사용자 가이드

AWS Secrets Manager

AWS Secrets Manager: 사용자 가이드

Table of Contents

이게 뭐야 AWS Secrets Manager?

AWS Secrets Manager 수명 주기 전반에 걸쳐 데이터베이스 자격 증명, 애플리케이션 자격 증명, OAuth 토큰, API 키 및 기타 비밀을 관리, 검색 및 교체할 수 있도록 지원합니다. 많은 AWS 서비스가 Secrets Manager에 시크릿을 저장하고 사용합니다.

Secrets Manager를 사용하면 더 이상 애플리케이션 소스 코드에 하드 코딩된 보안 인증 정보가 필요하지 않으므로 보안 태세를 개선할 수 있습니다. Secrets Manager에 보안 인증 정보를 저장하면 애플리케이션 또는 구성 요소를 조사할 수 있는 누군가로 인해 손상될 가능성을 방지할 수 있습니다. 하드 코딩된 보안 인증 정보를 Secrets Manager 서비스에 대한 런타임 호출로 대체하여 필요할 때 동적으로 보안 인증 정보를 검색합니다.

Secrets Manager를 사용하면 암호에 대한 자동 교체 일정을 구성할 수 있습니다. 따라서 단기 보안 암호로 장기 보안 암호를 교체할 수 있어 손상 위험이 크게 줄어듭니다. 보안 인증 정보가 더 이상 애플리케이션에 저장되지 않으므로 보안 인증 정보를 교체할 때 더 이상 애플리케이션을 업데이트하거나 애플리케이션 클라이언트에 변경 사항을 배포하지 않아도 됩니다.

조직에 있을 수 있는 다른 유형의 암호는 다음과 같습니다.

- AWS 자격 증명 — 사용하는 것이 좋습니다 AWS Identity and Access Management.
- 암호화 키 — AWS Key Management Service 권장.
- SSH 키 – Amazon EC2 Instance Connect 권장.
- 프라이빗 키 및 인증서 – AWS Certificate Manager 권장.

Secrets Manager 시작하기

Secrets Manager를 처음 사용하는 경우 다음 자습서 중 하나로 시작하십시오.

- the section called “하드 코딩된 보안 암호 대체 ”
- the section called “하드 코딩된 DB 보안 인증 정보 교체 ”
- the section called “대체 사용자 교체”
- the section called “단일 사용자 교체”

암호를 사용하여 수행할 수 있는 기타 작업은 다음과 같습니다.

- 보안 암호 생성 및 관리

- 암호에 대한 액세스 제어
- 비밀 찾기
- 보안 암호 교체
- 보안 암호 모니터링
- 규정 준수를 위한 비밀 모니터링
- 에서 시크릿을 생성하세요 AWS CloudFormation

표준 준수

AWS Secrets Manager 여러 표준에 대한 감사를 거쳤으며 규정 준수 인증을 받아야 하는 경우 솔루션에 포함될 수 있습니다. 자세한 정보는 규정 준수 확인을 참조하세요.

요금

Secrets Manager를 사용할 경우 사용하는 내역에 대해서만 지불하며 최소 또는 설정 요금이 없습니다. 삭제하도록 표시한 보안 암호에 대해서는 요금이 부과되지 않습니다. 현재 기준의 전체적인 요금 목록은 AWS Secrets Manager 요금을 참조하세요. 비용을 모니터링하려면 을 참조하십시오. the section called “Secrets Manager 비용 모니터링”

Secrets Manager에서 AWS 관리형 키 `aws/secretsmanager` 생성한 데이터를 사용하여 비밀을 무료로 암호화할 수 있습니다. 자체 KMS 키를 생성하여 암호를 암호화하는 경우 현재 요율로 AWS 요금이 부과됩니다. AWS KMS 자세한 내용은 AWS Key Management Service 요금을 참조하십시오.

자동 교체 기능을 켜면 (관리 로테이션 제외) Secrets Manager는 AWS Lambda 함수를 사용하여 암호를 교체하며, 현재 Lambda 요율로 순환 기능에 대한 요금이 부과됩니다. 자세한 내용은 AWS Lambda 요금을 참조하십시오.

AWS CloudTrail 계정에서 활성화하면 Secrets Manager가 보내는 API 호출 로그를 얻을 수 있습니다. Secrets Manager는 모든 이벤트를 관리 이벤트로 기록합니다. AWS CloudTrail 모든 관리 이벤트의 첫 번째 사본을 무료로 저장합니다. 하지만 알림을 활성화한 경우 로그 스토리지용 Amazon S3 및 Amazon SNS에 대한 비용이 발생할 수 있습니다. 또한 추적을 추가로 설정한 경우 관리 이벤트의 추가 사본으로 인해 비용이 발생할 수 있습니다. 자세한 내용은 AWS CloudTrail 요금 섹션을 참조하세요.

액세스 AWS Secrets Manager

다음 방법 중 하나를 사용하여 Secrets Manager로 작업할 수 있습니다.

- Secrets Manager 콘솔
- 명령행 도구
- AWS SDK
- HTTPS 쿼리 API
- AWS Secrets Manager 엔드포인트

Secrets Manager 콘솔

브라우저 기반 Secrets Manager 콘솔을 사용하여 보안 암호를 관리하고 보안 암호와 관련된 거의 모든 작업을 수행할 수 있습니다.

명령행 도구

AWS 명령줄 도구를 사용하면 시스템 명령줄에서 명령을 실행하여 Secrets Manager 및 기타 AWS 작업을 수행할 수 있습니다. 명령줄을 사용하는 것이 콘솔을 사용하는 것보다 더 빠르고 편리할 수 있습니다. 명령줄 도구는 스크립트를 작성하여 AWS 작업을 수행하려는 경우에 유용할 수 있습니다.

명령 셸에 명령을 입력하면 명령 기록이 액세스되거나 유틸리티가 명령 파라미터에 액세스할 위험이 있습니다. the section called “AWS Secrets Manager 보안 암호 저장 시 AWS CLI 사용으로 발생 가능한 위험 줄이기” 섹션을 참조하십시오.

명령줄 도구는 AWS 지역 내 서비스의 기본 엔드포인트를 자동으로 사용합니다. API 요청에 다른 엔드포인트를 지정할 수 있습니다. the section called “Secrets Manager 엔드포인트” 섹션을 참조하십시오.

AWS 두 가지 명령줄 도구 세트를 제공합니다.

- AWS Command Line Interface (AWS CLI)
- AWS Tools for Windows PowerShell

AWS SDK

AWS SDK는 다양한 프로그래밍 언어 및 플랫폼을 위한 라이브러리와 샘플 코드로 구성됩니다. SDK는 요청에 암호화 방식으로 서명, 오류 관리 및 자동으로 요청 재시도와 같은 작업을 포함합니다. SDK를 다운로드하고 설치하려면 Amazon Web Services용 도구를 참조하세요.

AWS SDK는 지역 내 서비스의 기본 엔드포인트를 자동으로 사용합니다. AWS API 요청에 다른 엔드포인트를 지정할 수 있습니다. the section called “Secrets Manager 엔드포인트” 섹션을 참조하십시오.

SDK 설명서는 다음 섹션을 참조하세요.

- C++
- Go
- Java
- JavaScript
- Kotlin
- .NET
- PHP
- 파이썬 (봇 3)
- Ruby
- Rust
- 최대한 빨리
- Swift

HTTPS 쿼리 API

HTTPS 쿼리 API를 사용하면 Secrets Manager 및 에 프로그래밍 방식으로 액세스할 수 있습니다. AWS HTTPS 쿼리 API를 이용하면 HTTPS 요청을 서비스에 직접 보낼 수 있습니다.

Secrets Manager HTTPS 쿼리 API를 직접 호출할 수도 있지만 SDK 중에서 한 가지를 대신 사용하는 것이 좋습니다. 직접 수행해야 하는 많은 유용한 작업을 SDK를 사용하여 수행할 수 있습니다. 예를 들어, SDK는 자동으로 요청에 서명하고, 응답을 해당 언어에 구문상 적절한 구조로 변환합니다.

Secrets Manager에 HTTPS 호출을 하려면 ???에 연결해야 합니다.

AWS Secrets Manager 엔드포인트

Secrets Manager에 프로그래밍 방식으로 연결하려면 해당 서비스에 대한 진입점의 URL인 엔드포인트를 사용합니다. Secrets Manager 엔드포인트는 듀얼 스택 엔드포인트이므로 IPv4와 IPv6를 모두 지원합니다.

Secrets Manager는 일부 리전에서 FIPS(Federal Information Processing Standard) 140-2를 지원하는 엔드포인트를 제공합니다.

Secrets Manager는 TLS 1.2 및 1.3을 지원합니다. Secrets Manager는 중국 리전을 제외한 모든 리전에서 PQTLS를 지원합니다.

> **Note**
>
> Python AWS SDK와 IPv6, IPv4를 차례로 AWS CLI 호출하려고 시도하므로 IPv6를 활성화하지 않은 경우 호출 시간이 초과되어 IPv4로 재시도하는 데 시간이 걸릴 수 있습니다. 이 문제를 해결하려면 IPv6를 완전히 비활성화하거나 IPv6로 마이그레이션합니다.

Secrets Manager에 대한 서비스 엔드포인트는 다음과 같습니다. 이름 지정은 일반적인 듀얼 스택 이름 지정 규칙과 다르다는 점에 유의하세요.

리전 이름	지역	엔드포인트	프로토콜
미국 동부(오하이오)	us-east-2	secretsmanager.us-east-2.amazonaws.com	HTTPS
		secretsmanager-fips.us-east-2.amazonaws.com	HTTPS
미국 동부(버지니아 북부)	us-east-1	secretsmanager.us-east-1.amazonaws.com	HTTPS
		secretsmanager-fips.us-east-1.amazonaws.com	HTTPS
미국 서부(캘리포니아 북부)	us-west-1	secretsmanager.us-west-1.amazonaws.com	HTTPS
		secretsmanager-fips.us-west-1.amazonaws.com	HTTPS

리전 이름	지역	엔드포인트	프로토콜
미국 서부(오레곤)	us-west-2	secretsmanager.us-west-2.amazonaws.com secretsmanager-fips.us-west-2.amazonaws.com	HTTPS HTTPS
아프리카(케이프타운)	af-south-1	secretsmanager.af-south-1.amazonaws.com	HTTPS
아시아 태평양(홍콩)	ap-east-1	secretsmanager.ap-east-1.amazonaws.com	HTTPS
아시아 태평양(하이데라바드)	ap-south-2	secretsmanager.ap-south-2.amazonaws.com	HTTPS
아시아 태평양(자카르타)	ap-southeast-3	secretsmanager.ap-southeast-3.amazonaws.com	HTTPS
아시아 태평양(멜버른)	ap-southeast-4	secretsmanager.ap-southeast-4.amazonaws.com	HTTPS
아시아 태평양(뭄바이)	ap-south-1	secretsmanager.ap-south-1.amazonaws.com	HTTPS
아시아 태평양(오사카)	ap-northeast-3	secretsmanager.ap-northeast-3.amazonaws.com	HTTPS
아시아 태평양(서울)	ap-northeast-2	secretsmanager.ap-northeast-2.amazonaws.com	HTTPS

리전 이름	지역	엔드포인트	프로토콜
아시아 태평양(싱가포르)	ap-southeast-1	secretsmanager.ap-southeast-1.amazonaws.com	HTTPS
아시아 태평양(시드니)	ap-southeast-2	secretsmanager.ap-southeast-2.amazonaws.com	HTTPS
아시아 태평양(도쿄)	ap-northeast-1	secretsmanager.ap-northeast-1.amazonaws.com	HTTPS
캐나다(중부)	ca-central-1	secretsmanager.ca-central-1.amazonaws.com secretsmanager-fips.ca-central-1.amazonaws.com	HTTPS HTTPS
캐나다 서부(캘거리)	ca-west-1	secretsmanager.ca-west-1.amazonaws.com secretsmanager-fips.ca-west-1.amazonaws.com	HTTPS HTTPS
유럽(프랑크푸르트)	eu-central-1	secretsmanager.eu-central-1.amazonaws.com	HTTPS
유럽(아일랜드)	eu-west-1	secretsmanager.eu-west-1.amazonaws.com	HTTPS
유럽(런던)	eu-west-2	secretsmanager.eu-west-2.amazonaws.com	HTTPS
유럽(밀라노)	eu-south-1	secretsmanager.eu-south-1.amazonaws.com	HTTPS
유럽(파리)	eu-west-3	secretsmanager.eu-west-3.amazonaws.com	HTTPS

리전 이름	지역	엔드포인트	프로토콜
유럽(스페인)	eu-south-2	secretsmanager.eu-south-2.amazonaws.com	HTTPS
유럽(스톡홀름)	eu-north-1	secretsmanager.eu-north-1.amazonaws.com	HTTPS
유럽(취리히)	eu-central-2	secretsmanager.eu-central-2.amazonaws.com	HTTPS
이스라엘(텔아비브)	il-central-1	secretsmanager.il-central-1.amazonaws.com	HTTPS
중동(바레인)	me-south-1	secretsmanager.me-south-1.amazonaws.com	HTTPS
중동(UAE)	me-central-1	secretsmanager.me-central-1.amazonaws.com	HTTPS
남아메리카(상파울루)	sa-east-1	secretsmanager.sa-east-1.amazonaws.com	HTTPS
AWS GovCloud(미국 동부)	us-gov-east-1	secretsmanager.us-gov-east-1.amazonaws.com secretsmanager-fips.us-gov-east-1.amazonaws.com	HTTPS HTTPS
AWS GovCloud(미국 서부)	us-gov-west-1	secretsmanager.us-gov-west-1.amazonaws.com secretsmanager-fips.us-gov-west-1.amazonaws.com	HTTPS HTTPS

Secrets Manager의 비밀에는 무엇이 들어 있나요?

Secrets Manager에서 보안 암호는 보안 암호 정보, 보안 암호 값 및 보안 암호에 대한 메타데이터로 구성됩니다. 보안 암호 값은 문자열 또는 이진수일 수 있습니다.

여러 문자열 값을 하나의 비밀에 저장하려면 키-값 쌍이 포함된 JSON 텍스트 문자열을 사용하는 것이 좋습니다. 예를 들면 다음과 같습니다.

```
{
  "host"       : "ProdServer-01.databases.example.com",
  "port"       : "8888",
  "username"   : "administrator",
  "password"   : "EXAMPLE-PASSWORD",
  "dbname"     : "MyDatabase",
  "engine"     : "mysql"
}
```

데이터베이스 암호의 경우 자동 회전을 켜려면 암호에 올바른 JSON 구조의 데이터베이스 연결 정보가 포함되어야 합니다. 자세한 정보는 the section called “보안 암호의 JSON 구조”을 참조하세요.

Metadata

보안 암호의 메타데이터에는 다음이 포함됩니다.

- 다음 형식의 Amazon 리소스 이름(ARN):

  ```
  arn:aws:secretsmanager:<Region>:<AccountId>:secret:SecretName-6RandomCharacters
  ```

  Secrets Manager는 보안 암호 ARN이 고유한지 확인하기 위해 보안 암호 이름의 끝에 임의의 문자 6개를 포함합니다. 원래 보안 암호를 삭제한 후 동일한 이름으로 새 보안 암호를 생성하면 해당 문자로 인해 두 보안 암호의 ARN이 달라집니다. ARN이 다르기 때문에 이전 보안 암호에 액세스할 수 있는 사용자는 새 보안 암호에 자동으로 액세스할 수 없습니다.
- 보안 암호의 이름, 설명, 리소스 정책 및 태그입니다.
- Secrets AWS KMS key Manager가 비밀 값을 암호화하고 해독하는 데 사용하는 암호화 키의 ARN입니다. Secrets Manager는 보안 암호 텍스트를 암호화된 형식으로 저장하고, 전송 중인 보안 암호를 암호화합니다. the section called “보안 암호 암호화 및 복호화” 섹션을 참조하세요.
- 교체를 설정한 경우의 보안 암호 교체 방법에 대한 정보입니다. 보안 암호 교체 섹션을 참조하세요.

Secrets Manager는 IAM 권한 정책을 사용하여 승인된 사용자만 비밀에 액세스하거나 수정할 수 있도록 합니다. 에 대한 인증 및 액세스 제어 AWS Secrets Manager 섹션을 참조하십시오.

시크릿에는 암호화된 비밀 값의 사본이 들어 있는 버전이 있습니다. 보안 암호 값을 변경하거나 보안 암호를 교체할 경우 Secrets Manager는 새 버전을 만듭니다. the section called “시크릿 버전” 섹션을 참조하세요.

암호를 AWS 리전 복제하여 여러 암호에서 사용할 수 있습니다. 보안 암호를 복제할 때 원본 또는 기본 보안 암호의 사본(복제 보안 암호)을 생성합니다. 복제 보안 암호는 기본 보안 암호에 연결된 상태로 유지됩니다. 지역 간 비밀 복제 단원을 참조하세요.

보안 암호 생성 및 관리을(를) 참조하세요.

시크릿 버전

시크릿에는 암호화된 시크릿 값의 사본이 들어 있는 버전이 있습니다. 보안 암호 값을 변경하거나 보안 암호를 교체할 경우 Secrets Manager는 새 버전을 만듭니다.

Secrets Manager는 보안 암호를 버전과 함께 저장하지 않습니다. 대신 레이블을 지정하여 세 가지 특정 버전을 추적합니다.

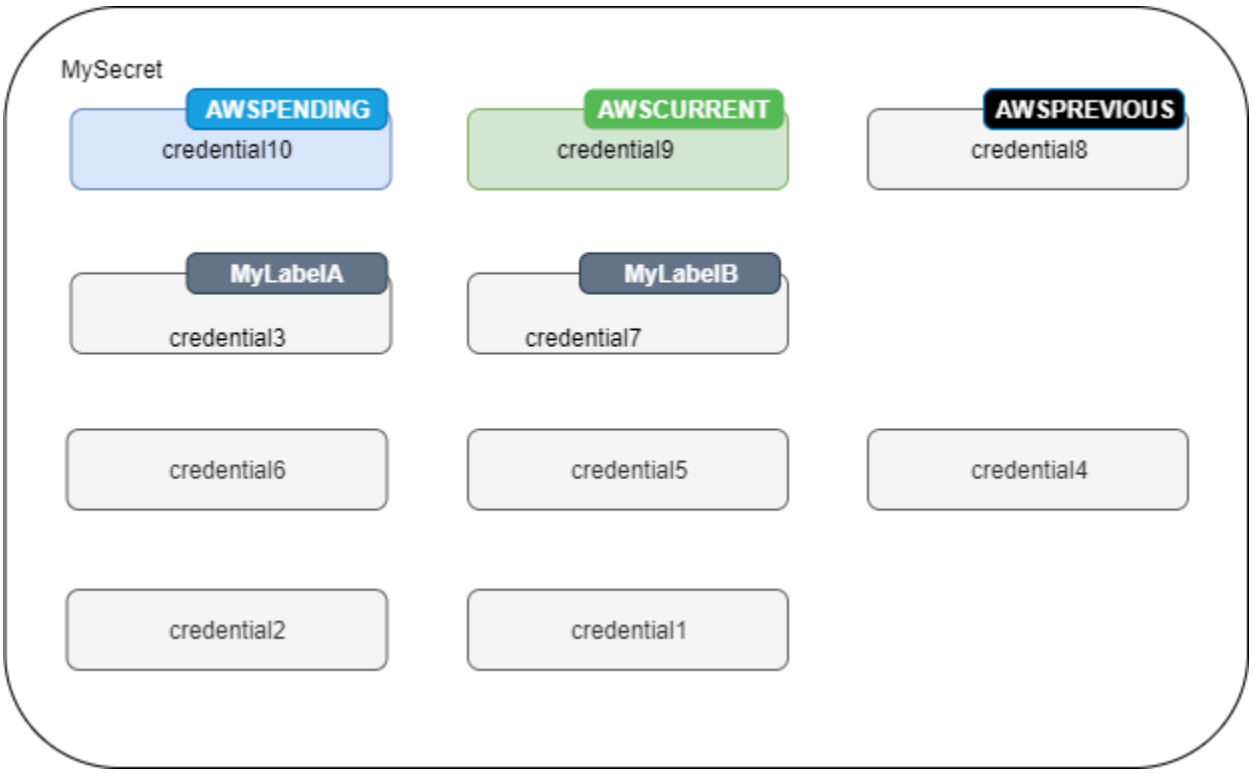
- 현재 버전 - `AWSCURRENT`
- 이전 버전 - `AWSPREVIOUS`
- 보류 버전(교체 중) - `AWSPENDING`

보안 암호에는 항상 라벨 `AWSCURRENT`이 붙은 버전이 있으며, Secrets Manager 사용자가 보안 암호 값을 검색할 때 기본적으로 해당 버전을 반환합니다.

를 `update-secret-version-stage`호출하여 자체 레이블로 버전에 레이블을 지정할 수도 AWS CLI있습니다. 보안 암호에는 20개까지 레이블을 지정할 수 있습니다. 보안 암호의 두 버전에는 동일한 스테이징 레이블을 지정할 수 없습니다. 버전은 라벨이 복수일 수도 있습니다.

Secrets Manager는 레이블이 지정된 버전을 제거하지 않지만 레이블이 지정되지 않은 버전은 더 이상 사용되지 않는 것으로 간주됩니다. Secrets Manager는 더 이상 사용되지 않는 보안 암호 버전이 100개를 초과하면 제거합니다. Secrets Manager는 24 시간 이내에 만든 버전을 제거하지 않습니다.

다음 그림은 레이블이 지정된 버전과 고객 AWS 레이블이 지정된 버전이 있는 비밀을 보여줍니다. 레이블이 없는 버전은 더 이상 사용되지 않는 것으로 간주되며 미래에 Secrets Manager에서 제거될 예정입니다.


MySecret
AWSPENDING
credential10
AWSCURRENT
credential9
AWSPREVIOUS
credential8
MyLabelA
credential3
MyLabelB
credential7
credential6
credential5
credential4
credential2
credential1

AWS Secrets Manager 자습서

주제

Amazon CodeGuru Reviewer를 사용하여 코드에서 보호되지 않는 보안 암호 찾기

Amazon CodeGuru Reviewer는 프로그램 분석 및 기계 학습을 사용하여 개발자가 찾기 어려운 잠재적 결함을 감지하고 Java 및 Python 코드를 개선하기 위한 제안을 제공하는 서비스입니다. CodeGuru Reviewer는 Secrets Manager와 통합되어 코드에서 보호되지 않는 보안 암호를 찾습니다. 찾을 수 있는 보안 암호 유형은 Amazon CodeGuru Reviewer 사용 설명서의 CodeGuru Reviewer가 감지하는 보안 암호 유형을 참조하세요.

하드 코딩된 보안 암호를 발견하면 이를 대체하기 위한 조치를 취하세요.

- the section called “하드 코딩된 DB 보안 인증 정보 교체 ”
- the section called “하드 코딩된 보안 암호 대체 ”

하드코딩된 시크릿을 다음으로 이동 AWS Secrets Manager

코드에 일반 텍스트 보안 암호가 있는 경우 보안 암호를 교체하고 Secrets Manager에 저장하는 것이 좋습니다. 보안 암호를 Secrets Manager로 이동하면 코드를 보는 모든 사용자가 보안 암호를 볼 수 있는 문제가 해결됩니다. 앞으로는 코드가 Secrets Manager에서 직접 보안 암호를 검색하기 때문입니다. 보안 암호를 교체하면 현재의 하드 코딩된 보안 암호가 취소되어 더 이상 유효하지 않게 됩니다.

데이터베이스 보안 인증 정보 보안 암호에 대해서는 하드코딩된 데이터베이스 자격 증명을 다음으로 이동 AWS Secrets Manager 섹션을 참조하세요.

시작하기 전에 보안 암호에 액세스해야 하는 사용자를 결정해야 합니다. 다음과 같이 두 개의 IAM 역할을 사용하여 보안 암호에 대한 권한을 관리하는 것이 좋습니다.

- 조직의 보안 암호를 관리하는 역할. 자세한 정보는 the section called “Secrets Manager 관리자 권한” 섹션을 참조하세요. 이 역할을 사용하여 보안 암호를 생성하고 교체합니다.
- 런타임 시 시크릿을 사용할 수 있는 역할 (예: 이 튜토리얼에서 사용하는 역할). *RoleToRetrieveSecretAtRuntime* 코드가 이 역할을 수임하여 보안 암호를 검색합니다. 이 자습서에서는 역할에 하나의 보안 암호 값을 검색할 수 있는 권한만 부여하고, 보안 암호의 리소스 정책을 사용하여 권한을 부여합니다. 다른 방법을 보려면 the section called “다음 단계” 섹션을 참조하세요.

단계:

1단계: 보안 암호 생성

첫 번째 단계는 기존의 하드 코딩된 보안 암호를 Secrets Manager로 복사하는 것입니다. 비밀이 리소스와 관련된 경우 AWS 리소스와 동일한 지역에 저장하세요. 그렇지 않으면 사용 사례에서 지연 시간이 가장 낮은 지역에 저장하세요.

보안 암호(콘솔) 생성

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. Store a new secret(새 보안 암호 저장)을 선택합니다.
3. 보안 암호 유형 선택(Choose secret type) 페이지에서 다음을 수행합니다.
   a. 보안 암호 유형에서 다른 유형의 보안 암호를 선택합니다.
   b. 보안 암호를 키/값 쌍(Key/value pairs) 또는 일반 텍스트(Plaintext)로 입력합니다. 다음은 일부 예입니다.

API 키 키/값 쌍:

ClientID: ***my_client_id***

ClientSecret : ***bPxRfiWJALR*** XU TN FEMI/K7M ENG/ CY 예제 키

보안 인증 정보 키/값 쌍:

Username: *saanvis*

Password: *EXAMPLE-PASSWORD*

OAuth 토큰 일반 텍스트:

AKIAI44QH8DHBEXAMPLE

디지털 인증서 일반 텍스트:

```
-----BEGIN CERTIFICATE-----
EXAMPLE
-----END CERTIFICATE-----
```

프라이빗 키 일반 텍스트:

```
----BEGIN PRIVATE KEY ---
EXAMPLE
---- END PRIVATE KEY ----
```

c. 암호화 키(Encryption key)에서 aws/secretsmanager를 선택하여 Secrets Manager에 대해 AWS 관리형 키 를 사용합니다. 이 키를 사용하는 데 드는 비용은 없습니다. 예를 들어 다른 AWS 계정에서 보안 암호에 액세스하기 위해 자체 고객 관리형 키를 사용할 수도 있습니다. 고객 관리형 키 사용 비용에 대한 자세한 내용은 요금을 참조하세요.

d. 다음을 선택합니다.

4. 보안 암호 유형 선택(Choose secret type) 페이지에서 다음을 수행합니다.

a. 설명이 포함된 Secret name(보안 암호 이름)과 Description(설명)을 입력합니다.

b. 리소스 권한(Resource permissions)에서 권한 편집(Edit permissions)을 선택합니다. 암호를 검색할 수 있는 다음 정책을 붙여넣은 다음 [Save] 를 선택합니다. ***RoleToRetrieveSecretAtRuntime***

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::AccountId:role/RoleToRetrieveSecretAtRuntime"
      },
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "*"
    }
  ]
}
```

c. 페이지 하단에서 다음(Next)을 선택합니다.

5. 교체 구성(Configure rotation) 페이지에서 교체를 꺼 둡니다. 다음을 선택합니다.
6. Review(검토) 페이지에서 보안 암호 세부 정보를 검토한 후 Store(저장)를 선택합니다.

2단계: 코드 업데이트

코드가 IAM 역할을 *RoleToRetrieveSecretAtRuntime*맡아야 암호를 검색할 수 있습니다. 자세한 내용은 IAM 역할 (AWS API) 로 전환을 참조하십시오.

그런 다음 Secrets Manager에서 제공하는 샘플 코드를 사용하여 Secrets Manager에서 보안 암호를 검색하도록 코드를 업데이트합니다.

샘플 코드 찾기

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호(Secrets) 페이지에서 보안 암호를 선택합니다.
3. 아래로 스크롤하여 샘플 코드(Sample code)로 이동합니다. 프로그래밍 언어를 선택한 다음 코드 조각을 복사합니다.

애플리케이션에서 하드 코딩된 보안 암호를 제거하고 코드 조각을 붙여넣습니다. 코드 언어에 따라 코드 조각의 함수 또는 메서드에 호출을 추가해야 할 수 있습니다.

하드 코딩된 보안 암호 대신 보안 암호를 사용하여 애플리케이션이 예상대로 작동하는지 테스트합니다.

3단계: 보안 암호 업데이트

마지막 단계는 하드 코딩된 보안 암호를 취소하고 업데이트하는 것입니다. 보안 암호의 소스를 참조하여 보안 암호를 취소하고 업데이트하기 위한 지침을 찾습니다. 예를 들어 현재 보안 암호를 비활성화하고 새 보안 암호를 생성해야 할 수 있습니다.

보안 암호를 새 값으로 업데이트

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호(Secrets)를 선택한 다음 보안 암호를 선택합니다.
3. 보안 암호 세부 정보(Secret details) 페이지에서 아래로 스크롤하여 보안 암호 값 검색(Retrieve secret value)을 선택한 다음 편집(Edit)을 선택합니다.
4. 보안 암호를 업데이트 한 다음 저장(Save)을 선택합니다.

그 다음, 새 보안 암호로 애플리케이션이 예상대로 작동하는지 테스트합니다.

다음 단계

코드에서 하드 코딩된 보안 암호를 제거한 후에는 다음과 같은 몇 가지 아이디어를 고려해 볼 수 있습니다.

- Java 및 Python 애플리케이션에서 하드코딩된 보안 정보를 찾으려면 Amazon CodeGuru Reviewer를 사용하는 것이 좋습니다.
- 보안 암호를 캐싱하여 성능을 개선하고 비용을 절감할 수 있습니다. 자세한 정보는 비밀 찾기 섹션을 참조하세요.
- 여러 리전에서 액세스하는 보안 암호의 경우 지연 시간을 개선하기 위해 보안 암호를 복제하는 것을 고려해 보세요. 자세한 정보는 지역 간 비밀 복제을 참조하세요.
- 이 자습서에서는 보안 값을 검색할 수 있는 *RoleToRetrieveSecretAtRuntime*권한만 부여했습니다. 역할에 추가 권한(예: 보안 암호에 대한 메타데이터를 가져오거나 보안 암호 목록 보기)을 부여하려면 the section called “권한 정책 예” 섹션을 참조하세요.
- 이 자습서에서는 시크릿의 리소스 정책을 *RoleToRetrieveSecretAtRuntime*사용하여 권한을 부여했습니다. 권한을 부여하는 다른 방법은 the section called “자격 증명에 권한 정책 연결” 섹션을 참조하세요.

하드코딩된 데이터베이스 자격 증명을 다음으로 이동 AWS Secrets Manager

코드에 일반 텍스트 데이터베이스 보안 인증 정보가 있는 경우 보안 인증 정보를 Secrets Manager로 이동한 다음 즉시 교체하는 것이 좋습니다. 보안 인증 정보를 Secrets Manager로 이동하면 코드를 보는 모든 사용자가 보안 인증 정보를 볼 수 있는 문제가 해결됩니다. 앞으로는 코드가 Secrets Manager에서 직접 보안 인증 정보를 검색하기 때문입니다. 보안 암호를 교체하면 암호가 업데이트된 다음 현재의 하드 코딩된 암호가 취소되어 더 이상 유효하지 않게 됩니다.

Amazon RDS, Amazon Redshift, Amazon DocumentDB 데이터베이스의 경우, 이 페이지에 제시된 단계를 사용하여 하드 코딩된 보안 인증 정보를 Secrets Manager로 이동합니다. 다른 유형의 보안 인증 정보 및 다른 보안 암호에 대해서는 the section called “하드 코딩된 보안 암호 대체 ” 섹션을 참조하세요.

시작하기 전에 보안 암호에 액세스해야 하는 사용자를 결정해야 합니다. 다음과 같이 두 개의 IAM 역할을 사용하여 보안 암호에 대한 권한을 관리하는 것이 좋습니다.

- 조직의 보안 암호를 관리하는 역할. 자세한 정보는 the section called “Secrets Manager 관리자 권한” 섹션을 참조하세요. 이 역할을 사용하여 보안 암호를 생성하고 교체합니다.
- 이 자습서에서는 런타임 시 자격 증명을 사용할 수 있는 역할을 *RoleToRetrieveSecretAtRuntime*설명합니다. 코드가 이 역할을 수임하여 보안 암호를 검색합니다.

단계:

- 1단계: 보안 암호 생성
- 2단계: 코드 업데이트
- 3단계: 보안 암호 교체
- 다음 단계

1단계: 보안 암호 생성

첫 번째 단계는 기존의 하드 코딩된 보안 인증 정보를 Secrets Manager의 보안 암호로 복사하는 것입니다. 지연 시간을 최소화하려면 보안 암호를 데이터베이스와 동일한 리전에 저장합니다.

보안 암호 생성

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. Store a new secret(새 보안 암호 저장)을 선택합니다.
3. 보안 암호 유형 선택(Choose secret type) 페이지에서 다음을 수행합니다.

   a. 보안 암호 유형(Secret type)에서 저장할 데이터베이스 보안 인증 정보 유형을 선택합니다.

      - Amazon RDS 데이터베이스
      - Amazon DocumentDB 데이터베이스
      - 아마존 Redshift 데이터 웨어하우스.
      - 다른 유형의 보안 암호에 대해서는 하드 코딩된 보안 암호 대체를 참조하세요.

   b. 보안 인증 정보에서 데이터베이스에 대한 기존의 하드 코딩된 보안 인증 정보를 입력합니다.

   c. 암호화 키(Encryption key)에서 aws/secretsmanager를 선택하여 Secrets Manager에 대해 AWS 관리형 키 를 사용합니다. 이 키를 사용하는 데 드는 비용은 없습니다. 예를 들어 다른 AWS 계정에서 보안 암호에 액세스하기 위해 자체 고객 관리형 키를 사용할 수도 있습니다. 고객 관리형 키 사용 비용에 대한 자세한 내용은 요금을 참조하세요.

   d. 데이터베이스에서 데이터베이스(Database)를 선택합니다.

   e. Next(다음)를 선택합니다.

4. 보안 구성(Configure secret) 페이지에서 다음을 수행합니다.

   a. 설명이 포함된 Secret name(보안 암호 이름)과 Description(설명)을 입력합니다.

   b. 리소스 권한(Resource permissions)에서 권한 편집(Edit permissions)을 선택합니다. 보안 검색을 허용하는 *RoleToRetrieveSecretAtRuntime*다음 정책을 붙여넣은 다음 [Save] 를 선택합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::AccountId:role/RoleToRetrieveSecretAtRuntime"
      },
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "*"
    }
  ]
```

```
}
```

c. 페이지 하단에서 다음(Next)을 선택합니다.

5. 교체 구성(Configure rotation) 페이지에서 지금은 교체를 꺼 둡니다. 나중에 켜게 됩니다. 다음을 선택합니다.
6. Review(검토) 페이지에서 보안 암호 세부 정보를 검토한 후 Store(저장)를 선택합니다.

2단계: 코드 업데이트

코드가 IAM 역할을 *RoleToRetrieveSecretAtRuntime*맡아야 암호를 검색할 수 있습니다. 자세한 내용은 IAM 역할 (AWS API) 로 전환을 참조하십시오.

그런 다음 Secrets Manager에서 제공하는 샘플 코드를 사용하여 Secrets Manager에서 보안 암호를 검색하도록 코드를 업데이트합니다.

샘플 코드 찾기

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호(Secrets) 페이지에서 보안 암호를 선택합니다.
3. 아래로 스크롤하여 샘플 코드(Sample code)로 이동합니다. 언어를 선택한 다음 코드 조각을 복사합니다.

애플리케이션에서 하드 코딩된 보안 인증 정보를 제거하고 코드 조각을 붙여넣습니다. 코드 언어에 따라 코드 조각의 함수 또는 메서드에 호출을 추가해야 할 수 있습니다.

하드 코딩된 보안 인증 정보 대신 보안 암호를 사용하여 애플리케이션이 예상대로 작동하는지 테스트합니다.

3단계: 보안 암호 교체

마지막 단계는 보안 암호를 교체하여 하드 코딩된 보안 인증 정보를 취소하는 것입니다. 교체는 주기적으로 보안 암호를 업데이트하는 프로세스입니다. 보안 암호를 교체하면 보안 암호와 데이터베이스 모두에서 보안 인증 정보가 업데이트됩니다. Secrets Manager는 사용자가 설정한 일정에 따라 보안 암호를 자동으로 교체할 수 있습니다.

교체 설정의 일부는 Lambda 교체 함수가 Secrets Manager와 데이터베이스에 모두 액세스할 수 있도록 하는 것입니다. 자동 교체를 설정하면 Secrets Manager가 데이터베이스에 대한 네트워크 액세스 권한을 갖도록 데이터베이스와 동일한 VPC에 Lambda 교체 함수를 생성합니다. 또한 Lambda 교체 함

수는 Secrets Manager를 호출하여 보안 암호를 업데이트할 수 있어야 합니다. Lambda에서 Secrets Manager로의 호출이 인프라를 벗어나지 않도록 VPC에 Secrets Manager 엔드포인트를 생성하는 것이 좋습니다. AWS 지침은 VPC 엔드포인트 섹션을 참조하세요.

교체 켜기

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호(Secrets) 페이지에서 보안 암호를 선택합니다.
3. 보안 암호 세부 정보(Secret details) 페이지의 교체 구성(Rotation configuration) 섹션에서 교체 편집(Edit rotation)을 선택합니다.
4. Edit rotation configuration(교체 구성 편집) 대화 상자에서 다음을 수행합니다.
   a. Automatic rotation(자동 교체)을 켭니다.
   b. 교체 일정(Rotation schedule)에서 UTC 표준 시간대로 일정을 입력합니다.
   c. 변경 사항을 저장할 때 보안 암호가 교체되게 하려면 보안 암호를 저장할 때 즉시 교체(Rotate immediately when the secret is stored)를 선택합니다.
   d. 교체 함수(Rotation function)에서 새 Lambda 함수 생성(Create a new Lambda function)을 선택하고 새 함수의 이름을 입력합니다. Secrets Manager에서 함수 이름의 시작 부분에 "SecretsManager"를 추가합니다.
   e. 교체 전략에서 단일 사용자를 선택합니다.
   f. 저장(Save)을 선택합니다.

보안 암호가 교체되었는지 확인

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호(Secrets)를 선택한 다음 보안 암호를 선택합니다.
3. 보안 암호 세부 정보(Secret details) 페이지를 아래로 스크롤하고 보안 암호 값 검색(Retrieve secret value)을 선택합니다.

   보안 암호 값이 변경되었으면 교체가 성공한 것입니다. 암호 값이 변경되지 않은 경우에는 CloudWatch 로그에서 교체 문제 해결 순환 함수를 확인해야 합니다.

교체된 보안 암호로 애플리케이션이 예상대로 작동하는지 테스트합니다.

다음 단계

코드에서 하드 코딩된 보안 암호를 제거한 후에는 다음과 같은 몇 가지 아이디어를 고려해 볼 수 있습니다.

- 보안 암호를 캐싱하여 성능을 개선하고 비용을 절감할 수 있습니다. 자세한 정보는 비밀 찾기 섹션을 참조하세요.
- 다른 교체 일정을 선택할 수 있습니다. 자세한 정보는 the section called “로테이션 스케줄”을 참조하세요.
- Java 및 Python 애플리케이션에서 하드코딩된 보안 정보를 찾으려면 Amazon CodeGuru Reviewer를 사용하는 것이 좋습니다.

대체 유저 로테이션 설정 AWS Secrets Manager

이 자습서에서는 데이터베이스 자격 증명이 포함된 보안 암호에 대해 대체 사용자 교체를 설정하는 방법을 알아봅니다. 대체 사용자 교체는 Secrets Manager가 사용자를 복제한 다음 업데이트되는 사용자의 자격 증명을 대체하는 교체 전략입니다. 이 전략은 보안 암호의 고가용성이 필요한 경우 선택하는 것이 좋습니다. 대체 사용자 중 한 명이 업데이트되는 동안 다른 사용자에게 데이터베이스에 대한 현재 자격 증명이 있기 때문입니다. 자세한 정보는 the section called “대체 사용자”을 참조하세요.

대체 사용자 교체를 설정하려면 다음 두 개의 보안 암호가 필요합니다.

- 교체하려는 자격 증명이 포함된 보안 암호 1개와
- 관리자 자격 증명이 있는 두 번째 암호입니다.

  이 사용자는 첫 번째 사용자를 복제하고 첫 번째 사용자의 암호를 변경할 권한이 있습니다. 이 자습서에서는 Amazon RDS에서 관리자 사용자를 위해 이 암호를 생성하도록 합니다. Amazon RDS는 관리자 암호 교체도 관리합니다. 자세한 정보는 the section called “관리형 교체”을 참조하세요.

이 자습서의 첫 부분은 사실적인 환경을 설정하는 것입니다. 이 자습서에서는 교체 방식을 보여주기 위해 Amazon RDS MySQL 데이터베이스 예제를 사용합니다. 보안을 위해 데이터베이스는 인바운드 인터넷 액세스를 제한하는 VPC에 있습니다. 인터넷을 통해 로컬 컴퓨터에서 데이터베이스에 연결하려면 Bastion Host를 사용합니다. 이것은 데이터베이스에 연결할 수 있지만 인터넷으로부터의 SSH 연결도 허용하는 VPC 서버입니다. 이 자습서의 Bastion Host는 Amazon EC2 인스턴스이며, 인스턴스의 보안 그룹은 다른 유형의 연결을 차단합니다.

자습서를 마친 후에는 자습서에서 리소스를 정리하는 것이 좋습니다. 프로덕션 환경에서 리소스를 사용하지 마세요.

Secrets Manager 로테이션은 AWS Lambda 함수를 사용하여 비밀과 데이터베이스를 업데이트합니다. Lambda 함수 사용 비용에 대한 자세한 내용은 요금 섹션을 참조하세요.

자습서

권한

자습서 사전 조건으로 AWS 계정에 대한 관리 권한이 필요합니다. 프로덕션 설정에서는 각 단계에 대해 서로 다른 역할을 사용하는 것이 가장 좋습니다. 예를 들어 데이터베이스 관리자 권한이 있는 역할은 Amazon RDS 데이터베이스를 생성하고 네트워크 관리자 권한이 있는 역할은 VPC 및 보안 그룹을 설정합니다. 자습서 단계에서는 동일한 자격 증명을 계속 사용하는 것이 좋습니다.

프로덕션 환경에서 권한을 설정하는 방법에 대한 자세한 내용은 인증 및 액세스 제어 섹션을 참조하세요.

사전 조건

이 자습서를 이해하려면 다음이 필요합니다.

사전 조건 A: Amazon VPC

이 단계에서는 Amazon RDS 데이터베이스와 Amazon EC2 인스턴스를 시작할 수 있는 VPC를 생성합니다. 나중 단계에서는 컴퓨터를 사용하여 인터넷을 통해 Bastion에 연결한 다음 데이터베이스에 연결하게 되므로 VPC에서 나가는 트래픽을 허용해야 합니다. 이를 위해 Amazon VPC는 VPC에 인터넷 게이트웨이를 연결하고 VPC 외부로 향하는 트래픽을 인터넷 게이트웨이로 보내도록 라우팅 테이블에 경로를 추가합니다.

VPC 내에서 Secrets Manager 엔드포인트와 Amazon RDS 엔드포인트를 생성합니다. 나중 단계에서 자동 교체를 설정하면 Secrets Manager가 데이터베이스에 액세스할 수 있도록 VPC 내에 Lambda 교체 함수를 생성합니다. 또한 Lambda 교체 함수는 Secrets Manager 를 호출하여 암호를 업데이트하고 Amazon RDS를 호출하여 데이터베이스 연결 정보를 가져옵니다. VPC 내에 엔드포인트를 생성하면 Lambda 함수에서 Secrets Manager 및 Amazon RDS로의 호출이 인프라를 벗어나지 않도록 할 수 있습니다. AWS 대신 VPC 내의 엔드포인트로 라우팅됩니다.

VPC를 생성하려면

1. https://console.aws.amazon.com/vpc/에서 Amazon VPC 콘솔을 여세요.
2. VPC 생성을 선택합니다.
3. Create VPC(VPC 생성) 페이지에서 VPC 등을 선택합니다.
4. Name tag auto-generation(이름 태그 자동 생성)의 Auto-generate(자동 생성)에서 **`SecretsManagerTutorial`**을 입력합니다.
5. DNS options(DNS 옵션)에서 **`Enable DNS hostnames`** 및 **`Enable DNS resolution`**을 모두 선택합니다.
6. VPC 생성을 선택합니다.

VPC 내에 Secrets Manager 엔드포인트를 생성하려면

1. Amazon VPC 콘솔의 Endpoints(엔드포인트)에서 Create Endpoint(엔드포인트 생성)를 선택합니다.
2. Endpoint settings(엔드포인트 설정)에서 Name(이름)에 **`SecretsManagerTutorialEndpoint`**를 입력합니다.
3. Services(서비스)에서 **`secretsmanager`**를 입력하여 목록을 필터링한 다음 AWS 리전에서 Secrets Manager 엔드포인트를 선택합니다. 예를 들어 미국 동부(버지니아 북부)에서 `com.amazonaws.us-east-1.secretsmanager`를 선택합니다.
4. VPC에 **`vpc**** (SecretsManagerTutorial)`**를 선택합니다.

5. 서브넷(Subnets)에 가용 영역(Availability Zones)을 모두 선택한 다음 각각에 대해 포함할 서브넷 ID(Subnet ID)를 선택합니다.
6. IP address type(IP 주소 유형)에서 **IPv4**를 선택합니다.
7. 보안 그룹(Security groups)에서 기본 보안 그룹을 선택합니다.
8. 정책(Policy)에서 **Full access**를 선택합니다.
9. 엔드포인트 생성(Create endpoint)을 선택합니다.

VPC 내에 Amazon RDS 엔드포인트를 생성하려면

1. Amazon VPC 콘솔의 Endpoints(엔드포인트)에서 Create Endpoint(엔드포인트 생성)를 선택합니다.
2. Endpoint settings(엔드포인트 설정)에서 Name(이름)에 **RDSTutorialEndpoint**를 입력합니다.
3. Services(서비스)에서 **rds**를 입력하여 목록을 필터링한 다음 AWS 리전에서 Amazon RDS 엔드포인트를 선택합니다. 예를 들어 미국 동부(버지니아 북부)에서 `com.amazonaws.us-east-1.rds`를 선택합니다.
4. VPC에 **vpc**** (SecretsManagerTutorial)**를 선택합니다.
5. 서브넷(Subnets)에 가용 영역(Availability Zones)을 모두 선택한 다음 각각에 대해 포함할 서브넷 ID(Subnet ID)를 선택합니다.
6. IP address type(IP 주소 유형)에서 **IPv4**를 선택합니다.
7. 보안 그룹(Security groups)에서 기본 보안 그룹을 선택합니다.
8. 정책(Policy)에서 **Full access**를 선택합니다.
9. 엔드포인트 생성(Create endpoint)을 선택합니다.

사전 조건 B: Amazon EC2 인스턴스

나중 단계에서 생성하는 Amazon RDS 데이터베이스는 VPC에 있게 되므로 여기에 액세스하려면 Bastion Host가 필요합니다. Bastion Host는 VPC에도 있지만 나중 단계에서 SSH를 사용하여 로컬 컴퓨터가 Bastion Host에 연결할 수 있도록 보안 그룹을 구성합니다.

Bastion Host에 대한 EC2 인스턴스를 생성하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 인스턴스(Instances)를 선택한 다음 인스턴스 시작(Launch Instances)을 선택합니다.

3. Name and tags(이름 및 태그) 아래의 Name(이름)에 **SecretsManagerTutorialInstance**을 입력하세요.
4. Application and OS Images(애플리케이션 및 OS 이미지)에서 기본값 **Amazon Linux 2 AMI (HMV) Kernel 5.10**을 유지합니다.
5. Instance type(인스턴스 유형)에서 기본값 **t2.micro**를 유지합니다.
6. Key pair(키 페어)에서 Create key pair(키 페어 생성)를 선택합니다.

   Create key pair(키 페어 생성) 대화 상자에서 Key pair name(키 페어 이름)에 **SecretsManagerTutorialKeyPair**를 입력한 다음 Create key pair(키 페어 생성)를 선택합니다.

   키 페어가 자동으로 다운로드됩니다.
7. Network settings(네트워크 설정)에서 Edit(편집)를 선택하고 다음을 수행합니다.
   a. VPC에 **vpc-**** SecretsManagerTutorial**를 선택합니다.
   b. 퍼블릭 IP 자동 할당(Auto-assign Public IP)에서 **Enable**을 선택합니다.
   c. Firewall(방화벽)에서 Select existing security group(기존 보안 그룹 선택)을 선택합니다.
   d. Common security groups(일반 보안 그룹)에서 **default**를 선택합니다.
8. 인스턴스 시작을 선택합니다.

사전 조건 C: Amazon RDS 데이터베이스 및 관리자 자격 증명을 위한 Secrets Manager 암호

이 단계에서는 Amazon RDS MySQL 데이터베이스를 생성하고 Amazon RDS가 관리자 자격 증명을 포함할 암호를 생성하도록 구성합니다. 그러면 Amazon RDS에서 관리자 암호의 교체를 자동으로 관리합니다. 자세한 정보는 관리형 교체을 참조하세요.

데이터베이스 생성 과정의 일환으로 이전 단계에서 생성한 Bastion Host를 지정합니다. 그러면 Amazon RDS에서 데이터베이스와 인스턴스가 서로 액세스할 수 있도록 보안 그룹을 설정합니다. 로컬 컴퓨터도 연결할 수 있도록 인스턴스에 연결된 보안 그룹에 규칙을 추가합니다.

관리자 자격 증명이 포함된 Secrets Manager 암호를 사용하여 Amazon RDS 데이터베이스를 생성하려면

1. Amazon RDS 콘솔에서 Create database(데이터베이스 생성)를 선택합니다.
2. Engine options(엔진 옵션) 섹션의 Engine type(엔진 유형)에서 **MySQL**을 선택합니다.

3. Templates(템플릿) 섹션에서 **Free tier**를 선택합니다.
4. Settings(설정) 섹션에서 다음을 수행합니다.
   a. DB instance identifier(DB 인스턴스 식별자)에 **SecretsManagerTutorial**을 입력합니다.
   b. 자격 증명 설정에서 마스터 자격 증명 관리를 선택합니다. AWS Secrets Manager
5. Connectivity(연결성)의 Computer resource(컴퓨터 리소스)에서 Connect to an EC2 computer resource(EC2 컴퓨터 리소스에 연결)를 선택한 다음 EC2 Instance(EC2 인스턴스)에서 **SecretsManagerTutorialInstance**를 선택합니다.
6. 데이터베이스 생성를 선택합니다.

