취소한 이후에 하나 이상의 스택을 제거하지 못했습니다. 스택을 삭제하십시오. 연결된 오류 메시지는 스택 이벤트를 참조하십시오.

스택 상태 및 선택적 세부 상태	설명
ROLLBACK_IN_PROGRESS	스택 생성이 실패하거나 스택 생성을 명시적으로 취소한 이후에 하나 이상의 스택을 제거하고 있습니다.
UPDATE_COMPLETE	하나 이상의 스택을 성공적으로 업데이트했습니다.
UPDATE_COMPLETE_CLEANUP_IN_PROGRESS	스택을 업데이트한 후 하나 이상의 스택에 대한 이전 리소스를 제거하고 있습니다. 스택 업데이트 중에 리소스를 대체해야 하는 경우 CloudFormation에서는 새 리소스를 생성한 다음 이전 리소스를 삭제하여 스택 종단을 줄여줍니다. 이 상태에서는 스택이 업데이트되고 사용 가능하지만 CloudFormation에서는 이전 리소스를 계속 삭제하고 있습니다.
UPDATE_FAILED	하나 이상의 스택을 업데이트하지 못했습니다. 스택 이벤트를 참조하여 관련 오류 메시지를 확인하세요.
UPDATE_IN_PROGRESS	하나 이상의 스택을 업데이트하고 있습니다.
UPDATE_ROLLBACK_COMPLETE	스택 업데이트를 실패한 이후에 하나 이상의 스택을 이전 작업 상태로 되돌렸습니다.
UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS	스택 업데이트를 실패한 후 하나 이상의 스택에 대한 새 리소스를 제거하고 있습니다. 이 상태에서는 스택이 이전 작업 상태를 롤백되고 사용 가능하지만 CloudFormation에서는 스택 업데이트 중에 생성된 새 리소스를 계속 삭제하고 있습니다.
UPDATE_ROLLBACK_FAILED	스택 업데이트를 실패한 이후에 하나 이상의 스택을 이전 작업 상태로 되돌리지 못했습니다. 이 상태에서는 스택 또는 continue rollback 을 삭제할 수 있습니다. 스택이 작동 상태로 돌아가기 전에 오류를 수정해야 할 수도 있습니다. 또는 지원에 문의하여 스택을 사용 가능한 상태로 복원할 수 있습니다.
UPDATE_ROLLBACK_IN_PROGRESS	스택 업데이트를 실패한 이후에 하나 이상의 스택을 이전 작업 상태로 되돌리고 있습니다.
IMPORT_IN_PROGRESS	가져오기 작업이 현재 진행 중입니다.

스택 상태 및 선택적 세부 상태	설명
IMPORT_COMPLETE	resource import를 지원하는 스택의 모든 리소스에 대해 가져오기 작업이 성공적으로 완료되었습니다.
IMPORT_ROLLBACK_IN_PROGRESS	가져오기는 이전 템플릿 구성으로 롤백됩니다.
IMPORT_ROLLBACK_FAILED	스택에 있는 하나 이상의 리소스에 대해 가져오기 롤백 작업이 실패했습니다. CloudFormation이 성공적으로 가져온 리소스에 대해 결과를 사용할 수 있습니다.
IMPORT_ROLLBACK_COMPLETE	가져오기가 이전 템플릿 구성으로 성공적으로 롤백되었습니다.

CloudFormation 레지스트리를 통해 익스텐션 관리

CloudFormation 레지스트리는 익스텐션을 관리하기 위한 중앙 집중식 허브 역할을 하며, 사용자의 AWS 계정 템플릿에 통합할 수 있습니다. 확장 프로그램에는 AWS 및 서드 파티 게시자의 리소스 유형, 모듈, 후크와 자체 사용자 지정 확장 프로그램이 포함됩니다. 레지스트리를 사용하면 AWS 제공 리소스를 사용하는 것과 같은 방식으로 CloudFormation 템플릿에서 익스텐션을 더 쉽게 검색하고 프로비저닝할 수 있습니다.

이 섹션에서는 CloudFormation 레지스트리를 사용하여 계정에서 다음과 같은 서드 파티 확장 프로그램을 활성화하는 방법을 설명합니다.

- 퍼블릭 익스텐션 활성화
- 비공개 확장 프로그램 등록 및 활성화

주제

- [관련 설명서](#)
- [CloudFormation 레지스트리 개념](#)
- [CloudFormation 레지스트리에서 사용 가능하고 활성화된 확장 프로그램 보기](#)
- [CloudFormation 레지스트리에서 서드 파티 공개 확장 프로그램 사용](#)
- [사용자와 공유된 서드 파티 비공개 확장 프로그램 사용](#)
- [계정의 확장 프로그램에 대한 구성 데이터 편집](#)
- [AWS Config에서 리소스 유형 기록](#)

관련 설명서

자체 확장 프로그램을 생성하는 데 관심이 있는 개발자라면 다음 설명서를 참조하세요.

- CloudFormation Command Line Interface User Guide의 [Developing modules using the CloudFormation CLI](#)
- CloudFormation Command Line Interface User Guide의 [Creating resource types using the CloudFormation CLI](#)
- CloudFormation Hooks User Guide의 [Developing custom Hooks using the CloudFormation CLI](#)

또한 CloudFormation 레지스트리에서 사용할 수 있는 모든 프로비저닝 가능한 AWS 리소스 유형을 AWS Cloud Control API에서 표준 JSON 스키마에 정의된 해당 특성 및 속성과 함께 사용할 수 있습니다. 자세한 내용은 [Cloud Control API 사용 설명서](#)를 참조하세요. Cloud Control API를 사용하여 AWS 리소스에 대한 생성, 읽기, 업데이트, 삭제, 나열(CRUDL) 작업을 수행하는 경우 자체 AWS 계정 내 AWS 리소스에서만 수행할 수 있습니다.

CloudFormation 레지스트리 개념

이 주제에서는 CloudFormation 레지스트리를 알아보고 이를 사용해 시작하는 데 도움이 되는 주요 개념을 설명합니다.

확장 프로그램 유형

CloudFormation 레지스트리는 다음과 같은 확장 프로그램 유형을 제공합니다.

후크

후크는 CloudFormation에서 생성, 업데이트 또는 삭제하기 전에 스택이나 특정 리소스를 검사하는 검증 검사입니다. 또한 변경 세트 생성 작업 중 후크를 간접 호출할 수도 있습니다. 특정 요구 사항에 대해 리소스 구성을 검증하여 조직의 표준과 모범 사례를 시행하는 메커니즘을 제공합니다. 후크가 후크 로직을 준수하지 않는 구성을 탐지하면 경고를 발행하거나 프로비저닝 프로세스에 실패하여 규정 미준수 리소스를 배포하지 않도록 할 수 있습니다. 자세한 내용은 [CloudFormation 후크 사용 설명서](#)를 참조하세요.

CloudFormation 콘솔을 사용하여 Hooks를 구성하는 방법에 대한 설명서는 CloudFormation Hooks 사용 설명서에서 다음 섹션을 참조하세요.

- [Hooks로서의 AWS Control Tower 사전 예방적 제어](#)
- [Guard Hooks](#)
- [Lambda Hooks](#)

모듈

모듈은 여러 CloudFormation 스택 템플릿에 포함할 수 있는 재사용 가능한 리소스 구성으로, 이 기능은 복잡하거나 자주 사용되는 리소스 구성을 재사용 가능한 구성 요소로 캡슐화하여 CloudFormation 템플릿의 생성 및 유지 관리를 간소화합니다. 이를 통해 조직의 인프라 배포 전반에 걸쳐 일관성과 표준화가 촉진됩니다.

Resource types

리소스 유형을 사용하면 CloudFormation에서 기본적으로 지원하지 않는 서드 파티 리소스나 사용자 지정 리소스를 모델링하고 자동화할 수 있습니다. 리소스 유형을 개발하면 CloudFormation의 기능을 확장하여 다양한 서드 파티 서비스의 리소스를 프로비저닝하고 관리할 수 있습니다.

퍼블릭 익스텐션

공개 확장 프로그램은 모든 AWS 계정에서 사용할 수 있도록 레지스트리에 공개적으로 게시되는 CloudFormation 확장 프로그램입니다. 다음이 포함됩니다.

- **AWS 확장 프로그램** - AWS에서 게시한 확장 프로그램은 항상 공개 프로그램이고 기본적으로 활성화되므로 계정에서 사용하기 전에 조치를 취할 필요가 없습니다. AWS에서는 이러한 확장 프로그램의 버전 관리를 제어하므로 항상 사용 가능한 최신 버전을 사용합니다.
- **서드 파티 확장 프로그램** - 이 확장 프로그램은 AWS 이외의 게시자가 일반적으로 사용할 수 있습니다. 서드 파티 공개 확장 프로그램을 사용하려면 먼저 계정과 리전에서 활성화해야 합니다.

활성화된 확장 프로그램

특정 AWS 계정의 CloudFormation 레지스트리에는 다음과 같이 세 가지 유형의 활성화된 확장 프로그램이 포함됩니다.

- **AWS 확장 프로그램** - 모든 AWS 공개 확장 프로그램은 자동으로 활성화됩니다.
- **활성화된 서드 파티** - 사용자의 계정과 리전에 대해 명시적으로 활성화한 서드 파티 공개 확장 프로그램의 로컬 사본입니다. 서드 파티 공개 확장 프로그램을 활성화하면 CloudFormation은 계정 레지스트리에서 로컬 사본을 생성합니다. 자세한 내용은 [CloudFormation 레지스트리에서 서드 파티 공개 확장 프로그램 사용](#) 섹션을 참조하세요.
- **비공개 등록** - 공개 CloudFormation 레지스트리에 나열되지 않은 비공개 확장 프로그램입니다. 이는 사용자가 직접 생성한 확장 프로그램이거나 조직이나 다른 서드 파티에서 사용자와 공유하는 확장 프로그램일 수 있습니다. 이러한 프라이빗 익스텐션을 계정에서 사용하려면 먼저 등록해야 합니다. 확장 프로그램을 등록하면 계정의 CloudFormation 레지스트리에 사본이 업로드되고 활성화됩니다. 자세한 내용은 [사용자와 공유된 서드 파티 비공개 확장 프로그램 사용](#) 섹션을 참조하세요.

계정의 서드 파티 게시자에서 비공개로 등록된 확장 프로그램 및 활성화된 공개 확장 프로그램을 사용하는 것은 샌드박스 환경에서 사용하는 것과 같습니다. 확장 프로그램은 버전 제어로 관리되므로 프로

비저닝 동작은 특정 버전과 연결됩니다. 따라서 이러한 확장 프로그램은 공개 확장 프로그램과 동일한 방식으로 작동하여 동일한 버전별 규칙을 준수합니다.

Note

비공개 등록 확장 프로그램 및 활성화된 서드 파티 공개 확장 프로그램은 생성, 읽기, 업데이트, 나열 및 삭제 작업 중에 실행되는 이벤트 핸들러를 구현할 수 있습니다. CloudFormation 스택에서 이러한 확장 프로그램을 사용하면 생성된 리소스에 대한 요금 외에도 계정에 청구되는 요금이 발생할 수 있습니다. 자세한 내용은 [CloudFormation 요금](#)을 참조하십시오.

CloudFormation 레지스트리에서 사용 가능하고 활성화된 확장 프로그램 보기

CloudFormation 레지스트리에서 사용 가능하고 활성화된 확장 프로그램을 보려면 AWS Management Console 또는 AWS CLI를 사용합니다.

사용 가능하고 활성화된 확장 프로그램을 보려면 다음을 수행하세요(콘솔).

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 탐색 창의 레지스트리에서 보려는 확장 프로그램 범주를 선택합니다.
 - Public extensions(퍼블릭 익스텐션)는 계정에서 사용할 수 있는 퍼블릭 익스텐션을 표시합니다.
 1. 필터, 확장 프로그램 유형에서 리소스 유형, 모듈 또는 후크를 선택합니다.
 2. 필터, 게시자에서 AWS 또는 서드 파티를 선택합니다.
 - Activated extensions(활성화된 익스텐션)는 계정에서 활성화된 퍼블릭 및 프라이빗 익스텐션을 표시합니다.
 1. Resource types(리소스 유형), Modules(모듈) 또는 Hooks(후크) 중 원하는 익스텐션 유형을 선택합니다.
 2. 필터 드롭다운 메뉴를 사용하여 보려는 확장 프로그램을 추가로 선택합니다.
 - AWS - AWS에서 게시한 익스텐션을 나열합니다. AWS에서 게시하는 익스텐션은 기본적으로 활성화되어 있습니다.
 - Third-party(타사) - 이 계정에서 활성화한 AWS 이외의 게시자의 모든 퍼블릭 익스텐션을 나열합니다.

- Registered(등록) – 이 계정에서 활성화한 모든 프라이빗 익스텐션을 나열합니다.
 - Publisher(게시자)는 이 계정을 사용하여 게시한 모든 퍼블릭 익스텐션을 표시합니다.
4. 확장 프로그램 이름을 검색하거나 선택하여 확장 프로그램 세부 정보를 봅니다.

사용 가능하고 활성화된 확장 프로그램을 보려면 다음을 수행하세요(AWS CLI).

[list-types](#) 명령을 사용합니다.

CloudFormation 레지스트리에서 서드 파티 공개 확장 프로그램 사용

템플릿에서 서드 파티 공개 확장 프로그램을 사용하려면 먼저 해당 확장 프로그램을 사용하려는 계정과 리전에 대해 확장 프로그램을 활성화해야 합니다. 확장 프로그램이 활성화된 계정과 리전의 스택 작업에서 해당 확장 프로그램을 사용할 수 있습니다.

서드 파티 공개 확장 프로그램을 활성화하면 CloudFormation은 활성화된 확장 프로그램에 대한 항목을 계정의 확장 프로그램 레지스트리에 비공개 확장 프로그램으로 생성합니다. 이를 통해 확장 프로그램에 포함된 모든 구성 속성을 설정할 수 있습니다. 구성 속성은 지정된 AWS 계정과 리전에 대해 확장 프로그램이 구성되는 방법을 정의합니다.

구성 속성을 설정하는 것 외에도 다음과 같은 방법으로 확장 프로그램을 사용자 지정할 수 있습니다.

- 익스텐션에 대한 로깅을 구성하는 것 외에도 CloudFormation이 익스텐션을 활성화하는 데 사용하는 실행 역할을 지정합니다.
- 새 마이너 버전이나 패치 버전을 사용할 수 있게 되면 익스텐션을 자동으로 업데이트할지 여부를 지정합니다.
- 서드 파티 공개 확장 프로그램 이름 대신 사용할 별칭을 지정합니다. 이렇게 하면 타사 익스텐션 간의 이름 충돌을 방지할 수 있습니다.

주제

- [공개 확장 프로그램 액세스를 위해 IAM 권한 및 신뢰 정책으로 실행 역할 구성](#)
- [자동으로 새로운 버전의 확장 프로그램 사용](#)
- [별칭을 사용하여 확장 프로그램 참조](#)
- [공개 확장 프로그램 작업에 일반적으로 사용되는 AWS CLI 명령](#)
- [계정에서 서드 파티 공개 확장 프로그램 활성화](#)

- [계정에서 공개 서드 파티 확장 프로그램 업데이트](#)
- [계정에서 서드 파티 공개 확장 프로그램 비활성화](#)

공개 확장 프로그램 액세스를 위해 IAM 권한 및 신뢰 정책으로 실행 역할 구성

CloudFormation 레지스트리에서 공개 확장 프로그램을 활성화하면 CloudFormation에 AWS 계정 및 리전에서 해당 확장 프로그램을 간접적으로 호출하는 데 필요한 권한을 부여하는 실행 역할을 제공할 수 있습니다.

실행 역할에 필요한 권한은 확장 프로그램 스키마의 핸들러 섹션에 정의되어 있습니다. 확장 프로그램에 필요한 특정 권한을 부여하는 IAM 정책을 생성하고 실행 역할에 연결해야 합니다.

권한 정책 외에도 실행 역할에 CloudFormation이 해당 역할을 수임할 수 있도록 허용하는 신뢰 정책도 있어야 합니다. IAM 사용 설명서의 [사용자 지정 트러스트 정책을 사용하여 역할 생성](#)에 설명된 지침에 따라 사용자 지정 신뢰 정책으로 역할을 생성합니다.

신뢰 관계

다음은 사용할 수 있는 신뢰 정책의 예입니다.

Condition 필드와 함께 하나 이상의 글로벌 조건 컨텍스트 키를 사용하여 서비스 간 혼동된 대리인 방지를 위해 권한 범위를 선택적으로 제한할 수 있습니다. 자세한 내용은 [교차 서비스 혼동된 대리인 방지](#) 섹션을 참조하세요.

- `aws:SourceAccount` 값을 계정 ID로 설정합니다.
- `aws:SourceArn` 값을 확장 프로그램의 ARN으로 설정합니다.

신뢰 정책 예제 1

다음은 리소스 유형 확장 프로그램에 대한 IAM 역할 신뢰 정책의 예입니다.

JSON

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

```

    "Principal": {
      "Service": "resources.cloudformation.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      },
      "ArnLike": {
        "aws:SourceArn": "arn:aws:cloudformation:us-
west-2:123456789012:type/resource/Organization-Service-Resource"
      }
    }
  }
]
}

```

신뢰 정책 예제 2

다음은 후크 확장 프로그램에 대한 IAM 역할 신뢰 정책의 예입니다.

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "resources.cloudformation.amazonaws.com",
          "hooks.cloudformation.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:cloudformation:us-
west-2:123456789012:type/hook/Organization-Service-Hook"
        }
      }
    }
  ]
}

```



```

    }
  }
}
]
}
```

자동으로 새로운 버전의 확장 프로그램 사용

익스텐션을 활성화할 때 최신 마이너 버전을 사용하도록 익스텐션 유형을 지정할 수도 있습니다. 게시자가 활성화된 익스텐션의 새 버전을 출시할 때마다 익스텐션 유형이 마이너 버전을 업데이트합니다.

예를 들어, 다음에 해당 익스텐션이 포함된 템플릿을 사용하여 스택 생성 또는 업데이트와 같은 스택 작업을 수행할 때 CloudFormation은 새로운 마이너 버전을 사용합니다.

새 익스텐션 버전으로 자동 또는 수동 업데이트해도 스택에 이미 프로비저닝된 익스텐션 인스턴스에는 영향이 없습니다.

CloudFormation은 익스텐션의 메이저 버전 업데이트를 잠재적으로 주요 변경 사항을 포함하는 것으로 취급하므로 새로운 메이저 버전의 익스텐션으로 수동 업데이트해야 합니다.

AWS에 의해 게시된 확장 프로그램은 사용 가능한 모든 계정과 리전에 대해 기본적으로 활성화되며 항상 각 AWS 리전에서 사용 가능한 최신 버전을 사용합니다.

Important

확장 프로그램이 계정의 최신 버전으로 업데이트되는지 여부와 시기를 제어하므로 동일한 확장 프로그램의 다른 버전이 다른 계정 및 리전에 배포될 수 있습니다.

이로 인해 해당 계정과 리전에서 해당 확장 프로그램이 포함된 동일한 템플릿을 사용할 때 예기치 않은 결과가 발생할 수 있습니다.

별칭을 사용하여 확장 프로그램 참조

지정된 AWS 계정과 리전에서 지정된 이름으로 여러 확장 프로그램을 활성화할 수 없습니다. 다른 게시자가 동일한 익스텐션 이름으로 퍼블릭 익스텐션을 제공할 수 있기 때문에 CloudFormation을 사용하면 활성화하는 모든 타사 퍼블릭 익스텐션에 대한 별칭을 지정할 수 있습니다.

확장 프로그램에 대한 별칭을 지정하면 CloudFormation은 별칭을 계정과 리전 내에서 확장 프로그램 유형 이름으로 처리합니다. 템플릿, API 호출 및 CloudFormation 콘솔에서 익스텐션을 참조하려면 별칭을 사용해야 합니다.

타사 별칭은 지정된 계정과 리전 내에서 고유해야 합니다. 서로 다른 유형 이름 별칭을 사용하여 동일한 계정과 리전에서 동일한 퍼블릭 리소스를 여러 번 활성화할 수 있습니다.

Important

확장 프로그램 별칭은 지정된 계정과 리전에서만 고유해야 하지만 사용자가 계정과 리전 전반에 걸쳐 다른 서드 파티 공개 확장 프로그램에 동일한 별칭을 할당하지 않는 것이 좋습니다. 이렇게 하면 여러 계정 또는 리전에서 확장 프로그램 별칭이 포함된 템플릿을 사용할 때 예상치 못한 결과가 발생할 수 있습니다.

공개 확장 프로그램 작업에 일반적으로 사용되는 AWS CLI 명령

공개 확장 프로그램 작업에 일반적으로 사용되는 명령은 다음과 같습니다.

- [activate-type](#) - 계정의 공개 서드 파티 모듈 또는 리소스 유형을 활성화합니다.
- [set-type-configuration](#) - 계정의 익스텐션에 대한 구성 데이터를 지정하고 Hooks를 비활성화 및 활성화합니다.
- [list-types](#) - 계정의 확장 프로그램을 나열합니다.
- [describe-type](#) - 현재 구성 데이터를 포함하여 특정 확장 프로그램 또는 특정 확장 프로그램 버전에 대한 자세한 정보를 반환합니다.
- [set-type-default-version](#) - 확장 프로그램의 기본 버전을 지정합니다.
- [deactivate-type](#) - 계정의 이전에 활성화된 공개 서드 파티 모듈 또는 리소스 유형을 비활성화합니다.

계정에서 서드 파티 공개 확장 프로그램 활성화

다음 주제에서는 계정에서 서드 파티 공개 확장 프로그램을 활성화하는 방법을 보여줍니다. 이를 통해 확장 프로그램이 활성화된 계정과 리전에서 해당 확장 프로그램을 사용할 수 있습니다.

Note

계속하기 전에 이 확장 프로그램에 사용할 [IAM 역할](#)을 생성했는지 확인합니다.

주제

- [공개 확장 프로그램 활성화\(콘솔\)](#)
- [공개 확장 프로그램 활성화\(AWS CLI\)](#)

공개 확장 프로그램 활성화(콘솔)

이 섹션의 다음 단계를 따라 콘솔을 사용합니다.

- 서드 파티 공개 확장 프로그램 활성화
- 계정에 대한 추가 확장 프로그램 구성 데이터 지정

계정에서 사용할 공개 확장 프로그램을 활성화하려면 다음을 수행하세요.

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 탐색 창의 레지스트리에서 공개 확장 프로그램을 선택하세요.
4. 필터를 사용하여 확장 프로그램 유형을 선택하고 서드 파티를 선택합니다. AWS에서 게시하는 익스텐션은 기본적으로 활성화되어 있습니다.
5. 확장 프로그램을 선택한 다음 활성화를 선택합니다.

여러 버전의 확장 프로그램을 사용할 수 있는 경우 버전 메뉴를 사용하여 활성화하려는 확장 프로그램 버전을 선택할 수 있습니다. 기본값은 가장 최신 버전입니다.

6. 확장 프로그램 이름의 경우 기본값 사용을 선택한 상태로 유지하거나 기본값 재정의의 선택한 다음 이 확장 프로그램에서 사용할 확장 프로그램 유형 별칭을 입력할 수 있습니다. 별칭은 익스텐션 유형에 대해 권장되는 형식을 따라야 합니다. 자세한 내용은 [별칭을 사용하여 확장 프로그램 참조](#) 섹션을 참조하세요.
7. 활성화하려는 확장 프로그램이 후크 또는 리소스 유형인 경우 실행 역할 ARN에 확장 프로그램을 간접 호출할 때 수임할 CloudFormation의 IAM 역할을 지정하세요. 자세한 내용은 [공개 확장 프로그램 액세스를 위해 IAM 권한 및 신뢰 정책으로 실행 역할 구성](#) 섹션을 참조하세요.
8. 원하는 경우 로깅 구성에서 확장 프로그램에 대한 로깅 구성 정보를 지정하세요. 예제:

```
{
  "logRoleArn": "arn:aws:iam::account:role/rolename",
  "logGroupName": "log-group-name"
}
```

로깅 구성 정보는 필수는 아니지만 디버깅 목적으로 권장됩니다. 후크와 함께 로깅 구성을 사용하면 로그 역할이 로그 그룹에 로그를 쓸 수 있도록 지정된 실행 역할과 동일한 신뢰 정책을 추가합니다.

logRoleArn 및 logGroupName 키 이름은 대소문자를 구분하지 않습니다.

9. 버전 관리, 자동 업데이트에서 업데이트를 받는 방법을 선택하세요.

- On(켜기) - 최신 마이너 버전으로 자동 업데이트됩니다. 메이저 버전은 수동으로 업데이트됩니다.
- Off(끄기) - 최신 마이너 버전으로 자동 업데이트되지 않습니다. 모든 버전이 수동으로 업데이트됩니다.

자세한 내용은 [자동으로 새로운 버전의 확장 프로그램 사용](#) 섹션을 참조하세요.

확장 프로그램에 추가 구성이 필요한 경우 지금 또는 확장 프로그램이 활성화된 후 구성 데이터를 지정할 수 있는 옵션이 있습니다.

Important

활성화하는 확장 프로그램이 후크인 경우 이 단계가 필요합니다. HookInvocationStatus 속성에 ENABLED를 지정해야 합니다. 이 작업은 후크의 스키마 properties 섹션에 정의된 후크의 속성을 활성화합니다. 자세한 내용은 CloudFormation Hooks User Guide의 [Hook configuration schema syntax reference](#)를 참조하세요.

구성 데이터를 지정하려면 다음을 수행하세요.

1. 구성에서 지금 구성을 선택한 다음 확장 프로그램 활성화를 선택합니다.

Configure extension(익스텐션 구성) 페이지가 표시됩니다. 익스텐션에 대한 현재 구성 스키마를 보려면 View configuration schema(구성 스키마 보기)가 활성화되어 있는지 확인합니다.

2. 구성 JSON 텍스트 상자에 이 확장 프로그램에 대해 지정하려는 구성 데이터를 나타내는 JSON 문자열을 입력합니다. 지정하는 JSON은 확장 프로그램의 구성 정의 스키마에 대해 검증되어야 합니다.

3. 확장 프로그램 구성을 선택합니다.

활성화 후 확장 프로그램을 구성하려는 경우 이 단계를 건너뛰고 나중에 구성 데이터를 제공할 수 있습니다.

1. 구성에서 나중에 구성을 선택한 다음 익스텐션 활성화를 선택합니다.

2. 확장 프로그램이 활성화되면 활성화된 확장 프로그램 페이지에서 확장 프로그램으로 이동하고 구성 데이터를 제공하여 확장 프로그램을 구성할 수 있습니다.

공개 확장 프로그램 활성화(AWS CLI)

이 섹션의 다음 단계를 따라 AWS CLI를 사용합니다.

- 서드 파티 공개 확장 프로그램 활성화
- 계정에 대한 추가 확장 프로그램 구성 데이터 지정

퍼블릭 후크 활성화

계정에서 후크를 활성화하면 후크가 AWS 계정에서 정의된 권한을 사용하도록 승인하는 것입니다. CloudFormation은 사용자의 권한을 후크에 전달하기 전에 필요하지 않은 권한을 제거합니다. CloudFormation은 고객 또는 후크 사용자에게 계정에서 후크를 활성화하기 전에 후크 권한을 검토하고 후크에 허용되는 권한을 인지할 것을 권장합니다.

계정에서 사용할 퍼블릭 후크를 활성화하려면 다음을 수행하세요(AWS CLI).

1. 후크의 ARN을 가져와 저장합니다. AWS Management Console 또는 AWS CLI를 사용하여 후크의 ARN을 가져올 수 있습니다. 자세한 내용은 [CloudFormation 레지스트리에서 사용 가능하고 활성화된 확장 프로그램 보기](#)를 참조하세요.

```
export HOOK_TYPE_ARN="arn:aws:cloudformation:us-west-2:123456789012:type/hook/  
Organization-Service-Hook/"
```

2. [set-type-configuration](#) 명령을 사용하여 구성 데이터를 지정합니다. --configuration에 대해 전달하는 JSON은 후크의 구성 스키마에 대해 검증되어야 합니다. 모든 스택 작업에 대해 후크를 활성화하려면 HookConfiguration 섹션에서 HookInvocationStatus 속성을 ENABLED로 설정해야 합니다.

```
aws cloudformation set-type-configuration \  
  --configuration '{"CloudFormationConfiguration":{"HookConfiguration":  
  {"HookInvocationStatus": "ENABLED", "FailureMode": "FAIL", "Properties":{}}}}' \  
  --type-arn $HOOK_TYPE_ARN --region us-west-2
```

HookConfiguration 구성 옵션에 대한 자세한 내용은 CloudFormation Hooks User Guide의 [Hook configuration schema syntax reference](#)를 참조하세요.

퍼블릭 모듈 및 리소스 유형 활성화

계정에서 사용할 공개 확장 프로그램을 활성화하려면 다음을 수행하세요(AWS CLI).

- [activate-type](#) 명령을 사용하여 확장 프로그램을 활성화하고 확장 프로그램의 새 마이너 버전이 게시될 때마다 확장 프로그램을 자동 업데이트할지 여부를 지정하세요.

아래 예제에서는 이 계정에 대해 활성화할 퍼블릭 익스텐션의 퍼블릭 Amazon 리소스 이름(ARN)을 지정합니다. 또한 새 마이너 버전이 게시될 때마다 CloudFormation이 익스텐션을 업데이트하도록 지정합니다.

```
aws cloudformation activate-type \  
  --public-type-arn public_extension_ARN \  
  --execution-role-arn arn:aws:iam::123456789012:role/my-execution-role \  
  --auto-update true --region us-west-2
```

이 명령은 활성화된 확장 프로그램의 ARN을 반환합니다.

```
{  
  "Arn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/My-  
Resource-Example"  
}
```

공개 확장 프로그램의 버전 업데이트(AWS CLI)

[activate-type](#)을 사용하여 익스텐션을 다시 활성화합니다.

--version-bump 옵션을 사용하여 확장 프로그램을 최신 MAJOR 버전으로 업데이트할지, 최신 MINOR 버전으로 업데이트할지 지정합니다.

```
aws cloudformation activate-type --type RESOURCE \  
  --type-name Example::Test::1234567890abcdef0 \  
  --type-name-alias Example::Test::Alias \  
  --version-bump MAJOR --region us-west-2
```

계정에서 공개 서드 파티 확장 프로그램 업데이트

서드 파티 공개 확장 프로그램을 활성화한 후 계정에서 대부분의 확장 프로그램 세부 정보를 업데이트할 수 있습니다.

계정에서 공개 확장 프로그램을 업데이트하는 방법(콘솔)

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 탐색 창의 레지스트리에서 활성화된 프로그램을 선택하세요.
4. 업데이트하려는 확장 프로그램을 찾아 선택하세요. 자세한 내용은 [CloudFormation 레지스트리에서 사용 가능하고 활성화된 확장 프로그램 보기](#) 섹션을 참조하세요.
5. 작업 메뉴에서 편집을 선택한 다음 적절한 편집 옵션을 선택합니다.
 - 구성 스키마를 업데이트하려면 [계정의 확장 프로그램에 대한 구성 데이터 편집](#) 섹션을 참조하세요.
 - 자동 업데이트 활성화 또는 비활성화
 1. 자동 업데이트 편집을 선택합니다.
 2. 활성 또는 비활성을 선택한 다음 저장을 선택합니다. 자세한 내용은 [자동으로 새로운 버전의 확장 프로그램 사용](#) 섹션을 참조하세요.
 - 실행 역할 업데이트
 1. 실행 역할 편집을 선택합니다.
 2. 이 확장 프로그램을 간접적으로 호출할 때 CloudFormation에서 사용할 IAM 역할의 ARN을 지정한 다음 저장을 선택합니다. 자세한 내용은 [공개 확장 프로그램 액세스를 위해 IAM 권한 및 신뢰 정책으로 실행 역할 구성](#) 섹션을 참조하세요.
 - 로깅 구성 업데이트
 1. 로깅 구성 편집을 선택합니다.
 2. 로깅 구성 JSON을 편집한 다음 저장을 선택합니다.

계정에서 서드 파티 공개 확장 프로그램 비활성화

활성화된 서드 파티 공개 확장 프로그램이 더 이상 필요하지 않은 경우 다음 절차에 따라 계정에서 해당 확장 프로그램을 비활성화하세요.

주제

- [계정에서 공개 확장 프로그램 비활성화\(콘솔\)](#)
- [계정에서 공개 확장 프로그램 비활성화\(AWS CLI\)](#)
- [계정에서 후크 비활성화\(AWS CLI\)](#)

계정에서 공개 확장 프로그램 비활성화(콘솔)

계정에서 공개 확장 프로그램을 비활성화하려면 다음을 수행하세요.

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 탐색 창의 레지스트리에서 활성화된 프로그램을 선택하세요.
4. 비활성화하려는 확장 프로그램을 찾아 선택하세요. 자세한 내용은 [CloudFormation 레지스트리에서 사용 가능하고 활성화된 확장 프로그램 보기](#) 섹션을 참조하세요.
5. 작업 메뉴에서 비활성화를 선택합니다.
6. 비활성화를 선택합니다.

계정에서 공개 확장 프로그램 비활성화(AWS CLI)

다음 [deactivate-type](#) 명령을 사용합니다.

```
aws cloudformation deactivate-type --type MODULE \  
  --type-name Example::Test::Type::MODULE \  
  --region us-west-2
```

계정에서 후크 비활성화(AWS CLI)

후크를 비활성화하면 후크를 제거하지 않고는 AWS 계정에서 후크가 실행되지 않습니다.

[set-type-configuration](#) 명령을 사용하고 HookInvocationStatus를 DISABLED로 지정하여 후크를 비활성화합니다.

다음 예제에서는 비활성화 중인 후크의 AWS 리전과 Amazon 리소스 이름(ARN)을 지정합니다.

```
aws cloudformation set-type-configuration \  
  --configuration '{"CloudFormationConfiguration":{"HookConfiguration":  
{"HookInvocationStatus": "DISABLED", "FailureMode": "FAIL", "Properties":{}}}}' \  
  --type-arn "arn:aws:cloudformation:us-west-2:123456789012:type/hook/MyTestHook" --  
  region us-west-2
```

자세한 내용은 CloudFormation Hooks User Guide의 [Disable and enable CloudFormation Hooks](#)를 참조하세요.

사용자와 공유된 서드 파티 비공개 확장 프로그램 사용

사용자와 공유된 서드 파티 비공개 확장 프로그램을 사용하려면 먼저 해당 확장 프로그램을 사용하려는 계정과 리전에서 CloudFormation에 등록해야 합니다. 익스텐션을 등록하면 계정의 CloudFormation 레지스트리에 익스텐션의 사본이 업로드되고 활성화됩니다. 비공개 확장 프로그램을 등록하고 나면 해당 AWS 계정과 리전의 CloudFormation 레지스트리에 표시되고, 스택 템플릿에서 사용할 수 있게 됩니다.

주제

- [서드 파티 비공개 확장 프로그램 등록을 위한 IAM 권한](#)
- [비공개 확장 프로그램 작업에 일반적으로 사용되는 AWS CLI 명령](#)
- [계정에 타사 프라이빗 익스텐션 등록](#)
- [계정에서 타사 프라이빗 확장 제거](#)

서드 파티 비공개 확장 프로그램 등록을 위한 IAM 권한

비공개 확장 프로그램 등록의 일부로 확장 프로그램 프로젝트 패키지가 포함된 Amazon S3 버킷을 지정할 수 있습니다. 이 패키지에는 등록하려는 익스텐션에 필요한 모든 소스 파일이 들어 있습니다. 익스텐션을 등록하는 사용자는 해당 Amazon S3 버킷의 프로젝트 패키지에 액세스할 수 있어야 합니다. 즉, 사용자에게 확장 프로그램 패키지에 대한 [GetObject](#) 권한이 있어야 합니다.

이는 AWS CLI의 [register-type](#) 명령을 사용하거나 또는 CloudFormation CLI의 [submit](#) 명령을 사용하는지에 관계없이 적용됩니다.

자세한 내용은 서비스 승인 참조에서 [Amazon S3에 사용되는 작업, 리소스 및 조건 키](#)를 참조하세요.

비공개 확장 프로그램 작업에 일반적으로 사용되는 AWS CLI 명령

비공개 확장 프로그램 작업에 일반적으로 사용되는 명령은 다음과 같습니다.

- [register-type](#) - 계정에 비공개 확장 프로그램을 등록합니다.
- [describe-type-registration](#) - 등록 요청의 현재 상태를 반환합니다.
- [list-types](#) - 계정의 확장 프로그램을 나열합니다.
- [describe-type](#) - 현재 구성 데이터를 포함하여 특정 확장 프로그램 또는 특정 확장 프로그램 버전에 대한 자세한 정보를 반환합니다.

- [set-type-configuration](#) - 계정의 익스텐션에 대한 구성 데이터를 지정하고 Hooks를 비활성화 및 활성화합니다.
- [set-type-default-version](#) - 확장 프로그램의 기본 버전을 지정합니다.
- [deregister-type](#) - 계정에서 프라이빗 익스텐션이나 익스텐션 버전을 제거합니다.

계정에 타사 프라이빗 익스텐션 등록

이 주제에서는 계정에서 사용할 수 있도록 공유된 타사 프라이빗 익스텐션을 등록하는 단계를 다룹니다.

Note

계속하기 전에 비공개 확장 프로그램을 등록하는 데 필요한 [IAM 권한](#)이 있는지 확인하세요.

사용자와 공유된 프라이빗 익스텐션을 등록하려면 다음을 수행하세요(AWS CLI).

1. 계정에 등록하려는 프라이빗 익스텐션에 대한 프로젝트 패키지가 포함되어 있는 Amazon S3 버킷의 위치를 찾습니다.
2. [register-type](#) 명령을 사용하여 계정에 프라이빗 익스텐션을 등록합니다.

예를 들어 다음 명령은 지정된 AWS 계정에 `My::Resource::Example` 리소스 유형을 등록합니다.

```
aws cloudformation register-type --type RESOURCE \  
  --type-name My::Resource::Example \  
  --schema-handler-package [s3 object path] --region us-west-2
```

RegisterType은 비동기식 작업으로, 등록 요청의 진행 상황을 추적하는 데 사용할 수 있는 등록 토큰을 반환합니다.

```
{  
  "RegistrationToken": "f5525280-104e-4d35-bef5-8f1fexample"  
}
```

익스텐션이 기능의 일부로 AWS API를 호출하는 경우 해당 AWS API를 호출하는 데 필요한 권한이 포함된 IAM 실행 역할을 생성하고 계정에서 해당 실행 역할을 프로비저닝해야 합니다. 그

런 다음 `--execution-role-arn` 옵션을 사용하여 이 실행 역할을 지정할 수 있습니다. 이제 CloudFormation은 해당 실행 역할을 수임하여 리소스 유형에 적절한 자격 증명을 제공합니다.

```
--execution-role-arn arn:aws:iam::123456789012:role/MyIAMRole
```

3. (선택 사항) 등록 토큰을 [describe-type-registration](#) 명령과 함께 사용하여 등록 요청의 진행 상황을 추적합니다.

CloudFormation이 등록 요청을 완료하면 요청의 진행 상태를 COMPLETE로 설정합니다.

다음 예제에서는 위의 `describe-type-registration` 명령에서 반환한 등록 토큰을 사용하여 등록 상태 정보를 반환합니다.

```
aws cloudformation describe-type-registration \
  --registration-token f5525280-104e-4d35-bef5-8f1fexample \
  --region us-west-2
```

명령은 다음 출력을 반환합니다.

```
{
  "ProgressStatus": "COMPLETE",
  "TypeArn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/My-Resource-Example",
  "Description": "Deployment is currently in DEPLOY_STAGE of status COMPLETED; ",
  "TypeVersionArn": "arn:aws:cloudformation:us-west-2:123456789012:type/resource/My-Resource-Example/00000001"
}
```

Important

등록하는 익스텐션이 후크인 경우 이 다음 단계가 필요합니다. `HookInvocationStatus` 속성에 `ENABLED`을 지정해야 합니다. 이 작업은 후크의 스키마 `properties` 섹션에 정의된 후크의 속성을 활성화합니다. 자세한 내용은 CloudFormation Hooks User Guide의 [Hook configuration schema syntax reference](#)를 참조하세요.

후크에 대한 구성 데이터를 지정하려면 다음을 수행하세요(AWS CLI).

1. 후크의 ARN을 가져와 저장합니다. AWS Management Console 또는 AWS CLI를 사용하여 후크의 ARN을 가져올 수 있습니다. 자세한 내용은 [CloudFormation 레지스트리에서 사용 가능하고 활성화된 확장 프로그램 보기](#)을 참조하세요.

```
export HOOK_TYPE_ARN="arn:aws:cloudformation:us-west-2:123456789012:type/hook/
Organization-Service-Hook/"
```

2. [set-type-configuration](#) 명령을 사용하여 구성 데이터를 지정합니다. --configuration에 대해 전달하는 JSON은 후크의 구성 스키마에 대해 검증되어야 합니다. 후크를 활성화하려면 HookConfiguration 섹션에서 HookInvocationStatus 속성을 ENABLED로 설정해야 합니다.

```
aws cloudformation set-type-configuration \
  --configuration '{"CloudFormationConfiguration":{"HookConfiguration":
{"HookInvocationStatus": "ENABLED", "FailureMode": "FAIL", "Properties":{}}}}' \
  --type-arn $HOOK_TYPE_ARN --region us-west-2
```

자세한 내용은 CloudFormation Hooks User Guide의 [Hook configuration schema syntax reference](#)를 참조하세요.

계정에서 타사 프라이빗 확장 제거

타사 프라이빗 익스텐션이나 익스텐션 버전을 제거하려면 [deregister-type](#) 명령을 사용하세요.

특정 확장 프로그램 버전 또는 전체 확장 프로그램을 등록 취소할 수 있습니다. 확장 프로그램을 등록 취소하려면 해당 확장 프로그램의 등록된 모든 버전을 개별적으로 등록 취소해야 합니다. 확장 프로그램에 등록된 버전이 하나뿐인 경우 해당 버전을 등록 취소하면 확장 프로그램 자체가 등록 취소됩니다. 확장 프로그램의 기본 버전은 해당 확장 프로그램의 유일한 등록 버전이 아닌 한 등록 취소할 수 없으며, 이 경우 확장 프로그램 자체도 등록 취소됩니다.

Warning

프라이빗 확장 프로그램의 등록 취소는 취소할 수 없습니다. 이 작업에서는 다음을 수행합니다.

- 모든 CloudFormation 작업에서 확장 프로그램을 사용할 수 없도록 설정합니다.

- 이 확장 프로그램을 사용하는 향후 스택 업데이트에서 오류가 발생합니다(모듈 및 리소스 유형). 나중에 확장 프로그램을 프라이빗으로 다시 등록할 수 있지만 CloudFormation에서 이전 버전을 사용하는 경우 오류가 발생할 수 있습니다.

계속하기 전에 [list-stacks](#) 및 [get-template](#) 명령을 사용하여 이 확장 프로그램을 사용하는 활성 스택이 없는지 확인합니다.

확장 등록 취소 명령 예제

이 섹션에서는 프라이빗 확장을 등록 취소하는 다양한 방법을 보여주는 예제를 제공합니다.

유형 이름으로 등록 취소

--type 및 --type-name 옵션과 함께 [deregister-type](#) 명령을 사용하여 확장을 등록 취소합니다.

```
aws cloudformation deregister-type \  
  --type MODULE \  
  --type-name My::S3::SampleBucket::MODULE
```

유형 이름 및 버전으로 등록 취소

확장 프로그램의 특정 버전을 등록 취소하려면 명령에서 --version-id 옵션을 지정하세요.

```
aws cloudformation deregister-type \  
  --type MODULE \  
  --type-name My::S3::SampleBucket::MODULE \  
  --version-id 00000001
```

Tip

먼저 다른 버전의 확장을 기본값으로 설정하려면 [set~type~default~version](#) 명령을 사용합니다.

ARN으로 등록 취소

--arn 옵션을 사용하고 확장의 ARN을 지정하여 확장을 등록 취소합니다.

```
aws cloudformation deregister-type \  
  --arn arn:aws:cloudformation:us-east-1:123456789012:stack-set/MyStackSet
```

```
--arn arn:aws:cloudformation:us-west-2:123456789012:type/resource/Organization-Service-Resource
```

계정의 확장 프로그램에 대한 구성 데이터 편집

이 주제에서는 특정 리전 내 계정의 확장 프로그램에 대한 구성 데이터 편집 지침을 제공합니다. 확장 프로그램에는 지정된 계정과 리전에 대한 확장 프로그램의 모든 인스턴스에 적용되는 구성 속성이 포함될 수 있습니다. 이러한 속성은 확장 프로그램 작성자가 확장 프로그램의 구성 정의에 정의합니다. 익스텐션의 구성 정의에 필수 속성이 있는 경우 해당 속성을 지정해야 계정과 리전에서 익스텐션을 사용할 수 있습니다.

확장 프로그램을 개발할 때 구성 정의가 정의되는 방법에 대한 자세한 내용은 다음 설명서를 참조하세요.

- [후크 구성 스키마 구문 참조](#)
- [익스텐션의 계정 수준 구성 정의](#)

주제

- [동적 참조를 사용하는 데 필요한 권한](#)
- [확장 프로그램에 대한 구성 데이터 편집\(콘솔\)](#)
- [확장 프로그램에 대한 구성 데이터 편집\(AWS CLI\)](#)

동적 참조를 사용하는 데 필요한 권한

구성 데이터에 AWS Systems Manager 또는 AWS Secrets Manager에 저장된 값에 대한 동적 참조가 있는 경우 유형을 프로비저닝하는 데 사용되는 모든 역할(예: 스택을 만들거나 업데이트할 때)에 해당 값을 검색할 수 있는 적절한 권한이 있어야 합니다. 구체적으로 설명하면 다음과 같습니다.

- 구성 데이터에 AWS Systems Manager Parameter Store에 저장된 파라미터가 포함된 경우 유형을 프로비저닝하는 데 사용되는 사용자 또는 역할에 [GetParameter](#)를 직접적으로 호출할 수 있는 권한이 있어야 합니다.
- 구성 데이터에 AWS Secrets Manager에 저장된 보안 암호가 포함된 경우 유형을 프로비저닝하는 데 사용되는 사용자 또는 역할에 [GetSecretValue](#)를 직접적으로 호출할 수 있는 권한이 있어야 합니다.

자세한 내용은 [동적 참조를 사용하여 다른 서비스에 저장된 값 가져오기](#) 섹션을 참조하세요.

확장 프로그램에 대한 구성 데이터 편집(콘솔)

이 섹션의 다음 단계를 따라 콘솔을 사용합니다.

- 확장 프로그램에 대한 현재 구성 데이터 보기
- 계정의 확장 프로그램 구성 데이터 업데이트

확장 프로그램에 대한 현재 구성 데이터를 보려면 다음을 수행하세요.

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 탐색 창 의 레지스트리에서 활성화된 프로그램을 선택하세요.
4. 보려는 확장 프로그램을 찾습니다. 자세한 내용은 [CloudFormation 레지스트리에서 사용 가능하고 활성화된 확장 프로그램 보기](#) 섹션을 참조하세요.
5. 익스텐션을 선택하여 익스텐션 세부 정보를 봅니다.
6. 익스텐션 세부 정보 페이지에서 Configuration(구성) 탭을 선택합니다.
7. Configuration schema(구성 스키마) 탭을 확장하여 익스텐션에 대해 정의된 구성 스키마를 봅니다.
8. Configuration(구성) 탭을 클릭하여 이 익스텐션에 대해 설정한 현재 구성을 봅니다.

확장 프로그램의 구성 데이터를 업데이트하려면 다음을 수행하세요.

1. 확장 프로그램 세부 정보 페이지의 구성 탭에서 구성 편집을 선택합니다.

또는 작업에서 편집을 선택한 다음 구성 편집을 선택합니다.

Configure extension(익스텐션 구성) 페이지가 표시됩니다. 익스텐션의 현재 구성 정의 스키마를 보려면 View configuration schema(구성 스키마 보기)가 켜져 있는지 확인합니다.

2. Configuration JSON(구성 JSON) 텍스트 상자에 이 익스텐션에 대해 설정하려는 구성 스키마를 나타내는 JSON 문자열을 입력합니다. Configuration schema(구성 스키마)에 정의된 스키마에 대해 검증해야 합니다.
3. 익스텐션 구성을 선택합니다.

확장 프로그램에 대한 구성 데이터 편집(AWS CLI)

이 섹션의 다음 단계를 따라 AWS CLI를 사용합니다.

- 확장 프로그램에 대한 현재 구성 데이터 보기
- 계정의 확장 프로그램 구성 데이터 업데이트

확장 프로그램에 대한 현재 구성 데이터를 보려면 다음을 수행하세요.

- [describe-type](#) 명령을 사용하여 확장 프로그램에 대한 세부 정보를 반환합니다. 출력의 ConfigurationSchema 요소는 지정된 리전에서 익스텐션의 현재 구성 정의를 포함합니다.

또는 [batch-describe-type-configurations](#) 명령을 사용하여 여러 익스텐션에 대한 구성 데이터를 반환합니다.

확장 프로그램의 구성 데이터를 업데이트하려면 다음을 수행하세요.

- [set-type-configuration](#) 명령을 사용하여 구성 데이터를 지정합니다. --configuration에 대해 전달하는 JSON은 확장 프로그램의 구성 스키마에 대해 검증되어야 합니다.

다음 예에서 set-type-configuration 명령은 --configuration 옵션에 대한 구성 데이터 `{"CredentialKey": "testUserCredential"}`를 지정합니다.

```
aws cloudformation set-type-configuration --type RESOURCE \  
  --type-name My::Resource::Example \  
  --configuration-alias default \  
  --configuration '{"CredentialKey": "testUserCredential"}' \  
  --region us-west-2
```

AWS Config에서 리소스 유형 기록

AWS Config가 프라이빗 리소스 유형을 자동으로 추적하고 해당 리소스의 변경 내용을 구성 항목으로 기록하도록 지정할 수 있습니다. 이렇게 하면 이러한 프라이빗 리소스 유형에 대한 구성 기록을 볼 수 있을 뿐 아니라 AWS Config 규칙 규칙을 작성하여 구성 모범 사례를 확인할 수 있습니다. 후크 확장 프로그램에는 AWS Config가 필요합니다.

AWS Config가 프라이빗 리소스 유형을 자동으로 추적하도록 하려면 다음과 같이 하세요.

- CloudFormation을 통해 리소스를 관리합니다. 여기에는 CloudFormation을 통한 모든 리소스 생성, 업데이트 및 삭제 작업이 포함됩니다.

Note

IAM 역할을 사용하여 스택 작업을 수행하는 경우 해당 IAM 역할에 다음 AWS Config 작업을 호출할 권한이 있어야 합니다.

- [PutResourceConfig](#)
- [DeleteResourceConfig](#)

- 모든 리소스 유형을 기록하도록 AWS Config를 구성합니다. 자세한 내용은 AWS Config 개발자 안내서의 [Recording Configurations for Third-Party Resources using the AWS CLI](#)를 참조하세요.

Note

AWS Config는 필수 및 쓰기 전용으로 정의된 속성을 포함하는 프라이빗 리소스의 기록을 지원하지 않습니다.

설계상, 쓰기 전용으로 정의된 리소스 속성은 AWS Config 구성 항목을 생성하는 데 사용되는 스키마에서 반환되지 않습니다. 따라서 쓰기 전용과 필수로 정의된 속성을 포함하면 필수 속성이 없으므로 구성 항목 생성에 실패합니다. 구성 항목을 생성하는 데 사용할 스키마를 보기 위해 [DescribeType](#) 작업의 schema 속성을 검토할 수 있습니다.

구성 항목에 대한 자세한 내용을 알아보려면 AWS Config 개발자 안내서의 [구성 항목](#)을 참조하세요.

구성 항목에 민감한 속성이 기록되지 않도록 방지

리소스 유형에는 구성 항목의 일부로 기록하지 않으려는 암호, 시크릿 또는 기타 민감한 데이터와 같은 민감한 정보를 고려하는 속성이 포함될 수 있습니다. 구성 항목에 속성이 기록되지 않도록 하려면 리소스 유형 스키마의 `writeOnlyProperties` 목록에 해당 속성을 포함할 수 있습니다. `writeOnlyProperties`로 나열된 리소스 속성은 사용자가 지정할 수 있지만 `read` 또는 `list` 요청에 의해 반환되지 않습니다.

자세한 내용은 CloudFormation CLI User Guide의 [writeOnlyProperties](#) 섹션을 참조하세요.

CodePipeline을 사용한 지속적 전달

지속적 전달은 코드 변경 사항이 자동으로 구축, 테스트 및 준비되어 프로덕션으로 릴리스되는 릴리스 사례입니다. CloudFormation 및 CodePipeline을 사용하면 지속적 전달을 통해 CloudFormation 템플릿을 프로덕션 스택으로 승격하기 전에 해당 템플릿에 대한 변경 사항을 자동으로 구축 및 테스트합니다. 이러한 릴리스 프로세스를 통해 신속하고 안정적으로 AWS 인프라를 변경할 수 있습니다.

예를 들어, 코드 리포지토리에 업데이트된 템플릿을 제출하는 경우 테스트 스택을 자동으로 구축하는 워크플로우를 생성할 수 있습니다. CloudFormation이 테스트 스택을 구축하면 해당 스택을 테스트한 다음 프로덕션 스택으로 변경 사항을 푸시할지 여부를 결정할 수 있습니다. 지속적 전달의 이점에 대한 자세한 내용은 [지속적 전달이란 무엇인가요?](#)를 참조하세요.

CodePipeline을 사용하여 CloudFormation 스택용 파이프라인을 구축하여 지속적 전달 워크플로를 구축합니다. CodePipeline은 CloudFormation에 기본적으로 통합되어 있어 파이프라인 내에서 스택 생성, 업데이트 또는 삭제와 같은 CloudFormation 관련 작업을 지정할 수 있습니다. CodePipeline에 대한 자세한 내용을 알아보려면 [AWS CodePipeline 사용 설명서](#)를 참조하세요.

주제

- [연습: 테스트 및 프로덕션 스택용 파이프라인 빌드](#)
- [CloudFormation 구성 속성 참조](#)
- [CloudFormation 아티팩트](#)
- [CodePipeline 파이프라인에서 파라미터 재정의 함수 사용](#)

연습: 테스트 및 프로덕션 스택용 파이프라인 빌드

CloudFormation 템플릿을 제출하면 CloudFormation에서 이 템플릿을 사용하여 테스트 스택을 자동으로 빌드하는 릴리스 프로세스가 있다고 상상해 보세요. 테스트 스택을 검토한 후, 변경 시 프로덕션 스택이 어떻게 수정될지 미리 보고 나서 구현 여부를 선택할 수 있습니다. 이 워크플로우를 실현하기 위해 CloudFormation을 사용하여 테스트 스택을 빌드한 후, 테스트 스택을 삭제한 다음, 변경 세트를 생성하고 나서 변경 세트를 실행할 수 있습니다. 하지만 각 작업마다 CloudFormation과 수동으로 상호 작용해야 합니다. 이 연습에서는 CloudFormation 스택을 사용하여 연속 전달 워크플로우를 실현하는 데 도움이 되도록 이러한 여러 작업을 자동화하는 CodePipeline 파이프라인을 빌드하겠습니다.

사전 조건

이 연습에서는 CodePipeline과 CloudFormation을 사용해본 적이 있으며 파이프라인과 AWS CloudFormation 템플릿 및 스택의 작동 방식에 대해 잘 알고 있다고 가정하겠습니다. CodePipeline에

대한 자세한 내용을 알아보려면 [AWS CodePipeline 사용 설명서](#)를 참조하세요. 파이프라인을 생성하는 동일한 AWS 리전에 Amazon S3 버킷도 있어야 합니다.

Important

샘플 WordPress 템플릿은 인터넷에 연결해야 하는 EC2 인스턴스를 생성합니다. 인터넷에 대한 트래픽을 허용하는 기본 VPC와 서브넷이 있는지 확인합니다.

연습 개요

이 연습은 스택에서 샘플 WordPress 사이트용 파이프라인을 빌드합니다. 파이프라인은 세 단계로 분리됩니다. 각 단계마다 최소 한 개의 작업이 포함되어야 하며, 이 때의 작업은 파이프라인에서 아티팩트(입력)에 대해 수행되는 작업입니다. 스테이지는 파이프라인에 작업을 구성합니다. CodePipeline이 스테이지에서 새 아티팩트를 처리하기 전에(예를 들어 파이프라인을 재실행하기 위해 새 입력을 제출한 경우) 스테이지의 모든 작업은 완료해야 합니다.

이 연습이 끝나면 다음 워크플로우를 수행하는 파이프라인이 생성됩니다.

1. 파이프라인의 첫 번째 단계는 리포지토리에서 소스 아티팩트(CloudFormation 템플릿과 관련 구성 파일)를 검색합니다.

샘플 WordPress 템플릿을 포함하는 아티팩트를 준비하여 S3 버킷에 업로드합니다.

2. 두 번째 단계에서는 파이프라인이 테스트 스택을 생성하고 승인을 기다립니다.

테스트 스택을 검토한 후 원래 파이프라인을 계속 사용할지 아니면 다른 아티팩트를 생성 및 제출하여 변경할지 선택할 수 있습니다. 승인하고 나면 이 단계에서 테스트 스택이 삭제되고 나서 파이프라인이 다음 단계로 이동합니다.

3. 세 번째 단계에서는 파이프라인이 프로덕션 스택에 대해 변경 세트를 생성하고 나서 승인을 기다립니다.

초기 실행에서는 프로덕션 스택이 없습니다. 변경 세트에는 CloudFormation에서 생성할 모든 리소스가 표시됩니다. 승인하면 이 단계에서 변경 세트가 실행되고 프로덕션 스택이 빌드됩니다.

Note

CloudFormation은 무료 서비스입니다. 하지만 각각에 대해 현재 요금에서 스택에 포함시키는 EC2 인스턴스 같은 AWS 리소스에 대해 요금이 청구됩니다. AWS 요금에 대한 자세한 내용을 알아보려면 <http://aws.amazon.com>의 제품별 세부 정보 페이지를 참조하세요.

1단계: 아티팩트 편집 및 S3 버킷에 업로드

파이프라인을 구축하기 전에 소스 리포지토리와 파일을 설정해야 합니다. CodePipeline에서는 이러한 소스 파일을 파이프라인의 [아티팩트 스토어](#)로 복사하고 나서 이러한 파일을 사용하여 CloudFormation 스택 생성 등의 작업을 파이프라인에서 수행합니다.

Amazon Simple Storage Service(Amazon S3)를 소스 리포지토리로 사용하는 경우 CodePipeline에서는 소스 파일을 S3 버킷으로 업로드하기 전에 이러한 파일을 압축해야 합니다. 압축한 파일은 CloudFormation 템플릿, 템플릿 구성 파일 또는 두 가지 모두를 포함할 수 있는 CodePipeline 아티팩트입니다. Amazon에서는 샘플 WordPress 템플릿 하나와 템플릿 구성 파일 두 개를 포함하는 아티팩트를 제공합니다. 2개의 구성 파일은 WordPress 템플릿의 파라미터 값을 지정합니다. CodePipeline은 WordPress 스택을 생성할 때 이를 사용합니다. 파일 하나에는 테스트 스택용 파라미터 값이 들어 있으며 다른 하나에는 프로덕션 스택용 파라미터 값이 들어 있습니다. 예를 들어 자신의 고유한 기존 EC2 키-페어 이름을 지정하려면 구성 파일을 편집해야 합니다. 아티팩트에 대한 자세한 내용은 [CloudFormation 아티팩트](#) 섹션을 참조하세요.

아티팩트를 빌드한 후에는 S3 버킷에 업로드합니다.

아티팩트를 편집 및 업로드하려면

1. 샘플 아티팩트인 <https://s3.amazonaws.com/cloudformation-examples/user-guide/continuous-deployment/wordpress-single-instance.zip>을 다운로드하여 엽니다.

아티팩트에는 세 가지 파일이 들어 있습니다.

- 샘플 WordPress 템플릿: `wordpress-single-instance.yaml`
- 테스트 스택용 템플릿 구성 파일: `test-stack-configuration.json`
- 프로덕션 스택용 템플릿 구성 파일: `prod-stack-configuration.json`

2. 모든 파일의 압축을 풀고 나서 텍스트 편집기를 사용하여 템플릿 구성 파일을 수정합니다.

구성 파일을 열어서 WordPress 템플릿의 파라미터로 매핑되는 키-값 페어가 들어 있는지 확인합니다. 이 구성 파일은 테스트 및 프로덕션 스택 생성 시 파이프라인에서 사용하는 파라미터 값을 지정합니다.

test-stack-configuration.json 파일에는 테스트 스택용 파라미터 값이 들어 있으며 prod-stack-configuration.json 파일에는 프로덕션 스택용 파라미터 값이 들어 있습니다.

- DBPassword 및 DBRootPassword 키의 값을 WordPress 데이터베이스에 로그인할 때 사용할 수 있는 암호로 변경합니다. WordPress 템플릿에 정의된 대로, 파라미터 값에는 영숫자만 포함되어야 합니다.
 - 파이프라인을 생성할 리전의 기존 EC2 키-페어 이름으로 KeyName 키의 값을 변경합니다.
3. 수정된 구성 파일을 원래 아티팩트(.zip) 파일에 추가하여 중복 파일을 대체합니다.

이제 S3 버킷에 업로드할 수 있는 사용자 지정 아티팩트가 있습니다.

4. [자신의 고유 S3 버킷에 아티팩트를 업로드합니다.](#)

파일 위치를 적어둡니다. 파이프라인을 빌드할 때 이 파일의 위치를 지정합니다.

아티팩트 및 S3 버킷에 대한 중요 정보:

- 파이프라인을 생성할 동일한 AWS 리전에 있는 버킷을 사용합니다.
- CodePipeline에서는 버킷에 대해 [버전 관리가 활성화](#)되어야 합니다.
- 소스 리포지토리로 업로드하기 전에 파일을 압축할 필요가 없는 GitHub 또는 CodeCommit 같은 서비스를 사용할 수도 있습니다.
- 암호 등 중요 정보가 아티팩트에 포함될 수 있습니다. 허용된 사용자만 파일을 볼 수 있도록 액세스를 제한하십시오. 그렇게 해도 CodePipeline에서는 여전히 파일에 액세스할 수 있습니다. 예를 들어, 아티팩트를 S3 버킷에 업로드하면 [S3 버킷 정책 또는 사용자 정책](#)을 사용하여 액세스를 제한합니다.

이제 CodePipeline에서 파이프라인에 아티팩트를 추가할 수 있습니다. 다음 단계에서는 아티팩트의 위치를 지정하고 WordPress 파이프라인을 빌드하겠습니다.

2단계: 파이프라인 스택 생성

WordPress 파이프라인을 생성하려면 샘플 CloudFormation 템플릿을 사용합니다. 파이프라인 빌드 외에도, 이 템플릿은 CodePipeline 및 CloudFormation용 AWS Identity and Access Management(IAM) 서비스 역할, CodePipeline 아티팩트 스토어용 S3 버킷, 그리고 파이프라인에서 알림(예: 검토에 대한

알림)을 보낼 Amazon Simple Notification Service(Amazon SNS) 주제를 설정합니다. 이 샘플 템플릿을 사용하면 이러한 리소스를 단일 CloudFormation 스택에서 쉽게 프로비저닝 및 구성할 수 있습니다.

파이프라인 구성에 대한 자세한 내용은 [파이프라인의 기능](#) 단원을 참조하십시오.

Important

샘플 WordPress 템플릿은 인터넷에 연결해야 하는 EC2 인스턴스를 생성합니다. 기본 VPC와 서브넷에서 인터넷에 대한 트래픽을 허용하는지 확인합니다.

파이프라인 스택을 생성하려면

1. <https://s3.amazonaws.com/cloudformation-examples/user-guide/continuous-deployment/basic-pipeline.yml>에서 샘플 템플릿을 다운로드합니다. 그런 다음 컴퓨터에 저장합니다.
2. <https://console.aws.amazon.com/cloudformation/>에서 CloudFormation 콘솔을 엽니다.
3. CodePipeline 및 CloudFormation을 지원하는 AWS 리전을 선택합니다.

자세한 내용은 AWS 일반 참조의 [CodePipeline endpoints and quotas](#)와 [CloudFormation endpoints and quotas](#)를 참조하세요.

4. 스택 생성을 선택합니다.
5. 템플릿 지정 섹션에서 템플릿 파일 업로드를 선택한 다음, 방금 다운로드한 템플릿인 basic-pipeline.yml을 선택합니다.
6. 다음을 선택합니다.
7. 스택 이름에는 sample-WordPress-pipeline을 입력합니다.
8. 파라미터 섹션에서 다음 파라미터 값을 지정하고 나서 다음을 선택합니다. 스택 파라미터를 설정할 때, WordPress 템플릿 및 관련 구성 파일에 대해 동일한 이름을 유지한 경우, 기본값을 사용할 수 있습니다. 그렇지 않은 경우에는 사용했던 파일 이름을 지정합니다.

PipelineName

WordPress-test-pipeline 같은 파이프라인의 이름입니다.

S3Bucket

아티팩트(.zip 파일)를 저장한 S3 버킷의 이름입니다.

SourceS3Key

아티팩트의 파일 이름입니다. 아티팩트를 폴더에 저장한 경우 folder/subfolder/wordpress-single-instance.zip과 같이 파일 이름에 이를 포함시킵니다.

이메일

CodePipeline에서 파이프라인 알림을 보내는 이메일 주소입니다(예: myemail@example.com).

9. 이 연습에서는 태그를 추가하거나 고급 설정을 지정할 필요가 없으므로 다음을 선택합니다.
10. 스택 이름과 템플릿 URL이 올바른지 확인한 다음 스택 생성을 선택합니다.
11. CloudFormation이 IAM 리소스를 생성할 수 있다는 것을 알고 있음을 확인하려면 확인란을 선택합니다.

AWS CloudFormation에서 스택을 생성하는 데 몇 분 정도 걸릴 수 있습니다. 진행 상황을 모니터링하려면 스택 이벤트를 확인합니다. 자세한 내용은 [스택 진행 상황 모니터링](#) 섹션을 참조하세요.

스택이 생성되고 나면 CodePipeline에서 새 파이프라인을 시작합니다. 상태를 보려면 [CodePipeline 콘솔](#)을 확인합니다. 파이프라인 목록에서 WordPress-test-pipeline을 선택합니다.

파이프라인의 기능

이 섹션에서는 샘플 WordPress 파이프라인 템플릿의 코드 조각을 사용하여 파이프라인의 세 단계를 설명합니다.

1단계: 소스

파이프라인의 첫 번째 단계는 소스 코드의 위치를 지정하는 소스 단계입니다. 개정을 이 위치로 푸시할 때마다 CodePipeline에서 파이프라인을 다시 실행합니다.

소스 코드는 S3 버킷에 있으며 파일 이름으로 식별합니다. 파이프라인 스택을 생성할 때 이러한 값을 입력 파라미터 값으로 지정했습니다. 후속 단계에서 소스 아티팩트 사용을 허용하기 위해 코드 조각에서 OutputArtifacts 속성을 TemplateSource라는 이름으로 지정합니다. 이후 단계에서 이 아티팩트를 사용하려면 TemplateSource를 입력 아티팩트로 지정하면 됩니다.

```
- Name: S3Source
  Actions:
    - Name: TemplateSource
      ActionTypeId:
      Category: Source
```

```

    Owner: AWS
    Provider: S3
    Version: '1'
  Configuration:
    S3Bucket: !Ref 'S3Bucket'
    S3ObjectKey: !Ref 'SourceS3Key'
  OutputArtifacts:
    - Name: TemplateSource

```

2단계: TestStage

TestStage 단계에서 파이프라인은 테스트 스택을 생성한 후, 승인을 기다렸다가 테스트 스택을 삭제합니다.

CreateStack 작업의 경우, 파이프라인은 테스트 구성 파일과 WordPress 템플릿을 사용하여 테스트 스택을 생성합니다. 두 파일은 TemplateSource 입력 아티팩트에 포함되어 있으며, 이 아티팩트는 소스 단계에서 가져옵니다. 이 코드 조각은 REPLACE_ON_FAILURE 작업 모드를 사용합니다. 스택 생성에 실패할 경우 파이프라인이 해당 스택을 대체하므로 파이프라인을 다시 실행하기 전에 스택을 정리하거나 문제를 해결할 필요가 없습니다. 작업 모드는 테스트 스택에 대해 신속히 반복할 때 유용합니다. RoleArn 속성의 경우, 값은 템플릿의 어딘가에 선언되어 있는 AWS CloudFormation 서비스 역할입니다.

ApproveTestStack 작업은 파이프라인을 일시 정지하고, 파이프라인 스택을 생성할 때 지정했던 이메일 주소로 알림을 전송합니다. 파이프라인이 일시 정지된 동안 WordPress 테스트 스택과 관련 리소스를 확인할 수 있습니다. CodePipeline을 사용하여 이 작업을 [승인 또는 거부](#)합니다. CustomData 속성에는 승인 대기 중인 작업에 대한 설명이 포함되어 있으며, 파이프라인은 이 설명을 알림 이메일에 추가합니다.

이 작업이 승인되고 나면 CodePipeline에서 DeleteTestStack 작업으로 이동하여 테스트 WordPress 스택과 관련 리소스를 삭제합니다.

```

- Name: TestStage
  Actions:
    - Name: CreateStack
      ActionTypeId:
        Category: Deploy
        Owner: AWS
        Provider: CloudFormation
        Version: '1'
      InputArtifacts:
        - Name: TemplateSource

```



```

Configuration:
  ActionMode: REPLACE_ON_FAILURE
  RoleArn: !GetAtt [CFNRole, Arn]
  StackName: !Ref TestStackName
  TemplateConfiguration: !Sub "TemplateSource::${TestStackConfig}"
  TemplatePath: !Sub "TemplateSource::${TemplateFileName}"
RunOrder: '1'
- Name: ApproveTestStack
  ActionTypeId:
    Category: Approval
    Owner: AWS
    Provider: Manual
    Version: '1'
  Configuration:
    NotificationArn: !Ref CodePipelineSNSTopic
    CustomData: !Sub 'Do you want to create a change set against the production
stack and delete the ${TestStackName} stack?'
    RunOrder: '2'
- Name: DeleteTestStack
  ActionTypeId:
    Category: Deploy
    Owner: AWS
    Provider: CloudFormation
    Version: '1'
  Configuration:
    ActionMode: DELETE_ONLY
    RoleArn: !GetAtt [CFNRole, Arn]
    StackName: !Ref TestStackName
    RunOrder: '3'

```

3단계: ProdStage

파이프라인의 ProdStage 단계에서는 기존 프로덕션 스택과 대조하여 변경 세트를 생성한 다음, 승인을 기다렸다가 변경 세트를 실행합니다.

변경 세트는 수정 사항을 구현하기 전에 AWS CloudFormation에서 프로덕션 스택에 대해 수행할 모든 수정 사항에 대한 미리 보기를 제공합니다. 파이프라인을 처음 실행할 때는 실행 중인 프로덕션 스택이 없습니다. 변경 세트는 테스트 스택 생성 시 AWS CloudFormation에서 수행한 작업을 보여줍니다. 변경 세트를 생성하기 위해 CreateChangeSet 작업은 TemplateSource 입력 아티팩트의 WordPress 샘플 템플릿과 프로덕션 템플릿 구성을 사용합니다.

이전 단계와 마찬가지로, ApproveChangeSet 작업은 파이프라인을 일시 정지하고 이메일 알림을 보냅니다. 파이프라인이 일시 정지된 동안은 변경 세트를 보고 프로덕션 WordPress 스택에 대해 제안된

모든 수정 사항을 확인할 수 있습니다. CodePipeline을 사용하여 이 작업을 [승인 또는 거부](#)함으로써 각 파이프라인을 계속 진행하거나 중지합니다.

이 작업을 승인한 후에는 ExecuteChangeSet 작업이 변경 세트를 실행하므로 AWS CloudFormation에서 변경 세트에 설명된 모든 작업을 수행합니다. 초기 실행의 경우, AWS CloudFormation에서 WordPress 프로덕션 스택을 생성합니다. 후속 실행 시 AWS CloudFormation에서는 스택을 업데이트합니다.

```
- Name: ProdStage
  Actions:
    - Name: CreateChangeSet
      ActionTypeId:
        Category: Deploy
        Owner: AWS
        Provider: CloudFormation
        Version: '1'
      InputArtifacts:
        - Name: TemplateSource
      Configuration:
        ActionMode: CHANGE_SET_REPLACE
        RoleArn: !GetAtt [CFNRole, Arn]
        StackName: !Ref ProdStackName
        ChangeSetName: !Ref ChangeSetName
        TemplateConfiguration: !Sub "TemplateSource::${ProdStackConfig}"
        TemplatePath: !Sub "TemplateSource::${TemplateFileName}"
      RunOrder: '1'
    - Name: ApproveChangeSet
      ActionTypeId:
        Category: Approval
        Owner: AWS
        Provider: Manual
        Version: '1'
      Configuration:
        NotificationArn: !Ref CodePipelineSNSTopic
        CustomData: !Sub 'A new change set was created for the ${ProdStackName} stack.
Do you want to implement the changes?'
      RunOrder: '2'
    - Name: ExecuteChangeSet
      ActionTypeId:
        Category: Deploy
        Owner: AWS
        Provider: CloudFormation
        Version: '1'
```

```
Configuration:
  ActionMode: CHANGE_SET_EXECUTE
  ChangeSetName: !Ref ChangeSetName
  RoleArn: !GetAtt [CFNRole, Arn]
  StackName: !Ref ProdStackName
  RunOrder: '3'
```

3단계: WordPress 스택 보기

CodePipeline은 파이프라인을 따라 실행하면서 CloudFormation을 사용하여 테스트 및 프로덕션 스택을 생성합니다. 이러한 스택 및 출력의 상태를 보려면 CloudFormation 콘솔을 사용합니다.

스택을 보려면

1. <https://console.aws.amazon.com/cloudformation/>에서 CloudFormation 콘솔을 엽니다.
2. 파이프라인이 테스트 단계에 있는지 프로덕션 단계에서 있는지에 따라, Test-MyWordPressSite 또는 Prod-MyWordPressSite 스택을 선택합니다.
3. 스택의 상태를 보려면 [이벤트](#)를 확인합니다.

스택이 실패 상태인 경우 상태 이유를 보고 스택 오류를 찾습니다. 오류를 수정하고 나서 파이프라인을 다시 실행합니다. 스택이 CREATE_COMPLETE 상태에 있는 경우 관련 출력을 보고 WordPress 사이트의 URL을 가져옵니다.

CodePipeline을 사용하여 샘플 WordPress 사이트에 대해 연속 전달 워크플로우를 빌드했습니다. 변경 내용을 S3 버킷으로 제출할 경우 CodePipeline에서는 새 버전을 자동으로 감지한 다음 파이프라인을 다시 실행합니다. 이 워크플로우를 사용하면 프로덕션 사이트를 변경하기 전에 변경 내용을 보다 쉽게 제출 및 테스트할 수 있습니다.

4단계: 리소스 정리

원치 않는 서비스에 대해 요금이 청구되지 않도록 하려면 리소스를 삭제합니다.

Important

파이프라인 스택을 삭제하기 전에 테스트 및 프로덕션 WordPress 스택을 삭제합니다. 파이프라인 스택에는 WordPress 스택을 삭제하는 데 필요한 서비스 역할이 포함되어 있습니다. 파이프라인 스택을 처음 삭제한 경우 다른 서비스 역할 Amazon 리소스 이름(ARN)과 WordPress 스택을 연결하고 나서 이러한 스택을 삭제할 수 있습니다.

아티팩트 스토어에서 객체를 삭제하려면

1. <https://console.aws.amazon.com/s3/>에서 Amazon S3 콘솔을 엽니다.
2. CodePipeline에서 파이프라인의 아티팩트 스토어로 사용된 S3 버킷을 선택합니다.

버킷의 이름은 *stackname-artifactstorebucket-id* 형식이어야 합니다. 이 연습을 살펴본 경우 버킷의 이름이 다음 예제와 비슷할 것입니다. sample-WordPress-pipeline-artifactstorebucket-12345abcd12345.

3. 아티팩트 스토어 S3 버킷에서 모든 객체를 삭제합니다.

다음 단계에서 파이프라인 스택을 삭제할 경우 이 버킷이 비어 있어야 합니다. 그렇지 않으면, CloudFormation에서 버킷을 삭제할 수 없습니다.

스택을 삭제하는 방법

1. CloudFormation 콘솔에서 삭제할 스택을 선택합니다.

파이프라인에서 생성된 WordPress 스택이 여전히 실행 중인 경우 이들 스택을 먼저 선택합니다. 기본적으로 스택 이름은 Test-MyWordPressSite 및 Prod-MyWordPressSite입니다.

이미 WordPress 스택을 삭제한 경우 sample-WordPress-pipeline 스택을 선택합니다.

2. [Actions]를 선택한 다음 [Delete Stack]을 선택합니다.
3. 확인 메시지에서 예, 삭제를 선택합니다.

CloudFormation은 EC2 인스턴스, 알림 주제, 서비스 역할 및 파이프라인 등 스택의 모든 리소스를 삭제합니다.

이제 CodePipeline을 사용하여 기본 CloudFormation 워크플로우를 구성하는 방법을 이해했으므로 샘플 템플릿 및 아티팩트에서 시작하여 자신만의 고유한 템플릿과 아티팩트를 빌드할 수 있습니다.

다음 사항도 참조하세요.

이러한 파라미터를 이용할 때 참조할 수 있는 관련 리소스는 다음과 같습니다.

- CodePipeline의 CloudFormation 작업 파라미터에 대한 자세한 내용은 AWS CodePipeline 사용 설명서의 [CloudFormation deploy action configuration reference](#)를 참조하세요.
- Owner 필드 또는 configuration 필드와 같은 작업 공급자별 예제 템플릿 값은 AWS CodePipeline 사용 설명서의 [Action structure reference](#)에서 확인하세요.

- YAML 또는 JSON 형식의 예제 파이프라인 스택 템플릿을 다운로드하려면 AWS CodePipeline 사용 설명서의 [Tutorial: Create a pipeline with CloudFormation](#)을 참조하세요.

CloudFormation 구성 속성 참조

CodePipeline 파이프라인을 구축하는 경우 CloudFormation을 공급자로 사용해 파이프라인에 Deploy 작업을 추가합니다. 그런 다음 파이프라인에서 호출하는 CloudFormation 작업과 해당 작업의 설정을 지정해야 합니다. 이 주제에서는 CloudFormation 구성 속성에 대해 설명합니다. 속성을 지정하기 위해 CodePipeline 콘솔을 사용하거나 AWS CLI, CodePipeline API 또는 CloudFormation 템플릿에 사용할 JSON 객체를 생성할 수 있습니다.

주제

- [구성 속성\(콘솔\)](#)
- [구성 속성\(JSON 객체\)](#)
- [다음 사항도 참조하세요.](#)

구성 속성(콘솔)

CodePipeline [콘솔](#)은 구성 속성을 보여주고 선택한 작업 모드를 기반으로 필요한 속성을 나타냅니다.

Note

파이프라인을 생성하는 경우 스택 생성 또는 업데이트 또는 변경 사항 세트 생성 또는 교체 작업 모드를 지정할 수 있습니다. 고급 섹션의 속성은 파이프라인을 편집하는 경우에만 사용할 수 있습니다.

작업 모드

연결된 스테이지를 처리할 때 CodePipeline에서 호출하는 CloudFormation 작업입니다. 다음 작업 모드 중 하나를 선택합니다.

- 변경 사항 세트 생성 또는 교체는 변경 세트가 존재하지 않는 경우 스택 이름 및 제출하는 템플릿을 기반으로 변경 세트를 생성합니다. 변경 세트가 존재하는 경우 CloudFormation에서는 해당 변경 세트를 삭제한 다음 새로 생성합니다.

- 스택 생성 또는 업데이트는 지정된 스택이 없는 경우 스택을 생성합니다. 스택이 있는 경우 CloudFormation에서는 스택을 업데이트합니다. 이 작업을 사용하여 기존 스택을 업데이트합니다. CodePipeline은 스택을 대체하지 않습니다.
- 스택 삭제는 스택을 삭제합니다. 존재하지 않는 스택을 지정하면 이 작업이 스택을 삭제하지 않고 성공적으로 완료됩니다.
- 변경 사항 세트 실행은 변경 세트를 실행합니다.
- 불량 스택 교체는 지정된 스택이 없는 경우 스택을 생성합니다. 스택이 존재하는데 실패 상태이면(ROLLBACK_COMPLETE, ROLLBACK_FAILED, CREATE_FAILED, DELETE_FAILED 또는 UPDATE_ROLLBACK_FAILED로 보고됨) CloudFormation에서는 해당 스택을 삭제한 다음 새로 생성합니다. 스택이 실패 상태가 아니면 CloudFormation에서는 스택을 업데이트합니다. 이 작업을 사용하면 실패한 스택을 복구하거나 문제를 해결하지 않고 실패한 스택을 대체합니다. 이 모드는 일반적으로 테스트를 위해 선택합니다.

스택 이름

기존 스택 또는 생성할 스택과 연결된 이름입니다. 이름은 스택을 생성하는 AWS 리전에서 고유해야 합니다.

Note

스택 이름에는 영숫자(대소문자 구분)와 하이픈만 사용할 수 있습니다. 영문자로 시작해야 하고 128자 이하여야 합니다.

변경 세트 이름

지정된 스택에 대해 생성하려는 새 변경 세트 또는 기존 변경 세트의 이름입니다.

템플릿

CloudFormation 템플릿 파일의 위치로, *ArtifactName::TemplateFileName* 형식을 따릅니다.

템플릿 구성

템플릿 구성 파일의 위치로, *ArtifactName::TemplateConfigurationFileName* 형식을 따릅니다. 템플릿 구성 파일에는 템플릿 파라미터 값, 스택 정책, 태그가 포함될 수 있습니다. 이 파일에 민감한 정보(예: 암호)를 포함하면 해당 파일에 대한 액세스를 제한합니다. 자세한 내용은 [CloudFormation 아티팩트](#) 섹션을 참조하세요.

기능

특정 리소스가 포함된 스택의 경우 CloudFormation에서 리소스를 생성 또는 업데이트할 수 있음을 나타내는 명시적으로 승인입니다. 예를 들어 스택 템플릿에 AWS Identity and Access Management(IAM) 리소스가 포함되어 있는 경우 CAPABILITY_IAM을 지정해야 합니다. 자세한 내용은 [CreateStack](#) API 작업 요청 파라미터를 참조하세요.

스택 템플릿에 IAM 리소스가 포함되어 있는 경우 이 속성을 지정해야 합니다.

여러 기능을 지정할 수 있습니다.

역할 이름

지정된 스택의 리소스에 대해 작동 중인 경우 CloudFormation에서 수입하는 IAM 서비스 역할의 이름입니다.

출력 파일 이름

고급 섹션에서는 CodePipeline이 지정된 작업을 수행한 후 [출력 아티팩트](#)에 추가하는 출력 파일(예: CreateStackOutput.json)을 지정할 수 있습니다. 출력 아티팩트에는 CloudFormation 템플릿의 Outputs 섹션의 내용으로 구성된 JSON 파일이 들어 있습니다.

이름을 지정하지 않으면 CodePipeline에서는 출력 아티팩트를 생성하지 않습니다.

파라미터 재정의

파라미터는 템플릿에서 정의되며, 이를 통해 스택을 생성하거나 업데이트 할 때 사용자 지정 값을 입력할 수 있습니다. 템플릿 구성 파일의 템플릿 파라미터 값을 재정의하는 JSON 객체를 지정할 수 있습니다. 스택 템플릿에는 모든 파라미터 이름이 존재해야 합니다. 자세한 내용은 [CloudFormation 템플릿 Parameters 구문](#) 섹션을 참조하세요.

Note

ParameterOverrides 속성에 저장할 수 있는 JSON 객체의 최대 크기 제한은 1킬로바이트입니다.

템플릿 구성 파일을 사용하여 대부분의 파라미터 값을 지정하는 것이 좋습니다. 파라미터 재정의를 사용하여 동적 파라미터 값만 지정합니다. 동적 파라미터는 파이프라인을 실행할 때까지 알 수 없습니다.

다음 예에서는 파라미터 재정의 기능을 사용하여 ParameterName 파라미터의 값을 정의합니다. 이 기능은 CodePipeline 입력 아티팩트에서 값을 검색합니다. 파라미터 재정의 기능에 대한 자세한 내용은 [CodePipeline 파이프라인에서 파라미터 재정의 함수 사용](#) 단원을 참조하세요.

```
{
  "ParameterName" : { "Fn::GetParam" : ["ArtifactName", "config-file-name.json",
    "ParamName"]}
}
```

구성 속성(JSON 객체)

단계 작업의 공급자로 CloudFormation을 지정하는 경우 Configuration 속성 내에서 다음 속성을 정의합니다. AWS CLI, CodePipeline API 또는 CloudFormation 템플릿에 JSON 객체를 사용합니다. 예제는 [연습: 테스트 및 프로덕션 스택용 파이프라인 빌드](#) 및 [CloudFormation 구성 속성 참조](#) 단원을 참조하세요.

ActionMode

연결된 스테이지를 처리할 때 CodePipeline에서 호출하는 CloudFormation 작업입니다. 다음 작업 모드 중 하나만 지정합니다.

- **CHANGE_SET_EXECUTE**: 변경 세트를 실행합니다.
- **CHANGE_SET_REPLACE**: 변경 세트가 존재하지 않는 경우 스택 이름 및 제출하는 템플릿을 기반으로 변경 세트를 생성합니다. 변경 세트가 존재하는 경우 CloudFormation에서는 해당 변경 세트를 삭제한 다음 새로 생성합니다.
- **CREATE_UPDATE**: 지정된 스택이 존재하지 않는 경우 스택을 생성합니다. 스택이 있는 경우 CloudFormation에서는 스택을 업데이트합니다. 이 작업을 사용하여 기존 스택을 업데이트합니다. CodePipeline은 스택을 대체하지 않습니다.
- **DELETE_ONLY**: 스택을 삭제합니다. 존재하지 않는 스택을 지정하면 이 작업이 스택을 삭제하지 않고 성공적으로 완료됩니다.
- **REPLACE_ON_FAILURE**: 지정된 스택이 존재하지 않는 경우 스택을 생성합니다. 스택이 존재하는데 실패 상태이면(ROLLBACK_COMPLETE, ROLLBACK_FAILED, CREATE_FAILED, DELETE_FAILED 또는 UPDATE_ROLLBACK_FAILED로 보고됨) CloudFormation에서는 스택을 삭제한 다음 새 스택을 생성합니다. 스택이 실패 상태가 아니면 CloudFormation에서는 스택을 업데이트합니다. 이 작업을 사용하면 실패한 스택을 복구하거나 문제를 해결하지 않고 실패한 스택을 자동으로 대체합니다. 이 모드는 일반적으로 테스트를 위해 선택합니다.

이 속성은 필수입니다.

Capabilities

특정 리소스가 포함된 스택의 경우 CloudFormation에서 리소스를 생성 또는 업데이트할 수 있음을 나타내는 명시적으로 승인입니다. 예를 들어 스택 템플릿에 AWS Identity and Access Management(IAM) 리소스가 포함되어 있는 경우 CAPABILITY_IAM을 지정해야 합니다. 자세한 내용은 [CreateStack](#) API 작업 요청 파라미터를 참조하세요.

이 속성은 조건부입니다. 스택 템플릿에 IAM 리소스가 포함되어 있는 경우 이 속성을 지정해야 합니다.

여러 기능을 지정할 수 있습니다. 다음 예제에서는 템플릿에 CAPABILITY_IAM 및 CAPABILITY_AUTO_EXPAND 속성을 추가합니다.

YAML

```
configuration:
  ActionMode: CHANGE_SET_REPLACE
  Capabilities: CAPABILITY_IAM,CAPABILITY_AUTO_EXPAND
  ChangeSetName: pipeline-changeset
  RoleArn: CloudFormation_Role_ARN
  StackName: my-pipeline-stack
  TemplateConfiguration: 'my-pipeline-stack::template-configuration.json'
  TemplatePath: 'my-pipeline-stack::template-export.yml'
```

JSON

```
"configuration": {
  "ActionMode": "CHANGE_SET_REPLACE",
  "Capabilities": "CAPABILITY_IAM,CAPABILITY_AUTO_EXPAND",
  "ChangeSetName": "pipeline-changeset",
  "RoleArn": "CloudFormation_Role_ARN",
  "StackName": "my-pipeline-stack",
  "TemplateConfiguration": "my-pipeline-stack::template-configuration.json",
  "TemplatePath": "my-pipeline-stack::template-export.yml"
}
```

ChangeSetName

지정된 스택에 대해 생성하려는 새 변경 세트 또는 기존 변경 세트의 이름입니다.

작업 모드 CHANGE_SET_REPLACE 및 CHANGE_SET_EXECUTE의 경우 이 속성은 필수입니다. 기타 모든 작업 모드의 경우 이 속성은 무시됩니다.

OutputFileName

출력 파일의 이름(예: CreateStackOutput.json)입니다. CodePipeline에서는 지정된 작업을 수행한 후 파일을 [출력 아티팩트](#)에 추가합니다. 출력 아티팩트에는 CloudFormation 템플릿의 Outputs 섹션의 내용으로 구성된 JSON 파일이 들어 있습니다.

이 속성은 선택 사항입니다. 이름을 지정하지 않으면 CodePipeline에서는 출력 아티팩트를 생성하지 않습니다.

ParameterOverrides

파라미터는 템플릿에서 정의되며, 이를 통해 스택을 생성하거나 업데이트 할 때 사용자 지정 값을 입력할 수 있습니다. 템플릿 구성 파일의 템플릿 파라미터 값을 재정의하는 JSON 객체를 지정할 수 있습니다. 스택 템플릿에는 모든 파라미터 이름이 존재해야 합니다. 자세한 내용은 [CloudFormation 템플릿 Parameters 구문](#) 섹션을 참조하세요.

다음 예제는 InstanceType 및 KeyName 파라미터 재정의의 템플릿에 추가합니다.

YAML

```
configuration:
  ActionMode: CHANGE_SET_REPLACE
  Capabilities: CAPABILITY_NAMED_IAM
  ChangeSetName: pipeline-changeset
  ParameterOverrides: '{"InstanceType": "t2.small","KeyName": "my-keypair"}'
  RoleArn: CloudFormation_Role_ARN
  StackName: my-pipeline-stack
  TemplateConfiguration: 'my-pipeline-stack::template-configuration.json'
  TemplatePath: 'my-pipeline-stack::template-export.yml'
```

JSON

```
"configuration": {
  "ActionMode": "CHANGE_SET_REPLACE",
  "Capabilities": "CAPABILITY_NAMED_IAM",
  "ChangeSetName": "pipeline-changeset",
  "ParameterOverrides": "{\"InstanceType\": \"t2.small\", \"KeyName\": \"my-keypair\"}",
  "RoleArn": "CloudFormation_Role_ARN",
  "StackName": "my-pipeline-stack",
  "TemplateConfiguration": "my-pipeline-stack::template-configuration.json",
  "TemplatePath": "my-pipeline-stack::template-export.yml"
```

```
}
```

Note

ParameterOverrides 속성에 저장할 수 있는 JSON 객체의 최대 크기는 1KB입니다.

템플릿 구성 파일을 사용하여 대부분의 파라미터 값을 지정하는 것이 좋습니다. 파라미터 재정의의 사용에 동적 파라미터 값만 지정합니다. 동적 파라미터 값은 파이프라인을 실행할 때까지 알 수 없습니다.

다음 예에서는 파라미터 재정의 기능을 사용하여 ParameterName 파라미터의 값을 정의합니다. 이 기능은 CodePipeline 입력 아티팩트에서 값을 검색합니다. 파라미터 재정의 기능에 대한 자세한 내용은 [CodePipeline 파이프라인에서 파라미터 재정의 함수 사용](#) 단원을 참조하세요.

```
{
  "ParameterName" : { "Fn::GetParam" : ["ArtifactName", "config-file-name.json",
    "ParamName"]}
}
```

이 속성은 선택 사항입니다.

RoleArn

스택의 리소스에 대해 작동 중인 경우 CloudFormation이 수입하는 IAM 서비스 역할의 Amazon 리소스 이름(ARN)입니다.

CREATE_UPDATE, REPLACE_ON_FAILURE, DELETE_ONLY 및 CHANGE_SET_REPLACE 작업 모드
의 경우 이 속성은 필수입니다. 변경 세트를 실행할 때 RoleArn이 적용되지 않습니다. 변경 세트를 생성하기 위해 CodePipeline을 사용하지 않는 경우 변경 세트 또는 스택에 연결된 역할이 있는지 확인해야 합니다.

StackName

기존 스택 또는 생성하려는 스택의 이름입니다.

모든 작업 모드의 경우 이 속성은 필수입니다.

TemplateConfiguration

TemplateConfiguration은 템플릿 구성 파일입니다. 이 작업의 입력 아티팩트에 파일을 포함합니다. 템플릿 구성 파일 이름은 다음 형식을 따릅니다.

Artifactname::TemplateConfigurationFileName

Artifactname은 CodePipeline에 표시될 때 입력 아티팩트 이름입니다. 예를 들어, 아티팩트 이름이 SourceArtifact이며 파일 이름이 test-configuration.json인 소스 단계는 다음 예제에 표시된 대로 TemplateConfiguration 이름을 생성합니다.

```
"TemplateConfiguration": "SourceArtifact::test-configuration.json"
```

템플릿 구성 파일에는 템플릿 파라미터 값과 스택 정책을 포함할 수 있습니다. 이 파일에 민감한 정보(예: 암호)를 포함하면 해당 파일에 대한 액세스를 제한합니다. 예제 템플릿 구성 파일은 [CloudFormation 아티팩트](#) 단원을 참조하십시오.

이 속성은 선택 사항입니다.

TemplatePath

TemplatePath는 CloudFormation 템플릿 파일을 나타냅니다. 이 작업의 입력 아티팩트에 파일을 포함합니다. 파일 이름은 다음과 같은 형식을 따릅니다.

Artifactname::TemplateFileName

Artifactname은 CodePipeline에 표시될 때 입력 아티팩트 이름입니다. 예를 들어 아티팩트 이름이 SourceArtifact이며 파일 이름이 template.yaml인 소스 단계는 다음 예제에 표시된 대로 TemplatePath 이름을 생성합니다.

```
"TemplatePath": "SourceArtifact::template.yaml"
```

작업 모드 CREATE_UPDATE, REPLACE_ON_FAILURE 및 CHANGE_SET_REPLACE의 경우 이 속성은 필수입니다. 기타 모든 작업 모드의 경우 이 속성은 무시됩니다.

다음 사항도 참조하세요.

이러한 파라미터를 이용할 때 참조할 수 있는 관련 리소스는 다음과 같습니다.

- CodePipeline의 CloudFormation 작업 파라미터에 대한 자세한 내용은 AWS CodePipeline 사용 설명서의 [CloudFormation deploy action configuration reference](#)를 참조하세요.
- Owner 필드 또는 configuration 필드와 같은 작업 공급자별 예제 템플릿 값은 AWS CodePipeline 사용 설명서의 [Action structure reference](#)에서 확인하세요.

- YAML 또는 JSON 형식의 예제 파이프라인 스택 템플릿을 다운로드하려면 AWS CodePipeline 사용 설명서의 [Tutorial: Create a pipeline with CloudFormation](#)을 참조하세요.

CloudFormation 아티팩트

CodePipeline에서 CodePipeline이 파이프라인을 실행할 때 아티팩트에 대한 작업을 수행합니다. CloudFormation의 경우 아티팩트에는 스택 템플릿 파일, 템플릿 구성 파일 또는 2가지 모두 포함될 수 있습니다. CodePipeline은 이러한 아티팩트를 사용하여 CloudFormation 스택 및 변경 세트 작업을 수행합니다.

Amazon Simple Storage Service(Amazon S3)를 소스 리포지토리로 사용하는 경우, 템플릿 및 템플릿 구성 파일을 파일 하나로 압축한 다음 S3 버킷에 업로드해야 합니다. 기타 리포지토리(예: GitHub 및 AWS CodeCommit)의 경우 압축하지 않고 아티팩트를 업로드합니다. 자세한 내용은 AWS CodePipeline 사용 설명서의 [Create a pipeline, stages, and actions](#)를 참조하세요.

리포지토리에는 파일을 원하는 만큼 추가할 수 있습니다. 예를 들어 동일한 템플릿에 대해 두 가지 구성, 즉 테스트 구성 및 프로덕션 구성을 포함할 수 있습니다.

이 주제에서는 각 아티팩트 유형에 대해 설명합니다.

주제

- [스택 템플릿 파일](#)
- [템플릿 구성 파일](#)
- [다음 사항도 참조하세요.](#)

스택 템플릿 파일

스택 템플릿 파일은 CloudFormation에서 프로비저닝 및 구성하는 리소스를 정의합니다. 이러한 파일은 CloudFormation을 사용하여 스택을 생성 또는 업데이트하는 경우 사용하는 템플릿 파일과 동일합니다. YAML 또는 JSON 형식 템플릿을 사용할 수 있습니다. 템플릿에 대한 자세한 내용은 단원을 참조하세요 [CloudFormation 템플릿 섹션](#)

템플릿 구성 파일

템플릿 구성 파일은 JSON 형식 텍스트 파일로, 템플릿 파라미터 값, [스택 정책](#) 및 태그를 지정할 수 있습니다. 템플릿 구성 파일을 사용하여 스택에 대한 파라미터 값 또는 스택 정책을 지정합니다. 지정한 파라미터 값은 모두 연결된 템플릿에서 선언되어 있어야 합니다.

이 파일에 민감한 정보(예: 암호)를 포함하면 해당 파일에 대한 액세스를 제한합니다. 예를 들어, 아티팩트를 S3 버킷에 업로드하면 [S3 버킷 정책 또는 사용자 정책](#)을 사용하여 액세스를 제한합니다.

구성 파일을 생성하려면 다음 형식을 사용합니다.

```
{
  "Parameters" : {
    "NameOfTemplateParameter" : "ValueOfParameter",
    ...
  },
  "Tags" : {
    "TagKey" : "TagValue",
    ...
  },
  "StackPolicy" : {
    "Statement" : [
      StackPolicyStatement
    ]
  }
}
```

다음 예에서는 TestEC2Key 파라미터에 대해 KeyName를 지정하고, 값이 Department인 Marketing 태그를 추가하고, 리소스를 삭제하는 업데이트를 제외하고 모든 업데이트 작업을 허용하는 스택 정책을 추가합니다.

```
{
  "Parameters" : {
    "KeyName" : "TestEC2Key"
  },
  "Tags" : {
    "Department" : "Marketing"
  },
  "StackPolicy" : {
    "Statement" : [
      {
        "Effect" : "Allow",
        "NotAction" : "Update:Delete",
        "Principal": "*",
        "Resource" : "*"
      }
    ]
  }
}
```

}

다음 사항도 참조하세요.

이러한 파라미터를 이용할 때 참조할 수 있는 관련 리소스는 다음과 같습니다.

- CodePipeline의 CloudFormation 작업 파라미터에 대한 자세한 내용은 AWS CodePipeline 사용 설명서의 [CloudFormation deploy action configuration reference](#)를 참조하세요.
- Owner 필드 또는 configuration 필드와 같은 작업 공급자별 예제 템플릿 값은 AWS CodePipeline 사용 설명서의 [Action structure reference](#)에서 확인하세요.
- YAML 또는 JSON 형식의 예제 파이프라인 스택 템플릿을 다운로드하려면 AWS CodePipeline 사용 설명서의 [Tutorial: Create a pipeline with CloudFormation](#)을 참조하세요.

CodePipeline 파이프라인에서 파라미터 재정의의 함수 사용

CodePipeline 스테이지에서 CloudFormation 작업에 대한 [파라미터 재정의](#)를 지정할 수 있습니다. 파라미터 재정의의 사용하면 템플릿 구성 파일에서 값을 재정의하는 템플릿 파라미터 값을 지정할 수 있습니다. CloudFormation은 동적 값(파이프라인이 실행될 때까지 알 수 없는 값)을 지정하는 데 도움이 되는 함수를 제공합니다.

주제

- [Fn::GetArtifactAtt](#)
- [Fn::GetParam](#)
- [다음 사항도 참조하세요.](#)

Fn::GetArtifactAtt

Fn::GetArtifactAtt 함수는 아티팩트가 저장되는 S3 버킷 이름과 같은 입력 아티팩트에서 속성의 값을 가져옵니다. 이 함수를 사용하여 파일 이름 또는 Amazon S3 버킷 이름과 같은 아티팩트의 속성을 지정합니다.

파이프라인을 실행하면 CodePipeline에서 파일을 복사하여 파이프라인의 아티팩트 스토어(S3 버킷)에 기록합니다. CodePipeline은 아티팩트 스토어에 파일 이름을 생성합니다. 파이프라인을 실행하기 전에는 이러한 파일 이름을 알 수 없습니다.

예를 들어, 파이프라인에는 CodePipeline에서 AWS Lambda 함수 소스 코드를 아티팩트 스토어에 복사하는 소스 스테이지가 있을 수 있습니다. 다음 스테이지에는 Lambda 함수를 생성하는

CloudFormation 템플릿이 있지만, CloudFormation에서 함수를 생성하려면 파일 이름이 필요합니다. `Fn::GetArtifactAtt` 함수를 사용하여 정확한 S3 버킷 및 파일 이름을 전달해야 합니다.

구문

다음 구문을 사용하여 아티팩트의 속성 값을 가져옵니다.

```
{ "Fn::GetArtifactAtt" : [ "artifactName", "attributeName" ] }
```

artifactName

입력 아티팩트의 이름입니다. 연결된 작업에 대한 입력으로 이 아티팩트를 선언해야 합니다.

attributeName

값을 가져올 아티팩트 속성의 이름입니다. 각 아티팩트 속성에 대한 세부 정보는 다음 속성 단원을 참조하십시오.

예제

다음 파라미터 재정의는 BucketName 아티팩트의 S3 버킷 이름과 파일 이름을 가져와서 ObjectKey 및 LambdaFunctionSource 파라미터를 지정합니다. 이 예제에서는 CodePipeline에서 Lambda 함수 소스 코드를 복사하여 아티팩트로(예: 소스 단계의 부분으로) 저장했다고 가정합니다.

```
{
  "BucketName" : { "Fn::GetArtifactAtt" : ["LambdaFunctionSource", "BucketName"] },
  "ObjectKey" : { "Fn::GetArtifactAtt" : ["LambdaFunctionSource", "ObjectKey"] }
}
```

속성

아티팩트의 다음과 같은 속성을 가져올 수 있습니다.

BucketName

아티팩트가 저장되는 S3 버킷의 이름입니다.

ObjectKey

CodePipeline에서 생성된 아티팩트가 포함되어 있는 .zip 파일의 이름입니다(예: 1ABCyZZ.zip).

URL

아티팩트의 Amazon Simple Storage Service(Amazon S3)URL입니다(예: `https://s3.us-west-2.amazonaws.com/artifactstorebucket-yivczw8jma0c/test/TemplateSo/1ABCyZZ.zip`).

Fn::GetParam

Fn::GetParam 함수는 JSON 형식 파일의 키-값 페어에서 값을 반환합니다. JSON 파일은 아티팩트에 포함되어야 합니다.

이 함수를 사용하여 CloudFormation 스택에서 출력 값을 가져오고 이 값을 다른 작업에 대한 입력으로 사용합니다. 예를 들어, CloudFormation 작업에 대한 출력 파일 이름을 지정할 경우 CodePipeline에서 출력을 JSON 파일에 저장한 다음 출력 아티팩트의 .zip 파일에 추가합니다. Fn::GetParam 함수를 사용하여 출력 값을 가져오고 이 값을 다른 작업에 대한 입력으로 사용합니다.

구문

다음 구문을 사용하여 키-값 페어에서 값을 가져옵니다.

```
{ "Fn::GetParam" : [ "artifactName", "JSONFileName", "keyName" ] }
```

artifactName

연결된 작업에 대한 입력 아티팩트로 포함되어야 하는 아티팩트의 이름입니다.

JSONFileName

아티팩트에 포함되는 JSON 파일의 이름입니다.

keyName

값을 가져올 키의 이름입니다.

예제

다음 예제에서는 파라미터 재정의에서 Fn::GetParam 함수를 사용하는 방법을 보여 줍니다.

구문

다음 파라미터 재정의는 WebSiteURL 아티팩트에 있는 URL 파일에서 `stack-output.json` 키의 값을 가져와 WebStackOutput 파라미터를 지정합니다.

```
{
  "WebSiteURL" : { "Fn::GetParam" : ["WebStackOutput", "stack-output.json", "URL"]}
}
```

CloudFormation 템플릿 코드 조각

CodePipeline 파이프라인의 다음 CloudFormation 템플릿 코드 조각은 스택 출력을 전달하는 방법을 보여줍니다. 이 코드 조각은 파이프라인 정의의 2단계를 보여 줍니다. 첫 번째 단계는 스택을 생성하고 해당 출력을 TestOutput.json 아티팩트의 StackAOutput 파일에 저장합니다. 이러한 값은 OutputFileName 및 OutputArtifacts 속성에서 지정됩니다.

단계에서 사용할 소스 입력 아티팩트의 이름은 TemplateSource입니다. 스택 템플릿의 파일 이름은 teststackA.yaml이며 구성 파일의 파일 이름은 test-configuration.json입니다. 두 단계 모두에서 이러한 값은 다음과 같이 TemplateConfiguration 및 TemplatePath 속성에 지정됩니다.

```
TemplateConfiguration: TemplateSource::test-configuration.json
TemplatePath: TemplateSource::teststackA.yaml
```

Example스택 A 생성 스테이지

```
- Name: CreateTestStackA
  Actions:
    - Name: CloudFormationCreate
      ActionTypeId:
        Category: Deploy
        Owner: AWS
        Provider: CloudFormation
        Version: '1'
      Configuration:
        ActionMode: CREATE_UPDATE
        Capabilities: CAPABILITY_IAM
        OutputFileName: TestOutput.json
        RoleArn: !GetAtt [CFNRole, Arn]
        StackName: StackA
        TemplateConfiguration: TemplateSource::test-configuration.json
        TemplatePath: TemplateSource::teststackA.yaml
      InputArtifacts:
        - Name: TemplateSource
      OutputArtifacts:
        - Name: StackAOutput
      RunOrder: '1'
```

후속 단계에서 스택 B는 스택 A의 출력을 사용합니다. 예제에서는 `ParameterOverrides` 속성에서 `Fn::GetParam` 함수를 사용하여 `StackBInputParam` 파라미터를 지정합니다. 결과 값은 `StackAOutputName` 키와 연결된 값입니다.

Example스택 B 생성 스테이지

```
- Name: CreateTestStackB
  Actions:
    - Name: CloudFormationCreate
      ActionTypeId:
        Category: Deploy
        Owner: AWS
        Provider: CloudFormation
        Version: '1'
      Configuration:
        ActionMode: CREATE_UPDATE
        Capabilities: CAPABILITY_IAM
        RoleArn: !GetAtt [CFNRole, Arn]
        StackName: StackB
        TemplateConfiguration: TemplateSource::test-configuration.json
        TemplatePath: TemplateSource::teststackB.yaml
        ParameterOverrides: |
          {
            "StackBInputParam" : { "Fn::GetParam" : ["StackAOutput", "TestOutput.json",
"StackAOutputName"]}
          }
      InputArtifacts:
        - Name: TemplateSource
        - Name: StackAOutput
      RunOrder: '1'
```

다음 사항도 참조하세요.

이러한 파라미터를 이용할 때 참조할 수 있는 관련 리소스는 다음과 같습니다.

- CodePipeline의 CloudFormation 작업 파라미터에 대한 자세한 내용은 AWS CodePipeline 사용 설명서의 [CloudFormation deploy action configuration reference](#)를 참조하세요.
- Owner 필드 또는 configuration 필드와 같은 작업 공급자별 예제 템플릿 값은 AWS CodePipeline 사용 설명서의 [Action structure reference](#)에서 확인하세요.
- YAML 또는 JSON 형식의 예제 파이프라인 스택 템플릿을 다운로드하려면 AWS CodePipeline 사용 설명서의 [Tutorial: Create a pipeline with CloudFormation](#)을 참조하세요.

CloudFormation 템플릿 참조 가이드

CloudFormation 템플릿 생성에 사용할 수 있는 리소스 및 구성 요소에 대한 자세한 내용은 이제 [CloudFormation 템플릿 참조 가이드](#)에서 확인할 수 있습니다. 이 가이드는 템플릿에 대한 기술 참조 역할을 하며 CloudFormation 사용 설명서의 시작하기 콘텐츠를 보완합니다.

CloudFormation 템플릿 참조 가이드는 다음과 같은 세부 정보를 제공합니다.

- 리소스 유형 및 해당 속성
- DeletionPolicy 및 기타 리소스 수준 속성
- 내장 함수(예: Ref, Sub)
- 변환(CloudFormation 매크로)
- EC2 인스턴스 구성을 위한 헬퍼 스크립트
- 시스템에서 읽을 수 있는 리소스 유형 정의

이 가이드는 다음을 반영하여 정기적으로 업데이트됩니다.

- 새로운 리소스 유형
- 속성 변경 사항
- 새 반환 값
- 설명서 업데이트

CloudFormation의 보안

AWS에서 클라우드 보안은 가장 중요합니다. AWS 고객은 보안에 가장 보안에 민감한 조직의 요구 사항에 부합하도록 빌드된 데이터 센터 및 네트워크 아키텍처의 혜택을 누릴 수 있습니다.

보안은 AWS와 사용자의 공동 책임입니다. [공동 책임 모델](#)은 이 사항을 클라우드의 보안 및 클라우드 내 보안으로 설명합니다.

- 클라우드의 보안: AWS는 AWS 클라우드에서 AWS 서비스를 실행하는 인프라를 보호할 책임이 있습니다. AWS는 안전하게 사용할 수 있는 서비스 또한 제공합니다. 서드 파티 감사자는 [AWS 규정 준수 프로그램](#)의 일환으로 보안 효과를 정기적으로 테스트하고 검증합니다. CloudFormation에 적용되는 규정 준수 프로그램에 대한 자세한 내용은 [규정 준수 프로그램의 범위에 속하는 AWS 서비스](#)를 참조하세요.
- 클라우드의 보안 – 귀하의 책임은 귀하가 사용하는 AWS 서비스에 의해 결정됩니다. 또한 귀하는 귀사의 데이터 민감도, 귀사의 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다.

이 설명서는 CloudFormation 사용 시 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다. 다음 주제에서는 보안 및 규정 준수 목표를 충족하도록 CloudFormation을(를) 구성하는 방법을 보여줍니다. 또한 CloudFormation 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스 사용 방법을 알아봅니다.

주제

- [CloudFormation 스택 삭제 방지](#)
- [스택 리소스에 대한 업데이트 방지](#)
- [CloudFormation의 데이터 보호](#)
- [AWS Identity and Access Management를 사용하여 CloudFormation 액세스 제어](#)
- [AWS CloudTrail를 사용하여 CloudFormation API 호출 로깅](#)
- [CloudFormation에서 인프라 보안](#)
- [CloudFormation의 복원성](#)
- [의 규정 준수 검증CloudFormation](#)
- [CloudFormation의 구성 및 취약성 분석](#)
- [CloudFormation의 보안 모범 사례](#)

- [인터페이스 엔드포인트를 사용하여 CloudFormation에 액세스\(AWS PrivateLink\)](#)

CloudFormation 스택 삭제 방지

스택의 종료 방지 기능을 활성화하여 스택이 실수로 삭제되는 것을 방지할 수 있습니다. 종료 방지 기능이 활성화된 스택을 삭제하려고 시도하면 삭제가 실패하고 스택은 상태를 포함하여 변함없이 그대로 유지됩니다. 스택을 생성할 때 종료 보호 기능을 활성화할 수 있습니다. 기본적으로 스택의 종료 방지 기능은 비활성화되어 있습니다. DELETE_IN_PROGRESS 또는 DELETE_COMPLETE를 제외한 모든 상태의 스택에 종료 보호를 설정할 수 있습니다.

스택 종료 방지 기능을 활성화하거나 비활성화하면 해당 스택에 속한 모든 중첩 스택에 대해서도 동일한 선택 사항이 전달됩니다. 종료 방지 기능을 중첩 스택에서 직접 활성화하거나 비활성화할 수 없습니다. 종료 방지 기능이 활성화된 스택에 속한 중첩 스택을 직접 삭제하려고 시도하면 작업이 실패하고 중첩 스택은 변함없이 그대로 유지됩니다.

그러나 사용자가 중첩 스택을 삭제하는 스택 업데이트를 하면 CloudFormation이 그에 따라 중첩 스택을 삭제합니다.

종료 방지 기능은 롤백을 비활성화하는 것과 다릅니다. 종료 방지 기능은 스택 삭제 시도 시에만 적용되고 롤백 비활성화는 스택 생성 실패 시 자동 롤백에 적용됩니다.

스택을 생성할 때 종료 방지 기능을 활성화하려면

Create stack(스택 생성) 마법사의 Specify stack options(스택 옵션 지정) 페이지에서 Advanced options(고급 옵션) 아래 Termination Protection(종료 방지) 섹션을 확장하고 Enable(활성화)을 선택합니다. 자세한 내용은 [스택 옵션 구성](#) 섹션을 참조하세요.

기존 스택에서 종료 방지 기능을 활성화하거나 비활성화하려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 원하는 스택을 선택합니다.

Note

스택 이름 옆에 NESTED가 표시되면 그 스택은 중첩 스택입니다. 중첩 스택이 속한 루트 스택에서만 종료 방지 기능을 변경할 수 있습니다.

4. 스택 세부 정보 창에서 Stack actions(스택 작업)를 선택한 다음 Edit termination protection(종료 방지 편집)을 선택합니다.

CloudFormation이 종료 방지 편집 대화 상자를 표시합니다.

5. 활성화 또는 비활성화를 선택한 다음 저장을 선택하십시오.

중첩 스택에서 종료 방지 기능을 활성화하거나 비활성화하려면

스택 이름 옆에 NESTED가 표시되면 그 스택은 중첩 스택입니다. 중첩 스택이 속한 루트 스택에서만 종료 방지 기능을 변경할 수 있습니다. 루트 스택에서 종료 방지 기능을 변경하려면

1. AWS Management Console에 로그인하여 <https://console.aws.amazon.com/cloudformation>에서 CloudFormation 콘솔을 엽니다.
2. 화면 상단의 탐색 모음에서 AWS 리전을 선택합니다.
3. 원하는 중첩 스택을 선택합니다.
4. Stack info(스택 정보) 창에서 개요 섹션의 Root stack(루트 스택)으로 나열된 스택 이름을 선택하십시오.

CloudFormation이 루트 스택에 대한 스택 세부 정보를 표시합니다.

5. Stack actions(스택 작업)을 선택하고 Edit Termination Protection(종료 방지 편집)을 선택하십시오.

CloudFormation이 종료 방지 편집 대화 상자를 표시합니다.

6. 활성화 또는 비활성화를 선택한 다음 저장을 선택하십시오.

명령줄을 사용한 종료 방지 기능의 활성화 또는 비활성화 방법

[update-termination-protection](#) 명령을 사용합니다.

스택에서 종료 방지 기능을 변경할 수 있는 대상 제어

스택에서 종료 방지 기능을 활성화하거나 비활성화하려면

cloudformation:UpdateTerminationProtection 작업에 대한 권한이 필요합니다. 예를 들어, 아래 정책을 통해 사용자가 스택에서 종료 방지 기능을 활성화하거나 비활성화할 수 있습니다.

CloudFormation에 권한을 지정하는 방법에 대한 자세한 내용은 [AWS Identity and Access Management를 사용하여 CloudFormation 액세스 제어](#) 단원을 참조하십시오.

Example스택 종료 방지 기능 변경 권한을 부여하는 샘플 정책

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "cloudformation:UpdateTerminationProtection"
    ],
    "Resource": "*"
  }]
}
```

스택 리소스에 대한 업데이트 방지

스택을 생성할 때 모든 리소스에 대한 모든 업데이트 작업이 허용됩니다. 기본적으로 스택 업데이트 권한이 있는 사용자는 스택에서 모든 리소스를 업데이트할 수 있습니다. 업데이트 중에 일부 리소스를 중단하거나 완전히 교체하여 새로운 물리적 ID 또는 완전히 새로운 스토리지를 생성해야 할 수 있습니다. 스택 정책을 사용하여 스택 업데이트 중에 스택 리소스를 실수로 업데이트하거나 삭제하는 것을 방지할 수 있습니다. 스택 정책은 지정된 리소스에 대해 수행할 수 있는 업데이트 작업을 정의하는 JSON 문서입니다.