사전 조건 D: 로컬 컴퓨터가 EC2 인스턴스에 연결하도록 허용

이 단계에서는 사전 조건 B에서 생성한 EC2 인스턴스를 구성하여 로컬 컴퓨터가 해당 인스턴스에 연결할 수 있도록 합니다. 이렇게 하려면 컴퓨터의 IP 주소가 SSH와 연결할 수 있도록 하는 규칙을 포함하도록 Amazon RDS가 사전 조건 C에 추가한 보안 그룹을 편집해야 합니다. 이 규칙을 사용하면 로컬 컴퓨터(현재 IP 주소로 식별됨)가 인터넷을 통해 SSH를 사용하여 Bastion Host에 연결할 수 있습니다.

로컬 컴퓨터가 EC2 인스턴스에 연결하도록 허용하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. EC2 인스턴스의 SecretsManagerTutorialInstance보안 탭에 있는 보안 그룹에서 선택합니다. **sg-*** (ec2-rds-X)**
3. Input rules(입력 규칙)에서 Edit inbound rules(인바운드 규칙 편집)를 선택합니다.
4. Add rule(규칙 추가)을 선택하고 규칙에 대해 다음을 수행합니다.
   a. 유형(Type)에서 **SSH**를 선택합니다.
   b. Source type(소스 유형)에 **My IP**를 선택합니다.

1단계: Amazon RDS 데이터베이스 사용자 생성

먼저 보안 암호에 자격 증명을 저장할 사용자가 필요합니다. 사용자를 생성하려면 관리자 자격 증명으로 Amazon RDS 데이터베이스에 로그인합니다. 간소화를 위해 자습서에서는 데이터베이스에 대한 전체 권한을 가진 사용자를 생성합니다. 프로덕션 환경에서 이는 일반적이지 않으므로 최소 권한 원칙을 따르는 것이 좋습니다.

데이터베이스에 연결하려면 MySQL 클라이언트 도구를 사용합니다. 이 자습서에서는 GUI 기반 애플리케이션인 MySQL Workbench를 사용합니다. MySQL Workbench를 설치하려면 MySQL Workbench 다운로드를 참조하세요.

데이터베이스에 연결하려면 MySQL Workbench에서 연결 구성을 생성합니다. 구성을 위해 Amazon EC2 및 Amazon RDS의 일부 정보가 필요합니다.

MySQL Workbench에서 데이터베이스 연결을 생성하려면

1. MySQL Workbench에서 MySQL 연결(MySQL Connections) 옆에 있는 (+) 버튼을 선택합니다.
2. 새 연결 설정(Setup New Connection) 대화 상자에서 다음을 수행합니다.
   a. 연결 이름(Connection Name)에 **SecretsManagerTutorial**을 입력합니다.
   b. 연결 방법(Connection Method)에서 **Standard TCP/IP over SSH**를 선택합니다.
   c. 파라미터(Parameters) 탭에서 다음을 수행합니다.
      i. SSH 호스트 이름(SSH Hostname)에 Amazon EC2 인스턴스의 퍼블릭 IP 주소를 입력합니다.

         Amazon EC2 콘솔에서 인스턴스를 선택하여 IP 주소를 찾을 수 있습니다. SecretsManagerTutorialInstance Public IPv4 DNS 아래에 IP 주소를 복사합니다.
      ii. SSH 사용자 이름(SSH Username)에 **ec2-user**를 입력합니다.
      iii. SSH 키파일의 경우 이전 사전 요구 사항에서 다운로드한 키 페어 SecretsManagerTutorialKeyPair파일.pem을 선택합니다.
      iv. MySQL 호스트 이름(MySQL Hostname)에 Amazon RDS 엔드포인트 주소를 입력합니다.

         데이터베이스 인스턴스 secretsmanagertutorialdb를 선택하여 Amazon RDS 콘솔에서 엔드포인트 주소를 찾을 수 있습니다. 엔드포인트(Endpoint) 아래에 주소를 복사합니다.
      v. 사용자 이름(Username)에 **admin**을 입력합니다.
   d. 확인(OK)을 선택합니다.

관리자 암호를 검색하려면

1. Amazon RDS 콘솔에서 데이터베이스로 이동합니다.
2. Configuration(구성) 탭의 Master Credentials ARN(마스터 자격 증명 ARN)에서 Manage in Secrets Manager(Secrets Manager에서 관리)를 선택합니다.

Secrets Manager 콘솔이 열립니다.

3. 보안 암호 세부 정보 페이지에서 Retrieve secret value(보안 암호 값 검색)를 선택합니다.
4. 암호가 Secret value(암호 값) 섹션에 표시됩니다.

데이터베이스 사용자를 생성하려면

1. MySQL 워크벤치에서 연결을 선택합니다. SecretsManagerTutorial
2. 암호에서 검색한 관리자 암호를 입력합니다.
3. MySQL Workbench의 Query(쿼리) 창에서 다음 명령(강력한 암호 포함)을 입력한 다음 Execute(실행)를 선택합니다.

```
CREATE DATABASE myDB;
CREATE USER 'appuser'@'%' IDENTIFIED BY 'EXAMPLE-PASSWORD';
GRANT ALL PRIVILEGES ON myDB . * TO 'appuser'@'%';
```

출력(Output) 창에서 명령이 성공한 것을 볼 수 있습니다.

2단계: 사용자 자격 증명에 대한 보안 암호 생성

다음으로, 방금 생성한 사용자의 자격 증명을 저장하는 보안 암호를 생성합니다. 이것이 교체하게 될 보안 암호입니다. 자동 교체를 설정하고 대체 사용자 전략을 나타내려면 첫 번째 사용자의 암호를 변경할 권한이 있는 별도의 슈퍼 사용자 보안 암호를 선택합니다.

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. Store a new secret(새 보안 암호 저장)을 선택합니다.
3. 보안 암호 유형 선택(Choose secret type) 페이지에서 다음을 수행합니다.
   a. 보안 암호 유형(Secret type)에서 Amazon RDS 데이터베이스에 대한 자격 증명(Credentials for Amazon RDS database)을 선택합니다.
   b. 자격 증명(Credentials)에서 사용자 이름 **appuser**와 MySQL Workbench를 사용하여 생성한 데이터베이스 사용자에 대해 입력한 암호를 입력합니다.
   c. 데이터베이스(Database)에서 secretsmanagertutorialdb를 선택합니다.
   d. 다음을 선택합니다.
4. 보안 암호 구성(Configure secret) 페이지에서 보안 암호 이름(Secret name)에 **SecretsManagerTutorialAppuser**를 입력한 후 다음(Next)을 선택합니다.

5. 교체 구성(Configure rotation) 페이지에서 다음을 수행합니다.

   a. Automatic rotation(자동 교체)을 켭니다.

   b. 교체 일정(Rotation schedule)에서 일(Days): **2**일, 기간(Duration): **2h**의 일정을 설정합니다. 즉시 교체(Rotate immediately)를 선택합니다.

   c. 교체 함수(Rotation function)에서 교체 함수 생성(Create a rotation function)을 선택한 다음 함수 이름에 **`tutorial-alternating-users-rotation`**을 입력합니다.

   d. 교체 전략에서 대체 사용자를 선택한 다음 관리자 보안 인증 정보 암호에서 이 자습서에서 생성한 데이터베이스의 이름 **`secretsmanagertutorial`**을 포함하는 설명이 있는(예: `Secret associated with primary RDS DB instance: arn:aws:rds:`*`Region`*`:`*`AccountId`*`:db:secretsmanagertutorial`) rds!cluster...라는 이름의 암호를 선택합니다.

   e. 다음(Next)을 선택합니다.

6. 검토(Review) 페이지에서 시작(Store)을 선택합니다.

   Secrets Manager가 비밀 세부 정보 페이지로 돌아갑니다. 페이지 상단에서 교체 구성 상태를 확인할 수 있습니다. Secrets Manager는 Lambda 순환 함수 및 Lambda 함수를 실행하는 실행 역할과 같은 리소스를 생성하는 데 사용합니다 CloudFormation . CloudFormation 완료하면 배너가 순환 예약된 시크릿으로 변경됩니다. 첫 번째 교체가 완료되었습니다.

3단계: 교체된 보안 암호 테스트

이제 보안 암호가 교체되었으므로 보안 암호에 유효한 새 자격 증명이 포함되어 있는지 확인할 수 있습니다. 보안 암호의 암호가 원래 자격 증명에서 변경되었습니다.

보안 암호에서 새 암호를 검색하려면

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호(Secrets)를 선택한 다음 보안 암호 **`SecretsManagerTutorialAppuser`**를 선택합니다.
3. 보안 암호 세부 정보(Secret details) 페이지를 아래로 스크롤하고 보안 암호 값 검색(Retrieve secret value)을 선택합니다.
4. 키/값(Key/value) 테이블에서 **`password`**에 대한 보안 암호 값(Secret value)을 복사합니다.

자격 증명을 테스트하려면

1. MySQL Workbench에서 연결을 마우스 오른쪽 단추로 클릭한 다음 연결 SecretsManagerTutorial편집을 선택합니다.
2. 서버 연결 관리(Manage Server Connections) 대화 상자에서 사용자 이름(Username)에 **appuser**를 입력한 다음 닫기(Close)를 선택합니다.
3. MySQL 워크벤치로 돌아가서 연결을 선택합니다. SecretsManagerTutorial
4. SSH 연결 열기(Open SSH Connection) 대화 상자에서 암호(Password)에 보안 암호에서 검색한 암호를 붙여넣은 다음 확인(OK)을 선택합니다.

   자격 증명이 유효한 경우 MySQL Workbench가 데이터베이스의 디자인 페이지에 열립니다.

이는 보안 암호 교체가 성공했음을 나타냅니다. 보안 암호의 자격 증명이 업데이트되었으며 데이터베이스에 연결할 수 있는 유효한 암호입니다.

4단계: 리소스 정리

다른 교체 전략인 single user rotation(단일 사용자 교체)을 시도하려는 경우 리소스 정리를 건너뛰고 the section called “단일 사용자 교체”로 이동합니다.

그렇지 않으면 잠재적 요금이 부과되지 않도록 하고 인터넷에 액세스할 수 있는 EC2 인스턴스를 제거하기 위해 이 자습서에서 생성한 다음 리소스와 그 사전 조건을 삭제합니다.

- Amazon RDS 데이터베이스 인스턴스. 자세한 내용은 Amazon RDS 사용 설명서의 DB 인스턴스 삭제를 참조하세요.
- Amazon EC2 인스턴스. 지침은 Amazon EC2 사용 설명서의 인스턴스 종료를 참조하십시오.
- Secrets Manager 보안 암호 `SecretsManagerTutorialAppuser` 지침은 the section called “보안 암호 삭제” 섹션을 참조하세요.
- Secrets Manager 엔드포인트. 자세한 내용은AWS PrivateLink 설명서의 VPC 엔드포인트 삭제를 참조하세요.
- VPC 엔드포인트. 자세한 내용은AWS PrivateLink 설명서의 VPC 삭제를 참조하세요.

다음 단계

- 애플리케이션에서 보안 암호를 검색하는 방법을 알아봅니다.
- 다른 교체 일정에 대해 알아봅니다.

AWS Secrets Manager에 대한 단일 사용자 교체 설정

이 자습서에서는 데이터베이스 자격 증명이 포함된 보안 암호에 대해 단일 사용자 교체를 설정하는 방법을 알아봅니다. 단일 사용자 교체는 Secrets Manager가 보안 암호 및 데이터베이스 모두에서 사용자의 자격 증명을 업데이트하는 교체 전략입니다. 자세한 정보는 the section called “단일 사용자”을 참조하세요.

자습서를 마친 후에는 자습서에서 리소스를 정리하는 것이 좋습니다. 프로덕션 환경에서 리소스를 사용하지 마세요.

Secrets Manager 로테이션은 AWS Lambda 함수를 사용하여 비밀과 데이터베이스를 업데이트합니다. Lambda 함수 사용 비용에 대한 자세한 내용은 요금 섹션을 참조하세요.

목차

권한

자습서 사전 조건으로 AWS 계정에 대한 관리 권한이 필요합니다. 프로덕션 설정에서는 각 단계에 대해 서로 다른 역할을 사용하는 것이 가장 좋습니다. 예를 들어 데이터베이스 관리자 권한이 있는 역할은 Amazon RDS 데이터베이스를 생성하고 네트워크 관리자 권한이 있는 역할은 VPC 및 보안 그룹을 설정합니다. 자습서 단계에서는 동일한 자격 증명을 계속 사용하는 것이 좋습니다.

프로덕션 환경에서 권한을 설정하는 방법에 대한 자세한 내용은 인증 및 액세스 제어 섹션을 참조하세요.

사전 조건

이 자습서의 사전 조건은 the section called “대체 사용자 교체”입니다. 첫 번째 자습서가 끝나면 리소스를 정리하지 마세요. 이 자습서를 마친 후에는 Amazon RDS 데이터베이스와 Secrets Manager 보안

암호가 있는 실제 환경을 갖춥니다. 데이터베이스 사용자의 자격 증명이 포함된 두 번째 암호도 있지만 이 자습서에서는 해당 암호를 사용하지 않습니다.

또한 관리자 자격 증명을 사용하여 데이터베이스에 연결하도록 MySQL Workbench에 구성되어 있습니다.

1단계: Amazon RDS 데이터베이스 사용자 생성

먼저 보안 암호에 자격 증명을 저장할 사용자가 필요합니다. 사용자를 생성하려면 암호에 저장된 관리자 자격 증명으로 Amazon RDS 데이터베이스에 로그인합니다. 간소화를 위해 자습서에서는 데이터베이스에 대한 전체 권한을 가진 사용자를 생성합니다. 프로덕션 환경에서 이는 일반적이지 않으므로 최소 권한 원칙을 따르는 것이 좋습니다.

관리자 암호를 검색하려면

1. Amazon RDS 콘솔에서 데이터베이스로 이동합니다.
2. Configuration(구성) 탭의 Master Credentials ARN(마스터 자격 증명 ARN)에서 Manage in Secrets Manager(Secrets Manager에서 관리)를 선택합니다.

   Secrets Manager 콘솔이 열립니다.
3. 보안 암호 세부 정보 페이지에서 Retrieve secret value(보안 암호 값 검색)를 선택합니다.
4. 암호가 Secret value(암호 값) 섹션에 표시됩니다.

데이터베이스 사용자를 생성하려면

1. MySQL Workbench에서 연결을 마우스 오른쪽 단추로 클릭한 다음 연결 SecretsManagerTutorial편집을 선택합니다.
2. 서버 연결 관리(Manage Server Connections) 대화 상자에서 사용자 이름(Username)에 **admin**를 입력한 다음 닫기(Close)를 선택합니다.
3. MySQL 워크벤치로 돌아가서 연결을 선택합니다. SecretsManagerTutorial
4. 암호에서 검색한 관리자 암호를 입력합니다.
5. MySQL Workbench의 Query(쿼리) 창에서 다음 명령(강력한 암호 포함)을 입력한 다음 Execute(실행)를 선택합니다.

```
CREATE USER 'dbuser'@'%' IDENTIFIED BY 'EXAMPLE-PASSWORD';
GRANT ALL PRIVILEGES ON myDB . * TO 'dbuser'@'%';
```

출력(Output) 창에서 명령이 성공한 것을 볼 수 있습니다.

2단계: 데이터베이스 사용자 자격 증명에 대한 보안 암호 생성

다음으로, 방금 생성한 사용자의 자격 증명을 저장하는 보안 암호를 생성하고, 즉시 교체를 포함한 자동 교체를 켭니다. Secrets Manager는 암호를 교체하므로 암호는 프로그래밍 방식으로 생성됩니다. 이 새 암호를 본 사람은 아무도 없습니다. 교체가 즉시 시작되도록 하면 교체가 올바르게 설정되었는지 확인할 수 있습니다.

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. Store a new secret(새 보안 암호 저장)을 선택합니다.
3. 보안 암호 유형 선택(Choose secret type) 페이지에서 다음을 수행합니다.
   a. 보안 암호 유형(Secret type)에서 Amazon RDS 데이터베이스에 대한 자격 증명(Credentials for Amazon RDS database)을 선택합니다.
   b. 자격 증명(Credentials)에서 사용자 이름 **dbuser**와 MySQL Workbench를 사용하여 생성한 데이터베이스 사용자에 대해 입력한 암호를 입력합니다.
   c. 데이터베이스(Database)에서 secretsmanagertutorialdb를 선택합니다.
   d. 다음을 선택합니다.
4. 보안 암호 구성(Configure secret) 페이지에서 보안 암호 이름(Secret name)에 **SecretsManagerTutorialDbuser**를 입력한 후 다음(Next)을 선택합니다.
5. 교체 구성(Configure rotation) 페이지에서 다음을 수행합니다.
   a. Automatic rotation(자동 교체)을 켭니다.
   b. 교체 일정(Rotation schedule)에서 일(Days): **2**일, 기간(Duration): **2h**의 일정을 설정합니다. 즉시 교체(Rotate immediately)를 선택합니다.
   c. 교체 함수(Rotation function)에서 교체 함수 생성(Create a rotation function)을 선택한 다음 함수 이름에 **tutorial-single-user-rotation**을 입력합니다.
   d. 교체 전략에서 단일 사용자를 선택합니다.
   e. 다음(Next)을 선택합니다.
6. 검토(Review) 페이지에서 시작(Store)을 선택합니다.

   Secrets Manager가 비밀 세부 정보 페이지로 돌아갑니다. 페이지 상단에서 교체 구성 상태를 확인할 수 있습니다. Secrets Manager는 Lambda 로테이션 함수 및 Lambda 함수를 실행하는 실행 역

할과 같은 리소스를 생성하는 데 사용합니다 CloudFormation . CloudFormation 완료하면 배너가 순환 예약된 시크릿으로 변경됩니다. 첫 번째 교체가 완료되었습니다.

3단계: 교체된 암호 테스트

몇 초가 걸릴 수 있는 첫 번째 보안 암호 교체 이후 보안 암호에 유효한 자격 증명이 여전히 포함되어 있는지 확인할 수 있습니다. 보안 암호의 암호가 원래 자격 증명에서 변경되었습니다.

보안 암호에서 새 암호를 검색하려면

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호(Secrets)를 선택한 다음 보안 암호 **SecretsManagerTutorialDbuser**를 선택합니다.
3. 보안 암호 세부 정보(Secret details) 페이지를 아래로 스크롤하고 보안 암호 값 검색(Retrieve secret value)을 선택합니다.
4. 키/값(Key/value) 테이블에서 **password**에 대한 보안 암호 값(Secret value)을 복사합니다.

자격 증명을 테스트하려면

1. MySQL Workbench에서 연결을 마우스 오른쪽 단추로 클릭한 다음 연결 SecretsManagerTutorial편집을 선택합니다.
2. 서버 연결 관리(Manage Server Connections) 대화 상자에서 사용자 이름(Username)에 **dbuser**를 입력한 다음 닫기(Close)를 선택합니다.
3. MySQL 워크벤치로 돌아가서 연결을 선택합니다. SecretsManagerTutorial
4. SSH 연결 열기(Open SSH Connection) 대화 상자에서 암호(Password)에 보안 암호에서 검색한 암호를 붙여넣은 다음 확인(OK)을 선택합니다.

   자격 증명이 유효한 경우 MySQL Workbench가 데이터베이스의 디자인 페이지에 열립니다.

4단계: 리소스 정리

잠재적 요금을 방지하려면 이 자습서에서 생성한 암호를 삭제합니다. 지침은 the section called “보안 암호 삭제” 섹션을 참조하세요.

이전 자습서에서 만든 리소스를 정리하려면 the section called “4단계: 리소스 정리”를 참조하세요.

다음 단계

- 애플리케이션에서 보안 암호를 검색하는 방법을 알아봅니다. 비밀 찾기 섹션을 참조하십시오.
- 다른 교체 일정에 대해 알아봅니다. the section called “로테이션 스케줄”을 참조하세요.

에 대한 인증 및 액세스 제어 AWS Secrets Manager

Secrets Manager는 AWS Identity and Access Management (IAM)를 사용하여 보안 암호에 대한 액세스를 제어합니다. IAM은 인증 및 액세스 제어를 제공합니다. 인증은 개인 요청의 자격 증명을 확인합니다. Secrets Manager는 암호, 액세스 키 및 멀티 팩터 인증(MFA) 토큰을 통해 로그인 프로세스를 사용하여 사용자의 자격 증명을 확인합니다. 로그인을 참조하십시오 AWS. 액세스 제어는 승인된 개인만 AWS 리소스(예: 보안 암호)에 대한 작업을 수행할 수 있도록 합니다. Secrets Manager는 정책을 사용하여 리소스에 액세스할 수 있는 사용자 및 해당 리소스에 대해 자격 증명이 수행할 수 있는 작업을 정의합니다. IAM의 정책 및 권한을 참조하세요.

Secrets Manager 관리자 권한

Secrets Manager 관리자 권한을 부여하려면 IAM 자격 증명 권한 추가 및 제거의 지침을 따르고 다음 정책을 연결합니다.

- SecretsManagerReadWrite
- IAMFullAccess

최종 사용자에게는 관리자 권한을 부여하지 않는 것이 좋습니다. 이를 통해 사용자는 자신의 보안 암호를 생성하고 관리할 수 있지만 교체를 활성화하는 데 필요한 권한(IAMFullAccess)은 최종 사용자에게 적합하지 않은 중요한 권한을 부여합니다.

보안 암호에 액세스할 수 있는 권한

IAM 권한 정책을 사용하여 보안 암호에 액세스하는 사용자 또는 서비스를 제어할 수 있습니다. 권한 정책은 누가 어떤 리소스에 대해 어떤 작업을 수행할 수 있는지 설명합니다. 다음을 할 수 있습니다.

- the section called “자격 증명에 권한 정책 연결”
- the section called “보안 암호에 권한 정책 연결”

Lambda 교체 함수에 대한 권한

Secrets Manager는 AWS Lambda 함수를 사용하여 비밀을 교체합니다. Lambda 함수는 보안 암호에 자격 증명이 포함된 데이터베이스 또는 서비스뿐 아니라 보안 암호에 액세스할 수 있어야 합니다. 교체 권한 단원을 참조하세요.

암호화 키 권한

Secrets Manager는 AWS Key Management Service (AWS KMS) 키를 사용하여 비밀을 암호화합니다. 에는 AWS 관리형 키 `aws/secretsmanager` 자동으로 올바른 권한이 부여됩니다. 다른 KMS 키를 사용하는 경우 Secrets Manager는 해당 키에 대한 권한이 필요합니다. the section called “KMS 키에 대한 권한” 단원을 참조하세요.

복제에 대한 권한

IAM 권한 정책을 사용하면 암호를 다른 지역에 복제할 수 있는 사용자 또는 서비스를 제어할 수 있습니다. the section called “복제 방지”를 참조하세요.

자격 증명에 권한 정책 연결

권한 정책을 IAM 자격 증명: 사용자, 사용자, 그룹 및 역할에 연결합니다. 자격 증명 기반 정책에서 자격 증명에 액세스할 수 있는 보안 암호와, 자격 증명이 보안 암호에 대해 수행할 수 있는 작업을 지정합니다. 자세한 내용은 IAM 자격 증명 권한 추가 및 제거를 참조하세요.

다른 서비스의 애플리케이션이나 사용자를 나타내는 역할에 권한을 부여할 수 있습니다. 예를 들어 Amazon EC2 인스턴스에서 실행되는 애플리케이션은 데이터베이스에 액세스해야 할 수 있습니다. EC2 인스턴스 프로파일에 연결된 IAM 역할을 생성한 후 권한 정책을 사용하여 데이터베이스의 보안 인증 정보를 포함한 보안 암호에 대한 액세스 권한을 역할에 부여할 수 있습니다. 자세한 내용은 IAM 역할을 사용하여 Amazon EC2 인스턴스에서 실행되는 애플리케이션에 권한 부여를 참조하세요. 역할을 연결할 수 있는 기타 서비스에는 Amazon Redshift, AWS Lambda, Amazon ECS가 있습니다.

IAM 이외의 자격 증명 시스템으로 인증된 사용자에게 권한을 부여할 수도 있습니다. 예를 들어 IAM 역할을 Amazon Cognito로 로그인하는 모바일 앱 사용자와 연결할 수 있습니다. 이 역할은 앱에 역할 권한 정책에 있는 권한과 함께 임시 자격 증명을 부여합니다. 그런 다음 권한 정책을 사용하여 해당 역할에 보안 암호에 대한 액세스 권한을 부여할 수 있습니다. 자세한 내용은 Identity providers and federation(자격 증명 공급자 및 페더레이션)을 참조하세요.

자격 증명 기반 정책을 사용하여 다음을 수행할 수 있습니다.

- 여러 보안 암호에 대한 자격 증명 액세스 권한을 부여합니다.
- 새 보안 암호를 만들 수 있는 사용자와, 아직 생성되지 않은 보안 암호에 액세스할 수 있는 사용자를 제어합니다.
- IAM 그룹에 보안 암호에 대한 액세스 권한을 부여합니다.

자세한 내용은 the section called “권한 정책 예” 섹션을 참조하세요.

AWS Secrets Manager 보안 암호에 권한 정책 연결

리소스 기반 정책에서 보안 암호에 액세스할 수 있는 사용자와, 보안 암호에 대해 수행할 수 있는 작업을 지정합니다. 리소스 기반 정책을 사용하여 다음을 수행할 수 있습니다.

- 여러 사용자 및 역할에 단일 보안 암호에 대한 액세스 권한을 부여합니다.
- 다른 AWS 계정의 사용자 또는 역할에 액세스 권한을 부여합니다.

the section called “권한 정책 예” 단원을 참조하세요.

리소스 기반 정책을 콘솔의 보안 암호에 연결하면 Secret Secrets Manager는 자동화된 추론 엔진 Zelkova 및 API `ValidateResourcePolicy`을(를) 사용하여 다양한 IAM 보안 주체에 보안 암호에 대한 액세스 권한을 부여하지 못하도록 방지할 수 있습니다. 또는 CLI나 SDK에서 `BlockPublicPolicy` 파라미터가 있는 `PutResourcePolicy` API 를 호출할 수 있습니다.

> **Important**
>
> 리소스 정책 유효성 검사 및 `BlockPublicPolicy` 파라미터는 비밀에 직접 연결된 리소스 정책을 통해 퍼블릭 액세스가 부여되는 것을 방지함으로써 리소스를 보호하는 데 도움이 됩니다. 이러한 기능을 사용하는 것 외에도 다음 정책을 주의 깊게 검토하여 공개 액세스를 허용하지 않는지 확인하십시오.
>
> - 관련 AWS 주체에 연결된 ID 기반 정책 (예: IAM 역할)
> - 관련 리소스에 연결된 AWS 리소스 기반 정책 (예: () 키) AWS Key Management Service AWS KMS
>
> 비밀에 대한 권한을 검토하려면 을 참조하십시오. 보안 암호에 대한 권한이 있는 사용자 확인

보안 암호(콘솔)에 대한 리소스 정책을 보거나 변경하거나 삭제하려면

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호 목록에서 보안 암호를 선택합니다.
3. 보안 암호 세부 정보 페이지의 개요 탭에 있는 리소스 권한 섹션에서 권한 편집을 선택합니다.
4. 코드 필드에서 다음 중 하나를 수행한 후 저장(Save)을 선택합니다.

- 리소스 정책을 연결하거나 수정하려면 정책을 입력합니다.
- 정책을 삭제하려면 코드 필드를 지웁니다.

AWS CLI

Example 리소스 정책의 검색

다음 get-resource-policy 예시에서는 보안 암호에 연결된 리소스 기반 정책을 검색합니다.

```
aws secretsmanager get-resource-policy \
    --secret-id MyTestSecret
```

Example 리소스 정책 삭제

다음 delete-resource-policy 예시에서는 보안 암호에 연결된 리소스 기반 정책을 삭제합니다.

```
aws secretsmanager delete-resource-policy \
    --secret-id MyTestSecret
```

Example 리소스 정책 추가

다음 put-resource-policy 예시에서는 보안 암호에 사용 권한 정책을 추가하여 해당 정책이 암호에 대한 광범위한 액세스를 제공하지 않는지 먼저 확인합니다. 파일에서 해당 정책을 읽습니다. 자세한 내용은 AWS CLI 사용 설명서의 파일에서 AWS CLI 매개변수 로드를 참조하십시오.

```
aws secretsmanager put-resource-policy \
    --secret-id MyTestSecret \
    --resource-policy file://mypolicy.json \
    --block-public-policy
```

mypolicy.json의 콘텐츠:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:role/MyRole"
            },
```

```
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "*"
        }
    ]
}
```

AWS SDK

보안 암호에 연결된 정책을 검색하려면 `GetResourcePolicy`를 사용합니다.

보안 암호에 연결된 정책을 삭제하려면 `DeleteResourcePolicy`를 사용합니다.

보안 암호에 정책을 연결하려면 `PutResourcePolicy`를 사용합니다. 이미 정책이 연결되어 있는 경우 이 명령은 해당 정책을 새 정책으로 대체합니다. 정책 문서는 JSON 구조의 텍스트 형식이어야 합니다. JSON 정책 문서 구조를 참조하세요. the section called “권한 정책 예”를 사용하여 정책 작성을 시작합니다.

자세한 내용은 the section called “AWS SDK”을(를) 참조하세요.

AWS 에 대한 관리형 정책 AWS Secrets Manager

AWS 관리형 정책은 에서 생성하고 관리하는 독립형 정책입니다. AWS AWS 관리형 정책은 많은 일반 사용 사례에 대한 권한을 제공하도록 설계되었으므로 사용자, 그룹 및 역할에 권한을 할당하기 시작할 수 있습니다.

AWS 관리형 정책은 모든 AWS 고객이 사용할 수 있으므로 특정 사용 사례에 대해 최소 권한 권한을 부여하지 않을 수도 있다는 점에 유의하세요. 사용 사례에 고유한 고객 관리형 정책을 정의하여 권한을 줄이는 것이 좋습니다.

관리형 정책에 정의된 권한은 변경할 수 없습니다. AWS AWS 관리형 정책에 정의된 권한을 업데이트하는 경우 AWS 해당 업데이트는 정책이 연결된 모든 주체 ID (사용자, 그룹, 역할) 에 영향을 미칩니다. AWS 새 API 작업이 시작되거나 기존 서비스에 새 AWS 서비스 API 작업을 사용할 수 있게 되면 AWS 관리형 정책을 업데이트할 가능성이 가장 높습니다.

자세한 내용은 IAM 사용 설명서의 AWS 관리형 정책을 참조하십시오.

AWS 관리형 정책: SecretsManagerReadWrite

이 정책은 Amazon RDS AWS Secrets Manager, Amazon Redshift 및 Amazon DocumentDB 리소스를 설명할 수 있는 권한과 암호를 암호화하고 해독하는 데 사용할 권한을 포함하여 읽기/쓰기 액세스를 제

공합니다. AWS KMS 또한 이 정책은 AWS CloudFormation 변경 세트를 생성하고, 관리하는 Amazon S3 버킷에서 순환 템플릿을 가져오고, AWS Lambda 함수를 나열하고 AWS, Amazon EC2 VPC를 설명할 수 있는 권한을 제공합니다. 콘솔에서 기존 교체 함수를 사용하여 교체를 설정하려면 이러한 권한이 필요합니다.

새 순환 함수를 생성하려면 AWS CloudFormation 스택과 AWS Lambda 실행 역할을 생성할 수 있는 권한도 있어야 합니다. IAM FullAccess 관리형 정책을 할당할 수 있습니다. 교체 권한 섹션을 참조하십시오.

권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- `secretsmanager` – 보안 주체가 모든 Secrets Manager 작업을 수행할 수 있도록 허용합니다.
- `cloudformation`— 보안 주체가 스택을 생성할 수 있습니다. AWS CloudFormation 이는 콘솔을 사용하여 로테이션을 켜는 보안 주체가 스택을 통해 Lambda 순환 함수를 생성할 수 있도록 하기 위해 필요합니다. AWS CloudFormation 자세한 정보는 the section called “Secrets Manager의 AWS CloudFormation 사용 방식”을 참조하세요.
- `ec2` – 보안 주체가 Amazon EC2 VPC에 대해 설명할 수 있도록 허용합니다. 이는 콘솔을 사용하는 보안 주체가 보안 암호에 저장하는 보안 인증 정보의 데이터베이스와 동일한 VPC에서 교체 함수를 생성할 수 있도록 하기 위해 필요합니다.
- `kms`— 보안 주체가 암호화 작업에 키를 사용할 수 있습니다. AWS KMS 이는 Secrets Manager가 보안 암호를 암호화하고 해독할 수 있도록 하기 위해 필요합니다. 자세한 정보는 the section called “보안 암호 암호화 및 복호화”을 참조하세요.
- `lambda` – 보안 주체가 Lambda 교체 함수를 나열할 수 있도록 허용합니다. 이는 콘솔을 사용하는 보안 주체가 기존 교체 함수를 선택할 수 있도록 하기 위해 필요합니다.
- `rds` – 보안 주체가 Amazon RDS의 클러스터 및 인스턴스를 설명할 수 있도록 허용합니다. 이는 콘솔을 사용하는 보안 주체가 Amazon RDS 클러스터 또는 인스턴스를 선택할 수 있도록 하기 위해 필요합니다.
- `redshift` – 보안 주체가 Amazon Redshift의 클러스터를 설명할 수 있도록 허용합니다. 이는 콘솔을 사용하는 보안 주체가 Amazon Redshift 클러스터를 선택할 수 있도록 하기 위해 필요합니다.
- `redshift-serverless`— 보안 주체가 Amazon Redshift 서버리스의 네임스페이스를 설명할 수 있도록 합니다. 이는 콘솔을 사용하는 보안 주체가 Amazon Redshift 서버리스 네임스페이스를 선택할 수 있도록 하기 위해 필요합니다.

- `docdb-elastic` – 보안 주체가 Amazon DocumentDB의 탄력적 클러스터를 설명할 수 있도록 허용합니다. 이는 콘솔을 사용하는 보안 주체가 Amazon DocumentDB 탄력적 클러스터를 선택할 수 있도록 하기 위해 필요합니다.
- `tag` – 보안 주체가 계정 내에서 태그가 지정된 모든 리소스를 가져올 수 있도록 허용합니다.
- `serverlessrepo`— 주도자가 변경 세트를 생성할 수 있습니다. AWS CloudFormation 이는 콘솔을 사용하는 보안 주체가 Lambda 교체 함수를 생성할 수 있도록 하기 위해 필요합니다. 자세한 정보는 the section called “Secrets Manager의 AWS CloudFormation 사용 방식”을 참조하세요.
- `s3`— 보안 주체가 에서 관리하는 Amazon S3 버킷에서 객체를 가져올 수 있습니다. AWS이 버킷에는 Lambda 교체 함수 템플릿이 포함되어 있습니다. 이 권한은 콘솔을 사용하는 보안 주체가 버킷의 템플릿을 기반으로 Lambda 교체 함수를 생성할 수 있도록 하기 위해 필요합니다. 자세한 정보는 the section called “Secrets Manager의 AWS CloudFormation 사용 방식”을 참조하세요.

정책을 보려면 SecretsManagerReadWrite JSON 정책 문서를 참조하십시오.

Secrets Manager의 AWS 관리형 정책 업데이트

Secrets Manager의 AWS 관리형 정책 업데이트에 대한 세부 정보를 확인하십시오.

변경 사항	설명	날짜
SecretsManagerReadWrite - 기존 정책에 대한 업데이트	이 정책은 콘솔 사용자가 Amazon Redshift 암호를 생성할 때 Amazon Redshift 서버리스 네임스페이스를 선택할 수 있도록 Amazon Redshift 서버리스에 대한 설명 액세스를 허용하도록 업데이트되었습니다.	2024년 3월 12일
SecretsManagerReadWrite-기존 정책 업데이트	이 정책은 콘솔 사용자가 Amazon DocumentDB 보안 암호를 생성할 때 탄력적 클러스터를 선택할 수 있도록 Amazon DocumentDB 탄력적 클러스터에 대한 설명 액세스를 허용하도록 업데이트되었습니다.	2023년 9월 12일

변경 사항	설명	날짜
SecretsManagerReadWrite-기존 정책 업데이트	이 정책은 콘솔 사용자가 Amazon Redshift 보안 암호를 생성할 때 Amazon Redshift 클러스터를 선택할 수 있도록 Amazon Redshift에 대한 설명 액세스를 허용하도록 업데이트되었습니다. 이 업데이트에는 Lambda 순환 함수 템플릿을 저장하는 Amazon S3 버킷에 AWS 대한 읽기 액세스를 허용하는 새로운 권한도 추가되었습니다.	2020년 6월 24일
SecretsManagerReadWrite-기존 정책 업데이트	이 정책은 콘솔 사용자가 Amazon RDS 암호를 생성할 때 클러스터를 선택할 수 있도록 Amazon RDS 클러스터에 대한 설명 액세스를 허용하도록 업데이트되었습니다.	2018년 5월 3일
SecretsManagerReadWrite - 새 정책	Secrets Manager는 Secrets Manager에 대한 모든 읽기/쓰기 액세스 권한과 함께 콘솔을 사용하는 데 필요한 권한을 부여하는 정책을 만들었습니다.	2018년 4월 4일
Secrets Manager, 변경 사항 추적 시작	Secrets Manager는 AWS 관리형 정책의 변경 사항을 추적하기 시작했습니다.	2018년 4월 4일

AWS Secrets Manager 보안 암호에 대한 권한이 있는 사용자 확인

기본적으로 IAM 자격 증명에는 보안 암호에 대한 액세스 권한이 없습니다. 보안 암호에 대한 액세스를 승인할 때 Secrets Manager는 보안 암호에 연결된 리소스 기반 정책과, 요청을 제출하는 IAM 사용자나

역할에 연결된 모든 자격 증명 기반 정책을 평가합니다. 이를 수행하기 위해 Secrets Manager는 IAM 사용 설명서의 요청 허용 또는 거부 여부 결정에 설명된 프로세스와 유사한 프로세스를 사용합니다.

요청에 적용되는 정책이 여러 개인 경우 Secrets Manager는 계층 구조를 사용하여 권한을 제어합니다.

1. 명시적인 deny가 있는 정책의 설명이 요청 작업 및 리소스와 일치하는 경우:

   명시적인 deny는 다른 모든 것을 무시하고 작업을 차단합니다.
2. 명시적인 deny가 없지만 명시적인 allow가 있는 설명이 요청 작업 및 리소스와 일치하는 경우:

   명시적인 allow는 설명의 리소스에 대한 액세스 권한을 요청의 작업에 부여합니다.

   자격 증명과 보안 암호가 서로 다른 두 계정에 있는 경우 보안 암호에 대한 리소스 정책 및 자격 증명에 연결된 정책 모두에 allow가 있어야 합니다. 그렇지 않은 경우에는 AWS가 요청을 거부합니다. 자세한 정보는 크로스 계정 액세스을 참조하세요.
3. 요청 작업 및 리소스와 일치하는 명시적인 allow가 있는 설명이 없는 경우:

   AWS는 기본적으로 요청을 거부하며, 이를 암묵적 거부라고도 합니다.

보안 암호에 대한 리소스 기반 정책을 보려면

- 다음 중 하나를 수행하세요.
  - https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다. 보안 암호에 대한 보안 암호 세부 정보 페이지의 리소스 권한(Resource permissions) 섹션에서 권한 편집(Edit permissions)을 선택합니다.
  - AWS CLI을(를) 사용하여 get-resource-policy을(를) 호출하거나 AWS SDK를 사용하여 GetResourcePolicy을(를) 호출합니다.

자격 증명 기반 정책을 통해 액세스할 수 있는 사용자를 확인하려면 다음을 수행합니다.

- IAM 정책 시뮬레이터를 사용합니다. IAM 정책 시뮬레이터로 IAM 정책 테스트를 참조하세요.

다른 AWS Secrets Manager 계정의 액세스 시크릿

하나의 계정에 속한 사용자가 다른 계정의 보안 암호에 액세스하도록 허용하려면(교차 계정 액세스) 리소스 정책 및 ID 정책 모두에서 액세스를 허용해야 합니다. 이는 보안 암호와 동일한 계정의 자격 증명에 액세스를 부여하는 것과 다릅니다.

또한 보안 암호를 암호화하는 KMS 키를 자격 증명에서 사용하도록 허용해야 합니다. 계정 간 액세스에는 AWS 관리형 키 (aws/secretsmanager) 를 사용할 수 없기 때문입니다. 대신 생성한 KMS 키로 보안 암호를 암호화한 후 키 정책을 연결해야 합니다. KMS 키를 생성하는 데에는 요금이 부과됩니다. 보안 암호에 대한 암호화 키를 변경하려면 the section called “보안 암호 수정” 단원을 참조하세요.

다음 예제 정책에서는 계정1에 보안 암호 및 암호화 키가 있고, 보안 암호 값에 액세스할 수 있도록 허용할 자격 증명이 계정2에 있다고 가정합니다.

1단계: Account1의 보안 암호에 리소스 정책 연결

- *## ### ApplicationRoleAccount2## Account1# ### ##### ## #####.* 이 정책을 사용하려면 the section called “보안 암호에 권한 정책 연결” 섹션을 참조하세요.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::Account2:role/ApplicationRole"
      },
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "*"
    }
  ]
}
```

2단계: Account1의 KMS 키에 대한 키 정책에 명령문 추가

- *## ## ## ApplicationRole##### ## 2## ## 1# KMS ## #### ## 1# ### ### # ### #####.* 이 명령문을 사용하려면 KMS 키의 키 정책에 추가합니다. 자세한 내용은 키 정책 변경을 참조하세요.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::Account2:role/ApplicationRole"
  },
  "Action": [
    "kms:Decrypt",
    "kms:DescribeKey"
```

```
  ],
  "Resource": "*"
}
```

3단계: Account2의 자격 증명에 자격 증명 정책 연결

- *## ### ## ApplicationRole2## ## 1# ### ##### ## 1## ## ### ## #### ## # # ##### #####.* 이 정책을 사용하려면 the section called “자격 증명에 권한 정책 연결” 단원을 참조하세요. Secrets Manager 콘솔의 보안 암호 세부 정보 페이지에 있는 보안 암호 ARN 아래에서 보안 암호에 대한 ARN을 찾을 수 있습니다. 또는 `describe-secret`을 호출할 수 있습니다.

```
{
  "Version" : "2012-10-17",
  "Statement" : [
    {
      "Effect": "Allow",
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "SecretARN"
    },
    {
      "Effect": "Allow",
      "Action": "kms:Decrypt",
      "Resource": "arn:aws:kms:Region:Account1:key/EncryptionKey"
    }
  ]
}
```

온-프레미스 환경의 액세스 시크릿

AWS Identity and Access Management Roles Anywhere를 사용하면 외부에서 실행되는 서버, 컨테이너 및 애플리케이션과 같은 워크로드에 대한 IAM의 임시 보안 자격 증명을 얻을 수 있습니다. AWS애플리케이션에서 리소스에 액세스하는 데 사용하는 것과 동일한 IAM 정책 및 IAM 역할을 워크로드에 사용할 수 있습니다. AWS AWS IAM Roles Anywhere를 사용하면 Secrets Manager를 사용하여 애플리케이션 서버와 같은 온프레미스 AWS 디바이스뿐만 아니라 내부 리소스가 액세스할 수 있는 자격 증명을 저장하고 관리할 수 있습니다. 자세한 내용은 IAM Roles Anywhere 사용 설명서를 참조하세요.

에 대한 권한 정책 예제 AWS Secrets Manager

권한 정책은 JSON 구조의 텍스트입니다. JSON 정책 문서 구조를 참조하세요.

리소스 및 자격 증명에 연결하는 권한 정책은 매우 유사합니다. 보안 암호에 액세스하기 위해 정책에 포함되는 몇 가지 요소는 다음과 같습니다.

- `Principal`: 액세스 권한을 부여할 사람. IAM 사용 설명서의 보안 주체 지정을 참조하세요. 자격 증명에 정책을 연결할 때는 정책에 `Principal` 요소를 포함하지 않습니다.
- `Action`: 수행할 수 있는 작업. the section called “Secrets Manager 작업” 섹션을 참조하세요.
- `Resource`: 액세스할 수 있는 보안 암호. the section called “Secrets Manager 리소스” 섹션을 참조하세요.

  와일드카드 문자(*)는 정책을 연결하는 대상에 따라 다른 의미를 갖습니다.

  - 보안 암호에 연결된 정책에서 *는 정책이 이 보안 암호에 적용됨을 의미합니다.
  - 자격 증명에 연결된 정책에서 *는 정책이 계정의 모든 리소스(보안 암호 포함)에 적용됨을 의미합니다.

보안 암호에 정책을 연결하려면 the section called “보안 암호에 권한 정책 연결” 섹션을 참조하세요.

정책을 자격 증명에 연결하려면 the section called “자격 증명에 권한 정책 연결” 섹션을 참조하세요.

주제

예: 개별 보안 암호 값을 검색할 수 있는 권한

보안 암호 값을 검색할 수 있는 권한을 부여하기 위해 정책을 암호 또는 자격 증명에 연결할 수 있습니다. 사용할 정책 유형을 결정하는 방법에 대한 도움말은 자격 증명 기반 정책 및 리소스 기반 정책을 참조하세요. 정책 연결 방법에 대한 자세한 내용은 the section called “보안 암호에 권한 정책 연결” 및 the section called “자격 증명에 권한 정책 연결” 섹션을 참조하세요.

다음 예시에서는 보안 암호에 대한 액세스 권한을 부여할 수 있는 두 가지 방법을 보여줍니다. 첫 번째 예시는 보안 암호에 연결할 수 있는 리소스 기반 정책입니다. 이 예시는 여러 사용자 또는 역할에 단일 보안 암호에 대한 액세스 권한을 부여하려는 경우에 유용합니다. 두 번째 예시는 IAM의 사용자 또는 역할에 연결할 수 있는 자격 증명 기반 정책입니다. 이 예시는 IAM 그룹에 대한 액세스 권한을 부여하려는 경우에 유용합니다. 배치 API 호출에서 보안 암호 그룹을 검색할 권한을 부여하려면 the section called “예: 비밀 값 그룹을 일괄적으로 검색할 수 있는 권한” 섹션을 참조하세요.

Example 하나의 보안 암호 읽기(보안 암호에 연결)

다음 정책을 보안 암호에 연결하여 보안 암호에 대한 액세스 권한을 부여할 수 있습니다. 이 정책을 사용하려면 the section called “보안 암호에 권한 정책 연결” 섹션을 참조하세요.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::AccountId:role/EC2RoleToAccessSecrets"
      },
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "*"
    }
  ]
}
```

Example 고객 관리형 키를 사용하여 암호화된 보안 암호 읽기(자격 증명에 연결)

고객 관리형 키를 사용하여 보안 암호를 암호화하는 경우 다음 정책을 자격 증명에 연결하여 보안 암호를 읽을 액세스 권한을 부여할 수 있습니다. 이 정책을 사용하려면 the section called “자격 증명에 권한 정책 연결” 섹션을 참조하세요.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "SecretARN"
    },
    {
      "Effect": "Allow",
      "Action": "kms:Decrypt",
      "Resource": "KMSKeyARN"
    }
  ]
}
```

예: 개별 비밀을 읽고 설명할 수 있는 권한

Example 비밀 한 가지 읽기 및 설명 (ID에 첨부)

자격 증명에 다음 정책을 연결하여 보안 암호에 대한 액세스 권한을 부여할 수 있습니다. 이 정책을 사용하려면 the section called “자격 증명에 권한 정책 연결” 단원을 참조하세요.

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
          "secretsmanager:GetSecretValue",
          "secretsmanager:DescribeSecret"
          ],
        "Resource": "SecretARN"
      }
    ]
  }
```

예: 비밀 값 그룹을 일괄적으로 검색할 수 있는 권한

Example 보안 암호 그룹 일괄 읽기(자격 증명에 연결)

자격 증명에 다음 정책을 연결하여 배치 API 직접 호출에서 보안 암호 그룹을 검색할 수 있는 액세스 권한을 부여할 수 있습니다. 이 정책은 배치 호출에 다른 보안 암호가 포함되어 있더라도

secretARN1, *secretaRN2* 및 *secretaRN3*으로 지정한 보안 암호만 검색할 수 있도록 호출자를 제한합니다. 호출자가 배치 API 호출에서 다른 보안 암호도 요청하는 경우 Secrets Manager는 이를 반환하지 않습니다. 자세한 내용은 을 참조하십시오 BatchGetSecretValue. . 이 정책을 사용하려면 the section called "자격 증명에 권한 정책 연결" 섹션을 참조하세요.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:BatchGetSecretValue",
        "secretsmanager:ListSecrets"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue"
      ],
      "Resource": [
        "SecretARN1",
        "SecretARN2",
        "SecretARN3"
        ]
    }
  ]
}
```

예: 와일드카드

와일드카드를 사용하여 정책 요소에 값 집합을 포함할 수 있습니다.

Example 경로의 모든 보안 암호 액세스(자격 증명에 연결)

다음 정책은 이름이 "*TestEnv/*"로 시작하는 모든 비밀을 검색할 수 있는 액세스 권한을 부여합니다. 이 정책을 사용하려면 the section called "자격 증명에 권한 정책 연결" 섹션을 참조하세요.

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

```
    "Effect": "Allow",
    "Action": "secretsmanager:GetSecretValue",
    "Resource": "arn:aws:secretsmanager:Region:AccountId:secret:TestEnv/*"
  }
}
```

Example 모든 보안 암호에 대한 메타데이터 액세스(자격 증명에 연결)

다음 정책은 DescribeSecret 및 List: ListSecrets, ListSecretVersionIds로 시작하는 권한을 부여합니다. 이 정책을 사용하려면 the section called “자격 증명에 권한 정책 연결” 섹션을 참조하세요.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "secretsmanager:DescribeSecret",
      "secretsmanager:List*"
    ],
    "Resource": "*"
  }
}
```

Example 보안 암호 이름 일치(자격 증명에 연결)

다음 정책은 보안 암호에 대한 모든 Secrets Manager 권한을 해당 이름으로 부여합니다. 이 정책을 사용하려면 the section called “자격 증명에 권한 정책 연결” 섹션을 참조하세요.

보안 암호 이름을 일치시키려면 리전, 계정 ID, 보안 암호 이름 및 와일드카드(?)를 조합하여 임의의 개별 문자와 일치시켜 보안 암호에 대한 ARN을 생성합니다. Secrets Manager는 보안 암호 이름에 6개의 임의 문자를 ARN의 일부로 추가하므로 이 와일드카드를 사용하여 해당 문자와 일치시킬 수 있습니다. "another_secret_name-*" 구문을 사용하는 경우 Secrets Manager는 6개의 임의 문자와 해당 보안 암호를 일치시킬 뿐 아니라 "another_secret_name-<anything-here>a1b2c3"과도 일치시킵니다.