스택 정책을 설정한 이후에는 기본적으로 스택 내의 모든 리소스가 보호됩니다. 특정 리소스에 대한 업데이트를 허용하려면 스택 정책에서 해당 리소스에 대해 명시적 Allow 문을 지정합니다. 스택 정책을 스택당 하나만 정의할 수 있지만, 단일 정책에서 여러 리소스를 보호할 수 있습니다. 스택 정책은 스택을 업데이트하려 하는 모든 CloudFormation 사용자에게 적용됩니다. 다른 스택 정책을 다른 사용자와 연결할 수 없습니다.

스택 정책은 스택 업데이트 중에만 적용됩니다. AWS Identity and Access Management(IAM) 정책처럼 액세스 제어를 제공하지 않습니다. 스택 정책을 안전 메커니즘으로만 사용하여 특정 스택 리소스가 실수로 업데이트되는 것을 방지하십시오. AWS 리소스 또는 작업에 대한 액세스를 제어하려면 IAM을 사용합니다.

주제

- [스택 정책 예제](#)
- [스택 정책 정의](#)

- [스택 정책 설정](#)
- [보호된 리소스 업데이트](#)
- [스택 정책 수정](#)
- [더 많은 스택 정책 예제](#)

스택 정책 예제

다음은 ProductionDatabase 리소스에 대한 업데이트를 방지하는 스택 정책 예제입니다.

```
{
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    },
    {
      "Effect" : "Deny",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "LogicalResourceId/ProductionDatabase"
    }
  ]
}
```

스택 정책을 설정한 경우 기본적으로 모든 리소스가 보호됩니다. 모든 리소스에 대한 업데이트를 허용하려면 모든 리소스에 대해 모든 작업을 허용하는 Allow 문을 추가합니다. Allow 문은 모든 리소스를 지정하지만 명시적 Deny 문은 ProductionDatabase 논리적 ID를 가진 리소스에 대해 해당 문을 무시합니다. 이 Deny 문은 ProductionDatabase 리소스에 대한 모든 업데이트 작업(예: 대체 또는 삭제)을 금지합니다.

Principal 요소는 필수이지만 와일드카드(*)만 지원합니다. 즉, 문이 모든 [보안 주체](#)에 적용됩니다.

Note

스택 업데이트 중에 CloudFormation에서는 다른 업데이트된 리소스에 종속되는 리소스를 자동으로 업데이트합니다. 예를 들어, CloudFormation에서는 업데이트된 리소스를 참조하는 리소스를 업데이트합니다. CloudFormation에서는 자동으로 업데이트된 리소스를 물리적으로 변

경하지 않지만(예: 리소스의 ID), 스택 정책이 해당 리소스와 연결되어 있는 경우 해당 리소스를 업데이트할 권한이 있어야 합니다.

스택 정책 정의

스택을 생성할 때에는 스택 정책이 설정되어 있지 않으므로 모든 리소스에 대한 모든 업데이트 작업이 허용됩니다. 스택 리소스를 업데이트 작업으로부터 보호하려면 스택 정책을 정의한 다음 스택에 대해 해당 정책을 설정합니다. 스택 정책은 CloudFormation 사용자가 수행할 수 있는 CloudFormation 스택 업데이트 작업과 작업이 적용되는 리소스를 정의하는 JSON 문서입니다. 스택을 생성할 때 스택 정책을 포함하는 텍스트 파일을 지정하거나 입력하여 스택 정책을 설정합니다. 스택에 대한 스택 정책을 설정하는 경우 명시적으로 허용되지 않는 업데이트는 기본적으로 거부됩니다.

Effect, Action, Principal, Resource, Condition의 다섯 가지 요소를 사용하여 스택 정책을 정의합니다. 다음은 의사(pseudo) 코드는 스택 정책 구문을 보여줍니다.

```
{
  "Statement" : [
    {
      "Effect" : "Deny_or-Allow",
      "Action" : "update_actions",
      "Principal" : "*",
      "Resource" : "LogicalResourceId/resource_logical_ID",
      "Condition" : {
        "StringEquals_or_StringLike" : {
          "ResourceType" : [resource_type, ...]
        }
      }
    }
  ]
}
```

Effect

지정한 리소스에 대해 지정한 작업을 거부할지 허용할지 여부를 결정합니다. Deny 또는 Allow만 지정할 수 있습니다(예:).

```
"Effect" : "Deny"
```

⚠ Important

스택 정책에 중복 문(리소스에 대한 업데이트 허용 및 거부)이 포함되어 있는 경우 항상 Deny 문이 Allow 문보다 우선합니다. 리소스를 보호하려면 해당 리소스에 대해 Deny 문을 사용합니다.

작업

거부되거나 허용되는 업데이트 작업을 지정합니다.

Update:Modify

변경 사항을 적용하는 동안 리소스 중단이 발생하지 않거나 일부 중단될 수 있는 업데이트 작업을 지정합니다. 모든 리소스는 물리적 ID를 유지합니다.

Update:Replace

도중에 리소스가 다시 생성되는 업데이트 작업을 지정합니다. CloudFormation에서는 지정된 업데이트를 사용하여 새 리소스를 생성한 다음에 이전 리소스를 삭제합니다. 리소스가 다시 생성되므로 새 리소스의 물리적 ID가 달라질 수 있습니다.

Update>Delete

도중에 리소스가 제거되는 업데이트 작업을 지정합니다. 스택 템플릿에서 리소스를 완전히 제거하는 업데이트의 경우 이 작업이 필요합니다.

업데이트:*

모든 업데이트 작업을 지정합니다. 별표는 모든 업데이트 작업을 나타내는 와일드카드입니다.

다음 예에서는 대체 및 삭제 작업을 지정하는 방법을 보여줍니다.

```
"Action" : ["Update:Replace", "Update>Delete"]
```

하나를 제외한 모든 업데이트 작업을 허용하려면 NotAction을 사용합니다. 예를 들어, Update>Delete를 제외한 모든 업데이트 작업을 허용하려면 이 예제와 같이 NotAction을 사용합니다.

```
{
  "Statement" : [
    {
      "Effect" : "Allow",
      "NotAction" : "Update>Delete",
```

```

    "Principal": "*",
    "Resource" : "*"
  }
]
}

```

위탁자

Principal 요소는 정책이 적용되는 개체를 지정합니다. 이 요소는 필수이지만 와일드카드(*)만 지원합니다. 즉, 정책이 모든 [보안 주체](#)에 적용됩니다.

리소스

정책이 적용되는 리소스의 논리적 ID를 지정합니다. 리소스 유형을 지정하려면 Condition 요소를 사용합니다.

단일 리소스를 지정하려면 논리적 ID를 사용합니다. 예제:

```
"Resource" : ["LogicalResourceId/myEC2instance"]
```

와일드카드를 논리적 ID와 함께 사용할 수 있습니다. 예를 들어, 모든 관련 리소스에 대해 일반 논리적 ID 접두사를 사용할 경우 와일드카드를 사용하여 모두 지정할 수 있습니다.

```
"Resource" : ["LogicalResourceId/CriticalResource*"]
```

Not 요소를 리소스와 함께 사용할 수도 있습니다. 예를 들어, 하나를 제외한 모든 리소스에 대한 업데이트를 허용하려면 NotResource 요소를 사용하여 해당 리소스를 보호합니다.

```

{
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "NotResource" : "LogicalResourceId/ProductionDatabase"
    }
  ]
}

```

스택 정책을 설정하는 경우 명시적으로 허용되지 않는 업데이트는 거부됩니다.

ProductionDatabase 리소스를 제외한 모든 리소스에 대한 업데이트를 허용하여 ProductionDatabase 리소스에 대한 업데이트를 거부합니다.

조건

정책이 적용되는 리소스 유형을 지정합니다. 특정 리소스의 논리적 ID를 지정하려면 Resource 요소를 사용합니다.

다음 예제와 같이 리소스 유형(예: 모든 EC2 및 RDS DB 인스턴스)을 지정할 수 있습니다.

```
{
  "Statement" : [
    {
      "Effect" : "Deny",
      "Principal" : "*",
      "Action" : "Update:*",
      "Resource" : "*",
      "Condition" : {
        "StringEquals" : {
          "ResourceType" : ["AWS::EC2::Instance", "AWS::RDS::DBInstance"]
        }
      }
    },
    {
      "Effect" : "Allow",
      "Principal" : "*",
      "Action" : "Update:*",
      "Resource" : "*"
    }
  ]
}
```

Allow 문은 모든 리소스에 대한 업데이트 권한을 부여하고 Deny 문은 EC2 및 RDS DB 인스턴스에 대한 업데이트를 거부합니다. Deny 문은 항상 허용 작업보다 우선합니다.

와일드카드를 리소스 유형과 함께 사용할 수 있습니다. 예를 들어, 다음 예제와 같이 와일드카드를 사용하여 모든 Amazon EC2 리소스(예: 인스턴스, 보안 그룹, 서브넷 등)에 대한 업데이트 권한을 거부할 수 있습니다.

```
"Condition" : {
  "StringLike" : {
    "ResourceType" : ["AWS::EC2::*"]
  }
}
```

와일드카드를 사용할 경우 StringLike 조건을 사용해야 합니다.

스택 정책 설정

스택을 생성할 때 콘솔 또는 AWS CLI를 사용하여 스택 정책을 적용할 수 있습니다. AWS CLI를 사용하여 기존 스택에 스택 정책을 적용할 수도 있습니다. 스택 정책을 적용한 이후에는 스택에서 스택 정책을 제거할 수 없지만, AWS CLI를 사용하여 스택 정책을 수정할 수 있습니다.

스택 정책은 스택을 업데이트하려 하는 모든 CloudFormation 사용자에게 적용됩니다. 다른 스택 정책을 다른 사용자와 연결할 수 없습니다.

스택 정책 작성에 대한 자세한 내용은 [스택 정책 정의](#) 단원을 참조하십시오.

스택을 생성할 때 스택 정책을 설정하려면(콘솔)

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 화면 상단의 탐색 모음에서 스택을 생성할 AWS 리전을 선택합니다.
3. CloudFormation Stacks(CloudFormation 스택) 페이지에서 스택 생성을 선택합니다.
4. 스택 생성 마법사의 Configure stack options(스택 옵션 구성) 페이지에서 고급 섹션을 확장하고 스택 정책을 선택하십시오.
5. 스택 정책을 지정합니다.
 - 콘솔에서 정책을 직접 작성하려면 Enter stack policy(스택 정책 입력)을 선택한 다음 텍스트 필드에 직접 스택 정책을 입력하십시오.
 - 별도의 파일에 정의된 정책을 사용하려면 파일 업로드를 선택한 다음 파일 선택을 선택하여 스택 정책이 포함된 파일을 선택하십시오.

스택을 생성할 때 스택 정책을 설정하려면(AWS CLI)

- [create-stack](#) 명령을 `--stack-policy-body` 옵션과 함께 사용하여 수정된 정책을 입력하거나 `--stack-policy-url` 옵션을 사용하여 정책이 포함된 파일을 지정합니다.

기존 스택에 대한 스택 정책을 설정하려면(AWS CLI 전용)

- [set-stack-policy](#) 명령을 `--stack-policy-body` 옵션과 함께 사용하여 수정된 정책을 입력하거나 `--stack-policy-url` 옵션을 사용하여 정책이 포함된 파일을 지정합니다.

Note

기존 스택에 정책을 추가하려면 CloudFormation [SetStackPolicy](#) 작업에 대한 권한이 있어야 합니다.

보호된 리소스 업데이트

보호된 리소스를 업데이트하려면 스택 정책을 재정의하고 해당 리소스에 대한 업데이트를 허용하는 임시 정책을 생성합니다. 스택을 업데이트할 때 재정의의 정책을 지정합니다. 재정의의 정책은 스택 정책을 영구히 변경하지 않습니다.

보호된 리소스를 업데이트하려면 CloudFormation [SetStackPolicy](#) 작업을 사용할 권한이 있어야 합니다. CloudFormation 권한 설정에 대한 자세한 내용은 [AWS Identity and Access Management를 사용하여 CloudFormation 액세스 제어](#) 섹션을 참조하세요.

Note

스택 업데이트 중에 CloudFormation에서는 다른 업데이트된 리소스에 종속되는 리소스를 자동으로 업데이트합니다. 예를 들어, CloudFormation에서는 업데이트된 리소스를 참조하는 리소스를 업데이트합니다. CloudFormation에서는 자동으로 업데이트된 리소스를 물리적으로 변경하지 않지만(예: 리소스의 ID), 스택 정책이 해당 리소스와 연결되어 있는 경우 해당 리소스를 업데이트할 권한이 있어야 합니다.

보호된 리소스를 업데이트하려면(콘솔)

1. CloudFormation 콘솔(<https://console.aws.amazon.com/cloudformation>)을 엽니다.
2. 업데이트할 스택을 선택하고 Stack actions(스택 작업), 스택 업데이트를 차례로 선택합니다.
3. 스택 템플릿을 수정하지 않은 경우, 현재 템플릿 사용을 선택하고 다음을 클릭합니다. 템플릿을 수정한 경우 Replace current template(현재 템플릿 교체)을 선택하고 Specify template(템플릿 지정) 섹션에서 업데이트된 템플릿의 위치를 지정하십시오.
 - 컴퓨터에 로컬로 저장된 템플릿의 경우 템플릿 파일 업로드를 선택합니다. 파일 선택을 선택하여 파일을 찾아서 선택한 다음 다음을 클릭합니다.
 - Amazon S3 버킷에 저장된 템플릿의 경우 Amazon S3URL을 선택하십시오. 템플릿에 대한 URL을 입력하거나 붙여넣고 다음을 클릭합니다.

템플릿이 버전 관리가 활성화된 버킷에 있는 경우 URL에 ?versionId=**version-id**를 추가하여 특정 템플릿 버전을 지정할 수 있습니다. 자세한 내용을 알아보려면 Amazon Simple Storage Service 사용 설명서의 [버전 관리를 사용하는 버킷의 객체 관리](#)를 참조하세요.

4. 템플릿에 파라미터가 포함되어 있는 경우 스택 세부 정보 지정 페이지에서 파라미터값을 입력하거나 수정하고 다음을 선택합니다.

CloudFormation에서 NoEcho 속성으로 선언된 파라미터를 제외하고 스택에 현재 설정된 값으로 각 파라미터를 채웁니다. 기존 값 사용을 선택하여 파라미터에 대한 현재 값을 사용할 수 있습니다.

동적 파라미터를 사용하여 보안 암호를 관리하는 방법 및 NoEcho를 사용하여 민감한 정보를 마스킹 처리하는 방법에 대한 자세한 내용은 [템플릿에 자격 증명을 포함하지 않음](#) 모범 사례를 참조하세요.

5. 스택 정책 재정의의 지정합니다.
 - a. Configure stack options(스택 옵션 구성) 페이지의 고급 옵션 섹션에서 스택 정책을 선택하십시오.
 - b. 파일 업로드를 선택하십시오.
 - c. 파일 선택을 클릭하고 스택 정책 재정의가 포함된 파일을 찾거나 정책을 입력하십시오.
 - d. 다음을 선택합니다.

재정의 정책에서는 업데이트하려는 보호된 리소스에 대해 Allow 문을 지정해야 합니다. 예를 들어, 모든 보호된 리소스를 업데이트하려면 모든 업데이트를 허용하는 임시 재정의 정책을 지정합니다.

```
{
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```


Note

CloudFormation에서는 이 업데이트 중에만 재정의 정책을 적용합니다. 재정의 정책은 스택 정책을 영구히 변경하지 않습니다. 스택 정책을 수정하려면 [스택 정책 수정](#) 단원을 참조하십시오.

6. 스택 정보와 제출한 변경 사항을 검토합니다.

올바른 정보(예: 올바른 파라미터값 또는 템플릿 URL)를 제출했는지 확인합니다. 템플릿에 IAM 리소스가 포함되어 있는 경우 이 템플릿이 IAM 리소스를 생성할 수 있음을 확인합니다(I acknowledge that this template may create IAM resources)를 선택하여 템플릿에서 IAM 리소스를 사용하도록 지정합니다. 자세한 내용은 [CloudFormation 템플릿에서 IAM 리소스 승인](#) 섹션을 참조하세요.

변경 미리 보기 섹션에서 CloudFormation에서 필요한 모든 변경을 수행하는지 확인합니다. 예를 들어, 추가, 제거 또는 수정하려는 리소스를 CloudFormation에서 추가, 제거 또는 수정하는지 확인합니다. CloudFormation에서는 스택에 대한 변경 세트를 만들어 이 미리 보기를 생성합니다. 자세한 내용은 [변경 세트를 사용하여 CloudFormation 스택 업데이트](#) 섹션을 참조하세요.

7. 변경 사항이 만족스러우면 업데이트를 클릭합니다.

Note

이때, 제안한 업데이트를 보다 철저하게 검토하기 위해 변경 세트를 볼 수도 있습니다. 이렇게 하려면 Update(업데이트) 대신 View change set(변경 세트 보기)를 클릭합니다. CloudFormation은 업데이트를 기반으로 생성된 변경 세트를 표시합니다. 스택 업데이트를 수행할 준비가 되면 실행을 클릭하십시오.

CloudFormation은 해당 스택의 스택 세부 정보 페이지를 표시합니다. 이제 스택의 상태는 UPDATE_IN_PROGRESS입니다. CloudFormation이 스택 업데이트를 성공적으로 완료하면 스택 상태가 UPDATE_COMPLETE로 설정됩니다.

스택 업데이트가 실패하면 CloudFormation에서 변경 사항을 자동으로 롤백하고 스택 상태를 UPDATE_ROLLBACK_COMPLETE로 설정합니다.

보호된 리소스 업데이트(AWS CLI)

- [update-stack](#) 명령을 `--stack-policy-during-update-body` 옵션과 함께 사용하여 수정된 정책을 입력하거나 `--stack-policy-during-update-url` 옵션을 사용하여 정책이 포함된 파일을 지정합니다.

Note

CloudFormation에서는 이 업데이트 중에만 재정의 정책을 적용합니다. 재정의 정책은 스택 정책을 영구히 변경하지 않습니다. 스택 정책을 수정하려면 [스택 정책 수정](#) 단원을 참조하십시오.

스택 정책 수정

추가 리소스를 보호하거나 리소스에서 보호를 제거하려면 스택 정책을 수정합니다. 예를 들어, 보호할 데이터베이스를 스택에 추가할 경우 데이터베이스에 대한 Deny 문을 스택 정책에 추가합니다. 정책을 수정하려면 [SetStackPolicy](#) 작업을 사용할 권한이 있어야 합니다.

AWS CLI를 사용하여 스택 정책을 수정합니다.

스택 정책 수정(AWS CLI)

- [set-stack-policy](#) 명령을 `--stack-policy-body` 옵션과 함께 사용하여 수정된 정책을 입력하거나 `--stack-policy-url` 옵션을 사용하여 정책이 포함된 파일을 지정합니다.

스택 정책을 삭제할 수 없습니다. 모든 리소스에서 모든 보호를 제거하려면 모든 리소스에 대해 모든 작업을 명시적으로 허용하도록 정책을 수정합니다. 다음 정책은 모든 리소스에 대한 모든 업데이트를 허용합니다.

```
{
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

```
}
```

더 많은 스택 정책 예제

다음 정책 예제에서는 모든 스택 리소스 및 특정 리소스에 대한 업데이트와 특정 유형의 업데이트를 방지하는 방법을 보여줍니다.

모든 스택 리소스에 대한 업데이트 방지

모든 스택 리소스에 대한 업데이트를 방지하기 위해 다음 정책에서는 모든 리소스에 관한 모든 업데이트 작업에 대해 Deny 문을 지정합니다.

```
{
  "Statement" : [
    {
      "Effect" : "Deny",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

단일 리소스에 대한 업데이트 방지

다음 정책은 논리적 ID가 MyDatabase인 데이터베이스에 대한 모든 업데이트 작업을 거부하고, Allow 문을 사용하여 나머지 모든 스택 리소스에 대한 모든 업데이트 작업을 허용합니다. Allow 문이 허용 작업보다 항상 우선하므로 MyDatabase 문은 Deny 리소스에 적용되지 않습니다.

```
{
  "Statement" : [
    {
      "Effect" : "Deny",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "LogicalResourceId/MyDatabase"
    },
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

```

    }
  ]
}
```

기본 거부를 사용하여 이전 예제와 동일한 결과를 얻을 수 있습니다. 스택 정책을 설정하는 경우 CloudFormation에서는 명시적으로 허용되지 않은 모든 업데이트를 거부합니다. 다음 정책은 기본적으로 거부되는 ProductionDatabase 리소스를 제외한 모든 리소스에 대한 업데이트를 허용합니다.

```

{
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "NotResource" : "LogicalResourceId/ProductionDatabase"
    }
  ]
}
```

Important

기본 거부를 사용할 경우 위험이 있습니다. 정책 내의 다른 위치에 Allow 문이 있는 경우(예: 와일드카드를 사용하는 Allow 문) 원치 않는 리소스에 대한 업데이트 권한을 자신도 모르게 허용할 수 있습니다. 명시적 거부가 허용 작업보다 우선하므로 Deny 문을 사용하여 리소스를 보호할 수 있습니다.

특정 리소스 유형의 모든 인스턴스에 대한 업데이트 방지

다음 정책은 RDS DB 인스턴스 리소스 유형에 대한 모든 업데이트 작업을 거부하고, Allow 문을 사용하여 나머지 모든 스택 리소스에 대한 모든 업데이트 작업을 허용합니다. Allow 문이 허용 작업보다 항상 우선하므로 Deny 문은 RDS DB 인스턴스 리소스에 적용되지 않습니다.

```

{
  "Statement" : [
    {
      "Effect" : "Deny",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*",
      "Condition" : {
```

```

        "StringEquals" : {
            "ResourceType" : ["AWS::RDS::DBInstance"]
        }
    },
    {
        "Effect" : "Allow",
        "Action" : "Update:*",
        "Principal": "*",
        "Resource" : "*"
    }
]
}

```

인스턴스에 대한 대체 업데이트 방지

다음 정책은 논리적 ID가 MyInstance인 인스턴스를 바꿔야 하는 업데이트를 거부하고, Allow 문을 사용하여 나머지 모든 스택 리소스에 대한 모든 업데이트 작업을 허용합니다. Allow 문이 허용 작업보다 항상 우선하므로 MyInstance 문은 Deny 리소스에 적용되지 않습니다.

```

{
    "Statement" : [
        {
            "Effect" : "Deny",
            "Action" : "Update:Replace",
            "Principal": "*",
            "Resource" : "LogicalResourceId/MyInstance"
        },
        {
            "Effect" : "Allow",
            "Action" : "Update:*",
            "Principal": "*",
            "Resource" : "*"
        }
    ]
}

```

중첩 스택에 대한 업데이트 방지

다음 정책은 CloudFormation 스택 리소스 유형(중첩 스택)에 대한 모든 업데이트 작업을 거부합니다. Allow 문을 사용하여 나머지 모든 스택 리소스에 대한 모든 업데이트 작업을 허용합니다. Allow 문이 허용 작업보다 항상 우선하므로 Deny 문은 CloudFormation 스택 리소스에 적용되지 않습니다.

```
{
  "Statement" : [
    {
      "Effect" : "Deny",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*",
      "Condition" : {
        "StringEquals" : {
          "ResourceType" : ["AWS::CloudFormation::Stack"]
        }
      }
    },
    {
      "Effect" : "Allow",
      "Action" : "Update:*",
      "Principal": "*",
      "Resource" : "*"
    }
  ]
}
```

CloudFormation의 데이터 보호

AWS [공동 책임 모델](#)은 AWS CloudFormation의 데이터 보호에 적용됩니다. 이 모델에서 설명하는 것처럼 AWS는 모든 AWS 클라우드를 실행하는 글로벌 인프라를 보호할 책임이 있습니다. 사용자는 인프라에서 호스팅되는 콘텐츠를 관리해야 합니다. 사용하는 AWS 서비스의 보안 구성과 관리 태스크에 대한 책임도 사용자에게 있습니다. 데이터 프라이버시에 대한 자세한 내용은 [데이터 프라이버시 FAQ](#)를 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은 AWS 보안 블로그의 [AWS 공동 책임 모델 및 GDPR](#) 블로그 게시물을 참조하세요.

데이터를 보호하려면 AWS 계정 자격 증명을 보호하고 AWS IAM Identity Center 또는 AWS Identity and Access Management(IAM)를 통해 개별 사용자 계정을 설정하는 것이 좋습니다. 이렇게 하면 개별 사용자에게 자신의 직무를 충실히 이행하는 데 필요한 권한만 부여됩니다. 또한 다음과 같은 방법으로 데이터를 보호하는 것이 좋습니다.

- 각 계정에 다중 인증(MFA)을 사용하세요.
- SSL/TLS를 사용하여 AWS 리소스와 통신하세요. TLS 1.2는 필수이며 TLS 1.3을 권장합니다.

- AWS CloudTrail로 API 및 사용자 활동 로깅을 설정하세요. AWS 활동 캡처에 CloudTrail 추적을 사용하는 방법에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 [CloudTrail 추적 작업을](#) 참조하세요.
- AWS 암호화 솔루션을 AWS 서비스 내의 모든 기본 보안 컨트롤과 함께 사용하세요.
- Amazon S3에 저장된 민감한 데이터를 검색하고 보호하는 데 도움이 되는 Amazon Macie와 같은 고급 관리형 보안 서비스를 사용하세요.
- 명령줄 인터페이스 또는 API를 통해 AWS에 액세스할 때 FIPS 140-3 검증된 암호화 모듈이 필요한 경우, FIPS 엔드포인트를 사용합니다. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 [Federal Information Processing Standard\(FIPS\) 140-3](#)을 참조하세요.

고객의 이메일 주소와 같은 기밀 정보나 중요한 정보는 태그나 이름 필드와 같은 자유 형식 텍스트 필드에 입력하지 않는 것이 좋습니다. 여기에는 CloudFormation 또는 기타 AWS 서비스에서 콘솔, API, AWS CLI 또는 AWS SDK를 사용하여 작업하는 경우가 포함됩니다. 이름에 사용되는 태그 또는 자유 형식 텍스트 필드에 입력하는 모든 데이터는 청구 또는 진단 로그에 사용될 수 있습니다. 외부 서버에 URL을 제공할 때 해당 서버에 대한 요청을 검증하기 위해 보안 인증 정보를 URL에 포함시켜서는 안 됩니다.

저장된 데이터 암호화

AWS 공동 책임 모델에 따라 CloudFormation에서는 암호화된 상태로 데이터를 저장합니다. 고객은 자신의 계정에 저장된 데이터에 대한 암호화 및 스토리지 정책을 설정할 책임이 있습니다. 예를 들어 S3 버킷 또는 SNS 주제에 저장된 템플릿 및 기타 데이터에 대해 저장 시 암호화를 활성화하는 것이 좋습니다. 고객은 CloudFormation에서 프로비저닝한 모든 데이터 스토리지 시스템에 대해서도 암호화 설정을 정의해야 합니다.

전송 중 암호화

CloudFormation에서는 공유 책임 모델에 따라 서비스 통신에 암호화된 채널을 사용합니다.

인터넷워크 트래픽 개인 정보

CloudFormation 서비스 통신은 기본적으로 리전 또는 가용 영역 간에 안전하게 암호화됩니다.

AWS Identity and Access Management를 사용하여 CloudFormation 액세스 제어

AWS Identity and Access Management(IAM)를 사용하여 IAM 사용자를 생성하고 AWS 계정에서 특정 리소스에 대한 해당 사용자의 액세스를 제어할 수 있습니다. IAM을 사용하는 경우 CloudFormation에서 수행할 수 있는 작업(예: 스택 템플릿 보기, 스택 생성 또는 스택 삭제)을 제어할 수 있습니다.

CloudFormation 특정 작업 외에도 각 사용자가 사용할 수 있는 AWS 서비스 및 리소스를 관리할 수 있습니다. 이렇게 하면 사용자가 CloudFormation을 사용할 때 액세스할 수 있는 리소스를 제어할 수 있습니다. 예를 들어, Amazon EC2 인스턴스를 생성하거나, 데이터베이스 인스턴스를 종료하거나, VPC를 업데이트할 수 있는 사용자를 지정할 수 있습니다. CloudFormation을 사용하여 이러한 작업을 수행할 경우 항상 동일한 권한이 적용됩니다.

다음 섹션의 정보를 사용하여 CloudFormation에 액세스할 수 있는 사용자를 제어합니다. 또한 템플릿에서 IAM 리소스 생성 권한을 부여하고, EC2 인스턴스에서 실행되는 애플리케이션에 필요한 권한을 부여하며, AWS 환경의 보안 강화를 위해 임시 보안 자격 증명을 사용하는 방법을 살펴봅니다.

CloudFormation에 대한 IAM ID 기반 정책 정의

CloudFormation에 대한 액세스 권한을 부여하려면 IAM ID(예: 사용자 또는 역할)에 필요한 API 작업을 직접 호출할 수 있는 권한을 부여하는 IAM 정책을 생성하고 할당해야 합니다.

IAM 자격 증명 기반 정책을 사용하면 허용되거나 거부되는 작업과 리소스는 물론, 작업이 허용되거나 거부되는 조건도 지정할 수 있습니다. CloudFormation은 특정 작업, 리소스 및 조건 키를 지원합니다.

IAM을 처음 사용하는 경우 먼저 IAM JSON 정책의 요소를 숙지합니다. 자세한 내용은 IAM 사용 설명서의 [IAM JSON 정책 요소 참조](#)를 참조하세요. IAM 정책을 생성하는 방법을 알아보려면 IAM 설명서의 튜토리얼 [첫 번째 고객 관리형 정책 만들기 및 연결](#)을 완료하세요.

주제

- [CloudFormation에 대한 정책 작업](#)
- [CloudFormation에 대한 콘솔 특정 작업](#)
- [CloudFormation에 대한 정책 리소스](#)
- [CloudFormation에 대한 정책 조건 키](#)

CloudFormation에 대한 정책 작업

IAM 정책 설명의 Action 요소에서 CloudFormation이 제공하는 모든 API 작업을 지정할 수 있습니다. 작업 이름에는 소문자 문자열 `cloudformation:`이 접두사로 포함되어야 합니다. 예를 들어, `cloudformation:CreateStack`, `cloudformation:CreateChangeSet` 및 `cloudformation:UpdateStack`입니다.

명령문 하나에 여러 태스크를 지정하려면 다음과 같이 쉼표로 구분합니다.

```
"Action": [ "cloudformation:action1", "cloudformation:action2" ]
```

와일드카드를 사용하여 여러 작업을 지정할 수도 있습니다. 예를 들어 다음과 같이 이름이 Get으로 시작되는 모든 작업을 지정할 수 있습니다.

```
"Action": "cloudformation:Get*"
```

`cloudformation` 서비스 접두사와 연결된 작업의 전체 목록을 보려면 서비스 승인 참조의 [Actions, resources, and condition keys for CloudFormation](#) 및 [Actions, resources, and condition keys for AWS Cloud Control API](#)를 참조하세요.

예제

다음에서는 CloudFormation 스택을 볼 수 있는 권한을 부여하는 권한 정책 예제를 보여줍니다.

Example 1: 스택 보기 권한을 부여하는 샘플 정책

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "cloudformation:DescribeStacks",
      "cloudformation:DescribeStackEvents",
      "cloudformation:DescribeStackResource",
      "cloudformation:DescribeStackResources"
    ],
    "Resource": "*"
  }]
}
```

스택을 생성하거나 삭제하는 사용자는 스택 템플릿을 기반으로 추가 권한이 필요합니다. 예를 들어 템플릿이 Amazon SQS 대기열을 설명하는 경우 다음 샘플 정책에 표시된 대로 사용자에게 CloudFormation 및 Amazon SQS 작업 모두에 대한 권한이 있어야 합니다.

Example 2: 스택 보기 및 생성 작업과 모든 Amazon SQS 작업을 허용하는 샘플 정책

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "sqs:*",
      "cloudformation:CreateStack",
      "cloudformation:DescribeStacks",
      "cloudformation:DescribeStackEvents",
      "cloudformation:DescribeStackResources",
      "cloudformation:GetTemplate",
      "cloudformation:ValidateTemplate"
    ],
    "Resource": "*"
  }]
}
```

CloudFormation에 대한 콘솔 특정 작업

CloudFormation 콘솔의 사용자는 AWS Command Line Interface 또는 CloudFormation API에 필요한 권한 이외의 추가 권한이 필요합니다. 이러한 추가 권한은 Amazon S3 버킷에 대한 템플릿 업로드 및 AWS 특정 파라미터 유형에 대한 드롭다운 목록과 같은 콘솔 특정 기능을 지원합니다.

아래 나열된 모든 작업의 경우 모든 리소스에 대한 권한을 부여합니다. 특정 스택이나 버킷으로 제한되지 않습니다.

다음 작업은 CloudFormation 콘솔에서만 사용되며 API 참조에 문서화되지 않습니다. 사용자가 작업을 통해 템플릿을 Amazon S3 버킷에 업로드할 수 있습니다.

- `cloudformation:CreateUploadBucket`

사용자가 템플릿을 업로드할 때 다음과 같은 Amazon S3 권한도 필요합니다.

- s3:PutObject
- s3:ListBucket
- s3:GetObject
- s3:CreateBucket

AWS 특정 파라미터 유형을 포함하는 템플릿의 파라미터 드롭다운 목록에서 값을 보려면 사용자에게 해당 Describe API 직접 호출을 수행할 수 있는 권한이 필요합니다. 예를 들어 템플릿에서 이러한 파라미터 유형을 사용하는 경우 다음 권한이 필요합니다.

- ec2:DescribeKeyPairs - AWS::EC2::KeyPair::KeyName 파라미터 유형에 필요합니다.
- ec2:DescribeSecurityGroups - AWS::EC2::SecurityGroup::Id 파라미터 유형에 필요합니다.
- ec2:DescribeSubnets - AWS::EC2::Subnet::Id 파라미터 유형에 필요합니다.
- ec2:DescribeVpcs - AWS::EC2::VPC::Id 파라미터 유형에 필요합니다.

AWS 특정 파라미터 유형에 대한 자세한 내용은 [CloudFormation에서 제공하는 파라미터 유형을 사용하여 런타임 시 기존 리소스 지정](#) 섹션을 참조하세요.

CloudFormation에 대한 정책 리소스

IAM 정책 구문에서 Resource 요소는 명령문이 다루는 하나 이상의 객체를 지정합니다.

CloudFormation에서 각 IAM 정책 설명은 Amazon 리소스 이름(ARN)을 사용하여 지정한 리소스에 적용됩니다. 구체적인 ARN 형식은 리소스에 따라 다릅니다.

CloudFormation 리소스 유형 및 해당 ARN의 전체 목록은 서비스 승인 참조의 [Resource types defined by CloudFormation](#)을 참조하세요. 각 리소스의 ARN을 지정할 수 있는 작업을 알아보려면 [Actions defined by CloudFormation](#)을 참조하세요.

다음 정책 예제에 표시된 대로 특정 스택에 대한 작업을 지정할 수 있습니다. ARN을 제공할 때 *placeholder text*를 리소스 특정 정보로 바꿉니다.

Example 1: 지정된 스택에 대한 스택 삭제 및 업데이트 작업을 거부하는 샘플 정책

JSON

```
{
```

```

    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Deny",
        "Action": [
          "cloudformation:DeleteStack",
          "cloudformation:UpdateStack"
        ],
        "Resource": "arn:aws:cloudformation:us-east-1:111122223333:stack/MyProductionStack/*"
      }
    ]
  }
}

```

위 정책에서는 스택 이름의 끝에 와일드카드를 사용하므로 전체 스택 ID(예: `arn:aws:cloudformation:region:account-id:stack/MyProductionStack/abc9dbf0-43c2-11e3-a6e8-50fa526be49c`) 및 스택 이름(예: `MyProductionStack`) 모두에서 스택 삭제 및 스택 업데이트 작업이 거부됩니다.

변경 세트를 만들도록 AWS::Serverless 변환을 허용하려면 다음 정책에 표시된 대로 `arn:aws:cloudformation:region:aws:transform/Serverless-2016-10-31` 리소스 수준 권한을 포함합니다.

Example 2: 지정된 변환에 대한 변경 세트 생성 작업을 허용하는 샘플 정책

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateChangeSet"
      ],
      "Resource": "arn:aws:cloudformation:us-east-1:aws:transform/Serverless-2016-10-31"
    }
  ]
}

```

CloudFormation에 대한 정책 조건 키

선택적으로 IAM 정책에서 정책이 적용되는 시점을 제어하는 조건을 지정할 수 있습니다. 예를 들어, 사용자가 특정 템플릿 URL을 지정하는 경우에만 스택 생성을 허용하는 정책을 정의할 수 있습니다. 정책 적용을 중지할 시점을 지정하는 CloudFormation 관련 조건 및 AWS 전체 조건(예: DateLessThan)을 정의할 수 있습니다. AWS 전체 조건에 대한 자세한 내용과 목록을 알아보려면 IAM 사용 설명서의 [IAM 정책 요소 참조](#)에서 조건을 참조하세요.

Note

`aws:SourceIp` AWS 전체 조건을 사용하지 마세요. CloudFormation에서는 원본 요청의 IP 주소가 아니라 자체 IP 주소를 사용하여 타겟 대상을 프로비저닝합니다. 예를 들어, 스택을 생성할 경우 CloudFormation에서는 `CreateStack` 호출 또는 `create-stack` 명령의 IP 주소가 아니라 자체 IP 주소에서 Amazon EC2 인스턴스를 시작하거나 Amazon S3 버킷을 생성하도록 요청합니다.

다음 목록에서는 CloudFormation 특정 조건에 대해 설명합니다. 이러한 조건은 사용자가 스택을 생성하거나 업데이트하는 경우에만 적용됩니다.

`cloudformation:ChangeSetName`

정책과 연결할 CloudFormation 변경 세트 이름입니다. 이 조건을 사용하여 사용자가 실행하거나 삭제할 수 있는 변경 세트를 제어합니다.

`cloudformation:ImportResourceTypes`

정책과 연결할 템플릿 리소스 유형(예: `AWS::EC2::Instance`)입니다. 이 조건을 사용하여 사용자가 스택에 리소스를 가져올 때 사용할 수 있는 리소스 유형을 제어합니다. 사용자가 `ResourcesToImport` 파라미터에서 선언하는 리소스 유형에 대해 이 조건을 확인합니다. 이 조건은 현재 AWS CLI 및 API 요청에 대해서만 지원됩니다. 이 파라미터를 사용하는 경우 가져오기 작업 중에 사용자가 제어할 모든 리소스 유형을 지정해야 합니다. `ResourcesToImport` 파라미터에 대한 자세한 내용은 AWS CloudFormation API 참조의 [CreateChangeSet](#) 작업을 참조하세요.

가능한 `ResourcesToImport` 목록은 [리소스 유형 지원](#) 섹션을 참조하세요.

세 부분으로 구성된 리소스 명명 규칙을 사용하면 조직 전체의 모든 리소스에서 개별 리소스 유형까지 사용자가 작업할 수 있는 리소스 유형을 지정할 수 있습니다.

`organization::*`

지정된 조직에 대한 모든 리소스 유형을 지정합니다.

*organization::service_name::**

지정된 조직 내에 지정된 서비스에 대한 모든 리소스 유형을 지정합니다.

organization::service_name::resource_type

특정 리소스 유형을 지정합니다.

예제:

AWS::*

지원되는 모든 AWS 리소스 유형을 지정합니다.

AWS::*service_name*::*

특정 AWS 서비스에 대한 모든 리소스를 지정합니다.

AWS::*service_name*::*resource_type*

특정 AWS 리소스 유형(예: AWS::EC2::Instance[모든 EC2 인스턴스])을 지정합니다.

cloudformation:ResourceTypes

정책과 연결할 템플릿 리소스 유형(예: AWS::EC2::Instance)입니다. 이 조건을 사용하여 사용자가 스택을 생성하거나 업데이트할 때 작업할 수 있는 리소스 유형을 제어합니다. 사용자가 ResourceTypes 파라미터에서 선언하는 리소스 유형에 대해 이 조건을 확인합니다. 이 조건은 현재 AWS CLI 및 API 요청에 대해서만 지원됩니다. 이 파라미터를 사용할 경우 사용자가 템플릿에 있는 모든 리소스 유형을 지정해야 합니다. ResourceTypes 파라미터에 대한 자세한 내용은 AWS CloudFormation API 참조의 [CreateStack](#) 작업을 참조하세요.

리소스 유형 목록은 [CloudFormation 템플릿 참조 가이드](#)를 참조하세요.

세 부분으로 구성된 리소스 명명 규칙을 사용하면 조직 전체의 모든 리소스에서 개별 리소스 유형까지 사용자가 작업할 수 있는 리소스 유형을 지정할 수 있습니다.

organization::*

지정된 조직에 대한 모든 리소스 유형을 지정합니다.

organization::service_name::*

지정된 조직 내에 지정된 서비스에 대한 모든 리소스 유형을 지정합니다.

organization::service_name::resource_type

특정 리소스 유형을 지정합니다.

예제:

`AWS::*`

지원되는 모든 AWS 리소스 유형을 지정합니다.

`AWS::service_name::*`

특정 AWS 서비스에 대한 모든 리소스를 지정합니다.

`AWS::service_name::resource_type`

특정 AWS 리소스 유형(예: `AWS::EC2::Instance`[모든 EC2 인스턴스])을 지정합니다.

`Alexa::ASK::*`

Alexa Skill Kit의 모든 리소스 유형을 지정합니다.

`Alexa::ASK::Skill`

개별 [Alexa::ASK::Skill](#) 리소스 유형을 지정합니다.

`Custom::*`

모든 사용자 지정 리소스를 지정합니다.

자세한 내용은 [사용자 지정 리소스를 사용하여 사용자 지정 프로비저닝 로직 생성](#) 섹션을 참조하세요.

`Custom::resource_type`

특정 사용자 정의 리소스 유형을 지정합니다.

자세한 내용은 [사용자 지정 리소스를 사용하여 사용자 지정 프로비저닝 로직 생성](#) 섹션을 참조하세요.

`cloudformation:RoleARN`

정책과 연결할 IAM 서비스 역할의 Amazon 리소스 이름(ARN)입니다. 이 조건을 사용하여 사용자가 스택 또는 변경 세트로 작업할 때 사용할 있는 서비스 역할을 제어합니다.

`cloudformation:StackPolicyUrl`

정책과 연결할 Amazon S3 스택 정책 URL입니다. 이 조건을 사용하여 사용자가 스택 생성 또는 업데이트 작업 중에 스택과 연결할 수 있는 스택 정책을 제어합니다. 스택 정책에 대한 자세한 내용은 [스택 리소스에 대한 업데이트 방지](#) 단원을 참조하십시오.

Note

사용자가 업로드된 스택 정책을 통해서만 스택을 생성하거나 업데이트할 수 있도록 하려면 해당 사용자에게 대해 S3 버킷을 읽기 전용으로 설정합니다.

cloudformation:TemplateUrl

정책과 연결할 Amazon S3 템플릿 URL입니다. 이 조건을 사용하여 사용자가 스택을 생성하거나 업데이트할 때 사용할 수 있는 템플릿을 제어합니다.

Note

사용자가 업로드된 템플릿을 통해서만 스택을 생성하거나 업데이트할 수 있도록 하려면 해당 사용자에게 대해 S3 버킷을 읽기 전용으로 설정합니다.

Note

동일한 이름의 API 파라미터에 다음과 같은 CloudFormation 관련 조건이 적용됩니다.

- `cloudformation:ChangeSetName`
- `cloudformation:RoleARN`
- `cloudformation:StackPolicyUrl`
- `cloudformation:TemplateUrl`

예를 들어 `cloudformation:TemplateUrl`은 `CreateStack`, `UpdateStack` 및 `CreateChangeSet` API의 `TemplateUrl` 파라미터에만 적용됩니다.

조건 키를 사용하여 액세스를 제어하는 IAM 정책 예제는 [CloudFormation에 대한 IAM ID 기반 정책 예제](#) 섹션을 참조하세요.

CloudFormation 템플릿에서 IAM 리소스 승인

스택을 생성하기 전에 CloudFormation에서 템플릿 유효성을 검사합니다. 유효성 검사 도중 CloudFormation은 템플릿에서 생성할 수 있는 IAM 리소스를 확인합니다. 전체 액세스 권한이 있는 사용자와 같은 IAM 리소스는 AWS 계정의 모든 리소스에 액세스하고 수정할 수 있습니다. 따라서 에스컬레이션된 권한을 가진 리소스를 실수로 생성하는 일이 없도록 계속하기 전에 각 IAM 리소스에 연결된

권한을 검토하는 것이 좋습니다. 모든 과정을 마쳤음을 알 수 있도록 CloudFormation에서 스택을 생성하기 전에 지정된 기능을 제공하여 템플릿에 이러한 리소스가 포함되어 있음을 승인해야 합니다.

CloudFormation 콘솔, AWS Command Line Interface(AWS CLI) 또는 API를 사용하여 CloudFormation 템플릿의 기능을 승인할 수 있습니다.

- CloudFormation 콘솔에서는 스택 생성 마법사 또는 스택 업데이트 마법사의 스택 옵션 구성 페이지에서 이 템플릿이 IAM 리소스를 생성할 수 있음을 확인합니다를 선택합니다.
- AWS CLI에서는 [create-stack](#) 및 [update-stack](#) 명령을 사용할 경우 `--capabilities` 옵션에 대해 `CAPABILITY_IAM` 또는 `CAPABILITY_NAMED_IAM` 값을 지정합니다. 템플릿에 IAM 리소스가 포함되어 있는 경우 어느 기능이든 지정할 수 있습니다. 템플릿에 IAM 리소스에 대한 사용자 지정 이름이 포함되어 있는 경우 `CAPABILITY_NAMED_IAM`을 지정해야 합니다.
- API에서 [CreateStack](#) 및 [UpdateStack](#) 작업을 사용하는 경우 `Capabilities.member.1=CAPABILITY_IAM` 또는 `Capabilities.member.1=CAPABILITY_NAMED_IAM`을 지정합니다. 템플릿에 IAM 리소스가 포함되어 있는 경우 어느 기능이든 지정할 수 있습니다. 템플릿에 IAM 리소스에 대한 사용자 지정 이름이 포함되어 있는 경우 `CAPABILITY_NAMED_IAM`을 지정해야 합니다.

Important

템플릿에 사용자 지정 이름의 IAM 리소스가 포함되어 있는 경우 동일한 템플릿을 재사용하여 여러 스택을 생성하지 마십시오. IAM 리소스는 계정 내에서 전역적으로 고유해야 합니다. 동일한 템플릿을 사용하여 다양한 리전에서 여러 스택을 생성할 경우 각 스택이 고유한 리소스를 갖지 않고 동일한 IAM 리소스를 공유할 수 있습니다. 스택 간에 리소스를 공유하면 복구할 수 없는 의도하지 않은 결과가 발생할 수 있습니다. 예를 들어, 한 스택에서 공유되는 IAM 리소스를 삭제하거나 업데이트할 경우 다른 스택의 리소스를 의도치 않게 수정하게 됩니다.

Amazon EC2 인스턴스에서 실행되는 애플리케이션의 자격 증명 관리

Amazon EC2 인스턴스에서 실행하고 Amazon S3 버킷, DynamoDB 테이블 등과 같은 AWS 리소스에 요청하는 데 필요한 애플리케이션이 있는 경우 해당 애플리케이션에 AWS 보안 자격 증명이 필요합니다. 그러나 시작하는 모든 인스턴스에 장기 보안 자격 증명을 배포하여 포함하는 것은 어려울 뿐만 아니라 보안 위험이 발생할 수 있습니다. IAM 사용자 자격 증명과 같은 장기 자격 증명을 사용하는 대신 인스턴스를 시작할 때 Amazon EC2 인스턴스와 연결되는 IAM 역할을 생성하는 것이 좋습니다. 그러면 애플리케이션에서 Amazon EC2 인스턴스로부터 임시 보안 자격 증명을 가져올 수 있습니다. 따라서 인스턴스에 장기 자격 증명을 포함할 필요가 없습니다. 또한 자격 증명을 더 쉽게 관리하려면 여러

Amazon EC2 인스턴스에 대해 단일 역할을 지정할 수 있습니다. 그러면 인스턴스별로 고유한 자격 증명을 생성할 필요가 없습니다.

역할을 사용하여 인스턴스를 시작하는 방법을 보여주는 템플릿 조각은 [IAM 역할 템플릿의 예](#) 단원을 참조하십시오.

Note

임시 보안 자격 증명을 사용하는 인스턴스 기반 애플리케이션은 모든 CloudFormation 작업을 호출할 수 있습니다. 하지만 CloudFormation은 많은 다른 AWS 서비스와 상호 작용하므로 사용하려는 모든 서비스에서 임시 보안 자격 증명을 지원하는지 확인해야 합니다. 임시 보안 인증 정보를 지원하는 서비스 목록은 IAM 사용 설명서의 [IAM으로 작업하는 AWS](#)를 참조하세요.

임시 액세스 권한 부여(페더레이션된 액세스)

경우에 따라 AWS 자격 증명이 없는 사용자에게 AWS 계정에 대한 임시 액세스 권한을 부여할 수 있습니다. 임시 액세스 권한을 부여할 때마다 장기 자격 증명을 생성하고 삭제하는 대신 AWS Security Token Service(AWS STS)를 사용합니다. 예를 들어, IAM 역할을 사용할 수 있습니다. 한 IAM 역할에서 액세스 키, 보안 액세스 키 및 보안 토큰을 포함하는 다양한 임시 보안 자격 증명을 프로그래밍 방식으로 생성한 다음 배포할 수 있습니다. 이러한 자격 증명은 수명이 제한적이므로 만료된 이후에는 AWS 계정에 액세스하는 데 사용할 수 없습니다. 개별 사용자에게 다양한 수준의 권한을 부여하기 위해 여러 IAM 역할을 생성할 수도 있습니다. IAM 역할은 페더레이션형 ID와 Single Sign-On과 같은 시나리오에 유용합니다.

연합 ID는 여러 시스템에 걸쳐 사용할 수 있는 개별 자격 증명입니다. 설정된 온프레미스 자격 증명 시스템(예: LDAP 또는 Active Directory)이 있는 엔터프라이즈 사용자의 경우 해당 시스템을 사용하여 모든 인증을 처리할 수 있습니다. 사용자가 인증된 후 해당 IAM 사용자 또는 역할의 임시 보안 자격 증명을 제공합니다. 예를 들어, administrators 역할과 developers 역할을 생성할 수 있습니다. 여기서 administrators 역할은 AWS 계정에 대한 모든 권한을 가지며 developers 역할은 CloudFormation 스택 작업 권한만 가집니다. 관리자는 인증을 받은 후 administrators 역할에서 임시 보안 자격 증명을 가져올 수 있는 권한이 부여됩니다. 하지만 개발자는 developers 역할에서만 임시 보안 자격 증명을 가져올 수 있습니다.

연합된 사용자에게 AWS Management Console에 대한 액세스 권한을 부여할 수도 있습니다. 사용자가 온프레미스 자격 증명 시스템을 통해 인증한 후 AWS Management Console에 대한 직접 액세스 권한을 부여하는 임시 URL을 프로그래밍 방식으로 구성할 수 있습니다. 임시 URL을 사용하는 경우에는 이미 인증되었으므로(Single Sign-On) AWS에 로그인할 필요가 없습니다. 또한 URL은 사용자의

임시 보안 자격 증명으로 구성되므로 이러한 자격 증명으로 사용 가능한 권한에 따라 사용자가 AWS Management Console에서 보유하는 권한이 결정됩니다.

여러 AWS STS API를 사용하여 임시 보안 자격 증명을 생성할 수 있습니다. 사용할 API에 대한 자세한 내용은 IAM 사용 설명서의 [AWS STS 자격 증명 비교](#)를 참조하세요.

Important

GetFederationToken API에서 생성된 임시 보안 자격 증명을 사용하는 경우 IAM으로 작업할 수 없습니다. IAM으로 작업해야 하는 경우 특정 역할의 임시 보안 자격 증명을 대신 사용합니다.

CloudFormation은 다양한 다른 AWS 서비스와 상호 작용합니다. CloudFormation에서 임시 보안 자격 증명을 사용할 경우 사용하려는 모든 서비스에서 임시 보안 자격 증명을 지원하는지 확인합니다. 임시 보안 인증 정보를 지원하는 서비스 목록은 IAM 사용 설명서의 [IAM으로 작업하는 AWS](#)를 참조하세요.

자세한 내용은 IAM 사용 설명서의 관련 리소스를 참조하세요.

- [임시 자격 증명과 관련된 일반적인 시나리오](#)
- [AWS 콘솔에 대한 사용자 지정 ID 브로커 액세스 활성화](#)

CloudFormation에 대한 IAM ID 기반 정책 예제

기본적으로 사용자 및 역할에는 CloudFormation 리소스를 생성하거나 수정할 수 있는 권한이 없습니다. 또한 AWS Management Console, AWS Command Line Interface(AWS CLI) 또는 AWS API를 사용해 태스크를 수행할 수 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다. 그런 다음 관리자가 IAM 정책을 역할에 추가하고, 사용자가 역할을 맡을 수 있습니다. 자세한 내용은 [CloudFormation에 대한 IAM ID 기반 정책 정의](#) 섹션을 참조하세요.

다음 예제에서는 하나 이상의 CloudFormation 작업을 사용하는 권한을 허용하거나 거부하는 데 사용할 수 있는 정책 설명을 보여줍니다.

주제

- [특정 템플릿 URL 요구](#)
- [모든 CloudFormation 가져오기 작업 거부](#)
- [특정 리소스 유형에 대한 가져오기 작업 허용](#)

- [스택 템플릿에서 IAM 리소스 거부](#)
- [특정 리소스 유형으로 스택 생성 허용](#)
- [리소스 변경 API 작업을 기반으로 액세스 제어](#)
- [리전 및 리소스 유형을 기반으로 스택 세트 작업 제한](#)
- [모든 IaC 생성기 작업 허용](#)

특정 템플릿 URL 요구

다음 정책에서는 <https://s3.amazonaws.com/amzn-s3-demo-bucket/test.template> 템플릿 URL을 통해서만 스택을 생성하거나 업데이트하는 권한을 부여합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStack",
        "cloudformation:UpdateStack"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "cloudformation:TemplateUrl": [
            "https://s3.amazonaws.com/amzn-s3-demo-bucket/
test.template"
          ]
        }
      }
    }
  ]
}
```

모든 CloudFormation 가져오기 작업 거부

다음 정책에서는 가져오기 작업을 제외한 모든 CloudFormation 작업을 완료하는 권한을 부여합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAllStackOperations",
      "Effect": "Allow",
      "Action": "cloudformation:*",
      "Resource": "*"
    },
    {
      "Sid": "DenyImport",
      "Effect": "Deny",
      "Action": "cloudformation:*",
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringLike": {
          "cloudformation:ImportResourceTypes": [
            "*"
          ]
        }
      }
    }
  ]
}
```

특정 리소스 유형에 대한 가져오기 작업 허용

다음 정책에서는 오직 지정된 리소스에서의 가져오기 작업(이 예제에서 `AWS::S3::Bucket`)을 비롯해 모든 스택 작업에 대한 권한을 부여합니다.

JSON

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

```

    "Action": "cloudformation:*",
    "Resource": "*",
    "Condition": {
      "ForAllValues:StringEqualsIgnoreCase": {
        "cloudformation:ImportResourceTypes": [
          "AWS::S3::Bucket"
        ]
      }
    }
  }
]
}

```

스택 템플릿에서 IAM 리소스 거부

다음 정책에서는 스택 생성 권한을 부여하지만, 스택의 템플릿에 IAM 서비스의 리소스가 포함된 경우 요청을 거부합니다. 또한 정책을 통해 사용자에게 ResourceTypes 파라미터를 지정하도록 요구합니다. 이 파라미터는 AWS CLI 및 API 요청에만 사용할 수 있습니다. 이 정책은 명시적 거부문을 사용하므로 다른 정책에서 추가 권한을 부여하더라도 이 정책은 항상 적용됩니다(명시적 거부문이 명시적 허용문보다 항상 우선).

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [ "cloudformation:CreateStack" ],
      "Resource": "*"
    },
    {
      "Effect": "Deny",
      "Action": [ "cloudformation:CreateStack" ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringLikeIfExists" : {
          "cloudformation:ResourceTypes" : [ "AWS::IAM::*" ]
        }
      }
    }
  ],
}

```

```
{
  "Effect": "Deny",
  "Action" : [ "cloudformation:CreateStack" ],
  "Resource": "*",
  "Condition": {
    "Null": {
      "cloudformation:ResourceTypes": "true"
    }
  }
}
```

특정 리소스 유형으로 스택 생성 허용

다음 정책은 이전 예제와 비슷합니다. 이 정책에서는 스택의 템플릿에 IAM 서비스의 리소스가 포함되지 않은 경우 스택을 생성하는 권한을 부여합니다. 또한 사용자에게 ResourceTypes 파라미터를 지정하도록 요구합니다. 이 파라미터는 AWS CLI 및 API 요청에 대해서만 사용할 수 있습니다. 이 정책은 더 간단하지만 명시적 거부문을 사용하지 않습니다. 추가적인 권한을 부여하는 다른 정책에서 이 정책을 재정의할 수 있습니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect" : "Allow",
      "Action" : [ "cloudformation:CreateStack" ],
      "Resource" : "*",
      "Condition" : {
        "ForAllValues:StringNotLikeIfExists" : {
          "cloudformation:ResourceTypes" : [ "AWS::IAM::*" ]
        },
        "Null": {
          "cloudformation:ResourceTypes": "false"
        }
      }
    }
  ]
}
```

}

리소스 변경 API 작업을 기반으로 액세스 제어

다음 정책에서는 리소스 변경 API 작업의 이름을 기준으로 액세스를 필터링하는 권한을 부여합니다. 이 정책은 IAM 사용자가 스택 또는 스택 세트에서 태그를 추가하거나 제거하는 데 사용할 수 있는 API를 제어하는 데 사용됩니다. 태그를 추가하거나 제거하는 데 사용되는 작업은 조건 키의 값으로 추가해야 합니다. 다음 정책은 변경 작업 CreateStack에 TagResource 및 UntagResource 권한을 부여합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "CreateActionConditionPolicyForTagUntagResources",
    "Effect": "Allow",
    "Action": [
      "cloudformation:TagResource",
      "cloudformation:UntagResource"
    ],
    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "cloudformation:CreateAction": [
          "CreateStack"
        ]
      }
    }
  ]
}
```

리전 및 리소스 유형을 기반으로 스택 세트 작업 제한

다음 정책에서는 서비스 관리형 스택 세트 권한을 부여합니다. 이 정책을 적용 받는 사용자는 Amazon S3 리소스 유형(AWS::S3::*) 또는 AWS::SES::ConfigurationSet 리소스 유형을 포함하는 템플릿으로만 스택 세트 작업을 수행할 수 있습니다. ID 123456789012로 조직 관리 계정에 로그인한 경우 사용자는 ID가 *ou-1fsfsrsdsfrewr*인 OU를 대상으로 하는 스택 세트에 대해서만 작업을 수행할

수 있고, ID가 **987654321012**인 AWS 계정을 대상으로 하는 ID `stack-set-id`의 스택 세트에 대해서만 작업을 수행할 수 있습니다.