6개의 임의 문자를 제외한 보안 암호 ARN의 모든 부분을 예측할 수 있기 때문에 와일드카드 문자 '??????' 구문을 사용하면 아직 존재하지 않는 보안 암호에 안전하게 권한을 부여할 수 있습니다. 하지만 보안 암호를 삭제한 후 이름이 동일한 보안 암호를 다시 생성할 경우, 임의의 문자 6자가 변경되더라도 새 보안 암호에 대한 권한이 사용자에게 자동으로 부여됩니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "secretsmanager:*",
      "Resource": [
        "arn:aws:secretsmanager:Region:AccountId:secret:a_specific_secret_name-a1b2c3",
        "arn:aws:secretsmanager:Region:AccountId:secret:another_secret_name-??????"
      ]
    }
  ]
}
```

예: 보안 암호 생성 권한

사용자에게 보안 암호 생성 권한을 부여하려면 사용자가 속한 IAM 그룹에 권한 정책을 연결하는 것이 좋습니다. IAM 사용자 그룹을 참조하세요.

Example 보안 암호 생성(자격 증명에 연결)

다음 정책은 보안 암호를 생성하고 보안 암호 목록을 볼 수 있는 권한을 부여합니다. 이 정책을 사용하려면 the section called “자격 증명에 권한 정책 연결” 섹션을 참조하세요.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:CreateSecret",
        "secretsmanager:ListSecrets"
      ],
      "Resource": "*"
    }
  ]
}
```

예: 암호를 암호화하기 위한 특정 AWS KMS 키 거부

> **Important**
>
> 고객 관리 키를 거부하려면 키 정책 또는 키 부여를 사용하여 액세스를 제한하는 것이 좋습니다. 자세한 내용은 AWS KMSAWS Key Management Service 개발자 안내서의 인증 및 액세스 제어를 참조하십시오.

Example AWS 관리 키 거부 **`aws/secretsmanager`** (ID에 연결)

다음 정책은 비밀 생성 또는 업데이트를 `aws/secretsmanager` 위한 AWS 관리 키 사용을 거부하는 방법을 보여줍니다. 즉, 고객 관리 키를 사용하여 암호를 암호화해야 합니다. `aws/secretsmanager`키가 있는 경우 키 ID도 포함해야 합니다. Secrets Manager가 이 문자열을 AWS 관리 키로 `aws/secretsmanager` 해석하므로 빈 문자열도 포함해야 합니다. 두 번째 명령문은 KMS 키가 포함되지 않은 비밀 생성 요청을 거부합니다. Secrets Manager는 KMS 키를 관리 키로 해석하기 때문입니다. AWS `aws/secretsmanager`

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Sid": "RequireCustomerManagedKeysOnSecrets",
          "Effect": "Deny",
          "Action": [
              "secretsmanager:CreateSecret",
              "secretsmanager:UpdateSecret"
          ],
          "Resource": "*",
          "Condition": {
              "ForAnyValue:StringLikeIfExists": {
                  "secretsmanager:KmsKeyId": [
                      "*alias/aws/secretsmanager",
                      "*<key_ID_of_the_AWS_managed_key>",
                      ""
                  ]
              }
          }
      },
      {
          "Sid": "RequireKmsKeyIdParameterOnCreate",
```

```
            "Effect": "Deny",
            "Action": "secretsmanager:CreateSecret",
            "Resource": "*",
            "Condition": {
                "Null": {
                    "secretsmanager:KmsKeyId": "true"
                }
            }
        }
    ]
}
```

예: 권한 및 VPC

VPC 내에서 Secrets Manager에 액세스해야 하는 경우 권한 정책에 조건을 포함시켜 Secrets Manager에 대한 요청이 VPC에서 전송되도록 할 수 있습니다. 자세한 내용은 VPC 엔드포인트 조건 및 VPC 엔드포인트 섹션을 참조하세요.

다른 AWS 서비스의 시크릿에 대한 액세스 요청도 VPC로부터 오는지 확인하십시오. 그렇지 않으면 이 정책에서 해당 서비스의 액세스가 거부됩니다.

Example VPC 엔드포인트를 통해 요청되도록 요구(보안 암호에 연결)

다음 정책은 요청이 VPC 엔드포인트 *vpce-1234a5678b9012c*를 통해 이루어질 때만 사용자가 Secrets Manager 작업을 수행할 수 있도록 허용합니다. 이 정책을 사용하려면 the section called “보안 암호에 권한 정책 연결” 섹션을 참조하세요.

```
{
  "Id": "example-policy-1",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "RestrictGetSecretValueoperation",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:sourceVpce": "vpce-1234a5678b9012c"
        }
      }
```

```
    }
  ]
}
```

Example VPC를 통해 요청되도록 요구(보안 암호에 연결)

다음 정책은 명령이 *vpc-12345678*에서 이루어진 경우에만 보안 암호를 생성 및 관리하는 명령을 허용합니다. 또한 정책에서는 요청이 vpc-2b2b2b2b에서 이루어진 경우에만 보안 암호와 암호화된 값에 액세스하는 작업을 허용합니다. 애플리케이션이 하나의 VPC에서 실행 중이지만 관리 용도로 두 번째 격리된 VPC를 사용하는 경우, 이와 같은 정책을 사용할 수 있습니다. 이 정책을 사용하려면 the section called “보안 암호에 권한 정책 연결” 섹션을 참조하세요.

```
{
  "Id": "example-policy-2",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAdministrativeActionsfromONLYvpc-12345678",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "secretsmanager:Create*",
        "secretsmanager:Put*",
        "secretsmanager:Update*",
        "secretsmanager:Delete*",
        "secretsmanager:Restore*",
        "secretsmanager:RotateSecret",
        "secretsmanager:CancelRotate*",
        "secretsmanager:TagResource",
        "secretsmanager:UntagResource"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:sourceVpc": "vpc-12345678"
        }
      }
    },
    {
      "Sid": "AllowSecretValueAccessfromONLYvpc-2b2b2b2b",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
```

```
        "secretsmanager:GetSecretValue"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:sourceVpc": "vpc-2b2b2b2b"
        }
      }
    }
  ]
}
```

예: 태그를 사용하여 보안 암호에 대한 액세스 제어

태그를 사용하여 보안 암호에 대한 액세스를 제어할 수 있습니다. 태그를 사용하여 권한을 제어하면 빠르게 성장하는 환경에서 유용하며 정책 관리가 번거로운 상황에 도움이 됩니다. 한 가지 전략은 보안 암호에 태그를 연결한 후 보안 암호에 특정 태그가 있을 때 자격 증명에 권한을 부여하는 것입니다.

Example 특정 태그를 사용하여 보안 암호에 대한 액세스 허용(자격 증명에 연결)

다음 정책은 키가 ""이고 값이 *ServerName*" *ServerABC"#* 태그가 있는 암호를 허용합니다DescribeSecret. 이 정책을 사용하려면 the section called “자격 증명에 권한 정책 연결” 섹션을 참조하세요.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "secretsmanager:DescribeSecret",
    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "secretsmanager:ResourceTag/ServerName": "ServerABC"
      }
    }
  }
}
```

예: 보안 암호의 태그와 일치하는 태그를 사용하여 자격 증명에 대한 액세스 제한

한 가지 전략은 보안 암호 및 IAM 자격 증명 모두에 태그를 연결하는 것입니다. 그런 다음 자격 증명 태그가 보안 암호의 태그와 일치할 때 작업을 허용하는 권한 정책을 생성합니다. 전체 자습서는 태그를 기반으로 보안 암호에 대한 액세스 권한 거부를 참조하세요.

태그를 사용하여 권한을 제어하면 빠르게 성장하는 환경에서 유용하며 정책 관리가 번거로운 상황에 도움이 됩니다. 자세한 내용은 AWS용 ABAC란 무엇인가요? 섹션을 참조하세요.

Example 보안 암호와 동일한 태그가 있는 역할에 대한 액세스 허용(보안 암호에 연결)

다음 정책은 태그 *AccessProject*에서 보안 암호와 역할의 값이 동일한 경우에만 계정 *123456789012*에 GetSecretValue를 부여합니다. 이 정책을 사용하려면 the section called "보안 암호에 권한 정책 연결" 단원을 참조하세요.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "AWS": "123456789012"
    },
    "Condition": {
      "StringEquals": {
        "aws:ResourceTag/AccessProject": "${ aws:PrincipalTag/AccessProject }"
      }
    },
    "Action": "secretsmanager:GetSecretValue",
    "Resource": "*"
  }
}
```

예: 서비스 보안 주체

시크릿에 연결된 리소스 정책에 AWS 서비스 주체가 포함된 경우 aws: SourceArn 및 aws: SourceAccount 글로벌 조건 키를 사용하는 것이 좋습니다. ARN 및 계정 값은 요청이 다른 AWS 서비스에서 Secrets Manager로 오는 경우에만 권한 부여 컨텍스트에 포함됩니다. 이러한 조건 조합은 잠재적 혼동된 대리자 시나리오를 방지합니다.

리소스 ARN에 리소스 정책에서 허용되지 않는 문자가 포함된 경우, 해당 리소스 ARN을 `aws:SourceArn` 조건 키의 값에 사용할 수 없습니다. 대신 `aws:SourceAccount` 조건 키를 사용합니다. 자세한 내용은 IAM 요구 사항을 참조하세요.

일반적으로 보안 주체는 암호에 연결된 정책에서 보안 주체로 사용되지 않지만 일부 AWS 서비스에서는 보안 주체가 필요합니다. 서비스에서 보안 암호에 연결하도록 요구하는 리소스 정책에 대한 자세한 내용은 서비스 설명서를 참조하세요.

Example 서비스가 서비스 보안 주체를 사용하여 보안 암호에 액세스하도록 허용(보안 암호에 연결)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "service-name.amazonaws.com"
        ]
      },
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "*",
      "Condition": {
        "ArnLike": {
          "aws:sourceArn": "arn:aws:service-name::123456789012:*"
        },
        "StringEquals": {
          "aws:sourceAccount": "123456789012"
        }
      }

    }
  ]
}
```

에 대한 권한 참조 AWS Secrets Manager

권한 정책을 구성하는 요소를 확인하려면 JSON 정책 문서 구조 및 IAM JSON 정책 요소 참조를 참조하세요.

사용자 고유의 권한 정책 작성을 시작하려면 the section called “권한 정책 예” 섹션을 참조하세요.

작업 테이블의 리소스 유형 열에는 각 작업이 리소스 수준 권한을 지원하는지 여부가 표시됩니다. 리소스 열에 값이 없으면 정책 문의 Resource 요소에서 정책이 적용되는 모든 리소스("*")를 지정해야 합니다. 리소스 열에 리소스 유형이 포함되어 있으면 해당 작업 시 문에서 해당 유형의 ARN을 지정할 수 있습니다. 작업에 필요한 리소스가 하나 이상 있는 경우, 호출자에게 해당 리소스와 함께 작업을 사용할 수 있는 권한이 있어야 합니다. 필수 리소스는 테이블에서 별표(*)로 표시됩니다. IAM 정책의 Resource 요소로 리소스 액세스를 제한하는 경우, 각 필수 리소스 유형에 대해 ARN 또는 패턴을 포함해야 합니다. 일부 작업은 다수의 리소스 유형을 지원합니다. 리소스 유형이 옵션(필수 리소스로 표시되지 않은 경우)인 경우에는 선택적 리소스 유형 중 하나를 사용하도록 선택할 수 있습니다.

작업 테이블의 조건 키 열에는 정책 설명의 Condition 요소에서 지정할 수 있는 키가 포함됩니다. 서비스의 리소스와 연결된 조건 키에 대한 자세한 내용은 리소스 유형 테이블의 조건 키 열을 참조하세요.

Secrets Manager 작업

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
BatchGetSecretValue	보안 암호의 목록을 검색 및 복호화할 수 있는 권한을 부여합니다.	나열			
CancelRotateSecret	진행 중인 보안 암호 교체를 취소할 수 있는 권한을 부여합니다.	쓰기	Secret*	secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:Res	

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				ourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
CreateSecret	쿼리 및 교체할 수 있는 암호화된 데이터를 저장하는 비밀을 만들 수 있는 권한을 부여합니다.	쓰기	Secret*		

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:Name secretsmanager:Description secretsmanager:KmsKeyId aws:RequestTag/${TagKey} aws:ResourceTag/${TagKey} aws:TagKeys secretsmanager:ResourceTag/tag-key secretsmanager:AddReplicaRegions secretsmanager:ForceOverwri	

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				teReplicaSecret	
DeleteResourcePolicy	보안 암호에 연결된 리소스 정책을 삭제할 수 있는 권한을 부여합니다.	권한 관리	Secret*		
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
DeleteSecret	보안 암호를 삭제할 수 있는 권한을 부여합니다.	쓰기	Secret*		

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:RecoveryWindowInDays secretsmanager:ForceDeleteWithoutRecovery secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
DescribeSecret	보안 암호에 대한 메타데이터를 검색할 수 있는 권한을 부여하지만 암호화된 데이터를 검색할 수 있는 권한은 부여하지 않습니다.	읽기	Secret*		
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
GetRandomPassword	암호 생성에 사용할 임의 문자열을 생성할 수 있는 권한을 부여합니다.	읽기			
GetResourcePolicy	보안 암호에 연결된 리소스 정책을 가져올 수 있는 권한을 부여합니다.	읽기	Secret*		

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
GetSecretValue	암호화된 데이터를 검색 및 복호화할 수 있는 권한을 부여합니다.	읽기	Secret*		

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:SecretId secretsmanager:VersionId secretsmanager:VersionStage secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
ListSecretVersionIds	보안 암호의 사용 가능한 버전을 나열할 수 있는 권한을 부여합니다.	읽기	Secret*		

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
ListSecrets	사용 가능한 보안 암호를 나열할 수 있는 권한을 부여합니다.	나열			
PutResourcePolicy	보안 암호에 리소스 정책을 연결할 수 있는 권한을 부여합니다.	권한 관리	Secret*		

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:BlockPublicPolicy secretsmanager:SecretPrimaryRegion	
PutSecretValue	새 암호화된 데이터를 사용하여 보안 암호의 새 버전을 업데이트할 수 있는 권한을 부여합니다.	쓰기	Secret*		

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
RemoveRegionsFromReplication	복제에서 지역을 제거할 수 있는 권한을 부여합니다.	쓰기	Secret*		

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
ReplicateSecretToRegions	기존 암호를 다중 지역 암호로 변환하고 암호를 새 지역 목록에 복제하기 시작할 수 있는 권한을 부여합니다.	쓰기	Secret*		

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion secretsmanager:AddReplicaRegions secretsmanager:ForceOverwriteReplicaSecret	

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
RestoreSecret	보안 암호의 삭제를 취소할 수 있는 권한을 부여합니다.	쓰기	Secret*		
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
RotateSecret	보안 암호의 교체을 시작할 수 있는 권한을 부여합니다.	쓰기	Secret*		

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:SecretId secretsmanager:RotationLambdaARN secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion secretsmanager:ModifyRotationRules	

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:RotateImmediately	
StopReplicationToReplica	복제에서 암호를 제거하고 해당 암호를 복제본 리전의 리전 암호로 승격할 수 있는 권한을 부여합니다.	쓰기	Secret*		
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
TagResource	보안 암호에 태그를 추가할 수 있는 권한을 부여합니다.	태그 지정	Secret*		

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:SecretId aws:RequestTag/${TagKey} aws:TagKeys secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
UntagResource	보안 암호에서 태그를 제거할 수 있는 권한을 부여합니다.	태그 지정	Secret*		

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:SecretId aws:TagKeys secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
UpdateSecret	새 메타데이터 또는 암호화된 데이터의 새 버전으로 보안 암호를 업데이트할 수 있는 권한을 부여합니다.	쓰기	Secret*		

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:SecretId secretsmanager:Description secretsmanager:KmsKeyId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
UpdateSecretVersionStage	보안 암호 간에 단계를 이동할 수 있는 권한을 부여합니다.	쓰기	Secret*		

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:SecretId secretsmanager:VersionStage secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
ValidateResourcePolicy	정책을 연결하기 전에 리소스 정책을 검증할 권한을 부여합니다.	권한 관리	Secret*		

작업	설명	액세스 레벨	리소스 유형(*필수)	조건 키	종속 작업
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	

Secrets Manager 리소스

리소스 유형	ARN	조건 키
Secret	`arn:${Partition}:secretsmanager:${Region}:${Account}:secret:${SecretId}`	aws:RequestTag/${TagKey} aws:ResourceTag/${TagKey}

리소스 유형	ARN	조건 키
		aws:TagKeys secretsmanager:ResourceTag/tag-key secretsmanager:resource/AllowRotationLambdaArn

Secrets Manager에서는 보안 암호 이름의 끝에 대시와 임의의 영숫자 6자를 추가해 보안 암호 ARN의 마지막 부분을 구성합니다. 보안 암호를 삭제한 다음 이름이 동일한 다른 보안 암호를 다시 생성할 경우 이 형식을 사용하면 Secrets Manager에서 임의의 문자 6자를 새로 생성하므로 원래 보안 암호에 대한 권한을 가진 개인이 새 보안 암호에 대한 액세스 권한을 자동으로 얻을 수 없습니다.

보안 암호 세부 정보 페이지에서 또는 `DescribeSecret`를 사용하여 Secrets Manager 콘솔의 보안 암호에 대한 ARN을 찾을 수 있습니다.

조건 키

권한 정책에 다음 표의 문자열 조건을 포함하는 경우, Secrets Manager에 대한 호출자가 일치하는 파라미터를 전달해야 합니다. 그렇지 않으면 액세스가 거부됩니다. 누락된 파라미터에 대해 호출자가 거부되는 것을 방지하려면 조건 연산자 이름의 끝에 `IfExists`를 추가합니다(예: `StringLikeIfExists`). 자세한 내용은 IAM JSON 정책 요소: 조건 연산자를 참조하세요.

조건 키	설명	유형
aws:RequestTag/${TagKey}	Secrets Manager 서비스에 대한 사용자의 요청에 있는 키를 기준으로 액세스를 필터링합니다.	String
aws:ResourceTag/${TagKey}	리소스와 연결된 태그를 기준으로 액세스를 필터링합니다.	String
aws:TagKeys	Secrets Manager 서비스에 대한 사용자의 요청에 있는 모든 태그 키 이름의 목록을 기준으로 액세스를 필터링합니다.	ArrayOfString

조건 키	설명	유형
secretsmanager:AddReplicaRegions	보안 암호를 복제할 리전 목록을 기준으로 액세스를 필터링합니다.	ArrayOfString
secretsmanager:BlockPublicPolicy	리소스 정책이 광범위한 AWS 계정 액세스를 차단하는지 여부를 기준으로 액세스를 필터링합니다.	부울
secretsmanager:Description	요청에 있는 설명 텍스트를 기준으로 액세스를 필터링합니다.	String
secretsmanager:ForceDeleteWithoutRecovery	암호를 복구 기간 없이 즉시 삭제해야 하는지 여부에 따라 액세스를 필터링합니다.	부울
secretsmanager:ForceOverwriteReplicaSecret	대상 리전에서 동일한 이름의 보안 암호를 덮어쓸지 여부를 기준으로 액세스를 필터링합니다.	부울
secretsmanager:KmsKeyId	요청에 있는 KMS 키의 ARN을 기준으로 액세스를 필터링합니다.	String
secretsmanager:ModifyRotationRules	보안 암호의 교체 규칙을 수정해야 하는지 여부에 따라 액세스를 필터링합니다.	부울
secretsmanager:Name	요청에 있는 보안 암호의 표시 이름을 기준으로 액세스를 필터링합니다.	String
secretsmanager:RecoveryWindowInDays	Secrets Manager가 보안 암호를 삭제하기 전에 대기해야 하는 기간(일)을 기준으로 액세스를 필터링합니다.	숫자

조건 키	설명	유형
secretsmanager:ResourceTag/tag-key	태그 키-값 페어를 기준으로 액세스를 필터링합니다.	String
secretsmanager:RotateImmediately	보안 암호를 즉시 교체해야 하는지 여부에 따라 액세스를 필터링합니다.	부울
secretsmanager:RotationLambdaARN	요청에 있는 교체 Lambda 함수의 ARN을 기준으로 액세스를 필터링합니다.	ARN
secretsmanager:SecretId	요청에 있는 SecretID 값을 기준으로 액세스를 필터링합니다.	ARN
secretsmanager:SecretPrimaryRegion	비밀이 생성된 기본 지역별로 액세스를 필터링합니다.	String
secretsmanager:VersionId	요청에 있는 보안 암호 버전의 고유 식별자를 기준으로 액세스를 필터링합니다.	String
secretsmanager:VersionStage	요청에 있는 버전 단계의 목록을 기준으로 액세스를 필터링합니다.	String
secretsmanager:resource/AllowRotationLambdaArn	보안 암호와 연결된 교체 Lambda 함수의 ARN을 기준으로 액세스를 필터링합니다.	ARN

`BlockPublicPolicy` 조건을 포함하는 보안 암호에 대한 광범위한 액세스 차단

`PutResourcePolicy` 작업을 허용하는 자격 증명 정책에서 `BlockPublicPolicy: true`를 사용하는 것이 좋습니다. 이 조건은 정책에서 광범위한 액세스를 허용하지 않는 경우에만 사용자가 리소스 정책을 보안 암호에 연결할 수 있음을 의미합니다.

Secrets Manager는 Zelkova 자동화된 추론을 사용하여 광범위한 액세스에 대한 리소스 정책을 분석합니다. Zelkova에 대한 자세한 내용은 보안 블로그에서 자동화된 추론을 AWS 사용하여 대규모 보안을 달성하는 방법을 참조하십시오. AWS

다음 예에서는 `BlockPublicPolicy`를 사용하는 방법을 보여줍니다.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "secretsmanager:PutResourcePolicy",
        "Resource": "SecretId",
        "Condition": {
            "Bool": {
                "secretsmanager:BlockPublicPolicy": "true"
            }
        }
    }
}
```

IP 주소 조건

그러나 Secrets Manager에 대한 액세스를 허용하거나 거부하는 정책문에서 IP 주소 조건 연산자 또는 `aws:SourceIp` 조건 키를 지정할 때는 주의해야 합니다 예를 들어 기업 네트워크 IP 주소 범위의 요청에 대한 AWS 작업을 비밀로 제한하는 정책을 추가하면 기업 네트워크에서 요청을 호출하는 IAM 사용자의 요청이 예상대로 작동합니다. 그러나 Lambda 함수로 교체를 활성화하는 경우와 같이 다른 서비스가 사용자를 대신하여 암호에 액세스할 수 있도록 하는 경우 해당 함수는 -internal 주소 공간에서 Secrets Manager 작업을 호출합니다. AWS IP 주소 필터가 있는 정책의 영향을 받는 요청은 실패합니다.

또한 요청이 Amazon VPC 엔드포인트에서 이루어지는 경우 `aws:sourceIP` 조건 키는 유효하지 않습니다. 요청을 특정 VPC 엔드포인트로 제한하려면 the section called “VPC 엔드포인트 조건”를 사용합니다.

VPC 엔드포인트 조건

특정 VPC 또는 VPC 엔드포인트의 요청으로 액세스를 허용하거나 거부하려면 `aws:SourceVpc`를 사용하여 지정된 VPC 요청으로 액세스를 제한하거나 `aws:SourceVpce`를 사용하여 지정된 VPC 엔드포인트의 요청으로 액세스를 제한합니다. the section called “예: 권한 및 VPC” 단원을 참조하세요.

- `aws:SourceVpc`는 지정된 VPC의 요청으로 액세스를 제한합니다.
- `aws:SourceVpce`는 지정된 VPC 엔드포인트의 요청으로 액세스를 제한합니다.

Secrets Manager 보안 암호에 대한 액세스를 허용하거나 거부하는 보안 암호 정책문에 이러한 조건 키를 사용하면 사용자를 대신해 Secrets Manager를 사용하여 보안 암호에 액세스하는 서비스에 대한 액세스를 실수로 거부하게 될 수 있습니다. 일부 AWS 서비스만 VPC 내에서 엔드포인트를 사용하여 실행할 수 있습니다. 보안 암호에 대한 요청을 VPC 또는 VPC 엔드포인트로 제한하면 구성되지 않은 서비스로부터의 Secrets Manager 호출에 실패할 수 있습니다.

VPC 엔드포인트을 참조하세요.

다음을 사용하여 시크릿 생성 및 관리 AWS Secrets Manager

보안 암호는 암호, 사용자 이름 및 암호와 같은 자격 증명 집합, OAuth 토큰 또는 Secrets Manager에 암호화된 형식으로 저장하는 기타 비밀 정보일 수 있습니다.

주제

AWS Secrets Manager 데이터베이스 시크릿 생성

Amazon RDS, Amazon Aurora, Amazon Redshift 또는 Amazon DocumentDB에 사용자를 생성한 후 다음 단계를 따라 Secrets Manager에 보안 인증을 저장할 수 있습니다. AWS CLI 또는 SDK 중 하나를 사용하여 암호를 저장하는 경우 올바른 JSON 구조로 암호를 제공해야 합니다. 콘솔을 사용하여 데이터베이스 보안 암호를 저장할 때 Secrets Manager는 올바른 JSON 구조로 보안 암호를 자동 생성합니다.

> **Tip**
>
> Amazon RDS 및 Amazon Redshift 관리자 사용자 자격 증명의 경우 관리형 암호를 사용하는 것이 좋습니다. 관리 서비스를 통해 관리 암호를 생성한 다음 관리 순환을 사용할 수 있습니다.

다른 리전으로 복제된 소스 데이터베이스에 대한 데이터베이스 보안 인증 정보를 저장하면 보안 암호에 해당 소스 데이터베이스에 대한 연결 정보가 포함됩니다. 그 후에 보안 암호를 복제하면 복제본은 소스 보안 암호의 복사본이 되며 동일한 연결 정보를 포함합니다. 리전 연결 정보를 위해 보안 암호에 추가로 키/값 쌍을 추가할 수 있습니다.

암호를 만들려면 에서 부여한 SecretsManagerReadWriteAWS 관리형 정책권한이 필요합니다.

시크릿 매니저는 시크릿을 생성할 때 CloudTrail 로그 항목을 생성합니다. 자세한 정보는 the section called “다음과 같이 로그하십시오. AWS CloudTrail ”을 참조하세요.

보안 암호(콘솔) 생성

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. Store a new secret(새 보안 암호 저장)을 선택합니다.
3. 보안 암호 유형 선택(Choose secret type) 페이지에서 다음을 수행합니다.
   a. 보안 암호 유형(Secret type)에서 저장할 데이터베이스 자격 증명 유형을 선택합니다.
      - Amazon RDS 데이터베이스(Aurora 포함)
      - Amazon DocumentDB 데이터베이스
      - 아마존 Redshift 데이터 웨어하우스
   b. 자격 증명(Credentials)에서 데이터베이스에 대한 자격 증명을 입력합니다.
   c. 암호화 키의 경우 Secrets AWS KMS key Manager가 보안 값을 암호화하는 데 사용하는 키를 선택합니다. 자세한 정보는 보안 암호 암호화 및 복호화을 참조하세요.
      - 대부분의 경우 Secrets Manager에 대한 AWS 관리형 키 를 사용하려면 aws/secretsmanager를 선택합니다. 이 키를 사용하는 데 드는 비용은 없습니다.
      - 다른 AWS 계정사람의 비밀에 액세스해야 하거나 자체 KMS 키를 사용하여 교체하거나 키 정책을 적용하려는 경우 목록에서 고객 관리 키를 선택하거나 Add new key를 선택하여 새로 만드십시오. 고객 관리형 키 사용 비용에 대한 자세한 내용은 요금을 참조하세요.

        the section called “KMS 키에 대한 권한”이(가) 있어야 합니다. 크로스 계정 액세스에 대한 자세한 내용은 the section called “크로스 계정 액세스” 섹션을 참조하세요.
   d. 데이터베이스에서 데이터베이스(Database)를 선택합니다.
   e. Next(다음)를 선택합니다.
4. 보안 구성(Configure secret) 페이지에서 다음을 수행합니다.

a. 설명이 포함된 Secret name(보안 암호 이름)과 Description(설명)을 입력합니다. 보안 암호 이름은 1~512자의 유니코드 문자를 포함해야 합니다.

b. (선택 사항) Tags(태그) 섹션에서 보안 암호에 태그를 추가합니다. 태깅 전략에 대한 자세한 내용은 the section called " 보안 암호 태그 지정" 단원을 참조하세요. 민감한 정보는 암호화되지 않으므로 태그에 저장하지 마세요.

c. (선택 사항) 리소스 권한(Resource permissions)에서 리소스 정책을 보안 암호에 추가하려면 권한 편집(Edit permissions)을 선택합니다. 자세한 정보는 the section called "보안 암호에 권한 정책 연결"을 참조하세요.

d. (선택 사항) 암호 복제에서 암호를 다른 AWS 리전암호로 복제하려면 암호 복제를 선택합니다. 보안 암호를 지금 복제하거나 페이지로 다시 돌아와서 나중에 복제할 수 있습니다. 자세한 정보는 지역 간 비밀 복제을 참조하세요.

e. Next(다음)를 선택합니다.

5. (선택 사항) 교체 구성(Configure rotation) 페이지에서 자동 교체를 켤 수 있습니다. 현재 교체를 끈 다음 나중에 켤 수도 있습니다. 자세한 정보는 보안 암호 교체을(를) 참조하세요. 다음을 선택하세요.

6. Review(검토) 페이지에서 보안 암호 세부 정보를 검토한 후 Store(저장)를 선택합니다.

   Secrets Manager는 보안 암호 목록으로 돌아갑니다. 암호가 표시되지 않으면 Refresh(새로 고침)를 선택합니다.

AWS CLI

명령 셸에 명령을 입력하면 명령 기록이 액세스되거나 유틸리티가 명령 파라미터에 액세스할 위험이 있습니다. the section called "AWS Secrets Manager 보안 암호 저장 시 AWS CLI 사용으로 발생 가능한 위험 줄이기"를 참조하세요.

Example JSON 파일의 보안 인증 정보로 보안 암호 만들기

다음 `create-secret` 예시에서는 파일의 보안 인증 정보를 사용하여 보안 암호를 만듭니다. 자세한 내용은 사용 설명서의 AWS CLI 파일에서 AWS CLI 매개변수 로드를 참조하십시오.

Secrets Manager에서 보안 암호를 교체할 수 있게 하려면 JSON이 보안 암호의 JSON 구조에 일치해야 합니다.

```
aws secretsmanager create-secret \
    --name MyTestSecret \
```

```
    --secret-string file://mycreds.json
```

mycreds.json의 콘텐츠:

```
{
    "engine": "mysql",
    "username": "saanvis",
    "password": "EXAMPLE-PASSWORD",
    "host": "my-database-endpoint.us-west-2.rds.amazonaws.com",
    "dbname": "myDatabase",
    "port": "3306"
}
```

AWS SDK

AWS SDK 중 하나를 사용하여 시크릿을 생성하려면 액션을 사용하세요. `CreateSecret` 자세한 내용은 the section called “AWS SDK”을(를) 참조하세요.

JSON 시크릿 구조 AWS Secrets Manager

텍스트 또는 이진수를 Secrets Manager 암호에 저장할 수 있습니다. Secrets Manager의 보안 암호에 자동 교체를 활성화하려면 올바른 JSON 구조여야 합니다. 교체하는 동안 Secrets Manager는 보안 암호의 정보를 사용하여 보안 인증 소스에 연결하고 해당 보안 인증 정보를 업데이트합니다. JSON 키 이름은 대소문자를 구분합니다.

콘솔을 사용하여 데이터베이스 보안 암호를 저장할 때 Secrets Manager는 올바른 JSON 구조로 된 보안 암호를 자동으로 생성합니다.

데이터베이스 보안 암호 등 보안 암호에 더 많은 키/값 쌍을 추가하여 다른 리전의 복제본 데이터베이스에 대한 연결 정보를 포함할 수 있습니다.

주제

- Amazon RDS Db2 보안 암호 구조
- Amazon RDS MariaDB 보안 암호 구조
- Amazon RDS 및 Amazon Aurora MySQL 보안 암호 구조
- Amazon RDS Oracle 보안 암호 구조
- Amazon RDS 및 Amazon Aurora PostgreSQL 보안 암호 구조
- Amazon RDS Microsoft SQLServer 보안 암호 구조

- Amazon DocumentDB 보안 암호 구조
- Amazon Redshift 보안 구조
- Amazon Redshift 서버리스 비밀 구조
- 아마존 ElastiCache 비밀 구조
- 액티브 디렉터리 비밀 구조

Amazon RDS Db2 보안 암호 구조

Amazon RDS Db2 인스턴스의 경우 사용자가 자신의 암호를 변경할 수 없으므로 별도의 보안 암호로 관리자 자격 증명을 제공해야 합니다.

```
{
  "engine": "db2",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 3306>,
  "masterarn": "<the ARN of the elevated secret>"
}
```

Amazon RDS MariaDB 보안 암호 구조

```
{
  "engine": "mariadb",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 3306>
}
```

를 사용하려면 관리자 또는 the section called “대체 사용자” 수퍼유저 자격 증명이 포함된 `masterarn` 비밀번호에 를 포함해야 합니다.

```
{
  "engine": "mariadb",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
```

```
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 3306>,
  "masterarn": "<the ARN of the elevated secret>"
}
```

Amazon RDS 및 Amazon Aurora MySQL 보안 암호 구조

```
{
  "engine": "mysql",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 3306>
}
```

를 사용하려면 관리자 또는 수퍼유저 자격 증명이 포함된 masterarn 비밀번호에 를 포함해야 합니다. the section called “대체 사용자”

```
{
  "engine": "mysql",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 3306>,
  "masterarn": "<the ARN of the elevated secret>"
}
```

Amazon RDS Oracle 보안 암호 구조

```
{
  "engine": "oracle",
  "host": "<required: instance host name/resolvable DNS name>",
  "username": "<required: username>",
  "password": "<required: password>",
  "dbname": "<required: database name>",
  "port": <optional: TCP port number. If not specified, defaults to 1521>
}
```

를 사용하려면 관리자 또는 수퍼유저 자격 증명이 포함된 masterarn 비밀번호에 를 포함해야 합니다. the section called “대체 사용자”

```
{
  "engine": "oracle",
  "host": "<required: instance host name/resolvable DNS name>",
  "username": "<required: username>",
  "password": "<required: password>",
  "dbname": "<required: database name>",
  "port": <optional: TCP port number. If not specified, defaults to 1521>,
  "masterarn": "<the ARN of the elevated secret>"
}
```

Amazon RDS 및 Amazon Aurora PostgreSQL 보안 암호 구조

```
{
  "engine": "postgres",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to 'postgres'>",
  "port": <TCP port number. If not specified, defaults to 5432>
}
```

를 사용하려면 관리자 또는 수퍼유저 자격 증명이 포함된 masterarn 비밀번호에 를 포함해야 합니다. the section called “대체 사용자”

```
{
  "engine": "postgres",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to 'postgres'>",
  "port": <TCP port number. If not specified, defaults to 5432>,
  "masterarn": "<the ARN of the elevated secret>"
}
```

Amazon RDS Microsoft SQLServer 보안 암호 구조

```
{
```

```
  "engine": "sqlserver",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to 'master'>",
  "port": <TCP port number. If not specified, defaults to 1433>
}
```

를 사용하려면 관리자 또는 수퍼유저 자격 증명이 포함된 masterarn 비밀번호에 를 포함해야 합니다. the section called “대체 사용자”

```
{
  "engine": "sqlserver",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to 'master'>",
  "port": <TCP port number. If not specified, defaults to 1433>,
  "masterarn": "<the ARN of the elevated secret>"
}
```

Amazon DocumentDB 보안 암호 구조

```
{
  "engine": "mongo",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 27017>,
  "ssl": <true|false. If not specified, defaults to false>
}
```

를 사용하려면 관리자 또는 수퍼유저 자격 증명이 포함된 masterarn 비밀번호에 를 포함해야 합니다. the section called “대체 사용자”

```
{
  "engine": "mongo",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
```

```
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 27017>,
  "masterarn": "<the ARN of the elevated secret>",
  "ssl": <true|false. If not specified, defaults to false>
}
```

Amazon Redshift 보안 구조

```
{
  "engine": "redshift",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 5439>
}
```

를 사용하려면 관리자 또는 수퍼유저 자격 증명이 포함된 masterarn 비밀번호에 를 포함해야 합니다. the section called “대체 사용자”

```
{
  "engine": "redshift",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 5439>,
  "masterarn": "<the ARN of the elevated secret>"
}
```

Amazon Redshift 서버리스 비밀 구조

```
{
  "engine": "redshift",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "namespaceName": <namespace name>,
  "port": <TCP port number. If not specified, defaults to 5439>
}
```

를 사용하려면 관리자 또는 수퍼유저 자격 증명이 포함된 masterarn 암호에 를 포함해야 합니다. the section called “대체 사용자”

```
{
  "engine": "redshift",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "namespaceName": <namespace name>,
  "port": <TCP port number. If not specified, defaults to 5439>,
  "masterarn": "<the ARN of the elevated secret>"
}
```

아마존 ElastiCache 비밀 구조

```
{
  "password": "<password>",
  "username": "<username>"
  "user_arn": "ARN of the Amazon EC2 user"
}
```

자세한 내용은 Amazon ElastiCache 사용 설명서의 사용자 암호 자동 교체를 참조하십시오.

액티브 디렉터리 비밀 구조

AWS Directory Service 비밀을 사용하여 Active Directory 자격 증명을 저장합니다. 자세한 내용은 관리 안내서의 관리형 AD Active Directory에 Amazon EC2 Linux 인스턴스를 원활하게 조인하는 섹션을 참조하십시오.AWS Directory Service 원활한 도메인 가입을 위해서는 다음 예제의 키 이름이 필요합니다. 원활한 도메인 조인을 사용하지 않는 경우 회전 함수 템플릿 코드에 설명된 대로 환경 변수를 사용하여 시크릿의 키 이름을 변경할 수 있습니다.

Active Directory 암호를 교체하려면 Active Directory 순환 템플릿을 사용할 수 있습니다.

액티브 디렉터리 자격 증명 비밀 구조

```
{
  "awsSeamlessDomainUsername": "<username>",
  "awsSeamlessDomainPassword": "<password>"
}
```

암호를 교체하려면 도메인 디렉터리 ID를 포함해야 합니다.

```
{
  "awsSeamlessDomainDirectoryId": "d-12345abc6e",
  "awsSeamlessDomainUsername": "<username>",
  "awsSeamlessDomainPassword": "<password>"
}
```

암호를 키탭이 포함된 암호와 함께 사용하는 경우 keytab 암호 ARN을 포함합니다.

```
{
  "awsSeamlessDomainDirectoryId": "d-12345abc6e",
  "awsSeamlessDomainUsername": "<username>",
  "awsSeamlessDomainPassword": "<password>",
  "directoryServiceSecretVersion": 1,
  "schemaVersion": "1.0",
  "keytabArns": [
    "<ARN of child keytab secret 1>,
    "<ARN of child keytab secret 2>,
    "<ARN of child keytab secret 3>,
  ],
  "lastModifiedDateTime": "2021-07-19 17:06:58"
}
```

액티브 디렉터리 키탭 비밀 구조

키탭 파일을 사용하여 Amazon EC2에서 Active Directory 계정을 인증하는 방법에 대한 자세한 내용은 Amazon Linux 2에서 SQL Server 2017을 사용한 Active Directory 인증 배포 및 구성을 참조하십시오.

```
{
  "awsSeamlessDomainDirectoryId": "d-12345abc6e",
  "schemaVersion": "1.0",
  "name": "< name>",
  "principals": [
    "aduser@MY.EXAMPLE.COM",
    "MSSQLSvc/test:1433@MY.EXAMPLE.COM"
  ],
  "keytabContents": "<keytab>",
  "parentSecretArn": "<ARN of parent secret>",
  "lastModifiedDateTime": "2021-07-19 17:06:58"
  "version": 1
}
```

AWS Secrets Manager 시크릿 생성

Secrets Manager에 API 키, 액세스 토큰, 데이터베이스용이 아닌 자격 증명, 기타 보안 암호를 저장하려면 다음 단계를 따릅니다. Amazon ElastiCache 암호의 경우 로테이션을 활성화하려면 예상 JSON 구조에 암호를 저장해야 합니다.

시크릿을 생성하려면 에서 부여한 권한이 필요합니다. SecretsManagerReadWriteAWS 관리형 정책

시크릿 매니저는 시크릿을 생성할 때 CloudTrail 로그 항목을 생성합니다. 자세한 정보는 the section called “다음과 같이 로그하십시오. AWS CloudTrail ”을 참조하세요.

보안 암호(콘솔) 생성

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. Store a new secret(새 보안 암호 저장)을 선택합니다.
3. 보안 암호 유형 선택(Choose secret type) 페이지에서 다음을 수행합니다.
   a. 보안 암호 유형(Secret type)에서 다른 유형의 보안 암호(Other type of secret)를 선택합니다.
   b. 키/값 페어에서, JSON 키/값 페어에 보안 암호를 입력하거나 일반 텍스트 탭을 클릭하고 원하는 형식으로 보안 암호를 입력합니다. 보안 암호에는 최대 65,536바이트까지 저장할 수 있습니다. 다음은 일부 예입니다.

API 키 키/값 쌍:

ClientID: ***my_client_id***

ClientSecret : ***bPxRfiWJALR*** XU TN FEMI/K7M ENG/ CY 예제 키

보안 인증 정보 키/값 쌍:

Username: *saanvis*

Password: *EXAMPLE-PASSWORD*

OAuth 토큰 일반 텍스트:

AKIAI44QH8DHBEXAMPLE

API 키 키/값 쌍:

ClientID: ***my_client_id***

ClientSecret : ***bPxRfiWJALR*** XU TN FEMI/K7M ENG/ CY 예제 키

디지털 인증서 일반 텍스트:

```
-----BEGIN CERTIFICATE-----
EXAMPLE
-----END CERTIFICATE-----
```

프라이빗 키 일반 텍스트:

```
----BEGIN PRIVATE KEY ---
EXAMPLE
---- END PRIVATE KEY ----
```

c. 암호화 키의 경우 Secrets AWS KMS key Manager가 보안 값을 암호화하는 데 사용하는 키를 선택합니다. 자세한 정보는 보안 암호 암호화 및 복호화을 참조하세요.

- 대부분의 경우 Secrets Manager에 대한 AWS 관리형 키 를 사용하려면 aws/secretsmanager를 선택합니다. 이 키를 사용하는 데 드는 비용은 없습니다.
- 다른 AWS 계정사람의 비밀에 액세스해야 하거나 자체 KMS 키를 사용하여 교체하거나 키 정책을 적용하려는 경우 목록에서 고객 관리 키를 선택하거나 Add new key를 선택하여 새로 만드십시오. 고객 관리형 키 사용 비용에 대한 자세한 내용은 요금을 참조하세요.

  the section called “KMS 키에 대한 권한”이(가) 있어야 합니다. 크로스 계정 액세스에 대한 자세한 내용은 the section called “크로스 계정 액세스” 섹션을 참조하세요.

d. Next(다음)를 선택합니다.

4. 보안 구성(Configure secret) 페이지에서 다음을 수행합니다.

a. 설명이 포함된 Secret name(보안 암호 이름)과 Description(설명)을 입력합니다. 보안 암호 이름은 1~512자의 유니코드 문자를 포함해야 합니다.

b. (선택 사항) Tags(태그) 섹션에서 보안 암호에 태그를 추가합니다. 태깅 전략에 대한 자세한 내용은 the section called “ 보안 암호 태그 지정” 단원을 참조하세요. 민감한 정보는 암호화되지 않으므로 태그에 저장하지 마세요.

c. (선택 사항) 리소스 권한(Resource permissions)에서 리소스 정책을 보안 암호에 추가하려면 권한 편집(Edit permissions)을 선택합니다. 자세한 정보는 the section called “보안 암호에 권한 정책 연결”을 참조하세요.

d. (선택 사항) 암호 복제에서 암호를 다른 AWS 리전암호로 복제하려면 암호 복제를 선택합니다. 보안 암호를 지금 복제하거나 페이지로 다시 돌아와서 나중에 복제할 수 있습니다. 자세한 정보는 지역 간 비밀 복제을 참조하세요.

e. Next(다음)를 선택합니다.

5. (선택 사항) 교체 구성(Configure rotation) 페이지에서 자동 교체를 켤 수 있습니다. 현재 교체를 끈 다음 나중에 켤 수도 있습니다. 자세한 정보는 보안 암호 교체을(를) 참조하세요. 다음을 선택하세요.

6. Review(검토) 페이지에서 보안 암호 세부 정보를 검토한 후 Store(저장)를 선택합니다.

   Secrets Manager는 보안 암호 목록으로 돌아갑니다. 암호가 표시되지 않으면 Refresh(새로 고침)를 선택합니다.

AWS CLI

명령 셸에 명령을 입력하면 명령 기록이 액세스되거나 유틸리티가 명령 파라미터에 액세스할 위험이 있습니다. the section called “AWS Secrets Manager 보안 암호 저장 시 AWS CLI 사용으로 발생 가능한 위험 줄이기”를 참조하세요.

Example 보안 암호 생성

다음 `create-secret` 예시에서는 두 개의 키-값 쌍으로 보안 암호를 만듭니다.

```
aws secretsmanager create-secret \
    --name MyTestSecret \
    --description "My test secret created with the CLI." \
    --secret-string "{\"user\":\"diegor\",\"password\":\"EXAMPLE-PASSWORD\"}"
```

Example JSON 파일의 보안 인증 정보로 보안 암호 만들기

다음 `create-secret` 예시에서는 파일의 보안 인증 정보를 사용하여 보안 암호를 만듭니다. 자세한 내용은 사용 설명서의 AWS CLI 파일에서 AWS CLI 매개변수 로드를 참조하십시오.

```
aws secretsmanager create-secret \
    --name MyTestSecret \
    --secret-string file://mycreds.json
```

mycreds.json의 콘텐츠:

```
{
    "username": "diegor",
    "password": "EXAMPLE-PASSWORD"
}
```

AWS SDK

AWS SDK 중 하나를 사용하여 시크릿을 생성하려면 액션을 사용하세요. `CreateSecret` 자세한 내용은 the section called “AWS SDK”을(를) 참조하세요.

AWS Secrets Manager보안 암호 값 업데이트

콘솔, CLI 또는 SDK를 사용하여 보안 암호 값을 업데이트할 수 있습니다. 보안 암호 값을 업데이트하면 Secrets Manager는 스테이징 레이블 `AWSCURRENT`을(를) 사용하여 보안 암호의 새 버전을 생성합니다. 레이블 `AWSPREVIOUS`이(가) 있는 이전 버전에는 계속 액세스할 수 있습니다. 자체 레이블을 추가할 수도 있습니다. 자세한 내용은 Secrets Manager 버저닝을 참조하세요.

보안 암호 값 업데이트(콘솔)

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호 목록에서 보안 암호를 선택합니다.
3. 보안 암호 세부 정보 페이지의 개요 탭에 있는 보안 암호 값 섹션에서 보안 암호 값 검색을 선택한 다음 편집을 선택합니다.

AWS CLI

보안 암호 값 업데이트(AWS CLI)

- 명령 셸에 명령을 입력하면 명령 기록이 액세스되거나 유틸리티가 명령 파라미터에 액세스할 위험이 있습니다. the section called “AWS Secrets Manager 보안 암호 저장 시 AWS CLI 사용으로 발생 가능한 위험 줄이기” 섹션을 참조하세요.

  다음 `put-secret-value`에서는 두 개의 키-값 쌍으로 새 버전의 보안 암호를 만듭니다.

  ```
  aws secretsmanager put-secret-value \
        --secret-id MyTestSecret \
  ```

```
      --secret-string "{\"user\":\"diegor\",\"password\":\"EXAMPLE-PASSWORD\"}"
```

다음 put-secret-value은(는) 사용자 지정 스테이징 레이블이 있는 새 버전을 생성합니다. 새 버전에는 MyLabel 및 AWSCURRENT 레이블이 있습니다.

```
aws secretsmanager put-secret-value \
      --secret-id MyTestSecret \
      --secret-string "{\"user\":\"diegor\",\"password\":\"EXAMPLE-PASSWORD\"}"
      --version-stages "MyLabel"
```

AWS SDK

10분마다 두 번 이상 지속적으로 PutSecretValue 또는 UpdateSecret을 호출하지 않는 것이 좋습니다. PutSecretValue 또는 UpdateSecret을 호출하여 보안 암호 값을 업데이트하면 Secrets Manager는 새 버전의 보안 암호를 생성합니다. Secrets Manager는 100개를 넘는 버전이 있을 때 레이블이 지정되지 않은 버전을 제거하지만 24시간 이내에 생성된 버전은 제거하지 않습니다. 10분마다 두 번 이상 보안 암호 값을 업데이트하면 Secrets Manager에서 제거하는 버전보다 더 많은 버전이 생성되고 보안 암호 버전 할당량에 도달하게 됩니다.

보안 암호 값을 업데이트하려면 UpdateSecret 또는 PutSecretValue을(를) 사용하세요. 자세한 내용은 the section called “AWS SDK” 섹션을 참조하세요.

Secrets Manager를 사용하여 비밀번호 생성하기

Secrets Manager를 사용하는 일반적인 패턴은 Secrets Manager에서 비밀번호를 생성한 다음 데이터베이스나 서비스에서 해당 비밀번호를 사용하는 것입니다. 다음 방법을 사용하여 이 작업을 수행할 수 있습니다.

- AWS CloudFormation — 을 참조하십시오AWS CloudFormation.
- AWS CLI — 보세요 get-random-password.
- AWS SDK — 참조하십시오 GetRandomPassword.

시크릿을 이전 버전으로 롤백하세요

를 사용하여 비밀 버전에 첨부된 레이블을 이동하여 암호를 이전 버전으로 되돌릴 수 있습니다. AWS CLI Secrets Manager가 보안 버전을 저장하는 방법에 대한 자세한 내용은 을 참조하십시오the section called “시크릿 버전”.

다음 update-secret-version-stage예제에서는 AWSCURRENT 스테이징 레이블을 암호의 이전 버전으로 이동합니다. 이렇게 하면 암호가 이전 버전으로 되돌아갑니다. 이전 버전의 ID를 찾으려면 Secrets Manager 콘솔에서 버전을 list-secret-version-ids사용하거나 확인하십시오.

이 예제에서 레이블이 있는 버전은 A1B2C3D4-5678-90AB-CDEF-Example11111이고 AWSCURRENT 레이블이 있는 버전은 A1B2C3D4-5678-90AB-CDEF-Example22222입니다. AWSPREVIOUS 이 AWSCURRENT 예제에서는 레이블을 버전 11111에서 22222로 이동합니다. 버전에서 AWSCURRENT 레이블이 제거되었으므로 AWSPREVIOUS 레이블이 해당 버전 (11111) 으로 update-secret-version-stage 자동으로 이동합니다. 그 결과 AWSCURRENT 및 AWSPREVIOUS 버전이 교체됩니다.

```
aws secretsmanager update-secret-version-stage \
  --secret-id MyTestSecret \
  --version-stage AWSCURRENT \
  --move-to-version-id a1b2c3d4-5678-90ab-cdef-EXAMPLE22222 \
  --remove-from-version-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

AWS Secrets Manager 시크릿의 암호화 키 변경

Secrets Manager는 AWS KMS 키와 데이터 키가 포함된 봉투 암호화를 사용하여 각 비밀 값을 보호합니다. 각 보안 암호에 사용할 KMS 키를 선택할 수 있습니다. AWS 관리형 키 AWS/secretsmanager를 사용하거나 고객 관리형 키를 사용할 수 있습니다. 대부분의 경우 aws/secretsmanager 사용을 권장하며, 이를 사용하는 데 드는 비용은 없습니다. 다른 AWS 계정사람의 비밀번호에 액세스해야 하거나 자체 KMS 키를 사용하여 교체하거나 키 정책을 적용하려는 경우에는 a를 사용하십시오. 고객 관리형 키the section called “KMS 키에 대한 권한”이(가) 있어야 합니다. 고객 관리형 키 사용 비용에 대한 자세한 내용은 요금을 참조하세요.

보안 암호에 대한 암호화 키를 변경할 수 있습니다. 예를 들어, 암호가 현재 AWS 관리 키를 aws/secretsmanager 사용하여 암호화되어 있는데 다른 계정에서 비밀에 액세스하려는 경우 고객 관리형 키 a로 전환할 수 있습니다.

> **Tip**
>
> 비밀번호를 고객 관리형 키교체하려는 경우 AWS KMS 자동 키 교체를 사용하는 것이 좋습니다. 자세한 내용은 AWS KMS 키 회전을 참조하십시오.

암호화 키를 변경하면 Secrets Manager가 다시 암호화하고 AWSCURRENT 새 키로 AWSPREVIOUS 버전을 변경합니다. AWSPENDING Secrets Manager는 사용자가 비밀 정보에 접근할 수 없도

록 기존 버전을 모두 이전 키로 암호화한 상태로 유지합니다. 즉, 이전 키 또는 새 키를 사용하여 AWSCURRENTAWSPENDING, 및 AWSPREVIOUS 버전을 해독할 수 있습니다.

새 암호화 키로만 암호를 AWSCURRENT 해독할 수 있도록 하려면 새 키로 새 버전의 암호를 만드십시오. 그러면 AWSCURRENT 보안 버전을 해독할 수 있으려면 새 키에 대한 권한이 있어야 합니다.

이전 암호화 키를 비활성화하면 AWSCURRENT, AWSPENDING 및 AWSPREVIOUS을(를) 제외한 암호 버전을 해독할 수 없습니다. 레이블이 지정된 다른 보안 암호 버전에 계속 액세스하려는 경우 the section called “AWS CLI”을(를) 사용하여 새 암호화 키로 해당 버전을 다시 생성해야 합니다.

보안 암호에 대한 암호화 키 변경(콘솔)

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호 목록에서 보안 암호를 선택합니다.
3. 보안 암호 세부 정보 페이지의 보안 암호 세부 정보 섹션에서 조치를 선택한 후 암호화 키 편집을 선택합니다.

AWS CLI

보안 암호의 암호화 키를 변경한 다음 이전 암호화 키를 비활성화하면 AWSCURRENT, AWSPENDING 및 AWSPREVIOUS을(를) 제외한 보안 암호 버전을 해독할 수 없습니다. 레이블이 지정된 다른 보안 암호 버전에 계속 액세스하려는 경우 the section called “AWS CLI”을(를) 사용하여 새 암호화 키로 해당 버전을 다시 생성해야 합니다.

보안 암호에 대한 암호화 키 변경(AWS CLI)

1. 다음 `update-secret` 예에서는 보안 암호 값을 암호화하는 데 사용되는 KMS 키를 업데이트합니다. KMS 키는 보안 암호와 동일한 리전에 있어야 합니다.

```
aws secretsmanager update-secret \
      --secret-id MyTestSecret \
      --kms-key-id arn:aws:kms:us-west-2:123456789012:key/EXAMPLE1-90ab-cdef-fedc-
ba987EXAMPLE
```

2. (선택 사항) 사용자 지정 레이블이 있는 보안 암호 버전이 있는 경우 새 키를 사용하여 다시 암호화하려면 해당 버전을 다시 만들어야 합니다.

명령 셸에 명령을 입력하면 명령 기록이 액세스되거나 유틸리티가 명령 파라미터에 액세스할 위험이 있습니다. the section called “AWS Secrets Manager 보안 암호 저장 시 AWS CLI 사용으로 발생 가능한 위험 줄이기” 섹션을 참조하십시오.

a. 보안 암호 버전의 값을 가져옵니다.

```
aws secretsmanager get-secret-value \
      --secret-id MyTestSecret \
      --version-stage MyCustomLabel
```

보안 암호 값을 기록해 둡니다.

b. 해당 값으로 새 버전을 생성합니다.

```
aws secretsmanager put-secret-value \
    --secret-id testDescriptionUpdate \
    --secret-string "SecretValue" \
    --version-stages "MyCustomLabel"
```

AWS Secrets Manager 시크릿 수정

보안 암호를 만든 사람에 따라 보안 암호가 생성된 후에 보안 암호의 메타데이터를 수정할 수 있습니다. 다른 서비스에서 생성한 보안 암호의 경우 다른 서비스를 사용하여 암호를 업데이트하거나 교체해야 할 수 있습니다.

보안 암호 관리자를 결정하기 위해 보안 암호 이름을 검토할 수 있습니다. 다른 서비스에서 관리하는 보안 암호는 해당 서비스의 ID가 접두사로 붙습니다. 또는 에서 describe-secret를 호출한 다음 필드를 검토하십시오. AWS CLI`OwningService` 자세한 정보는 관리형 시크릿을 참조하세요.

관리하는 보안 암호에 대해 설명, 리소스 기반 정책, 암호화 키, 태그를 수정할 수 있습니다. 암호화된 보안 암호 값을 변경할 수도 있지만 교체를 사용하여 자격 증명을 포함하는 보안 암호 값을 업데이트하는 것이 좋습니다. 교체를 하면 Secrets Manager의 보안 암호와 데이터베이스 또는 서비스의 자격 증명이 모두 업데이트됩니다. 이렇게 하면 클라이언트가 보안 암호 값을 요청할 때 항상 작업 중인 자격 증명 세트를 가져오도록 보안 암호가 자동으로 동기화됩니다. 자세한 정보는 보안 암호 교체을 참조하세요.

Secrets Manager는 암호를 수정할 때 CloudTrail 로그 항목을 생성합니다. 자세한 정보는 the section called “다음과 같이 로그하십시오. AWS CloudTrail ”을 참조하세요.

보안 암호(콘솔)를 업데이트하려면

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호 목록에서 보안 암호를 선택합니다.
3. 보안 암호 세부 정보 페이지에서 다음 작업 중 하나를 수행합니다.

   보안 암호의 이름이나 ARN은 변경할 수 없다는 것을 유념하세요.

   - 설명을 업데이트하려면 보안 암호 세부 정보(Secrets details) 섹션에서 작업(Actions)을 선택한 후 설명 편집(Edit description)을 선택합니다.
   - 암호화 키를 업데이트하려면 the section called “보안 암호에 대한 암호화 키 변경”을(를) 참조하세요.
   - 태그를 업데이트하려면 태그 탭에서 태그 편집을 선택합니다. the section called “ 보안 암호 태그 지정”를 참조하세요.
   - 보안 암호 값을 업데이트하려면 the section called “보안 암호 값 업데이트”을(를) 참조하세요.
   - 보안 암호에 대한 권한을 업데이트하려면 개요 탭에서 권한 편집을 선택합니다. the section called “보안 암호에 권한 정책 연결”를 참조하세요.
   - 보안 암호의 교체를 업데이트하려면 교체 탭에서 교체 편집을 선택합니다. 보안 암호 교체를 참조하세요.
   - 보안 암호를 다른 리전으로 복제하려면 지역 간 비밀 복제 섹션을 참조하세요.
   - 보안 암호에 복제본이 있는 경우 복제본에 대한 암호화 키를 변경할 수 있습니다. 복제 탭에서 복제본에 대한 라디오 버튼을 선택한 다음 작업 메뉴에서 암호화 키 편집을 선택합니다. the section called “보안 암호 암호화 및 복호화”를 참조하세요.
   - 보안 암호를 다른 서비스에서 관리하도록 변경하려면 해당 서비스에서 보안 암호를 다시 만들어야 합니다. 관리형 시크릿를 참조하세요.

AWS CLI

Example 보안 암호 설명 업데이트

다음 `update-secret` 예에서는 보안 암호에 대한 설명을 업데이트합니다.

```
aws secretsmanager update-secret \
    --secret-id MyTestSecret \
    --description "This is a new description for the secret."
```

AWS SDK

10분마다 두 번 이상 지속적으로 PutSecretValue 또는 UpdateSecret을 호출하지 않는 것이 좋습니다. PutSecretValue 또는 UpdateSecret을 호출하여 보안 암호 값을 업데이트하면 Secrets Manager는 새 버전의 보안 암호를 생성합니다. Secrets Manager는 100개를 넘는 버전이 있을 때 레이블이 지정되지 않은 버전을 제거하지만 24시간 이내에 생성된 버전은 제거하지 않습니다. 10분마다 두 번 이상 보안 암호 값을 업데이트하면 Secrets Manager에서 제거하는 버전보다 더 많은 버전이 생성되고 보안 암호 버전 할당량에 도달하게 됩니다.

보안 암호를 업데이트하려면 UpdateSecret 또는 ReplicateSecretToRegions을(를) 사용하세요. 자세한 내용은 the section called “AWS SDK”을(를) 참조하세요.

에서 비밀 찾기 AWS Secrets Manager

필터 없이 보안 암호를 검색할 경우, Secrets Manager는 보안 암호 이름, 설명, 태그 키 및 태그 값의 키워드를 일치시킵니다. 필터 없는 검색은 대/소문자를 구분하지 않으며 공백, /, _, =, #과 같은 특수 문자를 무시하고 숫자와 문자만 사용합니다. 필터 없이 검색할 때 Secrets Manager는 검색 문자열을 분석하여 별개의 단어로 변환합니다. 대문자에서 소문자로, 문자에서 숫자로, 또는 숫자/문자에서 구두점으로 단어를 바꾸어 구분합니다. 예를 들어 credsDatabase#892를 검색 단어로 입력하면 이름, 설명, 태그 키 및 값으로 creds, Database, 892를 검색합니다.

Secrets Manager는 암호를 나열할 때 CloudTrail 로그 항목을 생성합니다. 자세한 정보는 the section called “다음과 같이 로그하십시오. AWS CloudTrail ”을 참조하세요.

다음 필터를 검색에 적용할 수 있습니다.

명칭

보안 암호 이름의 시작 부분과 일치하고 대소문자를 구분합니다. 예를 들어 이름(Name): **Data**는 DatabaseSecret이라는 보안 암호를 반환하지만 databaseSecret 또는 MyData를 반환하지는 않습니다.

설명

보안 암호 설명의 단어와 일치하고 대소문자를 구분하지 않습니다. 예를 들어 설명(Description): **My Description**은 보안 암호를 다음 설명과 일치시킵니다.

- My Description
- my description
- My basic description

- `Description of my secret`

관리 기관:

예를 AWS들어 CyberArk 또는 외부의 서비스에서 관리하는 비밀을 찾습니다 HashiCorp.

소유 서비스

대소문자를 구분 없이 관리 서비스 ID 접두사의 시작 부분과 일치합니다. 예를 들어, **`my-ser`**은(는) 접두사가 `my-serv` 및 `my-service`인 서비스에서 관리하는 보안 암호와 일치합니다. 자세한 정보는 관리형 시크릿을 참조하세요.

복제된 보안 암호

기본 보안 암호, 복제본 보안 암호 또는 복제되지 않은 보안 암호를 필터링할 수 있습니다.

태그 키

태그 키의 시작 부분과 일치하고 대소문자를 구분합니다. 예를 들어 태그 키(Tag key): **`Prod`**는 태그 `Production` 및 `Prod1`이 포함된 보안 암호를 반환하지만 태그 `prod` 또는 `1` `Prod`가 포함된 보안 암호는 반환하지 않습니다.

태그 값

태그 값의 시작 부분과 일치하고 대소문자를 구분합니다. 예를 들어 태그 값(Tag value): **`Prod`**는 태그 `Production` 및 `Prod1`이 포함된 보안 암호를 반환하지만 태그 값 `prod` 또는 `1` `Prod`가 포함된 보안 암호는 반환하지 않습니다.

Secrets Manager는 리전 서비스이며 선택한 리전 내부의 보안 암호만 반환됩니다.

AWS CLI

Example 계정의 보안 암호 목록

다음 `list-secrets` 예시에서는 계정에 있는 보안 암호 목록을 가져옵니다.

```
aws secretsmanager list-secrets
```

Example 계정의 보안 암호 목록 필터링

다음 `list-secrets` 예시에서는 계정에서 이름에 Test가 있는 보안 암호 목록을 가져옵니다. 이름의 필터링은 대소문자를 구분합니다.

```
aws secretsmanager list-secrets \
```

```
    --filter Key="name",Values="Test"
```

Example 다른 AWS 서비스에서 관리하는 비밀 찾기

다음 `list-secrets` 예시는 서비스에서 관리하는 보안 암호 목록을 가져옵니다. ID로 서비스를 지정합니다. 자세한 정보는 관리형 시크릿을 참조하세요.

```
aws secretsmanager list-secrets --filter Key="owning-service",Values="<service ID
 prefix>"
```

AWS SDK

AWS SDK 중 하나를 사용하여 비밀을 찾으려면 를 사용하십시오. `ListSecrets` 자세한 내용은 the section called “AWS SDK”을(를) 참조하세요.

AWS Secrets Manager 시크릿 삭제

비밀의 중요한 특성 때문에 AWS Secrets Manager 의도적으로 비밀을 삭제하기가 어렵습니다. Secrets Manager는 보안 암호를 바로 삭제하지 않습니다. 대신 Secrets Manager는 즉시 이 보안 암호에 액세스할 수 없도록 하고 최소 7일의 복구 기간 후에 삭제되도록 예약합니다. 복구 기간이 끝날 때까지 이전에 삭제한 보안 암호를 복구할 수 있습니다. 삭제하도록 표시한 보안 암호에 대해서는 요금이 부과되지 않습니다.

다른 리전에 복제된 기본 보안 암호는 삭제할 수 없습니다. 우선 복제본을 삭제한 다음 기본 보안 암호를 삭제합니다. 복제본을 삭제할 경우, 즉시 삭제됩니다.

또한 보안 암호 버전을 직접 삭제할 수 없습니다. 대신 AWS CLI 또는 AWS SDK를 사용하여 버전에서 모든 스테이징 레이블을 제거합니다. 이는 버전을 사용 중지 상태로 표시하고 Secrets Manager가 배경에서 해당 버전을 자동으로 삭제할 수 있게 합니다.

애플리케이션에서 여전히 암호를 사용하는지 여부를 모르는 경우 Amazon CloudWatch 경보를 생성하여 복구 기간 중에 암호에 액세스하려는 모든 시도를 경고할 수 있습니다. 자세한 정보는 삭제 예정인 AWS Secrets Manager 비밀에 대한 액세스 시기 모니터링을 참조하세요.

보안 암호를 삭제하려면 `secretsmanager:ListSecrets` 및 `secretsmanager:DeleteSecret` 권한이 있어야 합니다.

Secrets Manager는 시크릿을 삭제할 때 CloudTrail 로그 항목을 생성합니다. 자세한 정보는 the section called “다음과 같이 로그하십시오. AWS CloudTrail ”을 참조하세요.

보안 암호(콘솔)를 삭제하려면

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호 목록에서 삭제할 보안 암호를 선택합니다.
3. 보안 암호 세부 정보(Secret details) 섹션에서 작업(Actions)을 선택한 후 보안 암호 삭제(Delete secret)를 선택합니다.
4. 보안 암호 사용 중지 및 삭제 예약(Disable secret and schedule deletion) 대화 상자에서 대기 기간(Waiting period)에 영구적으로 삭제되기 전까지 대기할 일수를 입력합니다. Secrets Manager는 `DeletionDate`라는 필드를 연결하고 해당 필드를 현재 날짜 및 시간에, 복구 기간에 지정된 일수를 더한 값으로 설정합니다.
5. 삭제 예약(Schedule deletion)을 선택합니다.

삭제된 보안 암호를 보려면

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호(Secrets) 페이지에서 기본 설정 (Preferences)()을 선택합니다.
3. 기본 설정 대화 상자에서 삭제 예정 암호 표시을 선택한 후 저장를 선택합니다.

복제본 보안 암호를 삭제하려면

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 기본 보안 암호를 선택합니다.
3. 보안 암호 복제(Replicate Secret) 섹션에서 복제본 보안 암호를 선택합니다.
4. 작업(Actions) 메뉴에서 복제본 삭제(Delete Replica)를 선택합니다.

AWS CLI

Example 보안 암호 삭제

다음 `delete-secret` 예시에서는 보안 암호를 삭제합니다. DeletionDate 응답 필드에 날짜 및 `restore-secret`시간까지 입력하여 암호를 복구할 수 있습니다. 다른 리전에 복제된 보안 암호를 삭제하려면 먼저 `remove-regions-from-replication`(으)로 해당 복제본을 삭제한 다음 `delete-secret`을(를) 호출합니다.

```
aws secretsmanager delete-secret \
    --secret-id MyTestSecret \
    --recovery-window-in-days 7
```

Example 보안 암호 즉시 삭제

다음 delete-secret 예시는 복구 기간 없이 즉시 보안 암호를 삭제합니다. 이러한 보안 암호는 복구할 수 없습니다.

```
aws secretsmanager delete-secret \
    --secret-id MyTestSecret \
    --force-delete-without-recovery
```

Example 복제본 보안 암호 삭제

다음 remove-regions-from-replication 예시에서는 eu-west-3의 복제 보안 암호를 삭제합니다. 다른 리전에 복제된 기본 보안 암호를 삭제하려면 먼저 복제본을 삭제한 다음 delete-secret을(를) 호출합니다.

```
aws secretsmanager remove-regions-from-replication \
    --secret-id MyTestSecret \
    --remove-replica-regions eu-west-3
```

AWS SDK

보안 암호를 삭제하려면 DeleteSecret 명령을 사용합니다. 보안 암호의 버전을 삭제하려면 UpdateSecretVersionStage 명령을 사용합니다. 복제본을 삭제하려면 StopReplicationToReplica 명령을 사용합니다. 자세한 내용은 the section called “AWS SDK”을(를) 참조하세요.

AWS Secrets Manager 비밀 복원

Secrets Manager에서는 삭제 예약된 보안 암호를 더 이상 사용되지 않는 것으로 간주하며 이제 직접 액세스할 수 없습니다. 복구 기간이 지난 후 Secrets Manager는 보안 암호를 영구적으로 삭제합니다. Secrets Manager에서 보안 암호를 삭제한 후에는 복구할 수 없습니다. 복구 기간이 끝나기 전에 보안 암호를 복구하고 다시 액세스할 수 있습니다. 이렇게 하면 영구 삭제 예약을 취소하는 DeletionDate 필드가 제거됩니다.

콘솔에서 보안 암호와 메타데이터를 복원하려면 `secretsmanager:ListSecrets` 및 `secretsmanager:RestoreSecret` 권한이 있어야 합니다.

Secrets Manager는 시크릿을 복원할 때 CloudTrail 로그 항목을 생성합니다. 자세한 정보는 the section called “다음과 같이 로그하십시오. AWS CloudTrail ”을 참조하세요.

보안 암호(콘솔) 복원

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호 목록에서 복원할 보안 암호를 선택합니다.

   삭제된 보안 암호가 보안 암호 목록에 나타나지 않는 경우 기본 설정 (Preferences)(⚙) 을 선택합니다. 기본 설정 대화 상자에서 삭제 예정 암호 표시을 선택한 후 저장를 선택합니다.
3. 보안 암호 세부 정보(Secret details) 페이지에서 삭제 취소(Cancel deletion)를 선택합니다.
4. 보안 암호 삭제 취소(Cancel secret deletion) 대화 상자에서 삭제 취소(Cancel deletion)를 선택합니다.

AWS CLI

Example 이전에 삭제한 보안 암호 복원하기

다음 `restore-secret` 예시에서는 이전에 삭제가 예정된 보안 암호를 복원합니다.

```
aws secretsmanager restore-secret \
    --secret-id MyTestSecret
```

AWS SDK

삭제하도록 표시된 보안 암호를 복원하려면 `RestoreSecret` 명령을 사용합니다. 자세한 내용은 the section called “AWS SDK”을(를) 참조하세요.

AWS Secrets Manager 보안 암호 태그 지정

Secrets Manager는 태그를 사용자가 정의하는 키와 선택적 값으로 구성된 레이블로 정의합니다. 태그를 사용하여 AWS 계정에서 보안 암호 및 기타 리소스 관리, 검색 및 필터링을 수행할 수 있습니다. 보안 암호에 태그를 지정할 때 모든 리소스에서 표준 이름 지정 체계를 사용하세요. 자세한 내용은 태그 지정 모범 사례 백서를 참조하세요.

보안 암호에 연결된 태그를 확인하여 보안 암호에 대한 액세스를 허가하거나 거부할 수 있습니다. 자세한 내용은 the section called “예: 태그를 사용하여 보안 암호에 대한 액세스 제어” 단원을 참조하세요.

콘솔, AWS CLI 및 SDK에서 태그별로 보안 암호를 찾을 수 있습니다. 또한 AWS는 Resource Groups 도구를 제공하여 태그를 기반으로 리소스를 통합하고 구성하는 사용자 지정 콘솔을 만듭니다. 특정 태그가 있는 보안 암호를 찾으려면 the section called “보안 암호 찾기”를 참조하세요. Secrets Manager는 태그 기반 비용 할당을 지원하지 않습니다.

태그에 보안 암호에 대한 민감한 정보를 절대 저장하지 마세요.

태그 할당량 및 이름 지정 제한은 AWS 일반 참조 가이드에서 태그 지정을 위한 Service Quotas을 참조하세요. 태그는 대/소문자를 구분합니다.

Secrets Manager는 암호에 태그를 지정하거나 태그를 해제할 때 CloudTrail 로그 항목을 생성합니다. 자세한 내용은 the section called “다음과 같이 로그하십시오. AWS CloudTrail ” 섹션을 참조하세요.

보안 암호의 태그를 변경하려면(콘솔)

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호 목록에서 보안 암호를 선택합니다.
3. 보안 암호 세부 정보 페이지의 태그 탭에서 태그 편집을 선택합니다. 태그 키 이름과 값은 대소문자를 구분하며 태그 키는 고유해야 합니다.

AWS CLI

Example 보안 암호에 태그 추가

다음 `tag-resource` 예시에서는 간편 구문으로 태그를 연결하는 방법을 보여줍니다.

```
aws secretsmanager tag-resource \
            --secret-id MyTestSecret \
            --tags Key=FirstTag,Value=FirstValue
```

Example 보안 암호에 여러 태그 추가

다음 `tag-resource` 예시에서는 두 개의 키-값 태그를 보안 암호에 연결합니다.

```
aws secretsmanager tag-resource \
            --secret-id MyTestSecret \
```

```
            --tags '[{"Key": "FirstTag", "Value": "FirstValue"}, {"Key": "SecondTag",
 "Value": "SecondValue"}]'
```

Example 보안 암호에서 태그 제거

다음 `untag-resource` 예시에서는 보안 암호에서 두 개의 태그를 제거합니다. 각 태그의 키와 값이 모두 제거됩니다.

```
aws secretsmanager untag-resource \
            --secret-id MyTestSecret \
            --tag-keys '[ "FirstTag", "SecondTag"]'
```

AWS SDK

보안 암호의 태그를 변경하려면 `TagResource` 또는 `UntagResource`를 사용합니다. 자세한 내용은 the section called “AWS SDK” 섹션을 참조하세요.

지역 간 AWS Secrets Manager 비밀 복제

비밀을 여러 곳에 복제하여 해당 지역에 분산된 애플리케이션을 AWS 리전 지원하여 지역 액세스 및 짧은 지연 시간 요구 사항을 충족할 수 있습니다. 나중에 필요할 경우 복제본 암호를 독립 실행형으로 승격한 다음 독립적으로 복제가 가능하도록 설정할 수 있습니다. Secrets Manager는 지정된 리전에 걸쳐 태그 및 리소스 정책과 같은 암호화된 보안 암호 데이터 및 메타데이터를 복제합니다.

복제된 암호의 ARN 지역을 제외하고 기본 암호와 동일합니다. 예를 들면 다음과 같습니다.

- 기본 보안 암호: `arn:aws:secretsmanager:Region1:123456789012:secret:MySecret-a1b2c3`
- 복제본 보안 암호: `arn:aws:secretsmanager:Region2:123456789012:secret:MySecret-a1b2c3`

복제본 보안 암호에 대한 요금 정보는 AWS Secrets Manager 요금을 참조하세요.

다른 리전으로 복제된 소스 데이터베이스에 대한 데이터베이스 보안 인증 정보를 저장하면 보안 암호에 해당 소스 데이터베이스에 대한 연결 정보가 포함됩니다. 그 후에 보안 암호를 복제하면 복제본은 소스 보안 암호의 복사본이 되며 동일한 연결 정보를 포함합니다. 리전 연결 정보를 위해 보안 암호에 추가로 키/값 쌍을 추가할 수 있습니다.

기본 보안 암호에 대한 교체를 켜면 Secrets Manager가 기본 리전의 보안 암호를 교체하고 새 보안 암호 값이 연결된 모든 복제본 보안 암호에 전파됩니다. 모든 복제본 보안 암호에 대해 개별적인 교체를 관리할 필요가 없습니다.

활성화된 모든 지역에서 암호를 복제할 수 있습니다. AWS 그러나 중국 AWS 지역과 같은 AWS GovCloud (US) 특수 지역에서 Secrets Manager를 사용하는 경우 이러한 특수 AWS 지역 내에서만 암호와 복제본을 구성할 수 있습니다. 활성화된 AWS 지역의 암호를 특정 지역에 복제하거나 특수 지역의 암호를 상업 지역으로 복제할 수 없습니다.

보안 암호를 다른 리전으로 복제하려면 먼저 해당 리전을 사용해야 합니다. 자세한 내용은 AWS 리전 관리를 참조하세요.

보안 암호가 저장된 리전의 Secrets Manager 엔드포인트를 호출하여 복제하지 않고 여러 리전에서 보안 암호를 사용할 수 있습니다. 엔드포인트 목록은 the section called “Secrets Manager 엔드포인트” 섹션을 참조하세요. 복제를 사용하여 워크로드의 복원력을 개선하려면 재해 복구 (DR) 아키텍처 AWS, 1부: 클라우드에서의 복구 전략을 참조하십시오.

Secrets Manager는 시크릿을 복제할 때 CloudTrail 로그 항목을 생성합니다. 자세한 정보는 the section called “다음과 같이 로그하십시오. AWS CloudTrail ”을 참조하세요.

보안 암호를 다른 리전으로 복제하려면(콘솔)

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호 목록에서 보안 암호를 선택합니다.
3. 보안 암호 세부 정보 페이지의 복제 탭에서 다음 중 하나를 수행합니다.
   - 보안 암호가 복제되지 않은 경우에는 보안 암호 복제(Replicate secret)를 선택합니다.
   - 보안 암호가 복제된 경우에는 보안 암호 복제(Replicate secret) 섹션에서 리전 추가(Add Region)를 선택합니다.
4. 복제본 리전 추가(Add replica regions) 대화 상자에서 다음을 수행합니다.

   a. AWS 리전(Region)에서 보안 암호를 복제하고자 하는 리전을 선택합니다.

   b. (선택 사항) 암호화 키(Encryption key)에서 보안 암호를 암호화할 KMS 키를 선택합니다. 키가 복제본 리전에 있어야 합니다.

   c. (선택 사항) 다른 리전을 추가하려면더 많은 리전 추가(Add more regions)를 선택합니다.

   d. 복제(Replicate)를 선택합니다.

   보안 암호 세부 정보 페이지로 돌아갑니다. 보안 암호 복제(Replicate Secret) 섹션에서 각 리전의 복제 상태(Replication Status)가 표시됩니다.

AWS CLI

Example 다른 리전으로 보안 암호 복제

다음 `replicate-secret-to-regions` 예시에서는 eu-west-3으로 보안 암호를 복제합니다. 복제본은 AWS 관리 키 aws/secretsmanager를 사용하여 암호화됩니다.

```
aws secretsmanager replicate-secret-to-regions \
        --secret-id MyTestSecret \
        --add-replica-regions Region=eu-west-3
```

Example 시크릿을 생성하고 복제하세요.

다음 예제에서는 시크릿을 생성하여 eu-west-3에 복제합니다. 복제본은 관리형 키 aws/secretsmanager를 사용하여 암호화됩니다. AWS

```
aws secretsmanager create-secret \
    --name MyTestSecret \
    --description "My test secret created with the CLI." \
    --secret-string "{\"user\":\"diegor\",\"password\":\"EXAMPLE-PASSWORD\"}"
    --add-replica-regions Region=eu-west-3
```

AWS SDK

보안 암호를 복제하려면 `ReplicateSecretToRegions` 명령을 사용합니다. 자세한 내용은 the section called “AWS SDK”을(를) 참조하세요.

복제본 보안 암호를 AWS Secrets Manager의 독립 실행형 보안 암호로 승격

복제본 보안 암호는 다른 AWS 리전의 기본에서 복제되는 보안 암호입니다. 이 보안 암호는 기본 암호 값과 메타데이터가 동일하지만 다른 KMS 키를 사용하여 암호화할 수 있습니다. 복제본 보안 암호는 암호화 키를 제외하고 기본 보안 암호와 독립적으로 업데이트할 수 없습니다. 복제본 보안 암호를 승격할 경우 복제본 보안 암호가 기본 보안 암호에서 분리되고 복제본 보안 암호가 독립 실행형 보안 암호로 설정됩니다. 기본 보안 암호에 대한 변경 사항은 독립 실행형 보안 암호에 복제되지 않습니다.

기본 암호를 사용할 수 없게 되면 재해 복구 솔루션으로 복제본 암호를 독립 실행형 암호로 승격할 수 있습니다. 또는 복제본에 대해 교체를 켜려는 경우 복제본을 독립 실행형 보안 암호로 승격해야 합니다.

복제본을 승격한 경우, 독립 실행형 보안 암호를 사용하도록 해당 애플리케이션을 업데이트해야 합니다.

Secrets Manager는 암호의 수준을 올릴 때 CloudTrail 로그 항목을 생성합니다. 자세한 내용은 the section called “다음과 같이 로그하십시오. AWS CloudTrail ” 섹션을 참조하세요.

복제본 보안 암호를 승격하려면(콘솔)

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔에 로그인합니다.
2. 복제본 리전으로 이동합니다.
3. 보안 암호(Secrets) 페이지에서 복제 보안 암호를 선택합니다.
4. 복제 보안 암호 세부 정보 페이지에서 독립 실행형 보안 암호로 승격(Promote to standalone secret)을 선택합니다.

5. 복제본을 독립형 보안 암호로 승격(Promote replica to standalone secret) 대화 상자에서 리전을 입력한 다음 복제본 승격(Promote replica)을 선택합니다.

AWS CLI

Example 복제 보안 암호를 기본으로 승격

다음 stop-replication-to-replica 예시에서는 복제 암호와 기본 암호 간의 링크를 제거합니다. 복제 보안 암호는 복제본 리전의 기본 보안 암호로 승격됩니다. 복제 리전 내에서 stop-replication-to-replica을(를) 호출해야 합니다.

```
aws secretsmanager stop-replication-to-replica \
    --secret-id MyTestSecret
```

AWS SDK

복제본을 독립 실행형 보안 암호로 승격하려면 StopReplicationToReplica 명령을 사용합니다. 이 명령은 반드시 복제본 보안 암호 리전에서 호출해야 합니다. 자세한 내용은 the section called “AWS SDK” 섹션을 참조하세요.

AWS Secrets Manager 복제 방지

를 ReplicateSecretToRegions사용하거나 를 사용하여 암호를 만들 때 암호를 복제할 수 있으므로 사용자가 암호를 복제하지 못하도록 하려면 매개 변수가 포함된 작업을 금지하는 것이 좋습니다. CreateSecretAddReplicaRegions 권한 정책에 Condition 명령문을 사용하여 복제 영역을 추가하지 않는 작업만 허용할 수 있습니다. 사용할 수 있는 조건문에 대한 다음 정책 예제를 참조하십시오.

Example 복제 권한 방지

다음 정책 예제는 복제 영역을 추가하지 않는 모든 작업을 허용하는 방법을 보여줍니다. 이렇게 하면 사용자가 및 ReplicateSecretToRegions 를 통해 암호를 복제할 수 없습니다. CreateSecret

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Action": "secretsmanager:*",
      "Resource": "*",
      "Condition": {
        "Null": {
          "secretsmanager:AddReplicaRegions": "true"
        }
      }
    }
  ]
}
```

Example 특정 지역에 대한 복제 권한만 허용

다음 정책은 다음을 모두 허용하는 방법을 보여줍니다.

- 복제 없이 시크릿 생성
- 미국과 캐나다의 지역에만 복제하여 시크릿을 생성합니다.
- 비밀번호는 미국 및 캐나다의 지역에만 복제할 수 있습니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:CreateSecret",
        "secretsmanager:ReplicateSecretToRegions"
      ],
      "Resource": "*",
      "Condition": {
        "ForAllValues:StringLike": {
          "secretsmanager:AddReplicaRegions": [
            "us-*",
            "ca-*"
          ]
        }
      }
    }
  ]
}
```

복제 문제 해결 AWS Secrets Manager

다음은 복제가 실패할 수 있는 몇 가지 이유입니다.

선택한 리전에 동일한 이름의 보안 암호가 있습니다

이 문제를 해결하려면 복제본 리전에서 중복된 이름 보안 암호를 덮어쓸 수 있습니다. 복제를 재시도한 다음 복제 재시도 대화 상자에서 덮어쓰기를 선택합니다.

복제를 완료하기 위해 KMS 키에 사용할 수 있는 권한이 없습니다

Secrets Manager는 복제본 리전의 새 KMS 키를 사용하여 다시 암호화하기 전에 먼저 암호를 해독합니다. 기본 리전의 암호화 키에 대한 `kms:Decrypt` 권한이 없는 경우 이 오류가 발생합니다. `aws/secretsmanager` 이외의 KMS 키를 사용하여 복제된 보안 암호를 암호화하려면 키에 `kms:GenerateDataKey` 및 `kms:Encrypt`가 필요합니다. the section called “KMS 키에 대한 권한” 섹션을 참조하십시오.

KMS 키가 비활성화되었거나 찾을 수 없음

기본 지역의 암호화 키가 비활성화되거나 삭제된 경우 Secrets Manager는 보안 암호를 복제할 수 없습니다. 보안 암호에 비활성화되거나 삭제된 암호화 키를 사용하여 암호화된 사용자 지정 레이블 버전이 있는 경우, 암호화 키를 변경한 경우에도 이 오류가 발생할 수 있습니다. Secrets Manager가 암호화를 수행하는 방법에 대한 자세한 내용은 the section called “보안 암호 암호화 및 복호화”을(를) 참조하세요. 이 문제를 해결하려면 Secrets Manager가 현재 암호화 키를 사용하여 암호화하도록 보안 암호 버전을 다시 생성하면 됩니다. 자세한 내용은 보안 암호에 대한 암호화 키 변경을 참조하세요. 그런 다음 복제를 다시 시도합니다.

```
aws secretsmanager put-secret-value \
  --secret-id testDescriptionUpdate \
  --secret-string "SecretValue" \
  --version-stages "MyCustomLabel"
```

복제가 발생하는 리전을 활성화하지 않았습니다

지역을 활성화하는 방법에 대한 자세한 내용은 AWS 지역 관리를 참조하십시오. AWS 계정 관리 참조 가이드에서.

에서 비밀 가져오기 AWS Secrets Manager

Secrets Manager는 시크릿을 검색할 때 CloudTrail 로그 항목을 생성합니다. 자세한 정보는 the section called “다음과 같이 로그하십시오. AWS CloudTrail ”을 참조하세요.

다음을 사용하여 비밀 값을 검색할 수 있습니다.

- 자바를 사용하여 Secrets Manager 시크릿 값 가져오기
- Python을 사용하여 Secrets Manager의 비밀 값 가져오기
- .NET을 사용하여 Secrets Manager 비밀 값 가져오기
- Go를 사용하여 Secrets Manager의 비밀 값 가져오기
- C++ AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기
- JavaScript AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기
- Kotlin AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기
- PHP AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기
- 루비 AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기
- Rust AWS SDK를 사용하여 Secrets Manager의 시크릿 값 가져오기
- 다음을 사용하여 비밀 값을 가져오세요. AWS CLI
- AWS 콘솔을 사용하여 비밀 값을 가져오세요.
- 에서 AWS Secrets Manager 시크릿 사용 AWS Batch
- AWS CloudFormation 리소스에서 AWS Secrets Manager 비밀 가져오기
- 아마존 엘라스틱 쿠버네티스 서비스에서 AWS Secrets Manager 시크릿 사용하기
- GitHub 작업에서 AWS Secrets Manager 비밀 사용
- AWS IoT Greengrass에서 AWS Secrets Manager 보안 암호 사용
- AWS Lambda 함수에 AWS Secrets Manager 시크릿 사용
- 파라미터 스토어에서 AWS Secrets Manager 보안 암호 사용

자바를 사용하여 Secrets Manager 시크릿 값 가져오기

애플리케이션에서는 SDK를 `GetSecretValue` 호출하거나 `BatchGetSecretValue` 원하는 AWS SDK를 사용하여 암호를 검색할 수 있습니다. 그러나 클라이언트 측 캐싱을 사용하여 보안 암호 값을 캐싱하는 것이 좋습니다. 보안 암호 캐싱은 속도를 향상시키고 비용을 절감합니다.

시크릿의 자격 증명을 사용하여 데이터베이스에 연결하려면 기본 JDBC 드라이버를 래핑하는 Secrets Manager SQL 연결 드라이버를 사용할 수 있습니다. 또한 클라이언트 측 캐싱을 사용하므로 Secrets Manager API 호출 비용을 줄일 수 있습니다.

주제

- 클라이언트측 캐싱과 함께 Java를 사용하여 Secrets Manager 비밀 값 가져오기
- 시크릿에 자격 증명이 있는 JDBC를 사용하여 SQL 데이터베이스에 연결 AWS Secrets Manager
- 자바 AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기

클라이언트측 캐싱과 함께 Java를 사용하여 Secrets Manager 비밀 값 가져오기

보안 암호를 검색할 때 Secret Manager Java 기반 캐싱 구성 요소를 사용하여 나중에 사용할 수 있도록 캐싱할 수 있습니다. 캐싱된 보안 암호를 검색하는 것이 Secret Manager에서 검색하는 것보다 빠릅니다. Secrets Manager API를 호출하는 데는 비용이 발생하므로 캐시를 사용하면 비용을 줄일 수 있습니다. 암호를 검색할 수 있는 모든 방법은 비밀 찾기을(를) 참조하세요.

캐시 정책은 LRU(가장 오랫동안 사용되지 않음)이므로, 캐시에서 보안 암호를 폐기해야 하는 경우 가장 오랫동안 사용되지 않은 보안 암호가 삭제됩니다. 기본적으로 캐시는 보안 암호를 매시간 새로 고칩니다. 캐시에서 보안 암호를 새로 고치는 주기를 구성하고 보안 암호 검색에 연결하여 더 많은 기능을 추가할 수 있습니다.

캐시 참조가 해제되면 캐시는 가비지 수집을 강제로 적용하지 않습니다. 캐시 구현에는 캐시 무효화가 포함되지 않습니다. 캐시 구현은 캐시 자체에 중점을 두며, 보안을 강화하거나 보안에 초점을 맞추지 않습니다. 캐시에서 항목 암호화와 같은 추가 보안이 필요한 경우 제공된 인터페이스 및 추상 메서드를 사용하세요.

이 구성 요소를 사용하려면 다음이 필요합니다.

- Java 8 이상 개발 환경입니다. Oracle 웹 사이트의 Java SE 다운로드를 참조하세요.
- 자바용 AWS SDK 1.x. 프로젝트에서 Java용 AWS SDK의 두 버전을 모두 사용할 수 있습니다. 자세한 내용은 Java 1.x 및 2.x용 SDK 사용을 참조하십시오. side-by-side

소스 코드를 다운로드하려면 에서 Secrets Manager 자바 기반 캐싱 클라이언트 구성 요소를 참조하십시오. GitHub

프로젝트에 구성 요소를 추가하려면 Maven pom.xml 파일에 다음 dependency를 포함합니다. Maven에 대한 자세한 내용은 Apache Maven Project 웹 사이트의 시작 안내서를 참조하세요.

```
<dependency>
  <groupId>com.amazonaws.secretsmanager</groupId>
  <artifactId>aws-secretsmanager-caching-java</artifactId>
  <version>1.0.2</version>
</dependency>
```

필요한 권한:

- `secretsmanager:DescribeSecret`
- `secretsmanager:GetSecretValue`

자세한 정보는 권한 참조을 참조하세요.

레퍼런스

- SecretCache
- SecretCacheConfiguration
- SecretCacheHook

Example 보안 암호 검색

다음 코드 예제에서는 보안 암호 문자열을 검색하는 Lambda 함수를 보여줍니다. 이 예제는 함수 핸들러 외부의 캐시를 인스턴스화하는 모범 사례를 따르므로 Lambda 함수를 다시 호출하는 경우 API를 계속 호출하지 않습니다.

```
package com.amazonaws.secretsmanager.caching.examples;

    import com.amazonaws.services.lambda.runtime.Context;
    import com.amazonaws.services.lambda.runtime.RequestHandler;
    import com.amazonaws.services.lambda.runtime.LambdaLogger;

    import com.amazonaws.secretsmanager.caching.SecretCache;

    public class SampleClass implements RequestHandler<String, String> {

         private final SecretCache cache  = new SecretCache();
```

```
        @Override public String handleRequest(String secretId,  Context context) {
            final String secret  = cache.getSecretString(secretId);

           // Use the secret, return success;

        }
    }
```

SecretCache

Secret Manager에서 요청한 보안 암호에 대한 인 메모리 캐시. the section called “getSecretString” 또는 the section called “getSecretBinary”를 사용하여 캐시에서 보안 암호를 검색합니다. 생성자의 the section called “SecretCacheConfiguration” 객체에 전달하여 캐시 설정을 구성할 수 있습니다.

예제를 포함한 자세한 내용은 the section called “클라이언트 측 캐싱을 지원하는 Java”을 참조하세요.

Constructors

`public SecretCache()`

SecretCache 객체에 대한 기본 생성자.

`public SecretCache(AWSSecretsManagerClientBuilder builder)`

제공된 `AWSSecretsManagerClientBuilder`로 생성한 Secrets Manager 클라이언트를 사용하여 새 캐시를 생성합니다. 이 생성자를 사용하여 Secrets Manager 클라이언트를 사용자 지정할 수 있습니다 (예: 특정 지역 또는 엔드포인트 사용).

`public SecretCache(AWSSecretsManager client)`

제공된 `AWSSecretsManagerClient`를 사용하여 새 보안 암호 캐시를 생성합니다. 이 생성자를 사용하여 Secrets Manager 클라이언트를 사용자 지정할 수 있습니다 (예: 특정 지역 또는 엔드포인트 사용).

`public SecretCache(SecretCacheConfiguration config)`

제공된 `the section called “SecretCacheConfiguration”`을 사용하여 새 보안 암호 캐시를 생성합니다.

메서드

getSecretString

`public String getSecretString(final String secretId)`

Secrets Manager에서 문자열 보안 암호를 검색합니다. `String`을 반환합니다.

getSecretBinary

```
public ByteBuffer getSecretBinary(final String secretId)
```

Secrets Manager에서 이진 보안 암호를 검색합니다. `ByteBuffer`을 반환합니다.

refreshNow

```
public boolean refreshNow(final String secretId) throws
InterruptedException
```

캐시를 강제로 새로 고칩니다. 오류 없이 새로 고침이 완료되는 경우 `true`를 반환하고 그렇지 않으면 `false`를 반환합니다.

close

```
public void close()
```

캐시를 닫습니다.

SecretCacheConfiguration

캐싱된 보안 암호에 대한 최대 캐시 크기 및 유지 시간(TTL)과 같은 the section called “SecretCache”에 대한 캐시 구성 옵션.

생성자

```
public SecretCacheConfiguration
```

`SecretCacheConfiguration` 객체에 대한 기본 생성자.

메서드

getClient

```
public AWSSecretsManager getClient()
```

캐시가 보안 암호를 검색할 `AWSSecretsManagerClient`를 반환합니다.

setClient

```
public void setClient(AWSSecretsManager client)
```

캐시가 보안 암호를 검색할 AWSSecretsManagerClient를 설정합니다.

getCacheHook

```
public SecretCacheHook getCacheHook()
```

캐시 업데이트를 연결하는 데 사용되는 the section called “SecretCacheHook” 인터페이스를 반환합니다.

setCacheHook

```
public void setCacheHook(SecretCacheHook cacheHook)
```

캐시 업데이트를 연결하는 데 사용되는 the section called “SecretCacheHook” 인터페이스를 설정합니다.

getMaxCache크기

```
public int getMaxCacheSize()
```

최대 캐시 크기를 반환합니다. 기본값은 보안 암호 1,024개입니다.

setMaxCache사이즈

```
public void setMaxCacheSize(int maxCacheSize)
```

최대 캐시 크기를 설정합니다. 기본값은 보안 암호 1,024개입니다.

getCacheItemTTL

```
public long getCacheItemTTL()
```

캐싱된 항목에 대한 TTL을 밀리초 단위로 반환합니다. 캐싱된 보안 암호가 이 TTL을 초과하면 캐시는 AWSSecretsManagerClient에서 보안 암호의 새 사본을 검색합니다. 기본값은 밀리초 단위로 1시간입니다.

TTL 이후에 보안 암호가 요청되면 캐시가 보안 암호를 동기식으로 새로 고칩니다. 동기식 새로 고침이 실패하면 캐시는 오래된 보안 암호를 반환합니다.

setCacheItemTTL

```
public void setCacheItemTTL(long cacheItemTTL)
```

캐싱된 항목에 대한 TTL을 밀리초 단위로 설정합니다. 캐싱된 암호가 이 TTL을 초과하면 캐시는 AWSSecretsManagerClient에서 보안 암호의 새 사본을 검색합니다. 기본값은 밀리초 단위로 1시간입니다.

getVersionStage

```
public String getVersionStage()
```

캐싱할 보안 암호의 버전을 반환합니다. 자세한 내용은 보안 암호 버전을 참조하세요. 기본값은 "AWSCURRENT"입니다.

setVersionStage

```
public void setVersionStage(String versionStage)
```

캐싱할 보안 암호의 버전을 설정합니다. 자세한 내용은 보안 암호 버전을 참조하세요. 기본값은 "AWSCURRENT"입니다.

SecretCacheConfiguration 클라이언트 포함

```
public SecretCacheConfiguration withClient(AWSSecretsManager client)
```

보안 암호를 검색할 AWSSecretsManagerClient를 설정합니다. 새 설정으로 업데이트된 SecretCacheConfiguration 객체를 반환합니다.

SecretCacheConfiguration withCacheHook

```
public SecretCacheConfiguration withCacheHook(SecretCacheHook cacheHook)
```

인 메모리 캐시를 연결하는 데 사용되는 인터페이스를 설정합니다. 새 설정으로 업데이트된 SecretCacheConfiguration 객체를 반환합니다.

SecretCacheConfiguration withMaxCache사이즈

```
public SecretCacheConfiguration withMaxCacheSize(int maxCacheSize)
```

최대 캐시 크기를 설정합니다. 새 설정으로 업데이트된 SecretCacheConfiguration 객체를 반환합니다.

SecretCacheConfiguration withCacheItemTTL

```
public SecretCacheConfiguration withCacheItemTTL(long cacheItemTTL)
```

캐싱된 항목에 대한 TTL을 밀리초 단위로 설정합니다. 캐싱된 암호가 이 TTL을 초과하면 캐시는 AWSSecretsManagerClient에서 보안 암호의 새 사본을 검색합니다. 기본값은 밀리초 단위로 1시간입니다. 새 설정으로 업데이트된 SecretCacheConfiguration 객체를 반환합니다.

SecretCacheConfiguration withVersionStage

```
public SecretCacheConfiguration withVersionStage(String versionStage)
```

캐싱할 보안 암호의 버전을 설정합니다. 자세한 내용은 보안 암호 버전을 참조하세요. 새 설정으로 업데이트된 SecretCacheConfiguration 객체를 반환합니다.

SecretCacheHook

캐시에 저장 중인 보안 암호에 대한 작업을 수행하기 위해 the section called “SecretCache”에 연결되는 인터페이스.

put

```
Object put(final Object o)
```

캐시에 저장할 객체를 준비합니다.

캐시에 저장할 객체를 반환합니다.

get

```
Object get(final Object cachedObject)
```

캐싱된 객체에서 객체를 추출합니다.

캐시에서 객체를 반환합니다.

시크릿에 자격 증명이 있는 JDBC를 사용하여 SQL 데이터베이스에 연결 AWS Secrets Manager

Java 애플리케이션에서는 Secrets Manager SQL Connection 드라이버를 사용하여 Secrets Manager에 저장된 자격 증명을 사용하여 MySQL, PostgreSQL, 오라클, MSSQLServer, Db2 및 Redshift 데이터베이스에 연결할 수 있습니다. 각 드라이버는 기본 JDBC 드라이버를 래핑하므로 JDBC 호출을 사용하여 데이터베이스에 액세스할 수 있습니다. 그러나 연결을 위한 사용자 이름과 암호를 전달하는 대신 보안 암호의 ID를 제공합니다. 드라이버는 Secrets Manager를 호출하여 보안 암호 값을 검색한 다음 보안 암호의 보안 인증 정보를 사용하여 데이터베이스에 연결합니다. 또한 드라이버는 Java 클라이언트 측 캐싱 라이브러리를 사용하여 자격 증명을 캐싱하므로 향후 연결에는 Secret Manager에 대한 호

출이 필요하지 않습니다. 기본적으로 캐시는 매시간, 그리고 보안 암호가 교체될 때마다 새로 고침됩니다. 캐시를 구성하려면 the section called “SecretCacheConfiguration” 섹션을 참조하세요.

에서 소스 코드를 GitHub다운로드할 수 있습니다.

Secrets Manager SQL Connection 드라이버를 사용하려면:

- 애플리케이션이 Java 8 이상이어야 합니다.
- 보안 암호는 다음 형식 중 하나이어야 합니다:
  - 예상 JSON 구조의 데이터베이스 보안 암호. 형식을 확인하려면 Secrets Manager 콘솔에서 보안 암호를 확인하고 보안 암호 값 검색(Retrieve secret value)을 선택합니다. 또는 AWS CLI, 를 호출해도 get-secret-value됩니다.
  - Amazon RDS 관리형 보안 암호. 이 유형의 암호에 대해서는 연결을 설정할 때 엔드포인트 및 포트를 지정해야 합니다.
  - 아마존 Redshift의 관리 비밀번호입니다. 이 유형의 암호에 대해서는 연결을 설정할 때 엔드포인트 및 포트를 지정해야 합니다.