스택 세트 템플릿에 정책에 지정된 리소스 유형이 아닌 다른 리소스 유형이 포함되어 있거나, 배포 대상이 해당 관리 계정 및 스택 세트 정책에 지정되지 않은 OU 또는 계정 ID인 경우 스택 세트 작업이 실패합니다.

이러한 정책 제한은 스택 세트 작업이 `us-east-1`, `us-west-2` 또는 `eu-west-2` AWS 리전을 대상으로 하는 경우에만 적용됩니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:*"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stackset/*",
        "arn:aws:cloudformation:*:*:type/resource/AWS-S3-*",
        "arn:aws:cloudformation:us-west-2:111122223333:type/resource/AWS-SES-ConfigurationSet",
        "arn:aws:cloudformation:*:111122223333:stackset-target/*/ou-1fsfsrdsdfrewr",
        "arn:aws:cloudformation:*:111122223333:stackset-target/stack-set-id/444455556666"
      ],
      "Condition": {
        "ForAllValues:StringEqualsIgnoreCase": {
          "cloudformation:TargetRegion": [
            "us-east-1",
            "us-west-2",
            "eu-west-1"
          ]
        }
      }
    }
  ]
}
```

모든 IaC 생성기 작업 허용

다음 정책에서는 IaC 생성기 리소스 스캔 및 템플릿 관리와 관련된 CloudFormation 작업에 대한 액세스를 허용합니다. 첫 번째 명령문은 리소스 스캔을 설명, 나열 및 시작하는 권한을 부여합니다. 또한 IaC 생성기가 리소스 및 사용 가능한 리소스 유형에 대한 정보를 검색할 수 있는 추가 필수 권한(cloudformation:GetResource, cloudformation:ListResources, cloudformation:ListTypes)에 대한 액세스를 허용합니다. 두 번째 명령문은 생성된 템플릿을 생성, 삭제, 설명, 나열 및 업데이트할 수 있는 모든 권한을 부여합니다.

또한 IaC 생성기로 리소스를 스캔할 모든 사용자에게 대상 AWS 서비스에 대한 읽기 권한을 부여해야 합니다. 자세한 내용은 [리소스 스캔에 필요한 IAM 권한](#) 섹션을 참조하세요.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ResourceScanningOperations",
      "Effect": "Allow",
      "Action": [
        "cloudformation:DescribeResourceScan",
        "cloudformation:GetResource",
        "cloudformation:ListResources",
        "cloudformation:ListResourceScanRelatedResources",
        "cloudformation:ListResourceScanResources",
        "cloudformation:ListResourceScans",
        "cloudformation:ListTypes",
        "cloudformation:StartResourceScan"
      ],
      "Resource": "*"
    },
    {
      "Sid": "TemplateGeneration",
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateGeneratedTemplate",
        "cloudformation>DeleteGeneratedTemplate",
        "cloudformation:DescribeGeneratedTemplate",
        "cloudformation:GetResource",
        "cloudformation:GetGeneratedTemplate",
        "cloudformation:ListGeneratedTemplates",

```

```

        "cloudformation:UpdateGeneratedTemplate"
      ],
      "Resource": "*"
    }
  ]
}

```

AWS CloudFormation의 AWS 관리형 정책

AWS 관리형 정책은 AWS에 의해 생성되고 관리되는 독립 실행형 정책입니다. AWS 관리형 정책은 사용자, 그룹 및 역할에 권한 할당을 시작할 수 있도록 많은 일반 사용 사례에 대한 권한을 제공하도록 설계되었습니다.

AWS 관리형 정책은 모든 AWS 고객이 사용할 수 있기 때문에 특정 사용 사례에 대해 최소 권한을 부여하지 않을 수 있습니다. 사용 사례에 고유한 [고객 관리형 정책](#)을 정의하여 권한을 줄이는 것이 좋습니다.

AWS 관리형 정책에서 정의한 권한은 변경할 수 없습니다. AWS에서 AWS 관리형 정책에 정의된 권한을 업데이트할 경우, 정책이 연결되어 있는 모든 보안 주체 엔터티(사용자, 그룹 및 역할)에도 업데이트가 적용됩니다. 새로운 AWS 서비스를 시작하거나 새로운 API 작업을 기존 서비스에 이용하는 경우, AWS가 AWS 관리형 정책을 업데이트할 가능성이 높습니다.

자세한 내용은 IAM 사용자 가이드의 [AWS 관리형 정책](#)을 참조하세요.

AWS 관리형 정책: AWSCloudFormationFullAccess

사용자, 그룹 및 역할에 AWSCloudFormationFullAccess를 연결할 수 있습니다.

이 정책은 CloudFormation 작업 및 리소스에 대한 전체 액세스를 허용하는 권한을 부여합니다.

권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- `cloudformation` - 위탁자가 모든 리소스에서 모든 CloudFormation 작업을 수행하도록 허용합니다.

이 정책의 권한을 보려면 AWS 관리형 정책 참조 가이드에서 [AWSCloudFormationFullAccess](#)를 참조하세요.

AWS 관리형 정책: AWSCloudFormationReadOnlyAccess

사용자, 그룹 및 역할에 AWSCloudFormationReadOnlyAccess를 연결할 수 있습니다.

이 정책은 CloudFormation 리소스 및 작업에 대한 읽기 전용 액세스를 허용하는 권한을 부여합니다.

권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- `cloudformation` - 위탁자가 스택 설명, 리소스 나열, 템플릿 보기와 같은 읽기 전용 CloudFormation 작업을 수행할 수 있도록 허용하지만 스택 생성, 업데이트 또는 삭제는 허용하지 않습니다.

이 정책의 권한을 보려면 AWS 관리형 정책 참조 가이드에서 [AWSCloudFormationReadOnlyAccess](#)를 참조하시기 바랍니다.

CloudFormation AWS 관리형 정책 업데이트 사항

이 서비스에서 AWS 관리형 정책의 변경 사항을 추적하기 시작한 이후부터의 업데이트 세부 정보를 확인하세요. 이 페이지의 변경 사항에 관한 자동 알림을 받으려면 CloudFormation 문서 기록 페이지에서 RSS 피드를 구독하세요.

변경	설명	날짜
AWSCloudFormationReadOnlyAccess – 기존 정책에 업데이트	CloudFormation에서 일괄 설명 작업을 허용하기 위해 <code>cloudformation:BatchDescribe*</code> 작업을 허용하는 새 권한을 추가했습니다.	2026년 1월 30일
AWSCloudFormationReadOnlyAccess – 기존 정책에 업데이트	CloudFormation에서 스택 드리프트 감지 기능을 지원하기 위해 <code>cloudformation:Detect*</code> 작업을 허용하는 새로운 권한을 추가했습니다.	2019년 11월 13일
AWSCloudFormationReadOnlyAccess – 기존 정책에 업데이트	CloudFormation에서 <code>cloudformation:EstimateTemplateCost</code> ,	2017년 11월 2일

변경	설명	날짜
	cloudformation:Get * , cloudformation:Val idateTemplate 작업을 허 용하는 새 권한을 추가했습니 다.	
AWSCloudFormationFullAccess – 새 정책	CloudFormation에서 CloudFormation 작업 및 리소 스에 대한 전체 액세스를 제공 하는 새로운 AWS 관리형 정책 을 추가했습니다.	2019년 7월 26일
AWSCloudFormationReadOnlyAccess – 기존 정책에 업데이트	CloudFormation에서 스택 드리 프트 감지를 위한 cloudform ation:DetectStackD rift 및 cloudform ation:DetectStackR esourceDrift 작업을 허 용하는 새 권한을 추가했습니 다.	2019년 2월 6일
AWSCloudFormationReadOnlyAccess - 새 정책	CloudFormation에서 CloudFormation 작업 및 리소 스에 대한 읽기 전용 액세스를 제공하는 새로운 AWS 관리형 정책을 추가했습니다.	2015년 2월 6일
CloudFormation에서 변경 내용 을 추적하기 시작했습니다	CloudFormation에서 AWS 관 리형 정책에 대한 변경 내용 추 적을 시작했습니다.	2015년 2월 6일

CloudFormation 서비스 역할

서비스 역할은 CloudFormation에서 사용자를 대신하여 스택의 리소스를 호출하도록 허용하는 AWS Identity and Access Management(IAM) 역할입니다. CloudFormation에서 스택 리소스를 생성, 업데이트

트 또는 삭제하도록 허용하는 IAM 역할을 지정할 수 있습니다. 기본적으로 CloudFormation에서는 스택 작업에 대한 사용자 자격 증명으로부터 생성되는 임시 세션을 사용합니다. 서비스 역할을 지정한 경우 CloudFormation에서는 역할의 자격 증명을 사용합니다.

서비스 역할을 사용하여 CloudFormation에서 수행할 수 있는 작업을 명시적으로 지정합니다. 이 작업은 사용자 또는 다른 사용자가 수행할 수 있는 작업과 다를 수도 있습니다. 예를 들어, 관리 권한이 있지만 CloudFormation 액세스를 Amazon EC2 작업만으로 제한할 수 있습니다.

IAM 서비스를 사용하여 서비스 역할과 해당 권한 정책을 생성합니다. 서비스 역할 생성에 대한 자세한 내용은 IAM 사용 설명서의 [AWS 서비스에 대한 권한을 위임할 역할 생성](#)을 참조하세요. 역할을 위임할 수 있는 서비스로 CloudFormation(`cloudformation.amazonaws.com`)을 지정합니다.

서비스 역할을 스택과 연결하려면 스택을 생성할 때 역할을 지정합니다. 자세한 내용은 [스택 옵션 구성](#)을 참조하세요. 콘솔에서 스택을 업데이트할 때 서비스 역할을 변경하거나 API를 통해 스택 [DeleteStack](#) 작업을 수행할 수도 있습니다. 서비스 역할을 지정하기 전에 역할을 전달할 수 있는 권한이 있는지 확인합니다(`iam:PassRole`). `iam:PassRole` 권한은 사용 가능한 역할을 지정합니다. 자세한 내용은 IAM 사용 설명서의 [사용자에게 AWS 서비스에 역할을 전달할 권한 부여](#)를 참조하세요.

Important

서비스 역할을 지정할 때 CloudFormation에서는 해당 스택에서 수행되는 모든 작업에 대해 항상 해당 역할을 사용합니다. 스택이 생성된 후에는 스택에 연결된 서비스 역할을 제거할 수 없습니다. 이 스택에서 작업을 수행할 수 있는 권한이 있는 다른 사용자는 `iam:PassRole` 권한의 보유 여부와 상관없이 이 역할을 사용할 수 있습니다. 역할에 사용자에게 불필요한 권한이 포함되어 있는 경우 사용자의 권한을 실수로 에스컬레이션할 수 있습니다. 이 역할은 최소한의 권한을 부여해야 합니다. 자세한 정보는 IAM 사용 설명서의 [최소 권한 적용](#)을 참조하십시오.

교차 서비스 혼동된 대리인 방지

혼동된 대리자 문제는 작업을 수행할 권한이 없는 엔터티가 권한이 더 많은 엔터티에게 작업을 수행하도록 강요할 수 있는 보안 문제입니다. AWS에서는 교차 서비스 가장으로 인해 혼동된 대리자 문제가 발생할 수 있습니다. 교차 서비스 가장은 한 서비스(직접적으로 호출하는 서비스)가 다른 서비스(직접적으로 호출되는 서비스)를 직접적으로 호출할 때 발생할 수 있습니다. 호출하는 서비스는 다른 고객의 리소스에 대해 액세스 권한이 없는 방식으로 작동하게 권한을 사용하도록 조작될 수 있습니다. 이를 방지하기 위해 AWS에서는 계정의 리소스에 대한 액세스 권한이 부여된 서비스 위탁자를 사용하여 모든 서비스에 대한 데이터를 보호하는 데 도움이 되는 도구를 제공합니다.

CloudFormation에서 다른 서비스에 CloudFormation 확장 프로그램과 같은 특정 리소스에 대해 부여하는 권한을 제한하려면 리소스 정책에서 [aws:SourceArn](#) 및 [aws:SourceAccount](#) 글로벌 조건 컨텍스트 키를 사용하는 것이 좋습니다. 하나의 리소스만 교차 서비스 액세스와 연결되도록 허용하려는 경우 `aws:SourceArn`을 사용하세요. 해당 계정의 모든 리소스가 교차 서비스 사용과 연결되도록 허용하려는 경우 `aws:SourceAccount`를 사용하세요.

`aws:SourceArn` 값이 CloudFormation이 저장하는 리소스의 ARN인지 확인하세요.

혼동된 대리인 문제로부터 보호하는 가장 효과적인 방법은 리소스의 전체 ARN이 포함된 `aws:SourceArn` 글로벌 조건 컨텍스트 키를 사용하는 것입니다. 리소스의 전체 ARN을 모를 경우 또는 여러 리소스를 지정하는 경우, ARN의 알 수 없는 부분에 대해 와일드카드(*)를 포함한 `aws:SourceArn` 전역 조건 컨텍스트 키를 사용합니다. 예제: `arn:aws:cloudformation:*:123456789012:*`.

`aws:SourceArn` 값에 계정 ID가 포함되어 있지 않은 경우 권한을 제한하려면 두 전역 조건 컨텍스트 키를 모두 사용해야 합니다.

다음 예는 CloudFormation에서 `aws:SourceArn` 및 `aws:SourceAccount` 전역 조건 컨텍스트 키를 사용하여 혼동된 대리자 문제를 방지하는 방법을 보여줍니다.

`aws:SourceArn` 및 `aws:SourceAccount` 조건 키를 사용하는 신뢰 정책 예

레지스트리 서비스의 경우 CloudFormation은 AWS Security Token Service(AWS STS)를 직접 호출하여 계정에서 서비스 역할을 수임합니다. 이 역할은 [RegisterType](#) 작업의 `ExecutionRoleArn`과 [LoggingConfig](#) 작업에 설정된 `LogRoleArn`에 대해 구성됩니다. 자세한 내용은 [공개 확장 프로그램 액세스를 위해 IAM 권한 및 신뢰 정책으로 실행 역할 구성](#) 섹션을 참조하세요.

이 예제 역할 신뢰 정책은 조건문을 사용하여 서비스 역할에 대한 AssumeRole 기능을 지정된 계정의 지정된 CloudFormation 확장 프로그램에 대한 작업으로만 제한합니다. `aws:SourceArn` 및 `aws:SourceAccount` 조건은 독립적으로 평가됩니다. 서비스 역할을 사용하려는 모든 요청은 두 조건을 모두 충족해야 합니다.

JSON

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

```

    "Principal": {
      "Service": [
        "resources.cloudformation.amazonaws.com"
      ]
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      },
      "ArnLike": {
        "aws:SourceArn": "arn:aws:cloudformation:us-east-1:123456789012:type/
resource/Organization-Service-Resource"
      }
    }
  }
}

```

추가 정보

StackSets에서 사용하는 서비스 역할에 `aws:SourceArn` 및 `aws:SourceAccount` 전역 조건 컨텍스트 키를 사용하는 정책의 예는 [혼란스러운 대리자 문제를 완화하기 위해 전역 키 설정](#) 섹션을 참조하세요.

자세한 내용은 IAM 사용 설명서의 [역할 트러스트 정책 업데이트](#)를 참조하세요.

전달 액세스 세션(FAS) 요청 및 권한 평가

CloudFormation 스택을 생성, 업데이트 및 삭제할 때 사용자는 선택적으로 IAM 역할 ARN을 지정할 수 있습니다. 역할이 제공되지 않은 경우 CloudFormation은 기본 서비스 메커니즘을 사용하여 다른 AWS 서비스와 상호 작용합니다. 이 시나리오에서 호출자는 관리되는 리소스에 대해 필요한 권한을 가지고 있어야 합니다. 또는 사용자가 자신의 IAM 역할을 제공하면 CloudFormation은 사용자 대신 해당 역할을 수입하여 서비스 상호 작용을 수행합니다.

사용자가 IAM 역할을 제공하는지 여부와 관계없이 CloudFormation은 각 리소스 작업에 대해 새로운 범위 축소 FAS 토큰을 생성합니다. 따라서 두 시나리오 모두에서 `aws:ViaAWSService`를 포함한 [FAS 관련 조건 키](#)가 채워집니다.

FAS 사용은 CloudFormation 작업 중에 IAM 정책이 평가되는 방식에 영향을 미칩니다. FAS 관련 조건 키의 영향을 받는 리소스가 포함된 템플릿으로 스택을 생성할 때 권한 거부가 발생할 수 있습니다.

IAM 정책 예제

다음 IAM 정책을 살펴보십시오. Statement2는 CloudFormation에서 AWS::KMS::Key 리소스 생성을 지속적으로 차단합니다. 이 제한은 스택 작업 중에 IAM 역할이 제공되었는지 여부와 관계없이 일관되게 적용됩니다. 이는 FAS 사용으로 인해 aws:ViaAWSService 조건 키가 항상 true로 설정되기 때문입니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Statement1",
      "Effect": "Allow",
      "Action": [
        "kms:CreateKey"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Sid": "Statement2",
      "Effect": "Deny",
      "Action": [
        "kms:CreateKey"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "Bool": {
          "aws:ViaAWSService": "true"
        }
      }
    }
  ]
}
```

스택 템플릿 예제

예를 들어 사용자가 다음 예제 템플릿으로 스택을 생성하면 `aws:ViaAWSService`가 `true`로 설정되고 FAS 정책에 의해 역할 권한이 재정의됩니다. 스택 생성은 `CreateKey` 작업을 거부하는 IAM 정책 `Statement2`의 영향을 받습니다. 이로 인해 권한 거부 오류가 발생합니다.

```
Resources:
  myPrimaryKey:
    Type: AWS::KMS::Key
    Properties:
      Description: An example multi-Region primary key
      KeyPolicy:
        Version: '2012-10-17'
        Id: key-default-1
        Statement:
          - Sid: Enable IAM User Permissions
            Effect: Allow
            Principal:
              AWS: !Join
                - ''
                - - 'arn:aws:iam::'
                  - !Ref AWS::AccountId
                  - ':root'
            Action: kms:*
            Resource: '*'
```

FAS에 대한 자세한 내용은 IAM 사용 설명서의 [전달 액세스 세션](#)을 참조하세요.

Note

대부분의 리소스는 이 동작을 준수합니다. 그러나 리소스를 생성, 업데이트 또는 삭제할 때 예기치 않은 성공 또는 실패가 발생하고 IAM 정책에 FAS 관련 조건 키가 포함된 경우 해당 리소스가 이 표준 패턴을 따르지 않는 일부 리소스 하위 집합에 속할 가능성이 높습니다.

AWS CloudTrail를 사용하여 CloudFormation API 호출 로깅

CloudFormation는 CloudFormation에서 사용자, 역할 또는 AWS 서비스가 수행한 작업에 대한 레코드를 제공하는 서비스인 AWS CloudTrail과 통합됩니다. CloudTrail은 CloudFormation 콘솔의 호출 및 CloudFormation API에 대한 코드 호출을 포함하여 CloudFormation에 대한 모든 API 호출을 이벤트로 캡처합니다. 추적을 생성하면 CloudFormation 이벤트를 비롯하여 CloudTrail 이벤트를 Amazon S3 버킷으로 지속적으로 배포할 수 있습니다. 추적을 구성하지 않은 경우에도 이벤트 기록에서 CloudTrail

콘솔의 최신 이벤트를 볼 수 있습니다. CloudTrail에서 수집하는 정보를 사용하여 CloudFormation에 수행된 요청, 요청이 수행된 IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다.

CloudTrail에 대한 자세한 내용은 [AWS CloudTrail 사용 설명서](#)를 참조하세요.

주제

- [CloudTrail의 CloudFormation 정보](#)
- [CloudFormation 로그 파일 항목 이해](#)

CloudTrail의 CloudFormation 정보

CloudTrail은 계정 생성 시 AWS 계정에서 활성화됩니다. CloudFormation에서 활동이 발생하면 해당 활동이 이벤트 기록(Event history)의 다른 AWS 서비스 이벤트와 함께 CloudTrail 이벤트에 기록됩니다. AWS 계정에서 최신 이벤트를 확인, 검색 및 다운로드할 수 있습니다. 자세한 내용은 [CloudTrail 이벤트 기록을 사용하여 이벤트 보기](#)를 참조하세요.

CloudFormation의 이벤트를 포함하여 AWS 계정에 이벤트를 지속적으로 기록하려면 추적을 생성합니다. CloudTrail은 추적을 사용하여 Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. 콘솔에서 추적을 생성하면 기본적으로 모든 Region에 추적이 적용됩니다. 추적은 AWS 파티션에 있는 모든 리전의 이벤트를 로깅하고 지정된 Amazon S3 버킷으로 로그 파일을 전송합니다. 또는 CloudTrail 로그에서 수집된 이벤트 데이터를 추가 분석 및 처리하도록 다른 AWS 서비스를 구성할 수 있습니다. 자세한 내용은 다음을 참조하세요.

- [추적 생성 개요](#)
- [CloudTrail 지원 서비스 및 통합](#)
- [CloudTrail에 대한 Amazon SNS 알림 구성](#)
- [여러 리전에서 CloudTrail 로그 파일 수신](#) 및 [여러 계정에서 CloudTrail 로그 파일 수신](#)

모든 CloudFormation 작업이 CloudTrail에서 로깅되고 [AWS CloudFormation API 참조](#)에서 문서화됩니다. 예를 들어, CreateStack, DeleteStack 및 ListStacks 섹션을 호출하면 CloudTrail 로그 파일에 항목이 생성됩니다.

모든 이벤트 또는 로그 항목에는 요청을 생성했던 사용자에 관한 정보가 포함됩니다. ID 정보를 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청을 루트로 했는지 아니면 IAM 사용자 자격 증명 정보로 했는지 여부.

- 역할 또는 페더레이션 사용자의 임시 자격 증명을 사용하여 요청이 생성되었는지 여부.
- 다른 AWS 서비스에서 요청했는지.

자세한 내용은 [CloudTrail userIdentity 요소](#)를 참조하세요.

CloudFormation 로그 파일 항목 이해

트레일이란 지정한 S3 버킷에 이벤트를 로그 파일로 입력할 수 있게 하는 구성입니다. CloudTrail 로그 파일에는 하나 이상의 로그 항목이 포함될 수 있습니다. 이벤트는 모든 소스로부터의 단일 요청을 나타내며 요청 작업, 작업 날짜와 시간, 요청 파라미터 등에 대한 정보가 들어 있습니다. CloudTrail 로그 파일은 퍼블릭 API 직접 호출의 주문 스택 트레이스가 아니므로 특정 순서로 표시되지 않습니다.

다음 예는 CreateStack작업을 보여주는 CloudTrail 로그 항목입니다. Alice라는 IAM 사용자가 수행한 작업입니다.

Note

입력 파라미터 키 이름만 로깅됩니다. 파라미터 값은 로깅되지 않습니다.

```
{
  "eventVersion": "1.01",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAABCDEFGHijklmnopq",
    "arn": "arn:aws:iam::012345678910:user/Alice",
    "accountId": "012345678910",
    "accessKeyId": "AKIDEXAMPLE",
    "userName": "Alice"
  },
  "eventTime": "2014-03-24T21:02:43Z",
  "eventSource": "cloudformation.amazonaws.com",
  "eventName": "CreateStack",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "127.0.0.1",
  "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
  "requestParameters": {
    "templateURL": "templateURL",
    "tags": [
      {
```

```

        "key": "test",
        "value": "tag"
    }
],
"stackName": "my-test-stack",
"disableRollback": true,
"parameters": [
    {
        "parameterKey": "password"
    },
    {
        "parameterKey": "securitygroup"
    }
]
},
"responseElements": {
    "stackId": "arn:aws:cloudformation:us-east-1:012345678910:stack/my-test-stack/
a38e6a60-b397-11e3-b0fc-08002755629e"
},
"requestID": "9f960720-b397-11e3-bb75-a5b75389b02d",
"eventID": "9bf6cfb8-83e1-4589-9a70-b971e727099b"
}

```

다음 예제에서는 Alice가 UpdateStack 스택에서 my-test-stack 작업을 호출한 것을 보여줍니다.

```

{
    "eventVersion": "1.01",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAABCDEFGHijklmnopq",
        "arn": "arn:aws:iam::012345678910:user/Alice",
        "accountId": "012345678910",
        "accessKeyId": "AKIDEXAMPLE",
        "userName": "Alice"
    },
    "eventTime": "2014-03-24T21:04:29Z",
    "eventSource": "cloudformation.amazonaws.com",
    "eventName": "UpdateStack",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
    "requestParameters": {
        "templateURL": "templateURL",
    }
}

```

```

    "parameters": [
      {
        "parameterKey": "password"
      },
      {
        "parameterKey": "securitygroup"
      }
    ],
    "stackName": "my-test-stack"
  },
  "responseElements": {
    "stackId": "arn:aws:cloudformation:us-east-1:012345678910:stack/my-test-stack/a38e6a60-b397-11e3-b0fc-08002755629e"
  },
  "requestID": "def0bf5a-b397-11e3-bb75-a5b75389b02d",
  "eventID": "637707ce-e4a3-4af1-8edc-16e37e851b17"
}

```

다음 예제에서는 Alice가 ListStacks 작업을 호출한 것을 보여줍니다.

```

{
  "eventVersion": "1.01",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAABCDEFGHijklmnopq",
    "arn": "arn:aws:iam::012345678910:user/Alice",
    "accountId": "012345678910",
    "accessKeyId": "AKIDEXAMPLE",
    "userName": "Alice"
  },
  "eventTime": "2014-03-24T21:03:16Z",
  "eventSource": "cloudformation.amazonaws.com",
  "eventName": "ListStacks",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "127.0.0.1",
  "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
  "requestParameters": null,
  "responseElements": null,
  "requestID": "b7d351d7-b397-11e3-bb75-a5b75389b02d",
  "eventID": "918206d0-7281-4629-b778-b91eb0d83ce5"
}

```

다음 예제에서는 Alice가 DescribeStacks 스택에서 my-test-stack 작업을 호출한 것을 보여줍니다.

```
{
  "eventVersion": "1.01",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAABCDEFGHJKLMNOPQ",
    "arn": "arn:aws:iam::012345678910:user/Alice",
    "accountId": "012345678910",
    "accessKeyId": "AKIDEXAMPLE",
    "userName": "Alice"
  },
  "eventTime": "2014-03-24T21:06:15Z",
  "eventSource": "cloudformation.amazonaws.com",
  "eventName": "DescribeStacks",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "127.0.0.1",
  "userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
  "requestParameters": {
    "stackName": "my-test-stack"
  },
  "responseElements": null,
  "requestID": "224f2586-b398-11e3-bb75-a5b75389b02d",
  "eventID": "9e5b2fc9-1ba8-409b-9c13-587c2ea940e2"
}
```

다음 예제에서는 Alice가 DeleteStack 스택에서 my-test-stack 작업을 호출한 것을 보여줍니다.

```
{
  "eventVersion": "1.01",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAABCDEFGHJKLMNOPQ",
    "arn": "arn:aws:iam::012345678910:user/Alice",
    "accountId": "012345678910",
    "accessKeyId": "AKIDEXAMPLE",
    "userName": "Alice"
  },
  "eventTime": "2014-03-24T21:07:15Z",
  "eventSource": "cloudformation.amazonaws.com",
  "eventName": "DeleteStack",
  "awsRegion": "us-east-1",
}
```

```
"sourceIPAddress": "127.0.0.1",
"userAgent": "aws-cli/1.2.11 Python/2.7.4 Linux/2.6.18-164.el5",
"requestParameters": {
  "stackName": "my-test-stack"
},
"responseElements": null,
"requestID": "42dae739-b398-11e3-bb75-a5b75389b02d",
"eventID": "4965eb38-5705-4942-bb7f-20ebe79aa9aa"
}
```

CloudFormation에서 인프라 보안

관리형 서비스인 AWS CloudFormation은 AWS 글로벌 네트워크 보안으로 보호됩니다. AWS 보안 서비스와 AWS의 인프라 보호 방법에 대한 자세한 내용은 [AWS 클라우드 보안](#)을 참조하세요. 인프라 보안에 대한 모범 사례를 사용하여 AWS 환경을 설계하려면 보안 원칙 AWS Well-Architected Framework의 [인프라 보호](#)를 참조하세요.

AWS에서 게시한 API 직접 호출을 사용하여 네트워크를 통해 CloudFormation에 액세스합니다. 고객은 다음을 지원해야 합니다.

- Transport Layer Security(TLS) TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- DHE(Ephemeral Diffie-Hellman) 또는 ECDHE(Elliptic Curve Ephemeral Diffie-Hellman)와 같은 완전 전송 보안(PFS)이 포함된 암호 제품군 Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

이러한 API 작업은 어떤 네트워크 위치에서든 호출할 수 있지만, CloudFormation(은)는 소스 IP 주소에 따른 제한 사항을 포함할 수 있는 리소스 기반 액세스 정책을 지원합니다. CloudFormation 정책을 사용하여 특정 Amazon Virtual Private Cloud(VPC) 엔드포인트 또는 특정 VPC에서 액세스를 제어할 수도 있습니다. 그러면 AWS 네트워크 내에 있는 특정 VPC에서만 특정 CloudFormation 리소스에 대한 네트워크 액세스가 효과적으로 격리됩니다.

CloudFormation의 복원성

AWS 글로벌 인프라는 AWS 지역 및 가용 영역을 중심으로 구축됩니다. AWS 리전은 물리적으로 분리되고 격리된 다수의 가용 리전을 제공하며 이러한 가용 리전은 짧은 지연 시간, 높은 처리량 및 높은 중복성을 갖춘 네트워크에 연결되어 있습니다. 가용 영역을 사용하면 중단 없이 영역 간에 자동으로 장애 극복 조치가 이루어지는 애플리케이션 및 데이터베이스를 설계하고 운영할 수 있습니다. 가용 영역은 기존의 단일 또는 다중 데이터 센터 인프라보다 가용성, 내결함성, 확장성이 뛰어납니다.

AWS 리전 및 가용 영역에 대한 자세한 정보는 [AWS 글로벌 인프라](#)를 참조하세요.

의 규정 준수 검증CloudFormation

AWS 서비스가 특정 규정 준수 프로그램의 범위에 포함되는지 알아보려면 [규정 준수 프로그램 제공 범위 내 AWS 서비스](#)를 참조하고 관심 있는 규정 준수 프로그램을 선택하십시오. 일반적인 정보는 [AWS 규정 준수 프로그램](#)을 참조하세요.

AWS Artifact를 사용하여 타사 감사 보고서를 다운로드할 수 있습니다. 자세한 내용은 [AWS Artifact에서 보고서 다운로드](#)를 참조하세요.

AWS 서비스 사용 시 규정 준수 책임은 데이터의 민감도, 회사의 규정 준수 목표 및 관련 법률 및 규정에 따라 결정됩니다. AWS 서비스 사용 시 규정 준수 책임에 대한 자세한 내용은 [AWS 보안 설명서](#)를 참조하세요.

CloudFormation의 구성 및 취약성 분석

구성 및 IT 컨트롤은 AWS와 고객 간의 공동 책임입니다. 자세한 내용은 AWS [공동 책임 모델](#)을 참조하세요.

CloudFormation의 보안 모범 사례

CloudFormation는 자체 보안 정책을 개발하고 구현할 때 고려해야 할 여러 보안 기능을 제공합니다. 다음 모범 사례는 일반적인 지침이며 완벽한 보안 솔루션을 나타내지는 않습니다. 이러한 모범 사례는 환경에 적절하지 않거나 충분하지 않을 수 있으므로 참고용으로만 사용해 주세요.

주제

- [IAM을 사용하여 액세스 제어](#)
- [템플릿에 자격 증명을 포함하지 않음](#)
- [CloudFormation 호출 기록에 AWS CloudTrail 사용](#)

IAM을 사용하여 액세스 제어

IAM은 AWS에서 사용자 및 사용자 권한을 관리하는 데 사용할 수 있는 AWS 서비스입니다.

CloudFormation에 IAM을 사용하여 스택 템플릿 보기, 스택 생성 또는 스택 삭제와 같은 사용자가 수행

할 수 있는 CloudFormation 작업을 지정할 수 있습니다. 또한 CloudFormation 스택을 관리하는 사람에게서는 누구나 이러한 스택 내의 리소스에 대한 권한이 필요합니다. 예를 들어 CloudFormation을 사용하여 Amazon EC2 인스턴스를 시작, 업데이트 또는 종료하려는 경우 관련 Amazon EC2 작업을 호출하기 위한 권한이 있어야 합니다.

대부분의 경우 템플릿에서 모든 리소스를 관리하기 위해서는 전체 액세스 권한이 필요합니다. CloudFormation은 사용자 대신 이러한 리소스를 생성, 수정 및 삭제하도록 호출합니다. 사용자와 CloudFormation 서비스 간에 권한을 분리하려면 서비스 역할을 사용합니다. CloudFormation에서는 사용자의 정책 대신 서비스 역할의 정책을 사용하여 호출합니다. 자세한 내용은 [CloudFormation 서비스 역할](#) 섹션을 참조하세요.

템플릿에 자격 증명을 포함하지 않음

CloudFormation 템플릿에 민감한 정보를 포함하는 대신, 스택 템플릿의 동적 참조를 사용하는 것이 좋습니다.

동적 참조는 AWS Systems Manager Parameter Store 또는 AWS Secrets Manager와 같이 다른 서비스에서 저장 및 관리되는 외부 값을 참조하는 간결하면서 강력한 방법을 제공합니다. 동적 참조 사용 시 CloudFormation은 스택 및 변경 세트 작업 중에 필요한 경우 지정된 참조의 값을 검색하고 적절한 리소스에 값을 전달합니다. 하지만 CloudFormation은 실제 참조 값을 저장하지 않습니다. 자세한 내용은 [동적 참조를 사용하여 다른 서비스에 저장된 값 가져오기](#) 섹션을 참조하세요.

[AWS Secrets Manager](#)는 데이터베이스와 다른 서비스의 자격 증명을 안전하게 암호화, 저장 및 검색하는 데 효과적입니다. [AWS Systems Manager Parameter Store](#)는 구성 데이터 관리를 위한 안전한 계층적 스토리지를 제공합니다.

템플릿 파라미터 정의에 대한 자세한 내용은 [CloudFormation 템플릿 Parameters 구문](#) 단원을 참조하십시오.

CloudFormation 호출 기록에 AWS CloudTrail 사용

AWS CloudTrail은 AWS 계정에서 CloudFormation API를 직접 호출하는 모든 사용자를 추적합니다. CloudFormation API, CloudFormation 콘솔, 백엔드 콘솔 또는 CloudFormation AWS CLI 명령을 사용할 때마다 API 직접 호출이 로깅됩니다. 로깅을 활성화하고 로그를 저장할 S3 버킷을 지정합니다. 이렇게 하면 필요한 경우 계정에서 누가 어떤 CloudFormation 호출을 실행했는지 감사할 수 있습니다. 자세한 내용은 [AWS CloudTrail를 사용하여 CloudFormation API 호출 로깅](#) 섹션을 참조하세요.

인터페이스 엔드포인트를 사용하여 CloudFormation에 액세스(AWS PrivateLink)

AWS PrivateLink를 사용하여 VPC와 CloudFormation 사이에 프라이빗 연결을 생성할 수 있습니다. 인터넷 게이트웨이, NAT 디바이스, VPN 연결 또는 Direct Connect 연결을 사용하지 않고 VPC에 있는 것처럼 CloudFormation에 액세스할 수 있습니다. VPC의 인스턴스에서 CloudFormation API에 액세스하는 데는 퍼블릭 IP 주소가 필요하지 않습니다.

AWS PrivateLink에서 제공되는 인터페이스 엔드포인트를 생성하여 이 프라이빗 연결을 설정합니다. 인터페이스 엔드포인트에 대해 사용 설정하는 각 서브넷에서 엔드포인트 네트워크 인터페이스를 생성합니다. 이는 CloudFormation로 향하는 트래픽의 진입점 역할을 하는 요청자 관리형 네트워크 인터페이스입니다.

CloudFormation에서는 인터페이스 엔드포인트를 통해 모든 API 작업에 대한 호출 수행을 지원합니다.

CloudFormation VPC 엔드포인트 고려 사항

인터페이스 엔드포인트를 설정하기 전에 먼저 AWS PrivateLink 가이드에서 [인터페이스 VPC 엔드포인트를 사용하여 AWS 서비스에 액세스](#) 주제의 사전 요구 사항을 충족했는지 확인하세요.

CloudFormation의 인터페이스 엔드포인트를 설정할 경우 다음과 같은 추가 사전 요구 사항 및 고려 사항이 적용됩니다.

- 사용자 지정 리소스 요청 또는 대기 조건에 응답해야 하는 VPC 내의 리소스가 있는 경우 해당 리소스가 필요한 CloudFormation 고유의 Amazon S3 버킷에 액세스할 수 있는지 확인합니다. CloudFormation은 [사용자 지정 리소스](#) 요청 또는 [대기 조건](#)에 대한 응답을 모니터링하기 위해 각 리전에 S3 버킷을 갖추고 있습니다. 템플릿에 VPC의 사용자 지정 리소스 또는 대기 조건이 포함되는 경우 VPC 엔드포인트 정책은 사용자가 다음 버킷에 대한 응답을 전송하도록 허용해야 합니다.
- 사용자 지정 리소스의 경우 `cloudformation-custom-resource-response-region` 버킷에 대한 트래픽을 허용합니다. 사용자 지정 리소스를 사용하는 경우 AWS 리전 이름에 대시가 포함되지 않습니다. 예를 들어 `uswest2`입니다.
- 대기 조건의 경우 `cloudformation-waitcondition-region` 버킷에 대한 트래픽을 허용합니다. 대기 조건을 사용하는 경우 AWS 리전 이름에 대시가 포함됩니다. 예를 들어 `us-west-2`입니다.

엔드포인트 정책이 이러한 버킷에 대한 트래픽을 차단하는 경우 CloudFormation은 응답을 수신하지 않으며 스택 작업이 실패합니다. 예를 들어, 대기 조건에 응답해야 하는 `us-west-2` 리전의 VPC에

리소스가 있는 경우 리소스는 cloudformation-waitcondition-us-west-2 버킷에 응답을 전송할 수 있어야 합니다.

현재 CloudFormation을 사용할 수 있는 AWS 리전의 목록은 Amazon Web Services 일반 참조의 [CloudFormation 엔드포인트 및 할당량](#) 페이지를 참조하세요.

- VPC 엔드포인트는 교차 리전 요청을 지원하지 않습니다. CloudFormation에 대한 API 호출 발급을 계획하는 리전과 동일한 리전에 엔드포인트를 생성하는지 확인하세요.
- VPC 엔드포인트는 Route 53을 통해 Amazon 제공 DNS만 지원합니다. 자체 DNS를 사용하는 경우에는 조건부 DNS 전달을 사용할 수 있습니다. 자세한 정보는 Amazon VPC 사용 설명서의 [Amazon VPC의 DHCP 옵션 세트](#)를 참조하세요.
- VPC 엔드포인트에 연결된 보안 그룹은 VPC의 프라이빗 서브넷에서 443 포트로 들어오는 연결을 허용해야 합니다.

CloudFormation에 대한 인터페이스 VPC 엔드포인트 생성

Amazon VPC 콘솔 또는 AWS Command Line Interface(AWS CLI)를 사용하여 CloudFormation에 대한 VPC 엔드포인트를 생성할 수 있습니다. 자세한 정보는 AWS PrivateLink 가이드의 [VPC 엔드포인트 생성](#)을 참조하세요.

다음 서비스 이름을 사용하여 CloudFormation에 대한 엔드포인트를 생성합니다.

- com.amazonaws.**region**.cloudformation – CloudFormation API 작업을 위한 엔드포인트를 생성합니다.

인터페이스 엔드포인트에 프라이빗 DNS를 사용하도록 설정하는 경우, 리전에 대한 기본 DNS 이름(예: CloudFormation)을 사용하여 API 요청을 할 수 있습니다. 예를 들어 cloudformation.us-east-1.amazonaws.com입니다.

FIPS 엔드포인트가 지원되는 CloudFormation에서는 다음 서비스 이름을 사용하여 AWS 리전에 대한 인터페이스 엔드포인트를 생성할 수도 있습니다.

- com.amazonaws.**region**.cloudformation-fips – [FIPS\(Federal Information Processing Standard\) 140-2](#)를 준수하는 CloudFormation API에 대한 엔드포인트를 생성합니다.

CloudFormation 엔드포인트의 전체 목록은 Amazon Web Services 일반 참조의 [CloudFormation 엔드포인트 및 할당량](#)을 참조하세요.

CloudFormation에 대한 VPC 엔드포인트 정책 생성

엔드포인트 정책은 인터페이스 엔드포인트에 연결할 수 있는 IAM 리소스입니다. 기본 엔드포인트 정책을 사용하면 인터페이스 엔드포인트를 통해 CloudFormation에 대한 전체 액세스를 허용합니다. VPC에서 CloudFormation에 허용되는 액세스를 제어하려면 사용자 지정 엔드포인트 정책을 인터페이스 엔드포인트에 연결합니다.

엔드포인트 정책은 다음 정보를 지정합니다.

- 작업을 수행할 수 있는 위탁자(AWS 계정, IAM 사용자, IAM 역할)
- 수행할 수 있는 작업
- 작업을 수행할 수 있는 리소스.

자세한 정보는 AWS PrivateLink 가이드의 [엔드포인트 정책을 사용하여 VPC 엔드포인트에 대한 액세스 제어](#)를 참조하세요.

예제: CloudFormation 작업에 대한 VPC 엔드포인트 정책

다음은 CloudFormation에 대한 엔드포인트 정책의 예입니다. 이 정책은 엔드포인트에 연결될 때 모든 리소스의 모든 보안 주체에 대한 액세스 권한을 나열된 CloudFormation 작업에 부여합니다. 다음 예제에서는 모든 사용자에게 VPC 엔드포인트를 통해 스택을 생성할 수 있는 권한을 거부하고 CloudFormation 서비스의 다른 모든 작업에 대한 전체 액세스를 허용합니다.

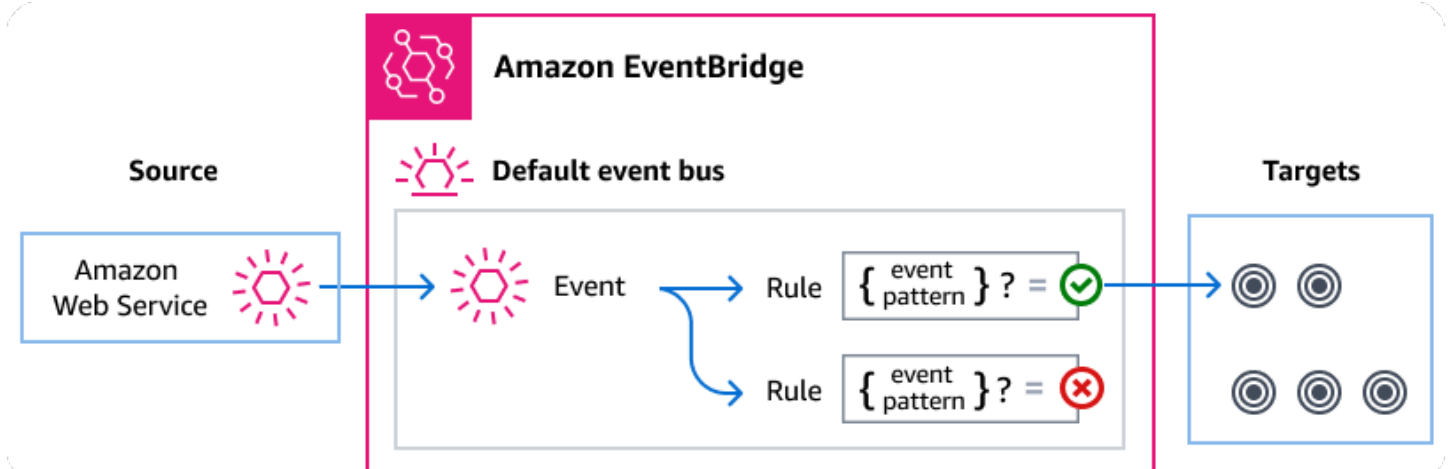
```
{
  "Statement": [
    {
      "Action": "cloudformation:*",
      "Effect": "Allow",
      "Principal": "*",
      "Resource": "*"
    },
    {
      "Action": "cloudformation:CreateStack",
      "Effect": "Deny",
      "Principal": "*",
      "Resource": "*"
    }
  ]
}
```

EventBridge를 사용한 CloudFormation 및 Git 동기화 이벤트 모니터링

Amazon EventBridge는 이벤트를 사용하여 애플리케이션 구성 요소를 서로 연결하는 서버리스 서비스로, 확장 가능한 이벤트 기반 애플리케이션을 더 쉽게 구축할 수 있습니다. 이벤트 기반 아키텍처는 이벤트를 내보내고 이에 응답하여 함께 작동하는 느슨하게 결합된 소프트웨어 시스템을 구축하는 스타일입니다. 이벤트는 리소스나 환경의 변화를 나타냅니다.

많은 AWS 서비스와 마찬가지로 CloudFormation은 이벤트를 생성하여 EventBridge 기본 이벤트 버스로 보냅니다. 이 이벤트 버스는 모든 AWS 계정에 자동으로 프로비저닝됩니다. 이벤트 버스는 이벤트를 수신하여 0개 이상의 목적지 또는 대상에 전달하는 라우터입니다. 이벤트 버스에 대해 지정된 규칙은 이벤트가 도착할 때 이벤트를 평가합니다. 각 규칙은 이벤트가 규칙의 이벤트 패턴과 일치하는지 여부를 확인합니다. 이벤트가 일치하면 이벤트 버스는 이벤트를 지정된 대상에게 전송합니다.

자세한 내용은 Amazon EventBridge 사용 설명서의 [Amazon EventBridge 시작하기](#)를 참조하세요.



주제

- [CloudFormation과 Git 동기화 이벤트 개요](#)
- [Amazon EventBridge 권한](#)
- [EventBridge 규칙을 위한 사용자 지정 이벤트 패턴 생성](#)
- [CloudFormation 이벤트 세부 정보 참조](#)

CloudFormation과 Git 동기화 이벤트 개요

CloudFormation은 스택에서 생성, 업데이트, 삭제 또는 드리프트 감지 작업이 수행될 때마다 EventBridge로 이벤트를 전송합니다. 또한 CloudFormation은 이벤트를 EventBridge로 전송하여 스택 세트 및 스택 세트 인스턴스의 상태를 변경합니다. EventBridge 규칙을 사용하여 이벤트를 정의된 대상으로 라우팅할 수 있습니다. 이러한 이벤트는 전송이 보장되며 순서와 다르게 전송될 수도 있습니다.

CloudFormation 이벤트는 스택 또는 스택 세트 및 해당 리소스의 변경을 나타내므로 이를 사용하여 각 이벤트와 연결된 워크플로를 시작할 수 있습니다. 예제:

- CloudFormation을 통해 프로비저닝된 모든 리소스에 스택 또는 스택 세트별 태그를 생성합니다.
- CloudFormation 스택 또는 스택 세트와 Amazon WorkSpaces Application Manager(Amazon WAM) 간의 연결을 설정합니다.
- 생성된 스택 또는 스택 세트에 대한 AppRegistry와의 연결을 지정합니다.

다음 이벤트는 CloudFormation에 의해 생성되어 EventBridge의 기본 이벤트 버스로 전송됩니다. 자세한 내용은 [???](#) 섹션을 참조하세요.

이벤트 유형	설명	이벤트 소스
리소스 상태 변경	기본 리소스 속성을 변경하는 스택에서 수행된 모든 업데이트입니다. 지원되는 AWS 리소스 유형의 전체 목록은 AWS 리소스 및 속성 유형 참조 를 참조하세요.	AWS CloudFormation
스택 상태 변경	지정된 스택의 상태 변경을 나타냅니다. 코드에 대한 자세한 내용은 스택 상태 코드 섹션을 참조하세요.	AWS CloudFormation
드리프트 감지 상태 변경	지정된 스택에서 사용자가 시작한 드리프트 감지 업데이트를 나타냅니다. 드리프트 감지를 지원하는 전체 변경 가능 유형 및 변경 불가 유형의 목록은 리소스 유형 지원 섹션을 참조하세요.	AWS CloudFormation

이벤트 유형	설명	이벤트 소스
StackSet 상태 변경	지정된 스택 세트의 상태 변경을 나타냅니다.	AWS CloudFormation
StackSet 스택 인스턴스 상태 변경	특정 StackSet 스택 인스턴스의 상태 변경을 나타냅니다. 코드에 대한 자세한 내용은 스택 인스턴스 상태 코드 섹션을 참조하세요.	AWS CloudFormation
StackSet 작업 상태	지정된 StackSet 작업의 상태 변경을 나타냅니다. 코드에 대한 자세한 내용은 StackSets 상태 코드 섹션을 참조하세요.	AWS CloudFormation

또한 AWS CloudFormation Git 동기화는 리포지토리 동기화 및 리소스 동기화의 상태 변경 이벤트를 EventBridge로 보냅니다.

다음 Git 동기화 이벤트는 CodeConnections에 의해 생성되어 EventBridge의 기본 이벤트 버스로 전송됩니다. 자세한 내용은 [???](#) 섹션을 참조하세요.

이벤트 유형	설명	이벤트 소스
리포지토리 동기화 상태 변경	Git 리포지토리 동기화의 상태 변경을 나타냅니다.	AWS CodeConnections
리소스 동기화 상태 변경	Git 리포지토리 동기화의 상태 변경을 나타냅니다.	AWS CodeConnections

Amazon EventBridge 권한

CloudFormation에는 이벤트를 EventBridge에 전송하기 위해 추가 권한이 필요하지 않습니다. 이벤트에는 CloudFormation의 API 작업을 통해 이미 제공되는 정보가 포함되어 있습니다.

지정하는 대상에는 특정 권한이나 구성이 필요할 수 있습니다. 대상에 특정 서비스를 사용하는 방법에 대한 자세한 내용은 Amazon EventBridge 사용 설명서의 [Amazon EventBridge 대상](#)을 참조하세요.

EventBridge 규칙을 위한 사용자 지정 이벤트 패턴 생성

CloudFormation용 EventBridge와 Git 동기화 이벤트에서 사전 정의된 여러 패턴을 찾을 수 있습니다. 이를 통해 이벤트 패턴이 생성되는 방법이 간소화됩니다. 이벤트 패턴을 직접 작성하는 대신 양식에서 필드 값을 선택하면 EventBridge가 자동으로 패턴을 생성합니다. 이러한 사전 정의된 이벤트 패턴 중 하나를 사용하여 새 규칙을 생성하거나 사용자 지정 이벤트 패턴을 생성할 수 있습니다.

CloudFormation과 같은 서비스가 이벤트를 기본 이벤트 버스에 전달할 때 EventBridge는 규칙에 정의된 이벤트 패턴을 사용하여 이벤트를 규칙의 대상으로 전달해야 하는지 여부를 결정합니다. 이벤트 패턴이 원하는 CloudFormation 이벤트의 데이터와 일치합니다.

각 이벤트 패턴은 다음을 포함하는 JSON 객체입니다.

- 이벤트의 출처 서비스를 식별하는 `source` 속성. 예: `aws.cloudformation` 또는 `aws.codeconnections`.
- (선택 사항): 일치시킬 이벤트 유형의 배열을 포함하는 `detail-type` 속성입니다.
- (선택 사항): 일치시킬 다른 이벤트 데이터를 포함하는 `detail` 속성입니다.

예를 들면 스택 ID, 관련 리소스, 다양한 리소스의 상태, 특정 유형의 이벤트와 관련된 기타 데이터 등이 있습니다.

예를 들어 다음 이벤트 패턴은 모든 리소스 상태 변경 이벤트와 일치합니다.

```
{
  "source": ["aws.cloudformation"],
  "detail-type": ["CloudFormation Resource Status Change"]
}
```

반면에 다음 이벤트 패턴은 이벤트 세부 정보 데이터를 사용하여 CloudFormation이 새 `AWS::S3::Bucket` 또는 `AWS::SNS::Topic` 리소스를 생성하는 리소스 상태 변경 이벤트만 일치시킵니다.

```
{
  "source": ["aws.cloudformation"],
  "detail-type": ["CloudFormation Resource Status Change"],
  "detail": {
    "status-details": {
      "status": ["CREATE_COMPLETE"]
    }
  },
}
```

```
"resource-type": ["AWS::S3::Bucket", "AWS::SNS::Topic"]
}
```

이벤트 작성에 대한 자세한 내용은 Amazon EventBridge 사용 설명서의 [이벤트 패턴](#)을 참조하세요.

CloudFormation 이벤트 세부 정보 참조

AWS 서비스의 모든 이벤트에는 이벤트의 소스인 AWS 서비스, 이벤트가 생성된 시간, 이벤트가 발생한 계정 및 리전 등과 같은 이벤트에 대한 메타데이터를 포함하는 공통 필드 집합이 있습니다. 이러한 일반 필드에 대한 정의는 AWS 이벤트 참조의 [AWS 서비스 이벤트 메타데이터](#)를 참조하세요.

또한 각 이벤트에는 해당 특정 이벤트와 관련된 데이터를 포함하는 detail 필드가 있습니다. 다음 참조는 다양한 CloudFormation 이벤트에 대한 세부 정보 필드를 정의합니다.

EventBridge를 사용하여 CloudFormation 이벤트를 선택하고 관리할 때는 다음 사항을 염두에 두는 것이 유용합니다.

- source 필드는 이벤트 유형을 지정합니다.

예: aws.cloudformation 또는 aws.codeconnections.

- detail-type 필드는 이벤트 유형을 지정합니다.

예: CloudFormation Resource Status Change 또는 CloudFormation Drift Detection Status Change.

- detail 필드는 해당 특정 이벤트와 관련된 데이터를 포함합니다.

예를 들면 스택 ID, 관련 리소스, 다양한 리소스의 상태, 특정 유형의 이벤트와 관련된 기타 데이터 등이 있습니다.

CloudFormation 이벤트와 일치하는 규칙을 활성화하는 이벤트 패턴을 구성하는 방법에 대한 자세한 내용은 Amazon EventBridge 사용 설명서의 [이벤트 패턴](#)을 참조하세요.

이벤트 및 EventBridge가 이벤트를 처리하는 방법에 대한 자세한 내용은 Amazon EventBridge 사용 설명서의 [EventBridge 이벤트](#)를 참조하세요.

주제

- [리소스 상태 변경 이벤트 세부 정보](#)
- [스택 상태 변경 이벤트 세부 정보](#)

- [드리프트 감지 상태 변경 이벤트 세부 정보](#)
- [StackSet 상태 변경 이벤트 세부 정보](#)
- [StackSet 스택 인스턴스 상태 변경 이벤트 세부 정보](#)
- [StackSet 작업 상태 변경 이벤트 세부 정보](#)
- [리포지토리 동기화 상태 변경 이벤트 세부 정보](#)
- [리소스 동기화 상태 변경 이벤트 세부 정보](#)

리소스 상태 변경 이벤트 세부 정보

다음은 리소스 상태 변경 이벤트의 세부 정보 필드입니다.

source 및 detail-type 필드는 이벤트에 대한 특정 값을 포함하므로 포함됩니다.

```
{
  . . . ,
  "detail-type": "CloudFormation Resource Status Change",
  "source": "aws.cloudformation",
  . . . ,
  "detail": {
    "stack-id" : "string",
    "logical-resource-id" : "string",
    "physical-resource-id": "string",
    "status-details": {
      "status": "string",
      "status-reason": "string"
    },
    "resource-type": "string",
    "client-request-token": "string"
  }
}
```

detail-type

이벤트의 유형을 식별합니다.

리소스 상태 이벤트의 경우 이 값은 CloudFormation Resource Status Change입니다.

source

이벤트를 생성한 서비스를 식별합니다. CloudFormation 이벤트의 경우 이 값은 aws.cloudformation입니다.

detail

이벤트에 대한 정보를 포함하는 JSON 객체입니다. 이벤트를 생성하는 서비스에 따라 이 필드의 내용이 결정됩니다.

리소스 상태 이벤트의 경우 이 데이터는 다음을 포함합니다.

stack-id

스택이 연결되어 있는 고유한 스택 ID입니다.

logical-resource-id

템플릿에 지정된 것과 같은 리소스의 논리명입니다.

physical-resource-id

CloudFormation에서 지원하는 리소스의 물리적 인스턴스 ID에 해당하는 이름 또는 고유 식별자입니다.

status-details

status

리소스의 상태입니다.

status-reason

리소스의 상태 사유입니다.

resource-type

리소스의 유형입니다. 예를 들어 `AWS::S3::Bucket`입니다.

client-request-token

API를 호출하는 데 사용되는 액세스 토큰입니다. 지정된 스택 작업에 의해 시작되는 모든 이벤트는 동일한 클라이언트 요청 토큰에 할당됩니다. 이 토큰은 작업을 추적하는 데 사용할 수 있습니다. 콘솔에서 시작되는 스택 작업은 `Console-StackOperation-ID` 토큰 형식을 사용합니다. 이러한 형식은 스택 작업을 쉽게 식별하는 데 도움이 됩니다. 예를 들어, 콘솔을 사용하여 스택을 생성할 경우, 각 결과 스택 이벤트는 `Console-CreateStack-7f59c3cf-00d2-40c7-b2ff-e75db0987002` 형식의 동일한 토큰에 할당됩니다.

Example예: 리소스 상태 변경 이벤트

다음은 리소스 상태 이벤트의 예입니다. 이 이벤트는 CloudFormation이 요청된 리소스인 Amazon S3 버킷을 지정된 스택에 성공적으로 생성했음을 자세히 설명합니다.

```
{
  "version":"0",
  "id":"6a7e8feb-b491-4cf7-a9f1-bf3703467718",
  "detail-type":"CloudFormation Resource Status Change",
  "source":"aws.cloudformation",
  "account":"111122223333",
  "time":"2017-12-22T18:43:48Z",
  "region":"us-west-1",
  "resources":[
    "arn:aws:cloudformation:us-west-1:111122223333:stack/teststack"
  ],
  "detail":{
    "stack-id":"arn:aws:cloudformation:us-west-1:111122223333:stack/teststack",
    "logical-resource-id":"my-s3-bucket",
    "physical-resource-id":"arn:aws:s3:::my-s3-bucket-us-east-1",
    "status-details":{
      "status":"CREATE_COMPLETE",
      "status-reason":""
    },
    "resource-type":"AWS::S3::Bucket",
    "client-request-token":""
  }
}
```

스택 상태 변경 이벤트 세부 정보

다음은 스택 상태 변경 이벤트의 세부 정보 필드입니다.

source 및 detail-type 필드는 이벤트에 대한 특정 값을 포함하므로 포함됩니다.

```
{
  . . . ,
  "detail-type":"CloudFormation Stack Status Change",
  "source":"aws.cloudformation",
  . . . ,
  "detail":{
    "stack-id":"string",
    "status-details":{
      "status":"string",
      "status-reason":"string"
    },
    "client-request-token":"string"
  }
}
```

```
}
```

detail-type

이벤트의 유형을 식별합니다.

스택 상태 이벤트의 경우 이 값은 CloudFormation Stack Status Change입니다.

source

이벤트를 생성한 서비스를 식별합니다. CloudFormation 이벤트의 경우 이 값은 `aws.cloudformation`입니다.

detail

이벤트에 대한 정보를 포함하는 JSON 객체입니다. 이벤트를 생성하는 서비스에 따라 이 필드의 내용이 결정됩니다.

스택 상태 이벤트의 경우 이 데이터는 다음을 포함합니다.

stack-id

스택과 연결된 고유한 스택 ID입니다.

status-details

status

스택의 상태입니다.

스택 상태 코드의 전체 목록은 [스택 상태 코드](#) 섹션을 참조하세요.

status-reason

리소스의 상태 설명입니다.

client-request-token

API를 호출하는 데 사용되는 액세스 토큰입니다. 지정된 스택 작업에 의해 시작되는 모든 이벤트는 동일한 클라이언트 요청 토큰에 할당됩니다. 이 토큰은 작업을 추적하는 데 사용할 수 있습니다. 콘솔에서 시작되는 스택 작업은 `Console-StackOperation-ID` 토큰 형식을 사용합니다. 이러한 형식은 스택 작업을 쉽게 식별하는 데 도움이 됩니다. 예를 들어, 콘솔을 사용하여 스택을 생성할 경우, 각 결과 스택 이벤트는 `Console-CreateStack-7f59c3cf-00d2-40c7-b2ff-e75db0987002` 형식의 동일한 토큰에 할당됩니다.

Example예: 스택 상태 이벤트

다음은 CloudFormation이 요청된 스택인 teststack을(를) 성공적으로 생성한 스택 상태 이벤트의 예입니다.

```
{
  "version":"0",
  "id":"6a7e8feb-b491-4cf7-a9f1-bf3703467718",
  "detail-type":"CloudFormation Stack Status Change",
  "source":"aws.cloudformation",
  "account":"111122223333",
  "time":"2017-12-22T18:43:48Z",
  "region":"us-west-1",
  "resources":[
    "arn:aws:cloudformation:us-west-1:111122223333:stack/teststack"
  ],
  "detail":{
    "stack-id":"arn:aws:cloudformation:us-west-1:111122223333:stack/teststack",
    "status-details":{
      "status":"CREATE_COMPLETE",
      "status-reason":""
    },
    "client-request-token":""
  }
}
```

드리프트 감지 상태 변경 이벤트 세부 정보

다음은 스택 드리프트 감지 이벤트의 세부 정보 필드입니다.

source 및 detail-type 필드는 이벤트에 대한 특정 값을 포함하므로 포함됩니다.

```
{
  . . . ,
  "detail-type":"CloudFormation Drift Detection Status Change",
  "source":"aws.cloudformation",
  . . . ,
  "detail":{
    "stack-id":"string",
    "stack-drift-detection-id":"string",
    "status-details":{
      "stack-drift-status":"string",
```

```
    "detection-status":"string"
  },
  "drift-detection-details":{
    "drifted-stack-resource-count":integer
  },
  "client-request-token":"string"
}
```

detail-type

이벤트의 유형을 식별합니다.

스택 드리프트 감지 이벤트의 경우 이 값은 CloudFormation Drift Detection Status Change입니다.

source

이벤트를 생성한 서비스를 식별합니다. CloudFormation 이벤트의 경우 이 값은 aws.cloudformation입니다.

detail

이벤트에 대한 정보를 포함하는 JSON 객체입니다. 이벤트를 생성하는 서비스에 따라 이 필드의 내용이 결정됩니다.

스택 드리프트 감지 이벤트의 경우 이 데이터는 다음을 포함합니다.

stack-id

스택이 연결되어 있는 고유한 스택 ID입니다.

stack-drift-detection-id

스택 드리프트 감지 ID입니다.

status-details

stack-drift-status

스택의 드리프트 상태입니다.

detection-status

드리프트 감지 작업의 상태입니다.

drift-detection-details**drifted-stack-resource-count**

드리프트된 리소스 수입니다. 값이 -1인 경우 드리프트 감지가 진행 중입니다. 음수가 아닌 다른 모든 정수는 드리프트된 리소스의 실제 수를 나타냅니다.

client-request-token

API를 호출하는 데 사용되는 액세스 토큰입니다. 지정된 스택 작업에 의해 시작되는 모든 이벤트는 동일한 클라이언트 요청 토큰에 할당됩니다. 이 토큰은 작업을 추적하는 데 사용할 수 있습니다. 콘솔에서 시작되는 스택 작업은 Console-StackOperation-ID 토큰 형식을 사용합니다. 이러한 형식은 스택 작업을 쉽게 식별하는 데 도움이 됩니다. 예를 들어, 콘솔을 사용하여 스택을 생성할 경우, 각 결과 스택 이벤트는 Console-CreateStack-7f59c3cf-00d2-40c7-b2ff-e75db0987002 형식의 동일한 토큰에 할당됩니다.

Example예: 스택 드리프트 감지 이벤트

다음은 스택 드리프트 감지 이벤트의 예입니다. 이 이벤트는 CloudFormation이 지정된 스택에서 드리프트 감지를 완료했으며 현재 하나의 드리프트된 리소스로 인해 스택의 드리프트 상태가 DRIFTED임을 자세히 설명합니다.

```
{
  "version": "0",
  "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
  "detail-type": "CloudFormation Drift Detection Status Change",
  "source": "aws.cloudformation",
  "account": "111122223333",
  "time": "2017-12-22T18:43:48Z",
  "region": "us-west-1",
  "resources": ["string"],
  "detail": {
    "stack-id": "arn:aws:cloudformation:us-west-1:111122223333:stack/teststack",
    "stack-drift-detection-id": "624af370-311a-11e8-b6b7-500cexample",
    "status-details": {
      "stack-drift-status": "DRIFTED",
      "detection-status": "DETECTION_COMPLETE"
    },
    "drift-detection-details": {
      "drifted-stack-resource-count": 1
    },
    "client-request-token": ""
  }
}
```

```
}
}
```

StackSet 상태 변경 이벤트 세부 정보

다음은 StackSet 상태 변경 이벤트의 세부 정보 필드입니다.

source 및 detail-type 필드는 이벤트에 대한 특정 값을 포함하므로 포함됩니다.

```
{
  . . . ,
  "detail-type": "CloudFormation StackSet Status Change",
  "source": "aws.cloudformation",
  . . . ,
  "detail": {
    "stack-set-arn" : "string",
    "status-details": {
      "status": "string"
    }
  }
}
```

detail-type

이벤트의 유형을 식별합니다.

StackSet 상태 이벤트 이벤트의 경우 이 값은 CloudFormation StackSet Status Change입니다.

source

이벤트를 생성한 서비스를 식별합니다. CloudFormation 이벤트의 경우 이 값은 aws.cloudformation입니다.

detail

이벤트에 대한 정보를 포함하는 JSON 객체입니다. 이벤트를 생성하는 서비스에 따라 이 필드의 내용이 결정됩니다.

StackSet 상태 이벤트 이벤트의 경우 이 데이터는 다음을 포함합니다.

stack-set-arn

스택 세트와 연결된 Amazon 리소스 이름(ARN)입니다.

status-details

status

StackSet 상태입니다.

유효한 값: ACTIVE | DELETED

Example예: StackSet 상태 변경 이벤트

다음은 StackSet 상태 변경 이벤트의 예입니다. 이 이벤트는 CloudFormation이 지정된 스택 세트를 삭제했음을 자세히 설명합니다.

```
{
  "version": "0",
  "id": "42h6hb90-hg0w-11op-b01v-0xhnh0934z09",
  "detail-type": "CloudFormation StackSet Status Change",
  "source": "aws.cloudformation",
  "account": "111122223333",
  "time": "2021-09-23T17:06:18Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:cloudformation:us-east-1:111122223333:stackset/test12345:3f3a3fbe-c937-4eb3-a87d-e36a0af3f663"
  ],
  "detail": {
    "stack-set-arn" : "arn:aws:cloudformation:us-east-1:111122223333:stackset/test12345:3f3a3fbe-c937-4eb3-a87d-e36a0af3f663",
    "status-details": {
      "status":"DELETED"
    }
  }
}
```

StackSet 스택 인스턴스 상태 변경 이벤트 세부 정보

다음은 StackSet 스택 인스턴스 상태 이벤트의 세부 정보 필드입니다.

source 및 detail-type 필드는 이벤트에 대한 특정 값을 포함하므로 포함됩니다.

```
{
  . . . ,
```

```

"detail-type": "CloudFormation StackSet StackInstance Status Change",
"source": "aws.cloudformation",
. . .,
"detail": {
  "stack-set-arn" : "string",
  "stack-id" : "string",
  "action" : "string",
  "status-details": {
    "status": "string",
    "status-reason": "string",
    "detailed-status": "string"
  }
}
}
}

```

detail-type

이벤트의 유형을 식별합니다.

StackSet 스택 인스턴스 상태 이벤트의 경우 이 값은 CloudFormation StackSet StackInstance Status Change입니다.

source

이벤트를 생성한 서비스를 식별합니다. CloudFormation 이벤트의 경우 이 값은 aws.cloudformation입니다.

detail

이벤트에 대한 정보를 포함하는 JSON 객체입니다. 이벤트를 생성하는 서비스에 따라 이 필드의 내용이 결정됩니다.

StackSet 스택 인스턴스 상태 이벤트의 경우 이 데이터는 다음을 포함합니다.

stack-set-arn

StackSet와 연결된 Amazon 리소스 이름(ARN)입니다.

stack-id

스택 인스턴스와 연결된 고유한 스택 ID입니다.

action

스택 세트 작업의 유형.

유효한 값: CREATE | UPDATE | DELETE | DETECT_DRIFT

status-details

status

StackSet 인스턴스 상태입니다.

자세한 내용은 [스택 인스턴스 상태 코드](#) 섹션을 참조하세요.

유효한 값: CURRENT | OUTDATED | INOPERABLE

status-reason

StackSet 인스턴스의 상태 사유입니다.

detailed-status

자세한 StackSet 인스턴스 세부 상태입니다.

유효한 값: CANCELLED | FAILED | FAILED_IMPORT | INOPERABLE | PENDING | RUNNING
| SKIPPED_SUSPENDED_ACCOUNT | SUCCEEDED

Example예: StackSet 스택 인스턴스 상태 변경 이벤트

다음은 StackSet 스택 인스턴스 상태 변경 이벤트의 예입니다.

```
{
  "version": "0",
  "id": "42h6hb90-hg0w-11op-b01v-0xhnh0934z09",
  "detail-type": "CloudFormation StackSet StackInstance Status Change",
  "source": "aws.cloudformation",
  "account": "111122223333",
  "time": "2021-09-22T19:19:23Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:cloudformation:us-east-1:111122223333:stackset/test1234:e5f54eea-d041-44ad-94f8-b8268aca1e59"
  ],
  "detail": {
    "stack-set-arn": "arn:aws:cloudformation:us-east-1:111122223333:stackset/test1234:e5f54eea-d041-44ad-94f8-b8268aca1e59",
    "stack-id": "arn:aws:cloudformation:us-west-1:111122223333:stack/teststack",
    "status-details": {
      "status": "OUTDATED",
```

```

        "status-reason": "User Initiated",
        "detailed-status": "PENDING"
    }
}
}

```

StackSet 작업 상태 변경 이벤트 세부 정보

다음은 StackSet 작업 상태 변경 이벤트의 세부 정보 필드입니다.

source 및 detail-type 필드는 이벤트에 대한 특정 값을 포함하므로 포함됩니다.

```

{
  . . . ,
  "detail-type": "CloudFormation StackSet Operation Status Change",
  "source": "aws.cloudformation",
  . . . ,
  "detail": {
    "stack-set-arn" : "string",
    "stack-set-operation-id" : "string",
    "status-details": {
      "status": "string"
    }
  }
}
}
}

```

detail-type

이벤트의 유형을 식별합니다.

StackSet 작업 상태 이벤트의 경우 이 값은 CloudFormation StackSet Operation Status Change입니다.

source

이벤트를 생성한 서비스를 식별합니다. CloudFormation 이벤트의 경우 이 값은 aws.cloudformation입니다.

detail

이벤트에 대한 정보를 포함하는 JSON 객체입니다. 이벤트를 생성하는 서비스에 따라 이 필드의 내용이 결정됩니다.

StackSet 작업 상태 이벤트의 경우 이 데이터는 다음을 포함합니다.

`stack-set-arn`

StackSet와 연결된 Amazon 리소스 이름(ARN)입니다.

`stack-set-operation-id`

StackSet 작업과 연결된 고유 ID입니다.

`status-details`

`status`

StackSet 작업 상태

자세한 내용은 [StackSets 상태 코드](#) 섹션을 참조하세요.

유효한 값: RUNNING | SUCCEEDED | FAILED | STOPPING | STOPPED | QUEUED

Example예: StackSet 작업 상태 변경 이벤트

다음은 StackSet 작업 상태 변경 이벤트의 예입니다. 이벤트는 CloudFormation이 지정된 스택 세트에서 요청된 작업을 성공적으로 완료했음을 자세히 설명합니다.

```
{
  "version": "0",
  "id": "4de89905-fd92-6a6b-9509-23c04bcb6a21",
  "detail-type": "CloudFormation StackSet Operation Status Change",
  "source": "aws.cloudformation",
  "account": "111122223333",
  "time": "2021-09-22T05:46:24Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:cloudformation:us-east-1:111122223333:stackset/test1234:e5f54eea-d041-44ad-94f8-b8268aca1e59"
  ],
  "detail": {
    "stack-set-arn": "arn:aws:cloudformation:us-east-1:111122223333:stackset/test1234:e5f54eea-d041-44ad-94f8-b8268aca1e59",
    "stack-set-operation-id": "ce69adce-2221-4483-8c4b-c51f284f25e8",
    "status-details": {
      "status": "SUCCEEDED"
    }
  }
}
```

}

리포지토리 동기화 상태 변경 이벤트 세부 정보

다음은 리포지토리 동기화 상태 변경 이벤트의 세부 정보 필드입니다.

source 및 detail-type 필드는 이벤트에 대한 특정 값을 포함하므로 포함됩니다.

```
{
  . . . ,
  "detail-type": "Git Sync Repository Sync Status Change",
  "source": "aws.codeconnections",
  . . . ,
  "detail": {
    "connectionArn" : "string",
    "providerType" : "string",
    "repositoryName": "string",
    "providerType": "string",
    "repositoryName": "string",
    "repositoryArn": "string",
    "repositoryLinkId": "string",
    "ownerId": "string",
    "commit": "string",
    "branch": "string",
    "syncType": "string",
    "status": "string",
    "previousSync": "string"
  }
}
```

detail-type

이벤트의 유형을 식별합니다.

리포지토리 동기화 상태 이벤트의 경우 이 값은 Git Sync Repository Sync Status Change입니다.

source

이벤트를 생성한 서비스를 식별합니다. Git 동기화 이벤트의 경우 이 값은 aws.codeconnections입니다.

detail

이벤트에 대한 정보를 포함하는 JSON 객체입니다. 이벤트를 생성하는 서비스에 따라 이 필드의 내용이 결정됩니다.

리포지토리 동기화 상태 이벤트의 경우 이 데이터는 다음을 포함합니다.

`connectionArn`

CodeConnections와 연결된 Amazon 리소스 이름(ARN).

`providerType`

CloudFormation에 연결된 Git 공급자.

유효한 값: GitHub | GitHub Enterprise | GitLab | BitBucket

`repositoryName`

Git 리포지토리 이름.

`repositoryArn`

Git 리포지토리 연결된 ARN.

`repositoryLinkId`

리포지토리 링크와 연결된 고유한 ID.

`ownerId`

리포지토리 소유자와 연결된 고유한 ID.

`commit`

리포지토리 커밋과 연결된 고유한 ID.

`branch`

리포지토리 브랜치와 연결된 고유한 ID.

`syncType`

수행 중인 동기화 유형.

`status`

현재 리포지토리 동기화 상태.

유효한 값: FAILED | INITIATED | IN_PROGRESS | SUCCEEDED

previousSync

현재 상태 이전의 동기화 상태.

유효한 값: FAILED | INITIATED | IN_PROGRESS | SUCCEEDED

Example예: 리포지토리 동기화 상태 변경 이벤트

다음은 리포지토리 동기화 상태 변경 이벤트의 예입니다. 이벤트는 CodeConnections가 리포지토리를 성공적으로 동기화했다는 세부 정보를 제공합니다.

```
{
  "version": "0",
  "id": "1b5d8feb-agbv-4cf7-a9f1-bf3703467718",
  "detail-type": "GitSync Repository Sync Status Change",
  "source": "aws.codeconnections",
  "account": "111122223333",
  "time": "2023-12-22T18:43:48Z",
  "region": "us-east-1",
  "resources": ["arn:aws:aws:codestar-connections:us-east-1:111122223333:repository-link/550e8400-e29b-41d4-a716-446655440000"],
  "detail": {
    "connectionArn": "arn:aws:codestar-connections:us-east-1:111122223333:connection/sample-connection-id",
    "providerType": "GitHub",
    "repositoryName": "sample-repository-name",
    "repositoryArn": "arn:aws:aws:codestar-connections:us-east-1:111122223333:repository-link/550e8400-e29b-41d4-a716-446655440000",
    "repositoryLinkId": "550e8400-e29b-41d4-a716-446655440000",
    "ownerId": "sample-owner-id",
    "commit": "sample-commit-id",
    "branch": "main",
    "syncType": "CFN_STACK_SYNC",
    "status": "SUCCEEDED",
    "previousStatus": "IN_PROGRESS",
  }
}
```

리소스 동기화 상태 변경 이벤트 세부 정보

다음은 리소스 동기화 상태 변경 이벤트의 세부 정보 필드입니다.

source 및 detail-type 필드는 이벤트에 대한 특정 값을 포함하므로 포함됩니다.

```
{
  . . . ,
  "detail-type": "Git Sync Repository Sync Status Change",
  "source": "aws.codeconnections",
  . . . ,
  "detail": {
    "providerType" : "string",
    "commit" : "string",
    "repositoryName": "string",
    "branch": "string",
    "syncType": "string",
    "syncTarget": "string",
    "status": "string",
    "previousSync": "string"
  }
}
```

detail-type

이벤트의 유형을 식별합니다.

리포지토리 동기화 상태 이벤트의 경우 이 값은 Git Sync Repository Sync Status Change입니다.

source

이벤트를 생성한 서비스를 식별합니다. Git 동기화 이벤트의 경우 이 값은 aws.codeconnections입니다.

detail

이벤트에 대한 정보를 포함하는 JSON 객체입니다. 이벤트를 생성하는 서비스에 따라 이 필드의 내용이 결정됩니다.

리소스 동기화 상태 이벤트의 경우 이 데이터는 다음을 포함합니다.

providerType

CloudFormation에 연결된 Git 공급자.

유효한 값: GitHub | GitHub Enterprise | GitLab | BitBucket

`commit`

리포지토리 커밋과 연결된 고유한 ID.

`repositoryName`

Git 리포지토리의 이름.

`branch`

리포지토리 브랜치와 연결된 고유한 ID.

`syncType`

수행 중인 동기화 유형.

`syncTarget`

리소스 동기화의 대상 스택.

`status`

현재 리포지토리 동기화 상태.

유효한 값: FAILED | INITIATED | IN_PROGRESS | SUCCEEDED

`previousSync`

현재 상태 이전의 동기화 상태.

유효한 값: FAILED | INITIATED | IN_PROGRESS | SUCCEEDED

Example에: 리소스 동기화 상태 변경 이벤트

다음은 리소스 동기화 상태 변경 이벤트의 예입니다. 이벤트는 CodeConnections가 리소스를 성공적으로 동기화했다는 세부 정보를 제공합니다.

```
{
  "version": "0",
  "id": "1b5d8feb-agbv-4cf7-a9f1-bf3703467718",
  "detail-type": "Git Sync Resource Sync Status Change",
  "source": "aws.codeconnections",
  "account": "111122223333",
  "time": "2023-12-22T18:43:48Z",
  "region": "us-east-1",
  "resources": ["arn:aws:aws:cloudformation:us-east-1:111122223333:stack/targetStack1"],
}
```

```
"detail": {
  "providerType": "GitHub",
  "commit": "sample-commit-id",
  "repositoryName": "sample-repository-name",
  "branch": "main",
  "syncType": "CFN_STACK_SYNC",
  "syncTarget": "arn:aws:aws:cloudformation:us-east-1:111122223333:stack/
targetStack1",
  "status": "SUCCEEDED",
  "previousStatus": "IN_PROGRESS"
}
}
```

CloudFormation 할당량 이해

AWS 계정에는 템플릿을 작성하고 스택을 생성할 때 알아야 하는 CloudFormation 할당량이 있습니다. 이러한 할당량을 이해하면 템플릿 또는 스택을 재설계해야 하는 제한 오류를 방지할 수 있습니다.

다음 표는 CloudFormation 할당량을 보여줍니다.

할당량	설명	값	튜닝 전략
cfn-signal 대기 조건 데이터	cfn-signal 에서 전달할 수 있는 최대 데이터 크기.	4,096바이트	많은 양을 전달하려면 데이터를 Amazon S3 버킷에 보낸 다음 cfn-signal 을 사용하여 Amazon S3 URL 을 해당 버킷에 전달합니다.
사용자 지정 리소스 응답	사용자 지정 리소스 제공업체가 전달할 수 있는 최대 데이터 크기.	4,096바이트	
템플릿당 동적 참조	단일 CloudFormation 스택 템플릿에 허용되는 최대 동적 참조 수입니다.	스택 템플릿의 동적 참조 최대 60개	
계정당 후크 수	계정별 리전당 허용되는 최대 후크 수.	후크 100개	
리소스당 후크 수	리소스당 구성할 수 있는 최대 후크 수.	후크 100개	
후크 구성 크기	후크에서 저장할 수 있는 최대 구성 데이터 크기.	204.8KB	

할당량	설명	값	튜닝 전략
매핑	CloudFormation 템플릿에서 선언할 수 있는 최대 매핑 수.	200개 매핑	더 많은 매핑을 지정하려면 중첩 스택 등을 사용하여 템플릿을 여러 템플릿으로 분리합니다.
매핑 속성	CloudFormation 템플릿에서 선언할 수 있는 각 매핑의 최대 매핑 속성 수.	200개 속성	더 많은 매핑 속성을 지정하려면 속성을 여러 매핑으로 분리합니다.
매핑 이름 및 매핑 속성 이름	각 매핑 이름의 최대 크기.	255자	
모듈	계정 및 리전별로 CloudFormation 레지스트리에 등록할 수 있는 최대 모듈 수.	모듈 100개	
모듈 버전	지정된 모듈에 대해 CloudFormation 레지스트리에 등록할 수 있는 최대 버전 수.	버전 100개	새 버전을 등록하려면 먼저 DeregisterType 을 사용하여 더 이상 사용하지 않는 버전을 해제하세요.

할당량	설명	값	튜닝 전략
중첩 스택	중첩 스택이 작업당 생성, 업데이트 또는 삭제할 수 있는 최대 CloudFormation 리소스 수. 예를 들어 총 리소스가 2,500개를 초과하는 중첩 스택 계층 구조를 보유할 수 있지만 단일 배포에서는 이 2,500개를 초과하는 리소스를 생성, 업데이트 또는 삭제할 수 없습니다.	2,500개 리소스	스택 계층 구조를 여러 스택으로 분할합니다.
출력	CloudFormation 템플릿에서 선언할 수 있는 최대 출력 수.	200개 출력	
출력 이름	출력 이름의 최대 크기.	255자	
파라미터	CloudFormation 템플릿에서 선언할 수 있는 최대 파라미터 수.	200개 파라미터	더 많은 파라미터를 지정하려면 매핑 또는 목록을 사용하여 단일 파라미터에 여러 값을 할당할 수 있습니다.
파라미터 이름	파라미터 이름의 최대 크기.	255자	

할당량	설명	값	튜닝 전략
파라미터값	파라미터 값의 최대 크기.	4,096바이트	더 큰 파라미터화된 값을 사용하려면 여러 파라미터를 생성한 다음 Fn::Join 함수를 사용하여 여러 값을 단일 값으로 연결합니다.
프라이빗 리소스	계정 및 리전별로 CloudFormation 레지스트리에 등록할 수 있는 최대 프라이빗 리소스 수.	프라이빗 리소스 50개	
프라이빗 리소스 버전	지정된 프라이빗 리소스에 대해 CloudFormation 레지스트리에 등록할 수 있는 최대 버전 수.	프라이빗 리소스 50개	새 버전을 등록하려면 먼저 DeregisterType 을 사용하여 더 이상 사용하지 않는 버전을 해제하세요.
리소스	CloudFormation 템플릿에서 선언할 수 있는 최대 리소스 수.	500개 리소스	더 많은 리소스를 지정하려면 중첩 스택 등을 사용하여 템플릿을 여러 템플릿으로 분리합니다.
동시 스택 작업의 리소스	지정된 시간에 리전에서 스택 작업(생성, 업데이트 또는 삭제 작업)에 포함할 수 있는 최대 리소스 수.	DescribeAccountLimits API를 사용하여 특정 리전의 계정에 대한 현재 한도를 결정합니다.	
리소스 이름	리소스 이름의 최대 크기.	255자	

할당량	설명	값	튜닝 전략
스택	생성할 수 있는 최대 CloudFormation 스택 수.	스택 2,000개	더 많은 스택을 생성하려면 필요 없는 스택을 삭제하거나 AWS 계정의 최대 스택 수를 늘립니다. 자세한 내용은 AWS 일반 참조의 AWS Service Quotas 를 참조하세요.
스택 이름	스택 이름의 최대 크기.	128자	
StackSets	관리자 계정에서 생성할 수 있는 최대 CloudFormation 스택 세트 수.	스택 세트 1,000개	더 많은 스택 세트를 생성하려면 필요 없는 스택 세트를 삭제하거나 AWS 계정의 최대 스택 세트 수를 늘리도록 요청합니다. 자세한 내용은 AWS 일반 참조의 AWS Service Quotas 를 참조하세요.
스택 인스턴스	스택 세트당 생성할 수 있는 최대 스택 인스턴스 수.	스택 세트당 스택 인스턴스 100,000개	더 많은 스택 인스턴스를 생성하려면 필요 없는 스택 인스턴스를 삭제하거나 AWS 계정의 최대 스택 인스턴스 수를 늘리도록 요청합니다. 자세한 내용은 AWS 일반 참조의 AWS Service Quotas 를 참조하세요.

할당량	설명	값	튜닝 전략
StackSets 인스턴스 작업	관리자 계정당 각 리전에서 동시에 작업을 실행할 수 있는 최대 스택 인스턴스 수(모든 스택 세트에 있는 스택 인스턴스).	작업 10,000개	이 한도는 리전에 관련된 모든 스택 세트에 적용됩니다. 스택 인스턴스의 직접 생성, 업데이트 또는 삭제는 물론 스택 세트 생성 및 업데이트 작업의 영향을 받는 스택 인스턴스가 포함됩니다.
StackSets 대기 작업입니다.	지정된 시간에 스택 세트에 대해 대기 중인 최대 작업 수.	작업 10,000개	
스택 세트 작업당 S3 객체를 사용하여 가져온 스택 수	스택 세트 작업당 S3 객체를 사용하여 가져올 수 있는 최대 스택 수.	200개 스택	
스택 세트 작업당 인라인 스택 ID를 사용하여 가져온 스택 수	스택 세트 작업당 인라인 스택 ID를 사용하여 가져올 수 있는 최대 스택 수.	10개 스택	
요청의 템플릿 본문 크기	CreateStack , UpdateStack 또는 ValidateTemplate 요청에서 전달할 수 있는 최대 템플릿 본문 크기.	51,200 bytes	더 큰 템플릿 본문을 사용하려면 중첩 스택 등을 사용하여 템플릿을 여러 템플릿으로 분리합니다. 또는 템플릿을 Amazon S3 버킷에 업로드합니다.

할당량	설명	값	튜닝 전략
Amazon S3 객체의 템플릿 본문 크기	Amazon S3 객체에서 Amazon S3 템플릿 URL을 사용하여 CreateStack , UpdateStack , ValidateTemplate 요청에 대해 전달할 수 있는 최대 템플릿 본문 크기.	1MB	더 큰 템플릿 본문을 사용하려면 중첩 스택 등을 사용하여 템플릿을 여러 템플릿으로 분리합니다. 또는 축소를 사용하여 CloudFormation 템플릿 크기를 줄입니다.
템플릿 설명	템플릿 설명의 최대 크기.	1,024 bytes	
후크당 버전 수	후크당 생성할 수 있는 최대 버전 수.	버전 100개	

기능 가용성

일부 리전에서는 CloudFormation의 일부 기능을 사용하지 못할 수 있습니다. AWS 리전에 대한 자세한 내용은 [글로벌 인프라 리전 표](#)를 참조하세요.

- 현재 다음 리전에서는 [매크로](#)를 사용할 수 없습니다.
 - 아시아 태평양(자카르타)
- 현재, 다음 리전에서는 [CloudFormation을 사용하여 CodeDeploy를 통한 ECS 블루/그린 배포 수행](#)을 사용할 수 없습니다.
 - 아프리카(케이프타운)
 - 아시아 태평양(오사카)
 - 유럽(밀라노)

StackSets 및 매크로

StackSets는 현재 매크로가 포함된 템플릿에서 서비스 관리형 권한으로 스택 세트를 생성하거나 업데이트하는 것을 지원하지 않습니다. 여기에는 CloudFormation에서 호스트하는 매크로인 변환이 포함됩니다. 매크로에 대한 자세한 내용은 [템플릿 매크로](#) 단원을 참조하십시오.

CloudFormation 문제 해결

CloudFormation을 사용할 때 CloudFormation 스택 생성, 업데이트 또는 삭제 시 문제가 발생할 수 있습니다. 다음 단원은 발생할 수 있는 몇 가지 일반적인 문제를 해결하는 데 도움이 될 수 있습니다.

CloudFormation에 대한 일반적인 질문은 [CloudFormation FAQ](#)를 참조하세요. AWS re:Post의 [CloudFormation 커뮤니티](#)에서 답변을 검색하고 질문을 게시할 수도 있습니다.

주제

- [문제 해결 가이드](#)
- [오류 해결](#)
- [Support에 문의](#)

문제 해결 가이드

CloudFormation에서 스택 생성, 업데이트 또는 삭제에 실패하면 오류 메시지 또는 로그를 보고 해당 문제에 대한 자세한 내용을 확인할 수 있습니다. 다음 태스크는 CloudFormation 문제 해결을 위한 일반적인 방법을 설명합니다. 구체적인 오류와 해결책에 대해서는 [오류 해결](#) 단원을 참조하십시오.

- 대상 문제 해결을 위해 작업 ID 및 장애 필터링을 사용합니다. CloudFormation은 배포 이벤트에 고유한 작업 ID를 할당하므로 이를 통해 관련 이벤트를 더 쉽게 추적하고 상호 연관시킬 수 있습니다. 작업 ID별로 스택 이벤트를 필터링하여 특정 배포 단계에 중점을 두고 장애 필터를 사용하여 배포 장애의 근본 원인을 빠르게 식별할 수 있습니다. [작업별 스택 이벤트 보기](#) 섹션을 참조하세요.
- CloudFormation 콘솔을 사용하여 스택 상태를 볼 수 있습니다. 콘솔에서는 스택 생성, 업데이트 또는 삭제 중 스택 이벤트 목록을 볼 수 있습니다. 스택 이벤트에는 관련 이벤트를 상호 연관시키고 개별 작업을 추적하는 데 도움이 되는 작업 ID가 포함됩니다. 집중적인 문제 해결을 위해 작업 ID를 클릭할 수 있습니다. 이 목록에서 장애 이벤트를 찾은 다음 해당 이벤트에 대한 상태 사유를 확인할 수 있습니다. 상태 사유에는 문제 해결에 도움이 될 수 있는 CloudFormation 또는 특정 서비스의 오류 메시지가 포함되어 있을 수 있습니다. 작업 ID로 스택 이벤트를 보는 방법에 대한 자세한 내용은 [작업별 스택 이벤트 보기](#) 섹션을 참조하세요.
- Amazon EC2 문제의 경우 cloud-init 및 cfn 로그를 확인하십시오. Amazon EC2 인스턴스에 대한 이러한 로그는 /var/log/ 디렉터리에 게시됩니다. 이러한 로그는 CloudFormation에서 인스턴스를 설정하는 동안 발생하는 프로세스 및 명령 출력을 캡처합니다. Windows의 경우 %ProgramFiles%\Amazon\EC2ConfigService에서 EC2Configure 서비스를 보고, %ProgramData%\Amazon\EC2-Windows\Launch\Logs에서 EC2 Launch를 보고,

%ProgramData%\Amazon\EC2Launch\log에서 EC2 Launch v2를 보고, C:\cfn\log에서 cfn 로그를 봅니다.

또한 AWS Management Console에 로그를 표시하는 Amazon CloudWatch에 로그가 게시되도록 CloudFormation 템플릿을 구성할 수 있습니다. 그러면 Amazon EC2 인스턴스에 연결할 필요가 없습니다. 자세한 내용은 Application Management 블로그의 [View CloudFormation Logs in the Console](#)을 참조하세요.

오류 해결

CloudFormation 스택과 관련해 다음 오류가 발생하면 아래 해결책에 따라 문제의 원인을 찾아 해결할 수 있습니다.

주제

- [스택 삭제 실패](#)
- [종속성 오류](#)
- [AWS Config 및 AWS Systems Manager 충돌](#)
- [목록 전달 시 발생하는 파라미터 구분 분석 오류](#)
- [IAM 권한 부족](#)
- [잘못된 값 또는 지원되지 않는 리소스 속성](#)
- [할당량 초과](#)
- [중첩 스택이 UPDATE_COMPLETE_CLEANUP_IN_PROGRESS, UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS 또는 UPDATE_ROLLBACK_IN_PROGRESS 상태에서 정체됨](#)
- [수행할 업데이트 없음](#)
- [스택 생성, 업데이트 또는 삭제 작업 중 리소스 안정화에 실패](#)
- [VPC에 보안 그룹이 없음](#)
- [업데이트 롤백에 실패](#)
- [대기 조건이 Amazon EC2 인스턴스에서 필요한 수의 신호를 수신하지 못함](#)
- [리소스가 스택에서 제거되었지만 삭제되지 않음](#)

스택 삭제 실패

이러한 상황을 해결하려면 다음 작업을 시도해 보십시오.

- 삭제하려면 일부 리소스는 비어 있어야 합니다. 예를 들어, 버킷 또는 보안 그룹을 삭제하려면 Amazon S3 버킷의 객체를 모두 삭제하거나 Amazon EC2 보안 그룹의 인스턴스를 모두 제거해야 합니다.
- 스택의 리소스를 삭제하려면 필수 IAM 권한이 있어야 합니다. CloudFormation 권한 이외에 기본 서비스(예: Amazon S3 또는 Amazon EC2)를 사용할 수 있어야 합니다.
- CloudFormation에서 리소스를 삭제할 수 없어 스택이 DELETE_FAILED 상태인 경우 RetainResources 파라미터를 사용하여 [삭제](#)를 다시 실행한 다음 CloudFormation에서 삭제할 수 없는 리소스를 지정합니다. CloudFormation에서는 유지된 리소스를 삭제하지 않고 스택을 삭제합니다. 유지하려는 객체가 포함된 리소스(예: S3 버킷)를 삭제할 수 없지만 스택을 삭제하고자 하는 경우에 리소스를 보존하면 유용합니다.

스택을 삭제한 후에는 연결된 AWS 서비스를 사용하여 보존한 리소스를 수동으로 삭제할 수 있습니다.

또는 DeletionMode 파라미터를 포함하여 FORCE_DELETE_STACK 옵션을 사용하는 것도 고려할 수 있습니다. 스택의 강제 삭제에 대한 자세한 내용은 [CloudFormation 콘솔에서 스택 삭제](#)를 참조하세요.

- 종료 방지 기능이 활성화된 스택은 삭제할 수 없습니다. 종료 방지 기능이 활성화된 스택을 삭제하려고 시도하면 삭제가 실패하고 스택(상태 포함)은 변함없이 그대로 유지됩니다. 스택의 종료 방지 기능을 비활성화한 다음 삭제 작업을 다시 수행하십시오.

여기에는 루트 스택의 종료 방지 기능이 활성화된 [중첩 스택](#)이 포함됩니다. 루트 스택의 종료 방지 기능을 비활성화한 다음 삭제 작업을 다시 수행합니다. 중첩 스택은 직접 삭제하지 말고 루트 스택과 모든 리소스를 삭제하는 과정에서 삭제하는 것이 좋습니다.

자세한 내용은 [CloudFormation 스택 삭제 방지](#) 섹션을 참조하세요.

- 다른 모든 문제의 경우 AWS Support이 있으면 지원 사례를 생성할 수 있습니다. [Support에 문의](#)을 (를) 참조하세요.

종속성 오류

종속성 오류를 해결하려면 템플릿의 기타 리소스에 따라 리소스에 DependsOn 속성을 추가합니다. 경우에 따라 CloudFormation이 올바른 순서로 리소스를 생성 또는 삭제할 수 있도록 종속성을 명시적으로 선언해야 합니다. 예를 들어, 동일한 스택에서 인터넷 게이트웨이를 사용하여 탄력적 IP 및 VPC를 생성하는 경우 탄력적 IP는 인터넷 게이트웨이 연결에 따라 달라져야 합니다. 자세한 내용은 [DependsOn 속성](#)을 참조하세요.

AWS Config 및 AWS Systems Manager 충돌

AWS Config 및 AWS Systems Manager에서는 CloudFormation 스택 배포와 충돌하는 원인이 될 수 있는 인프라 관리 태스크를 자동화할 수 있습니다. 잠재적 충돌을 방지하는 방법은 다음과 같습니다.

- 연결된 AWS 계정 및 AWS 리전에서 AWS Config 및 Systems Manager의 구성을 검토합니다.
- CloudFormation 배포 중에 트리거될 수도 있는 활성 규칙 또는 자동화 문서를 확인합니다. 이러한 규칙과 문서가 잠재적으로 배포와 충돌하는 리소스 종속성 또는 충돌의 원인이 될 수 있습니다.
- CloudFormation 템플릿에서 AWS Config 및 Systems Manager에서 관리하는 리소스를 확인합니다. 잠재적인 중복 또는 상호 종속성을 확인하고 충돌이 방지되도록 템플릿 또는 자동화 구성을 조정해 봅니다.
- CloudFormation 배포 중에 관련 AWS Config 규칙 또는 Systems Manager 자동화를 일시적으로 비활성화하거나 일시 중지합니다. 배포 성공 후 원래 구성을 복원하여 원하는 자동화 및 규정 준수 수준을 유지 관리해야 합니다.
- 충돌 소스를 정확히 찾아내는 데 도움이 되도록 CloudFormation 로그에서 오류 메시지에서 AWS Config에 대한 참조와 Systems Manager 관련 문제를 검토합니다.

AWS Config 규칙에 대한 자세한 내용은 [AWS Config 규칙을 사용하여 리소스 평가](#)를 참조하세요.

Systems Manager 자동화에 대한 자세한 내용은 [AWS Systems Manager 자동화](#)를 참조하세요.

목록 전달 시 발생하는 파라미터 구분 분석 오류

AWS CLI 또는 을 사용하여 목록을 전달하는 경우 각 쉼표 앞에 이스케이프 문자(\)를 추가합니다. 다음 샘플은 AWS CLI를 사용하는 경우 입력 파라미터를 지정하는 방법을 보여줍니다.

```
ParameterKey=CIDR,ParameterValue='10.10.0.0/16\,10.10.0.0/24\,10.10.1.0/24'
```

IAM 권한 부족

CloudFormation 스택 작업을 수행하는 경우 CloudFormation 사용 권한뿐만 아니라 템플릿에 설명된 기본 서비스 사용 권한도 있어야 합니다. 예를 들어, Amazon S3 버킷을 생성하거나 Amazon EC2 인스턴스를 생성하는 경우 Amazon S3 또는 Amazon EC2에 대한 권한이 필요합니다. IAM 정책을 검토하고 CloudFormation 스택 작업을 수행하기 전에 필수 권한이 있는지 확인합니다. 자세한 내용은 [AWS Identity and Access Management를 사용하여 CloudFormation 액세스 제어](#) 단원을 참조하십시오.

잘못된 값 또는 지원되지 않는 리소스 속성

CloudFormation 스택을 생성 또는 업데이트하는 경우 잘못된 입력 파라미터, 지원되지 않는 리소스 속성 이름 또는 지원되지 않는 리소스 속성값으로 인해 스택에 오류가 발생할 수 있습니다. 입력 파라미터의 경우 리소스가 존재하는지 확인합니다. 예를 들어, Amazon EC2 키 페어 또는 VPC ID를 지정하는 경우 해당 리소스가 사용자 계정에 있어야 하고, 스택을 생성 또는 업데이트한 리전에 있어야 합니다. [AWS 고유 파라미터 유형](#)을 사용하여 올바른 값을 사용하는지 확인할 수 있습니다.

리소스 속성 이름 및 값의 경우 올바른 이름 및 값을 사용하도록 템플릿을 업데이트합니다. 자세한 내용은 [AWS 리소스 및 속성 유형 참조](#)를 참조하세요.

할당량 초과

리소스 할당량에 도달하지 않았는지 확인합니다. 예를 들어 시작할 수 있는 Amazon EC2 온디맨드 인스턴스의 최대 개수는 5개입니다. 계정 할당량보다 더 많은 Amazon EC2 온디맨드 인스턴스를 생성하려고 하면 인스턴스 생성에 실패하고 오류 Status=start_failed가 수신됩니다. 서비스별 기본 AWS 할당량을 보려면 AWS 일반 참조의 [AWS Service Quotas](#)를 참조하세요.

CloudFormation 할당량 및 조정 전략은 [CloudFormation 할당량 이해](#) 섹션을 참조하세요.

또한 업데이트 중에 리소스가 바뀌면 CloudFormation은 이전 리소스를 삭제하기 전에 새 리소스를 생성합니다. 이로 인해 계정이 리소스 할당량을 초과할 수 있고, 결국 업데이트에 실패할 수 있습니다. 초과 리소스를 삭제하거나 [할당량 증가](#)를 요청할 수 있습니다.

중첩 스택이 UPDATE_COMPLETE_CLEANUP_IN_PROGRESS, UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS 또는 UPDATE_ROLLBACK_IN_PROGRESS 상태에서 정체됨

중첩 스택 롤백에 실패합니다. 중첩 스택 간의 잠재적인 리소스 종속성으로 인해 모든 중첩 스택이 업데이트되거나 롤백될 때까지 CloudFormation에서 중첩 스택 리소스 정리를 시작하지 않습니다. 중첩 스택 롤백에 실패하면 CloudFormation은 나머지 중첩 스택의 상태와 관계없이 작업을 모두 취소합니다. 업데이트 또는 롤백을 완료했으나 다른 중첩 스택의 롤백 실패로 인해 CloudFormation으로부터 정리 시작 신호를 수신하지 못한 중첩 스택의 상태는 UPDATE_COMPLETE_CLEANUP_IN_PROGRESS 또는 UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS입니다. 업데이트에 실패했으나 롤백 신호를 수신하지 못한 중첩 스택의 상태는 UPDATE_ROLLBACK_IN_PROGRESS입니다.

스택 템플릿이 스택의 상태를 정확하게 반영하지 않은 경우 CloudFormation 외부에서 수행된 변경 사항으로 인해 중첩 스택 롤백에 실패할 수 있습니다. 또한 그룹이 생성 또는 업데이트된 경우 중첩 스택

의 Auto Scaling 그룹에 리소스 신호 제한 시간이 짧게 지정되어 있으면 중첩 스택에 실패할 수 있습니다.

스택을 수정하려면 [AWS Support](#)에 문의하세요.

수행할 업데이트 없음

CloudFormation 스택을 업데이트하려면 템플릿 또는 파라미터값 변경 사항을 CloudFormation에 제출해야 합니다. 그러나 CloudFormation은 일부 템플릿 변경 사항을 업데이트로 인식하지 않습니다(예: 삭제 정책, 업데이트 정책, 조건 선언 또는 출력 선언에 대한 변경 사항). 다른 변경 없이 이러한 변경을 수행해야 하는 경우 리소스에 대한 메타데이터 속성을 추가 또는 수정할 수 있습니다. 메타데이터 속성은 CloudFormation에서 해당 콘텐츠를 해석하지 않으므로 메타임의의 값일 수 있습니다. 자세한 내용은 [Metadata 속성](#)을 참조하세요.

스택 생성, 업데이트 또는 삭제 작업 중 리소스 안정화에 실패

작업이 CloudFormation 제한 시간을 초과했거나 AWS 서비스가 중단되어 리소스가 응답하지 않았습니다. 서비스 중단의 경우 관련 AWS 서비스가 실행 중인지 [확인](#)한 다음 스택 작업을 다시 시도합니다.

AWS 서비스가 성공적으로 실행된 경우에는 스택에 다음 리소스 중 하나가 포함되어 있는지 확인합니다.

- AWS::AutoScaling::AutoScalingGroup(생성, 업데이트 및 삭제 작업의 경우)
- AWS::CertificateManager::Certificate(생성 작업의 경우)
- AWS::CloudFormation::Stack(생성, 업데이트 및 삭제 작업의 경우)
- AWS::ElasticSearch::Domain(업데이트 작업의 경우)
- AWS::RDS::DBCluster(생성 및 업데이트 작업의 경우)
- AWS::RDS::DBInstance(생성, 업데이트 및 삭제 작업의 경우)
- AWS::Redshift::Cluster(업데이트 작업의 경우)

이러한 리소스에 대한 작업은 기본 제한 시간보다 오래 걸릴 수 있습니다. 제한 시간은 사용하는 리소스와 자격 증명에 따라 달라집니다. 제한 시간을 연장하려면 스택 작업을 수행할 때 [서비스 역할](#)을 지정합니다. 이미 서비스 역할을 사용 중이거나 스택에 목록에 없는 리소스가 포함된 경우 [AWS Support](#)에 문의하세요.

스택 상태가 UPDATE_ROLLBACK_FAILED인 경우 [업데이트 롤백 실패](#)를 참조하십시오.

VPC에 보안 그룹이 없음

VPC에 지정한 보안 그룹이 있는지 확인합니다. 보안 그룹이 없으면 보안 그룹 이름이 아니라 보안 그룹 ID를 지정했는지 확인합니다. 예를 들어, `AWS::EC2::SecurityGroupIngress` 리소스에는 `SourceSecurityGroupName` 및 `SourceSecurityGroupId` 속성이 있습니다. VPC 보안 그룹의 경우 `SourceSecurityGroupId` 속성을 사용하고 보안 그룹 ID를 지정해야 합니다.

업데이트 롤백에 실패

종속 리소스가 원래 상태로 되돌아 갈 수 없으면 롤백에 실패합니다(`UPDATE_ROLLBACK_FAILED` 상태). 예를 들어, CloudFormation 외부에서 삭제된 이전 데이터베이스 인스턴스로 롤백되는 스택이 있을 수 있습니다. CloudFormation에서는 이 데이터베이스가 삭제되었는지 알지 못하므로 해당 데이터베이스 인스턴스가 존재하는 것으로 간주하고 롤백을 시도합니다. 따라서 업데이트 롤백에 실패합니다.

실패 원인에 따라 오류를 수동으로 수정하거나 계속해서 롤백할 수 있습니다. 계속해서 롤백하여 스택을 작업 상태(`UPDATE_ROLLBACK_COMPLETE` 상태)로 되돌린 다음 스택을 다시 업데이트해 볼 수 있습니다. 다음 목록에는 업데이트 롤백 실패를 일으키는 일반적인 오류에 대한 해결책이 나와 있습니다.

- 필요한 신호 수 수신에 실패

[signal-resource](#) 명령을 사용하여 성공 신호를 대기 중인 리소스에 필요한 수의 성공 신호를 수동으로 전송한 다음 계속해서 업데이트를 롤백합니다. 예를 들어 업데이트 롤백 중 Auto Scaling 그룹의 인스턴스가 지정된 제한 시간 이내에 성공 신호를 보내지 못할 수 있습니다. Auto Scaling 그룹에 성공 신호를 수동으로 전송합니다. 업데이트 롤백을 계속 진행하면 CloudFormation에서 수동으로 보낸 신호를 감지해 롤백을 계속 진행합니다.

- CloudFormation 외부에서 리소스 변경이 수행됨

원본 스택의 템플릿과 일치하도록 리소스를 수동으로 동기화한 다음 계속해서 업데이트를 롤백합니다. 예를 들어, CloudFormation에서 롤백하려는 리소스를 수동으로 삭제한 경우 원본 스택과 이름 및 속성이 동일한 리소스를 수동으로 생성해야 합니다.

- 권한 부족

리소스를 수정할 IAM 권한이 충분한지 확인한 다음 업데이트 롤백을 계속 진행합니다. 예를 들어, IAM 정책이 S3 버킷의 생성은 허용하지만 해당 버킷의 수정은 허용하지 않을 수 있습니다. 정책에 수정 작업을 추가하십시오.

- 잘못된 보안 토큰

CloudFormation에는 새로운 자격 증명 세트가 필요합니다. 아무 변경도 필요하지 않습니다. 계속해서 업데이트를 롤백하면 자격 증명이 갱신됩니다.

- 제한 오류

필요 없는 리소스를 삭제하거나 [할당량 증가](#)를 요청한 다음 계속해서 업데이트 롤백을 진행합니다. 예를 들어, EC2 온디맨드 인스턴스 개수에 대한 계정 할당량이 5개인데 업데이트 롤백이 이 할당량을 초과한 경우 롤백에 실패합니다.

- 리소스가 안정화되지 않음

작업이 CloudFormation 제한 시간을 초과했거나 AWS 서비스가 중단되어 리소스가 응답하지 않았습니다. 아무 변경도 필요하지 않습니다. 리소스 작업이 완료되거나 AWS 서비스가 다시 작동 상태가 되면 계속해서 업데이트를 롤백합니다.

계속해서 업데이트를 롤백하려면 CloudFormation 콘솔 또는 AWS 명령줄 인터페이스(AWS CLI)를 사용할 수 있습니다. 자세한 내용은 [업데이트 롤백 계속](#) 섹션을 참조하세요.

이러한 해결책을 모두 수행해도 문제가 해결되지 않으면 CloudFormation에서 성공적으로 롤백할 수 없는 리소스를 건너뛸 수 있습니다. 자세한 내용은 AWS CloudFormation API 참조의 [ContinueUpdateRollback](#) API 작업에 대한 ResourcesToSkip 파라미터를 참조하세요. CloudFormation에서 지정된 리소스의 상태를 UPDATE_COMPLETE로 설정하고 스택을 계속 롤백합니다. 롤백이 완료된 이후에는 건너뛴 리소스의 상태가 스택 템플릿에 있는 리소스의 상태와 일치하지 않게 됩니다. 다른 스택을 업데이트하기 전에 서로 일치하도록 리소스를 수정하거나 스택을 업데이트해야 합니다. 그렇지 않으면 후속 스택 업데이트에 실패할 수 있고, 이 경우 스택을 복구할 수 없습니다.

대기 조건이 Amazon EC2 인스턴스에서 필요한 수의 신호를 수신하지 못함

이러한 상황을 해결하려면 다음 작업을 시도해 보십시오.

- 사용 중인 AMI에 CloudFormation 헬퍼 스크립트가 설치되어 있는지 확인합니다. AMI에 헬퍼 스크립트가 포함되어 있지 않은 경우에는 인스턴스로 다운로드할 수 있습니다. 자세한 내용은 CloudFormation 템플릿 참조 가이드의 [CloudFormation 헬퍼 스크립트 참조](#)를 참조하세요.
- 인스턴스에서 cfn-signal 명령이 성공적으로 실행되었는지 확인합니다. /var/log/cloud-init.log 또는 /var/log/cfn-init.log와 같은 로그를 보면 인스턴스 시작을 디버깅하는 데 도움이 될 수 있습니다. 인스턴스에 로그인해 로그를 검색할 수 있지만 [실패 시 롤백을 사용하지 않도록 설정](#)하거나 그렇지 않으면 스택 생성 실패 후 CloudFormation에서 인스턴스를 삭제합니다. 또한 Amazon CloudWatch에 [로그를 게시](#)할 수도 있습니다. Windows의 경우 C:\cfn\log에서 cfn 로

그를, %ProgramFiles%\Amazon\EC2ConfigService에서 EC2Configure 서비스 로그를 확인할 수 있습니다.

- 인스턴스가 인터넷에 연결되어 있는지 확인합니다. 인스턴스가 VPC에 있는 경우 인스턴스가 프라이빗 서브넷에 있는 경우에는 NAT 디바이스를 통해 또는 퍼블릭 서브넷에 있는 경우에는 인터넷 게이트웨이를 통해 인터넷에 연결할 수 있어야 합니다. 인스턴스의 인터넷 연결을 테스트하려면 퍼블릭 웹 페이지(예: <http://aws.amazon.com>)에 액세스해 봅니다. 예를 들어 인스턴스에서 다음 명령을 실행할 수 있습니다. 이 명령이 HTTP 200 상태 코드를 반환해야 합니다.