데이터베이스가 다른 리전으로 복제되는 경우 다른 리전의 복제본 데이터베이스에 연결하려면 연결을 생성할 때 리전 엔드포인트 및 포트를 지정합니다. 리전 연결 정보를 보안 암호에 추가 키/값 쌍으로 저장하거나, SSM Parameter Store 파라미터에 저장하거나, 코드 구성에 저장할 수 있습니다.

프로젝트에 드라이버를 추가하려면 Maven 빌드 파일 `pom.xml`에 드라이버에 대한 다음 종속성을 추가합니다. 자세한 내용은 Maven Central Repository 웹 사이트의 Secrets Manager SQL Connection Library를 참조하세요.

```
<dependency>
    <groupId>com.amazonaws.secretsmanager</groupId>
    <artifactId>aws-secretsmanager-jdbc</artifactId>
    <version>1.0.12</version>
</dependency>
```

드라이버는 기본 보안 인증 공급자 체인을 사용합니다. Amazon EKS에서 드라이버를 실행하는 경우 서비스 계정 역할 대신 실행 중인 노드의 보안 인증 정보를 가져올 수 있습니다. 이 사항을 처리하려면 Gradle 또는 Maven 프로젝트 파일에 `com.amazonaws:aws-java-sdk-sts` 버전 1을 종속 항목으로 추가하세요.

`secretsmanager.properties`파일에 AWS PrivateLink DNS 엔드포인트 URL과 리전을 설정하려면:

```
drivers.vpcEndpointUrl = endpoint URL
drivers.vpcEndpointRegion = endpoint region
```

기본 리전을 재정의하려면 AWS_SECRET_JDBC_REGION 환경 변수를 설정하거나 secretsmanager.properties 파일을 다음과 같이 변경하세요.

```
drivers.region = region
```

필요한 권한:

- secretsmanager:DescribeSecret
- secretsmanager:GetSecretValue

자세한 정보는 권한 참조을 참조하세요.

예:

- 데이터베이스에 대한 연결 설정
- 엔드포인트 및 포트를 지정하여 연결 설정
- c3p0 연결 풀링을 사용하여 연결 설정
- c3p0 연결 풀링을 사용하여 엔드포인트 및 포트 지정을 통한 연결 설정

데이터베이스에 대한 연결 설정

다음 예시에서는 보안 암호의 자격 증명 및 연결 정보를 사용하여 데이터베이스에 대한 연결을 설정하는 방법을 보여줍니다. 연결이 설정되면 JDBC 호출을 사용하여 데이터베이스에 액세스할 수 있습니다. 자세한 내용은 Java 설명서 웹 사이트의 JDBC Basics를 참조하세요.

MySQL

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerMySQLDriver" ).newInstance

// Retrieve the connection info from the secret using the secret ARN
String URL = "secretId";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
```

```
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

PostgreSQL

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerPostgreSQLDriver" ).newIns

// Retrieve the connection info from the secret using the secret ARN
String URL = "secretId";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

Oracle

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerOracleDriver" ).newInstanc

// Retrieve the connection info from the secret using the secret ARN
String URL = "secretId";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

MSSQLServer

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerMSSQLServerDriver" ).newIn
```

```
// Retrieve the connection info from the secret using the secret ARN
String URL = "secretId";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

Db2

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerDb2Driver" ).newInstance()

// Retrieve the connection info from the secret using the secret ARN
String URL = "secretId";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

Redshift

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerRedshiftDriver" ).newInsta

// Retrieve the connection info from the secret using the secret ARN
String URL = "secretId";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

엔드포인트 및 포트를 지정하여 연결 설정

다음 예에서는 지정된 엔드포인트 및 포트와 함께 보안 암호의 보안 인증 정보를 사용하여 데이터베이스에 대한 연결을 설정하는 방법을 보여줍니다.

Amazon RDS 관리형 보안 암호에는 데이터베이스의 엔드포인트 및 포트가 포함되지 않습니다. Amazon RDS에서 관리하는 보안 암호의 마스터 보안 인증을 사용하여 데이터베이스에 연결하려면 이를 해당 코드로 지정해야 합니다.

다른 리전으로 복제된 보안 암호는 리전 데이터베이스에 대한 연결 지연 시간을 줄여 줄 수 있지만, 소스 보안 암호와 다른 연결 정보는 포함하지 않습니다. 각 복제본은 소스 보안 암호의 복사본입니다. 리전 연결 정보를 보안 암호에 저장하려면 엔드포인트에 대한 키/값 쌍 및 리전에 대한 포트 정보를 추가합니다.

연결이 설정되면 JDBC 호출을 사용하여 데이터베이스에 액세스할 수 있습니다. 자세한 내용은 Java 설명서 웹 사이트의 JDBC Basics를 참조하세요.

MySQL

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerMySQLDriver" ).newInstance

// Set the endpoint and port. You can also retrieve it from a key/value pair in the
 secret.
String URL = "jdbc-secretsmanager:mysql://example.com:3306";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

PostgreSQL

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerPostgreSQLDriver" ).newIns

// Set the endpoint and port. You can also retrieve it from a key/value pair in the
 secret.
String URL = "jdbc-secretsmanager:postgresql://example.com:5432/database";
```

```
// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

Oracle

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerOracleDriver" ).newInstanc

// Set the endpoint and port. You can also retrieve it from a key/value pair in the
 secret.
String URL = "jdbc-secretsmanager:oracle:thin:@example.com:1521/ORCL";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

MSSQLServer

```
// Load the JDBC driver
Class.forName( "com.amazonaws.secretsmanager.sql.AWSSecretsManagerMSSQLServerDriver" ).newIn

// Set the endpoint and port. You can also retrieve it from a key/value pair in the
 secret.
String URL = "jdbc-secretsmanager:sqlserver://example.com:1433";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

Db2

```
// Load the JDBC driver
Class.forName( "com.amazonaws.com.amazonaws.secretsmanager.sql.AWSSecretsManagerDb2Driver" )

// Set the endpoint and port. You can also retrieve it from a key/value pair in the
 secret.
String URL = "jdbc-secretsmanager:db2://example.com:50000";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

Redshift

```
// Load the JDBC driver
Class.forName( "com.amazonaws.com.amazonaws.secretsmanager.sql.AWSSecretsManagerRedshiftDriv

// Set the endpoint and port. You can also retrieve it from a key/value pair in the
 secret.
String URL = "jdbc-secretsmanager:redshift://example.com:5439";

// Populate the user property with the secret ARN to retrieve user and password from
 the secret
Properties info = new Properties( );
info.put( "user", "secretId" );

// Establish the connection
conn = DriverManager.getConnection(URL, info);
```

c3p0 연결 풀링을 사용하여 연결 설정

다음 예에서는 드라이버를 사용하여 보안 암호에서 보안 인증 정보 및 연결 정보를 검색하는 `c3p0.properties` 파일로 연결 풀을 설정하는 방법을 보여줍니다. `user`와 `jdbcUrl`에 대해 보안 암호 ID를 입력하여 연결 풀을 구성합니다. 그런 다음 풀에서 연결을 검색하여 다른 데이터베이스 연결과 동일하게 사용할 수 있습니다. 자세한 내용은 Java 설명서 웹 사이트의 JDBC Basics를 참조하세요.

c3p0에 대한 자세한 내용은 Machinery For Change 웹 사이트의 c3p0을 참조하세요.

MySQL

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerMySQLDriver
c3p0.jdbcUrl=secretId
```

PostgreSQL

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerPostgreSQLDriver
c3p0.jdbcUrl=secretId
```

Oracle

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerOracleDriver
c3p0.jdbcUrl=secretId
```

MSSQLServer

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerMSSQLServerDriver
c3p0.jdbcUrl=secretId
```

Db2

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerDb2Driver
c3p0.jdbcUrl=secretId
```

Redshift

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerRedshiftDriver
c3p0.jdbcUrl=secretId
```

c3p0 연결 풀링을 사용하여 엔드포인트 및 포트 지정을 통한 연결 설정

다음 예제는 드라이버를 사용하여 지정한 엔드포인트 및 포트가 있는 시크릿의 자격 증명을 검색하는 `c3p0.properties` 파일로 연결 풀을 설정하는 방법을 보여줍니다. 그런 다음 풀에서 연결을 검색하여 다른 데이터베이스 연결과 동일하게 사용할 수 있습니다. 자세한 내용은 Java 설명서 웹 사이트의 JDBC Basics를 참조하세요.

Amazon RDS 관리형 보안 암호에는 데이터베이스의 엔드포인트 및 포트가 포함되지 않습니다. Amazon RDS에서 관리하는 보안 암호의 마스터 보안 인증을 사용하여 데이터베이스에 연결하려면 이를 해당 코드로 지정해야 합니다.

다른 리전으로 복제된 보안 암호는 리전 데이터베이스에 대한 연결 지연 시간을 줄여 줄 수 있지만, 소스 보안 암호와 다른 연결 정보는 포함하지 않습니다. 각 복제본은 소스 보안 암호의 복사본입니다. 리전 연결 정보를 보안 암호에 저장하려면 엔드포인트에 대한 키/값 쌍 및 리전에 대한 포트 정보를 추가합니다.

MySQL

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerMySQLDriver
c3p0.jdbcUrl=jdbc-secretsmanager:mysql://example.com:3306
```

PostgreSQL

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerPostgreSQLDriver
c3p0.jdbcUrl=jdbc-secretsmanager:postgresql://example.com:5432/database
```

Oracle

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerOracleDriver
c3p0.jdbcUrl=jdbc-secretsmanager:oracle:thin:@example.com:1521/ORCL
```

MSSQLServer

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerMSSQLServerDriver
c3p0.jdbcUrl=jdbc-secretsmanager:sqlserver://example.com:1433
```

Db2

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerDb2Driver
c3p0.jdbcUrl=jdbc-secretsmanager:db2://example.com:50000
```

Redshift

```
c3p0.user=secretId
c3p0.driverClass=com.amazonaws.secretsmanager.sql.AWSSecretsManagerRedshiftDriver
c3p0.jdbcUrl=jdbc-secretsmanager:redshift://example.com:5439
```

자바 AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기

애플리케이션에서는 SDK를 GetSecretValue 호출하거나 BatchGetSecretValue 다른 AWS SDK를 사용하여 암호를 검색할 수 있습니다. 그러나 클라이언트 측 캐싱을 사용하여 보안 암호 값을 캐싱하는 것이 좋습니다. 보안 암호 캐싱은 속도를 향상시키고 비용을 절감합니다.

- 보안 암호에 데이터베이스 자격 증명을 저장하는 경우, Secrets Manager SQL 연결 드라이버를 사용하여 보안 암호의 자격 증명으로 데이터베이스에 연결합니다.
- 다른 유형의 시크릿의 경우 Secrets Manager 자바 기반 캐싱 구성요소를 사용하거나 또는 를 사용하여 SDK를 직접 호출할 수 있습니다. GetSecretValueBatchGetSecretValue

다음 코드 예제는 GetSecretValue의 사용 방법을 보여줍니다.

필요한 권한:secretsmanager:GetSecretValue

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.secretsmanager.SecretsManagerClient;
import software.amazon.awssdk.services.secretsmanager.model.GetSecretValueRequest;
import software.amazon.awssdk.services.secretsmanager.model.GetSecretValueResponse;
import software.amazon.awssdk.services.secretsmanager.model.SecretsManagerException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
```

```
 *
 * We recommend that you cache your secret values by using client-side caching.
 *
 * Caching secrets improves speed and reduces your costs. For more information,
 * see the following documentation topic:
 *
 * https://docs.aws.amazon.com/secretsmanager/latest/userguide/retrieving-secrets.html
 */
public class GetSecretValue {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <secretName>\s

                Where:
                    secretName - The name of the secret (for example, tutorials/
MyFirstSecret).\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String secretName = args[0];
        Region region = Region.US_EAST_1;
        SecretsManagerClient secretsClient = SecretsManagerClient.builder()
                .region(region)
                .build();

        getValue(secretsClient, secretName);
        secretsClient.close();
    }

    public static void getValue(SecretsManagerClient secretsClient, String secretName)
 {
        try {
            GetSecretValueRequest valueRequest = GetSecretValueRequest.builder()
                    .secretId(secretName)
                    .build();

            GetSecretValueResponse valueResponse =
 secretsClient.getSecretValue(valueRequest);
```

```
            String secret = valueResponse.secretString();
            System.out.println(secret);

        } catch (SecretsManagerException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

Python을 사용하여 Secrets Manager의 비밀 값 가져오기

애플리케이션에서는 AWS SDK를 `GetSecretValue` 호출하거나 `BatchGetSecretValue` 사용하여 암호를 검색할 수 있습니다. 그러나 클라이언트 측 캐싱을 사용하여 보안 암호 값을 캐싱하는 것이 좋습니다. 보안 암호 캐싱은 속도를 향상시키고 비용을 절감합니다.

주제

- 클라이언트 측 캐싱과 함께 Python을 사용하여 Secrets Manager의 비밀 값 가져오기
- Python AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기
- Python AWS SDK를 사용하여 Secrets Manager 시크릿 값 일괄 가져오기

클라이언트 측 캐싱과 함께 Python을 사용하여 Secrets Manager의 비밀 값 가져오기

보안 암호를 검색할 때 Secret Manager Python 기반 캐싱 구성 요소를 사용하여 나중에 사용할 수 있도록 캐싱할 수 있습니다. 캐싱된 보안 암호를 검색하는 것이 Secret Manager에서 검색하는 것보다 빠릅니다. Secrets Manager API를 호출하는 데는 비용이 발생하므로 캐시를 사용하면 비용을 줄일 수 있습니다. 암호를 검색할 수 있는 모든 방법은 비밀 찾기을(를) 참조하세요.

캐시 정책은 LRU(가장 오랫동안 사용되지 않음)이므로, 캐시에서 보안 암호를 폐기해야 하는 경우 가장 오랫동안 사용되지 않은 보안 암호가 삭제됩니다. 기본적으로 캐시는 보안 암호를 매시간 새로 고칩니다. 캐시에서 보안 암호를 새로 고치는 주기를 구성하고 보안 암호 검색에 연결하여 더 많은 기능을 추가할 수 있습니다.

캐시 참조가 해제되면 캐시는 가비지 수집을 강제로 적용하지 않습니다. 캐시 구현에는 캐시 무효화가 포함되지 않습니다. 캐시 구현은 캐시 자체에 중점을 두며, 보안을 강화하거나 보안에 초점을 맞추지 않습니다. 캐시에서 항목 암호화와 같은 추가 보안이 필요한 경우 제공된 인터페이스 및 추상 메서드를 사용하세요.

이 구성 요소를 사용하려면 다음이 필요합니다.

- Python 3.6 이상
- botocore 1.12 이상. AWS SDK for Python 및 Botocore를 참조하세요.
- setuptools_scm 3.2 이상. https://pypi.org/project/setuptools-scm/을 참조하세요.

소스 코드를 다운로드하려면 에서 Secrets Manager Python 기반 캐싱 클라이언트 구성 요소를 참조하십시오. GitHub

구성 요소를 설치하려면 다음 명령을 사용합니다.

```
$ pip install aws-secretsmanager-caching
```

필요한 권한:

- `secretsmanager:DescribeSecret`
- `secretsmanager:GetSecretValue`

자세한 정보는 권한 참조을 참조하세요.

레퍼런스

- SecretCache
- SecretCacheConfig
- SecretCacheHook
- @InjectSecretString
- @InjectKeywordedSecretString

Example 보안 암호 검색

다음 예에서는 *mysecret*이라는 보안 암호에 대한 보안 암호 값을 가져오는 방법을 보여 줍니다.

```
import botocore
import botocore.session
from aws_secretsmanager_caching import SecretCache, SecretCacheConfig

client = botocore.session.get_session().create_client('secretsmanager')
cache_config = SecretCacheConfig()
```

```
cache = SecretCache( config = cache_config, client = client)

secret = cache.get_secret_string('mysecret')
```

SecretCache

Secret Manager에서 검색된 암호에 대한 인 메모리 캐시. the section called “get_secret_string” 또는 the section called “get_secret_binary”를 사용하여 캐시에서 보안 암호를 검색합니다. 생성자의 the section called “SecretCacheConfig” 객체에 전달하여 캐시 설정을 구성할 수 있습니다.

예제를 포함한 자세한 내용은 the section called “클라이언트 측 캐싱을 사용하는 Python”을 참조하세요.

```
cache = SecretCache(
    config = the section called “SecretCacheConfig”,
    client = client
)
```

사용 가능한 메서드는 다음과 같습니다.

- get_secret_string
- get_secret_binary

get_secret_string

보안 암호 문자열 값을 검색합니다.

요청 구문

```
response = cache.get_secret_string(
    secret_id='string',
    version_stage='string' )
```

파라미터

- `secret_id` (문자열) -- [필수] 보안 암호의 이름.
- `version_stage` (문자열) -- 검색하려는 보안 암호의 버전. 자세한 내용은 시크릿 버전을 참조하십시오. 기본값은 'AWSCURRENT'입니다.

반환 타입

문자열

get_secret_binary

보안 암호 이진 값을 검색합니다.

요청 구문

```
response = cache.get_secret_binary(
    secret_id='string',
    version_stage='string'
)
```

파라미터

- secret_id (문자열) -- [필수] 보안 암호의 이름.
- version_stage (문자열) -- 검색하려는 보안 암호의 버전. 자세한 내용은 비밀 버전을 참조하십시오. 기본값은 'AWSCURRENT'입니다.

반환 타입

Base64로 인코딩된 문자열

SecretCacheConfig

캐싱된 보안 암호에 대한 최대 캐시 크기 및 유지 시간(TTL)과 같은 the section called “SecretCache”에 대한 캐시 구성 옵션.

파라미터

max_cache_size (int)

최대 캐시 크기. 기본값은 보안 암호 1024개입니다.

exception_retry_delay_base (int)

예외가 발생한 후 요청을 재시도하기 전에 대기할 시간(초). 기본값은 1입니다.

exception_retry_growth_factor (int)pur

실패한 요청 재시도 간의 대기 시간을 계산하는 데 사용할 성장 계수. 기본값은 2입니다.

exception_retry_delay_max (int)

실패한 요청 사이에 대기할 최대 시간(초). 기본값은 3600입니다.

`default_version_stage` (str)

캐싱할 보안 암호의 버전. 자세한 내용은 보안 암호 버전을 참조하세요. 기본값은 `'AWSCURRENT'`입니다.

`secret_refresh_interval` (int)

캐싱된 보안 암호 정보를 새로 고치는 동안 대기할 시간(초). 기본값은 `3600`입니다.

`secret_cache_hook` (SecretCacheHook)

`SecretCacheHook` 추상 클래스의 구현. 기본값은 `None`입니다.

SecretCacheHook

캐시에 저장 중인 보안 암호에 대한 작업을 수행하기 위해 the section called “SecretCache”에 연결되는 인터페이스.

사용 가능한 메서드는 다음과 같습니다.

- put
- get

put

캐시에 저장할 객체를 준비합니다.

요청 구문

```
response = hook.put(
    obj='secret_object'
)
```

파라미터

- `obj` (객체) -- [필수] 보안 암호 또는 보안 암호를 포함하는 객체.

반환 타입

객체

get

캐싱된 객체에서 객체를 추출합니다.

요청 구문

```
response = hook.get(
    obj='secret_object'
)
```

파라미터

- obj (객체) -- [필수] 보안 암호 또는 보안 암호를 포함하는 객체.

반환 타입

객체

@InjectSecretString

이 데코레이터는 첫 번째와 두 번째 인수로 보안 암호 ID 문자열과 the section called “SecretCache”을(를) 기대합니다. 데코레이터는 보안 암호 문자열 값을 반환합니다. 보안 암호는 문자열을 포함해야 합니다.

```
from aws_secretsmanager_caching import SecretCache
from aws_secretsmanager_caching import InjectKeywordedSecretString,
 InjectSecretString

cache = SecretCache()

@InjectSecretString ( 'mysecret' ,  cache )
def function_to_be_decorated( arg1,  arg2,  arg3):
```

@InjectKeywordedSecretString

이 데코레이터는 첫 번째와 두 번째 인수로 보안 암호 ID 문자열과 the section called “SecretCache”을(를) 기대합니다. 나머지 인수는 래핑된 함수의 파라미터를 보안 암호의 JSON 키로 매핑합니다. 보안 암호에는 JSON 구조의 문자열이 포함되어야 합니다.

이 JSON이 포함된 보안 암호의 경우:

```
{
  "username": "saanvi",
  "password": "EXAMPLE-PASSWORD"
}
```

다음 예에서는 보안 암호에서 username과 password에 대한 JSON 값을 추출하는 방법을 보여줍니다.

```
from aws_secretsmanager_caching import SecretCache
  from aws_secretsmanager_caching import InjectKeywordedSecretString,
 InjectSecretString

  cache = SecretCache()

  @InjectKeywordedSecretString ( secret_id = 'mysecret' ,  cache = cache ,
 func_username = 'username' ,  func_password = 'password' )
  def function_to_be_decorated( func_username,  func_password):
       print( 'Do something with the func_username and func_password parameters')
```

Python AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기

애플리케이션에서는 SDK를 GetSecretValue 호출하거나 BatchGetSecretValue 다른 AWS SDK를 사용하여 암호를 검색할 수 있습니다. 그러나 클라이언트 측 캐싱을 사용하여 보안 암호 값을 캐싱하는 것이 좋습니다. 보안 암호 캐싱은 속도를 향상시키고 비용을 절감합니다.

Python 애플리케이션의 경우 Secrets Manager Python 기반 캐싱 구성 요소를 사용하거나 get_secret_value 또는 batch_get_secret_value를 사용하여 SDK를 직접 호출합니다.

다음 코드 예제는 GetSecretValue의 사용 방법을 보여줍니다.

필요한 권한:secretsmanager:GetSecretValue

```
class GetSecretWrapper:
    def __init__(self, secretsmanager_client):
        self.client = secretsmanager_client


    def get_secret(self, secret_name):
        """
        Retrieve individual secrets from AWS Secrets Manager using the get_secret_value
 API.
        This function assumes the stack mentioned in the source code README has been
 successfully deployed.
        This stack includes 7 secrets, all of which have names beginning with
 "mySecret".

        :param secret_name: The name of the secret fetched.
```

```
        :type secret_name: str
        """
        try:
            get_secret_value_response = self.client.get_secret_value(
                SecretId=secret_name
            )
            logging.info("Secret retrieved successfully.")
            return get_secret_value_response["SecretString"]
        except self.client.exceptions.ResourceNotFoundException:
            msg = f"The requested secret {secret_name} was not found."
            logger.info(msg)
            return msg
        except Exception as e:
            logger.error(f"An unknown error occurred: {str(e)}.")
            raise
```

Python AWS SDK를 사용하여 Secrets Manager 시크릿 값 일괄 가져오기

다음 코드 예시에서는 Secrets Manager 보안 암호 값의 배치를 가져오는 방법을 보여줍니다.

필요한 권한:

- `secretsmanager:BatchGetSecretValue`
- `secretsmanager:GetSecretValue`검색하려는 각 비밀번호에 대한 권한.
- 필터를 사용하는 경우 `secretsmanager:ListSecrets`도 있어야 합니다.

권한 정책 예시는 the section called “예: 비밀 값 그룹을 일괄적으로 검색할 수 있는 권한” 섹션을 참조하세요.

> **Important**
>
> 검색 중인 그룹에서 개별 보안 암호를 검색할 수 있는 권한을 거부하는 VPCE 정책이 있는 경우 `BatchGetSecretValue`는 보안 암호 값을 반환하지 않고 오류를 반환합니다.

```
class BatchGetSecretsWrapper:
    def __init__(self, secretsmanager_client):
        self.client = secretsmanager_client
```

```
   def batch_get_secrets(self, filter_name):
       """
       Retrieve multiple secrets from AWS Secrets Manager using the
batch_get_secret_value API.
       This function assumes the stack mentioned in the source code README has been
successfully deployed.
       This stack includes 7 secrets, all of which have names beginning with
"mySecret".

       :param filter_name: The full or partial name of secrets to be fetched.
       :type filter_name: str
       """
       try:
           secrets = []
           response = self.client.batch_get_secret_value(
               Filters=[{"Key": "name", "Values": [f"{filter_name}"]}]
           )
           for secret in response["SecretValues"]:
               secrets.append(json.loads(secret["SecretString"]))
           if secrets:
               logger.info("Secrets retrieved successfully.")
           else:
               logger.info("Zero secrets returned without error.")
           return secrets
       except self.client.exceptions.ResourceNotFoundException:
           msg = f"One or more requested secrets were not found with filter:
{filter_name}"
           logger.info(msg)
           return msg
       except Exception as e:
           logger.error(f"An unknown error occurred:\n{str(e)}.")
           raise
```

.NET을 사용하여 Secrets Manager 비밀 값 가져오기

애플리케이션에서는 SDK를 `GetSecretValue` 호출하거나 `BatchGetSecretValue` 원하는 AWS SDK를 사용하여 암호를 검색할 수 있습니다. 그러나 클라이언트 측 캐싱을 사용하여 보안 암호 값을 캐싱하는 것이 좋습니다. 보안 암호 캐싱은 속도를 향상시키고 비용을 절감합니다.

주제

.NET을 클라이언트측 캐싱과 함께 사용하여 Secrets Manager 비밀 값 가져오기

보안 암호를 검색할 때 Secret Manager .NET 기반 캐싱 구성 요소를 사용하여 나중에 사용할 수 있도록 캐싱할 수 있습니다. 캐싱된 보안 암호를 검색하는 것이 Secret Manager에서 검색하는 것보다 빠릅니다. Secrets Manager API를 호출하는 데는 비용이 발생하므로 캐시를 사용하면 비용을 줄일 수 있습니다. 암호를 검색할 수 있는 모든 방법은 비밀 찾기을(를) 참조하세요.

캐시 정책은 LRU(가장 오랫동안 사용되지 않음)이므로, 캐시에서 보안 암호를 폐기해야 하는 경우 가장 오랫동안 사용되지 않은 보안 암호가 삭제됩니다. 기본적으로 캐시는 보안 암호를 매시간 새로 고칩니다. 캐시에서 보안 암호를 새로 고치는 주기를 구성하고 보안 암호 검색에 연결하여 더 많은 기능을 추가할 수 있습니다.

캐시 참조가 해제되면 캐시는 가비지 수집을 강제로 적용하지 않습니다. 캐시 구현에는 캐시 무효화가 포함되지 않습니다. 캐시 구현은 캐시 자체에 중점을 두며, 보안을 강화하거나 보안에 초점을 맞추지 않습니다. 캐시에서 항목 암호화와 같은 추가 보안이 필요한 경우 제공된 인터페이스 및 추상 메서드를 사용하세요.

이 구성 요소를 사용하려면 다음이 필요합니다.

- .NET Framework 4.6.2 이상 또는 .NET Standard 2.0 이상. Microsoft .NET 웹 사이트의 .NET 다운로드를 참조하세요.
- .NET용 AWS SDK. the section called “AWS SDK”를 참조하세요.

소스 코드를 다운로드하려면 .NET용 캐싱 클라이언트를 참조하십시오. GitHub

캐시를 사용하려면 먼저 캐시를 인스턴스화한 다음 `GetSecretString` 또는 `GetSecretBinary`를 사용하여 보안 암호를 검색합니다. 연속적인 검색에서 캐시는 보안 암호의 캐싱된 사본을 반환합니다.

캐싱 패키지를 가져오려면

- 다음 중 하나를 수행하십시오.
  - 프로젝트 디렉터리에서 다음 .NET CLI 명령을 실행합니다.

```
dotnet add package AWSSDK.SecretsManager.Caching --version 1.0.6
```

- `.csproj` 파일에 다음 패키지 참조를 추가합니다.

```
<ItemGroup>
    <PackageReference Include="AWSSDK.SecretsManager.Caching" Version="1.0.6" /
>
</ItemGroup>
```

필요한 권한:

- `secretsmanager:DescribeSecret`
- `secretsmanager:GetSecretValue`

자세한 정보는 권한 참조을 참조하세요.

레퍼런스

- SecretsManagerCache
- SecretCacheConfiguration
- 저는 SecretCacheHook

Example 보안 암호 검색

다음 코드 예제는 라는 암호를 검색하는 메서드를 보여줍니다. *MySecret*

```
using Amazon.SecretsManager.Extensions.Caching;

namespace LambdaExample
{
    public class CachingExample
    {
        private const string MySecretName ="MySecret";

        private SecretsManagerCache cache = new SecretsManagerCache();

        public async Task<Response>  FunctionHandlerAsync(string input, ILambdaContext
 context)
        {
```

```
            string MySecret = await cache.GetSecretString(MySecretName);

            // Use the secret, return success

        }
    }
}
```

Example TTL(Time To Live) 캐시 새로 고침 기간 구성

다음 코드 예제는 이름이 지정된 *MySecret*암호를 검색하고 TTL 캐시 새로 고침 기간을 24시간으로 설정하는 메서드를 보여줍니다.

```
using Amazon.SecretsManager.Extensions.Caching;

namespace LambdaExample
{
    public class CachingExample
    {
        private const string MySecretName = "MySecret";

        private static SecretCacheConfiguration cacheConfiguration = new
 SecretCacheConfiguration
        {
            CacheItemTTL = 86400000
        };
        private SecretsManagerCache cache = new
 SecretsManagerCache(cacheConfiguration);
        public async Task<Response> FunctionHandlerAsync(string input, ILambdaContext
 context)
        {
            string mySecret = await cache.GetSecretString(MySecretName);

            // Use the secret, return success
        }
    }
}
```

SecretsManagerCache

Secret Manager에서 요청한 보안 암호에 대한 인 메모리 캐시. the section called “GetSecretString” 또는 the section called “GetSecretBinary”를 사용하여 캐시에서 보안 암호를 검색합니다. 생성자의 the section called “SecretCacheConfiguration” 객체에 전달하여 캐시 설정을 구성할 수 있습니다.

예제를 포함한 자세한 내용은 the section called “.NET (클라이언트 측 캐싱 사용)”을 참조하세요.

Constructors

```
public SecretsManagerCache()
```

SecretsManagerCache 객체에 대한 기본 생성자.

```
public SecretsManagerCache(IAmazonSecretsManager secretsManager)
```

제공된 AmazonSecretsManagerClient것을 사용하여 만든 Secrets Manager 클라이언트를 사용하여 새 캐시를 구성합니다. 이 생성자를 사용하여 Secrets Manager 클라이언트를 사용자 지정합니다(예: 특정 리전 또는 엔드포인트를 사용하도록 사용자 지정).

파라미터

secretsManager

에서 AmazonSecretsManagerClient시크릿을 검색할 수 있습니다.

```
public SecretsManagerCache(SecretCacheConfiguration config)
```

제공된 the section called “SecretCacheConfiguration”을 사용하여 새 보안 암호 캐시를 생성합니다. 이 생성자를 사용하여 캐시를 구성합니다(예: 캐싱할 보안 암호 수 및 새로 고침 빈도).

파라미터

config

캐시에 대한 구성 정보를 포함한 the section called “SecretCacheConfiguration”.

```
public SecretsManagerCache(IAmazonSecretsManager secretsManager,
SecretCacheConfiguration config)
```

제공된 AmazonSecretsManagerClient및 a를 사용하여 만든 Secrets Manager 클라이언트를 사용하여 새 캐시를 구성합니다. the section called “SecretCacheConfiguration” 이 생성자를 사용하여 Secrets Manager 클라이언트를 사용자 지정(예: 특정 리전 또는 엔드포인트를 사용하도록 사용자 지정)하거나 캐시를 구성합니다(예: 캐싱할 보안 암호 수 및 새로 고침 빈도).

파라미터

secretsManager

에서 비밀을 검색할 AmazonSecretsManagerClient수 있습니다.

config

캐시에 대한 구성 정보를 포함한 the section called “SecretCacheConfiguration”.

메서드

GetSecretString

```
public async Task<String> GetSecretString(String secretId)
```

Secrets Manager에서 문자열 보안 암호를 검색합니다.

파라미터

SecretId

검색할 보안 암호의 ARN 또는 이름입니다.

GetSecretBinary

```
public async Task<byte[]> GetSecretBinary(String secretId)
```

Secrets Manager에서 이진 보안 암호를 검색합니다.

파라미터

SecretId

검색할 보안 암호의 ARN 또는 이름입니다.

RefreshNowAsync

```
public async Task<bool> RefreshNowAsync(String secretId)
```

Secrets Manager에서 보안 암호 값을 요청하고 변경 사항으로 캐시를 업데이트합니다. 기존 캐시 항목이 없는 경우 새 캐시 항목을 생성합니다. 새로 고침이 성공한 경우 `true`를 반환합니다.

파라미터

SecretId

검색할 보안 암호의 ARN 또는 이름입니다.

GetCachedSecret

```
public SecretCacheItem GetCachedSecret(string secretId)
```

지정한 보안 암호가 캐시에 존재하는 경우 해당 암호에 대한 캐시 항목을 반환합니다. 그렇지 않으면 Secrets Manager에서 보안 암호를 검색하고 새 캐시 항목을 생성합니다.

파라미터

SecretId

검색할 보안 암호의 ARN 또는 이름입니다.

SecretCacheConfiguration

캐싱된 보안 암호에 대한 최대 캐시 크기 및 유지 시간(TTL)과 같은 the section called "SecretsManagerCache"에 대한 캐시 구성 옵션.

속성

CacheItemTTL

```
public uint CacheItemTTL { get; set; }
```

밀리초 단위로 된 캐시 항목에 대한 TTL. 기본값은 3600000ms 또는 1시간입니다. 최댓값은 4294967295ms이며 약 49.7일입니다.

MaxCacheSize

```
public ushort MaxCacheSize { get; set; }
```

최대 캐시 크기. 기본값은 보안 암호 1,024개입니다. 최댓값은 65,535입니다.

VersionStage

```
public string VersionStage { get; set; }
```

캐싱할 보안 암호의 버전. 자세한 내용은 보안 암호 버전을 참조하세요. 기본값은 "AWSCURRENT"입니다.

클라이언트

```
public IAmazonSecretsManager Client { get; set; }
```

에서 비밀을 검색할 AmazonSecretsManagerClient수 있습니다. null인 경우 캐시가 새 클라이언트를 인스턴스화합니다. 기본값은 null입니다.

CacheHook

```
public ISecretCacheHook CacheHook { get; set; }
```

the section called “저는 SecretCacheHook”.

저는 SecretCacheHook

캐시에 저장 중인 보안 암호에 대한 작업을 수행하기 위해 the section called “SecretsManagerCache”에 연결되는 인터페이스.

메서드

Put

```
object Put(object o);
```

캐시에 저장할 객체를 준비합니다.

캐시에 저장할 객체를 반환합니다.

Get

```
object Get(object cachedObject);
```

캐싱된 객체에서 객체를 추출합니다.

캐시에서 객체를 반환합니다.

.NET AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기

애플리케이션에서는 SDK를 GetSecretValue 호출하거나 BatchGetSecretValue 다른 SDK를 사용하여 암호를 검색할 수 있습니다. AWS 그러나 클라이언트 측 캐싱을 사용하여 보안 암호 값을 캐싱하는 것이 좋습니다. 보안 암호 캐싱은 속도를 향상시키고 비용을 절감합니다.

.NET 애플리케이션의 경우 Secrets Manager .NET 기반 캐싱 구성 요소를 사용하거나 GetSecretValue 또는 BatchGetSecretValue를 사용하여 SDK를 직접 호출합니다.

다음 코드 예제는 GetSecretValue의 사용 방법을 보여줍니다.

필요한 권한:secretsmanager:GetSecretValue

```
using System;
using System.IO;
using System.Threading.Tasks;
using Amazon.SecretsManager;
using Amazon.SecretsManager.Model;

/// <summary>
/// This example uses the Amazon Web Service Secrets Manager to retrieve
/// the secret value for the provided secret name.
/// </summary>
public class GetSecretValue
{
    /// <summary>
    /// The main method initializes the necessary values and then calls
    /// the GetSecretAsync and DecodeString methods to get the decoded
    /// secret value for the secret named in secretName.
    /// </summary>
    public static async Task Main()
    {
        string secretName = "<<{{MySecretName}}>>";
        string secret;

        IAmazonSecretsManager client = new AmazonSecretsManagerClient();

        var response = await GetSecretAsync(client, secretName);

        if (response is not null)
        {
            secret = DecodeString(response);

            if (!string.IsNullOrEmpty(secret))
            {
                Console.WriteLine($"The decoded secret value is: {secret}.");
            }
            else
            {
                Console.WriteLine("No secret value was returned.");
```

```
                }
            }
        }

        /// <summary>
        /// Retrieves the secret value given the name of the secret to
        /// retrieve.
        /// </summary>
        /// <param name="client">The client object used to retrieve the secret
        /// value for the given secret name.</param>
        /// <param name="secretName">The name of the secret value to retrieve.</param>
        /// <returns>The GetSecretValueReponse object returned by
        /// GetSecretValueAsync.</returns>
        public static async Task<GetSecretValueResponse> GetSecretAsync(
            IAmazonSecretsManager client,
            string secretName)
        {
            GetSecretValueRequest request = new GetSecretValueRequest()
            {
                SecretId = secretName,
                VersionStage = "AWSCURRENT", // VersionStage defaults to AWSCURRENT if
unspecified.
            };

            GetSecretValueResponse response = null;

            // For the sake of simplicity, this example handles only the most
            // general SecretsManager exception.
            try
            {
                response = await client.GetSecretValueAsync(request);
            }
            catch (AmazonSecretsManagerException e)
            {
                Console.WriteLine($"Error: {e.Message}");
            }

            return response;
        }

        /// <summary>
        /// Decodes the secret returned by the call to GetSecretValueAsync and
        /// returns it to the calling program.
        /// </summary>
```

```
        /// <param name="response">A GetSecretValueResponse object containing
        /// the requested secret value returned by GetSecretValueAsync.</param>
        /// <returns>A string representing the decoded secret value.</returns>
        public static string DecodeString(GetSecretValueResponse response)
        {
            // Decrypts secret using the associated AWS Key Management Service
            // Customer Master Key (CMK.) Depending on whether the secret is a
            // string or binary value, one of these fields will be populated.
            if (response.SecretString is not null)
            {
                var secret = response.SecretString;
                return secret;
            }
            else if (response.SecretBinary is not null)
            {
                var memoryStream = response.SecretBinary;
                StreamReader reader = new StreamReader(memoryStream);
                string decodedBinarySecret =
System.Text.Encoding.UTF8.GetString(Convert.FromBase64String(reader.ReadToEnd()));
                return decodedBinarySecret;
            }
            else
            {
                return string.Empty;
            }
        }
    }
```

Go를 사용하여 Secrets Manager의 비밀 값 가져오기

애플리케이션에서는 SDK를 `GetSecretValue` 호출하거나 `BatchGetSecretValue` 원하는 AWS SDK를 사용하여 암호를 검색할 수 있습니다. 그러나 클라이언트 측 캐싱을 사용하여 보안 암호 값을 캐싱하는 것이 좋습니다. 보안 암호 캐싱은 속도를 향상시키고 비용을 절감합니다.

주제

- 클라이언트측 캐싱과 함께 Go를 사용하여 Secrets Manager 비밀 값 가져오기
- Go AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기

클라이언트측 캐싱과 함께 Go를 사용하여 Secrets Manager 비밀 값 가져오기

보안 암호를 검색할 때 Secret Manager Go 기반 캐싱 구성 요소를 사용하여 나중에 사용할 수 있도록 캐싱할 수 있습니다. 캐싱된 보안 암호를 검색하는 것이 Secret Manager에서 검색하는 것보다 빠릅니다. Secrets Manager API를 호출하는 데는 비용이 발생하므로 캐시를 사용하면 비용을 줄일 수 있습니다. 암호를 검색할 수 있는 모든 방법은 비밀 찾기을(를) 참조하세요.

캐시 정책은 LRU(가장 오랫동안 사용되지 않음)이므로, 캐시에서 보안 암호를 폐기해야 하는 경우 가장 오랫동안 사용되지 않은 보안 암호가 삭제됩니다. 기본적으로 캐시는 보안 암호를 매시간 새로 고칩니다. 캐시에서 보안 암호를 새로 고치는 주기를 구성하고 보안 암호 검색에 연결하여 더 많은 기능을 추가할 수 있습니다.

캐시 참조가 해제되면 캐시는 가비지 수집을 강제로 적용하지 않습니다. 캐시 구현에는 캐시 무효화가 포함되지 않습니다. 캐시 구현은 캐시 자체에 중점을 두며, 보안을 강화하거나 보안에 초점을 맞추지 않습니다. 캐시에서 항목 암호화와 같은 추가 보안이 필요한 경우 제공된 인터페이스 및 추상 메서드를 사용하세요.

이 구성 요소를 사용하려면 다음이 필요합니다.

- AWS Go용 SDK. the section called “AWS SDK” 섹션을 참조하십시오.

소스 코드를 다운로드하려면 Secrets Manager Go 캐싱 클라이언트를 GitHub 참조하십시오.

Go 개발 환경을 설정하려면 Go Programming Language 웹 사이트의 Golang 시작하기를 참조하세요.

필요한 권한:

- `secretsmanager:DescribeSecret`
- `secretsmanager:GetSecretValue`

자세한 정보는 권한 참조을 참조하세요.

레퍼런스

- type Cache
- 유형 CacheConfig
- 유형 CacheHook

Example 보안 암호 검색

다음 코드 예제에서는 보안 암호를 검색하는 Lambda 함수를 보여줍니다.

```
package main

import (
  "github.com/aws/aws-lambda-go/lambda"
  "github.com/aws/aws-secretsmanager-caching-go/secretcache"
)

var (
  secretCache, _ = secretcache.New()
)

func HandleRequest(secretId string) string {
  result, _ := secretCache.GetSecretString(secretId)

  // Use the secret, return success
}

 func main() {
  lambda. Start( HandleRequest)
}
```

type Cache

Secret Manager에서 요청한 보안 암호에 대한 인 메모리 캐시. the section called “GetSecretString” 또는 the section called “GetSecretBinary”를 사용하여 캐시에서 보안 암호를 검색합니다.

다음 예제에서는 캐시 설정을 구성하는 방법을 보여줍니다.

```
// Create a custom secretsmanager client
client := getCustomClient()

// Create a custom CacheConfig struct
config := secretcache. CacheConfig{
    MaxCacheSize:  secretcache.DefaultMaxCacheSize + 10,
    VersionStage:  secretcache.DefaultVersionStage,
    CacheItemTTL:  secretcache.DefaultCacheItemTTL,
}

// Instantiate the cache
```

```
cache, _ := secretcache.New(
    func( c *secretcache.Cache) {  c. CacheConfig = config },
    func( c *secretcache.Cache) {  c. Client = client },
)
```

예제를 포함한 자세한 내용은 the section called “클라이언트측 캐싱을 활용하세요.”을 참조하세요.

메서드

New

```
func New(optFns ...func(*Cache)) (*Cache, error)
```

New는 함수 옵션을 사용하여 보안 암호 캐시를 구성하고, 그렇지 않으면 기본값을 사용합니다. 새 세션에서 SecretsManager 클라이언트를 초기화합니다. 기본값으로 초기화합니다 CacheConfig . LRU 캐시를 기본 최대 크기로 초기화합니다.

GetSecretString

```
func (c *Cache) GetSecretString(secretId string) (string, error)
```

GetSecretString 캐시에서 지정된 비밀 ID의 비밀 문자열 값을 가져옵니다. 작업이 실패하면 보안 암호 문자열과 오류를 반환합니다.

GetSecretStringWithStage

```
func (c *Cache) GetSecretStringWithStage(secretId string, versionStage
string) (string, error)
```

GetSecretStringWithStage 지정된 비밀 ID 및 버전 단계의 캐시에서 비밀 문자열 값을 가져옵니다. 작업이 실패하면 보안 암호 문자열과 오류를 반환합니다.

GetSecretBinary

```
func (c *Cache) GetSecretBinary(secretId string) ([]byte, error) {
```

GetSecretBinary 캐시에서 지정된 비밀 ID의 비밀 바이너리 값을 가져옵니다. 보안 암호 이진 값을 반환하고 작업이 실패하면 오류를 반환합니다.

GetSecretBinaryWithStage

```
func (c *Cache) GetSecretBinaryWithStage(secretId string, versionStage
string) ([]byte, error)
```

GetSecretBinaryWithStage 지정된 시크릿 ID 및 버전 스테이지에 대해 캐시에서 시크릿 바이너리 값을 가져옵니다. 보안 암호 이진 값을 반환하고 작업이 실패하면 오류를 반환합니다.

유형 CacheConfig

캐싱된 보안 암호에 대한 최대 캐시 크기, 기본 버전 단계 및 유지 시간(TTL)과 같은 캐시에 대한 캐시 구성 옵션.

```
type CacheConfig struct {

    // The maximum cache size. The default is 1024 secrets.
    MaxCacheSize int

    // The TTL of a cache item in nanoseconds. The default is
    // 3.6e10^12 ns or 1 hour.
    CacheItemTTL int64

    // The version of secrets that you want to cache. The default
    // is "AWSCURRENT".
    VersionStage string

    // Used to hook in-memory cache updates.
    Hook CacheHook
}
```

유형 CacheHook

캐시에 저장 중인 보안 암호에 대한 작업을 수행하기 위해 캐시에 연결되는 인터페이스.

메서드

Put

`Put(data interface{}) interface{}`

캐시에 저장할 객체를 준비합니다.

Get

`Get(data interface{}) interface{}`

캐싱된 객체에서 객체를 추출합니다.

Go AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기

애플리케이션에서는 SDK를 GetSecretValue 호출하거나 BatchGetSecretValue 원하는 AWS SDK를 사용하여 암호를 검색할 수 있습니다. 그러나 클라이언트 측 캐싱을 사용하여 보안 암호 값을 캐싱하는 것이 좋습니다. 보안 암호 캐싱은 속도를 향상시키고 비용을 절감합니다.

Go 애플리케이션의 경우 Secrets Manager Go 기반 캐싱 구성 요소를 사용하거나 GetSecretValue 또는 BatchGetSecretValue를 사용하여 SDK를 직접 호출합니다.

다음 코드 예제에서는 Secrets Manager 보안 암호 값을 가져오는 방법을 보여줍니다.

필요한 권한:secretsmanager:GetSecretValue

```
 // Use this code snippet in your app.
 // If you need more information about configurations or implementing the sample code,
visit the AWS docs:
 // https://aws.github.io/aws-sdk-go-v2/docs/getting-started/

 import (
   "context"
   "log"

   "github.com/aws/aws-sdk-go-v2/aws"
   "github.com/aws/aws-sdk-go-v2/config"
   "github.com/aws/aws-sdk-go-v2/service/secretsmanager"
 )

 func main() {
   secretName := "<<{{MySecretName}}>>"
   region := "<<{{MyRegionName}}>>"

   config, err := config.LoadDefaultConfig(context.TODO(), config.WithRegion(region))
   if err != nil {
     log.Fatal(err)
   }

   // Create Secrets Manager client
   svc := secretsmanager.NewFromConfig(config)

   input := &secretsmanager.GetSecretValueInput{
     SecretId:     aws.String(secretName),
     VersionStage: aws.String("AWSCURRENT"), // VersionStage defaults to AWSCURRENT if
unspecified
```

```
    }

    result, err := svc.GetSecretValue(context.TODO(), input)
    if err != nil {
      // For a list of exceptions thrown, see
      // https://<<{{DocsDomain}}>>/secretsmanager/latest/apireference/
API_GetSecretValue.html
      log.Fatal(err.Error())
    }

    // Decrypts secret using the associated KMS key.
    var secretString string = *result.SecretString

    // Your code goes here.
  }
```

C++ AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기

C++ 애플리케이션의 경우 또는 를 사용하여 SDK를 직접 호출하십시오.
GetSecretValueBatchGetSecretValue

다음 코드 예제에서는 Secrets Manager 보안 암호 값을 가져오는 방법을 보여줍니다.

필요한 권한:`secretsmanager:GetSecretValue`

```
//! Retrieve an AWS Secrets Manager encrypted secret.
/*!
  \param secretID: The ID for the secret.
  \return bool: Function succeeded.
 */
bool AwsDoc::SecretsManager::getSecretValue(const Aws::String &secretID,
                                            const Aws::Client::ClientConfiguration
 &clientConfiguration) {
    Aws::SecretsManager::SecretsManagerClient
 secretsManagerClient(clientConfiguration);

    Aws::SecretsManager::Model::GetSecretValueRequest request;
    request.SetSecretId(secretID);

    Aws::SecretsManager::Model::GetSecretValueOutcome getSecretValueOutcome =
 secretsManagerClient.GetSecretValue(
            request);
    if (getSecretValueOutcome.IsSuccess()) {
```

```
        std::cout << "Secret is: "
                  << getSecretValueOutcome.GetResult().GetSecretString() << std::endl;
    }
    else {
        std::cerr << "Failed with Error: " << getSecretValueOutcome.GetError()
                  << std::endl;
    }

    return getSecretValueOutcome.IsSuccess();
}
```

JavaScript AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기

JavaScript 애플리케이션의 경우 또는 를 사용하여 SDK를 직접 호출하십시오 `getSecretValue`. `batchGetSecretValue`

다음 코드 예제에서는 Secrets Manager 보안 암호 값을 가져오는 방법을 보여줍니다.

필요한 권한:`secretsmanager:GetSecretValue`

```
import {
  GetSecretValueCommand,
  SecretsManagerClient,
} from "@aws-sdk/client-secrets-manager";

export const getSecretValue = async (secretName = "SECRET_NAME") => {
  const client = new SecretsManagerClient();
  const response = await client.send(
    new GetSecretValueCommand({
      SecretId: secretName,
    }),
  );
  console.log(response);
  // {
  //   '$metadata': {
  //     httpStatusCode: 200,
  //     requestId: '584eb612-f8b0-48c9-855e-6d246461b604',
  //     extendedRequestId: undefined,
  //     cfId: undefined,
  //     attempts: 1,
  //     totalRetryDelay: 0
```

```
  //   },
  //   ARN: 'arn:aws:secretsmanager:us-east-1:xxxxxxxxxxxx:secret:binary-
secret-3873048-xxxxxx',
  //   CreatedDate: 2023-08-08T19:29:51.294Z,
  //   Name: 'binary-secret-3873048',
  //   SecretBinary: Uint8Array(11) [
  //      98, 105, 110, 97, 114,
  //     121,  32, 100, 97, 116,
  //      97
  //   ],
  //   VersionId: '712083f4-0d26-415e-8044-16735142cd6a',
  //   VersionStages: [ 'AWSCURRENT' ]
  // }

  if (response.SecretString) {
    return response.SecretString;
  }

  if (response.SecretBinary) {
    return response.SecretBinary;
  }
};
```

Kotlin AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기

Kotlin 애플리케이션의 경우 또는 를 사용하여 SDK를 직접 호출하세요. GetSecretValueBatchGetSecretValue

다음 코드 예제에서는 Secrets Manager 보안 암호 값을 가져오는 방법을 보여줍니다.

필요한 권한:`secretsmanager:GetSecretValue`

```
suspend fun getValue(secretName: String?) {
    val valueRequest =
        GetSecretValueRequest {
            secretId = secretName
        }

    SecretsManagerClient { region = "us-east-1" }.use { secretsClient ->
        val response = secretsClient.getSecretValue(valueRequest)
        val secret = response.secretString
        println("The secret value is $secret")
    }
```

```
}
```

PHP AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기

PHP 애플리케이션의 경우 GetSecretValue 또는 BatchGetSecretValue를 사용하여 SDK를 직접 호출합니다.

다음 코드 예제에서는 Secrets Manager 보안 암호 값을 가져오는 방법을 보여줍니다.

필요한 권한:secretsmanager:GetSecretValue

```
<?php

  /**
    * Use this code snippet in your app.
    *
    * If you need more information about configurations or implementing the sample
 code, visit the AWS docs:
    * https://aws.amazon.com/developer/language/php/
    */

  require 'vendor/autoload.php';

  use Aws\SecretsManager\SecretsManagerClient;
  use Aws\Exception\AwsException;

  /**
    * This code expects that you have AWS credentials set up per:
    * https://<<{{DocsDomain}}>>/sdk-for-php/v3/developer-guide/guide_credentials.html
    */

  // Create a Secrets Manager Client
  $client = new SecretsManagerClient([
      'profile' => 'default',
      'version' => '2017-10-17',
      'region' => '<<{{MyRegionName}}>>',
  ]);

  $secret_name = '<<{{MySecretName}}>>';

  try {
      $result = $client->getSecretValue([
```

```
          'SecretId' => $secret_name,
      ]);
  } catch (AwsException $e) {
      // For a list of exceptions thrown, see
      // https://<<{{DocsDomain}}>>/secretsmanager/latest/apireference/
API_GetSecretValue.html
      throw $e;
  }

  // Decrypts secret using the associated KMS key.
  $secret = $result['SecretString'];

  // Your code goes here
```

루비 AWS SDK를 사용하여 Secrets Manager 시크릿 값 가져오기

Ruby 애플리케이션의 경우get_secret_value 또는 batch_get_secret_value를 사용하여 SDK를 직접 호출합니다.

다음 코드 예제에서는 Secrets Manager 보안 암호 값을 가져오는 방법을 보여줍니다.

필요한 권한:secretsmanager:GetSecretValue

```
  # Use this code snippet in your app.
  # If you need more information about configurations or implementing the sample code,
 visit the AWS docs:
  # https://aws.amazon.com/developer/language/ruby/

  require 'aws-sdk-secretsmanager'

  def get_secret
    client = Aws::SecretsManager::Client.new(region: '<<{{MyRegionName}}>>')

    begin
      get_secret_value_response = client.get_secret_value(secret_id:
 '<<{{MySecretName}}>>')
    rescue StandardError => e
      # For a list of exceptions thrown, see
      # https://<<{{DocsDomain}}>>/secretsmanager/latest/apireference/
API_GetSecretValue.html
      raise e
    end
```

```
    secret = get_secret_value_response.secret_string
    # Your code goes here.
  end
```

Rust AWS SDK를 사용하여 Secrets Manager의 시크릿 값 가져오기

Rust 애플리케이션의 경우 또는 를 사용하여 SDK를 직접 호출하세요 GetSecretValue. BatchGetSecretValue

다음 코드 예제에서는 Secrets Manager 보안 암호 값을 가져오는 방법을 보여줍니다.

필요한 권한:`secretsmanager:GetSecretValue`

```
async fn show_secret(client: &Client, name: &str) -> Result<(), Error> {
    let resp = client.get_secret_value().secret_id(name).send().await?;

    println!("Value: {}", resp.secret_string().unwrap_or("No value!"));

    Ok(())
}
```

다음을 사용하여 비밀 값을 가져오세요. AWS CLI

필요한 권한:`secretsmanager:GetSecretValue`

Example 보안 암호의 암호화된 암호 값 검색

다음 `get-secret-value` 예에서는 현재 보안 암호 값을 가져옵니다.