```
curl -I https://aws.amazon.com
```

NAT 디바이스 구성에 대한 내용은 Amazon VPC 사용 설명서의 [NAT](#)를 참조하세요.

리소스가 스택에서 제거되었지만 삭제되지 않음

스택 업데이트 중 CloudFormation이 스택에서 리소스를 제거했지만 리소스를 삭제하지 않았습니다. 리소스가 아직 남아 있지만 더 이상 CloudFormation을 통해 액세스할 수 없습니다. 스택 업데이트 중에 다음과 같은 경우에 문제가 발생할 수 있습니다.

- CloudFormation이 기존 리소스를 교체해야 하므로, 먼저 새로운 리소스를 만들고 이전 리소스를 삭제하려고 시도합니다.
- 스택 템플릿에서 리소스를 제거했기 때문에 CloudFormation이 스택에서 리소스를 삭제하려고 시도합니다.

그러나 CloudFormation이 리소스를 삭제하지 못하는 경우가 있습니다. 예를 들어 사용자가 특정 유형의 리소스를 삭제할 권한이 없을 수 있습니다.

CloudFormation은 세 번에 걸쳐 이전 리소스를 삭제하려고 시도합니다. CloudFormation이 이전 리소스를 삭제하지 못하면 스택에서 이전 리소스를 제거하고 스택 업데이트를 진행합니다. 스택 업데이트가 완료되면 CloudFormation이 UPDATE_COMPLETE 스택 이벤트를 실행하지만 하나 이상의 리소스를 삭제할 수 없다고 되어 있는 StatusReason이(가) 포함됩니다. CloudFormation은 특정 리소스에 대한 DELETE_FAILED 이벤트를 실행하는데, CloudFormation이 리소스 삭제에 실패한 이유가 자세히 나와 있는 해당 StatusReason도 제공됩니다.

이 문제를 해결하려면 콘솔 또는 API를 사용하여 기본 서비스에 대한 리소스를 직접 삭제하세요.

Support에 문의

AWS Support를 사용하는 경우 <https://console.aws.amazon.com/support/home#/>에서 기술 지원 사례를 생성할 수 있습니다. 지원 센터에 연락하기 전에 다음 정보를 수집합니다.

- 스택의 ID. 스택 ID는 [CloudFormation 콘솔](#)의 개요 탭에서 확인할 수 있습니다. 자세한 내용은 [CloudFormation 콘솔에서 스택 정보 보기](#) 섹션을 참조하세요.

Important

CloudFormation 외부에서 스택 리소스를 변경하지 마세요. CloudFormation 외부에서 스택의 리소스를 변경하면 스택이 복구 불가능한 상태가 될 수 있습니다.

- 모든 스택 오류 메시지. 스택 오류 메시지 보기에 대한 내용은 [문제 해결 가이드](#) 단원을 참조하십시오.
- Amazon EC2 문제의 경우 cloud-init 및 cfn 로그를 수집합니다. Amazon EC2 인스턴스에 대한 이러한 로그는 /var/log/ 디렉터리에 게시됩니다. 이러한 로그는 인스턴스를 설정하는 동안 발생하는 프로세스 및 명령 출력을 캡처합니다. Windows의 경우 %ProgramFiles%\Amazon\EC2ConfigService 및 C:\cfn\log에서 EC2Configure 서비스 및 cfn 로그를 수집합니다.

AWS re:Post의 [CloudFormation 커뮤니티](#)에서 답변을 검색하고 질문을 게시할 수도 있습니다.

Amazon Q Developer를 사용하여 실패한 CloudFormation 스택 배포 문제 해결

Amazon Q Developer는 AWS 애플리케이션을 이해하고, 구축하고, 확장하고, 운영하는 데 도움을 줄 수 있는 생성형 인공지능(AI) 기반 대화형 어시스턴트입니다. Amazon Q Developer에 대한 자세한 내용은 [Amazon Q Developer 사용 설명서](#)를 참조하세요.

이 문서에서는 실패한 CloudFormation 배포 문제를 해결하는 데 도움이 되는 Amazon Q Developer를 사용하는 방법을 설명합니다. 이 기능은 스택 배포 중 발생하는 문제를 빠르게 식별하고 해결하는 데 도움이 되도록 설계되었습니다.

Amazon Bedrock 기반: Amazon Q는 Amazon Bedrock을 기반으로 구축되었으며, Amazon Bedrock에 구현된 [자동화된 침해 탐지](#)를 포함하여 AI의 안전, 보안 및 책임감 있는 사용을 지원할 수 있습니다.

주제

- [특성](#)
- [작동 방식](#)
- [가격 책정](#)
- [리전별 가용성](#)
- [제한 사항](#)

특성

Amazon Q Developer 솔루션을 사용한 스택 배포 문제 해결은 다음과 같은 기능을 제공합니다.

- 오류 분석 - CloudFormation 스택 배포 오류에 대한 이해하기 쉬운 분석을 제공합니다.
- 해결 방법 제안 - 식별된 문제를 해결하기 위한 단계별 지침을 제공합니다.

작동 방식

CloudFormation 스택 생성 또는 업데이트에 실패하면 AWS Management Console에 오류 메시지가 표시됩니다.

오류 메시지 옆에 있는 Q로 진단 버튼을 선택할 수 있습니다.

Amazon Q Developer는 오류를 분석하고 문제에 대한 설명을 제공합니다.

추가 지원을 받으려면 해결 도움말을 선택하여 자세한 해결 단계를 안내받으세요.

가격 책정

요금에 대한 자세한 내용은 [Amazon Q Developer 요금](#)을 참조하세요.

리전별 가용성

Amazon Q Developer는 대부분의 AWS 리전에서 사용할 수 있습니다. 현재 Amazon Q Developer를 사용할 수 있는 모든 리전 목록은 Amazon Q Developer User Guide의 [Supported Regions for Amazon Q Developer](#)를 참조하세요.

제한 사항

- 이 기능은 AWS Management Console을 통해서만 액세스할 수 있습니다.
- 이 기능은 가장 많이 발생하는 상위 20개 오류를 다룹니다. 가능한 모든 오류를 다루지는 않습니다.
- 복잡한 문제 해결 사례의 경우 여전히 지원의 개입이 필요할 수 있습니다.

AWS CloudFormation 사용 설명서 기록

다음 표는 2018년 5월 이후 AWS CloudFormation 사용 설명서 내용에서 변경된 중요 사항을 기술한 것입니다. 설명서 업데이트에 대한 알림을 받으려면 RSS 피드를 구독하면 됩니다.

Important

2018년 5월 이후 템플릿 참조 콘텐츠에 대한 업데이트를 설명하는 테이블 행이 새로운 AWS CloudFormation 템플릿 참조로 이동했습니다. 이러한 업데이트 및 향후 변경 사항은 AWS CloudFormation 템플릿 참조의 [AWS CloudFormation 템플릿 참조에 대한 문서 기록](#)을 참조하세요.

변경 사항	설명	날짜
PDF 안내서 사용 가능	이제 AWS CloudFormation 사용 설명서를 PDF로 다운로드할 수 있습니다.	2025년 5월 30일
템플릿 참조 콘텐츠를 새 안내서로 이동	CloudFormation에서 CloudFormation 템플릿 참조 안내서를 게시했습니다. 자세한 내용은 CloudFormation 템플릿 참조 안내서 를 참조하세요.	2025년 5월 30일
IaC 생성기에서 부분 스캔 지원	이제 스캔할 특정 리소스 유형을 선택하여 기존 리소스에서 코드형 인프라(IaC) 템플릿을 더 쉽게 생성할 수 있습니다. 자세한 내용은 CloudFormation IaC 생성기로 리소스 스캔 시작 을 참조하세요.	2025년 3월 27일
스택 리팩터링	스택 리팩터링은 기존 리소스 속성과 데이터를 유지하면서 CloudFormation 스택에서 리소	2025년 2월 6일

스 재구성을 간소화합니다. 자세한 내용은 [스택 리팩터링](#)을 참조하세요.

[Amazon Q Developer로 스택 배포 문제 해결](#)

이제 CloudFormation 스택을 배포할 때 Amazon Q Developer를 사용하여 일반적인 오류를 해결할 수 있습니다. 자세한 내용은 [Amazon Q Developer를 사용하여 실패한 CloudFormation 스택 배포 문제 해결](#)을 참조하세요.

2024년 11월 22일

[스택 배포 타임라인 그래프](#)

이제 스택 배포의 시각적 표현을 확인할 수 있습니다. 스택 배포 타임라인 그래프에는 스택 배포 상태, 개별 리소스 배포 상태 및 배포 상태가 변경된 시간이 표시됩니다. 자세한 내용은 [CloudFormation 스택 배포의 타임라인 보기](#)를 참조하세요.

2024년 11월 11일

[스캔한 리소스와 생성된 템플릿 시각화](#)

이제 인프라 스택을 배포하기 전에 스캔 요약 세부 정보를 시각화하고 생성된 템플릿을 미리 봄으로써 코드형 인프라 (IaC) 생성기 워크플로를 간소화할 수 있습니다. 자세한 내용은 [CloudFormation 콘솔의 스캔 요약 보기](#) 및 [스캔한 리소스에서 CloudFormation 스택 생성](#)을 참조하세요.

2024년 8월 22일

[AWS CloudFormation Git 동기화와 Amazon EventBridge 통합](#)

이제 AWS CloudFormation Git 동기화는 동기화 상태 변경 사항을 Amazon EventBridge에 이벤트로 게시합니다. 자세한 내용은 [리포지토리 동기화 상태 변경 이벤트 세부 정보 및 리소스 동기화 상태 변경 이벤트 세부 정보](#)를 참조하세요.

2024년 7월 29일

[멈춘 스택 강제 삭제](#)

멈춘 스택 삭제 작업에 대해 스택을 강제 삭제하는 두 가지 새로운 옵션을 사용할 수 있습니다. 이제 스택을 강제 삭제된 리소스를 유지하거나 전체 스택을 강제 삭제하도록 선택할 수 있습니다. 자세한 내용은 [CloudFormation 콘솔에서 스택 삭제](#)를 참조하세요.

2024년 5월 22일

[AWS CloudTrail 이벤트 스택 작업 근본 원인](#)

CloudFormation은 새로운 AWS CloudTrail 딥 링크 통합을 통해 스택 운영의 문제 해결 경험을 개선합니다. 이 기능은 CloudFormation 콘솔의 스택 작업 이벤트를 관련 CloudTrail 이벤트에 직접 연결합니다. 자세한 내용은 [스택 실패 원인 파악](#)을 참조하세요.

2024년 5월 15일

[속성 수준 변경 세트](#)

속성 수준 변경 세트를 사용하면 CloudFormation 배포에서 리소스의 속성 값에 대해 수행할 변경 사항을 미리 볼 수 있습니다. 자세한 내용은 [CloudFormation 스택에 대한 변경 세트 보기](#)를 참조하세요.

2024년 4월 12일

[CloudFormation에서
CONFIGURATION_COMP
LETE 이벤트 도입](#)

이제 CONFIGURATION_COMP
LETE 이벤트를 사용하여 리
소스 생성과 관련된 워크플로
를 더 빠르게 진행할 수 있습니
다. 자세한 내용은 [CloudForm
ation 스택 생성 이벤트 이해](#)를
참조하세요.

2024년 3월 11일

[기존 AWS 리소스에서 AWS
CloudFormation 템플릿과
AWS CDK 애플리케이션 생성](#)

이제 계정에 프로비저닝된 리
소스 중 CloudFormation에서
아직 관리하지 않는 리소스를
사용하여 템플릿을 생성할 수
있습니다. 자세한 내용은 [IaC
생성기를 사용하여 기존 리소
스에서 템플릿 생성](#)을 참조하
세요.

2024년 2월 2일

[StackSets 동시성 모드](#)

동시성 모드는 스택 세트 작
업 중에 동시성 수준이 작동
하는 방식을 선택할 수 있는
StackSetOperationP
references 의 파라미
터입니다. 자세한 내용은
[CloudFormation StackSets에
대한 동시성 모드 선택](#)을 참조
하세요.

2023년 11월 9일

[상세 StackSet 드리프트 정보](#)

다음 API를 통해 StackSet 템플릿에서 드리프트된 스택 인스턴스와 드리프트된 리소스를 확인할 수 있습니다.

2023년 7월 24일

[ListStackInstanceResourceDrifts](#)

스택 인스턴스의 리소스에 대한 드리프트 정보를 반환합니다.

[StackInstanceResourceDriftsSummary](#)

스택 인스턴스의 리소스 드리프트에 대한 요약 정보를 포함하는 구조입니다.

[AWS Organizations 신뢰
액세스를 제어하기 위한
CloudFormation StackSets
API](#)

CloudFormation StackSets는 AWS Organizations 신뢰 액세스를 관리하기 위한 다음 API를 고객에게 제공합니다.

2023년 6월 5일

[ActivateOrganizationsAccess](#)

AWS Organizations를 사용하여 신뢰할 수 있는 액세스를 활성화합니다. StackSets와 Organizations 간의 신뢰할 수 있는 액세스가 활성화되면 관리 계정에 서 조직의 StackSets를 생성하고 관리하는 권한을 보유하게 됩니다.

[DeactivateOrganizationsAccess](#)

AWS Organizations를 사용하여 신뢰할 수 있는 액세스를 비활성화합니다. 신뢰할 수 있는 액세스가 비활성화되면 관리 계정에서 조직의 서비스 관리형 StackSets를 생성하고 관리하는 권한을 잃게 됩니다.

[DescribeOrganizationsAccess](#)

계정의 OrganizationAccess 상태에 대한 정보를 검색합니다. 이 API는 관리 계정이나 위임된 관리자가 CallAs 파라미터를 사용하여 직접적으로 호출할 수 있습니다. 관리 계정에서 CallAs 파라미터 없

이 이 API를 직접적으로 호출할 수도 있습니다.

[DescribeStackSet API](#)

DescribeStackSet API에는 지정된 스택 세트가 배포되는 리전 목록에 대한 새 파라미터가 있습니다. 자세한 내용은 [DescribeStackSet](#) 섹션을 참조하세요.

2023년 2월 1일

[CloudFormation 및 Amazon EventBridge를 사용하여 StackSets 이벤트 관리](#)

CloudFormation StackSets는 Amazon EventBridge를 통해 이벤트 알림을 시작합니다. CloudFormation 스택 세트를 생성, 업데이트 또는 삭제한 후 이벤트 기반 작업을 트리거할 수 있습니다. 자세한 내용은 [EventBridge를 사용한 CloudFormation 및 Git 동기화 이벤트 모니터링](#)을 참조하세요.

2022년 11월 16일

[스택 세트 작업을 위한 스택 인스턴스에 대한 인사이트 개선](#)

CloudFormation StackSets는 스택 세트 작업을 위한 스택 인스턴스에 대한 자세한 정보를 제공합니다.

2022년 11월 4일

[DescribeStackSetOperation](#)

이제 DescribeStackSetOperation 을 사용하여 배포 중에 스택 세트 작업에 실패한 스택 인스턴스 수를 제공할 수 있습니다.

[ListStackInstances](#)

이제 LastOperationID 필터링 옵션을 사용하여 스택 세트 작업에 대한 스택 인스턴스를 나열할 수 있습니다.

[CloudFormation 및 Amazon EventBridge를 사용하여 이벤트 관리](#)

특정 CloudFormation 이벤트가 발생할 때 알림을 수신합니다. 자세한 내용은 [EventBridge를 사용한 CloudFormation 및 Git 동기화 이벤트 모니터링](#)을 참조하세요.

2022년 7월 20일

[계정 수준](#)

CloudFormation에서는 고객이 배포 대상을 개별 계정으로 제한하거나 제공된 OU에 추가 계정을 포함할 수 있도록 하는 기능인 계정 필터 유형의 정식 출시를 발표합니다. 자세한 내용은 [서비스 관리형 StackSets에 대한 계정 수준 대상](#)을 참조하세요.

2022년 7월 7일

[CloudFormation 레지스트리](#)

CloudFormation은 고객이 스택 생성, 업데이트 또는 삭제 작업 전에 리소스 구성을 검사하거나 작업을 자동화하기 위해 사용자 지정 논리를 호출할 수 있는 기능인 후크의 정식 출시를 발표합니다. 자세한 내용은 [CloudFormation 후크 사용 설명서](#)를 참조하세요.

2022년 2월 10일

[스택 실패 옵션](#)

프로비저닝 실패가 발생한 경우 성공적으로 프로비저닝된 리소스를 롤백하지 않고 실패 지점부터 시작하여 애플리케이션을 반복적으로 개발할 수 있습니다. 스택 실패 옵션을 지정하여 CREATE_FAILED 또는 UPDATE_FAILED 상태의 리소스 문제를 해결할 수 있습니다. 모든 스택 배포 및 변경 세트 작업에 대한 실패 옵션을 프로비저닝할 수 있습니다. 자세한 내용은 [리소스 프로비저닝 시 실패 처리 방법 선택](#)을 참조하세요.

2021년 8월 30일

[스택 세트로 스택 가져오기](#)

이제 기존 스택을 새 스택 세트 또는 기존 스택 세트로 가져올 수 있습니다. 자세한 내용은 [CloudFormation StackSets로 스택 가져오기](#)를 참조하세요.

2021년 7월 28일

[늘어난 할당량](#)

이제 AWS 계정에서 기본적으로 최대 2000 스택을 선언할 수 있습니다. 자세한 내용은 [CloudFormation 할당량 이해](#)를 참조하세요.

2021년 7월 15일

[퍼블릭 타사 익스텐션](#)

이제 AWS의 익스텐션과 마찬가지로 서드 파티 게시자가 제공하는 퍼블릭 익스텐션을 사용할 수 있습니다. 자세한 내용은 [CloudFormation 레지스트리에서 서드 파티 공개 확장 프로그램 사용](#)을 참조하세요.

2021년 6월 21일

[스택 세트 템플릿의 참조 매크로](#)

이제 StackSets가 매크로를 참조하는 템플릿에서 자체 관리형 권한으로 스택 세트를 생성 또는 업데이트할 수 있도록 지원합니다. 매크로에 대한 자세한 내용은 [템플릿 매크로를 사용하여 CloudFormation 템플릿에서 사용자 지정 처리 수행](#)을 참조하세요.

2021년 4월 14일

[동적 참조에서 Systems Manager 파라미터의 최신 값 사용](#)

이제 스택을 생성하거나 업데이트할 때마다 CloudFormation에 최신 버전의 Systems Manager 파라미터가 사용되도록 할 수 있습니다. 더 이상 특정 버전을 지정할 필요가 없습니다. 자세한 내용은 [Systems Manager Parameter Store에서 일반 텍스트 값 가져오기](#)를 참조하세요.

2021년 4월 13일

[모듈에서 리소스 이름에 마침표 구분 기호 사용 지원](#)

이제 모듈에 포함된 리소스의 정규화된 논리적 이름을 지정할 때 마침표를 구분 기호로 사용할 수 있습니다. 자세한 내용은 [CloudFormation 템플릿의 모듈 리소스 참조](#)를 참조하세요.

2021년 4월 8일

[이제 CloudFormation StackSets에서 병렬 리전 배포 지원](#)

이제 StackSets를 리전에 순차 또는 병렬로 배포하도록 선택할 수 있습니다. 자세한 내용은 [스택 세트 작업 옵션](#)을 참조하세요.

2021년 4월 6일

[이제 CloudFormation StackSets에서 AWS Organizations를 통해 위임된 관리자 지원](#)

조직의 관리 계정에 더해 위임된 관리자 계정에서 조직의 서비스 관리형 권한을 사용하여 스택 세트를 생성하고 관리할 수 있습니다. 자세한 내용은 [위임된 관리자 멤버 계정 등록 및 서비스 관리형 권한으로 CloudFormation StackSets 생성](#)을 참조하세요.

2021년 2월 18일

[CloudFormation StackSets 리전 가용성](#)

이제 CloudFormation StackSets를 아시아 태평양(오사카) 리전에서 사용할 수 있습니다. 자세한 내용은 [StackSets를 사용하여 여러 계정 및 리전의 스택 관리](#)를 참조하세요.

2021년 2월 10일

[모듈](#)

모듈을 사용하면 모든 스택 템플릿에 포함하기 위한 리소스 구성을 투명하고 관리가 편리하면서도 반복적으로 패키징할 수 있습니다. 모듈은 공통적 서비스 구성과 모범 사례를 모듈식 맞춤형 구성 요소로 캡슐화하고 스택 템플릿에 포함할 수 있습니다. 자세한 내용은 [CloudFormation 모듈을 사용하여 템플릿 전체에 포함할 수 있는 재사용 가능한 리소스 구성 생성](#)을 참조하세요.

2020년 11월 24일

중첩 스택에 대한 변경 세트

중첩 스택에 대한 변경 세트를 사용하면 모든 중첩 스택 계층의 애플리케이션과 인프라 리소스 변경 사항을 미리 확인하고, 모든 변경 사항이 의도한 대로인지 확인한 후 업데이트를 진행할 수 있습니다. 자세한 내용은 [중첩 스택에 대한 변경 세트](#)를 참조하세요.

2020년 11월 18일

늘어난 할당량

다음 AWS CloudFormation 할당량이 업데이트되었습니다.

2020년 10월 22일

- 이제 AWS CloudFormation 템플릿에서 최대 200개의 매핑을 선언할 수 있습니다.
- 이제 AWS CloudFormation 템플릿의 각 매핑에 대해 최대 200개의 매핑 속성을 선언할 수 있습니다.
- 이제 AWS CloudFormation 템플릿에서 최대 200개의 출력을 선언할 수 있습니다.
- 이제 AWS CloudFormation 템플릿에서 최대 200개의 파라미터를 선언할 수 있습니다.
- 이제 AWS CloudFormation 템플릿에서 최대 500개의 리소스를 선언할 수 있습니다.
- 이제 Amazon S3 객체에서 최대 1 MB 크기의 템플릿 본문을 전달할 수 있습니다.

[프라이빗 리소스에 대한 드립트 감지](#)

이제 CloudFormation은 확장된 AWS 리소스 목록과 CloudFormation 레지스트리의 프로비저닝 가능한 것으로 정의된 프라이빗 리소스에서 드립트 감지 작업을 지원합니다. 자세한 내용은 [리소스 유형 지원](#)을 참조하세요.

2020년 10월 1일

[리소스 제공자를 등록하는 데 필요한 권한 업데이트](#)

이제 계정에 리소스 공급자를 등록하려면 해당 리소스 공급자의 S3 버킷에 업로드된 스키마 핸들러 패키지에 액세스할 수 있는 권한이 있어야 합니다. 자세한 내용은 [서드 파티 비공개 확장 프로그램 등록을 위한 IAM 권한](#)을 참조하세요.

2020년 8월 7일

[리소스 가져오기에서 프로비저닝 가능한 프라이빗 리소스 유형 지원](#)

이제 가져오기 작업은 프로비저닝 가능한 프라이빗 리소스 유형, 즉 프로비저닝 유형이 FULLY_MUTABLE 또는 IMMUTABLE 인 프라이빗 리소스 유형을 지원합니다. 자세한 내용은 [가져오기 작업을 지원하는 리소스](#)를 참조하십시오.

2020년 6월 3일

[CodeDeploy를 통한 ECS 블루/그린 배포](#)

이제 CloudFormation을 사용하여 CodeDeploy를 통한 ECS 블루/그린 배포를 수행할 수 있습니다. 블루/그린 배포는 애플리케이션 버전 변경으로 인한 중단을 최소화하기 위해 AWS CodeDeploy에서 제공하는 안전한 배포 전략입니다. 자세한 내용은 [CloudFormation을 사용하여 CodeDeploy를 통한 ECS 블루/그린 배포 수행](#)을 참조하세요.

2020년 5월 19일

[CloudFormation StackSets 리전 가용성](#)

이제 CloudFormation StackSets를 AWS GovCloud(미국 서부) 리전에서 사용할 수 있습니다.

2020년 5월 18일

[AWS Organizations와 AWS CloudFormation StackSet 통합](#)

이제 StackSets을 사용하여 조직의 모든 계정 또는 AWS Organizations의 특정 조직 단위(OU)에 대한 배포를 중앙에서 관리할 수 있습니다. 조직 또는 OU에 추가되는 새로운 계정에 자동 배포를 활성화할 수 있습니다. 여러 계정에 걸쳐 배포하는 데 필요한 권한은 StackSets이 자동으로 처리합니다. 자세한 내용은 [StackSets를 사용하여 여러 계정 및 리전의 스택 관리](#)를 참조하세요.

2020년 2월 11일

[StackSets에 대한 드리프트 감지](#)

이제 스택 세트와 여기에 포함된 모든 스택 인스턴스에 대해 드리프트 감지를 실행할 수 있습니다. 자세한 내용은 [CloudFormation StackSets에서 드리프트 감지 수행](#)을 참조하세요.

2019년 11월 19일

[이제 CloudFormation 레지스트리 사용 가능](#)

이제 CloudFormation 콘솔을 사용하여 계정에서 사용할 수 있는 프라이빗 및 퍼블릭 리소스를 볼 수 있습니다. 자세한 내용은 [CloudFormation 레지스트리에서 사용 가능하고 활성화된 확장 프로그램 보기](#)를 참조하세요.

2019년 11월 18일

[CloudFormation 레지스트리 API 작업](#)

이제 CloudFormation 레지스트리에서 유형을 관리하기 위한 다음과 같은 API 작업을 사용할 수 있습니다.

2019년 11월 18일

[DeregisterType](#)

CloudFormation 레지스트리의 활성 사용에서 유형 또는 유형 버전을 제거합니다.

[DescribeType](#)

등록된 유형에 대한 자세한 정보를 반환합니다.

[DescribeTypeRegistration](#)

현재 상태, 유형 및 버전 식별자를 포함하여 유형 등록에 대한 정보를 반환합니다.

[ListTypeRegistrations](#)

지정된 유형의 등록 요청 식별자 목록을 반환합니다.

[ListTypes](#)

CloudFormation에 등록된 유형에 대한 요약 정보를 반환합니다.

[ListTypeVersions](#)

유형의 버전에 대한 요약 정보를 반환합니다.

[RegisterType](#)

CloudFormation 레지스트리에 유형을 등록합니다. 유형을 등록하면 AWS 계정의

CloudFormation 템플릿에서 사용할 수 있습니다.

[SetTypeDefaultVersion](#)

유형의 기본 버전을 지정합니다. 유형의 기본 버전은 CloudFormation 작업에 사용됩니다.

자세한 내용은 [CloudFormation 레지스트리를 통해 익스텐션 관리](#)를 참조하세요.

[리소스 가져오기 추가됨](#)

CloudFormation 관리 외부에서 AWS 리소스를 생성한 경우, `resource import`를 사용하여 이러한 기존 리소스를 CloudFormation 관리로 가져올 수 있습니다. 자세한 내용은 [리소스 가져오기를 사용하여 AWS 리소스를 CloudFormation 스택으로 가져오기](#)를 참조하세요.

2019년 11월 11일

[스택 세트 한도 증가](#)

이제 관리자 계정에서 최대 100개 스택 세트를 생성하고, 스택 세트당 최대 2,000개 스택 인스턴스를 생성하고, 각 리전에서 관리자 계정당 최대 3,500개 스택 인스턴스 작업을 동시에 실행할 수 있습니다. 자세한 내용은 [CloudFormation 할당량 이해](#)를 참조하세요.

2019년 8월 2일

동시 스택 작업의 리소스에 대한 제한

CloudFormation은 이제 동시 스택 작업의 리소스 수에 대한 계정 제한을 적용합니다. 이 제한은 리전에 따라 결정됩니다. 자세한 내용은 [CloudFormation 할당량 이해](#)를 참조하세요.

2019년 4월 30일

스택 인스턴스 작업 제한

이제 StackSets의 경우 관리자 계정당 해당 리전에서 동시에 최대 1,500개의 스택 인스턴스 작업을 실행할 수 있습니다. 자세한 내용은 [CloudFormation 할당량 이해](#)를 참조하세요.

2018년 12월 13일

이제 CAPABILITY_AUTO_EXPAND 기능을 사용할 수 있습니다.

이제 CAPABILITY_AUTO_EXPAND 기능을 사용하여 변경 세트의 변경 사항을 먼저 검토하지 않고 매크로가 포함된 스택 템플릿에서 직접 스택을 생성 또는 업데이트할 수 있습니다. 자세한 내용은 AWS CloudFormation API 참조의 [CreateStack](#) 또는 [UpdateStack](#) 섹션을 참조하세요.

2018년 12월 7일

스택 드리프트 감지 추가됨

이제 스택의 실제 구성이 CloudFormation에서 정의한 예상 템플릿 구성이 드리프트되었는지 여부를 감지할 수 있습니다. 전체 스택 또는 개별 스택 리소스에서 드리프트를 감지할 수 있습니다. 자세한 내용은 [드리프트 감지를 사용하여 스택 및 리소스에 대한 비관리형 구성 변경 감지](#)를 참조하세요.

2018년 11월 13일

이제 secretsmanager 동적 참조 사용 가능

이제 secretsmanager 동적 참조를 사용하여 AWS Secrets Manager에 저장된 전체 보안 암호 또는 보안 암호 값을 검색할 수 있습니다. 보안 암호는 데이터베이스 보안 인증 정보, 암호, 타사 API 키 및 임의 텍스트가 될 수 있습니다. secretsmanager 동적 참조 사용을 통해 Secrets Manager 및 CloudFormation에서 확인된 보안 암호 값을 로깅 또는 보존하지 않도록 보장합니다. 자세한 내용은 [Secrets Manager에서 보안 암호 또는 보안 암호 값 가져오기](#)를 참조하세요.

2018년 11월 9일

매크로 사용 가능

이제 매크로를 사용하여 찾기 및 바꾸기 작업 같은 간단한 작업부터 전체 템플릿 변형과 같은 작업까지 템플릿에 대한 사용자 지정 처리를 수행할 수 있습니다. 자세한 내용은 [템플릿 매크로를 사용하여 CloudFormation 템플릿에서 사용자 지정 처리 수행](#)을 참조하세요.

2018년 9월 6일

[이제 CloudFormation에서 PrivateLink를 사용하는 VPC 엔드포인트 지원](#)

VPC 엔드포인트를 생성하여 NAT 인스턴스, VPN 연결 또는 Direct Connect를 통해 인터넷에 액세스할 필요 없이 VPC와 CloudFormation 사이의 프라이빗 연결을 생성할 수 있습니다. 자세한 내용은 [인터페이스 엔드포인트\(AWS PrivateLink\)를 사용하여 CloudFormation에 액세스](#)를 참조하세요.

2018년 8월 22일

[동적 참조에서 보안 문자열 지원](#)

이제 새 동적 참조를 사용하여 스택 템플릿의 Systems Manager Parameter Store SecureString 유형 파라미터를 포함한 다른 서비스에 저장 및 관리되는 값을 지정할 수 있습니다. 자세한 내용은 [Systems Manager Parameter Store에서 보안 문자열 값 가져오기](#)를 참조하세요.

2018년 8월 16일

[이제 스택 세트에서 사용자 지정 실행 역할 지원](#)

이제 대상 계정의 사용자 지정 실행 역할을 사용하여 사용자 또는 그룹에서 스택 세트에 포함할 수 있는 스택 리소스를 제어할 수 있습니다. 자세한 내용은 [스택 세트 작업에 대한 고급 권한 옵션 설정](#)을 참조하세요.

2018년 5월 30일

[스택 인스턴스의 선택적 업데이트](#)

이제 선택 사항인 Accounts 및 Regions 파라미터를 사용하여 스택 세트 업데이트 작업 도중 스택 인스턴스를 업데이트할 계정 및 리전을 지정할 수 있습니다. 자세한 내용을 알아보려면 [CloudFormation API 참조의 UpdateStackSet](#) 섹션을 참조하세요.

2018년 5월 30일

[이제 CloudFormation에서 암호화가 활성화된 S3 버킷 생성](#)

CloudFormation이 업로드된 스택 템플릿을 저장하기 위해 생성하는 Amazon S3 버킷의 경우 서버 측 암호화가 이제 기본값으로 활성화되어 버킷에 저장된 모든 객체를 암호화합니다. 자세한 내용은 [CloudFormation 콘솔에서 스택 생성](#)을 참조하세요.

2018년 5월 24일

[FIPS 엔드포인트 추가됨](#)

이제 CloudFormation은 다음 퍼블릭 US 리전(US-East-1, US-East-2, US-West-1, US-West-2)에서 FIPS 140-2 검증 암호화 모듈을 사용하는 새 엔드포인트를 제공합니다. 새로운 FIPS 준수 엔드포인트 URL은 Amazon Web Services 일반 참조의 [CloudFormation 엔드포인트 및 할당량](#)을 참조하세요.

2018년 5월 17일

CloudFormation 후크 사용 설명서 업데이트는 CloudFormation 후크 사용 설명서의 [CloudFormation 사용 설명서 관련 문서 기록](#)을 참조하세요.

보관된 업데이트

다음 표에서는 2018년 5월 이전 AWS CloudFormation 사용 설명서의 각 릴리스에서 변경된 중요 사항에 대해 설명합니다.

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2019년 7월 22일	<code>encryptionOptions</code> 속성을 사용하여 Amazon MQ 브로커에 대한 AWS 소유 키 또는 고객 관리형 키를 지정합니다.	2010-05-15
스택 세트 명명 규칙	2018년 4월 10일	스택 세트를 사용하여 생성된 CloudFormation 스택은 이제 새로운 명명 규칙에 따라 스택 이름에 스택 세트 이름이 포함됩니다.	2010-05-15
신규 리소스	2018년 4월 10일	AWS::AppSync::ApiKey <code>AWS::AppSync::ApiKey</code> 리소스를 사용하여 AWS AppSync에서 GraphQL 작업을 실행 중인 클라이언트에 배포할 수 있는 고유 키를 생성합니다. AWS::AppSync::DataSource <code>AWS::AppSync::DataSource</code> 리소스를 사용하여 AWS AppSync에서 해석기에 대한 데이터 원본을 생성합니다. AWS::AppSync::GraphQLApi <code>AWS::AppSync::GraphQLApi</code> 리소스를 사용하여 새로운 AWS AppSync GraphQL API를 생성합니다. AWS::AppSync::GraphQLSchema <code>AWS::AppSync::GraphQLSchema</code> 리소스를 사용하여 AWS AppSync GraphQL API에 대한 데이터 모델을 생성합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		AWS::AppSync::Resolver AWS::AppSync::Resolver 리소스를 사용하여 스키마의 필드에 연결할 논리적 GraphQL 해석기를 정의합니다.	
업데이트된 리소스	2018년 4월 10일	AWS::Config::ConfigurationAggregator OrganizationAggregationSource 속성 유형을 사용하여 AWS Config 구성 집계자에 집계할 AWS Config 데이터의 리전과 AWS Organizations 세부 정보를 검색하는 데 사용할 IAM 역할을 지정합니다.	2010-05-15
신규 리소스	2018년 4월 4일	AWS::Config::AggregationAuthorization AWS::Config::AggregationAuthorization 리소스를 사용하여 AWS Config 데이터를 수집할 수 있도록 집계자 계정에 권한을 부여합니다. AWS::Config::ConfigurationAggregator AWS::Config::ConfigurationAggregator 리소스를 사용하여 AWS Config에 대한 구성 집계자를 생성합니다.	2010-05-15
스택 세트는 현재 사용자 지정 관리자 역할을 지원	2018년 3월 29일	사용자 지정 관리자 역할을 사용하여 동일한 관리자 계정 내에서 특정 스택 세트를 관리할 수 있는 사용자나 그룹을 제어합니다. 자세한 내용은 스택 세트 작업에 대한 고급 권한 옵션 설정 을 참조하세요.	2010-05-15
신규 리소스	2018년 3월 29일	AWS::EC2::LaunchTemplate AWS::EC2::LaunchTemplate 리소스를 사용하여 Amazon EC2 인스턴스의 시작 템플릿을 생성합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2018년 3월 29일	AWS::AutoScaling::AutoScalingGroup LaunchTemplate 속성을 사용하여 인스턴스 시작에 사용할 시작 템플릿을 지정합니다. AWS::EC2::SpotFleet SpotFleetRequestConfigData 속성 유형에서 LaunchTemplateConfigs 속성을 사용하여 시작 템플릿과 재정의 설명합니다.	2010-05-15
새로운 Fn::Cidr 내장 함수	2018년 3월 6일	지정한 Cidr 주소 블록을 반환합니다. 자세한 내용은 Fn::Cidr 섹션을 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
신규 리소스	2018년 3월 6일	AWS::ApiGateway::VpcLink AWS::ApiGateway::VpcLink 리소스를 사용하여 Amazon Virtual Private Cloud(VPC)에 있는 리소스에 액세스하기 위해 AWS::ApiGateway::RestApi 에 API Gateway VPC 링크를 지정합니다. AWS::GuardDuty::Master AWS::GuardDuty::Master 리소스를 사용하여 GuardDuty 기본 계정을 생성합니다. AWS::GuardDuty::Member AWS::GuardDuty::Member 리소스를 사용하여 GuardDuty 멤버 계정을 생성합니다. AWS::SES::ConfigurationSet AWS::SES::ConfigurationSet 리소스를 사용하여 전송하는 이메일에 적용 가능한 규칙 그룹을 생성합니다. AWS::SES::ConfigurationSetEventDestination AWS::SES::ConfigurationSetEventDestination 리소스를 사용하여 구성 세트 이벤트 대상을 지정합니다. AWS::SES::ReceiptFilter AWS::SES::ReceiptFilter 리소스를 사용하여 특정 IP 주소 또는 특정 범위의 IP 주소에서 발신한 메일의 수락 여부를 지정합니다. AWS::SES::ReceiptRule AWS::SES::ReceiptRule 리소스를 사용하여 이메일 주소나 도메인이 둘 이상인 경우 이메일을 대신 수신할 Amazon SES 작업을 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		<u>AWS::SES::ReceiptRuleSet</u> AWS::SES::ReceiptRuleSet 리소스를 사용하여 Amazon SES에 빈 규칙 세트를 지정합니다. <u>AWS::SES::Template</u> AWS::SES::Template 리소스를 사용하여 제목 줄, HTML 부분, 텍스트 전용 부분으로 구성된 이메일 내용을 지정합니다.	

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2018년 3월 6일	AWS::AutoScaling::AutoScalingGroup AutoScalingGroupName 속성을 사용하여 Auto Scaling 그룹의 이름을 지정합니다. AWS::ApiGateway::RestApi ApiKeySourceType 속성을 사용하여 사용량 계획에 따라 측정 요청을 위한 API 키의 원본을 지정합니다. MinimumCompressionSize 속성을 사용하여 API에서 압축의 활성화 또는 비활성화에 사용하는 NULL 값을 허용하는 정수를 지정합니다. AWS::ApplicationAutoScaling::ScalingPolicy TargetTrackingScalingPolicyConfiguration 속성 유형에서 DisableScaleIn 속성을 사용하여 대상 추적 정책에 의해 축소의 비활성화 여부를 지정합니다. AWS::EC2::SpotFleet LaunchSpecifications 속성 유형에서 TagSpecifications 속성을 사용하여 SpotFleet 생성 도중에 적용할 태그를 지정합니다. AWS::Elasticsearch::Domain Arn 속성을 사용하여 Fn::GetAtt 이 도메인의 Amazon 리소스 이름(ARN)을 반환하도록 합니다. Fn::GetAtt 의 DomainArn 속성은 더 이상 사용되지 않습니다. AWS::RDS::DBCluster DBClusterIdentifier 속성을 사용하여 DB 클러스터 식별자를 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		AWS::RDS::DBCluster DBClusterIdentifier 속성을 사용하여 DB 클러스터 식별자를 지정합니다. AWS::Redshift::Cluster ClusterIdentifier 속성을 사용하여 클러스터의 고유 식별자를 지정합니다. AWS::Route53::HealthCheck HealthCheckConfig 속성 유형에서 Regions 속성을 사용하여 Route 53 상태 확인 프로그램이 지정 엔드포인트를 확인할 리전을 지정합니다. AWS::SSM::Document Tags 속성을 사용하여 문서에 적용할 CloudFormation 리소스 태그를 지정합니다.	
업데이트된 리소스	2018년 2월 19일	AWS::CodeBuild::Project Triggers 속성을 사용하여 코드 변경이 리포지토리로 푸시될 때마다 자동으로 소스 코드를 다시 빌드하도록 프로젝트에 대한 Webhook를 구성합니다. 이 방법은 CloudFormation의 GitHub 프로젝트에만 사용할 수 있습니다. GitHub Enterprise 프로젝트에는 사용할 수 없습니다.	2010-05-15
업데이트된 리소스	2018년 2월 8일	AWS::DynamoDB::Table SSESpecification 속성을 사용하여 서버 측 암호화를 활성화하도록 설정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2018년 2월 5일	AWS::CodeBuild::Project Source CodeBuild Project Source 속성 유형의 경우: • <code>GitCloneDepth</code> 속성을 사용하여 다운로드할 이력의 수준을 지정합니다. • <code>InsecureSsl</code> 속성을 사용하여 GitHub Enterprise 프로젝트 리포지토리에 연결되어 있는 동안 SSL 경고를 무시할 것인지 여부를 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2018년 1월 23일	AWS::AutoScaling::LifecycleHook LifecycleHookName 속성을 사용하여 수명 주기 후크의 이름을 지정합니다. AWS::DynamoDB::Table AttributeDefinitions 속성의 경우 업데이트 시 교체가 필요합니다. AWS::EC2::Instance CreditSpecification 속성을 사용하여 T2 인스턴스의 CPU 사용량에 대한 크레딧 옵션을 지정합니다. ElasticGpuSpecifications 속성을 사용하여 인스턴스에 연결하여 애플리케이션의 그래픽 성능을 높일 수 있는 GPU 리소스인 탄력적 GPU를 지정합니다. AWS::EC2::VPC InstanceTenancy 속성의 경우, "dedicated" 에서 "default" 로 업데이트할 때 무중단이 요구됩니다. AWS::ECS::Service HealthCheckGracePeriodSeconds 속성을 사용하여 Amazon ECS 서비스 스케줄러에서 작업이 처음 시작된 후 비정상적으로 표시되는 Elastic Load Balancing 대상 상태 확인을 무시해야 하는 기간(초)을 지정합니다. AWS::IoT::TopicRule DynamoDBAction 속성 유형에서는 RangeKeyField 및 RangeKeyValue 속성이 더 이상 필요하지 않습니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		AWS::KinesisAnalytics::ApplicationOutput ApplicationOutput 속성 유형에서 LambdaOutput 속성을 사용하여 애플리케이션 출력을 구성할 때 Lambda 함수를 대상으로 식별합니다. AWS::Kinesis::Stream StreamEncryption 속성을 사용하여 지정된 스트림에서 AWS KMS key를 사용해 서버 측 암호화를 활성화하거나 업데이트합니다. AWS::Lambda::Function ReservedConcurrentExecutions 속성을 사용하여 함수에 대해 예약하고 싶은 최대 동시 실행 수를 지정합니다. AWS::RDS::DBSubnetGroup DBSubnetGroupName 속성을 사용하여 DB 서브넷 그룹의 이름을 지정합니다. AWS::S3::Bucket BucketEncryption 속성을 사용하여 Amazon S3 관리형 키 SSE-S3 또는 AWS KMS keys(SSE-KMS) 버킷에서 서버 측 암호화를 사용하여 버킷에 대한 기본 암호화를 지정합니다. ReplicationRule 속성 유형에서 SourceSelectionCriteria 속성을 사용하여 복제하려는 소스 객체를 식별하기 위한 추가 필터를 지정합니다. ReplicationDestination 속성 유형의 경우: AccessControlTranslation 속성을 사용하여 대상 버킷을 소유하는 AWS 계정의 복제본 소유권을 지정합니다.	

변경	릴리스 날짜	설명	API 버전
		- Account 속성을 사용하여 대상 버킷 소유자 계정 ID를 지정합니다. - EncryptionConfiguration 속성을 사용하여 복제된 객체의 대상인 버킷에 대한 암호화 관련 정보를 지정합니다. AWS::SSM::Association AssociationName 속성을 사용하여 SSM 문서와 문서를 처리하는 구성 에이전트가 포함된 EC2 인스턴스 간의 연결 이름을 지정합니다.	
CloudFormation 콘솔에 롤백 트리거를 추가했습니다.	2018년 1월 15일	롤백 트리거를 사용하면 스택 생성 및 업데이트 중 CloudFormation이 애플리케이션의 상태를 모니터링할 수 있으며, 애플리케이션이 지정한 경보의 임계값을 위반한 경우 해당 작업을 롤백할 수 있습니다. 자세한 내용은 스택 작업 모니터링 및 롤백 단원을 참조하십시오.	2010-05-15
업데이트된 리소스	2018년 1월 12일	AWS::SSM::Parameter AllowedPattern 속성을 사용하여 파라미터값을 검증하는 데 사용되는 정규식을 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
신규 리소스	2017년 12월 5일	AWS::Inspector::AssessmentTarget AWS::Inspector::AssessmentTarget 리소스를 사용하여 Amazon Inspector 평가 대상을 생성합니다. AWS::Inspector::AssessmentTemplate AWS::Inspector::AssessmentTemplate 리소스를 사용하여 Amazon Inspector 평가 템플릿을 생성합니다. AWS::Inspector::ResourceGroup AWS::Inspector::ResourceGroup 리소스를 사용하여 Amazon Inspector 평가 대상인 AWS 리소스를 식별하는 태그를 정의하는 Amazon Inspector 리소스 그룹을 생성합니다. AWS::ServiceDiscovery::Instance AWS::ServiceDiscovery::Instance 리소스를 사용하여 Amazon Route 53에서 생성한 인스턴스에 대한 정보를 지정합니다. AWS::ServiceDiscovery::PrivateDnsNamespace AWS::ServiceDiscovery::PrivateDnsNamespace 리소스를 사용하여 Amazon Route 53에 대한 프라이빗 네임스페이스의 정보를 지정합니다. AWS::ServiceDiscovery::PublicDnsNamespace AWS::ServiceDiscovery::PublicDnsNamespace 리소스를 사용하여 Amazon Route 53에 대한 퍼블릭 네임스페이스의 정보를 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		AWS::ServiceDiscovery::Service AWS::ServiceDiscovery::Service 리소스를 사용하여 인스턴스를 등록할 때 Amazon Route 53에서 생성하기 원하는 선택 사항인 상태 확인과 최대 5개의 레코드에 대한 템플릿을 지정합니다.	
업데이트된 리소스	2017년 12월 5일	AWS::KinesisAnalytics::Application Input 속성 유형에서 InputProcessingConfiguration 속성을 사용하여 스트림에서 받은 레코드를 변환합니다.	2010-05-15
업데이트된 리소스	2017년 12월 1일	AWS::CodeBuild::Project BadgeEnabled 속성을 사용하여 프로젝트의 빌드 배지에 공개 액세스할 수 있는 URL을 생성합니다. Cache 속성을 사용하여 빌드 종속성에 대한 캐시 설정을 구성합니다. VpcConfig 속성을 사용하여 CodeBuild가 Amazon VPC의 리소스에 액세스할 수 있도록 합니다. EnvironmentVariable 속성 유형에서 Type 속성을 사용하여 환경 변수의 유형을 지정합니다.	2010-05-15
신규 리소스	2017년 11월 30일	AWS::Cloud9::EnvironmentEC2 AWS::Cloud9::EnvironmentEC2 리소스를 사용하여 AWS Cloud9에 Amazon EC2 개발 환경을 생성합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2017년 11월 29일	AWS::ECS::TaskDefinition Cpu 속성을 사용하여 작업에 필요한 CPU 단위의 수를 지정합니다. ExecutionRoleArn 속성을 사용하여 실행 역할의 ARN을 지정합니다. Memory 속성을 사용하여 작업에 필요한 메모리의 양 (MiB)을 지정합니다. RequiresCompatibilities 속성을 사용하여 작업에 필요한 시작 유형을 지정합니다. AWS::ECS::Service LaunchType 속성을 사용하여 서비스를 실행시키는 시작 유형을 지정합니다. NetworkConfiguration 속성을 사용하여 서비스의 네트워크 구성을 지정합니다. PlatformVersion 속성을 사용하여 서비스를 실행시키는 플랫폼 버전을 지정합니다.	2010-05-15
신규 리소스	2017년 11월 28일	AWS::GuardDuty::Detector AWS::GuardDuty::Detector 리소스를 사용하여 Amazon GuardDuty 감지기 1개를 생성합니다. AWS::GuardDuty::IPSet AWS::GuardDuty::IPSet 리소스를 사용하여 Amazon GuardDutyIP 세트를 생성합니다. AWS::GuardDuty::ThreatIntelSet AWS::GuardDuty::ThreatIntelSet 리소스를 사용하여 ThreatIntelSet를 생성합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2017년 11월 28일	AWS::CodeDeploy::Application ComputePlatform 속성을 사용하여 애플리케이션을 배포할 CodeDeploy에 대한 AWS Lambda 컴퓨팅 플랫폼을 지정합니다. AWS::CodeDeploy::DeploymentGroup DeploymentStyle 속성 유형에서 DeploymentType 속성을 사용하여 Lambda 컴퓨팅 플랫폼의 블루/그린 배포를 지정합니다. AWS::EC2::SpotFleet SpotFleetRequestConfigData 속성 유형에서 SpotPrice 속성은 새로운 선택 사항입니다. AWS::Lambda::Alias RoutingConfig 속성을 사용하여 AWS Lambda 함수를 두 가지 버전으로 지정해서, 각 버전을 호출할 트래픽 백분율을 지정할 수 있습니다.	2010-05-15
새 CodeDeployLambdaAliasUpdate 업데이트 정책	2017년 11월 28일	CodeDeployLambdaAliasUpdate 업데이트 정책을 사용하여 AWS::Lambda::Alias 리소스의 버전이 변경될 때 CodeDeploy 배포 작업을 수행합니다. 자세한 내용은 UpdatePolicy 속성을 참조하십시오.	2010-05-15
새 SSM 파라미터 유형	2017년 11월 21일	SSM 파라미터 유형을 사용하여 시스템 관리자 Parameter Store의 기존 파라미터를 사용합니다. 참고: CloudFormation은 현재 SecureString 유형을 지원하지 않습니다. 자세한 내용은 SSM 파라미터 유형을 참조하십시오.	2010-05-15

변경	릴리스 날짜	설명	API 버전
ResolvedValue 데이터 유형에 대한 새 Parameter 필드	2017년 11월 21일	ResolvedValue 필드는 SSM 파라미터의 스택 정의에 사용된 값을 반환합니다. 자세한 내용은 CloudFormation API 참조의 Parameter 데이터 유형을 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2017년 11월 20일	AWS::ApiGateway::ApiKey CustomerId 속성을 사용하여 AWS Marketplace 고객 식별자를 지정합니다. GenerateDistinctId 속성을 사용하여 키 식별자가 생성한 API 키 값과 다른지 여부를 지정합니다. AWS::ApiGateway::Authorizer AuthType 속성을 사용하여 기능에 영향을 주지 않고, Swagger 가져오기 및 내보내기에 사용하는 사용자 지정 필드를 지정합니다. AWS::ApiGateway::DomainName EndpointConfiguration 속성을 사용하여 특정 API Gateway 도메인 네임의 엔드포인트 유형을 지정합니다. RegionalCertificateArn 속성을 사용하여 도메인 이름에 대해 리전 엔드포인트가 사용할 인증서를 참조시킵니다. AWS::ApiGateway::Method Integration 및 IntegrationResponse 속성 유형에서 ContentHandling 속성을 사용하여 요청 페이로드 콘텐츠 유형 변환을 처리하는 방법을 지정합니다. AWS::ApiGateway::RestApi EndpointConfiguration 속성을 사용하여 API Gateway REST API의 엔드포인트 유형을 지정합니다. AWS::ApplicationAutoScaling::ScalableTarget ScheduledActions 속성을 사용하여 Application Auto Scaling 확장 가능 대상에 대해 예약된 작업을 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		AWS::ECR::Repository LifecyclePolicy 속성을 사용하여 Amazon ECR 리포지토리에 대한 수명 주기 정책을 지정합니다. AWS::ECS::TaskDefinition ContainerDefinition 속성 유형에서 LinuxParameters 속성을 사용하여 Amazon ECS 컨테이너에 대한 Linux 선택 사항을 지정합니다. AWS::ElastiCache::ReplicationGroup AtRestEncryptionEnabled 속성을 사용하여 유 휴 암호화를 활성화합니다. AuthToken 속성을 사용하여 암호로 보호된 서버에 액세스 할 때 사용하는 암호를 지정합니다. TransitEncryptionEnabled 속성을 사용하여 전송 중 암호화를 활성화합니다. AWS::ElasticLoadBalancingV2::TargetGroup Fn::GetAtt 함수의 TargetGroupName 속성을 사용하여 Elastic Load Balancing 대상 그룹의 이름을 가져옵니다. AWS::Elasticsearch::Domain VPCOptions 속성을 사용하여 OpenSearch Service 도메인에 대한 VPC 구성을 지정합니다. AWS::EMR::Cluster EbsRootVolumeSize 속성을 사용하여 Amazon EMR 클러스터의 EBS 루트 볼륨 크기를 지정합니다.	

변경	릴리스 날짜	설명	API 버전
		AWS::RDS::DBInstance SourceRegion 및 KmsKeyId 속성을 사용하여 교차 리전 소스 DB 인스턴스로부터 암호화된 읽기 전용 복제본을 생성합니다. AWS::Route53::HostedZone QueryLoggingConfig 속성을 사용하여 DNS 쿼리 로깅에 대한 구성을 지정합니다.	
고객 리소스 객체에 대한 새 NoEcho 필드 Response	2017년 11월 20일	선택 사항인 NoEcho 필드를 사용하여 사용자 지정 리소스의 출력을 마스킹 할 수 있습니다. 자세한 내용은 사용자 지정 리소스 응답 객체를 참조하십시오. 해당 noEcho 파라미터는 send 방법으로 지원됩니다. 자세한 내용은 cfn-response 모듈을 참조하십시오.	2010-05-15
스택 세트에 추가된 스택 인스턴스 재정의.	2017년 11월 17일	CloudFormation StackSet를 통해 계정과 리전별로 스택 인스턴스의 파라미터 값을 재정의할 수 있습니다. 스택 인스턴스를 생성하거나 기존 스택 인스턴스를 업데이트할 때 파라미터값을 재정의할 수 있습니다. 자세한 내용은 스택 인스턴스의 파라미터 재정의 를 참조하십시오.	2010-05-15
업데이트된 리소스	2017년 11월 15일	AWS::StepFunctions::StateMachine 상태 머신을 생성할 때 AWS::StepFunctions::StateMachine 를 사용하여 StateMachineName 을 지정할 수 있고, 상태 머신을 교체하지 않고 DefinitionString 및 RoleArn를 모두 업데이트할 수 있습니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
StackSets는 이제 스택 세트당 최대 500개의 스택 인스턴스를 지원합니다.	2017년 11월 6일	이제 스택 세트당 최대 500개의 스택 인스턴스를 생성할 수 있습니다. CloudFormation 제한에 대한 자세한 내용은 CloudFormation 할당량 이해 를 참조하세요.	2010-05-15
신규 리소스	2017년 11월 2일	AWS::CloudFront::CloudFrontOriginAccessIdentity AWS::CloudFront::CloudFrontOriginAccessIdentity 리소스를 사용하여 CloudFront 배포의 오리진과 연결할 Amazon CloudFront 원본 액세스 ID를 지정합니다. AWS::CloudFront::StreamingDistribution AWS::CloudFront::StreamingDistribution 리소스를 사용하여 CloudFront에 대한 Adobe RTMP(Real-Time Messaging Protocol) 스트리밍 배포를 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2017년 11월 2일	AWS::ApiGateway::Deployment StageName 속성은 StageDescription 속성 유형에서 사용되지 않습니다. AWS::ApiGateway::Method OperationName 속성을 사용하여 API Gateway 메서드에 기억하기 쉬운 이름을 할당합니다. RequestValidatorId 속성을 사용하여 요청 검사기를 메서드와 연결합니다. AWS::AutoScaling::AutoScalingGroup LifecycleHookSpecificationList 속성을 사용하여 Auto Scaling에서 인스턴스를 시작하거나 종료할 때 수행할 작업을 지정합니다. AWS::CloudFront::Distribution Tags 속성을 사용하여 CloudFront 배포에 연결할 임의의 태그 집합(키-값 페어)을 지정합니다. CacheBehavior 및 DefaultCacheBehavior 속성 유형에서 LambdaFunctionAssociations 속성을 사용하여 CloudFront 배포에 대한 Lambda 함수 연결을 지정합니다. CustomOriginConfig 속성 유형에서 OriginKeepaliveTimeout 속성을 사용하여 사용자 지정 연결 유지 제한 시간을 지정하고 OriginReadTimeout 속성을 사용하여 사용자 지정 오리진 읽기 제한 시간을 지정합니다. DistributionConfig 속성 유형에서 IPV6Enabled 속성을 사용하여 CloudFront에서 배포의 IPv6 주소로 IPv6 DNS 요청에 응답하는지 여부를 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		AWS::CodeDeploy::DeploymentGroup LoadBalancerInfo 속성 유형에서 TargetGroupInfoList 속성을 사용하여 배포에서 사용할 Elastic Load Balancing의 대상 그룹에 대한 정보를 지정합니다. AWS::EC2::SecurityGroup, AWS::EC2::SecurityGroupEgress 및 AWS::EC2::SecurityGroupIngress Description 속성을 사용하여 보안 그룹 규칙의 설명을 지정합니다. AWS::EC2::Subnet Ipv6CidrBlock 속성이 이제 No interruption 업데이트를 지원합니다. AWS::EC2::VPNGateway AmazonSideAsn 속성을 사용하여 Amazon 측 BGP 세션의 프라이빗 자율 시스템 번호(ASN)를 지정합니다. AWS::EC2::VPNConnection VpnTunnelOptionsSpecifications 속성을 사용하여 VPN 연결에 대한 터널 옵션을 구성합니다. AWS::ElasticBeanstalk::ConfigurationTemplate 및 AWS::ElasticBeanstalk::Environment ConfigurationOptionSetting 및 OptionSetting 속성 유형에서 ResourceName 속성을 사용하여 시간 기반 조정 구성 옵션에 대한 리소스 이름을 지정합니다. AWS::EMR::Cluster CustomAmiId 속성을 사용하여 클러스터에 대한 사용자 지정 Amazon Linux AMI를 지정합니다.	

변경	릴리스 날짜	설명	API 버전
		AWS::KinesisFirehose::DeliveryStream Arn 함수를 통해 Fn::GetAtt 속성을 사용하여 전송 스트림의 Amazon 리소스 이름(ARN)을 가져옵니다. AWS::KMS::Key Tags 속성을 사용하여 고객 관리형 키와 연결할 임의의 태그 세트(키-값 페어)를 지정합니다. AWS::OpsWorks::Layer 및 AWS::OpsWorks::Stack Tags 속성을 사용하여 OpsWorks 계층 또는 스택에 연결할 임의의 태그 세트(키-값 페어)를 지정합니다. AWS::RDS::OptionGroup OptionConfiguration 속성 유형에서 OptionVersion 속성을 사용하여 옵션 버전을 지정합니다. AWS::S3::Bucket AnalyticsConfigurations 속성을 사용하여 Amazon S3 버킷의 분석 필터를 구성합니다.	

변경	릴리스 날짜	설명	API 버전
신규 리소스	2017년 10월 24일	AWS::Glue::Classifier AWS::Glue::Classifier 리소스를 사용하여 AWS Glue 분류자를 생성합니다. AWS::Glue::Connection AWS::Glue::Connection 리소스를 사용하여 데이터 소스에 AWS Glue 연결을 지정합니다. AWS::Glue::Crawler AWS::Glue::Crawler 리소스를 사용하여 AWS Glue 크롤러를 지정합니다. AWS::Glue::Database AWS::Glue::Database 리소스를 사용하여 AWS Glue 데이터베이스를 생성합니다. AWS::Glue::DevEndpoint AWS::Glue::DevEndpoint 리소스를 사용하여 ETL 스크립트를 원격으로 디버깅하기 위한 개발 엔드포인트를 지정합니다. AWS::Glue::Job AWS::Glue::Job 리소스를 사용하여 데이터 카탈로그에서 AWS Glue 작업을 지정합니다. AWS::Glue::Partition AWS::Glue::Partition 리소스를 사용하여 테이블 데이터 조각을 나타내는 AWS Glue 파티션을 생성합니다. AWS::Glue::Table AWS::Glue::Table 리소스를 사용하여 AWS Glue 테이블을 생성합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		AWS::Glue::Trigger AWS::Glue::Trigger 리소스를 사용하여 AWS Glue 작업을 실행하는 트리거를 지정합니다.	
신규 리소스	2017년 10월 11일	AWS::SSM::MaintenanceWindow AWS::SSM::MaintenanceWindow 리소스를 사용하여 AWS Systems Manager 유지 관리 기간을 생성합니다. AWS::SSM::MaintenanceWindowTarget AWS::SSM::MaintenanceWindowTarget 리소스를 사용하여 유지 관리 기간에 대상을 등록합니다. AWS::SSM::MaintenanceWindowTask AWS::SSM::MaintenanceWindowTask 리소스를 사용하여 유지 관리 기간 작업을 정의합니다. AWS::SSM::PatchBaseline AWS::SSM::PatchBaseline 리소스를 사용하여 시스템 관리자 패치 기준을 정의합니다.	2010-05-15
신규 리소스	2017년 10월 10일	AWS::ElasticLoadBalancingV2::ListenerCertificate AWS::ElasticLoadBalancingV2::ListenerCertificate 리소스를 사용하여 Elastic Load Balancing 리스너에 대한 인증서를 지정합니다.	2010-05-15
신규 리소스	2017년 9월 27일	AWS::Athena::NamedQuery AWS::Athena::NamedQuery 리소스를 사용하여 Amazon Athena 쿼리를 생성합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2017년 9월 27일	AWS::EC2::NatGateway Tags 속성을 사용하여 NAT 게이트웨이의 리소스 태그를 지정합니다. AWS::ElasticBeanstalk::Application ResourceLifecycleConfig 속성을 사용하여 애플리케이션에 속한 리소스의 수명 주기 설정과 수명 주기 설정을 적용하기 위해 Elastic Beanstalk에서 수입하는 서비스 역할을 정의합니다. AWS::ElasticBeanstalk::ConfigurationTemplate 및 AWS::ElasticBeanstalk::Environment PlatformArn 속성을 사용하여 Elastic Beanstalk의 사용자 지정 플랫폼을 지정합니다. AWS::ElasticLoadBalancingV2::TargetGroup TargetDescription 속성 유형에서 AvailabilityZone 속성을 사용하여 IP 주소가 등록될 가용 영역을 지정합니다. AWS::Events::Rule Target 속성 유형에서 이벤트의 입력 변환 및 Amazon ECS 작업과 Kinesis 스트림 대상 설정에 대해 다음 속성을 사용합니다. • EcsParameters • InputTransformer • KinesisParameters • RunCommandParameters AWS::KinesisFirehose::DeliveryStream DeliveryStreamType 속성을 사용하여 스트림 유형을 지정하고 KinesisStreamSourceConfiguration 속성을 사용하여 전송 스트림의	2010-05-15

변경	릴리스 날짜	설명	API 버전
		소스로 사용되는 Kinesis 스트림을 위한 스트림 및 역할 ARN을 지정합니다. AWS::RDS::DBInstance Engine 속성의 경우 <code>oracle-se</code> 또는 <code>oracle-se1</code> 을 지정하면 데이터베이스 인스턴스 교체 없이 <code>oracle-se2</code> 로 업데이트할 수 있습니다. AWS::S3::Bucket <code>AccelerateConfiguration</code> 속성을 사용하여 Amazon S3 버킷의 Transfer Acceleration 상태를 구성합니다.	
스택에 종료 방지 기능이 추가되었습니다.	2017년 9월 26일	종료 방지 기능은 스택이 실수로 삭제되는 것을 방지합니다. 사용자는 종료 방지 기능이 활성화된 스택을 삭제할 수 없습니다. 자세한 내용은 스택 삭제 방지 를 참조하십시오.	2010-05-15
버전 1.4-22 부터 기본 umask 값 변경	2017년 9월 14일	<code>cfn-hup.conf</code> 구성 파일의 기본 umask 파라미터값은 현재 022입니다. 자세한 내용은 cfn-hup 를 참조하십시오.	
업데이트된 리소스	2017년 9월 7일	AWS::ElasticLoadBalancingV2::LoadBalancer <code>SubnetMappings</code> 속성을 사용하여 로드 밸런서에 연결할 서브넷의 ID를 지정합니다. <code>Type</code> 속성을 사용하여 생성할 로드 밸런서의 유형을 지정합니다. AWS::ElasticLoadBalancingV2::TargetGroup <code>TargetType</code> 속성을 사용하여 이 대상 그룹에 있는 대상의 등록 유형을 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
CloudFormation API에 롤백 트리거가 추가됨	2017년 8월 31일	롤백 트리거를 사용하면 스택 생성 및 업데이트 중 CloudFormation이 애플리케이션의 상태를 모니터링할 수 있으며, 애플리케이션이 지정한 경보의 임계값을 위반한 경우 해당 작업을 롤백할 수 있습니다. 자세한 내용을 알아보려면 CloudFormation API 참조의 RollbackConfiguration 섹션을 참조하세요.	2010-05-15
cfn-hup.conf 파일에 대한 새 umask 파라미터	2017년 8월 31일	cfn-hup.conf 구성 파일에서 umask 파라미터를 사용하여 cfn-hup 데몬에서 사용된 파일 권한을 제어합니다(버전 1.4-21). 자세한 내용은 cfn-hup 를 참조하십시오.	
VPC 크기를 위해 업데이트된 리소스	2017년 8월 29일	AWS::EC2::VPC::CidrBlock CidrBlock 속성을 사용하여 VPC에 IPv4 CIDR 블록을 연결합니다. AWS::EC2::VPC CidrBlockAssociations 속성을 Fn::GetAtt 함수와 함께 사용하여 VPC에 연결된 IPv4 CIDR 블록 연결 ID의 목록을 가져옵니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2017년 8월 23일	AWS::S3::Bucket Rule 속성 유형에서 TagFilters 속성을 사용하여 Amazon S3 버킷에 대한 객체의 하위 집합을 식별하는데 사용할 태그를 지정합니다. MetricsConfiguration 속성을 사용하여 Amazon S3 버킷에서 CloudWatch 요청 지표에 대한 지표 구성을 지정합니다. AWS::IoT::TopicRule Action 속성 유형에서 DynamoDBv2Action 속성을 사용하여 DynamoDB 테이블에 데이터를 쓰는 AWS IoT 작업을 설명합니다. Action 속성 유형에서 DynamoDBAction 속성은 이제 HashKeyType 및 RangeKeyType 속성을 지원합니다. AWS::Lambda::Permission EventSourceToken 속성을 사용하여 함수를 호출하는 보안 주체가 제공해야 하는 고유한 토큰을 지정합니다.	2010-05-15
새로운 가상 파라미터	2017년 8월 23일	AWS::Partition 가상 파라미터를 사용하여 리소스가 있는 파티션을 반환합니다. AWS::URLSuffix 가상 파라미터를 사용하여 도메인에 대한 접미사를 반환합니다. 자세한 내용은 가상 파라미터 참조를 참조하십시오.	2010-05-15

변경	릴리스 날짜	설명	API 버전
DAX 지원을 위한 새 리소스	2017년 8월 22일	AWS::DAX::Cluster AWS::DAX::Cluster 리소스를 사용하여 Amazon DynamoDB에 사용할 DAX 클러스터를 생성합니다. AWS::DAX::ParameterGroup AWS::DAX::ParameterGroup 리소스를 사용하여 Amazon DynamoDB에 사용할 파라미터 그룹을 생성합니다. AWS::DAX::SubnetGroup AWS::DAX::SubnetGroup 리소스를 사용하여 DAX(DynamoDB Accelerator)에 사용할 서브넷 그룹을 생성합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
신규 리소스	2017년 8월 18일	AWS::ApiGateway::DocumentationPart 및 AWS::ApiGateway::DocumentationPart AWS::ApiGateway::DocumentationPart 및 AWS::ApiGateway::DocumentationVersion 리소스를 사용하여 API Gateway API에 대한 설명서를 생성합니다. AWS::ApiGateway::GatewayResponse AWS::ApiGateway::GatewayResponse 리소스를 사용하여 API Gateway API에 대한 사용자 지정 응답을 생성합니다. AWS::ApiGateway::RequestValidator AWS::ApiGateway::RequestValidator 리소스를 사용하여 API Gateway API에 대한 수신 요청을 위한 검증 규칙을 설정합니다. AWS::EC2::NetworkInterfacePermission AWS::EC2::NetworkInterfacePermission 리소스를 사용하여 네트워크 인터페이스에 AWS 계정 권한을 부여합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2017년 8월 18일	AWS::ApiGateway::Stage DocumentationVersion 속성을 사용하여 API 설명서의 버전이 지정된 스냅샷을 지정합니다. AWS::AutoScaling::ScalingPolicy TargetTrackingConfiguration 속성을 사용하여 Auto Scaling 대상 추적 조정 정책 구성을 지정합니다. AWS::CloudTrail::Trail Amazon S3 데이터 이벤트 지원을 위해 EventSelectors 속성을 사용합니다. AWS::CodeDeploy::DeploymentGroup LoadBalancerInfo 및 DeploymentStyle 속성을 사용하여 인 플레이스(in-place) 배포를 위한 Elastic Load Balancing 로드 밸런서를 지정합니다. AutoRollbackConfiguration 속성을 사용하여 배포를 위한 자동 롤백을 구성합니다. AWS::EC2::SpotFleet SpotFleetRequestConfigData 속성 유형에서 ReplaceUnhealthyInstances 속성을 사용하여 스팟 플릿이 비정상 인스턴스를 대체해야 하는지 여부를 나타내고, Type 속성을 사용하여 요청 유형을 지정합니다. AWS::EC2::Subnet AssignIpv6AddressOnCreation 및 Ipv6CidrBlock 속성을 사용하여 IPv6 CIDR 블록으로 서브넷을 생성합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		AWS::KinesisFirehose::DeliveryStream ExtendedS3DestinationConfiguration 속성을 사용하여 Amazon S3에서 대상을 구성합니다. 각 대상 구성 내에서 ProcessingConfiguration 하위 속성을 사용하여 수신되는 소스 데이터를 변환하고 변환된 데이터를 대상으로 전송하는 Lambda 함수를 호출합니다. AWS::RDS::DBCluster 및 AWS::RDS::DBInstance 이제 기본 DeletionPolicy 는 DBCluster Identifier 속성을 지정하지 않는 AWS::RDS::DBCluster 리소스 및 AWS::RDS::DBInstance 리소스에 대한 Snapshot입니다. 자세한 내용은 DeletionPolicy 속성을 참조하세요. AWS::S3::Bucket Rule 속성 유형에서 AbortIncompleteMultipartUpload 속성을 사용하여 Amazon S3 버킷에 대한 불완전 멀티파트 업로드를 취소하는 수명 주기 규칙을 지정합니다. AWS::SQS::Queue KmsMasterKeyId 및 KmsDataKeyReusePeriodSeconds 속성을 사용하여 Amazon SQS에 대한 서버 측 암호화를 구성합니다. 다음 리소스를 위해 Arn 속성이 Fn::GetAtt 내장 함수에 추가됨: AWS::CloudTrail::Trail. SnsTopicArn 도 추가되었습니다. AWS::CloudWatch::Alarm	

변경	릴리스 날짜	설명	API 버전
		• AWS::DynamoDB::Table • AWS::ECS::Cluster • AWS::IoT::Policy • AWS::IoT::TopicRule • AWS::Logs::Destination	
CodePipeline 아티팩트의 스택 태그에 대한 지원	2017년 8월 18일	이제 CodePipeline 파이프라인을 위한 아티팩트로 사용하기 위해 템플릿 구성 파일의 스택에 대한 태그를 지정할 수 있습니다. 지정된 태그는 템플릿 구성 파일을 사용하여 생성된 스택에 적용됩니다. 자세한 내용을 알아보려면 CloudFormation 아티팩트 를 참조하세요.	2010-05-15
암호화된 파일 시스템 생성	2017년 8월 14일	AWS::EFS::FileSystem Encrypted 속성을 사용하여 생성 중 Amazon EFS 파일 시스템을 암호화합니다. 필요할 경우 KmsKeyId 속성을 사용하여 암호화된 파일 시스템을 보호하는 데 사용할 사용자 지정 고객 관리형 키를 지정합니다.	2010-05-15
AWS Batch 지원을 위한 새 리소스	2017년 8월 8일	AWS::Batch::ComputeEnvironment AWS::Batch::ComputeEnvironment 리소스를 사용하여 AWS Batch 컴퓨팅 환경을 정의합니다. AWS::Batch::JobDefinition AWS::Batch::JobDefinition 리소스를 사용하여 AWS Batch 작업 정의에 대한 파라미터를 지정합니다. AWS::Batch::JobQueue AWS::Batch::JobQueue 리소스를 사용하여 AWS Batch 작업 대기열을 정의합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
Amazon Managed Service for Apache Flink 지원을 위한 새로운 리소스	2017년 7월 28일	AWS::KinesisAnalytics::Application AWS::KinesisAnalytics::Application 리소스를 사용하여 Amazon Managed Service for Apache Flink 애플리케이션을 생성합니다. AWS::KinesisAnalytics::ApplicationOutput AWS::KinesisAnalytics::ApplicationOutput 리소스를 사용하여 Amazon Managed Service for Apache Flink 애플리케이션에 외부 대상을 추가합니다. AWS::KinesisAnalytics::ApplicationReferenceDataSource AWS::KinesisAnalytics::ApplicationReferenceDataSource 리소스를 사용하여 기존 Amazon Managed Service for Apache Flink 애플리케이션에 참조 데이터 소스를 추가합니다.	2010-05-15
StackSets를 사용하여 계정 및 리전에 대해 중앙에서 스택 관리	2017년 7월 25일	StackSets를 사용하면 단일 작업으로 여러 계정 및 리전에 대해 스택을 생성, 업데이트 또는 삭제할 수 있습니다. 관리자 계정을 사용하여 CloudFormation 템플릿을 정의 및 관리하고, 템플릿을 지정된 리전에 대해 선택한 대상 계정으로 스택을 프로비저닝하기 위한 기본으로 사용합니다. 자세한 내용은 StackSets를 사용하여 여러 계정 및 리전의 스택 관리 를 참조하세요.	2010-05-15
클라이언트 요청 토큰별로 스택 이벤트 보기	2017년 14월 7일	콘솔에서 스택 작업은 이벤트 탭에 클라이언트 요청 토큰을 표시합니다. 지정된 스택 작업에 의해 트리거되는 모든 이벤트는 동일한 클라이언트 요청 토큰에 할당됩니다. 이 토큰은 작업을 추적하는 데 사용할 수 있습니다. 자세한 내용은 AWS CloudFormation API 참조에서 AWS Management Console에서 CloudFormation 스택 데이터 및 리소스 보기 및 StackEvent 를 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
스택 빠른 생성 링크 사용	2017년 14월 7일	빠른 생성 링크를 사용하여 스택을 빠르게 준비하고 실행합니다. 템플릿 URL, 스택 이름 및 템플릿 파라미터를 지정하여 단일 스택 생성 마법사 페이지를 미리 채울 수 있습니다. 자세한 내용은 스택에 대한 빠른 생성 링크 생성 을 참조하십시오.	2010-05-15
AWS Database Migration Service 지원을 위한 새 리소스	2017년 7월 12일	AWS::DMS::Certificate AWS::DMS::Certificate 리소스를 사용하여 AWS DMS 엔드포인트와 복제 인스턴스 간의 연결을 암호화하는 SSL 인증서를 생성합니다. AWS::DMS::Endpoint AWS::DMS::Endpoint 리소스를 사용하여 AWS DMS 엔드포인트를 생성합니다. AWS::DMS::EventSubscription AWS::DMS::EventSubscription 리소스를 사용하여 Amazon Simple Notification Service를 통해 AWS DMS 이벤트에 대한 알림을 받습니다. AWS::DMS::ReplicationInstance AWS::DMS::ReplicationInstance 리소스를 사용하여 AWS DMS 복제 인스턴스를 생성합니다. AWS::DMS::ReplicationSubnetGroup AWS::DMS::ReplicationSubnetGroup 리소스를 사용하여 AWS DMS 복제 서브넷 그룹을 생성합니다. AWS::DMS::ReplicationTask AWS::DMS::ReplicationTask 리소스를 사용하여 AWS DMS 복제 작업을 생성합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
신규 리소스	2017년 7월 5일	AWS::CloudWatch::Dashboard AWS::CloudWatch::Dashboard 리소스를 사용하여 CloudWatch 콘솔에 대한 사용자 지정 CloudWatch 대시보드를 지정합니다. AWS::ApiGateway::DomainName AWS::ApiGateway::DomainName 리소스를 사용하여 Amazon API Gateway에 배포된 API에 대해 알기 쉬운 사용자 지정 URL을 지정합니다. AWS::EC2::EgressOnlyInternetGateway AWS::EC2::EgressOnlyInternetGateway 리소스를 사용하여 VPC에 대한 외부 전용 인터넷 게이트웨이를 생성합니다. InstanceFleetConfig InstanceFleetConfig 리소스를 사용하여 Amazon EMR 클러스터에 대한 스팟 인스턴스 풀릿을 구성합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2017년 7월 5일	AWS::ApiGateway::RestApi BinaryMediaTypes 속성을 사용하여 지원되는 이진 미디어 형식을 지정합니다. AWS::ApplicationAutoScaling::ScalingPolicy TargetTrackingScalingPolicyConfiguration 속성을 사용하여 대상 추적 조정 정책 구성을 지정합니다. AWS::CloudTrail::Trail TrailName 속성을 사용하여 AWS CloudTrail 리소스에 대한 사용자 지정 이름을 지정합니다. Tags 속성을 사용하여 리소스 태그를 지정합니다. AWS::CodeDeploy::DeploymentGroup AlarmConfiguration 속성을 사용하여 배포 그룹을 위한 경보를 구성합니다. TriggerConfigurations 속성을 사용하여 배포 그룹을 위한 알림 트리거를 구성합니다. AWS::EMR::Cluster JobFlowInstancesConfig 속성 유형의 CoreInstanceFleet 속성 및 MasterInstanceFleet 속성을 사용하여 Amazon EMR 클러스터에 대한 스팟 인스턴스 플릿을 구성합니다. AWS::DynamoDB::Table TimeToLiveSpecification 속성을 사용하여 Amazon DynamoDB 테이블에 대한 TTL(Time to Live) 설정을 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		Tags 속성을 사용하여 DynamoDB 테이블에 대한 리소스 태그를 지정합니다. AWS::EC2::Instance IamInstanceProfile 속성이 이제 No interruption 업데이트를 지원합니다. AWS::EC2::Route EgressOnlyInternetGatewayId 속성을 사용하여 EC2 경로에 대한 외부 전용 인터넷 게이트웨이를 지정합니다. AWS::Kinesis::Stream RetentionPeriodHours 속성을 사용하여 샤드에 저장된 데이터 레코드에 계속 액세스할 수 있는 기간(시간)을 지정합니다. AWS::RDS::DBCluster ReplicationSourceIdentifier 속성을 사용하여 Amazon RDS MySQL DB 인스턴스 또는 다른 DB 클러스터의 읽기 전용 복제본으로 DB 클러스터를 생성합니다. AWS::Redshift::Cluster LoggingProperties 속성을 사용하여 감사 로그 파일을 생성하고 Amazon S3에 저장합니다.	
신규 리소스	2017년 6월 6일	AWS::EMR::SecurityConfiguration AWS::EMR::SecurityConfiguration 리소스를 사용하여 보안 구성을 생성합니다. 이 구성은 서비스에 저장되며 클러스터 생성 시 지정할 수 있습니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2017년 6월 6일	AWS::AutoScaling::LifecycleHook NotificationTargetARN 및 RoleARN 속성은 이제 선택 사항입니다. AWS::CloudWatch::Alarm 이제 AWS::CloudWatch::Alarm 리소스를 생성할 때 EvaluateLowSampleCountPercentile, ExtendedStatistic 및 TreatMissingData 속성을 사용할 수 있습니다. AWS::EC2::SpotFleet CloudFormation에서는 스팟 플릿 속성에 대한 변경 가능한 변경을 지원합니다. SpotFleetRequestConfigData 속성의 다음 속성은 Replacement 업데이트를 지원합니다. • AllocationStrategy • IamFleetRole • LaunchSpecifications • SpotPrice • TerminateInstancesWithExpiration • ValidFrom • ValidUntil SpotFleetRequestConfigData 속성의 다음 속성은 No interruption 업데이트를 지원합니다. • ExcessCapacityTerminationPolicy • TargetCapacity AWS::EMR::InstanceGroupConfig 이제 CloudFormation에서는 Amazon EMR 작업 인스턴스 그룹에 대한 Auto Scaling을 지원합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		AWS::Events::Rule RoleArn 속성은 Rule 리소스에서 사용되지 않습니다. Target 속성 유형에 대한 RoleArn 속성을 사용하여 대상에 사용할 IAM 역할을 지정합니다. AWS::Kinesis::Stream ShardCount 속성이 이제 No interruption 업데이트를 지원합니다. AWS::Lambda::Function TracingConfig 속성을 사용하여 Lambda 함수에 대한 추적 설정을 구성합니다. AWS::Redshift::Cluster, AWS::Redshift::ClusterParameterGroup, AWS::Redshift::ClusterSecurityGroup, 및 AWS::Redshift::ClusterSubnetGroup Tags 속성을 사용하여 리소스 태그를 지정합니다. AWS::RDS::DBCluster ReadEndpoint.Address 내장 함수에 Fn::GetAtt 속성이 추가되었습니다. AWS::S3::Bucket Arn 내장 함수에 Fn::GetAtt 속성이 추가되었습니다.	

변경	릴리스 날짜	설명	API 버전
신규 리소스	2017년 5월 11일	다음 신규 리소스에서는 Elastic Load Balancing(ELB) Application Load Balancer에서 AWS WAF 사용을 지원합니다. AWS::WAFRegional::ByteMatchSet AWS::WAFRegional::ByteMatchSet 리소스를 사용하여 검사하려는 웹 요청의 일부를 식별합니다. AWS::WAFRegional::IPSet AWS::WAFRegional::IPSet 리소스를 사용하여 요청이 시작되는 IP 주소를 기반으로 허용하거나 차단할 웹 요청을 지정합니다. AWS::WAFRegional::Rule AWS::WAFRegional::Rule 리소스를 사용하여 허용, 차단 또는 계산할 웹 요청을 식별하는 IPSet, ByteMatchSet 및 SqlInjectionMatchSet 객체의 조합을 지정합니다. AWS::WAFRegional::SizeConstraintSet AWS::WAFRegional::SizeConstraintSet 리소스를 사용하여 웹 요청의 크기와 검사할 요청의 부분을 확인하는 데 사용되는 크기 제약을 지정합니다. AWS::WAFRegional::SqlInjectionMatchSet AWS::WAFRegional::SqlInjectionMatchSet 리소스를 사용하여 웹 요청의 특정 부분에 악성 SQL 코드가 포함된 요청을 허용, 차단 또는 계산합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		<u>AWS::WAFRegional::WebACL</u> AWS::WAFRegional::WebACL 리소스를 사용하여 허용, 차단 또는 계산하려는 웹 요청을 식별합니다. <u>AWS::WAFRegional::WebACLAssociation</u> AWS::WAFRegional::WebACLAssociation 리소스를 사용하여 웹 액세스 제어 그룹(ACL)을 리소스와 연결합니다. <u>AWS::WAFRegional::XssMatchSet</u> AWS::WAFRegional::XssMatchSet 리소스를 사용하여 AWS WAF에서 크로스 사이트 스크립팅 공격이 있는지 검사할 웹 요청 부분과 검사할 헤더의 이름을 지정합니다.	

변경	릴리스 날짜	설명	API 버전
신규 리소스	2017년 4월 28일	AWS::Cognito::IdentityPool AWS::Cognito::IdentityPool 리소스를 사용하여 Amazon Cognito 자격 증명 풀을 생성합니다. AWS::Cognito::IdentityPoolRoleAttachment AWS::Cognito::IdentityPoolRoleAttachment 리소스를 사용하여 Amazon Cognito 자격 증명 풀에 대한 역할 구성을 관리합니다. AWS::Cognito::UserPool AWS::Cognito::UserPool 리소스를 사용하여 Amazon Cognito 사용자 풀을 생성합니다. AWS::Cognito::UserPoolClient AWS::Cognito::UserPoolClient 리소스를 사용하여 사용자 풀 클라이언트를 생성합니다. AWS::Cognito::UserPoolGroup AWS::Cognito::UserPoolGroup 리소스를 사용하여 Amazon Cognito 사용자 풀에서 사용자 그룹을 생성합니다. AWS::Cognito::UserPoolUser AWS::Cognito::UserPoolUser 리소스를 사용하여 Amazon Cognito 사용자 풀 사용자를 생성합니다. AWS::Cognito::UserPoolUserToGroupAttachment AWS::Cognito::UserPoolUserToGroupAttachment 리소스를 사용하여 Amazon Cognito 사용자 풀 그룹에 사용자를 연결합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2017년 4월 28일	SourceDetails AWS::Config::ConfigRule 리소스의 MaximumExecutionFrequency 하위 속성을 사용하여 주기적 트리거를 통해 사용자 지정 규칙에 대한 평가를 실행합니다. AWS::EC2::Volume 이제 CloudFormation에서 Amazon Elastic Block Store(Amazon EBS)에 대한 탄력적인 볼륨을 지원합니다. 이제 VolumeType , Size 및 Iops 등 세 가지 속성에 대한 No interruption 업데이트를 지원합니다. AWS::EC2::SecurityGroup GroupName 속성을 사용하여 Amazon EC2 보안 그룹의 이름을 지정합니다. AWS::ECS::Service AWS::ECS::Service : PlacementConstraints , PlacementStrategies 및 ServiceName 에 대한 새로운 속성이 세 개 있습니다. AWS::ECS::TaskDefinition PlacementConstraints 속성을 사용하여 서비스에서 작업에 대한 배치 제약을 정의합니다. AWS::ElastiCache::ReplicationGroup Fn::GetAtt 내장 함수에 ConfigurationEndpoint.Address 속성 및 ConfigurationEndpoint.Port 속성이 추가되었습니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		<u>AWS::ElasticLoadBalancingV2::LoadBalancer</u> IpAddressType 속성을 사용하여 로드 밸런서의 서브넷에서 사용하는 IP 주소의 유형을 지정합니다. <u>AWS::EMR::Cluster</u> 이제 CloudFormation에서는 Amazon EMR 클러스터에 대한 Auto Scaling을 지원합니다. <u>AWS::IAM::ManagedPolicy</u> ManagedPolicyName 속성을 사용하여 IAM 관리형 정책의 사용자 지정 이름을 설정합니다. <u>AWS::Lambda::Function</u> Tags 속성을 사용하여 Lambda 함수에 태그를 추가합니다. <u>AWS::OpsWorks::Instance</u> Fn::GetAtt 내장 함수에 AvailabilityZone , PrivateDnsName , PrivateIp 및 PublicDnsName 속성이 추가되었습니다. <u>AWS::OpsWorks::UserProfile</u> SshUsername 속성을 사용하여 사용자의 SSH 이름을 지정합니다. SshUsername 내장 함수에 Fn::GetAtt 속성이 추가되었습니다. <u>AWS::Redshift::Cluster</u> IamRoles 속성을 사용하여 Amazon Redshift 클러스터가 기타 AWS 서비스에 액세스하는 데 사용할 수 있는 하나 이상의 AWS Identity and Access Management 역할 목록을 제공합니다.	

변경	릴리스 날짜	설명	API 버전
CloudFormation Designer를 사용하여 YAML 및 JSON 형식의 템플릿 편집	2017년 4월 6일	Designer를 사용하여 CloudFormation 템플릿을 생성하는 경우 이제 통합 편집기에서 YAML 및 JSON 형식의 템플릿을 둘 다 편집할 수 있습니다. 또한 기본 템플릿 작성 언어에 따라 JSON 템플릿을 YAML 템플릿으로 변환하거나 그 반대로 변환할 수도 있습니다. 자세한 내용은 CloudFormation Designer란 무엇입니까? 를 참조하세요.	2010-05-15
신규 리소스	2017년 4월 6일	AWS::SSM::Parameter AWS::SSM::Parameter 리소스를 사용하여 Parameter Store에서 SSM 파라미터를 생성합니다.	2010-05-15
AWS::Include 변환	2017년 3월 28일	AWS::Include 변환을 사용하여 Amazon S3 버킷에 저장된 재사용 가능 코드 조각을 참조합니다. 자세한 내용은 AWS::Include 변환 을 참조하세요.	2010-05-15
다른 계정과 Amazon VPC 피어링	2017년 3월 28일	CloudFormation을 사용하여 자신의 Amazon VPC를 다른 AWS 계정의 VPC와 피어링할 수 있습니다. 자세한 내용을 알아보려면 다른 AWS 계정의 Amazon VPC와 피어링 을 참조하세요.	2010-05-15
신규 리소스	2017년 3월 28일	AWS::ApiGateway::UsagePlanKey AWS::ApiGateway::UsagePlanKey 리소스를 사용하여 사용량 계획 키를 연결하고 해당 사용량 계획이 적용되는 사용자를 결정할 수 있습니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2017년 3월 28일	AWS::EC2::VPCPeeringConnection PeerOwnerId 및 PeerRoleArn 속성을 사용하여 다른 AWS 계정의 VPC와 피어링합니다. 자세한 내용을 알아보려면 다른 AWS 계정의 Amazon VPC와 피어링을 참조하세요. AWS::IAM::InstanceProfile InstanceProfileName 속성을 사용하여 인스턴스 프로파일을 구성합니다. AWS::Lambda::Function DeadLetterConfig 속성을 사용하여 AWS Lambda에서 처리 불가능한 이벤트를 처리하는 방법을 구성합니다. Runtime 속성에 대해 Node.js v0.10이 더 이상 지원되지 않습니다. AWS::Route53::HealthCheck HealthCheckConfig HealthCheckConfig 속성에 대한 새로운 리소스 하위 속성 유형 7개 즉, AlarmIdentifier , ChildHealthChecks , EnableSNI , HealthThreshold , InsufficientDataHealthStatus , Inverted 및 MeasureLatency 가 있습니다. AWS::SQS::Queue ContentBasedDeduplication 및 FifoQueue 속성을 사용하여 FIFO(선입선출) Amazon Simple Queue Service 대기열을 생성합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		AWS::S3::Bucket 이제 Amazon S3 버킷에 대한 IPv6 도메인 이름을 지정할 수 있습니다.	
신규 리소스	2017년 2월 10일	AWS::StepFunctions::Activity AWS::StepFunctions::Activity 리소스를 사용하여 AWS Step Functions 활동을 생성합니다. AWS::StepFunctions::StateMachine AWS::StepFunctions::StateMachine 리소스를 사용하여 Step Functions 상태 시스템을 생성합니다.	2010-05-15
새로운 내장 함수	2017년 1월 17일	Fn::Split 함수를 사용하여 문자열을 문자열 값 목록으로 분할합니다. 자세한 내용은 Fn::Split 섹션을 참조하세요.	2010-05-15
가져오기 목록에 대한 콘솔 지원	2017년 1월 17일	CloudFormation 콘솔을 사용하여 내보낸 출력 값을 가져오는 모든 스택을 볼 수 있습니다. 자세한 내용은 출력값을 가져오고 내보내는 스택 나열 을 참조하십시오.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2017년 1월 17일	AWS::AutoScaling::AutoScalingGroup LoadBalancerNames 속성은 Auto Scaling 그룹을 바꾸지 않고 업데이트할 수 있습니다. AWS::ECS::TaskDefinition NetworkMode 및 MemoryReservation 속성이 추가되었습니다. AWS::RDS::DBCluster CloudFormation에서는 Tags 속성에 대한 업데이트를 지원합니다. AWS::RDS::DBInstance Timezone 속성이 추가되었습니다. FirehoseAction Separator 속성이 추가되었습니다. AWS::OpsWorks::Instance Fn::GetAtt 내장 함수에 대한 PublicIp 속성이 추가되었습니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
신규 리소스	2016년 12월 1일	AWS::CodeBuild::Project AWS::CodeBuild::Project 리소스를 사용하여 CodeBuild에서 소스 코드를 빌드하는 방법을 정의하는 AWS CodeBuild 프로젝트를 생성합니다. AWS::SSM::Association AWS::SSM::Association 리소스를 사용하여 Amazon EC2 Systems Manager 문서를 EC2 인스턴스와 연결합니다. AWS::EC2::SubnetCidrBlock AWS::EC2::SubnetCidrBlock 리소스를 사용하여 단일 IPv6 CIDR 블록을 Amazon VPC 서브넷과 연결합니다. AWS::EC2::VPCCidrBlock AWS::EC2::VPCCidrBlock 리소스를 사용하여 Amazon 제공 단일 IPv6 CIDR 블록을 Amazon VPC와 연결합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
IPv6 지원을 위해 업데이트된 리소스	2016년 12월 1일	AWS::EC2::Instance Ipv6AddressCount 및 Ipv6Addresses 속성이 추가되었습니다. AWS::EC2::NetworkAclEntry Ipv6CidrBlock 속성이 추가되었습니다. AWS::EC2::NetworkInterface Ipv6AddressCount 및 Ipv6Addresses 속성이 추가되었습니다. AWS::EC2::Route DestinationIpv6CidrBlock 속성이 추가되었습니다. AWS::EC2::SecurityGroupEgress CidrIpv6 속성이 추가되었습니다. AWS::EC2::SecurityGroupIngress CidrIpv6 속성이 추가되었습니다. AWS::EC2::SpotFleet 시작 사양 네트워크 인터페이스에 대한 Ipv6AddressCount 및 Ipv6Addresses 속성이 추가되었습니다. AWS::EC2::Subnet Ipv6CidrBlocks 함수에 대한 Fn::GetAtt 속성이 추가되었습니다. AWS::EC2::VPC Ipv6CidrBlocks 함수에 대한 Fn::GetAtt 속성이 추가되었습니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		AWS::SSM::Document DocumentType 속성이 추가되었습니다.	
리소스 사양	2016년 11월 22일	CloudFormation 리소스 사양으로 CloudFormation 템플릿을 생성하기 위한 도구를 빌드합니다. 사양은 머신 판독 가능한 JSON 형식의 텍스트 파일입니다. 자세한 내용을 알아보려면 CloudFormation 리소스 사양 을 참조하세요.	2010-05-15
신규 리소스	2016년 11월 22일	AWS::OpsWorks::UserProfile AWS::OpsWorks::UserProfile 리소스를 사용하여 OpsWorks 스택의 인스턴스에 대한 액세스 권한이 필요한 사용자에게 SSH 액세스를 구성합니다. AWS::OpsWorks::Volume AWS::OpsWorks::Volume 리소스를 사용하여 OpsWorks 스택에 Amazon Elastic Block Store 볼륨을 등록합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2016년 11월 22일	AWS::OpsWorks::App DataSource 속성이 추가되었습니다. AWS::OpsWorks::Instance BlockDeviceMappings , AgentVersion , ElasticIps , Hostname, Tenancy 및 Volumes 속성이 추가되었습니다. AWS::OpsWorks::Layer CustomJson 및 VolumeConfigurations 속성이 추가되었습니다. AWS::OpsWorks::Stack ElasticIps , EcsClusterArn , RdsDbInstances , CloneAppIds , ClonePermissions 및 SourceStackId 속성이 추가되었습니다. AWS::RDS::DBInstance CopyTagsToSnapshot 속성이 추가되었습니다.	2010-05-15
가져오기 나열	2016년 11월 22일	값을 가져오는 CloudFormation 스택을 추적하기 위해 내보낸 출력 값의 가져오기를 나열합니다. 자세한 내용은 출력값을 가져오고 내보내는 스택 나열 을 참조하십시오.	2010-05-15
변형	2016년 11월 17일	서버리스 애플리케이션에 대한 AWS SAM 구문을 처리하는 데 사용하는 AWS Serverless Application Model(AWS SAM)을 지정합니다. 자세한 내용은 변환 을 참조하세요.	2010-05-15
신규 리소스	2016년 11월 17일	AWS::SNS::Subscription AWS::SNS::Subscription 리소스를 사용하여 Amazon Simple Notification Service 주제에 엔드포인트를 가입합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2016년 11월 17일	AWS::Lambda::Function Environment 속성을 사용하여 AWS Lambda 함수에서 액세스할 수 있는 키-값 페어(환경 변수)를 지정합니다. KmsKeyArn 속성을 사용하여 AWS Lambda에서 환경 변수를 암호화 및 암호 해독하는 데 사용할 KMS 키를 지정합니다.	2010-05-15
새 CLI 명령	2016년 11월 17일	S3 버킷에 로컬 아티팩트 업로드 package 명령을 사용하여 CloudFormation 템플릿에서 참조되는 로컬 아티팩트를 S3 버킷으로 업로드합니다. 변형이 포함된 템플릿을 빠르게 배포 deploy 명령을 사용하여 변경 세트 생성 및 실행 작업을 단일 명령으로 결합합니다. 이 명령을 변형이 포함된 스택을 빠르게 생성 또는 업데이트하는 데 유용합니다.	2010-05-15
업데이트된 리소스	2016년 11월 3일	AWS::CloudFront::Distribution DistributionConfig 속성의 경우, HttpVersion 속성을 사용하여 최종 사용자가 Amazon CloudFront와 통신하는 데 사용할 수 있는 최신 HTTP 버전을 지정합니다. ForwardedValues 속성의 경우 QueryStringCacheKeys 속성을 사용하여 CloudFront에서 캐시할 콘텐츠를 결정하는 데 사용하는 쿼리 문자열 파라미터를 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
스택 내보내기 나열	2016년 11월 3일	CloudFormation 콘솔, API 또는 AWS CLI를 사용하여 리전에 대해 내보낸 모든 출력값 목록을 봅니다. 자세한 내용은 스택 출력값 내보내기 를 참조하십시오.	2010-05-15
스택이 포함된 지속적 전달	2016년 11월 3일	AWS CodePipeline을 사용하여 CloudFormation 스택이 있는 지속적 전달 워크플로우를 빌드합니다. 자세한 내용은 CodePipeline을 사용한 지속적 전달 을 참조하십시오.	2010-05-15
롤백 중 리소스 건너뛰기	2016년 11월 3일	UPDATE_ROLLBACK_FAILED 상태인 스택이 있는 경우 ContinueUpdateRollback 작업에 ResourcesToSkip 파라미터를 사용하여 CloudFormation에서 롤백할 수 없는 리소스를 건너뜁니다. 자세한 내용은 롤백 업데이트 실패 의 문제 해결 단원을 참조하십시오.	2010-05-15
변경 세트 개선행 사항	2016년 11월 3일	변경 세트를 사용하여 새로운 스택을 생성 할 수 있습니다.	2010-05-15
업데이트된 리소스	2016년 10월 12일	AWS::ElastiCache::CacheCluster 클러스터를 바꾸지 않고 CacheNodeType 속성을 업데이트합니다. AWS::ElastiCache::ReplicationGroup 여러 노드 그룹(샤드)을 포함할 수 있는 Redis(클러스터 모드 활성화) 복제 그룹을 생성할 수 있습니다. 각 노드 그룹에는 기본 클러스터와 읽기 전용 복제본이 있습니다. AWS::ElastiCache::SubnetGroup CacheSubnetGroupName 속성을 사용하여 Amazon ElastiCache 서브넷 그룹의 이름을 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
신규 리소스	2016년 10월 6일	AWS::ApiGateway::UsagePlan AWS::ApiGateway::UsagePlan 리소스를 사용하여 배포된 Amazon API Gateway API에 대한 사용량 계획을 지정합니다. AWS::CodeCommit::Repository AWS::CodeCommit::Repository 리소스를 사용하여 Amazon Web Services에서 호스팅하는 CodeCommit 리포지토리를 생성합니다.	2010-05-15
업데이트된 리소스	2016년 10월 6일	AWS::ApiGateway::Authorizer ProviderARNs 속성을 사용하여 Amazon Cognito 사용자 풀을 Amazon API Gateway API 권한 부여자로 사용합니다. AWS::ApiGateway::Deployment StageName 속성은 더 이상 필요하지 않습니다. AWS::ElasticLoadBalancingV2::TargetGroup GetAtt 함수의 경우 LoadBalancerArns 속성을 사용하여 트래픽을 대상 그룹으로 라우팅하는 로드 밸런서의 Amazon 리소스 이름(ARN)을 검색합니다. AWS::RDS::DBInstance 사용자가 RDS DB 인스턴스에 연결되는 경우 Domain 및 DomainIAMRoleName 속성을 사용하여 Windows 인증을 사용합니다. AWS::EC2::SecurityGroupEgress DestinationPrefixListId 속성을 사용하여 Amazon VPC 엔드포인트의 AWS 서비스 접두사를 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
교차 스택 참조 개선 사항	2016년 10월 6일	내장 함수를 사용하여 내보내기 의 Name 값을 사용자 지정하거나 ImportValue 함수의 값을 참조합니다.	2010-05-15
CloudFormation 서비스 역할	2016년 9월 26일	CloudFormation 스택 작업에 AWS Identity and Access Management(IAM) 서비스 역할을 사용합니다. CloudFormation은 역할의 자격 증명을 사용하여 사용자를 대신하여 스택 리소스를 호출합니다. 자세한 내용은 CloudFormation 서비스 역할 을 참조하세요.	2010-05-15
새로운 기능	2016년 9월 19일	Export 출력 필드와 Fn::ImportValue 내장 함수를 사용하여 스택 하나가 다른 스택의 리소스 출력을 참조하도록 할 수 있습니다. 자세한 내용은 출력 , Fn::ImportValue 및 연습: 다른 CloudFormation 스택의 리소스 출력 참조 를 참조하세요.	2010-05-15
YAML 지원	2016년 9월 19일	YAML 형식을 사용하여 CloudFormation 템플릿을 작성할 수 있습니다. 또한 YAML에서는 예를 들어, 템플릿에 주석을 추가하거나 내장 함수에 짧은 형식을 사용할 수 있습니다. 자세한 내용은 CloudFormation 템플릿 형식 을 참조하세요.	2010-05-15
새로운 내장 함수	2016년 9월 19일	Fn::Sub 함수를 사용하여 입력 문자열의 변수를 지정한 값으로 바꿉니다. 자세한 내용은 Fn::Sub 섹션을 참조하세요.	2010-05-15
신규 리소스	2016년 9월 19일	AWS::KMS::Alias AWS::KMS::Alias 리소스를 사용하여 AWS KMS key에 대한 별칭을 생성합니다.	

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2016년 9월 19일	AWS::EC2::SpotFleet LaunchSpecifications 속성의 경우 SpotPrice 속성을 사용하여 특정 인스턴스 유형에 대한 입찰 가격을 지정합니다. AWS::ECS::Cluster ClusterName 속성을 사용하여 Amazon Elastic Container Service 클러스터의 이름을 지정합니다. AWS::ECS::TaskDefinition TaskRoleArn 속성을 사용하여 Amazon Elastic Container Service 컨테이너가 사용자를 대신해 AWS를 호출하는 데 사용하는 AWS Identity and Access Management 역할을 지정합니다. Family 속성을 사용하여 특정 패밀리에 대한 작업 정의를 등록합니다. AWS::Elasticsearch::Domain ElasticsearchVersion 속성을 사용하여 사용할 Elasticsearch의 버전을 지정합니다.	2010-05-15
신규 리소스	2016년 8월 11일	다음 Elastic Load Balancing Application Load Balancer 리소스를 사용하여 여러 가용 영역에서 EC2 인스턴스 같은 여러 대상에 수신 애플리케이션 트래픽을 분산합니다. • AWS::ElasticLoadBalancingV2::Listener • AWS::ElasticLoadBalancingV2::ListenerRule • AWS::ElasticLoadBalancingV2::LoadBalancer • AWS::ElasticLoadBalancingV2::TargetGroup	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2016년 8월 11일	AWS::AutoScaling::AutoScalingGroup TargetGroupARNs 속성을 사용하여 Auto Scaling 그룹을 하나 이상의 Application Load Balancer 대상 그룹과 연결합니다. AWS::ECS::Service 로드 LoadBalancers 속성의 경우 TargetGroupArn 속성을 사용하여 Amazon Elastic Container Service 서비스를 Application Load Balancer 대상 그룹과 연결합니다.	2010-05-15
신규 리소스	2016년 8월 09일	CloudFormation에서 다음 리소스가 추가되었습니다. AWS::ApplicationAutoScaling::ScalableTarget 및 AWS::ApplicationAutoScaling::ScalingPolicy Application Auto Scaling 조정 정책을 사용하여 대상 리소스 조정 시점 및 방법을 정의합니다. AWS::CertificateManager::Certificate 보안 연결을 활성화하기 위해 기타 AWS 서비스와 함께 사용할 수 있는 AWS Certificate Manager 인증서를 프로비저닝합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2016년 8월 09일	CloudFormation에서 다음 리소스가 업데이트되었습니다. AWS::CloudFront::Distribution 배포 구성 ViewerCertificate 속성의 경우 AWS Certificate Manager 인증서를 지정할 수 있습니다. 배포 구성 Origin 속성의 경우 사용자 지정 오리진에 대한 사용자 지정 헤더 및 SSL 프로토콜을 지정할 수 있습니다. AWS::EFS::FileSystem Amazon Elastic File System 파일 시스템의 성능 모드를 지정할 수 있습니다.	2010-05-15
신규 리소스	2016년 7월 20일	AWS IoT AWS IoT를 사용하여 AWS IoT 정책, X.509 인증서, 정책과 보안 주체(X.509 인증서 또는 기타 자격 증명) 간의 연결, AWS IoT 사물, 보안 주체와 사물 간의 연결 또는 AWS IoT 규칙을 선언합니다. • AWS::IoT::Certificate • AWS::IoT::Policy • AWS::IoT::PolicyPrincipalAttachment • AWS::IoT::Thing • AWS::IoT::ThingPrincipalAttachment • AWS::IoT::TopicRule	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2016년 7월 20일	CloudFormation에서 다음 리소스가 업데이트되었습니다. AWS::IAM::Group, AWS::IAM::Role, AWS::IAM::User 이름 속성을 사용하여 AWS Identity and Access Management(IAM) 리소스에 대한 사용자 지정 이름을 지정합니다. AWS::ApiGateway::Method Integration 속성의 경우 PassthroughBehavior 속성을 사용하여 Amazon API Gateway에서 대상 백엔드로 요청을 전달하는 시점을 지정합니다. AWS::ApiGateway::Model 및 AWS::ApiGateway::RestApi Schema 및 Body 속성에 대한 JSON 객체를 지정할 수 있습니다.	2010-05-15
Auto Scaling 그룹 UpdatePolicy	2016년 6월 9일	UpdatePolicy 속성의 경우 AutoScalingReplacingUpdate 속성을 사용하여 Auto Scaling 그룹 업데이트 시 Auto Scaling 그룹과 이 그룹에 포함된 인스턴스를 바꿀지 여부를 지정합니다. 교체 도중 CloudFormation은 새 Auto Scaling 그룹을 생성할 때까지 이전 그룹을 유지하므로 업데이트에 실패하더라도 CloudFormation은 이전 Auto Scaling 그룹으로 롤백할 수 있습니다. 자세한 내용은 UpdatePolicy 속성을 참조하십시오.	2010-05-15

변경	릴리스 날짜	설명	API 버전
신규 리소스	2016년 6월 9일	CloudFormation에서 다음 리소스가 추가되었습니다. AWS::EC2::FlowLog 지정된 네트워크 인터페이스, 서브넷 또는 VPC에 대한 IP 트래픽을 캡처하는 Amazon Elastic Compute Cloud 흐름 로그를 생성합니다. AWS::KinesisFirehose::DeliveryStream 대상(예: Amazon Simple Storage Service, Amazon Redshift 또는 Amazon OpenSearch Service)에 실시간 스트리밍 데이터를 전달하는 전달 스트림을 생성합니다.	2010-05-15
업데이트된 리소스	2016년 6월 9일	CloudFormation에서 다음 리소스가 업데이트되었습니다. AWS::Kinesis::Stream Name 속성을 사용하여 Amazon Kinesis 스트림의 이름을 지정합니다. AWS::Lambda::Function Code 속성의 경우 ZipFile 실행 시간 환경에 nodejs4.3 속성 및 cfn 응답 모듈을 사용합니다. AWS::SNS::Topic CloudFormation에서는 Amazon Simple Notification Service 주제 리소스에 대한 업데이트를 지원합니다.	2010-05-15
신규 리소스	2016년 4월 25일	AWS::EC2::Host 리소스를 사용하여 EC2 인스턴스를 시작하는 데 사용되는 전용 물리적 서버를 할당합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2016년 4월 25일	AWS::EC2::Instance Affinity 및 HostId 속성을 사용하여 Amazon Elastic Compute Cloud 전용 호스트에서 인스턴스를 시작합니다. AWS::ECS::Service DeploymentConfiguration 속성을 사용하여 배포 중 실행 가능한 작업 수를 구성합니다. AWS::ECS::TaskDefinition CloudFormation에서 추가 Amazon Elastic Container Service 컨테이너 정의 속성에 대한 지원이 추가되었습니다. AWS::GameLift::Fleet MaxSize 및 MinSize 속성을 사용하여 Amazon GameLift Servers 플릿에서 허용되는 EC2 인스턴스의 최대 및 최소 개수를 지정합니다. AWS::Lambda::Function FunctionName 속성을 사용하여 AWS Lambda 함수의 이름을 지정합니다. 또한 Python 2.7을 사용하여 인라인 함수를 지정할 수도 있습니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