```
aws secretsmanager get-secret-value \
    --secret-id MyTestSecret
```

Example 이전 보안 암호 값 검색

다음 `get-secret-value` 예에서는 이전 보안 암호 값을 가져옵니다.

```
aws secretsmanager get-secret-value \
        --secret-id MyTestSecret
        --version-stage AWSPREVIOUS
```

를 사용하여 일괄적으로 비밀 그룹을 가져옵니다. AWS CLI

필요한 권한:

- `secretsmanager:BatchGetSecretValue`
- `secretsmanager:GetSecretValue`검색하려는 각 비밀에 대한 권한.
- 필터를 사용하는 경우 `secretsmanager:ListSecrets`도 있어야 합니다.

권한 정책 예시는 the section called “예: 비밀 값 그룹을 일괄적으로 검색할 수 있는 권한” 섹션을 참조하세요.

> **Important**
>
> 검색 중인 그룹에서 개별 보안 암호를 검색할 수 있는 권한을 거부하는 VPCE 정책이 있는 경우 `BatchGetSecretValue`는 보안 암호 값을 반환하지 않고 오류를 반환합니다.

Example 이름별로 나열된 보안 암호 그룹의 보안 암호 값을 검색합니다.

다음 `batch-get-secret-value` 예시에서는 현재 보안 암호 값을 가져옵니다.

```
aws secretsmanager batch-get-secret-value \
          --secret-id-list MySecret1 MySecret2 MySecret3
```

Example 필터로 선택한 보안 암호 그룹의 보안 암호 값을 검색합니다.

다음 `batch-get-secret-value` 예시에서는 태그가 "Test"인 보안 암호의 보안 암호 값을 가져옵니다.

```
aws secretsmanager batch-get-secret-value \
          --filters Key="tag-key",Values="Test"
```

AWS 콘솔을 사용하여 비밀 값을 가져오세요.

보안 암호를 검색하려면(콘솔)

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.

2. 보안 암호 목록에서 검색할 보안 암호를 선택합니다.
3. 보안 암호 값(Secret value) 섹션에서 보안 암호 값 검색(Retrieve secret value)을 선택합니다.

   Secrets Manager는 보안 암호의 현재 버전(AWSCURRENT) 을 표시합니다. 보안 암호의 다른 버전(예AWSPREVIOUS: 사용자 지정 레이블이 지정된 버전)을 보려면 the section called “AWS CLI”을(를) 사용하세요.

에서 AWS Secrets Manager 시크릿 사용 AWS Batch

AWS Batch 에서 배치 컴퓨팅 워크로드를 AWS 클라우드실행하는 데 도움이 됩니다. 를 사용하면 민감한 데이터를 AWS Secrets Manager 비밀로 저장한 다음 작업 정의에서 참조하여 민감한 데이터를 작업에 주입할 수 있습니다. AWS Batch자세한 내용은 Secrets Manager를 사용해 민감한 데이터 지정을 참조하십시오.

AWS CloudFormation 리소스에서 AWS Secrets Manager 비밀 가져오기

AWS CloudFormation를 사용하면 암호를 검색하여 다른 AWS CloudFormation 리소스에서 사용할 수 있습니다. 일반적인 시나리오는 먼저 Secrets Manager에서 생성한 암호로 보안 암호를 생성한 새 데이터베이스의 자격 증명으로 사용할 보안 암호에서 사용자 이름과 암호를 검색합니다. 를 사용하여 암호를 만드는 방법에 대한 자세한 내용은 AWS CloudFormation을 참조하십시오AWS CloudFormation.

AWS CloudFormation 템플릿에서 비밀을 검색하려면 동적 참조를 사용합니다. 스택을 생성하면 동적 참조가 비밀 값을 AWS CloudFormation 리소스로 가져오므로 비밀 정보를 하드코딩하지 않아도 됩니다. 대신, 이름이나 ARN을 사용하여 보안 암호를 참조합니다. 모든 리소스 속성의 보안 암호에 동적 참조를 사용할 수 있습니다. `AWS::CloudFormation::Init`와(과) 같은 리소스 메타데이터의 보안 암호에 동적 참조를 사용할 수 없습니다. 이렇게 하면 콘솔에 보안 암호 값이 표시되기 때문입니다.

보안 암호에 대한 동적 참조 패턴은 다음과 같습니다.

```
{{resolve:secretsmanager:secret-id:SecretString:json-key:version-stage:version-id}}
```

secret-id

보안 암호의 이름 또는 ARN입니다. AWS 계정의 비밀에 접근하려면 시크릿 이름을 사용하면 됩니다. 다른 AWS 계정의 비밀에 액세스하려면 해당 비밀의 ARN을 사용하십시오.

json-key(선택)

검색하고자 하는 값을 보유한 키-값 페어의 키 이름입니다. `json-keya`를 지정하지 않으면 전체 비밀 AWS CloudFormation 텍스트를 검색합니다. 이 세그먼트에는 콜론 문자(`:`)가 포함되지 않을 수 있습니다.

version-stage(선택)

사용하려는 보안 암호의 버전입니다. Secrets Manager는 교체 프로세스 도중 다른 버전을 추적하는 데 스테이징 레이블을 사용합니다. `version-stage`을 사용하는 경우 `version-id`를 지정하지 마세요. `version-stage` 또는 `version-id`를 지정하지 않은 경우 기본값은 `AWSCURRENT` 버전입니다. 이 세그먼트에는 콜론 문자(`:`)가 포함되지 않을 수 있습니다.

version-id(선택)

사용하고자 하는 보안 암호의 버전에 대한 고유 식별자입니다. `version-id`을 지정할 경우 `version-stage`을 지정하지 마세요. `version-stage` 또는 `version-id`를 지정하지 않은 경우 기본값은 `AWSCURRENT` 버전입니다. 이 세그먼트에는 콜론 문자(`:`)가 포함되지 않을 수 있습니다.

자세한 내용은 동적 참조를 사용하여 Secrets Manager 보안 암호 지정 섹션을 참조하세요.

> **Note**
>
> 백슬래시를 최종 (`\`) 값으로 사용하여 동적 참조를 만들지 마십시오. AWS CloudFormation 이러한 참조를 해결할 수 없어 리소스 장애가 발생합니다.

아마존 엘라스틱 쿠버네티스 서비스에서 AWS Secrets Manager 시크릿 사용하기

Secrets Manager의 시크릿을 Amazon EKS 포드에 마운트된 파일로 표시하려면 쿠버네티스 AWS 시크릿 스토어 CSI 드라이버용 시크릿 및 구성 공급자 (ASCP) 를 사용하면 됩니다. ASCP는 아마존 EC2 노드 그룹을 실행하는 아마존 Elastic Kubernetes Service (아마존 EKS) 1.17+와 함께 작동합니다. AWS Fargate 노드 그룹은 지원되지 않습니다. ASCP를 사용하면 Secrets Manager 보안 암호를 저장 및 관리한 후 Amazon EKS에서 실행되는 워크로드를 통해 해당 검색할 수 있습니다. 암호에 JSON 형식의 여러 키/값 쌍이 포함되어 있는 경우 Amazon EKS에서 탑재할 키/값 쌍을 선택할 수 있습니다. ASCP는 JMEPath 구문을 사용하여 보안 암호의 키/값 쌍을 쿼리할 수 있습니다. ASCP는 파라미터 스토어 파라미터로도 작동합니다.

프라이빗 Amazon EKS 클러스터를 사용하는 경우 클러스터가 있는 VPC에 Secrets Manager 엔드포인트가 있어야 합니다. 보안 암호 스토어 CSI 드라이버는 엔드포인트를 사용하여 Secrets Manager를 호출합니다. VPC에서 엔드포인트를 만드는 데 대한 자세한 내용은 VPC 엔드포인트을(를) 참조하세요.

보안 암호에 대해 Secrets Manager 자동 교체를 사용하는 경우 보안 암호 스토어 CSI 드라이버 교체 조정자 기능을 사용하여 Secrets Manager에서 최신 보안 암호를 검색할 수도 있습니다. 자세한 내용은 탑재된 콘텐츠 및 동기화된 Kubernetes 보안 암호의 자동 교체를 참조하세요.

주제

- 1단계: 액세스 제어 설정
- 2단계: ASCP 설치 및 구성
- 3단계: 마운트할 시크릿 식별
- 4단계: Amazon EKS 포드에 시크릿을 파일로 마운트합니다.
- 문제 해결
- SecretProviderClass

1단계: 액세스 제어 설정

ASCP는 Amazon EKS 포드 ID를 검색하고 이를 IAM 역할로 교환합니다. IAM 정책에서 해당 IAM 역할에 대한 권한을 설정합니다. ASCP가 IAM 역할을 맡으면 사용자가 승인한 비밀에 액세스할 수 있습니다. 다른 컨테이너는 IAM 역할과 연결하지 않는 한 보안 암호에 액세스할 수 없습니다.

포드와 관련된 지역 및 IAM 역할을 조회하기 위한 ASCP의 호출이 Kubernetes에 의해 제한되는 경우 2단계에 표시된 대로 을 사용하여 제한 할당량을 변경할 수 있습니다. `helm install`

Secrets Manager에서 Amazon EKS 포드에 비밀에 대한 액세스 권한을 부여하려면

1. 포드가 액세스하는 데 필요한 비밀을 `secretsmanager:GetSecretValue` 부여하고 `secretsmanager:DescribeSecret` 권한을 부여하는 권한 정책을 생성하십시오. 정책 예제는 the section called “예: 개별 비밀을 읽고 설명할 수 있는 권한”을 참조하세요.
2. 아직 없는 경우 클러스터에 대한 IAM OpenID Connect(OIDC) 공급자를 생성합니다. 자세한 내용은 Amazon EKS 사용 설명서의 클러스터용 IAM OIDC 공급자 생성을 참조하십시오.
3. 서비스 계정의 IAM 역할을 생성하고 정책을 해당 역할에 연결합니다. 자세한 내용은 Amazon EKS 사용 설명서의 서비스 계정에 대한 IAM 역할 생성을 참조하십시오.

4. 프라이빗 Amazon EKS 클러스터를 사용하는 경우 클러스터가 속한 VPC에 엔드포인트가 있는지 확인하십시오. AWS STS 엔드포인트 생성에 대한 자세한 내용은 사용 설명서의 인터페이스 VPC 엔드포인트를 참조하십시오.AWS Identity and Access Management

2단계: ASCP 설치 및 구성

ASCP는 secrets-store-csi-provider-aws GitHub 리포지토리에서 사용할 수 있습니다. 리포지토리에는 보안 암호를 생성하고 탑재하기 위한 예제 YAML 파일도 포함되어 있습니다.

설치 중에 FIPS 엔드포인트를 사용하도록 ASCP를 구성할 수 있습니다. 엔드포인트 목록은 the section called “Secrets Manager 엔드포인트” 섹션을 참조하세요.

헬름을 사용하여 ASCP를 설치하려면

1. 리포지토리가 최신 차트를 가리키도록 하려면 다음을 사용하십시오. `helm repo update.`
2. 시크릿 스토어 CSI 드라이버 차트를 추가하세요.

```
helm repo add secrets-store-csi-driver https://kubernetes-sigs.github.io/secrets-store-csi-driver/charts
```

3. 차트를 설치합니다. 스로틀링을 구성하려면 다음 플래그를 추가하십시오. `--set-json 'k8sThrottlingParams={"qps": "<number of queries per second>", "burst": "<number of queries per second>"}'`

```
helm install -n kube-system csi-secrets-store secrets-store-csi-driver/secrets-store-csi-driver
```

4. ASCP 차트를 추가합니다.

```
helm repo add aws-secrets-manager https://aws.github.io/secrets-store-csi-driver-provider-aws
```

5. 차트를 설치합니다. FIPS 엔드포인트를 사용하려면 다음 플래그를 추가합니다. `--set useFipsEndpoint=true`

```
helm install -n kube-system secrets-provider-aws aws-secrets-manager/secrets-store-csi-driver-provider-aws
```

리포지토리의 YAML을 사용하여 설치하려면

- 다음 명령을 사용합니다.

```
helm repo add secrets-store-csi-driver https://kubernetes-sigs.github.io/secrets-store-csi-driver/charts
helm install -n kube-system csi-secrets-store secrets-store-csi-driver/secrets-store-csi-driver
kubectl apply -f https://raw.githubusercontent.com/aws/secrets-store-csi-driver-provider-aws/main/deployment/aws-provider-installer.yaml
```

3단계: 마운트할 시크릿 식별

ASCP가 Amazon EKS에 파일 시스템의 파일로 탑재하는 보안 암호를 확인하려면 the section called "SecretProviderClass" YAML 파일을 생성합니다. 에는 마운트할 비밀과 이를 마운트할 파일 이름이 SecretProviderClass 나열됩니다. 이 SecretProviderClass는 참조하는 Amazon EKS 포드와 동일한 네임스페이스에 있어야 합니다.

다음 예시는 SecretProviderClass을(를) 사용하여 탑재할 보안 암호를 설명하는 방법과 Amazon EKS 포드에 탑재된 파일의 이름을 지정하는 방법을 보여줍니다.

예:

- 예: 이름 또는 ARN으로 보안 암호 탑재
- 예: 보안 암호에서 키/값 쌍 탑재
- 예: 다중 리전 보안 암호에 대한 장애 조치 리전 정의
- 예: 탑재할 장애 조치 암호 선택

예: 이름 또는 ARN으로 보안 암호 탑재

다음 예시는 Amazon EKS에 세 개의 파일을 탑재하는 SecretProviderClass을(를) 보여줍니다:

1. 전체 ARN에 의해 지정된 보안 암호입니다.
2. 이름으로 지정된 보안 암호입니다.
3. 보안 암호의 특정 버전입니다.

```
apiVersion: secrets-store.csi.x-k8s.io/v1
```

```
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
    objects: |
        - objectName: "arn:aws:secretsmanager:us-east-2:111122223333:secret:MySecret2-
d4e5f6"
        - objectName: "MySecret3"
          objectType: "secretsmanager"
        - objectName: "MySecret4"
          objectType: "secretsmanager"
          objectVersionLabel: "AWSCURRENT"
```

예: 보안 암호에서 키/값 쌍 탑재

다음 예시는 Amazon EKS에 세 개의 파일을 탑재하는 SecretProviderClass을(를) 보여줍니다:

1. 전체 ARN에 의해 지정된 보안 암호입니다.
2. 동일한 보안 암호의 username 키/값 쌍입니다.
3. 동일한 보안 암호의 password 키/값 쌍입니다.

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
    objects: |
      - objectName: "arn:aws:secretsmanager:us-east-2:111122223333:secret:MySecret-
a1b2c3"
        jmesPath:
            - path: username
              objectAlias: dbusername
            - path: password
              objectAlias: dbpassword
```

예: 다중 리전 보안 암호에 대한 장애 조치 리전 정의

연결 중단 시 또는 재해 복구 구성을 위한 가용성을 제공하기 위해 ASCP는 보조 리전에서 보안 암호를 검색하는 자동 장애 조치 기능을 지원합니다.

다음 예시는 여러 리전에 복제된 보안 암호를 검색하는 SecretProviderClass을(를) 보여줍니다. 이 예시에서 ASCP는us-east-1 및 us-east-2 모두에서 보안 암호 검색을 시도합니다. 두 리전 중 하나가 예를 들어 인증 문제에 대해 4xx 오류를 반환하는 경우 ASCP는 어느 암호도 탑재하지 않습니다. us-east-1에서 보안 암호가 성공적으로 검색되면 ASCP는 해당 암호 값을 탑재합니다. us-east-1에서는 보안 암호가 성공적으로 검색되지 않고 us-east-2에서 성공적으로 검색되는 경우 ASCP는 해당 암호 값을 탑재합니다.

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
    region: us-east-1
    failoverRegion: us-east-2
    objects: |
      - objectName: "MySecret"
```

예: 탑재할 장애 조치 암호 선택

다음 예시는 장애 조치 시 탑재할 보안 암호를 지정하는 SecretProviderClass을(를) 보여 줍니다. 장애 조치 암호는 복제본이 아닙니다. 이 예시에서 ASCP는 objectName에서 지정한 두 개의 보안 암호를 검색하려고 시도합니다. 둘 중 하나가 예를 들어 인증 문제에 대해 4xx 오류를 반환하는 경우 ASCP는 어느 암호도 탑재하지 않습니다. us-east-1에서 보안 암호가 성공적으로 검색되면 ASCP는 해당 암호 값을 탑재합니다. us-east-1에서는 보안 암호가 성공적으로 검색되지 않고 us-east-2에서 성공적으로 검색되는 경우 ASCP는 해당 암호 값을 탑재합니다. Amazon EKS에 탑재된 파일의 이름은 MyMountedSecret입니다.

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
```

```
    region: us-east-1
    failoverRegion: us-east-2
    objects: |
      - objectName: "arn:aws:secretsmanager:us-east-1:111122223333:secret:MySecret-
a1b2c3"
        objectAlias: "MyMountedSecret"
        failoverObject:
          - objectName: "arn:aws:secretsmanager:us-
east-2:111122223333:secret:MyFailoverSecret-d4e5f6"
```

4단계: Amazon EKS 포드에 시크릿을 파일로 마운트합니다.

Amazon EKS에 시크릿을 마운트하려면

1. 명령을 SecretProviderClass kubectl apply -f ExampleSecretProviderClass.yaml 사용하여 포드에 적용합니다.
2. 명령어를 사용하여 포드를 kubectl apply -f ExampleDeployment.yaml 배포합니다.
3. ASCP는 파일을 마운트합니다.

문제 해결

포드 배포를 설명하여 대부분의 오류를 볼 수 있습니다.

컨테이너에 대한 오류 메시지 확인

1. 다음 명령을 사용하여 포드 이름 목록을 가져옵니다. 기본 네임스페이스를 사용하지 않는 경우에는 -n <NAMESPACE>를 사용합니다.

   ```
   kubectl get pods
   ```

2. 포드를 설명하기 위해 다음 명령에서 *<PODID>*에 대해 이전 단계에서 찾은 포드의 포드 ID를 사용합니다. 기본 네임스페이스를 사용하지 않는 경우에는 -n <NAMESPACE>를 사용합니다.

   ```
   kubectl describe pod/<PODID>
   ```

ASCP에 대한 오류를 확인하려면

- 제공자 로그에서 자세한 정보를 찾으려면 다음 명령에서 <PODID>csi-secrets-store-provider-aws 포드의 ID를 사용하십시오.

```
kubectl -n kube-system get pods
kubectl -n kube-system logs pod/<PODID>
```

SecretProviderClass

YAML을 사용하여 ASCP를 사용하여 Amazon EKS에 어떤 시크릿을 마운트할지 설명합니다. 예를 보려면 the section called “이름 또는 ARN으로 보안 암호 탑재”을 참조하세요.

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
   name: <NAME>
spec:
  provider: aws
  parameters:
    region:
    failoverRegion:
    pathTranslation:
    objects:
```

필드 parameters에는 탑재 요청의 세부 정보가 포함됩니다:

region

(선택 사항) 시크릿 AWS 리전 . 이 필드를 사용하지 않는 경우 ASCP는 노드의 주석에서 지역을 조회합니다. 이 조회는 탑재 요청에 오버헤드를 추가하므로 많은 수의 포드를 사용하는 클러스터에 리전을 제공하는 것이 좋습니다.

또한 failoverRegion을(를) 지정하는 경우 ASCP는 두 리전 모두에서 보안 암호 검색을 시도합니다. 두 리전 중 하나가 예를 들어 인증 문제에 대해 4xx 오류를 반환하는 경우 ASCP는 어느 암호도 탑재하지 않습니다. region에서 보안 암호가 성공적으로 검색되면 ASCP는 해당 암호 값을 탑재합니다. region에서는 보안 암호가 성공적으로 검색되지 않고 failoverRegion에서 성공적으로 검색되는 경우 ASCP는 해당 암호 값을 탑재합니다.

failoverRegion

(선택 사항) 이 필드를 포함하는 경우 ASCP는 region 및 이 필드에 정의된 리전에서 보안 암호를 검색하려고 시도합니다. 두 리전 중 하나가 예를 들어 인증 문제에 대해 4xx 오류를 반환하는 경우 ASCP는 어느 암호도 탑재하지 않습니다. region에서 보안 암호가 성공적으로 검색되면 ASCP는 해당 암호 값을 탑재합니다. region에서는 보안 암호가 성공적으로 검색되지 않고

failoverRegion에서 성공적으로 검색되는 경우 ASCP는 해당 암호 값을 탑재합니다. 이 필드 사용 방법의 예는 다중 리전 보안 암호에 대한 장애 조치 리전 정의 섹션을 참조하세요.

pathTranslation

(선택 사항) Amazon EKS의 파일 이름에 Linux의 슬래시(/)와 같은 경로 구분 문자가 포함된 경우 사용할 단일 대체 문자입니다. ASCP는 경로 구분 문자가 포함된 탑재 파일을 생성할 수 없습니다. 그 대신 ASCP는 경로 구분 문자를 다른 문자로 대체합니다. 이 필드를 사용하지 않는 경우 대체 문자는 밑줄(_)이므로 예를 들어 My/Path/Secret은 My_Path_Secret으로 탑재됩니다.

문자 대체를 방지하려면 False 문자열을 입력합니다.

객체

탑재할 보안 암호의 YAML 선언을 포함하는 문자열입니다. YAML 다중 행 문자열 또는 파이프(|) 문자를 사용하는 것이 좋습니다.

objectName

보안 암호의 이름 또는 전체 ARN입니다. ARN을 사용하는 경우 objectType을 생략할 수 있습니다. 이 필드는 사용자가 objectAlias을(를) 지정하지 않는 한 Amazon EKS 포드에서 보안 암호의 파일 이름이 됩니다. ARN을 사용하는 경우 ARN의 리전이 필드 region와(과) 일치해야 합니다. failoverRegion을(를) 포함하는 경우 이 필드는 기본 objectName을(를) 나타냅니다.

objectType

objectName으로 Secrets Manager ARN을 사용하지 않는 경우에 필요합니다. secretsmanager 또는 ssmparameter입니다.

objectAlias

(선택 사항) Amazon EKS 포드에 있는 보안 암호의 파일 이름입니다. 이 필드를 지정하지 않은 경우 objectName이 파일 이름으로 나타납니다.

objectVersion

(선택 사항) 보안 암호의 버전 ID입니다. 보안 암호를 업데이트할 때마다 버전 ID를 업데이트해야 하므로 권장하지 않습니다. 기본적으로 가장 최신 버전이 사용됩니다. failoverRegion을(를) 포함하는 경우 이 필드는 기본 objectVersion을(를) 나타냅니다.

objectVersionLabel

(선택 사항) 버전의 별칭입니다. 기본값은 가장 최신 AWSCURRENT 버전입니다. 자세한 정보는 the section called “시크릿 버전”을 참조하세요. failoverRegion을(를) 포함하는 경우 이 필드는 기본 objectVersionLabel을(를) 나타냅니다.

jmesPath

(선택 사항) Amazon EKS에 탑재할 파일에 대한 보안 암호의 키 맵입니다. 이 필드를 사용하려면 보안 암호 값이 JSON 형식이어야 합니다. 이 필드를 사용하는 경우 `path` 및 `objectAlias` 하위 필드를 포함해야 합니다.

경로

보안 암호 값의 JSON에 있는 키/값 쌍의 키입니다. 필드에 하이픈이 포함된 경우 작은 따옴표를 사용하여 하이픈을 이스케이프 처리합니다. 예: `path: '"hyphenated-path"'`

objectAlias

Amazon EKS 포드에 탑재할 파일 이름입니다. 필드에 하이픈이 포함된 경우 작은 따옴표를 사용하여 하이픈을 이스케이프 처리합니다. 예: `objectAlias: '"hyphenated-alias"'`

failoverObject

(선택 사항) 이 필드를 지정하는 경우 ASCP는 기본 `objectName`에 지정된 보안 암호와 `failoverObject` `objectName` 하위 필드에 지정된 보안 암호를 모두 검색하려고 시도합니다. 둘 중 하나가 예를 들어 인증 문제에 대해 4xx 오류를 반환하는 경우 ASCP는 어느 암호도 탑재하지 않습니다. 기본 `objectName`에서 보안 암호가 성공적으로 검색되면 ASCP는 해당 암호 값을 탑재합니다. 기본 `objectName`에서는 보안 암호가 성공적으로 검색되지 않고 장애 조치 `objectName`에서 성공적으로 검색되는 경우 ASCP는 해당 암호 값을 탑재합니다. 이 필드를 포함하는 경우 필드 `objectAlias`도 포함해야 합니다. 이 필드 사용 방법의 예는 탑재할 장애 조치 암호 선택 섹션을 참조하세요.

일반적으로 장애 조치 암호가 복제본이 아닌 경우 이 필드를 사용합니다. 복제본을 지정하는 방법에 대한 예는 다중 리전 보안 암호에 대한 장애 조치 리전 정의 섹션을 참조하세요.

objectName

장애 조치 암호의 이름 또는 전체 ARN입니다. ARN을 사용하는 경우 ARN의 리전이 필드 `failoverRegion`와(과) 일치해야 합니다.

objectVersion

(선택 사항) 보안 암호의 버전 ID입니다. 기본 `objectVersion`와(과) 일치해야 합니다. 보안 암호를 업데이트할 때마다 버전 ID를 업데이트해야 하므로 권장하지 않습니다. 기본적으로 가장 최신 버전이 사용됩니다.

objectVersionLabel

(선택 사항) 버전의 별칭입니다. 기본값은 가장 최신 AWSCURRENT 버전입니다. 자세한 내용은 the section called “시크릿 버전”을(를) 참조하세요.

GitHub 작업에서 AWS Secrets Manager 비밀 사용

작업에서 비밀을 사용하려면 GitHub 작업을 사용하여 비밀을 검색하고 이를 GitHub 워크플로의 마스크된 환경 변수로 추가할 수 있습니다. GitHub AWS Secrets Manager GitHub 액션에 대한 자세한 내용은 GitHub 문서 내 GitHub 액션 이해를 참조하십시오.

GitHub 환경에 암호를 추가하면 GitHub 작업의 다른 모든 단계에서 해당 암호를 사용할 수 있습니다. 사용자 환경의 비밀이 오용되는 것을 방지하려면 GitHub 조치를 위한 보안 강화의 지침을 따르세요.

보안 암호 값의 전체 문자열을 환경 변수 값으로 설정하거나 문자열이 JSON인 경우 JSON을 구문 분석하여 각 JSON 키-값 쌍에 대한 개별 환경 변수를 설정할 수 있습니다. 보안 암호 값이 이진수인 경우 이 작업을 수행하면 보안 암호 값이 문자열로 변환됩니다.

보안 암호에서 생성된 환경 변수를 보려면 디버그 로깅을 켜세요. 자세한 내용은 문서에서 디버그 로깅 활성화를 참조하십시오. GitHub

시크릿으로 만든 환경 변수를 사용하려면 문서의 환경 변수를 참조하십시오. GitHub

사전 조건

이 작업을 사용하려면 먼저 `configure-aws-credentials` 단계를 사용하여 사용자 GitHub 환경에 AWS 자격 증명을 구성하고 설정해야 합니다. AWS 리전 GitHub OIDC 공급자를 사용하여 직접 역할을 수임할 GitHub 작업에 대한 AWS 자격 증명 작업 구성의 지침을 따르십시오. 그러면 수명이 짧은 보안 인증을 사용할 수 있으므로 Secrets Manager 외부에 추가 액세스 키를 저장하지 않아도 됩니다.

작업에서 맡는 IAM 역할은 다음과 같은 권한이 있어야 합니다.

- 검색하려는 보안 암호에 대한 `GetSecretValue`
- 모든 보안 암호에 대한 `ListSecrets`
- (선택 사항) `Decrypt` 암호가 a로 암호화된 KMS key 경우 고객 관리형 키

자세한 정보는 인증 및 액세스 제어을 참조하세요.

사용량

작업을 사용하려면 워크플로에 다음 구문을 사용하는 단계를 추가하세요.

```
- name: Step name
  uses: aws-actions/aws-secretsmanager-get-secrets@v2
  with:
    secret-ids: |
      secretId1
      ENV_VAR_NAME, secretId2
    name-transformation: (Optional) uppercase|lowercase|none
    parse-json-secrets: (Optional) true|false
```

파라미터

`secret-ids`

보안 암호 ARNS, 이름 및 이름 접두사.

환경 변수 이름을 설정하려면 보안 암호 ID 앞에 환경 변수 이름을 입력한 다음 쉼표를 입력합니다. 예를 들어 `ENV_VAR_1, secretId`는 보안 암호 `secretId`에서 ENV_VAR_1이라는 환경 변수를 생성합니다. 환경 변수 이름은 대문자, 숫자, 밑줄로 구성될 수 있습니다.

접두사를 사용하려면 3자 이상을 입력하고 그 뒤에 별표를 붙입니다. 예를 들어 `dev*`는 모든 보안 암호를 dev로 시작하는 이름과 일치시킵니다. 검색할 수 있는 일치하는 보안 암호의 최대 개수는 100개입니다. 변수 이름을 설정하고 접두사가 여러 보안 암호와 일치하는 경우 작업이 실패합니다.

`name-transformation`

기본적으로 이 단계에서는 보안 암호 이름에서 각 환경 변수 이름을 생성하며, 이름은 대문자, 숫자 및 밑줄만 포함하고 숫자로 시작되지 않도록 변환됩니다. 이름에 포함된 `lowercase` 문자의 경우 소문자를 사용하거나 대소문자를 바꾸지 않도록 단계를 구성할 수 있습니다. none 기본 값은 `uppercase`입니다.

`parse-json-secrets`

(선택 사항) 기본적으로 이 작업에서는 환경 변수 값을 보안 암호 값의 전체 JSON 문자열로 설정합니다. JSON의 각 키-값 쌍에 대한 환경 변수를 `parse-json-secrets` `true` 생성하도록 설정합니다.

JSON에서 “name” 및 “Name”과 같이 대소문자를 구분하는 키를 사용하는 경우 이 작업에서는 중복 이름 충돌이 발생합니다. 이 경우 `parse-json-secrets`를 `false`로 설정하고 JSON 보안 암호 값을 별도로 구문 분석합니다.

환경 변수 이름 지정

액션으로 생성되는 환경 변수는 해당 변수가 가져온 시크릿의 이름과 동일하게 지정됩니다. 환경 변수의 이름 지정 요구 사항은 시크릿보다 더 엄격하므로 액션은 이러한 요구 사항에 맞게 시크릿 이름을 변환합니다. 예를 들어 이 작업을 수행하면 소문자가 대문자로 변환됩니다. 예를 들어, 암호의 JSON을 파싱하면 환경 변수 이름에 암호 이름과 JSON 키 이름이 모두 포함됩니다. MYSECRET_KEYNAME 소문자를 변환하지 않도록 액션을 구성할 수 있습니다.

두 환경 변수의 이름이 같으면 작업이 실패합니다. 이 경우 환경 변수에 사용할 이름을 별칭으로 지정해야 합니다.

이름이 충돌할 수 있는 경우의 예:

- 이름이 MySecret ""“인 비밀과 이름이 “mysecret”인 시크릿은 모두 “MYSECRET”이라는 환경 변수가 됩니다.
- 이름이 “SECRET_Keyname”인 시크릿과 JSON 파싱된 시크릿과 이름이 “keyname”이고 JSON 파싱된 시크릿은 모두 “SECRET_KEYNAME”이라는 환경 변수가 된다.

다음 예제와 같이 별칭을 지정하여 환경 변수 이름을 설정할 수 있습니다. 그러면 이라는 변수가 생성됩니다. ENV_VAR_NAME

```
secret-ids: |
  ENV_VAR_NAME, secretId2
```

빈 별칭

- 빈 parse-json-secrets: true 별칭과 쉼표, 비밀 ID를 차례로 설정하고 입력하면 작업 시 파싱된 JSON 키와 동일하게 환경 변수의 이름을 지정합니다. 변수 이름에는 비밀 이름이 포함되지 않습니다.

  시크릿에 유효한 JSON이 포함되어 있지 않은 경우 액션은 하나의 환경 변수를 만들고 시크릿 이름과 같은 이름을 지정합니다.
- 빈 parse-json-secrets: false 별칭과 쉼표, 암호 ID를 설정하고 입력하면 작업은 별칭을 지정하지 않은 것처럼 환경 변수의 이름을 지정합니다.

다음 예에서는 빈 별칭을 보여 줍니다.

```
,secret2
```

예

Example 1 이름 및 ARN으로 보안 암호 가져오기

다음 예제에서는 이름 및 ARN으로 식별된 보안 암호에 대한 환경 변수를 생성합니다.

```
- name: Get secrets by name and by ARN
  uses: aws-actions/aws-secretsmanager-get-secrets@v2
  with:
    secret-ids: |
      exampleSecretName
      arn:aws:secretsmanager:us-east-2:123456789012:secret:test1-a1b2c3
      0/test/secret
      /prod/example/secret
      SECRET_ALIAS_1,test/secret
      SECRET_ALIAS_2,arn:aws:secretsmanager:us-east-2:123456789012:secret:test2-a1b2c3
      ,secret2
```

생성된 환경 변수:

```
EXAMPLESECRETNAME: secretValue1
TEST1: secretValue2
_0_TEST_SECRET: secretValue3
_PROD_EXAMPLE_SECRET: secretValue4
SECRET_ALIAS_1: secretValue5
SECRET_ALIAS_2: secretValue6
SECRET2: secretValue7
```

Example 2 접두사로 시작하는 모든 보안 암호 가져오기

다음 예제에서는 *beta*로 시작하는 이름을 가진 모든 보안 암호에 대한 환경 변수를 생성합니다.

```
- name: Get Secret Names by Prefix
  uses: 2
  with:
    secret-ids: |
      beta*    # Retrieves all secrets that start with 'beta'
```

생성된 환경 변수:

```
BETASECRETNAME: secretValue1
BETATEST: secretValue2
```

```
BETA_NEWSECRET: secretValue3
```

Example 3 보안 암호에서 JSON 구문 분석

다음 예제에서는 보안 암호에서 JSON을 구문 분석하여 환경 변수를 생성합니다.

```
- name: Get Secrets by Name and by ARN
  uses: aws-actions/aws-secretsmanager-get-secrets@v2
  with:
    secret-ids: |
      test/secret
      ,secret2
    parse-json-secrets: true
```

`test/secret` 보안 암호의 보안 암호 값은 다음과 같습니다.

```
{
  "api_user": "user",
  "api_key": "key",
  "config": {
    "active": "true"
  }
}
```

`secret2` 보안 암호의 보안 암호 값은 다음과 같습니다.

```
{
  "myusername": "alejandro_rosalez",
  "mypassword": "EXAMPLE_PASSWORD"
}
```

생성된 환경 변수:

```
TEST_SECRET_API_USER: "user"
TEST_SECRET_API_KEY: "key"
TEST_SECRET_CONFIG_ACTIVE: "true"
MYUSERNAME: "alejandro_rosalez"
MYPASSWORD: "EXAMPLE_PASSWORD"
```

Example 4 환경 변수 이름에는 소문자를 사용합니다.

다음 예제에서는 이름이 소문자로 된 환경 변수를 만듭니다.

```
- name: Get secrets
  uses: aws-actions/aws-secretsmanager-get-secrets@v2
  with:
    secret-ids: exampleSecretName
    name-transformation: lowercase
```

환경 변수 생성:

```
examplesecretname: secretValue
```

AWS IoT Greengrass에서 AWS Secrets Manager 보안 암호 사용

AWS IoT Greengrass는 클라우드 기능을 로컬 디바이스로 확장하는 소프트웨어입니다. 덕분에 디바이스는 정보 소스와 보다 밀접한 데이터를 수집 및 분석하고, 로컬 이벤트에 자율적으로 응답하며, 로컬 네트워크에서 서로 안전하게 통신할 수 있습니다.

AWS IoT Greengrass를 사용하면 하드 코딩된 암호, 토큰 또는 기타 암호를 사용하지 않고도 Greengrass 디바이스의 서비스와 애플리케이션에 인증할 수 있습니다. AWS Secrets Manager는 클라우드에서 보안 암호를 안전하게 저장하고 관리하는 데 사용할 수 있는 서비스입니다. AWS IoT Greengrass는 Secrets Manager를 Greengrass 코어 디바이스로 확장하며, 따라서 커넥터 및 Lambda 함수가 로컬 보안 암호를 사용해 서비스 및 애플리케이션과 상호 작용할 수 있습니다.

보안 암호를 Greengrass 그룹에 통합하려면 Secrets Manager 보안 암호를 참조하는 그룹 리소스를 생성합니다. 이 보안 암호 리소스는 연결된 ARN을 사용하여 클라우드 보안 암호를 참조합니다. 보안 암호 리소스를 생성, 관리 및 사용하는 방법에 대한 자세한 내용은 AWS IoT 개발자 안내서의 보안 암호 리소스 작업을 참조하세요.

보안 암호를 AWS IoT Greengrass 코어에 배포하려면 보안 암호를 AWS IoT Greengrass 코어에 배포를 참조하세요.

AWS Lambda 함수에 AWS Secrets Manager 시크릿 사용

AWS 파라미터 및 시크릿 Lambda 익스텐션을 사용하면 SDK를 사용하지 않고도 Lambda 함수에서 AWS Secrets Manager 시크릿을 검색하고 캐싱할 수 있습니다. 캐싱된 보안 암호를 검색하는 것이 Secret Manager에서 검색하는 것보다 빠릅니다. Secrets Manager API를 호출하는 데는 비용이 발생하므로 캐시를 사용하면 비용을 줄일 수 있습니다. 이 익스텐션은 Secrets Manager 보안 암호와 Parameter Store 파라미터를 모두 검색할 수 있습니다. Parameter Store에 대한 자세한 내용은 AWS

Systems Manager 사용 설명서의 Using Parameter Store parameters in Lambda functions(Lambda 익스텐션과 Parameter Store 통합)를 참조하세요.

Lambda 익스텐션은 Lambda 함수의 기능에 추가되는 동반 프로세스입니다. 자세한 내용은 Lambda 개발자 안내서의 Lambda 익스텐션을 참조하세요. 컨테이너 이미지의 확장 사용에 대한 자세한 내용은 컨테이너 이미지의 Lambda 계층 및 확장을 통한 작업을 참조하세요. Lambda는 Amazon Logs를 사용하여 함수와 함께 확장 프로그램에 대한 실행 정보를 기록합니다. CloudWatch 기본적으로 확장 프로그램은 최소한의 정보를 에 기록합니다. CloudWatch 세부 정보를 기록하려면 환경 변수 `PARAMETERS_SECRETS_EXTENSION_LOG_LEVEL`을 debug로 설정합니다.

파라미터 및 보안 암호에 대한 인 메모리 캐시를 제공하기 위해 확장은 로컬 HTTP 엔드포인트인 localhost 포트 2773을 Lambda 환경에 노출합니다. 환경 변수 `PARAMETERS_SECRETS_EXTENSION_HTTP_PORT`를 설정하여 포트를 구성할 수 있습니다.

Lambda는 함수에 필요한 동시성 레벨에 해당하는 별도의 인스턴스를 인스턴스화합니다. 각 인스턴스는 격리되며 구성 데이터의 자체 로컬 캐시를 유지합니다. Lambda 인스턴스 및 동시성에 대한 자세한 내용은 Lambda 개발자 안내서의 Lambda 예약된 동시성 관리를 참조하세요.

ARM 익스텐션을 추가하려면 Lambda 함수의 `arm64` 아키텍처를 사용해야 합니다. 자세한 내용은 Lambda 개발자 안내서의 Lambda 명령 세트 아키텍처를 참조하세요. 이 익스텐션은 아시아 태평양(뭄바이), 미국 동부(오하이오), 유럽(아일랜드), 유럽(프랑크푸르트), 유럽(취리히), 미국 동부(버지니아 북부), 유럽(런던), 유럽(스페인), 아시아 태평양(도쿄), 미국 서부(오레곤), 아시아 태평양(싱가포르), 아시아 태평양(하이데라바드), 아시아 태평양(시드니) 리전에서 ARM을 지원합니다.

확장 프로그램은 AWS 클라이언트를 사용합니다. AWS 클라이언트 구성에 대한 자세한 내용은 AWS SDK 및 도구 참조 안내서의 설정 참조를 참조하십시오. Lambda 함수가 VPC에서 실행되는 경우 확장 프로그램이 Secrets Manager를 호출할 수 있도록 VPC 엔드포인트를 생성해야 합니다. 자세한 정보는 VPC 엔드포인트을 참조하세요.

필요한 권한:

- Lambda 실행 역할에는 시크릿에 `secretsmanager:GetSecretValue` 대한 권한이 있어야 합니다.
- 암호가 대신 고객 관리형 키로 암호화된 경우 실행 역할에도 KMS 키에 대한 `kms:Decrypt` 권한이 필요합니다. AWS 관리형 키 `aws/secretsmanager`

AWS 파라미터 및 시크릿 Lambda 확장 프로그램을 사용하려면

1. AWS 파라미터 및 시크릿 Lambda 익스텐션이라는 AWS 레이어를 함수에 추가합니다. 지침은 Lambda 개발자 안내서의 함수에 계층 추가를 참조하십시오. 를 사용하여 레이어를 AWS CLI 추가하는 경우 확장의 ARN이 필요합니다. ARN 목록은AWS Systems Manager 사용 설명서의 AWS 매개변수 및 보안 암호 Lambda 확장 ARN을 참조하세요.
2. Lambda 실행 역할에게 보안 암호에 액세스할 수 있는 권한을 부여합니다.
   - 보안 암호에 대한 `secretsmanager:GetSecretValue` 권한. the section called “예: 개별 보안 암호 값을 검색할 수 있는 권한” 섹션을 참조하십시오.
   - (선택 사항) 암호가 가 아닌 고객 관리 키로 암호화된 경우 실행 역할에도 KMS 키에 대한 `kms:Decrypt` 권한이 필요합니다. AWS 관리형 키 `aws/secretsmanager`
   - ABAC(속성 기반 액세스 제어)를 Lambda 역할과 함께 사용하여 계정의 보안 암호에 더 세밀하게 액세스할 수 있습니다. 자세한 내용은 the section called “예: 태그를 사용하여 보안 암호에 대한 액세스 제어” 및 the section called “예: 보안 암호의 태그와 일치하는 태그를 사용하여 자격 증명에 대한 액세스 제한” 단원을 참조하세요.
3. Lambda 환경 변수를 사용하여 캐시를 구성합니다.
4. 익스텐션 캐시에서 보안 암호를 검색하려면 먼저 요청 헤더에 `X-AWS-Parameters-Secrets-Token`을 추가해야 합니다. 모든 실행 중인 함수에 대해 Lambda에서 제공하는 `AWS_SESSION_TOKEN`으로 토큰을 설정합니다. 이 헤더를 사용하면 호출자가 Lambda 환경 내에 있음을 나타냅니다.

   다음 Python 예제에서는 헤더를 추가하는 방법을 보여줍니다.

   ```
   import os
   headers = {"X-Aws-Parameters-Secrets-Token": os.environ.get('AWS_SESSION_TOKEN')}
   ```

5. Lambda 함수 내에서 보안 암호를 검색하려면 다음 HTTP GET 요청 중 하나를 사용합니다.
   - `secretId`에 대한 보안 암호를 검색하려면 보안 암호의 ARN 또는 이름을 사용합니다.

     ```
     GET: /secretsmanager/get?secretId=secretId
     ```

   - 이전 보안 암호 값이나 스테이징 레이블별 특정 버전을 검색하려면, `secretId`의 경우 보안 암호의 ARN 또는 이름을 사용하고, `versionStage`인 경우 스테이징 레이블을 사용하십시오.

     ```
     GET: /secretsmanager/get?secretId=secretId&versionStage=AWSPREVIOUS
     ```

- ID별로 특정 암호 버전을 검색하려면, `secretId`의 경우 보안 암호의 ARN 또는 이름을 사용하고, `versionId`인 경우 버전 ID를 사용하십시오.

```
GET: /secretsmanager/get?secretId=secretId&versionId=versionId
```

Example 보안 암호 검색(Python)

다음 Python 예제에서는 `json.loads`를 사용하여 보안 암호를 검색하고 결과를 구문 분석하는 방법을 보여줍니다.