신규 리소스	2016년 4월 18일	Amazon API Gateway Amazon API Gateway 리소스를 사용하여 모든 규모에서 API를 게시, 관리 및 모니터링합니다. 클라이언트에서 백엔드 서비스(예: EC2 인스턴스를 실행 중인 애플리케이션 또는 AWS Lambda에서 실행 중인 코드)에 액세스하기 위해 호출할 수 있는 API를 생성할 수 있습니다. • AWS::ApiGateway::Account • AWS::ApiGateway::ApiKey • AWS::ApiGateway::Authorizer • AWS::ApiGateway::BasePathMapping • AWS::ApiGateway::ClientCertificate • AWS::ApiGateway::Deployment • AWS::ApiGateway::Method • AWS::ApiGateway::Model • AWS::ApiGateway::Resource • AWS::ApiGateway::RestApi • AWS::ApiGateway::Stage • AWS::Events::Rule 자신의 계정에서 AWS 리소스에 대한 변경 사항(이벤트)을 모니터링하는 Amazon CloudWatch Events 규칙을 생성합니다. 수신 이벤트가 규칙에서 설명한 조건과 일치하는 경우 Amazon CloudWatch Events에서는 메시지를 전송하고 사용자가 지정한 대상(예: AWS Lambda 함수 또는 Amazon Simple Notification Service 주제)을 활성화합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		AWS::WAF::SizeConstraintSet 및 AWS::WAF::XssMatchSet AWS WAF 규칙 2개를 사용하여 웹 요청의 크기를 확인하거나 교차 사이트 스크립팅 공격을 방지합니다.	
신규 리소스	2016년 3월 31일	AWS::Lambda::Alias 리소스를 사용하여 AWS Lambda 함수의 별칭을 생성하고 AWS::Lambda::Version 리소스를 사용하여 함수 버전을 생성합니다.	2010-05-15
업데이트된 리소스	2016년 3월 31일	CloudFormation에서 다음 리소스가 업데이트되었습니다. AWS::EMR::Cluster 및 AWS::EMR::InstanceGroupConfig EbsConfiguration 속성을 사용하여 Amazon EMR 클러스터 또는 인스턴스 그룹에 대한 Amazon Elastic Block Store 스토리지 볼륨을 구성합니다. AWS::Lambda::Function VpcConfig 속성을 사용하여 AWS Lambda 함수가 VPC의 리소스에 액세스하도록 합니다. AWS::S3::Bucket Amazon Simple Storage Service 수명 주기 규칙의 경우 객체가 지정된 스토리지 클래스로 전환할 시점을 지정하는 전환 규칙을 여러 개 지정할 수 있습니다.	2010-05-15
변경 세트	2016년 3월 29일	스택을 업데이트하기 전에 변경 세트를 사용하여 변경 사항이 실행 중인 리소스에 미칠 수 있는 영향을 살펴봅니다. 자세한 내용은 변경 세트를 사용하여 스택 업데이트 를 참조하십시오.	2010-05-15
신규 리소스	2016년 3월 15일	AWS::GameLift::Alias , AWS::GameLift::Build 및 AWS::GameLift::Fleet 리소스를 사용하여 멀티플레이어 게임 서버를 AWS에 배포합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
신규 리소스	2016년 2월 26일	CloudFormation에서 다음 리소스가 추가되었습니다. AWS::ECR::Repository 사용자가 도커 이미지를 푸시하고 가져올 수 있는 Amazon Elastic Container Registry 리포지토리를 생성합니다. AWS::EC2::NatGateway 네트워크 주소 변환기(NAT) 게이트웨이를 사용하여 프라이빗 서브넷의 EC2 인스턴스를 인터넷에 연결할 수 있습니다. AWS::Elasticsearch::Domain 레거시 Elasticsearch OSS 클러스터를 실행하는 Amazon OpenSearch Service 도메인을 생성합니다. AWS::EMR::Cluster, AWS::EMR::InstanceGroupConfig, AWS::EMR::Step Amazon EMR 리소스를 사용하여 엄청난 양의 데이터를 분석 및 처리할 수 있습니다. 클러스터를 생성한 다음 해당 클러스터에서 작업을 실행할 수 있습니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
업데이트된 리소스	2016년 2월 26일	CloudFormation에서 다음 리소스가 업데이트되었습니다. AWS::CloudTrail::Trail IsMultiRegionTrail 속성을 사용하여 스택을 생성하는 리전 또는 모든 리전에서 AWS CloudTrail 추적을 생성할지 여부를 지정합니다. AWS::Config::ConfigurationRecorder 기록 그룹의 경우 IncludeGlobalResourceTypes 속성을 사용하여 전역 리소스 유형을 모두 기록합니다. AWS::RDS::DBCluster KmsKeyId 및 StorageEncrypted 속성을 사용하여 클러스터에서 데이터베이스 인스턴스를 암호화합니다.	2010-05-15
리소스 보존	2016년 2월 26일	DELETE_FAILED 상태 스택의 경우 RetainResources 파라미터를 사용하여 CloudFormation에서 삭제할 수 없는 리소스를 보존합니다. 자세한 내용은 스택 삭제 실패 단원을 참조하십시오.	2010-05-15
스택 태그 업데이트	2016년 2월 26일	스택을 업데이트할 때 스택 태그를 추가, 수정 또는 제거할 수 있습니다. 자세한 내용을 알아보려면 CloudFormation 스택 업데이트를 참조하세요.	2010-05-15
실패한 업데이트 롤백을 계속해서 롤백	2016년 1월 25일	UPDATE_ROLLBACK_FAILED 상태 스택의 경우 계속해서 업데이트를 롤백해 스택을 작업 상태로 되돌릴 수 있습니다. 이와 같이 스택을 원래 설정으로 되돌린 후 다시 업데이트를 시도할 수 있습니다. 자세한 내용은 업데이트 롤백 계속을 참조하십시오.	2010-05-15

변경	릴리스 날짜	설명	API 버전
아시아 태평양 (서울) 리전에 사용할 수 있는 새로운 샘플 템플릿	2016년 1월 7일	다음은 ap-northeast-2 리전을 위한 CloudFormation 샘플 템플릿 모음입니다. • 샘플 솔루션 • 애플리케이션 프레임워크 • 서비스 자세한 내용은 CloudFormation 템플릿 작업을 참조하세요.	2010-05-15
신규 리소스	2015년 12월 28일	CloudFormation에서 다음 리소스가 추가되었습니다. AWS::DirectoryService::MicrosoftAD Microsoft Active Directory 리소스를 사용하여 AWS에서 Microsoft Active Directory 디렉터리를 생성합니다. AWS::Logs::Destination 및 AWS::Logs::LogStream Amazon CloudWatch Logs 리소스를 사용하여 로그 데이터의 실시간 처리를 위한 대상을 생성하거나 로그 스트림을 각각 생성합니다. AWS::WAF::ByteMatchSet, AWS::WAF::IPSet, AWS::WAF::Rule, AWS::WAF::SqlInjectionMatchSet 및 AWS::WAF::WebACL AWS WAF 리소스를 사용하여 콘텐츠에 대한 웹 요청을 제어 및 모니터링합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
리소스 업데이트	2015년 12월 28일	CloudFormation에서 다음 리소스가 업데이트되었습니다. AWS::CloudFront::Distribution 배포 구성의 경우 WebACLIId 속성을 사용하여 AWS WAF 웹 액세스 제어 목록(ACL)을 Amazon CloudFront 분산과 연결합니다. 캐시 동작 및 기본 캐시 동작의 경우 기본 및 최대 TTL(Time-to-Live) 값을 지정할 수 있습니다. AWS::DynamoDB::Table Amazon DynamoDB 테이블을 바꾸지 않고 전역 보조 인덱스를 생성, 업데이트 또는 삭제할 수 있습니다. AWS::S3::Bucket ReplicationConfiguration 속성을 사용하여 복제할 객체와 해당 객체를 저장할 위치를 지정합니다. NotificationConfiguration 속성의 속성을 사용하여 사용자가 지정한 객체에 대한 알림을 Amazon Simple Storage Service에서 보내도록 필터를 지정합니다.	2010-05-15
파라미터 그룹화 및 정렬	2015년 12월 3일	스택을 생성하거나 템플릿으로 업데이트할 때 AWS::CloudFormation::Interface 메타데이터 키를 사용하여 CloudFormation 콘솔에서 파라미터를 그룹화하고 정렬합니다.	2010-05-15
업데이트 정책 속성	2015년 12월 3일	Auto Scaling 업데이트 정책 속성의 경우 MinSuccessfulInstancesPercent 속성을 사용하여 성공적인 업데이트에 대해 성공 신호를 보내야 하는 인스턴스의 비율을 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
신규 리소스	2015년 12월 3일	CloudFormation에서 다음 리소스가 추가되었습니다. AWS::CodePipeline::Pipeline 및 AWS::CodePipeline::CustomActionType CodePipeline 리소스를 사용하여 소프트웨어 변경 사항이 출시 프로세스에 적용되는 방법을 설명하는 파이프라인을 생성합니다. AWS::Config::ConfigurationRecorder, AWS::Config::DeliveryChannel 및 AWS::Config::ConfigRule AWS Config 리소스를 사용하여 특정 AWS 리소스에 대한 구성 변경 사항을 모니터링합니다. AWS::KMS::Key AWS Key Management Service(AWS KMS) 리소스를 사용하여 사용자가 소량의 데이터를 암호화하는 데 사용할 수 있는 고객 관리형 키를 AWS KMS에서 생성합니다. AWS::SSM::Document Amazon EC2 Systems Manager을 사용하여 온인스턴스 구성을 지정하는 문서를 생성합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
리소스 업데이트	2015년 12월 3일	CloudFormation에서 다음 리소스가 업데이트되었습니다. AWS::AutoScaling::LaunchConfiguration EBS 볼륨을 암호화할지 여부를 지정합니다. AWS::AutoScaling::ScalingPolicy 두 가지 다른 정책 유형(간단 및 단계 조정)을 사용하여 Amazon CloudWatch(CloudWatch) 경보 위반 시 Auto Scaling 그룹을 조정하는 방법을 지정할 수 있습니다. AWS::CloudTrail::Trail CloudWatch 속성을 사용하여 CloudWatch 로그 그룹에 로그를 전송합니다. 추적에 태그를 추가하고 로그 암호화에 사용할 AWS KMS key를 지정할 수 있습니다. AWS::CodeDeploy::Application, AWS::CodeDeploy::DeploymentConfig 및 AWS::CodeDeploy::DeploymentGroup ApplicationName , DeploymentConfigName 및 DeploymentGroupName 속성을 사용하여 CodeDeploy 리소스에 대한 사용자 지정 이름을 지정합니다. AWS::DynamoDB::Table StreamSpecification 속성을 사용하여 Amazon DynamoDB(DynamoDB) 테이블에 저장된 항목에 대한 변경 사항의 캡처를 위한 설정을 지정합니다. AWS::EC2::Instance SsmAssociations 속성을 사용하여 Amazon EC2 Systems Manager 문서를 인스턴스와 연결합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		AWS::EC2::SpotFleet AllocationStrategy 속성을 사용하여 스팟 풀 전체에서 목표 용량을 할당하는 방법을 지정합니다. ExcessCapacityTerminationPolicy 속성을 사용하여 목표 용량이 스팟 플릿의 크기 미만인 경우 인스턴스를 종료하는 방법을 지정합니다. AWS::Redshift::Cluster KmsKeyId 속성을 사용하여 Amazon Redshift 클러스터에서 데이터를 암호화하기 위한 AWS KMS key를 지정합니다. AWS::WorkSpaces::Workspace 암호화 속성을 사용하여 볼륨에 저장된 데이터를 암호화합니다.	
리소스 업데이트	2015년 11월 4일	AWS::EC2::Volume 리소스의 경우 AutoEnableIO 속성을 사용하여 볼륨의 데이터가 일치하지 않는 경우 I/O 작업을 자동으로 재개합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
신규 리소스	2015년 10월 1일	CloudFormation에서 다음 리소스가 추가되었습니다. AWS::CodeDeploy::Application, AWS::CodeDeploy::DeploymentGroup 및 AWS::CodeDeploy::DeploymentConfig CodeDeploy 리소스를 사용하여 배포를 생성하여 EC2 또는 온프레미스 인스턴스에 적용합니다. AWS::DirectoryService::SimpleAD Simple Active Directory 리소스를 사용하여 Microsoft Active Directory와 호환되는 디렉터리인 Directory Service Simple AD를 생성합니다. AWS::EC2::PlacementGroup 배치 그룹을 사용하여 지연 시간이 짧은 네트워크에서 인스턴스 클러스터를 생성합니다. AWS::EC2::SpotFleet 스팟 플릿을 사용하여 중단 가능한 작업을 실행하는 스팟 인스턴스 모음을 시작합니다. AWS::Lambda::EventSourceMapping 이벤트 소스 매핑 리소스를 사용하여 스트림을 AWS Lambda(Lambda) 함수에 대한 이벤트 소스로 지정합니다. AWS::Lambda::Permission Lambda 권한을 사용하여 Lambda 함수의 정책에 문을 추가합니다. AWS::Logs::SubscriptionFilter 구독 필터를 사용하여 Kinesis 스트림에 전달되는 로그 이벤트를 정의합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		AWS::RDS::DBCluster 및 AWS::RDS::DBClusterParameterGroup 클러스터 및 클러스터 파라미터 그룹 리소스를 사용하여 Amazon Aurora DB 클러스터를 생성합니다. AWS::WorkSpaces::Workspace WorkSpaces를 사용하여 클라우드 기반 데스크톱 환경을 생성합니다.	
리소스 업데이트	2015년 10월 1일	CloudFormation에서 다음 리소스가 업데이트되었습니다. AWS::ElastiCache::ReplicationGroup <code>Fn::GetAtt</code> 내장 함수를 사용하여 읽기 전용 복제본 주소 및 포트 목록을 가져옵니다. AWS::OpsWorks::Stack <code>AgentVersion</code> 속성을 사용하여 특정 OpsWorks 에이전트를 지정합니다. AWS::OpsWorks::App <code>Environment</code> 속성을 사용하여 OpsWorks 앱에 대한 환경 변수를 지정합니다. AWS::S3::Bucket NotificationConfiguration 속성의 경우 Lambda 함수 및 Amazon Simple Queue Service(Amazon SQS) 대기열에 대한 알림 설정을 구성할 수 있습니다.	2010-05-15
IAM 조건 키	2015년 10월 1일	AWS Identity and Access Management(IAM) 정책의 경우 CloudFormation 전용 조건 키를 사용하여 IAM 정책이 적용되는 시점을 지정합니다. 자세한 내용을 알아보려면 AWS Identity and Access Management를 사용한 액세스 제어를 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
CloudFormation Designer	2015년 10월 1일	CloudFormation Designer 를 사용해 끌어 놓기 인터페이스를 사용하여 템플릿을 생성 및 수정합니다.	2010-05-15
신규 리소스	2015년 8월 24일	AWS::EC2::VPCEndpoint 리소스를 사용하여 사용자의 VPC와 다른 AWS 서비스 간에 프라이빗 연결을 설정합니다.	2010-05-15
리소스 업데이트	2015년 8월 24일	CloudFormation에서 다음 리소스가 업데이트되었습니다. AWS::ElasticBeanstalk::Environment Tags 속성을 사용하여 AWS Elastic Beanstalk(Elastic Beanstalk) 환경에 대한 태그(키-값 페어)를 지정합니다. AWS::Lambda::Function Code 속성의 경우 ZipFile 속성을 사용하여 템플릿에서 직접 Lambda 함수의 소스 코드를 작성합니다. 현재, ZipFile 실행 시간 환경에는 nodejs 속성만 사용할 수 있습니다. 모든 실행 시간 환경에 대해 S3 버킷의 파일(예: java8 및 nodejs)을 가리킬 수 있습니다. AWS::OpsWorks::Instance EbsOptimized 속성을 사용하여 Amazon Elastic Block Store(Amazon EBS) I/O에 대해 인스턴스를 최적화할지 여부를 나타냅니다. AWS::RDS::DBInstance SourceDBInstanceIdentifier 속성의 경우 교차 리전 읽기 전용 복제본을 생성하기 위해 또 다른 리전에 데이터베이스 인스턴스를 지정할 수 있습니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
Amazon S3 템플릿 URL	2015년 8월 24일	버전 관리를 사용하는 버킷의 경우 스택을 생성 또는 업데이트할 때 Amazon S3 템플릿 URL에 버전 ID를 지정할 수 있습니다(예: https://s3.amazonaws.com/templates/myTemplate.template?versionId=123ab1cdeKd0W5IH4GAcYbEngcpTJTDW).	2010-05-15
신규 리소스	2015년 8월 3일	AWS::EFS::FileSystem 리소스를 사용하여 Amazon Elastic File System(Amazon EFS) 파일 시스템을 생성하고 AWS::EFS::MountTarget 리소스를 사용하여 파일 시스템의 탑재 지점을 생성합니다.	2010-05-15
권한 요구 사항 변경	2015년 6월 11일	AWS::RDS::DBInstance 리소스를 생성하거나 업데이트하는 경우 이제 <code>ec2:DescribeAccountAttributes</code> 작업을 직접적으로 호출하는 권한도 있어야 합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
신규 리소스	2015년 6월 11일	CloudFormation에서 다음 리소스가 추가되었습니다. AWS::DataPipeline::Pipeline 데이터 파이프라인을 사용하여 데이터의 이동과 변환을 자동화합니다. Amazon Elastic Container Service 리소스 AWS::ECS::Service, AWS::ECS::Cluster 및 AWS::ECS::TaskDefinition 리소스를 사용하여 EC2 인스턴스 클러스터에서 Docker 컨테이너를 생성합니다. AWS::ElastiCache::ReplicationGroup 복제 그룹을 사용하여 기본 읽기-쓰기 클러스터 하나와 보조 읽기 전용 클러스터가 최대 5개 포함된 노드 모음을 생성합니다. AWS::IAM::ManagedPolicy 관리형 정책을 사용하여 IAM 사용자, 그룹 및 역할에 권한을 적용하는 데 사용할 수 있는 정책을 AWS 계정에서 생성합니다. AWS::Lambda::Function Lambda 함수를 사용하여 이벤트에 대한 응답으로 코드를 실행합니다. AWS::RDS::OptionGroup 옵션 그룹을 사용하여 Amazon Relational Database Service(Amazon RDS) 데이터베이스를 생성 및 관리할 수 있습니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
리소스 업데이트	2015년 6월 11일	CloudFormation에서 다음 리소스가 업데이트되었습니다. AWS::EC2::Subnet MapPublicIpOnLaunch 속성을 사용하여 서브넷의 인스턴스로 퍼블릭 IP 주소를 자동으로 할당합니다. AWS::ElastiCache::CacheCluster SnapshotName 속성을 사용하여 스냅샷 데이터를 새 Redis 캐시 클러스터로 복원합니다. AWS::IAM::User LoginProfile 속성의 경우 PasswordResetRequired 속성을 사용합니다. 따라서 사용자가 AWS Management Console에 로그인할 때 새 암호를 설정해야 합니다. AWS::OpsWorks::Layer LifecycleEventConfiguration 속성을 사용하여 OpsWorks 계층에 대한 수명 주기 이벤트를 구성합니다. AWS::S3::Bucket LifecycleConfiguration 속성의 경우 NoncurrentVersionExpirationInDays 및 NoncurrentVersionTransition 속성을 사용하여 최신이 아닌 객체 버전에 대한 수명 주기 규칙을 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
새로운 파라미터 유형	2015년 5월 19일	CloudFormation 콘솔을 사용하여 스택을 생성 또는 업데이트할 때마다 ID, 이름 또는 이름 태그 값별로 AWS 특정 파라미터 유형 값을 검색할 수 있습니다. 또한 CloudFormation에서 다음 AWS 특정 파라미터 유형에 대한 지원이 추가되었습니다. 자세한 내용은 파라미터를 참조하세요. • AWS::EC2::AvailabilityZone::Name • List<AWS::EC2::AvailabilityZone ::Name> • AWS::EC2::Instance::Id • List<AWS::EC2::Instance ::Id> • AWS::EC2::Image::Id • List<AWS::EC2::Image::Id > • AWS::EC2::SecurityGroup::GroupName • List<AWS::EC2::SecurityGroup::Gr oupName > • AWS::EC2::Volume::Id • List<AWS::EC2::Volume::Id > • AWS::Route53::HostedZone::Id • List<AWS::Route53::HostedZone ::Id>	2010-05-15
신규 리소스	2015년 16월 4일	CloudFormation에서 다음 리소스가 추가되었습니다. AWS::AutoScaling::LifecycleHook Auto Scaling 수명 주기 후크를 사용하여 인스턴스가 시작되거나 종료된 이후에 인스턴스 상태를 제어할 수 있습니다. AWS::RDS::EventSubscription 이벤트 구독을 사용하여 Amazon RDS 이벤트에 대한 알림을 얻습니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
리소스 업데이트	2015년 16월 4일	CloudFormation에서 다음 리소스가 업데이트되었습니다. AWS::AutoScaling::AutoScalingGroup NotificationConfigurations 속성을 사용하여 여러 알림을 지정합니다. AWS::AutoScaling::LaunchConfiguration PlacementTenancy 속성을 사용하여 인스턴스 테넌트를 지정합니다. ClassicLinkVPCId 및 ClassicLinkVPCSecurityGroups 속성을 사용하여 EC2-Classic 인스턴스를 ClassicLink 가능 VPC에 연결합니다. AWS::AutoScaling::ScalingPolicy MinAdjustmentStep 속성을 사용하여 조정 이벤트 중 추가 또는 제거되는 인스턴스의 최소 개수를 지정합니다. AWS::CloudFront::Distribution 최종 사용자 인증서의 경우 MinimumProtocolVersion 속성을 사용하여 최소 프로토콜 버전을 지정합니다. 캐시 동작의 경우 CachedMethods 속성을 사용하여 Amazon CloudFront(CloudFront)에서 응답을 캐시할 방법을 지정합니다. 오리진의 경우 OriginPath 를 사용하여 CloudFront에서 콘텐츠를 요청하는 데 사용하는 경로를 지정합니다. AWS::ElastiCache::CacheCluster Memcached 캐시 클러스터의 경우 AZMode 및 PreferredAvailabilityZones 속성을 사용하여 여러 가용 영역(AZ)에서 노드를 지정합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
		AWS::EC2::Volume KmsKeyId 속성을 사용하여 암호화된 볼륨에 대한 고객 관리형 키를 지정합니다. AWS::OpsWorks::Instance TimeBasedAutoScaling 속성을 사용하여 지정된 일정에 따라 인스턴스를 자동으로 조정합니다. AWS::OpsWorks::Layer LoadBasedAutoScaling 속성을 사용하여 로드 기반 조정 정책을 지정합니다. 볼륨 구성의 경우 VolumeType 및 Iops 속성을 사용하여 볼륨 유형 및 초당 I/O 작업 수를 각각 지정합니다. AWS::RDS::DBInstance CharacterSetName 속성을 사용하여 지원되는 데이터베이스 엔진에 대한 문자 세트를 지정합니다. StorageEncrypted 속성을 사용하여 데이터베이스 인스턴스를 암호화할지 여부를 지정하고, KmsKeyId 코드를 사용하여 암호화된 데이터베이스 인스턴스에 대한 고객 관리형 키를 지정합니다. AWS::Route53::HealthCheck HealthCheckTags 속성을 사용하여 태그를 상태 확인과 연결합니다. AWS::Route53::HostedZone VPCs 속성을 사용하여 프라이빗 호스팅 영역을 생성합니다. HostedZoneTags 속성을 사용하여 태그를 호스팅 영역과 연결합니다.	

변경	릴리스 날짜	설명	API 버전
새 템플릿 섹션	2015년 16월 4일	템플릿을 설명하는(예: 설계 및 구현 세부 정보) 임의 JSON 객체를 포함하기 위해 템플릿에 메타데이터 섹션이 추가됩니다.	2010-05-15
리소스 업데이트	2015년 4월 8일	AWS::CloudFormation::CustomResource 리소스의 경우 ServiceToken 속성에서 Lambda 함수 Amazon 리소스 이름(ARN)을 지정할 수 있습니다.	2010-05-15
Amazon RDS update	2014년 12월 24일	CloudFormation에서 RDS DB 인스턴스에 대한 새 속성이 2개 추가되었습니다. DB 인스턴스와 옵션 그룹을 연결하고 DB 인스턴스 스토리지 유형을 지정할 수 있습니다. 자세한 내용은 AWS::RDS::DBInstance 섹션을 참조하세요.	2010-05-15
Elastic Load Balancing update	2014년 12월 24일	ConnectionSettings 속성을 사용하여 연결이 유효 상태로 남아 있을 수 있는 기간을 지정합니다. 자세한 내용은 AWS::ElasticLoadBalancing::LoadBalancer 섹션을 참조하세요.	2010-05-15
Route 53 업데이트	2014년 11월 6일	이제 Route 53 호스팅 영역 , 상태 확인 , 장애 조치 레코드 세트 및 지리적 위치 레코드 세트 를 프로비저닝하고 관리할 수 있습니다.	2010-05-15
Auto Scaling 롤링 업데이트 개선 사항	2014년 11월 6일	업데이트 중 WaitOnResourceSignals 플래그를 사용하여 CloudFormation에 인스턴스가 성공 신호를 보낼 때까지 대기하도록 지시할 수 있습니다. 이러한 방식으로 CloudFormation에서는 현재 배치가 준비될 때까지 인스턴스의 다음 배치를 업데이트하지 않습니다. 자세한 내용은 UpdatePolicy 속성 을 참조하십시오.	2010-05-15
새 VPC Fn::GetAtt 속성	2014년 11월 6일	VPC ID로 해당 VPC의 기본 보안 그룹 및 네트워크 ACL을 검색할 수 있습니다. 자세한 내용은 Fn::GetAtt 섹션을 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
새로운 AWS 고유 파라미터 유형	2014년 11월 6일	CloudFormation 템플릿에서 AWS 특정 파라미터 유형을 지정할 수 있습니다. CloudFormation 콘솔에서 이러한 파라미터 유형은 유효한 값의 드롭다운 목록을 제공합니다. API 또는 AWS CLI를 사용하여 CloudFormation에서는 스택을 생성 또는 업데이트하기 전에 이러한 파라미터 유형에 대한 값을 신속하게 확인할 수 있습니다. 자세한 내용은 파라미터 를 참조하십시오.	2010-05-15
CreationPolicy 속성	2014년 11월 6일	CreationPolicy 속성을 사용하여 CloudFormation에 스택 생성을 진행하기 전에 EC2 인스턴스에서 애플리케이션이 준비될 때까지 대기하도록 지시할 수 있습니다. 대기 조건 및 대기 조건 핸들 대신 생성 정책을 사용할 수 있습니다. 자세한 내용은 CreationPolicy 속성 을 참조하십시오.	2010-05-15
Amazon CloudFront에서 전달된 값	2014년 9월 29일	캐시 동작의 경우 오리진에 헤더를 전달할 수 있습니다. ForwardedValues 를 참조하십시오.	2010-05-15
AWS OpsWorks update	2014년 9월 29일	Chef 11.10의 경우 ChefConfiguration 속성을 사용해 Berkshelf를 활성화할 수 있습니다. 또한 AWS OpsWorks 스택과 함께 AWS OpsWorks 기본 제공 보안 그룹을 사용할 수 있습니다. 자세한 내용은 AWS::OpsWorks::Stack 섹션을 참조하세요.	2010-05-15
Elastic Load Balancing 태그 지정 지원	2014년 9월 29일	AWS CloudFormation에서는 스택 수준 태그를 사용하여 Elastic Load Balancing 로드 밸런서에 태그를 지정합니다. 또한 로드 밸런서에 고유한 태그를 추가할 수 있습니다. AWS::ElasticLoadBalancing::LoadBalancer 을(를) 참조하세요.	2010-05-15
Amazon Simple Notification Service 주제 정책 업데이트	2014년 9월 29일	이제 Amazon SNS 주제 정책을 업데이트할 수 있습니다. 자세한 내용은 AWS::SNS::TopicPolicy 섹션을 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
RDS DB 인스턴스 업데이트	2014년 9월 5일	AWS::RDS::DBInstance 리소스의 PubliclyAccessible 속성을 사용하여 DB 인스턴스가 인터넷 경계인지 지정할 수 있습니다.	2010-05-15
UpdatePolicy 속성 업데이트	2014년 9월 05일	연결된 예약된 작업이 있는 Auto Scaling 그룹에 대해 업데이트 정책을 지정할 수 있습니다. 자세한 내용은 UpdatePolicy 속성 을 참조하십시오.	2010-05-15
Amazon CloudWatch 지원	2014년 7월 10일	CloudFormation을 사용하여 Amazon CloudWatch Logs(CloudWatch Logs) 로그 그룹 및 지표 필터를 프로 비저닝 및 관리할 수 있습니다. 자세한 내용은 AWS::Logs::LogGroup 또는 AWS::Logs::MetricFilter 섹션을 참조하 세요.	2010-05-15
Amazon CloudFront 배 포 구성 업데이 트	2014년 6월 17일	추가 CloudFront 배포 구성 속성을 지정할 수 있습니다. - 사용자 지정 오류 응답은 4xx 및 5xx HTTP 상태 코드에 대한 사용자 지정 오류 메시지를 정의합니다. - 가격 등급은 CloudFront 서비스를 위해 지불하려는 최 고 가격을 정의합니다. - 제한은 콘텐츠를 볼 수 있는 사람을 정의합니다. - 최종 사용자 인증서는 최종 사용자가 HTTPS를 사용할 때 필요한 인증서를 지정합니다. - 캐시 동작의 경우 허용되는 HTTP 방법을 지정하고 쿠키를 전달할지 여부를 지정할 수 있습니다. 자세한 내용은 AWS::CloudFront::Distribution 섹션을 참 조하세요.	2010-05-15
EC2 인스턴스 업데이트	2014년 6월 17일	인스턴스의 운영 체제 종료 명령을 호출하면 인스턴스를 중지 또는 종료할지 여부를 지정할 수 있습니다. 자세한 내용은 AWS::EC2::Instance 섹션을 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
EBS 볼륨 업데이트	2014년 6월 17일	지원되는 인스턴스 유형과 함께 암호화된 EBS 볼륨을 사용할 수 있습니다. 자세한 내용은 AWS::EC2::Volume 섹션을 참조하세요.	2010-05-15
새로운 Amazon VPC 피어링 연결	2014년 6월 17일	CloudFormation을 사용하여 두 VPC 간에 네트워크 연결을 설정하는 Amazon Virtual Private Cloud(Amazon VPC) 피어링 연결을 생성할 수 있습니다. 자세한 내용은 AWS::EC2::VPCPeeringConnection 섹션을 참조하세요.	2010-05-15
Amazon EC2 Auto Scaling 그룹 업데이트	2014년 6월 17일	Amazon EC2 Auto Scaling 그룹에 대한 인스턴스를 시작할 기존 클러스터 배치 그룹을 지정할 수 있습니다. 자세한 내용은 AWS::AutoScaling::AutoScalingGroup 섹션을 참조하세요.	2010-05-15
AWS CloudTrail 지원	2014년 6월 17일	CloudFormation에서는 AWS 계정에서 실행한 API 호출을 캡처하고 사용자가 지정한 위치에서 로그를 게시하는 AWS CloudTrail을 지원합니다. 자세한 내용은 AWS::CloudTrail::Trail 섹션을 참조하세요.	2010-05-15
스택 업데이트 개선 사항	2014년 5월 12일	CloudFormation에서는 스택 업데이트를 위한 다음 추가 기능을 지원합니다. - 스택의 템플릿을 다시 제출하지 않고 CloudFormation 스택 템플릿을 업데이트할 수 있습니다. - CloudFormation 스택에 대한 Amazon SNS 알림 주제를 추가 또는 제거할 수 있습니다. 자세한 내용을 알아보려면 CloudFormation 스택 업데이트를 참조하세요.	2010-05-15
Amazon Kinesis 지원	2014년 5월 6일	CloudFormation을 사용하여 데이터 소스에서 데이터 레코드를 캡처 및 전송하는 Amazon Kinesis 스트림을 생성할 수 있습니다. 자세한 내용은 AWS::Kinesis::Stream 섹션을 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
새 S3 버킷 속성	2014년 5월 5일	CloudFormation에서는 다음 추가 S3 버킷 속성을 지원합니다. - CORS(Cross-origin 리소스 공유)는 버킷에서 객체의 Cross-origin 리소스 공유를 정의합니다. - 수명 주기는 객체의 수명 주기 동안 Amazon S3가 해당 객체를 관리하는 방법을 정의합니다. - 버킷에 대해 실행된 요청에 대한 로깅 정책 캡처 정보에 액세스합니다. - 알림은 보고할 이벤트와 메시지를 전송할 Amazon SNS 주제를 정의합니다. - 버전 관리는 버킷의 모든 객체에 대해 여러 변형을 지원합니다. - 리디렉션 규칙 및 라우팅 규칙은 버킷의 웹사이트 엔드 포인트에 대해 실행된 요청의 리디렉션 동작을 관리합니다. 자세한 내용은 AWS::S3::Bucket 섹션을 참조하세요.	2010-05-15
Amazon EC2 Auto Scaling 지원	2014년 5월 5일	CloudFormation에서는 Auto Scaling 그룹에 대한 지표 수집을 지원합니다. 자세한 내용은 AWS::AutoScaling::AutoScalingGroup 섹션을 참조하세요.	2010-05-15
Fn::If update	2014년 5월 5일	템플릿의 출력 섹션에서 Fn::If 내장 함수를 사용할 수 있습니다. 자세한 내용은 조건 함수를 참조하십시오.	2010-05-15
AWS CloudTrail을 사용한 API 로깅	2014년 4월 2일	AWS CloudTrail(CloudTrail)을 사용하여 CloudFormation 요청을 로깅할 수 있습니다. CloudTrail을 사용하여 계정에 대한 CloudFormation API 호출 기록을 가져올 수 있습니다. 자세한 내용은 AWS CloudTrail을 사용하여 CloudFormation API 호출 로깅을 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
Elastic Load Balancing update	2014년 3월 20일	로드 밸런서에 대해 실행된 요청에 대한 정보를 캡처하기 위한 액세스 로깅 정책을 지정할 수 있습니다. 또한 인스턴스가 등록 취소되거나 비정상 상태가 될 때 인플라이트 요청을 처리하는 방법을 설명하는 Connection Draining 정책을 지정할 수도 있습니다. 자세한 내용은 AWS::ElasticLoadBalancing::LoadBalancer 섹션을 참조하세요.	2010-05-15
OpsWorks 지원	2014년 3월 3일	CloudFormation을 사용하여 OpsWorks 스택을 프로비저닝 및 관리할 수 있습니다. 자세한 내용은 AWS::OpsWorks::Stack 또는 AWS OpsWorks Template Snippets 를 참조하세요.	2010-05-15
Amazon S3 템플릿 크기 제한 증가	2014년 2월 18일	Amazon S3에서 템플릿 크기를 최대 460,800바이트까지 지정할 수 있습니다.	2010-05-15
Amazon Redshift 지원	2014년 2월 10일	CloudFormation을 사용하여 Amazon Redshift 클러스터를 프로비저닝 및 관리할 수 있습니다. 자세한 내용은 Amazon Redshift Template Snippets 또는 AWS::Redshift::Cluster 를 참조하세요.	2010-05-15
S3 버킷 및 버킷 정책 업데이트	2014년 2월 10일	S3 버킷 및 버킷 정책 리소스의 일부 속성을 업데이트할 수 있습니다. 자세한 내용은 AWS::S3::Bucket 또는 AWS::S3::BucketPolicy 섹션을 참조하세요.	2010-05-15
Elastic Beanstalk 환경 및 애플리케이션 버전 업데이트	2014년 2월 10일	Elastic Beanstalk 환경 구성 및 애플리케이션 버전을 업데이트할 수 있습니다. 자세한 내용은 AWS::ElasticBeanstalk::Environment , AWS::ElasticBeanstalk::ConfigurationTemplate 또는 AWS::ElasticBeanstalk::ApplicationVersion 을 참조하세요.	2010-05-15
Amazon SQS update	2014년 1월 29일	Amazon SQS 대기열에 대한 배달 못한 편지 대기열을 지정할 수 있습니다. 자세한 내용은 AWS::SQS::Queue 섹션을 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
Auto Scaling 예약된 작업	2014년 1월 27일	일정에 따라 Auto Scaling 그룹의 EC2 인스턴스 수를 조정할 수 있습니다. 일정을 사용하여 예측 가능한 로드 변경에 맞게 애플리케이션을 조정할 수 있습니다. 자세한 내용은 AWS::AutoScaling::ScheduledAction 섹션을 참조하세요.	2010-05-15
DynamoDB 보조 인덱스	2014년 1월 27일	DynamoDB 데이터베이스에 대한 로컬 및 전역 보조 인덱스를 생성할 수 있습니다. 보조 인덱스를 사용하면 기본 키 이외에 속성을 사용하여 데이터에 효율적으로 액세스할 수 있습니다. 자세한 내용은 AWS::DynamoDB::Table 섹션을 참조하세요.	2010-05-15
Auto Scaling update	2014년 1월 2일	Auto Scaling 그룹에 대한 인스턴스 ID 또는 시작 구성을 지정할 수 있습니다. 또한 추가 Auto Scaling 블록 디바이스 속성을 지정할 수도 있습니다. 자세한 내용은 AWS::AutoScaling::AutoScalingGroup 또는 AWS::AutoScaling::LaunchConfiguration 섹션을 참조하세요.	2010-05-15
Amazon SQS update	2014년 1월 2일	SQS 대기열을 업데이트하고 추가 속성을 지정할 수 있습니다. 자세한 내용은 AWS::SQS::Queue 섹션을 참조하세요.	2010-05-15
제한 증가	2014년 1월 2일	CloudFormation 템플릿에서 파라미터와 출력을 각각 최대 60개까지 지정할 수 있습니다.	2010-05-15
새 콘솔	2013년 12월 19일	새로운 CloudFormation 콘솔은 스택 이벤트 자동 새로 고침 및 스택 파라미터의 알파벳순 정렬과 같은 기능을 추가합니다.	2010-05-15
교차 영역 로드 밸런싱	2013년 12월 19일	교차 영역 로드 밸런싱을 사용하면 모든 가용 영역(AZ)에 걸쳐 있는 백엔드 인스턴스로 트래픽을 라우팅할 수 있습니다. 자세한 내용은 AWS::ElasticLoadBalancing::LoadBalancer 섹션을 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
AWS Elastic Beanstalk 환경 티어	2013년 12월 19일	웹 서버를 지원하거나 백그라운드에서 처리 중인 작업을 처리하기 위해 AWS Elastic Beanstalk에서 리소스를 프로비저닝할지 여부를 지정할 수 있습니다. 자세한 내용은 AWS::ElasticBeanstalk::Environment 섹션을 참조하세요.	2010-05-15
리소스 이름	2013년 12월 19일	다음 리소스에 이름(물리적 ID)을 할당할 수 있습니다. - ElastiCache 클러스터 - Elastic Load Balancing 로드 밸런서 - RDS DB 인스턴스 자세한 내용은 이름 유형을 참조하십시오.	2010-05-15
VPN 지원	2013년 11월 22일	가상 프라이빗 게이트웨이(VGW)를 사용하여 VPC의 라우팅 테이블로 경로를 전파할 수 있습니다. 자세한 내용은 AWS::EC2::VPNGatewayRoutePropagation 섹션을 참조하세요.	2010-05-15
조건부로 리소스 생성 및 속성 할당	2013년 11월 8일	입력 파라미터를 사용하여 CloudFormation 템플릿에서 조건을 정의해 지정된 스택 리소스의 생성 및 설정을 제어할 수 있습니다. 예를 들어, 조건을 사용하여 프로덕션 환경에 대한 스택 리소스를 생성할 수 있습니다. 동일한 템플릿을 사용하여 테스트 환경의 축소된 용량으로 유사한 스택 리소스를 생성할 수 있습니다. 자세한 내용은 조건 함수 를 참조하십시오.	2010-05-15
스택 리소스에 대한 뜻하지 않은 업데이트 방지	2013년 11월 8일	스택 리소스에 대한 의도치 않은 변경으로 이어질 수 있는 스택 업데이트를 방지할 수 있습니다. 예를 들어 좀처럼 업데이트되면 안 되는 데이터베이스 계층이 포함된 스택이 있는 경우 대부분의 사용자가 해당 데이터베이스 계층을 업데이트하지 않도록 방지하는 스택 정책을 설정할 수 있습니다. 자세한 내용은 스택 리소스에 대한 업데이트 방지 를 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
리소스에 이름 지정	2013년 11월 8일	CloudFormation에서 생성된 물리적 ID를 사용하는 대신 특정 리소스에 이름을 할당할 수 있습니다. 다음 CloudFormation 리소스에서 이름 지정 지원 • Amazon CloudWatch 경보 • DynamoDB 테이블 • AWS Elastic Beanstalk 애플리케이션 및 환경 • Amazon S3 버킷 • Amazon SNS 주제 • Amazon SQS 대기열 자세한 내용은 이름 유형을 참조하십시오.	2010-05-15
사용자 지정 리소스 유형 할당	2013년 11월 8일	템플릿에서 CloudFormation 사용자 지정 리소스 (AWS::CloudFormation::CustomResource)에 대해 고유한 리소스 유형을 지정할 수 있습니다. 고유한 사용자 지정 리소스 유형 이름을 사용하면 스택에 있는 사용자 지정 리소스 유형을 신속하게 식별할 수 있습니다. 예를 들어, "Type": "Custom:: <i>MyCustomResource</i> "를 지정할 수 있습니다. 자세한 내용은 AWS::CloudFormation::CustomResource 섹션을 참조하십시오.	2010-05-15
가상 파라미터 추가	2013년 11월 8일	이제 AWS::AccountId 가상 파라미터를 참조해 CloudFormation 템플릿 내에서 AWS AccountID를 참조할 수 있습니다. 자세한 내용은 가상 파라미터 참조를 참조하십시오.	2010-05-15

변경	릴리스 날짜	설명	API 버전
IAM 정책에서 스택 지정	2013년 11월 8일	IAM 사용자, 그룹 또는 역할이 특정 CloudFormation 스택에서 작업하도록 허용하거나 거부할 수 있습니다. 예를 들어 특정 스택 ID에 대한 스택 삭제 작업을 거부할 수 있습니다. 자세한 내용을 알아보려면 AWS Identity and Access Management를 사용한 액세스 제어 를 참조하세요.	2010-05-15
연동 지원	2013년 10월 14일	CloudFormation에서는 IAM 역할의 임시 보안 자격 증명을 지원합니다. 따라서 AWS Management Console에 대한 페더레이션 및 Single Sign-On과 같은 시나리오가 가능합니다. 또한 IAM 역할을 사용하여 장기 보안 자격 증명을 임베딩하지 않아도 EC2 인스턴스에서 CloudFormation에 대한 호출을 실행할 수 있습니다. IAM 및 CloudFormation에 대한 자세한 내용은 AWS ID 및 액세스 관리를 사용하여 액세스 제어 를 참조하세요.	2010-05-15
Amazon RDS 읽기 전용 복제본 지원	2013년 9월 24일	이제 원본 DB 인스턴스 하나에서 Amazon RDS 읽기 전용 복제본을 생성할 수 있습니다. 자세한 내용은 AWS::RDS::DBInstance 리소스의 SourceDBInstanceId 속성을 참조하세요.	2010-05-15
Auto Scaling 그룹의 인스턴스와 퍼블릭 IP 주소 연결	2013년 9월 19일	이제 Auto Scaling 그룹의 인스턴스와 퍼블릭 IP 주소를 연결할 수 있습니다. 자세한 내용은 AWS::AutoScaling::LaunchConfiguration 섹션을 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
추가 VPC 지원	2013년 9월 17일	CloudFormation에서 VPC 및 VPN 기능을 지원하기 위해 여러 가지 개선 사항 추가 - Amazon EC2 네트워크 인터페이스에 퍼블릭 IP 주소 하나와 프라이빗 IP 주소 여러 개를 연결할 수 있습니다. 자세한 내용은 AWS::EC2::NetworkInterface 섹션을 참조하세요. 또한 탄력적 IP 주소(EIP)에 기본 프라이빗 IP 주소를 연결할 수도 있습니다. - DNS 지원을 활성화하고 DNS 호스트 이름을 지정할 수 있습니다. 자세한 내용은 AWS::EC2::VPC 섹션을 참조하세요. - 가상 프라이빗 게이트웨이와 VPN 게이트웨이 간에 정적 경로를 지정할 수 있습니다. 자세한 내용은 AWS::EC2::VPNConnectionRoute 섹션을 참조하세요.	2010-05-15
Amazon ElastiCache에 대한 Redis 및 VPC 보안 그룹 지원	2013년 9월 3일	이제 Redis를 Amazon ElastiCache(ElastiCache) 클러스터의 캐시 엔진으로 지정할 수 있습니다. 뿐만 아니라 이제 VPC 보안 그룹을 ElastiCache 클러스터에 할당할 수 있습니다. 자세한 내용은 AWS::ElastiCache::CacheCluster 섹션을 참조하세요.	2010-05-15
병렬 스택 생성, 업데이트 및 삭제, 중첩된 스택 업데이트	2013년 8월 12일	이제 AWS CloudFormation에서 리소스를 동시에 생성, 업데이트 및 삭제할 수 있어 작업 성능이 개선되었습니다. 최상위 템플릿을 업데이트하는 경우 AWS CloudFormation에서는 변경된 중첩 스택을 자동으로 업데이트합니다. 자세한 내용을 알아보려면 CloudFormation 스택 업데이트 를 참조하세요.	2010-05-15
RDS DB 인스턴스에서 VPC 보안 그룹 설정 가능	2013년 2월 28일	이제 CloudFormation을 사용하여 VPC 보안 그룹을 RDS DB 인스턴스에 할당할 수 있습니다. 자세한 내용은 AWS::RDS::DBInstance 의 VPCSecurityGroups 속성을 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
Amazon EC2 Auto Scaling 그룹에 대한 배포 롤링	2013년 2월 20일	이제 CloudFormation에서 Amazon EC2 Auto Scaling 그룹에서 인스턴스를 추가 또는 제거할 때 Amazon EC2 Auto Scaling 그룹의 인스턴스를 바꾸거나 수정하는 방법을 설명하는 업데이트 정책을 Amazon EC2 Auto Scaling 그룹에 대해 사용할 수 있습니다. 스택 생성 시 또는 스택 업데이트 도중 이러한 설정을 수정할 수 있습니다. 자세한 내용과 예제는 UpdatePolicy 속성을 참조하십시오.	2010-05-15
스택 업데이트에 대한 취소 및 롤백 작업	2013년 2월 20일	CloudFormation에서는 스택 업데이트를 취소할 수 있는 기능을 지원합니다. 업데이트 요청이 이루어진 경우 스택이 UPDATE_IN_PROGRESS 상태여야 합니다. 자세한 내용은 다음 주제에서 확인할 수 있습니다. • 스택 업데이트 취소 • cancel-update-stack • CloudFormation API 참조의 CancelUpdateStack	2010-05-15
Amazon EC2 Auto Scaling 그룹에 대한 EBS 최적화 인스턴스	2013년 2월 20일	이제 자동 조정된 인스턴스에서 Amazon Elastic Block Store(Amazon EBS) 전용 처리량을 위해 Amazon EC2 Auto Scaling 그룹의 EBS 최적화 인스턴스를 프로비저닝할 수 있습니다. 최적화된 Amazon EBS EC2 인스턴스에 대해 이전에 릴리스된 지원과 유사하게 구현됩니다. 자세한 내용은 AWS::AutoScaling::LaunchConfiguration의 EbsOptimized 속성을 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
새 설명서	2012년 12월 21일	이제 AWS::EC2::Instance는 EC2 인스턴스의 블록 디바이스 매핑을 설정할 수 있는 BlockDeviceMappings 속성을 제공합니다. 이 변경 사항으로 인해 두 가지 새로운 유형이 추가되었습니다. • BlockDeviceMapping • 블록 디바이스	2010-05-15
새 설명서	2012년 12월 21일	최근에 재설계된 AWS Management Console을 사용하여 스택을 생성하고 보는 절차를 설명하는 새 섹션이 추가되었습니다. 이러한 섹션은 여기에서 찾을 수 있습니다. • CloudFormation 콘솔에서 스택 생성 • CloudFormation 콘솔에서 스택 정보 보기	2010-05-15
새 설명서	2012년 11월 15일	사용자 정의 리소스에 대한 정보는 다음 주제에서 제공됩니다. • 사용자 지정 리소스 • AWS::CloudFormation::CustomResource • 사용자 정의 리소스 참조	2010-05-15
업데이트된 설명서	2012년 11월 15일	이제 CloudFormation에서 RDS DB 인스턴스에 대해 프로비저닝된 IOPS(초당 I/O 작업)를 지정할 수 있습니다. AWS::RDS::DBInstance의 새로운 IOPS 속성을 사용하여 1,000~10,000 범위에서 1000 IOPS 단위로 이 값을 설정할 수 있습니다. RDS DB 인스턴스에 대한 IOPS 지정에 대한 자세한 내용을 알아보려면 Amazon Relational Database Service 사용 설명서의 프로비저닝된 IOPS를 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
새로운 설명서 및 업데이트된 설명서	2012년 8월 27일	AWS Management Console 및 CloudFormation 명령줄 인터페이스(CLI) 사용에 대한 특정 정보를 더욱 명확하게 제공하기 위해 주제가 재구성되었습니다. 새로운 설명서 및 업데이트된 참조 항목을 비롯하여 CloudFormation 스택 태그 지정에 대한 정보가 추가되었습니다. • 새 주제: 스택 옵션 구성. • CloudFormation API 참조의 태그에 대한 새로운 정보: CreateStack, Stack, 및 Tag Windows 스택 사용에 대한 새로운 정보: • Microsoft Windows Amazon 머신 이미지(AMI) 및 CloudFormation 템플릿 • CloudFormation Windows 스택 부트스트랩 새로운 주제: CloudFormation 템플릿에서 정규식 사용.	2010-05-15

변경	릴리스 날짜	설명	API 버전
새로운 기능	2012년 4월 25일	CloudFormation에서는 Amazon EC2로 Virtual Private Cloud(VPC) 보안을 완벽하게 지원합니다. 이제 단일 CloudFormation 템플릿을 사용하여 전체 VPC를 생성하고 모든 VPC 리소스 유형(서브넷, 게이트웨이, 네트워크 ACL, 라우팅 테이블 등)으로 이러한 VPC를 채울 수 있습니다. 다음 리소스 유형의 설명서가 업데이트되었습니다. • AWS::EC2::SecurityGroup • AWS::EC2::SecurityGroupIngress • AWS::EC2::SecurityGroupEgress • AWS::EC2::Instance • AWS::AutoScaling::AutoScalingGroup • AWS::EC2::EIP • AWS::EC2::EIPAssociation • AWS::ElasticLoadBalancing::LoadBalancer 다음과 같은 새로운 리소스 유형이 설명서에 추가되었습니다. • AWS::EC2::VPC • AWS::EC2::InternetGateway • AWS::EC2::DHCPOptions • AWS::EC2::DHCPOptions • AWS::EC2::RouteTable • AWS::EC2::NetworkAcl • AWS::EC2::NetworkAclEntry • AWS::EC2::Subnet • AWS::EC2::VPNGateway • AWS::EC2::CustomerGateway	2010-05-15

변경	릴리스 날짜	설명	API 버전
새로운 기능	2012년 4월 13일	이제 CloudFormation을 사용하여 스택을 업데이트 할 때 스택에서 요소를 추가 또는 제거할 수 있습니다. CloudFormation 스택 업데이트 가 업데이트되었고, 스택을 업데이트할 때 리소스를 추가 및 제거하는 방법을 설명하는 연습: 스택의 리소스 변경 에 새로운 섹션이 추가되었습니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
새로운 기능	2012년 2월 2일	이제 CloudFormation에서는 기존 Amazon Virtual Private Cloud(Amazon VPC)의 리소스를 지원합니다. 이 릴리스를 사용하여 다음 작업을 수행할 수 있습니다. • EC2 전용 인스턴스를 기존 Amazon VPC로 시작합니다. 자세한 내용은 AWS::EC2::Instance 섹션을 참조하세요. • 기존 Amazon VPC에 있는 EC2 인스턴스의 SourceDestCheck 속성을 설정합니다. 자세한 내용은 AWS::EC2::Instance 섹션을 참조하세요. • 기존 Amazon VPC에서 탄력적 IP 주소를 생성합니다. 자세한 내용은 AWS::EC2::EIP 섹션을 참조하세요. • CloudFormation을 사용하여 기존 VPC에서 Amazon VPC 보안 그룹 및 수신/송신 규칙을 생성합니다. 자세한 내용은 AWS::EC2::SecurityGroup 섹션을 참조하세요. • <code>AWS::AutoScaling::AutoScalingGroup</code> 리소스의 <code>VPCZoneIdentifier</code> 속성을 설정하여 Auto Scaling 그룹을 기존 Amazon VPC와 연결합니다. 자세한 내용은 AWS::AutoScaling::AutoScalingGroup 섹션을 참조하세요. • Elastic Load Balancing 로드 밸런서를 Amazon VPC 서브넷에 연결하고 로드 밸런서에 대한 보안 그룹을 생성합니다. 자세한 내용은 AWS::ElasticLoadBalancing::LoadBalancer 섹션을 참조하세요. • 기존 Amazon VPC에서 RDS DB 인스턴스를 생성합니다. 자세한 내용은 AWS::RDS::DBInstance 섹션을 참조하세요.	2010-05-15

변경	릴리스 날짜	설명	API 버전
새로운 기능	2012년 2월 2일	이제 기존 스택에서 다음 리소스의 속성을 업데이트할 수 있습니다. • AWS::EC2::SecurityGroupIngress • AWS::EC2::SecurityGroupEgress • AWS::EC2::EIPAssociation • AWS::RDS::DBSubnetGroup • AWS::RDS::DBSecurityGroup • AWS::RDS::DBSecurityGroupIngress • AWS::Route53::RecordSetGroup 업데이트 가능한 리소스의 전체 목록과 스택 업데이트 시 고려할 수 있는 항목에 대한 세부 정보를 보려면 CloudFormation 스택 업데이트를 참조하세요.	2010-05-15
재구성된 설명서	2012년 2월 2일	기존 섹션을 새로운 섹션 CloudFormation 템플릿 작업 및 스택 관리로 재구성했습니다. 템플릿 참조를 목차의 상단으로 이동했습니다. CloudFormation 스택의 비용 추정을 시작하기 섹션으로 이동했습니다.	2010-05-15
새 콘텐츠	2012년 2월 2일	새 섹션 추가됨: • 연습: 스택 업데이트는 LAMP 스택 업데이트 프로세스를 안내하는 자습서입니다. • AWS CloudFormation을 사용하여 Amazon EC2에 애플리케이션 배포는 CloudFormation 헬퍼 스크립트를 사용하여 템플릿에 저장된 메타데이터를 이용해 애플리케이션을 배포하는 방법을 설명합니다. • CloudFormation 헬퍼 스크립트 참조는 CloudFormation 헬퍼 스크립트(cfn-init, cfn-get-metadata, cfn-signal, cfn-hup)에 대한 참조 자료를 제공합니다.	2010-05-15

변경	릴리스 날짜	설명	API 버전
새로운 기능	2011년 5월 26일	이제 CloudFormation에서는 스택 상태별로 스택을 필터링하여 표시할 수 있는 <code>list-stacks</code> 명령을 제공합니다. 삭제된 스택은 삭제된 후 최대 90일 동안 표시될 수 있습니다. 자세한 내용은 스택 설명 및 나열 을 참조하세요.	2010-05-15
새로운 기능	2011년 5월 26일	이제 <code>describe-stack-resources</code> 및 <code>get-template</code> 명령을 사용하면 스택이 삭제된 이후 90일 동안 삭제된 스택의 정보를 가져올 수 있습니다. 자세한 내용은 리소스 나열 및 템플릿 검색 을 참조하세요.	2010-05-15
새로운 링크	2011년 3월 1일	이제 CloudFormation 엔드포인트 정보의 위치는 AWS 일반 참조입니다. 자세한 내용은 AWS 일반 참조 의 리전 및 엔드포인트를 참조하세요.	2010-05-15
초기 릴리스	2011년 2월 25일	CloudFormation의 최초 공개 릴리스입니다.	2010-05-15