```
secrets_extension_endpoint = "http://localhost:" + \
    secrets_extension_http_port + \
    "/secretsmanager/get?secretId=" + \
    <secret_name>

  r = requests.get(secrets_extension_endpoint, headers=headers)

  secret = json.loads(r.text)["SecretString"] # load the Secrets Manager response
 into a Python dictionary, access the secret
```

AWS 파라미터 및 시크릿 Lambda 익스텐션 환경 변수

다음과 같은 환경 변수를 사용하여 익스텐션을 구성할 수 있습니다.

환경 변수를 사용하는 방법에 대한 자세한 내용은 Lambda 개발자 안내서의 Lambda 환경 변수 사용을 참조하세요.

`PARAMETERS_SECRETS_EXTENSION_CACHE_ENABLED`

파라미터와 보안 암호를 캐시하려면 true로 설정합니다. 캐싱이 없는 경우 false로 설정합니다. 기본값은 true입니다.

`PARAMETERS_SECRETS_EXTENSION_CACHE_SIZE`

캐시할 최대 보안 암호 및 파라미터 수입니다. 0~1000 사이의 값이어야 합니다. 값이 0이면 적용된 캐싱이 없는 것입니다. 이 변수는 `SSM_PARAMETER_STORE_TTL` 및 `SECRETS_MANAGER_TTL`이 모두 0일 경우 무시됩니다. 기본값은 1000입니다.

`PARAMETERS_SECRETS_EXTENSION_HTTP_PORT`

로컬 HTTP 서버의 포트입니다. 기본값은 2773입니다.

PARAMETERS_SECRETS_EXTENSION_LOG_LEVEL

익스텐션 로깅 수준은debug, info, warn, error 또는 none을 제공합니다. 캐시 구성을 보려면 debug로 설정합니다. 기본값은 info입니다.

PARAMETERS_SECRETS_EXTENSION_MAX_CONNECTIONS

익스텐션에서 Parameter Store 또는 Secrets Manager에 요청하는 데 사용되는 HTTP 클라이언트의 최대 연결 수입니다. 이는 클라이언트별 구성입니다. 기본값은 3입니다.

SECRETS_MANAGER_TIMEOUT_MILLIS

Secrets Manager에 대한 요청 제한 시간(밀리초)입니다. 값이 0이면 적용된 시간 제한이 없는 것입니다. 기본값은 0.

SECRETS_MANAGER_TTL

캐시에 있는 보안 암호의 TTL(초)입니다. 값이 0이면 적용된 캐싱이 없는 것입니다. 최댓값은 300초입니다. PARAMETERS_SECRETS_CACHE_SIZE가 0인 경우 이 변수는 무시됩니다. 기본값은 300초입니다.

SSM_PARAMETER_STORE_TIMEOUT_MILLIS

Parameter Store에 대한 요청 제한 시간(밀리초)입니다. 값이 0이면 적용된 시간 제한이 없는 것입니다. 기본값은 0.

SSM_PARAMETER_STORE_TTL

캐시에 있는 파라미터의 TTL(초)입니다. 값이 0이면 적용된 캐싱이 없는 것입니다. 최댓값은 300초입니다. PARAMETERS_SECRETS_CACHE_SIZE가 0인 경우 이 변수는 무시됩니다. 기본값은 300초입니다.

파라미터 스토어에서 AWS Secrets Manager 보안 암호 사용

AWS Systems Manager Parameter Store는 구성 데이터 관리 및 암호 관리를 위한 안전한 계층적 스토리지를 제공합니다. 암호, 데이터베이스 문자열, 라이선스 코드와 같은 데이터를 파라미터 값으로 저장할 수 있습니다. 하지만 파라미터 스토어에서는 저장된 보안 암호에 대한 자동 교체 서비스를 제공하지 않습니다. 대신, 파라미터 스토어를 사용하여 Secrets Manage에 보안 암호를 저장한 다음 해당 보안 암호를 파라미터 스토어 파라미터로 참조할 수 있습니다.

Secrets Manage를 사용하여 파라미터 스토어를 구성하는 경우 secret-id 파라미터 스토어의 이름 문자열 앞에 슬래시(/)가 필요합니다.

자세한 내용은 AWS Systems Manager사용 설명서의 파라미터 스토어 파라미터에서 AWS Secrets Manager 보안 암호 참조를 참조하세요.

로테이션 AWS Secrets Manager 시크릿

교체는 주기적으로 보안 암호를 업데이트하는 프로세스입니다. 보안 암호를 교체하면 보안 암호 및 데이터베이스 또는 서비스 모두에서 자격 증명이 업데이트됩니다. Secrets Manager에서 보안 암호에 대한 자동 교체를 설정할 수 있습니다. 회전에는 두 가지 형태가 있습니다.

- 관리형 교체— 대부분의 관리 비밀의 경우 서비스가 사용자를 대신하여 순환을 구성하고 관리하는 관리 순환을 사용합니다. 관리형 로테이션은 Lambda 함수를 사용하지 않습니다.
- the section called “Lambda 함수에 의한 회전”— 다른 유형의 비밀의 경우 Secrets Manager 로테이션은 Lambda 함수를 사용하여 시크릿과 데이터베이스 또는 서비스를 업데이트합니다.

AWS Secrets Manager 시크릿 로테이션 관리

일부 서비스는 서비스에서 자동으로 교체를 구성하고 관리하는 관리형 교체를 제공합니다. 순환 관리에서는 데이터베이스의 암호와 자격 증명을 업데이트하는 AWS Lambda 함수를 사용하지 않습니다.

다음 서비스는 관리형 교체를 제공합니다:

- Amazon Aurora는 마스터 사용자 자격 증명에 대한 관리 순환 기능을 제공합니다. 자세한 내용은 Amazon Aurora 사용 설명서의 Amazon Aurora 및 AWS Secrets Manager을(를) 사용한 암호 관리를 참조하세요.
- Amazon ECS Service AWS Private Certificate Authority Connect는 TLS 인증서에 대한 관리형 교체 기능을 제공합니다. 자세한 내용은 Amazon Elastic 컨테이너 서비스 개발자 안내서의 서비스 연결을 통한 TLS를 참조하십시오.
- Amazon RDS는 마스터 사용자 자격 증명에 대한 관리 순환 기능을 제공합니다. 자세한 내용은 Amazon RDS 사용 설명서의 Amazon RDS 및 AWS Secrets Manager을(를) 사용한 암호 관리를 참조하세요.
- Amazon Redshift는 관리자 암호에 대한 관리 순환 기능을 제공합니다. 자세한 내용은 Amazon Redshift 관리 안내서의 AWS Secrets Manager을(를) 사용하는 Amazon Redshift 관리자 암호 관리를 참조하세요.

> **Tip**
>
> 다른 유형의 보안 암호에 대해서는 the section called “Lambda 함수에 의한 회전” 섹션을 참조하세요.

관리형 보안 암호의 교체는 일반적으로 1분 이내에 완료됩니다. 교체 중에 보안 암호를 검색하는 새 연결은 이전 버전의 보안 인증 정보를 가져올 수 있습니다. 애플리케이션에서는 마스터 사용자를 사용하는 대신에 애플리케이션에 필요한 최소한의 권한으로 생성한 데이터베이스 사용자를 사용하는 모범 사례를 따르는 것이 좋습니다. 애플리케이션 사용자의 경우 가용성을 극대화하기 위해 대체 사용자 교체 전략을 사용할 수 있습니다.

순환 관리 일정을 변경하려면

1. Secrets Manager 콘솔에서 관리형 보안 암호를 엽니다. 관리 서비스에서 링크를 따라가거나 Secrets Manager 콘솔에서 보안 암호를 검색할 수 있습니다.
2. Rotation schedule(교체 일정)에서 Schedule expression builder(예약 표현식 빌더) 또는 Schedule expression(예약 표현식)으로 일정(UTC 표준 시간대)을 입력합니다. Secrets Manager는 일정을 `rate( )` 또는 `cron( )` 표현식으로 저장합니다. 교체 기간은 Start time(시작 시간)을 지정하지 않는 한 자정에 자동으로 시작됩니다. 보안 암호를 4시간마다 교체할 수 있습니다. 자세한 정보는 로테이션 스케줄을 참조하세요.
3. (선택 사항) Window duration(지속 시간)에서 Secrets Manager가 보안 암호를 교체할 기간의 길이를 입력합니다. 예를 들어 **3h**는 3시간입니다. 기간은 그 다음 교체 기간까지 연장되지 않아야 합니다. 시간 단위의 교체 일정에서 지속 시간을 지정하지 않으면 한 시간 후에 자동으로 종료됩니다. 일 단위의 교체 일정인 경우 해당 날짜의 하루가 끝나면 자동으로 종료됩니다.
4. 저장을 선택합니다.

관리형 교체 일정 변경하기 (AWS CLI)

- `rotate-secret`을 호출합니다. 다음 예에서는 매월 1일과 15일 16:00 ~ 18:00 (UTC) 사이에 암호가 교체됩니다. 자세한 내용은 로테이션 스케줄을(를) 참조하세요.

```
aws secretsmanager rotate-secret \
    --secret-id MySecret \
    --rotation-rules "{\"ScheduleExpression\": \"cron(0 16 1,15 * ? *)\",
 \"Duration\": \"2h\"}"
```

Lambda 함수에 의한 회전

여러 유형의 비밀에 대해 Secrets Manager는 AWS Lambda 함수를 사용하여 비밀과 데이터베이스 또는 서비스를 업데이트합니다. Lambda 함수 사용 비용에 대한 자세한 내용은 요금 섹션을 참조하세요.

일부 관리형 시크릿의 경우 관리형 교체를 사용합니다. 관리형 교체을(를) 사용하려면 먼저 관리 서비스를 통해 보안 암호를 생성해야 합니다.

교체하는 동안 Secrets Manager는 교체 상태를 나타내는 이벤트를 로그합니다. 자세한 정보는 the section called “다음과 같이 로그하십시오. AWS CloudTrail ”을 참조하세요.

시크릿을 교체하기 위해 Secrets Manager는 사용자가 설정한 순환 일정에 따라 Lambda 함수를 호출합니다. 자동 교체가 설정된 상태에서 암호 값을 수동으로도 업데이트하는 경우, Secrets Manager는 다음 교체 날짜를 계산할 때 이를 유효한 교체로 간주합니다.

교체를 수행하는 동안 Secrets Manager에서는 파라미터가 다를 때마다 동일한 함수를 여러 번 호출합니다. Secrets Manager에서는 파라미터의 다음 JSON 요청 구조를 사용하여 함수를 호출합니다.

```
{
    "Step" : "request.type",
    "SecretId" : "string",
    "ClientRequestToken" : "string"
    }
```

교체 단계가 실패하면 Secrets Manager는 전체 교체 프로세스를 여러 번 다시 시도합니다.

주제

- 아마존 RDS, 아마존 Aurora, 아마존 Redshift 또는 아마존 DocumentDB 시크릿에 대한 자동 로테이션을 설정합니다.
- 비데이터베이스 AWS Secrets Manager 비밀번호에 대한 자동 교체 설정
- 를 사용하여 자동 회전을 설정합니다. AWS CLI
- Lambda 함수 회전 전략
- Lambda 회전 함수
- AWS Secrets Manager 회전 함수 템플릿
- 에 대한 Lambda 순환 함수 실행 역할 권한 AWS Secrets Manager
- Lambda 로테이션 함수를 위한 네트워크 액세스
- 회전 문제 해결 AWS Secrets Manager

아마존 RDS, 아마존 Aurora, 아마존 Redshift 또는 아마존 DocumentDB 시크릿에 대한 자동 로테이션을 설정합니다.

이 자습서에서는 데이터베이스 the section called “Lambda 함수에 의한 회전” 시크릿을 설정하는 방법을 설명합니다. 교체는 주기적으로 보안 암호를 업데이트하는 프로세스입니다. 보안 암호를 교체하면 보안 암호와 데이터베이스 모두에서 보안 인증 정보가 업데이트됩니다. Secrets Manager에서 데이터베이스 보안 암호에 대한 자동 교체를 설정할 수 있습니다.

콘솔을 사용하여 교체를 설정하려면 먼저 교체 전략을 선택해야 합니다. 그런 다음 교체에 대한 보안 암호를 구성하여 Lambda 교체 함수(아직 없는 경우)를 생성합니다. 콘솔은 Lambda 함수 실행 역할에 대한 권한도 설정합니다. 마지막 단계에서는 Lambda 교체 함수가 네트워크를 통해 Secrets Manager와 데이터베이스 모두에 액세스할 수 있는지 확인합니다.

> **Warning**
>
> 자동 순환을 활성화하려면 Lambda 순환 함수에 대한 IAM 실행 역할을 생성하고 권한 정책을 연결할 수 있는 권한이 있어야 합니다. `iam:CreateRole` 및 `iam:AttachRolePolicy` 권한이 모두 필요합니다. 이러한 권한을 부여하면 자격 증명이 자신에게 모든 권한을 부여할 수 있습니다.

단계:

- 1단계: 교체 전략 선택 및 (선택 사항) 슈퍼유저 보안 암호 생성
- 2단계: 교체 구성 및 교체 함수 생성
- 3단계: (선택 사항) 교체 함수에 대한 추가 권한 조건 설정
- 4단계: 교체 함수에 대한 네트워크 액세스 설정
- 다음 단계

1단계: 교체 전략 선택 및 (선택 사항) 슈퍼유저 보안 암호 생성

Secrets Manager에서 제공하는 전략에 대한 자세한 내용은 을 참조하십시오the section called “Lambda 함수 회전 전략”.

대체 사용자 전략을 선택하는 경우 데이터베이스 보안 암호 생성하고 데이터베이스 슈퍼유저 보안 인증을 저장해야 합니다. 교체에서는 첫 번째 사용자를 복제하므로 슈퍼유저 보안 인증을 가진 보안 암호

가 필요하며, 대부분의 사용자는 해당 권한이 없습니다. Amazon RDS 프록시는 대체 사용자 전략을 지원하지 않는다는 점에 유의하십시오.

2단계: 교체 구성 및 교체 함수 생성

Amazon RDS, Amazon DocumentDB, Amazon Redshift 보안 암호 교체를 켜려면

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호(Secrets) 페이지에서 보안 암호를 선택합니다.
3. 보안 암호 세부 정보(Secret details) 페이지의 교체 구성(Rotation configuration) 섹션에서 교체 편집(Edit rotation)을 선택합니다.
4. Edit rotation configuration(교체 구성 편집) 대화 상자에서 다음을 수행합니다.

    a. Automatic rotation(자동 교체)을 켭니다.

    b. Rotation schedule(교체 일정)에서 Schedule expression builder(예약 표현식 빌더) 또는 Schedule expression(예약 표현식)으로 일정(UTC 표준 시간대)을 입력합니다. Secrets Manager는 일정을 `rate( )` 또는 `cron( )` 표현식으로 저장합니다. 교체 기간은 Start time(시작 시간)을 지정하지 않는 한 자정에 자동으로 시작됩니다. 보안 암호를 4시간마다 교체할 수 있습니다. 자세한 정보는 로테이션 스케줄을 참조하세요.

    c. (선택 사항) Window duration(지속 시간)에서 Secrets Manager가 보안 암호를 교체할 기간의 길이를 입력합니다. 예를 들어 **3h**는 3시간입니다. 기간은 그 다음 교체 기간까지 연장되지 않아야 합니다. 시간 단위의 교체 일정에서 지속 시간을 지정하지 않으면 한 시간 후에 자동으로 종료됩니다. 일 단위의 교체 일정인 경우 해당 날짜의 하루가 끝나면 자동으로 종료됩니다.

    d. (선택 사항) 변경 사항을 저장할 때 보안 암호가 교체되게 하려면 보안 암호를 저장할 때 즉시 교체(Rotate immediately when the secret is stored)를 선택합니다. 확인란의 선택을 취소하는 경우에는 설정한 예약에 따라 첫 번째 교체가 시작됩니다.

    예를 들어 3단계와 4단계를 아직 완료하지 않아서 교체에 실패하는 경우 Secrets Manager는 교체 프로세스를 여러 번 재시도합니다.

    e. 교체 함수(Rotation function)에서 다음 중 하나를 수행합니다.

    - Create a new Lambda function(새 Lambda 함수 생성)을 선택하고 새 함수의 이름을 입력합니다. Secrets Manager에서 함수 이름의 시작 부분에 `SecretsManager`를 추가합니다. Secrets Manager는 적절한 템플릿을 기반으로 함수를 생성하고 Lambda 실행 역할에 필요한 권한을 설정합니다.
    - 다른 보안 암호에 사용한 교체 함수를 재사용하려면 Use an existing Lambda function(기존 Lambda 함수 사용)을 선택합니다. Recommended VPC configurations(권장 VPC 구

성)에 나열된 교체 함수에는 데이터베이스와 동일한 VPC 및 보안 그룹이 있으므로, 함수가 데이터베이스에 액세스하는 데 도움이 됩니다.

f. 교체 전략에서 단일 사용자 또는 대체 사용자 전략을 선택합니다. 자세한 정보는 the section called “1단계: 교체 전략 선택 및 (선택 사항) 슈퍼유저 보안 암호 생성”을 참조하세요.

5. 저장(Save)을 선택합니다.

3단계: (선택 사항) 교체 함수에 대한 추가 권한 조건 설정

교체 함수에 대한 리소스 정책에서는 컨텍스트 키 `aws:SourceAccount`를 포함하여 Lambda가 혼동된 대리자로 사용되지 않도록 하는 것이 좋습니다. 일부 AWS 서비스의 경우 혼란스러운 대리자 시나리오를 피하려면 글로벌 조건 `aws:SourceArn`키와 `aws:SourceAccount`글로벌 조건 키를 모두 사용할 AWS 것을 권장합니다. 그러나 교체 함수 정책에 `aws:SourceArn` 조건을 포함하는 경우 교체 함수는 해당 ARN에서 지정한 보안 암호를 교체하는 데만 사용할 수 있습니다. 여러 보안 암호에 대해 교체 기능을 사용할 수 있도록 컨텍스트 키 `aws:SourceAccount`만 포함하는 것이 좋습니다.

교체 함수 리소스 정책을 업데이트하려면

1. Secrets Manager 콘솔에서 보안 암호를 선택한 다음 세부 정보 페이지의 Rotation configuration(교체 구성)에서 Lambda 교체 함수를 선택합니다. Lambda 콘솔이 열립니다.
2. Lambda에 리소스 기반 정책 사용의 지침에 따라 `aws:sourceAccount` 조건을 추가합니다.

```
"Condition": {
    "StringEquals": {
        "AWS:SourceAccount": "123456789012"
    }
},
```

AWS 관리형 키 `aws/secretsmanager` 이외의 KMS 키를 사용하여 보안 암호를 암호화할 경우 Secrets Manager가 Lambda 실행 역할에 키 사용 권한을 부여합니다. SecretARN 암호화 컨텍스트를 사용하여 암호 해독 기능의 사용을 제한할 수 있으므로 교체 함수 역할은 교체를 담당하는 암호의 암호 해독에만 액세스할 수 있습니다.

교체 함수의 실행 역할 업데이트

1. Lambda 교체 함수에서 구성을 선택한 다음 실행 역할에서 역할 이름을 선택합니다.
2. `kms:EncryptionContext:SecretARN` 조건을 추가하려면 역할 권한 정책 수정의 지침을 따르세요.

```
"Condition": {
    "StringEquals": {
        "kms:EncryptionContext:SecretARN": "SecretARN"
    }
},
```

4단계: 교체 함수에 대한 네트워크 액세스 설정

자세한 정보는 the section called “Lambda 로테이션 함수를 위한 네트워크 액세스”을 참조하세요.

다음 단계

the section called “ 교체 문제 해결”을 참조하세요.

비데이터베이스 AWS Secrets Manager 비밀번호에 대한 자동 교체 설정

이 자습서에서는 데이터베이스가 아닌 암호를 the section called “Lambda 함수에 의한 회전” 설정하는 방법을 설명합니다. 교체는 주기적으로 보안 암호를 업데이트하는 프로세스입니다. 보안 암호를 교체하면 보안 암호와 보안 암호가 사용되는 데이터베이스 또는 서비스 모두에서 보안 인증이 업데이트됩니다.

데이터베이스 보안 암호는 데이터베이스 보안 암호 자동 교체(콘솔) 섹션을 참조하세요.

> **Warning**
>
> 자동 순환을 활성화하려면 Lambda 순환 함수에 대한 IAM 실행 역할을 생성하고 권한 정책을 연결할 수 있는 권한이 있어야 합니다. `iam:CreateRole` 및 `iam:AttachRolePolicy` 권한이 모두 필요합니다. 이러한 권한을 부여하면 자격 증명이 자신에게 모든 권한을 부여할 수 있습니다.

단계:

- 1단계: 일반 순환 함수 만들기
- 2단계: 교체 함수 코드 작성
- 3단계: 로테이션을 위한 암호 구성
- 4단계: 순환 함수가 Secrets Manager와 데이터베이스 또는 서비스에 액세스하도록 허용
- 5단계: Secrets Manager가 로테이션 함수를 호출하도록 허용

- 6단계: 로테이션 함수를 위한 네트워크 액세스를 설정합니다.
- 다음 단계

1단계: 일반 순환 함수 만들기

시작하려면 Lambda 회전 함수를 생성하십시오. 시크릿을 교체하는 코드는 포함되어 있지 않으므로 이후 단계에서 작성해 보겠습니다. 회전 함수의 작동 방식에 대한 자세한 내용은 을 참조하십시오the section called “Lambda 회전 함수”.

지원되는 지역에서는 를 AWS Serverless Application Repository 사용하여 템플릿에서 함수를 생성할 수 있습니다. 지원되는 지역 목록은 AWS Serverless Application Repository FAQ를 참조하십시오. 다른 지역에서는 함수를 처음부터 만들고 템플릿 코드를 함수에 복사합니다.

일반 회전 함수를 만들려면

1. 해당 지역에서 AWS Serverless Application Repository 지원되는지 확인하려면 일반 참조의 AWS Serverless Application RepositoryAWS 엔드포인트 및 할당량을 참조하십시오.
2. 다음 중 하나를 수행하십시오.
   - AWS Serverless Application Repository 해당 지역에서 지원되는 경우:
     a. Lambda 콘솔에서 애플리케이션을 선택한 다음 애플리케이션 생성을 선택합니다.
     b. 애플리케이션 생성 페이지에서 서버리스 애플리케이션 탭을 선택합니다.
     c. 공용 애플리케이션 아래의 검색 상자에 를 입력합니다**SecretsManagerRotationTemplate**.
     d. 사용자 지정 IAM 역할 또는 리소스 정책을 생성하는 앱 보기를 선택합니다.
     e. SecretsManagerRotationTemplate타일을 선택합니다.
     f. 검토, 구성 및 배포 페이지의 애플리케이션 설정 타일에서 필수 필드를 채웁니다.
        - 엔드포인트에 다음을 포함하여 해당 지역의 엔드포인트를 입력합니다**https://**. 엔드포인트 목록은 the section called “Secrets Manager 엔드포인트” 섹션을 참조하세요.
        - Lambda 함수를 VPC에 넣으려면 Ids 및 를 포함하십시오. vpcSecurityGroup vpcSubnetIds
     g. 배포를 선택합니다.
   - 해당 지역에서 AWS Serverless Application Repository 지원되지 않는 경우:
     a. Lambda 콘솔에서 함수를 선택한 다음 함수 생성을 선택합니다.

b. 함수 생성 페이지에서 다음을 수행합니다.

i. 새로 작성을 선택합니다.

ii. Function name(함수 이름)에 교체 함수의 이름을 입력합니다.

iii. Runtime(런타임)에서 Python 3.9를 선택합니다.

iv. 함수 생성을 선택합니다.

2단계: 교체 함수 코드 작성

이 단계에서는 암호와 암호가 사용되는 서비스 또는 데이터베이스를 업데이트하는 코드를 작성합니다. 회전 함수를 직접 작성하는 방법에 대한 팁을 포함하여 회전 함수의 기능에 대한 자세한 내용은 을 참조하십시오the section called “Lambda 회전 함수”. 를 교체 함수 템플릿 참조로 사용할 수도 있습니다.

3단계: 로테이션을 위한 암호 구성

이 단계에서는 암호의 순환 일정을 설정하고 암호에 순환 기능을 연결합니다.

교체를 구성하고 교체 함수를 생성하려면

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호(Secrets) 페이지에서 보안 암호를 선택합니다.
3. 보안 암호 세부 정보(Secret details) 페이지의 교체 구성(Rotation configuration) 섹션에서 교체 편집(Edit rotation)을 선택합니다. Edit rotation configuration(교체 구성 편집) 대화 상자에서 다음을 수행합니다.

   a. Automatic rotation(자동 교체)을 켭니다.

   b. Rotation schedule(교체 일정)에서 Schedule expression builder(예약 표현식 빌더) 또는 Schedule expression(예약 표현식)으로 일정(UTC 표준 시간대)을 입력합니다. Secrets Manager는 일정을 `rate()` 또는 `cron()` 표현식으로 저장합니다. 교체 기간은 Start time(시작 시간)을 지정하지 않는 한 자정에 자동으로 시작됩니다. 보안 암호를 4시간마다 교체할 수 있습니다. 자세한 정보는 로테이션 스케줄을 참조하세요.

   c. (선택 사항) Window duration(지속 시간)에서 Secrets Manager가 보안 암호를 교체할 기간의 길이를 입력합니다. 예를 들어 **3h**는 3시간입니다. 기간은 그 다음 교체 기간까지 연장되지 않아야 합니다. 시간 단위의 교체 일정에서 지속 시간을 지정하지 않으면 한 시간 후에 자동으로 종료됩니다. 일 단위의 교체 일정인 경우 해당 날짜의 하루가 끝나면 자동으로 종료됩니다.

d. (선택 사항) 변경 사항을 저장할 때 보안 암호가 교체되게 하려면 보안 암호를 저장할 때 즉시 교체(Rotate immediately when the secret is stored)를 선택합니다. 확인란의 선택을 취소하는 경우에는 설정한 예약에 따라 첫 번째 교체가 시작됩니다.

e. 회전 함수에서 1단계에서 생성한 Lambda 함수를 선택합니다.

f. 저장을 선택합니다.

4단계: 순환 함수가 Secrets Manager와 데이터베이스 또는 서비스에 액세스하도록 허용

Lambda 교체 함수에는 Secrets Manager의 보안 암호에 액세스할 수 있는 권한과, 데이터베이스 또는 서비스에 액세스할 수 있는 권한이 필요합니다. 이 단계에서는 Lambda 실행 역할에 이러한 권한을 부여합니다. AWS 관리형 키 `aws/secretsmanager` 이외의 KMS 키를 사용하여 보안 암호를 암호화할 경우 Lambda 실행 역할에 키 사용 권한을 부여해야 합니다. SecretARN 암호화 컨텍스트를 사용하여 암호 해독 기능의 사용을 제한할 수 있으므로 교체 함수 역할은 교체를 담당하는 암호의 암호 해독에만 액세스할 수 있습니다. 정책 예시는 교체 권한 섹션을 참조하세요.

지침은 AWS Lambda 개발자 안내서의 Lambda 실행 역할을 참조하세요.

5단계: Secrets Manager가 로테이션 함수를 호출하도록 허용

설정한 순환 일정에 따라 Secrets Manager가 순환 함수를 호출하도록 허용하려면 Lambda 함수의 리소스 정책에서 Secrets Manager 서비스 보안 주체에 `lambda:InvokeFunction` 권한을 부여해야 합니다.

교체 함수에 대한 리소스 정책에서는 컨텍스트 키 `aws:SourceAccount`를 포함하여 Lambda가 혼동된 대리자로 사용되지 않도록 하는 것이 좋습니다. 일부 AWS 서비스의 경우 혼동되는 부정 시나리오를 피하려면 글로벌 조건 `aws:SourceArn`키와 `aws:SourceAccount`글로벌 조건 키를 모두 사용할 AWS 것을 권장합니다. 그러나 교체 함수 정책에 `aws:SourceArn` 조건을 포함하는 경우 교체 함수는 해당 ARN에서 지정한 보안 암호를 교체하는 데만 사용할 수 있습니다. 여러 보안 암호에 대해 교체 기능을 사용할 수 있도록 컨텍스트 키 `aws:SourceAccount`만 포함하는 것이 좋습니다.

리소스 정책을 Lambda 함수에 연결하려면 Lambda에 대한 리소스 기반 정책 사용을 참조하세요.

다음 정책은 Secrets Manager가 Lambda 함수를 호출하도록 허용합니다.

```
{
    "Version": "2012-10-17",
    "Id": "default",
    "Statement": [
```

```
    {
        "Effect": "Allow",
        "Principal": {
            "Service": "secretsmanager.amazonaws.com"
            },
        "Action": "lambda:InvokeFunction",
        Condition": {
            "StringEquals": {
                "AWS:SourceAccount": "123456789012"
            }
        },
        "Resource": "LambdaRotationFunctionARN"
    }
    ]
}
```

6단계: 로테이션 함수를 위한 네트워크 액세스를 설정합니다.

이 단계에서는 순환 함수가 Secrets Manager와 암호의 대상 서비스 또는 데이터베이스 모두에 연결되도록 허용합니다. 암호를 교체할 수 있으려면 순환 함수가 두 기능에 모두 액세스할 수 있어야 합니다. the section called “Lambda 로테이션 함수를 위한 네트워크 액세스” 섹션을 참조하십시오.

다음 단계

3단계에서 순환을 구성할 때 암호를 교체하기 위한 일정을 설정합니다. 예약된 로테이션이 실패하면 Secrets Manager는 로테이션을 여러 번 시도합니다. 에 나와 있는 지침에 따라 즉시 로테이션을 시작할 수도 보안 암호 즉시 교체 있습니다.

회전이 실패할 경우 을 참조하십시오 교체 문제 해결.

를 사용하여 자동 회전을 설정합니다. AWS CLI

이 자습서에서는 를 the section called “Lambda 함수에 의한 회전” 사용하여 설정하는 방법을 설명합니다 AWS CLI. 보안 암호를 교체하면 보안 암호와 보안 암호가 사용되는 데이터베이스 또는 서비스 모두에서 보안 인증이 업데이트됩니다.

콘솔을 사용하여 회전을 설정할 수도 있습니다. 데이터베이스 보안 암호는 데이터베이스 보안 암호 자동 교체(콘솔) 섹션을 참조하세요. 다른 유형의 보안 암호에 대해서는 비데이터베이스 비밀번호의 자동 교체 (콘솔) 섹션을 참조하세요.

를 사용하여 순환을 설정하려면 데이터베이스 암호를 교체하는 경우 먼저 순환 전략을 선택해야 합니다. AWS CLI대체 사용자 전략을 선택하는 경우 별도의 보안 암호를 데이터베이스 슈퍼유저의 보안 인

증과 함께 저장해야 합니다. 그런 다음 교체 함수 코드를 작성합니다. Secrets Manager는 함수의 기반이 될 수 있는 템플릿을 제공합니다. 그러면 코드를 사용하여 Lambda 함수를 생성하고 Lambda 함수와 Lambda 실행 역할 모두에 대한 권한을 설정합니다. 다음 단계는 Lambda 함수가 네트워크를 통해 Secrets Manager와 데이터베이스 또는 서비스 모두에 액세스할 수 있는지 확인하는 것입니다. 마지막으로 교체에 대한 보안 암호를 구성합니다.

단계:

- 데이터베이스 암호에 대한 사전 요구 사항: 순환 전략 선택
- 1단계: 회전 함수 코드 작성
- 2단계: Lambda 함수 만들기
- 3단계: 네트워크 액세스 설정
- 4단계: 로테이션을 위한 암호 구성
- 다음 단계

데이터베이스 암호에 대한 사전 요구 사항: 순환 전략 선택

Secrets Manager에서 제공하는 전략에 대한 자세한 내용은 을 참조하십시오the section called “Lambda 함수 회전 전략”.

옵션 1: 단일 사용자 전략

단일 사용자 전략을 선택한 경우 1단계를 계속할 수 있습니다.

옵션 2: 대체 사용자 전략

대체 사용자 전략을 선택하는 경우 다음을 수행해야 합니다.

- 데이터베이스 암호를 만들고 여기에 데이터베이스 수퍼유저 자격 증명을 저장합니다. 수퍼유저 자격 증명이 포함된 암호가 필요합니다. 교대로 사용하면 첫 번째 사용자가 복제되고 대부분의 사용자에게는 해당 권한이 없기 때문입니다.
- 수퍼유저 암호의 ARN을 원본 암호에 추가합니다. 자세한 정보는 the section called “보안 암호의 JSON 구조”을 참조하세요.

Amazon RDS 프록시는 대체 사용자 전략을 지원하지 않는다는 점에 유의하십시오.

1단계: 회전 함수 코드 작성

보안 암호를 교체하려면 교체 함수가 필요합니다. 교체 함수는 Secrets Manager에서 보안 암호를 교체하기 위해 호출하는 Lambda 함수입니다. 자세한 정보는 the section called “Lambda 함수에 의한 회전”을 참조하세요. 이 단계에서는 암호와 암호가 사용되는 서비스 또는 데이터베이스를 업데이트하는 코드를 작성합니다.

Secrets Manager는 아마존 RDS, 아마존 오로라, 아마존 레드시프트 및 아마존 DocumentDB 데이터베이스 시크릿을 위한 템플릿을 제공합니다. 교체 함수 템플릿

회전 함수 코드를 작성하려면

1. 다음 중 하나를 수행하십시오.
   - 회전 함수 템플릿 목록을 확인하세요. 서비스 및 로테이션 전략과 일치하는 코드가 있으면 코드를 복사하세요.
   - 다른 유형의 비밀의 경우 로테이션 함수를 직접 작성합니다. 지침은 the section called “Lambda 회전 함수” 섹션을 참조하세요.
2. 파일을 필요한 종속 항목과 함께 ZIP 파일 *my-function.zip* 에 저장합니다.

2단계: Lambda 함수 만들기

이 단계에서는 1단계에서 생성한 ZIP 파일을 사용하여 Lambda 함수를 생성합니다. 또한 Lambda 실행 역할을 설정합니다. 이 역할은 함수가 호출될 때 Lambda가 맡는 역할입니다.

Lambda 교체 함수 및 실행 역할을 생성하려면

1. Lambda 실행 역할에 대한 신뢰 정책을 생성한 후 JSON 파일로 저장합니다. 예제와 자세한 내용은 을 참조하십시오. the section called “교체 권한” 정책은 다음을 충족해야 합니다.
   - 역할이 보안 암호에 대해 Secrets Manager 작업을 호출할 수 있도록 허용합니다.
   - 예를 들어, 역할이 암호의 대상 서비스를 호출하도록 허용하여 새 암호를 만들 수 있습니다.
2. Lambda 실행 역할을 생성하고 호출을 통해 이전 단계에서 생성한 신뢰 정책을 적용합니다. `iam create-role`

   ```
   aws iam create-role \
       --role-name rotation-lambda-role \
       --assume-role-policy-document file://trust-policy.json
   ```

3. `lambda create-function`을 호출하여 ZIP 파일에서 Lambda 함수를 생성합니다.

```
aws lambda create-function \
  --function-name my-rotation-function \
  --runtime python3.7 \
  --zip-file fileb://my-function.zip \
  --handler .handler \
  --role arn:aws:iam::123456789012:role/service-role/rotation-lambda-role
```

4. Secrets Manager가 lambda add-permission을 호출하여 호출할 수 있도록 Lambda 함수에 대한 리소스 정책을 설정합니다.

```
aws lambda add-permission \
  --function-name my-rotation-function \
  --action lambda:InvokeFunction \
  --statement-id SecretsManager \
  --principal secretsmanager.amazonaws.com \
  --source-account 123456789012
```

3단계: 네트워크 액세스 설정

자세한 정보는 the section called “Lambda 로테이션 함수를 위한 네트워크 액세스”을 참조하세요.

4단계: 로테이션을 위한 암호 구성

보안 암호에 대한 자동 교체를 켜려면 rotate-secret을 호출합니다. cron( ) 또는 rate( ) 일정 표현식을 사용하여 교체 일정을 설정하고 교체 기간을 설정할 수 있습니다. 자세한 정보는 the section called “로테이션 스케줄”을 참조하세요.

```
aws secretsmanager rotate-secret \
    --secret-id MySecret \
    --rotation-lambda-arn arn:aws:lambda:Region:123456789012:function:my-rotation-
function \
    --rotation-rules "{\"ScheduleExpression\": \"cron(0 16 1,15 * ? *)\", \"Duration\":
 \"2h\"}"
```

다음 단계

the section called “ 교체 문제 해결”을 참조하세요.

Lambda 함수 회전 전략

데이터베이스 암호의 the section called “Lambda 함수에 의한 회전” 경우 Secrets Manager는 두 가지 순환 전략을 제공합니다.

교체 전략: 단일 사용자

이 전략은 한 보안 암호로 한 사용자에 대한 보안 인증을 업데이트합니다. Amazon RDS Db2 인스턴스의 경우 사용자가 자신의 암호를 변경할 수 없으므로 별도의 보안 암호로 관리자 자격 증명을 제공해야 합니다. 이 전략은 가장 간단한 교체 전략이며 대부분의 사용 사례에 적합합니다. 특히 일회성 (임시) 또는 대화형 사용자의 보안 인증에 이 전략을 사용하는 것이 좋습니다.

보안 암호가 교체될 때 열린 데이터베이스 연결은 삭제되지 않습니다. 교체가 진행되는 동안 데이터베이스의 암호가 변경되는 시점부터 보안 암호가 업데이트되는 시점까지 짧은 기간이 소요됩니다. 이 시간동안 데이터베이스가 교체된 보안 인증을 사용하는 호출을 거부할 위험은 적습니다. 적절한 재시도 전략을 사용하여 이 위험을 완화할 수 있습니다. 교체 후에는 새 연결에서 새 보안 인증이 사용됩니다.

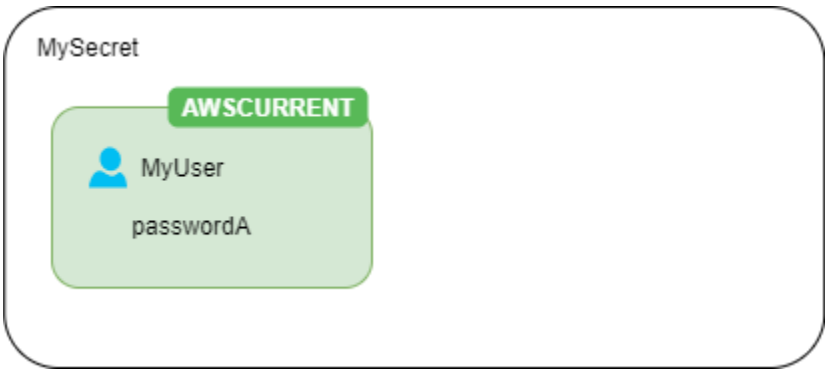
교체 전략: 대체 사용자

이 전략은 한 보안 암호로 두 사용자에 대한 보안 인증을 업데이트합니다. 첫 번째 사용자를 생성하고 첫 번째 교체 중에 교체 함수가 해당 사용자를 복제하여 두 번째 사용자를 생성합니다. 보안 암호가 교체될 때마다 교체 함수는 업데이트하는 사용자의 암호를 교체합니다. 대부분의 사용자는 본인을 복제할 수 있는 권한이 없으므로 다른 보안 암호에서 `superuser`에 대한 보안 인증을 제공해야 합니다. 데이터베이스의 복제된 사용자가 원래 사용자와 동일한 권한을 가지고 있지 않은 경우(일회성(임시) 또는 대화형 사용자라고도 함), 단일 사용자 교체 전략을 사용하는 것이 좋습니다.

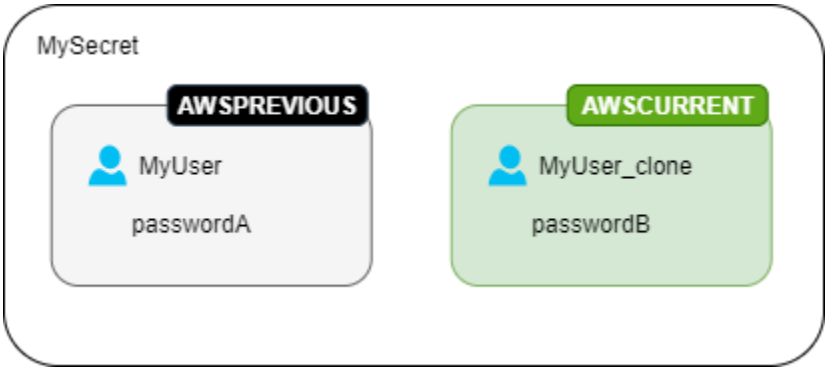
이 전략은 첫 번째 역할이 데이터베이스 테이블을 소유하고 두 번째 역할이 데이터베이스 테이블에 액세스할 수 있는 권한이 있는 권한 모델을 포함하는 데이터베이스에 적합합니다. 이 전력은 고가용성이 필요한 애플리케이션에도 적합합니다. 애플리케이션은 교체 중에 보안 암호가 검색되더라도 유효한 보안 인증 세트를 가져옵니다. 교체 후에는 `user` 및`user_clone` 보안 인증이 모두 유효합니다. 이러한 유형의 교체 중에 애플리케이션이 거부될 가능성은 단일 사용자 교체보다 훨씬 적습니다. 암호 변경 사항이 모든 서버로 전파되는 데 시간이 걸리는 서버 팜에서 데이터베이스가 호스트되는 경우 데이터베이스가 새 보안 인증을 사용하는 호출을 거부할 위험이 있습니다. 적절한 재시도 전략을 사용하여 이 위험을 완화할 수 있습니다.

Secrets Manager는 원래 사용자와 동일한 권한이 있는 복제된 사용자를 생성합니다. 클론이 생성된 후 원래 사용자의 권한을 변경하는 경우 복제된 사용자의 권한도 변경해야 합니다.

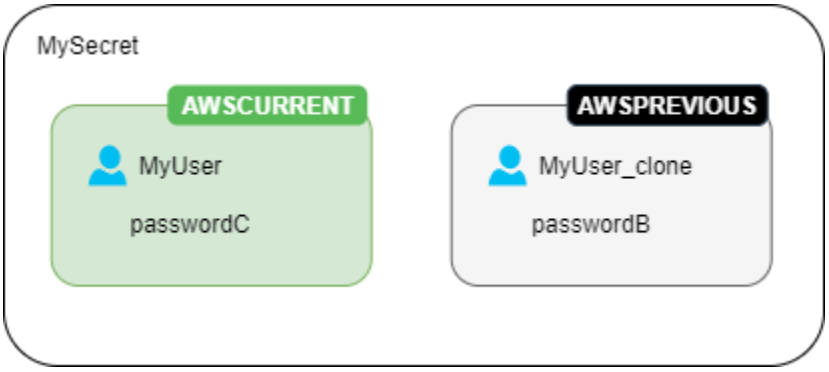
예를 들어 데이터베이스 사용자의 보안 인증으로 암호를 만드는 경우 암호에는 해당 보안 인증이 있는 버전 하나가 포함됩니다.



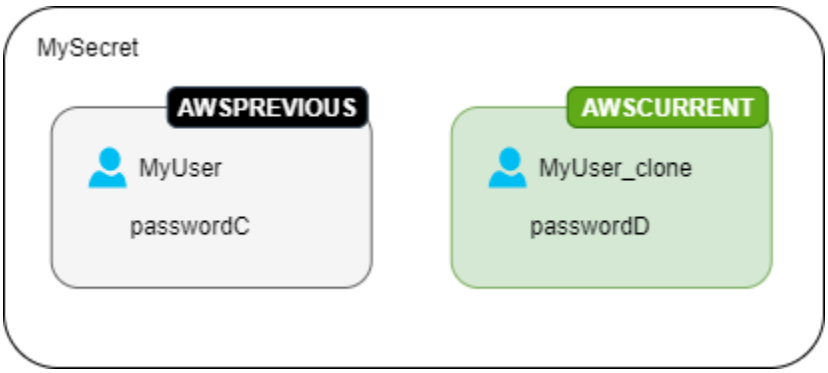
첫 번째 교체 - 교체 기능은 생성된 암호로 사용자의 클론을 생성하고 해당 보안 인증은 현재 암호 버전이 됩니다.



두 번째 교체 - 교체 기능은 원래 사용자의 암호를 업데이트합니다.



세 번째 교체 - 교체 기능은 복제된 사용자의 암호를 업데이트합니다.



Lambda 회전 함수

the section called “Lambda 함수에 의한 회전”에서는 Lambda 함수가 시크릿을 회전시키는 작업을 수행합니다. 교체하는 동안 Secrets Manager는 스테이징 레이블을 사용하여 보안 암호의 버전을 지정합니다.

Secrets Manager에서 암호 유형에 맞는 순환 함수 템플릿을 제공하지 않는 경우 순환 함수를 만들 수 있습니다. 로테이션 함수를 작성할 때는 각 단계의 지침을 따르세요.

회전 함수를 직접 작성하기 위한 팁

- 일반 회전 템플릿을 출발점으로 삼아 자신만의 회전 함수를 작성하세요.
- 함수를 직접 작성할 때와 마찬가지로 디버깅 또는 로깅 명령문을 포함하는 데 주의해야 합니다. 이러한 명령문으로 인해 함수의 정보가 CloudWatch Amazon에 기록될 수 있으므로 개발 중에 수집된 민감한 정보가 로그에 포함되지 않도록 해야 합니다.

  로그 문의 예는 the section called “교체 함수 템플릿” 소스 코드를 참조하세요.
- 보안을 위해 Secrets Manager는 Lambda 교체 함수에서 암호를 직접 교체하는 것만 허용합니다. 교체 함수는 두 번째 Lambda 함수를 호출하여 암호를 교체할 수 없습니다.
- 디버깅 제안 사항은 서버리스 애플리케이션 테스트 및 디버깅을 참조하세요.
- 예를 들어 외부 바이너리와 라이브러리를 사용하여 리소스에 연결하는 경우 이를 패치하고 보관하려면 관리해야 합니다. up-to-date
- 교체 함수를 필요한 종속성과 함께 *my-function.zip* ZIP 파일에 저장합니다.

로테이션 함수의 4단계

주제

- create_secret: 시크릿의 새 버전 만들기

- set_secret: 데이터베이스 또는 서비스의 자격 증명 변경
- test_secret: 새 시크릿 버전 테스트
- finish_secret: 로테이션 완료

create_secret: 시크릿의 새 버전 만들기

메서드는 create_secret 먼저 ClientRequestToken 전달된 get_secret_value암호로 호출하여 비밀이 존재하는지 확인합니다. 비밀이 없는 경우 토큰을 다음과 같이 하여 새 시크릿을 create_secret생성합니다. VersionId 그런 다음 를 사용하여 새 비밀 값을 생성합니다 get_random_password. 다음으로 스테이징 AWSPENDING 레이블과 함께 put_secret_value저장하도록 호출합니다. 새 보안 암호 값을 AWSPENDING에 저장하면 멱등성을 보장하는 데 도움이 됩니다. 어떤 이유로든 교체에 실패할 경우 후속 호출에서 해당 보안 암호 값을 참조할 수 있습니다. 멱등성 Lambda 함수를 만들려면 어떻게 해야 합니까? 섹션을 참조하세요.

회전 함수를 직접 작성하기 위한 팁

- 새 암호 값에는 데이터베이스 또는 서비스에 유효한 문자만 포함되도록 하십시오. ExcludeCharacters 파라미터를 사용하여 문자를 제외합니다.
- 함수를 테스트할 때는 를 사용하여 버전 단계를 확인하십시오: 호출 describe-secret및 살펴보기VersionIdsToStages. AWS CLI
- Amazon RDS MySQL의 경우 사용자 교대로 교체하여 Secrets Manager는 이름이 16자를 넘지 않는 클론 사용자를 생성합니다. 더 긴 사용자 이름을 허용하도록 교체 함수를 수정할 수 있습니다. MySQL 버전 5.7 이상에서는 최대 32자의 사용자 이름을 지원하지만 Secrets Manager는 사용자 이름 끝에 "_clone"(6자)을 추가하므로 사용자 이름을 최대 26자로 유지해야 합니다.

set_secret: 데이터베이스 또는 서비스의 자격 증명 변경

이 메서드는 데이터베이스 또는 서비스의 자격 증명을 암호 AWSPENDING 버전의 새 암호 값과 일치하도록 set_secret 변경합니다.

자체 회전 함수를 작성하기 위한 팁

- 명령문을 해석하는 서비스 (예: 데이터베이스) 에 명령문을 전달하는 경우 쿼리 매개 변수화를 사용하십시오. 자세한 내용은 OWASP 웹 사이트의 쿼리 매개 변수화 치트 시트를 참조하십시오.
- 교체 함수는 Secrets Manager 보안 암호와 대상 리소스 모두에서 고객 보안 인증을 액세스하고 수정할 수 있는 권한을 가진 대리자입니다. 잠재적으로 혼동된 대리자 공격을 방지하려면 공격자가 함수를 사용하여 다른 리소스에 액세스할 수 없도록 해야 합니다. 보안 인증을 업데이트하기 전에:

  - AWSCURRENT 버전 보안 암호의 보안 인증이 유효한지 확인합니다. AWSCURRENT 보안 인증이 유효하지 않은 경우 교체 시도를 중단합니다.
  - AWSCURRENT 및 AWSPENDING 보안 암호 값이 동일한 리소스에 대한 값인지 확인합니다. 사용자 이름과 암호의 경우 AWSCURRENT 및 AWSPENDING 사용자 이름이 동일한지 확인합니다.
  - 대상 서비스 리소스가 동일한지 확인합니다. 데이터베이스의 경우 AWSCURRENT 및 AWSPENDING 호스트 이름이 동일한지 확인합니다.
- 드문 경우이긴 하지만 데이터베이스의 기존 순환 함수를 사용자 지정해야 할 수도 있습니다. 예를 들어 대체 사용자 교체가 있는 경우 Secrets Manager는 첫 번째 사용자의 런타임 구성 매개변수를 복사하여 복제된 사용자를 생성합니다. 더 많은 속성을 포함하거나 복제된 사용자에게 부여되는 속성을 변경하려면 set_secret 함수의 코드를 업데이트해야 합니다.

test_secret: 새 시크릿 버전 테스트

다음으로 Lambda 교체 함수는 데이터베이스나 서비스에 액세스하는 데 보안 암호의 AWSPENDING 버전을 사용하여 해당 버전을 테스트합니다. 교체 함수 템플릿 기반 교체 함수는 읽기 액세스를 사용하여 새 보안 암호를 테스트합니다.

finish_secret: 로테이션 완료

마지막으로 Lambda 교체 함수는 AWSCURRENT 레이블을 이전 보안 암호 버전에서 이 버전으로 이동하며 동일한 API 호출에서 AWSPENDING 레이블도 제거합니다. Secrets Manager에서 AWSPREVIOUS 스테이징 레이블을 이전 버전으로 추가하여 보안 암호의 마지막으로 확인된 정상 버전을 유지할 수 있습니다.

이 메서드는 스테이징 레이블을 이전 비밀 AWSCURRENT 버전에서 새 비밀 버전으로 이동하는 데 finish_secret 사용합니다 update_secret_version_stage. Secrets Manager에서 AWSPREVIOUS 스테이징 레이블을 이전 버전에 자동으로 추가하여 보안 암호의 마지막으로 확인된 정상 버전을 유지할 수 있습니다.

자체 회전 함수 작성을 위한 팁

- 이 시점 AWSPENDING 이전에는 제거하지 말고 별도의 API 호출을 사용하여 제거하지 마십시오. 그러면 순환이 성공적으로 완료되지 않았음을 Secrets Manager에 알릴 수 있기 때문입니다. Secrets Manager에서 AWSPREVIOUS 스테이징 레이블을 이전 버전으로 추가하여 보안 암호의 마지막으로 확인된 정상 버전을 유지할 수 있습니다.

교체가 성공하면 AWSPENDING 스테이징 레이블은 AWSCURRENT 버전과 동일한 버전에 첨부되거나 어떤 버전에도 연결되지 않을 수 있습니다. AWSPENDING 스테이징 레이블이 있지만 AWSCURRENT와 동

일한 버전에 연결되지 않은 경우 이후의 교체 호출은 이전 교체 요청이 아직 진행 중인 것으로 간주하여 오류를 반환합니다. 교체에 실패하면 AWSPENDING 스테이징 레이블은 빈 암호 버전에 연결될 수 있습니다. 자세한 내용은 교체 문제 해결을(를) 참조하세요.

AWS Secrets Manager 회전 함수 템플릿

예를 들어the section called “Lambda 함수에 의한 회전”, Secrets Manager는 다양한 순환 함수 템플릿을 제공합니다. 템플릿을 사용하려면 다음 섹션을 참조하세요.

- 데이터베이스 보안 암호 자동 교체(콘솔)
- 비데이터베이스 비밀번호의 자동 교체 (콘솔)

템플릿은 Python 3.9를 지원합니다.

회전 함수를 직접 작성하려면 회전 함수 작성을 참조하십시오.

템플릿

- Amazon RDS 및 Amazon Aurora
  - Amazon RDS Db2 단일 사용자
  - Amazon RDS Db2 대체 사용자
  - Amazon RDS MariaDB 단일 사용자
  - Amazon RDS MariaDB 대체 사용자
  - Amazon RDS 및 Amazon Aurora MySQL 단일 사용자
  - Amazon RDS 및 Amazon Aurora MySQL 대체 사용자
  - Amazon RDS Oracle 단일 사용자
  - Amazon RDS Oracle 대체 사용자
  - Amazon RDS 및 Amazon Aurora PostgreSQL 단일 사용자
  - Amazon RDS 및 Amazon Aurora PostgreSQL 대체 사용자
  - Amazon RDS Microsoft SQLServer 단일 사용자
  - Amazon RDS Microsoft SQLServer 대체 사용자
- Amazon DocumentDB(MongoDB 호환)
  - Amazon DocumentDB 단일 사용자
  - Amazon DocumentDB 대체 사용자
- Amazon Redshift

  - Amazon Redshift 단일 사용자
  - Amazon Redshift 대체 사용자
- 아마존 ElastiCache
- Active Directory
  - 액티브 디렉터리 자격 증명
  - 액티브 디렉터리 키탭
- 다른 유형의 보안 암호

Amazon RDS 및 Amazon Aurora

Amazon RDS Db2 단일 사용자

- 템플릿 이름: SecretsManager RdsDB2 RotationSingleUser
- 교체 전략: 교체 전략: 단일 사용자.
- **`SecretString`** 구조: the section called “Amazon RDS Db2 보안 암호 구조”.
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation -람다스/트리/마스터/ RDSdB2 /lambda_function.py SecretsManager RotationSingleUser
- 종속성: python-ibmdb

Amazon RDS Db2 대체 사용자

- 템플릿 SecretsManager 이름: RdsDB2 RotationMultiUser
- 교체 전략: the section called “대체 사용자”.
- **`SecretString`** 구조: the section called “Amazon RDS Db2 보안 암호 구조”.
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation -람다스/트리/마스터/ RDSdB2 /lambda_function.py SecretsManager RotationMultiUser
- 종속성: python-ibmdb

Amazon RDS MariaDB 단일 사용자

- 템플릿 이름 SecretsManager: RDSMariaDB RotationSingleUser
- 교체 전략: 교체 전략: 단일 사용자.
- **`SecretString`** 구조: the section called “Amazon RDS MariaDB 보안 암호 구조”.

- 소스 코드: https://github.com/aws-samples/ -람다스/트리/마스터/ RDSMariaDB /lambda_function.py aws-secrets-manager-rotation SecretsManager RotationSingleUser
- 종속성PyMy: SQL 1.0.2. 인증에 sha256 비밀번호를 사용하는 경우 PyMy SQL [rsa] 를 사용하십시오. Lambda 런타임에서 컴파일된 코드가 포함된 패키지를 사용하는 방법에 대한 자세한 내용은 컴파일된 바이너리가 있는 Python 패키지를 배포 패키지에 추가하고 패키지를 Lambda와 호환되도록 하려면 어떻게 해야 합니까? 를 참조하십시오. AWS 지식 센터에서.

Amazon RDS MariaDB 대체 사용자

- 템플릿 이름: SecretsManager RDSMariaDB RotationMultiUser
- 교체 전략: the section called “대체 사용자”.
- **`SecretString`** 구조: the section called “Amazon RDS MariaDB 보안 암호 구조”.
- 소스 코드: https://github.com/aws-samples/ -람다스/트리/마스터/ RDSMariaDB /lambda_function.py aws-secrets-manager-rotation SecretsManager RotationMultiUser
- 종속성PyMy: SQL 1.0.2. 인증에 sha256 비밀번호를 사용하는 경우 PyMy SQL [rsa] 를 사용하십시오. Lambda 런타임에서 컴파일된 코드가 포함된 패키지를 사용하는 방법에 대한 자세한 내용은 컴파일된 바이너리가 있는 Python 패키지를 배포 패키지에 추가하고 패키지를 Lambda와 호환되도록 하려면 어떻게 해야 합니까? 를 참조하십시오. AWS 지식 센터에서.

Amazon RDS 및 Amazon Aurora MySQL 단일 사용자

- 템플릿 이름: SecretsManager RDSmySQL RotationSingleUser
- 교체 전략: the section called “단일 사용자”.
- 예상 **`SecretString`** 구조: the section called “Amazon RDS 및 Amazon Aurora MySQL 보안 암호 구조”.
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation -람다스/트리/마스터/ RDSmySQL /lambda_function.py SecretsManager RotationSingleUser
- 종속성PyMy: SQL 1.0.2. 인증에 sha256 비밀번호를 사용하는 경우 PyMy SQL [rsa] 를 사용하십시오. Lambda 런타임에서 컴파일된 코드가 포함된 패키지를 사용하는 방법에 대한 자세한 내용은 컴파일된 바이너리가 있는 Python 패키지를 배포 패키지에 추가하고 패키지를 Lambda와 호환되도록 하려면 어떻게 해야 합니까? 를 참조하십시오. AWS 지식 센터에서.

Amazon RDS 및 Amazon Aurora MySQL 대체 사용자

- 템플릿 이름: SecretsManager RDSmySQL RotationMultiUser

- 교체 전략: the section called “대체 사용자”.
- 예상 **SecretString** 구조: the section called “Amazon RDS 및 Amazon Aurora MySQL 보안 암호 구조”.
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation -람다스/트리/마스터/ RDSmySQL /lambda_function.py SecretsManager RotationMultiUser
- 종속성PyMy: SQL 1.0.2. 인증에 sha256 비밀번호를 사용하는 경우 PyMy SQL [rsa] 를 사용하십시오. Lambda 런타임에서 컴파일된 코드가 포함된 패키지를 사용하는 방법에 대한 자세한 내용은 컴파일된 바이너리가 있는 Python 패키지를 배포 패키지에 추가하고 패키지를 Lambda와 호환되도록 하려면 어떻게 해야 합니까? 를 참조하십시오. AWS 지식 센터에서.

Amazon RDS Oracle 단일 사용자

- 템플릿 이름: SecretsManager RDS OracleRotationSingleUser
- 교체 전략: the section called “단일 사용자”.
- 예상 **SecretString** 구조: the section called “Amazon RDS Oracle 보안 암호 구조”.
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManager -람다스/트리/마스터/ RDS /lambda_function.py OracleRotationSingleUser
- 종속성: 파이톤-오라클DB 2.0.1

Amazon RDS Oracle 대체 사용자

- 템플릿 이름: RDS SecretsManager OracleRotationMultiUser
- 교체 전략: the section called “대체 사용자”.
- 예상 **SecretString** 구조: the section called “Amazon RDS Oracle 보안 암호 구조”.
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManager -람다스/트리/마스터/ RDS /lambda_function.py OracleRotationMultiUser
- 종속성: 파이톤-오라클DB 2.0.1

Amazon RDS 및 Amazon Aurora PostgreSQL 단일 사용자

- 템플릿 이름: RDSPostgreSQL SecretsManager RotationSingleUser
- 교체 전략: 교체 전략: 단일 사용자.
- 예상 **SecretString** 구조: the section called “Amazon RDS 및 Amazon Aurora PostgreSQL 보안 암호 구조”.

- 소스 코드: https://github.com/aws-samples/ -람다스/트리/마스터/ RDSPostgreSQL / lambda_function.py aws-secrets-manager-rotation SecretsManager RotationSingleUser
- PyGre종속성: SQL 5.0.7

Amazon RDS 및 Amazon Aurora PostgreSQL 대체 사용자

- 템플릿 이름: RDSpostgreSQL SecretsManager RotationMultiUser
- 교체 전략: the section called “대체 사용자”.
- 예상 **`SecretString`** 구조: the section called “Amazon RDS 및 Amazon Aurora PostgreSQL 보안 암호 구조”.
- 소스 코드: https://github.com/aws-samples/ -람다스/트리/마스터/ RDSPostgreSQL / lambda_function.py aws-secrets-manager-rotation SecretsManager RotationMultiUser
- PyGre종속성: SQL 5.0.7

Amazon RDS Microsoft SQLServer 단일 사용자

- 템플릿 이름: RDSSQL SecretsManager ServerRotationSingleUser
- 교체 전략: the section called “단일 사용자”.
- 예상 **`SecretString`** 구조: the section called “Amazon RDS Microsoft SQLServer 보안 암호 구조”.
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation -람다스/트리/마스터/ RDSSQL /lambda_function.py SecretsManager ServerRotationSingleUser
- 종속성: Pymssql 2.2.2

Amazon RDS Microsoft SQLServer 대체 사용자

- 템플릿 이름 SecretsManager: RDSSQL ServerRotationMultiUser
- 교체 전략: the section called “대체 사용자”.
- 예상 **`SecretString`** 구조: the section called “Amazon RDS Microsoft SQLServer 보안 암호 구조”.
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation -람다스/트리/마스터/ RDSSQL /lambda_function.py SecretsManager ServerRotationMultiUser
- 종속성: Pymssql 2.2.2

Amazon DocumentDB(MongoDB 호환)

Amazon DocumentDB 단일 사용자

- 템플릿 이름: SecretsManagerMongo DB RotationSingleUser
- 교체 전략: the section called “단일 사용자”.
- 예상 **`SecretString`** 구조: the section called “Amazon DocumentDB 보안 암호 구조”.
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerMongo -람다스/트리/마스터/ DB /lambda_function.py RotationSingleUser
- 종속성: Pymongo 3.2

Amazon DocumentDB 대체 사용자

- 템플릿 이름: SecretsManagerMongo DB RotationMultiUser
- 교체 전략: the section called “대체 사용자”.
- 예상 **`SecretString`** 구조: the section called “Amazon DocumentDB 보안 암호 구조”.
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerMongo -람다스/트리/마스터/ DB /lambda_function.py RotationMultiUser
- 종속성: Pymongo 3.2

Amazon Redshift

Amazon Redshift 단일 사용자

- 템플릿 이름: SecretsManagerRedshiftRotationSingleUser
- 교체 전략: the section called “단일 사용자”.
- 예상 **`SecretString`** 구조: the section called “Amazon Redshift 보안 구조” 또는the section called “Amazon Redshift 서버리스 비밀 구조”.
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerRedshiftRotationSingleUser -람다스/트리/마스터/ /lambda_function.py
- PyGre종속성: SQL 5.0.7

Amazon Redshift 대체 사용자

- 템플릿 이름: SecretsManagerRedshiftRotationMultiUser

- 교체 전략: the section called “대체 사용자”.
- 예상 **SecretString** 구조: the section called “Amazon Redshift 보안 구조” 또는the section called “Amazon Redshift 서버리스 비밀 구조”.
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerRedshiftRotationMultiUser -람다스/트리/마스터/ /lambda_function.py
- PyGre종속성: SQL 5.0.7

아마존 ElastiCache

이 템플릿을 사용하려면 Amazon 사용 설명서의 ElastiCache 사용자 암호 자동 교체를 참조하십시오.

- 템플릿 이름: SecretsManagerElasticacheUserRotation
- 예상 **SecretString** 구조: the section called “아마존 ElastiCache 비밀 구조”.
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerElasticacheUserRotation -람다스/트리/마스터/ /lambda_function.py

Active Directory

액티브 디렉터리 자격 증명

- 템플릿 이름: SecretsManagerActiveDirectoryRotationSingleUser
- 예상 **SecretString** 구조: the section called “액티브 디렉터리 자격 증명 비밀 구조”.
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerActiveDirectoryRotationSingleUser -람다스/트리/마스터/ /lambda_function.py

액티브 디렉터리 키탭

- 템플릿 이름: SecretsManagerActiveDirectoryAndKeytabRotationSingleUser
- 예상 **SecretString** 구조: the section called “액티브 디렉터리 비밀 구조”.
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerActiveDirectoryAndKeytabRotationSingleUser -람다스/트리/마스터/ / lambda_function.py
- 종속성: msktutil

다른 유형의 보안 암호

Secrets Manager은 이 템플릿을 모든 유형의 보안 암호에 대한 교체 함수를 생성할 수 있는 시작점으로 제공합니다.

- 템플릿 이름: SecretsManagerRotationTemplate
- 소스 코드: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerRotationTemplate -람다스/트리/마스터/ /lambda_function.py

에 대한 Lambda 순환 함수 실행 역할 권한 AWS Secrets Manager

예를 들어the section called “Lambda 함수에 의한 회전”, Secrets Manager가 Lambda 함수를 사용하여 시크릿을 교체하면 Lambda는 IAM 실행 역할을 맡아 Lambda 함수 코드에 해당 자격 증명을 제공합니다. 자동 교체 설정 방법에 대한 지침은 다음을 참조하십시오.

- 데이터베이스 보안 암호 자동 교체(콘솔)
- 비데이터베이스 비밀번호의 자동 교체 (콘솔)
- 자동 교체(AWS CLI)

다음 예제에서는 Lambda 교체 함수 실행 역할에 대한 인라인 정책을 보여 줍니다. 실행 역할을 생성하고 권한 정책을 연결하려면 AWS Lambda 실행 역할을 참조하세요.

예:

- Lambda 교체 함수 실행 역할에 대한 정책
- 고객 관리형 키에 대한 정책 설명
- 대체 사용자 전략에 대한 정책 설명

Lambda 교체 함수 실행 역할에 대한 정책

다음 예제 정책에서는 교체 함수가 다음 작업을 수행할 수 있도록 허용합니다.

- *SecretARN*에 대한 Secrets Manager 작업을 실행합니다.
- 새 암호를 생성합니다.
- 데이터베이스 또는 서비스가 VPC에서 실행되는 경우 필수 구성을 설정하십시오. VPC에서 리소스에 액세스하도록 Lambda 함수 구성을 참조하세요.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:UpdateSecretVersionStage"
            ],
            "Resource": "SecretARN"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword"
            ],
            "Resource": "*"
        },
        {
            "Action": [
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DetachNetworkInterface"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

고객 관리형 키에 대한 정책 설명

AWS 관리형 키 `aws/secretsmanager` 이외의 KMS 키를 사용하여 보안 암호를 암호화할 경우 Lambda 실행 역할에 키 사용 권한을 부여해야 합니다. SecretARN 암호화 컨텍스트를 사용하여 암호 해독 기능의 사용을 제한할 수 있으므로 교체 함수 역할은 교체를 담당하는 암호의 암호 해독에만 액세스할 수 있습니다. 다음 예시에서는 함수가 KMS 키를 이용하여 비밀을 해독할 수 있도록 실행 역할 정책에 추가할 선언문을 보여 줍니다.

```
        {
```

```
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:DescribeKey",
                "kms:GenerateDataKey"
            ],
            "Resource": "KMSKeyARN"
            "Condition": {
                "StringEquals": {
                    "kms:EncryptionContext:SecretARN": "SecretARN"
                }
            }
        }
```

고객 관리형 키로 암호화된 여러 암호에 교체 기능을 사용하려면 다음 예와 같은 명령문을 추가하여 실행 역할이 암호를 해독하도록 허용하세요.

```
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:DescribeKey",
                "kms:GenerateDataKey"
            ],
            "Resource": "KMSKeyARN"
            "Condition": {
                "StringEquals": {
                    "kms:EncryptionContext:SecretARN": [
                        "arn1",
                        "arn2"
                    ]
                }
            }
        }
```

대체 사용자 전략에 대한 정책 설명

대체 사용자 교체 전략에 대한 자세한 내용은 the section called “Lambda 함수 회전 전략” 단원을 참조하세요.

Amazon RDS 보안 인증 정보가 포함된 암호의 경우, 대체 사용자 전략을 사용하고 Amazon RDS에서 슈퍼 사용자 암호를 관리한다면 교체 함수가 Amazon RDS에서 읽기 전용 API를 호출하도록 허용해야

데이터베이스의 연결 정보를 가져올 수도 있습니다. ReadOnlyAccessAmazonRDS의 AWS 관리형 정책을 첨부하는 것이 좋습니다.

다음 예제 정책에서는 함수가 다음 작업을 수행할 수 있도록 허용합니다.

- *SecretARN*에 대한 Secrets Manager 작업을 실행합니다.
- 슈퍼유저 보안 암호에서 보안 인증을 검색합니다. 슈퍼유저 보안 암호의 보안 인증은 Secrets Manager가 교체된 보안 암호의 보안 인증을 업데이트하는 데 사용됩니다.
- 새 암호를 생성합니다.
- 데이터베이스 또는 서비스가 VPC에서 실행되는 경우 필수 구성을 설정하십시오. 자세한 내용은 VPC의 리소스에 액세스하도록 Lambda 함수 구성(Configuring a Lambda function to access resources in a VPC)을 참조하세요.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:UpdateSecretVersionStage"
            ],
            "Resource": "SecretARN"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": "SuperuserSecretARN"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword"
            ],
            "Resource": "*"
        },
        {
```

```
            "Action": [
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DetachNetworkInterface"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Lambda 로테이션 함수를 위한 네트워크 액세스

왜냐하면 the section called “Lambda 함수에 의한 회전” Secrets Manager가 Lambda 함수를 사용하여 시크릿을 교체하는 경우 Lambda 로테이션 함수가 시크릿에 액세스할 수 있어야 하기 때문입니다. 보안 암호에 보안 인증이 포함된 경우 Lambda 함수가 해당 보안 인증의 소스(예: 데이터베이스 또는 서비스)에도 액세스할 수 있어야 합니다.

보안 암호에 액세스하려면

- Lambda 교체 함수는 Secrets Manager 엔드포인트에 액세스할 수 있어야 합니다. Lambda 함수가 인터넷에 액세스할 수 있는 경우 퍼블릭 엔드포인트를 사용할 수 있습니다. 엔드포인트를 찾으려면 the section called “Secrets Manager 엔드포인트” 섹션을 참조하세요.

  Lambda 함수가 인터넷에 액세스할 수 없는 VPC에서 실행되는 경우 VPC 내에서 Secrets Manager 서비스 프라이빗 엔드포인트를 구성하는 것이 좋습니다. 그러면 VPC가 퍼블릭 리전 엔드포인트로 전달된 요청을 가로채서 프라이빗 엔드포인트로 리디렉션할 수 있습니다. 자세한 정보는 VPC 엔드포인트을 참조하세요.

  또는 VPC에 NAT 게이트웨이 또는 인터넷 게이트웨이를 추가하여 Lambda 함수를 통해 퍼블릭 Secrets Manager 엔드포인트에 액세스할 수 있도록 합니다. 그러면 VPC에서 트래픽이 퍼블릭 엔드포인트에 도달할 수 있습니다. 이렇게 하면 게이트웨이에 대한 IP 주소가 퍼블릭 인터넷에서 공격을 받을 수 있으므로 VPC가 더 많은 위험에 노출됩니다.

(선택 사항) 데이터베이스 또는 서비스에 액세스하려면

- 보안 암호(예: API 키)의 경우 보안 암호와 함께 업데이트해야 하는 소스 데이터베이스 또는 서비스가 없습니다.

데이터베이스 또는 인스턴스가 VPC의 Amazon EC2 인스턴스에서 실행 중인 경우, 동일한 VPC에서 Lambda 함수가 실행되도록 구성하는 것이 좋습니다. 그러면 교체 함수가 서비스와 직접 통신할 수 있습니다. 자세한 내용은 VPC 액세스 구성을 참조하세요.

Lambda 함수가 데이터베이스 또는 서비스에 액세스하도록 허용하려면 Lambda 교체 함수에 연결된 보안 그룹이 데이터베이스 또는 서비스에 아웃바운드 연결을 허용하는지 확인해야 합니다. 데이터베이스 또는 서비스에 연결된 보안 그룹이 Lambda 교체 함수에 인바운드 연결을 허용하는지도 확인해야 합니다.

회전 문제 해결 AWS Secrets Manager

많은 서비스에서 Secrets Manager는 Lambda 함수를 사용하여 보안 암호를 교체합니다. 자세한 정보는 the section called “Lambda 함수에 의한 회전”을 참조하세요. Lambda 교체 함수는 Secrets Manager뿐만 아니라 보안 암호가 사용되는 데이터베이스 또는 서비스와도 상호 작용합니다. 로테이션이 예상대로 작동하지 않으면 먼저 CloudWatch 로그를 확인해야 합니다.

> **Note**
>
> 자동 교체 관리를 비롯한 일부 서비스는 보안 암호를 대신 관리할 수 있습니다. 자세한 정보는 the section called “관리형 교체”을 참조하세요.

Lambda 함수의 CloudWatch 로그를 보려면

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호를 선택한 다음 세부 정보 페이지의 교체 구성(Rotation configuration)에서 Lambda 교체 함수를 선택합니다. Lambda 콘솔이 열립니다.
3. 모니터 탭에서 [Logs] 를 선택한 다음 [로그인 보기] 를 선택합니다. CloudWatch

   CloudWatch 콘솔이 열리고 함수에 대한 로그가 표시됩니다.

로그를 해석하려면

- “환경 변수에서 보안 인증을 찾았습니다.” 이후 활동 없음
- “createSecret” 이후 활동 없음
- 오류: “KMS에 대한 액세스가 허용되지 않습니다.”
- 오류: “보안 암호 JSON에 키가 없습니다.”

- 오류: “setSecret: 데이터베이스에 로그인할 수 없음”
- 오류: “모듈 'lambda_function'을 가져올 수 없음”
- 기존 교체 함수를 Python 3.7에서 3.9로 업그레이드

“환경 변수에서 보안 인증을 찾았습니다.” 이후 활동 없음

“환경 변수에서 보안 인증을 찾았습니다.” 이후에 활동이 없고 작업이 오래 실행되는 경우(예: 기본 Lambda 제한 시간 30000ms), Secrets Manager 엔드포인트에 연결하는 동안 Lambda 함수가 시간 초과될 수 있습니다.

Lambda 교체 함수는 Secrets Manager 엔드포인트에 액세스할 수 있어야 합니다. Lambda 함수가 인터넷에 액세스할 수 있는 경우 퍼블릭 엔드포인트를 사용할 수 있습니다. 엔드포인트를 찾으려면 the section called “Secrets Manager 엔드포인트” 섹션을 참조하세요.

Lambda 함수가 인터넷에 액세스할 수 없는 VPC에서 실행되는 경우 VPC 내에서 Secrets Manager 서비스 프라이빗 엔드포인트를 구성하는 것이 좋습니다. 그러면 VPC가 퍼블릭 리전 엔드포인트로 전달된 요청을 가로채서 프라이빗 엔드포인트로 리디렉션할 수 있습니다. 자세한 정보는 VPC 엔드포인트을 참조하세요.

또는 VPC에 NAT 게이트웨이 또는 인터넷 게이트웨이를 추가하여 Lambda 함수를 통해 퍼블릭 Secrets Manager 엔드포인트에 액세스할 수 있도록 합니다. 그러면 VPC에서 트래픽이 퍼블릭 엔드포인트에 도달할 수 있습니다. 이렇게 하면 게이트웨이에 대한 IP 주소가 퍼블릭 인터넷에서 공격을 받을 수 있으므로 VPC가 더 많은 위험에 노출됩니다.

“createSecret” 이후 활동 없음

다음은 createSecret 이후에 교체가 중지될 수 있는 문제입니다.

VPC 네트워크 ACL은 HTTPS 트래픽 송수신을 허용하지 않습니다.

> 자세한 내용은 Amazon VPC 사용 설명서의 네트워크 ACL을 사용하여 서브넷에 대한 트래픽 제어를 참조하세요.

Lambda 함수 제한 시간 구성이 너무 짧아서 작업을 수행할 수 없습니다.

> 자세한 내용은 AWS Lambda 개발자 안내서의 Lambda 함수 옵션 구성을 참조하세요.

Secrets Manager VPC 엔드포인트는 할당된 보안 그룹에서 VPC CIDR 수신을 허용하지 않습니다.

> 자세한 내용은 Amazon VPC 사용 설명서의 보안 그룹을 사용하여 리소스에 대한 트래픽 제어를 참조하세요.

Secrets Manager VPC 엔드포인트 정책에서는 Lambda에서 VPC 엔드포인트를 사용하는 것을 허용하지 않습니다.

자세한 정보는 VPC 엔드포인트을 참조하세요.

암호는 교대로 사용자를 교체하여 사용하고, 슈퍼 사용자 암호는 Amazon RDS에서 관리하며, Lambda 함수는 RDS API에 액세스할 수 없습니다.

수퍼유저 암호가 다른 AWS 서비스에 의해 관리되는 대체 사용자 로테이션의 경우, Lambda rotation 함수는 서비스 엔드포인트를 호출하여 데이터베이스 연결 정보를 가져올 수 있어야 합니다. 데이터베이스 서비스의 VPC 엔드포인트를 구성하는 것이 좋습니다. 자세한 내용은 다음 섹션을 참조하세요.

- Amazon RDS 사용 설명서의 Amazon RDS API 및 인터페이스 VPC 엔드포인트.
- Amazon Redshift 관리 가이드의 VPC 엔드포인트 작업.

오류: “KMS에 대한 액세스가 허용되지 않습니다.”

`ClientError: An error occurred (AccessDeniedException) when calling the GetSecretValue operation: Access to KMS is not allowed`에서 보듯 교체 함수에는 암호를 암호화하는 데 사용된 KMS 키를 사용하여 암호를 해독할 권한이 없습니다. 권한 정책에 암호화 컨텍스트를 특정 보안 암호로 제한하는 조건이 있을 수도 있습니다. 필요한 권한에 대한 자세한 내용은 the section called “고객 관리형 키에 대한 정책 설명” 섹션을 참조하세요.

오류: “보안 암호 JSON에 키가 없습니다.”

Lambda 교체 함수를 사용하려면 보안 암호 값이 특정 JSON 구조에 있어야 합니다. 이 오류가 표시되면 JSON에 교체 함수에서 액세스하려는 키가 없을 수 있습니다. 각 보안 암호 유형별 JSON 구조에 대한 자세한 내용은 the section called “보안 암호의 JSON 구조” 섹션을 참조하세요.

오류: “setSecret: 데이터베이스에 로그인할 수 없음”

이 오류를 유발할 수 있는 문제는 다음과 같습니다.

교체 함수가 데이터베이스에 액세스할 수 없습니다.

작업 기간이 긴 경우(예: 5000ms 이상) Lambda 교체 함수가 네트워크를 통해 데이터베이스에 액세스하지 못할 수 있습니다.

데이터베이스 또는 인스턴스가 VPC의 Amazon EC2 인스턴스에서 실행 중인 경우, 동일한 VPC에서 Lambda 함수가 실행되도록 구성하는 것이 좋습니다. 그러면 교체 함수가 서비스와 직접 통신할 수 있습니다. 자세한 내용은 VPC 액세스 구성을 참조하세요.

Lambda 함수가 데이터베이스 또는 서비스에 액세스하도록 허용하려면 Lambda 교체 함수에 연결된 보안 그룹이 데이터베이스 또는 서비스에 아웃바운드 연결을 허용하는지 확인해야 합니다. 데이터베이스 또는 서비스에 연결된 보안 그룹이 Lambda 교체 함수에 인바운드 연결을 허용하는지도 확인해야 합니다.

보안 암호의 보안 인증이 올바르지 않습니다.

작업 기간이 짧은 경우 Lambda 교체 함수가 보안 암호의 보안 인증으로 인증하지 못할 수 있습니다. 명령을 사용하여 시크릿의 AWSCURRENT 및 AWSPREVIOUS 버전에 있는 정보로 수동으로 로그인하여 자격 증명을 확인합니다. AWS CLI get-secret-value

데이터베이스는 **scram-sha-256**을(를) 사용하여 암호를 암호화합니다.

데이터베이스가 Aurora PostgreSQL 버전 13 이상이고 scram-sha-256을(를) 사용하여 암호를 암호화하지만, 교체 함수가 scram-sha-256을(를) 지원하지 않는 libpq 버전 9 이하를 사용하는 경우, 교체 함수가 데이터베이스에 연결할 수 없습니다.

scram-sha-256 암호화를 사용하는 데이터베이스 사용자 확인 방법

- PostgreSQL 13을 위한 RDS의 SCRAM 인증 블로그에서 비-SCRAM 암호를 사용하는 사용자의 확인을 참조하세요.

해당 교체 함수가 사용하는 **libpq**의 버전 확인 방법

1. Linux 기반 컴퓨터의 Lambda 콘솔에서 교체 함수로 이동하여 배포 번들을 다운로드합니다. 작업 디렉터리에 zip 파일을 압축 해제합니다.
2. 명령줄의 작업 디렉터리에서 다음을 실행합니다:

   ```
   readelf -a libpq.so.5 | grep RUNPATH
   ```

3. 문자열 *PostgreSQL-9.4.x*이 표시되거나 메이저 버전이 10 미만인 경우 교체 함수가 scram-sha-256을(를) 지원하지 않는 것입니다.
   - scram-sha-256을(를) 지원하지 않는 교체 함수의 출력:

     ```
     0x000000000000001d (RUNPATH) Library runpath: [/
     local/p4clients/pkgbuild-a1b2c/workspace/build/
     PostgreSQL/PostgreSQL-9.4.x_client_only.123456.0/AL2_x86_64/
     DEV.STD.PTHREAD/build/private/tmp/brazil-path/build.libfarm/lib:/
     local/p4clients/pkgbuild-a1b2c/workspace/src/PostgreSQL/build/
     private/install/lib]
     ```

- scram-sha-256을(를) 지원하는 교체 함수의 출력:

  ```
  0x000000000000001d (RUNPATH) Library runpath: [/
  local/p4clients/pkgbuild-a1b2c/workspace/build/
  PostgreSQL/PostgreSQL-10.x_client_only.123456.0/AL2_x86_64/
  DEV.STD.PTHREAD/build/private/tmp/brazil-path/build.libfarm/lib:/
  local/p4clients/pkgbuild-a1b2c/workspace/src/PostgreSQL/build/
  private/install/lib]
  ```

> **Note**
>
> 2021년 12월 30일 이전에 자동 암호 교체를 설정한 경우, 교체 함수가 scram-sha-256을(를) 지원하지 않는 이전 버전의 libpq을(를) 번들로 제공합니다. scram-sha-256을(를) 지원하려면 교체 함수를 다시 생성해야 합니다.

데이터베이스에 SSL/TLS 액세스가 필요합니다.

데이터베이스에 SSL/TLS 연결이 필요하지만 교체 함수가 암호화되지 않은 연결을 사용하는 경우, 교체 함수가 데이터베이스에 연결할 수 없습니다. Amazon RDS(Oracle 및 Db2 제외) 및 Amazon DocumentDB에 대한 교체 함수는 사용 가능한 보안 소켓 계층(SSL) 또는 전송 계층 보안(TLS)을 사용하여 데이터베이스에 연결합니다. 그렇지 않으면 암호화되지 않은 연결을 사용합니다.

> **Note**
>
> 2021년 12월 20일 전에 자동 보안 암호 교체을 설정한 경우, 교체 함수는 SSL/TLS를 지원하지 않는 이전 템플릿을 기반으로 할 수 있습니다. SSL/TLS를 사용하는 연결을 지원하려면 교체 함수를 재생성해야 합니다.

교체 함수가 언제 생성되었는지 확인하려면

1. Secrets Manager 콘솔(https://console.aws.amazon.com/secretsmanager/)에서 보안 암호를 엽니다. 교체 구성(Rotation configuration) 섹션의 Lambda 교체 함수(Lambda rotation function)에서 arn:aws:lambda:*aws-region*:*123456789012*:function:*SecretsManagerMyRotationFunction* 과 같은 Lambda 함수 ARN(Lambda function ARN)을 볼 수 있습니다. ARN 끝에서 함수 이름을 복사합니다(이 예에서는 *SecretsManagerMyRotationFunction* 입니다).

2. AWS Lambda 콘솔 https://console.aws.amazon.com/lambda/ 의 함수에서 검색 상자에 Lambda 함수 이름을 붙여넣고 [Enter] 를 선택한 다음 Lambda 함수를 선택합니다.
3. 함수 세부 정보 페이지의 구성(Configuration) 탭에서, 태그(Tags) 아래의 aws:cloudformation:stack-name 키 옆에 있는 값을 복사합니다.
4. AWS CloudFormation 콘솔 https://console.aws.amazon.com/cloudformation 의 스택에서 검색 상자에 키 값을 붙여넣은 다음 Enter를 선택합니다.
5. 스택 목록은 Lambda 교체 함수를 생성한 스택만 표시되도록 필터링합니다. 생성된 날짜(Created date) 열에서 스택이 생성된 날짜를 확인합니다. 이것은 Lambda 교체 함수가 생성된 날짜입니다.

오류: “모듈 'lambda_function'을 가져올 수 없음”

Python 3.7에서 최신 버전의 Python으로 자동 업그레이드된 이전 Lambda 함수를 실행하는 경우 이 오류가 발생할 수 있습니다. 오류를 해결하려면 Lambda 함수 버전을 다시 Python 3.7로 변경한 다음 the section called “기존 교체 함수를 Python 3.7에서 3.9로 업그레이드” 합니다. 자세한 내용은 AWS re:Post에서 Secrets Manager Lambda 함수 교체가 “pg 모듈을 찾을 수 없음” 오류로 인해 실패한 이유는 무엇인가요?를 참조하세요.

기존 교체 함수를 Python 3.7에서 3.9로 업그레이드

2022년 11월 이전에 생성된 일부 교체 함수는 Python 3.7을 사용했습니다. 파이썬용 AWS SDK는 2023년 12월에 파이썬 3.7에 대한 지원을 중단했습니다. 자세한 내용은 AWS SDK 및 도구에 대한 Python 지원 정책 업데이트를 참조하십시오. Python 3.9를 사용하는 새 교체 함수로 전환하려면 기존 교체 함수에 런타임 속성을 추가하거나 교체 함수를 다시 생성할 수 있습니다.

Python 3.7을 사용하는 Lambda 교체 함수를 찾으려면

1. https://console.aws.amazon.com/lambda/ 에서 AWS Management Console 로그인하고 AWS Lambda 콘솔을 엽니다.
2. 함수 목록에서 **`SecretsManager`** 를 필터링합니다.
3. 필터링된 함수 목록의 런타임에서 Python 3.7을 찾습니다.

Python 3.9로 업그레이드하려면

- 옵션 1: 를 사용하여 회전 함수 다시 만들기 AWS CloudFormation
- 옵션 2: 를 사용하여 기존 회전 함수의 런타임을 업데이트합니다. AWS CloudFormation
- 옵션 3: AWS CDK 사용자의 경우 CDK 라이브러리를 업그레이드하세요.

옵션 1: 를 사용하여 회전 함수 다시 만들기 AWS CloudFormation

Secrets Manager 콘솔을 사용하여 로테이션을 켜면 Secrets Manager는 Lambda 로테이션 함수를 비롯한 필요한 리소스를 생성하는 AWS CloudFormation 데 사용합니다. 콘솔을 사용하여 로테이션을 켜거나 스택을 사용하여 로테이션 함수를 생성한 경우 동일한 AWS CloudFormation AWS CloudFormation 스택을 사용하여 새 이름으로 로테이션 함수를 다시 생성할 수 있습니다. 새 함수는 최신 버전의 Python을 사용합니다.

회전 함수를 만든 AWS CloudFormation 스택을 찾으려면

- Lambda 함수 세부 정보 페이지의 구성 탭에서 태그를 선택합니다. aws:cloudformation:stack-id 옆의 ARN을 확인합니다.

  스택 이름은 다음 예와 같이 ARN에 포함되어 있습니다.

  - ARN: `arn:aws:cloudformation:us-west-2:408736277230:stack/`**`SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda`**`-3CUDHZMDMBO8/79fc9050-2eef-11ed-`
  - 스택 이름: **`SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda`**

교체 함수를 다시 생성하려면(AWS CloudFormation)

1. 에서 AWS CloudFormation이름으로 스택을 검색한 다음 업데이트를 선택합니다.

   루트 스택 업데이트를 권장하는 대화 상자가 나타나면 루트 스택으로 이동을 선택한 다음 업데이트를 선택합니다.
2. 스택 업데이트 페이지에서 Designer에서 템플릿 편집을 선택한 다음 Designer에서 보기를 선택합니다.
3. Designer에서 템플릿 코드에 있는 `SecretRotationScheduleHostedRotationLambda` 의 `"functionName": "SecretsManagerTestRotationRDS"` 값을 새 함수 이름(예: JSON의 경우 **`"functionName": "SecretsManagerTestRotationRDSupdated"`**)으로 바꿉니다.
4. AWS CloudFormation 스택 워크플로를 계속 진행한 다음 제출을 선택합니다.

옵션 2: 를 사용하여 기존 회전 함수의 런타임을 업데이트합니다. AWS CloudFormation

Secrets Manager 콘솔을 사용하여 로테이션을 켜면 Secrets Manager는 Lambda 로테이션 함수를 비롯한 필요한 리소스를 생성하는 AWS CloudFormation 데 사용합니다. 콘솔을 사용하여 회전을 켜거나

스택을 사용하여 회전 함수를 생성한 경우 동일한 AWS CloudFormation AWS CloudFormation 스택을 사용하여 회전 함수의 런타임을 업데이트할 수 있습니다.

회전 함수를 생성한 AWS CloudFormation 스택을 찾으려면

- Lambda 함수 세부 정보 페이지의 구성 탭에서 태그를 선택합니다. aws:cloudformation:stack-id 옆의 ARN을 확인합니다.

  스택 이름은 다음 예와 같이 ARN에 포함되어 있습니다.

  - ARN: `arn:aws:cloudformation:us-west-2:408736277230:stack/`**`SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda`**`-3CUDHZMDMBO8/79fc9050-2eef-11ed-`
  - 스택 이름: **`SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda`**

교체 함수의 런타임을 업데이트하려면(AWS CloudFormation)

1. 에서 AWS CloudFormation이름으로 스택을 검색한 다음 업데이트를 선택합니다.

   루트 스택 업데이트를 권장하는 대화 상자가 나타나면 루트 스택으로 이동을 선택한 다음 업데이트를 선택합니다.
2. 스택 업데이트 페이지에서 Designer에서 템플릿 편집을 선택한 다음 Designer에서 보기를 선택합니다.
3. 디자이너의 템플릿 JSON에서, 언더, 언더 `SecretRotationScheduleHostedRotationLambdaProperties`, 추가에 대해 `Parameters` **`"runtime": "python3.9"`**
4. AWS CloudFormation 스택 워크플로를 계속 진행한 다음 제출을 선택합니다.

옵션 3: AWS CDK 사용자의 경우 CDK 라이브러리를 업그레이드하세요.

v2.94.0 AWS CDK 이전 버전을 사용하여 시크릿에 대한 로테이션을 설정한 경우 v2.94.0 이상으로 업그레이드하여 Lambda 함수를 업데이트할 수 있습니다. 자세한 내용은 AWS Cloud Development Kit (AWS CDK) v2 개발자 안내서를 참조하세요.

AWS Secrets Manager 시크릿을 즉시 교체하세요

교체가 구성된 보안 암호만 교체할 수 있습니다. 교체에 대한 보안 암호가 구성되어 있는지 확인하려면 콘솔에서 보안 암호를 확인하고 Rotation configuration(교체 구성) 섹션으로 스크롤합니다. Rotation status(교체 상태)가 Enabled(활성)이면 교체에 대한 보안 암호가 구성되어 있습니다. 그렇지 않은 경우 보안 암호 교체를 참조하세요.

보안 암호를 즉시 교체하려면(콘솔)

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호를 선택합니다.
3. 보안 암호 세부 정보 페이지의 교체 구성(Rotation configuration)에서 보안 암호 즉시 교체(Rotate secret immediately)를 선택합니다.
4. 보안 암호 교체(Rotate secret) 대화 상자에서 교체(Rotate)를 선택합니다.

AWS CLI

Example 보안 암호 즉시 교체

다음 `rotate-secret` 예에서는 즉시 교체를 시작합니다. 보안 암호에 교체가 미리 구성되어 있어야 합니다.

```
aws secretsmanager rotate-secret \
    --secret-id MyTestSecret
```

로테이션 스케줄

자동 교체를 켜면 cron() 또는 rate() 표현식을 사용하여 보안 암호 교체 일정을 설정할 수 있습니다. rate 표현식을 사용하면 시간 또는 일 간격으로 반복되는 교체 일정을 생성할 수 있습니다. cron 표현식을 사용하면 교체 간격보다 세부적인 교체 일정을 생성할 수 있습니다. Secrets Manager 교체 일정은 UTC 표준 시간대를 사용합니다. 보안 암호를 4시간마다 교체할 수 있습니다. Secrets Manager는 교체 기간 중 임의의 시점에 보안 암호를 교체합니다.

교체를 켜려면 다음을 참조하세요.

- the section called “관리형 교체”
- the section called “데이터베이스 보안 암호 자동 교체(콘솔)”

- the section called “비데이터베이스 비밀번호의 자동 교체 (콘솔)”

rate 표현식

Secrets Manager rate 표현식의 형식은 다음과 같습니다. 여기서 *Value*는 양의 정수이고 *Unit*은 hour, hours, day 또는 days이(가) 될 수 있습니다:

```
rate(Value Unit)
```

보안 암호를 4시간마다 교체할 수 있습니다. 예:

- `rate(4 hours)`은(는) 보안 암호가 4시간마다 교체된다는 뜻입니다.
- `rate(1 day)`은(는) 보안 암호가 매일 교체된다는 뜻입니다.
- `rate(10 days)`은(는) 보안 암호가 10일마다 교체된다는 뜻입니다.

시간 단위의 주기인 경우 기본 교체 시간은 자정에 시작하여 1시간 후에 종료됩니다. 지속 시간(Window duration)을 설정하여 교체 기간을 변경할 수 있습니다. 교체 기간은 그 다음 교체 기간까지 연장되지 않아야 합니다. 이를 확인하는 한 가지 방법은 교체 기간이 교체 사이의 간격 시간보다 작거나 같은지 확인하는 것입니다.

일 단위 주기인 경우, 기본 교체 기간은 자정에 시작하여 하루가 끝날 때 종료됩니다. 지속 시간(Window duration)을 설정하여 교체 기간을 변경할 수 있습니다. 교체 기간은 그 다음 UTC 날짜까지 연장되지 않아야 합니다. 이를 확인하는 한 가지 방법은 시작 시간과 지속 시간을 더한 값이 24시간 이하인지 확인하는 것입니다.

cron 표현식

cron 표현식의 형식은 다음과 같습니다.

```
cron(Minutes Hours Day-of-month Month Day-of-week Year)
```

시간 증분을 포함하는 cron 표현식은 매일 재설정됩니다. 예를 들어, `cron(0 4/12 * * ? *)`은(는) 오전 4시, 오후 4시, 다음 날 오전 4시, 오후 4시를 의미합니다. Secrets Manager 교체 일정은 UTC 표준 시간대를 사용합니다.

시간 단위의 일정에서 기본 교체 기간은 1시간 후에 종료됩니다. 지속 시간(Window duration)을 설정하여 교체 기간을 변경할 수 있습니다. 교체 기간은 그 다음 교체 기간까지 연장되지 않아야 합니다. 보안 암호를 4시간마다 교체할 수 있습니다.

일정 예약 예	표현식
자정부터 8시간마다.	`cron(0 /8 * * ? *)`
오전 8시부터 8시간마다.	`cron(0 8/8 * * ? *)`
오전 2시부터 10시간마다. 교체 기간은 2:00시, 12:00, 22:00, 다음 날 2:00시, 12:00, 22:00에 시작됩니다.	`cron(0 2/10 * * ? *)`
매일 오전 10시	`cron(0 10 * * ? *)`
매주 토요일 오후 6시.	`cron(0 18 ? * SAT *)`
매월 1일 오전 8시.	`cron(0 8 1 * ? *)`
매 3개월마다 첫 번째 일요일 오전 1시.	`cron(0 1 ? 1/3 SUN#1 *)`
매월 마지막날 오후 5시.	`cron(0 17 L * ? *)`
월요일부터 금요일까지 오전 8시.	`cron(0 8 ? * MON-FRI *)`
매월 1일과 15일 오후 4시.	`cron(0 16 1,15 * ? *)`
매월 첫 번째 일요일 자정.	`cron(0 0 ? * SUN#1 *)`

Secrets Manager의 Cron 표현식 요구 사항

Secrets Manager에는 cron 표현식에 사용할 수 있는 항목에 대한 제한 사항이 몇 가지 있습니다. Secrets Manager 교체 기간은 정시에 시작되므로 Secrets Manager 대한 cron 표현식에서 minutes(분) 필드의 값은 0이 되어야 합니다. Secrets Manager는 1년 이상 간격의 교체 일정을 지원하지 않으므로 Year 필드의 값은 *여야 합니다. 다음 표에 사용할 수 있는 옵션이 나와 있습니다.

필드	값	와일드카드
Minutes	0이어야 함	None
Hours	0~23	/(슬래시)를 사용하여 증분을 지정합니다. 예를 들어 2/10은

필드	값	와일드카드
		(는) 오전 2시부터 10시간마다를 의미합니다. 보안 암호를 4시간마다 교체할 수 있습니다.

필드	값	와일드카드
D ay-of-month	1~31	,(쉼표)를 사용하여 추가 값을 포함합니다. 예를 들어 `1,15`은(는) 해당 월의 1일과 15일을 의미합니다. -(대시)를 사용하여 범위를 지정합니다. 예를 들어 `1-15`은(는) 해당 월의 1일~15일을 의미합니다. *(별표)를 사용하여 필드에 모든 값을 포함합니다. 예를 들어 `*`은(는) 해당 월의 모든 날짜를 의미합니다. ?(물음표) 와일드카드는 어떤 한 가지나 다른 것을 지정합니다. 동일한 cron 표현식에 `Day-of-month` 와 `Day-of-week` 필드를 지정할 수 없습니다. 이 필드 중 하나에 값을 지정하는 경우에는 다른 필드에서 반드시 ?(물음표)를 사용해야 합니다. /(슬래시)를 사용하여 증분을 지정합니다. 예를 들어, `1/2`은(는) 1일째부터 이틀마다, 즉 1일째, 3일째, 5일째 등을 의미합니다. L을 사용하여 해당 월의 마지막 날을 지정합니다. ***DAY***L을 사용하여 해당 월의 마지막 명명일을 지정합니다. 예

필드	값	와일드카드
		를 들어 SUNL은(는) 해당 월의 마지막 일요일을 의미합니다.
Month	1~12 또는 JAN~DEC	,(쉼표)를 사용하여 추가 값을 포함합니다. 예를 들어, JAN,APR,JUL,OCT 은(는) 1월, 4월, 7월, 10월을 의미합니다. -(대시)를 사용하여 범위를 지정합니다. 예를 들어 1-3은(는) 해당 연도의 1월~3월을 의미합니다. *(별표)를 사용하여 필드에 모든 값을 포함합니다. 예를 들어 *은(는) 모든 월을 의미합니다. /(슬래시)를 사용하여 증분을 지정합니다. 예를 들어, 1/3은(는) 첫번째 월에서 시작하여 매 3개월마다, 즉 첫째 달, 넷째 달, 일곱째 달, 열번째 달을 의미합니다.

필드	값	와일드카드
D ay-of-week	1~7 또는 SUN~SAT	#을 사용하여 한 달 내의 요일을 지정합니다. 예를 들어 TUE#3은 그 달의 세 번째 화요일을 의미합니다. ,(쉼표)를 사용하여 추가 값을 포함합니다. 예를 들어 1,4은(는) 해당 주의 첫째 요일과 넷째 요일을 의미합니다. -(대시)를 사용하여 범위를 지정합니다. 예를 들어 1-4은(는) 해당 주의 첫째 요일~넷째 요일을 의미합니다. *(별표)를 사용하여 필드에 모든 값을 포함합니다. 예를 들어 *은(는) 해당 주의 모든 요일을 의미합니다. ?(물음표) 와일드카드는 어떤 한 가지나 다른 것을 지정합니다. 동일한 cron 표현식에 Day-of-month 와 Day-of-week 필드를 지정할 수 없습니다. 이 필드 중 하나에 값을 지정하는 경우에는 다른 필드에서 반드시 ?(물음표)를 사용해야 합니다. /(슬래시)를 사용하여 증분을 지정합니다. 예를 들어, 1/2은(는) 해당 주의 첫 번째 요일부터 시작하여 격일로, 즉 첫째 요일, 셋째 요일, 다섯째 요일, 일곱째 요일을 의미합니다.

필드	값	와일드카드
		L을 사용하여 해당 주의 마지막 요일을 지정합니다.
Year	*여야 합니다.	None

로테이션되지 않은 비밀 찾기

AWS Config 이를 사용하여 비밀이 표준에 따라 순환하고 있는지 확인할 수 있습니다. AWS Config 규칙을 사용하여 비밀에 대한 내부 보안 및 규정 준수 요구 사항을 정의합니다. 그러면 AWS Config 규칙을 준수하지 않는 비밀을 식별할 수 있습니다. 또한 보안 암호 메타데이터, 교체 구성, 보안 암호 암호화에 사용되는 KMS 키, Lambda 교체 함수 및 보안 암호와 연결된 태그에 대한 변경 사항을 추적할 수 있습니다.

조직 내 여러 곳에 비밀이 있는 경우 해당 구성 AWS 계정 및 AWS 리전 규정 준수 데이터를 집계할 수 있습니다. 자세한 내용은 다중 계정 다중 지역 데이터 집계를 참조하십시오.

비밀의 순환 여부를 평가하려면

1. AWS Config 규칙을 사용한 리소스 평가의 지침을 따르고 다음 규칙 중 하나를 선택하세요.
   - `secretsmanager-rotation-enabled-check` — Secrets Manager에 저장된 보안 암호에 대해 교체가 구성되었는지 확인합니다.
   - `secretsmanager-scheduled-rotation-success-check`— 마지막으로 성공한 교체가 설정된 교체 주기 내에 있는지 확인합니다. 최소 확인 주기는 매일입니다.
   - `secretsmanager-secret-periodic-rotation` - 보안 암호를 지정된 일수 내에 교체했는지 확인합니다.
2. 비밀이 규정을 준수하지 않는 경우 AWS Config 알리도록 구성할 수도 있습니다. 자세한 내용은 Amazon SNS 주제로 AWS Config 보내는 알림을 참조하십시오.

Secrets Manager에서 자동 로테이션 취소하기

암호에 대해 자동 회전을 구성한 후 암호 회전을 중지하려면 회전을 취소할 수 있습니다.

자동 회전을 취소하려면

1. https://console.aws.amazon.com/secretsmanager/에서 Secrets Manager 콘솔을 엽니다.
2. 보안 암호를 선택합니다.
3. 비밀 세부 정보 페이지의 순환 구성에서 순환 편집을 선택합니다.
4. 순환 구성 편집 대화 상자에서 자동 순환을 끄고 저장을 선택합니다.

   Secrets Manager는 순환 구성 정보를 보관하므로 나중에 순환을 다시 설정하기로 결정한 경우 사용할 수 있습니다.

AWS Secrets Manager 다른 AWS 서비스에서 관리하는 비밀

많은 AWS 서비스가 시크릿을 저장하고 사용합니다 AWS Secrets Manager. 이러한 보안 암호는 관리형 보안 암호인 경우도 있습니다. 즉, 이러한 보안 암호를 생성한 서비스가 이를 관리하는 데 도움이 됩니다. 예를 들어 일부 관리형 보안 암호에는 관리형 교체가 포함되므로 직접 교체를 구성할 필요가 없습니다. 관리 서비스는 사용자가 보안 암호를 업데이트하거나 복구 기간 없이 삭제하지 못하도록 제한할 수도 있습니다. 이렇게 하면 보안 암호에 의존하여 관리 서비스가 수행되므로 중단을 방지하는 데 도움이 됩니다.

관리형 보안 암호는 식별하기 쉽도록 관리 서비스 ID가 포함된 명명 규칙을 사용합니다.

```
Secret name: ServiceID!MySecret
Secret ARN : arn:aws:us-east-1:ServiceID!MySecret-a1b2c3
```

보안 암호를 관리하는 서비스의 ID

- `appflow` – the section called “아마존 AppFlow”
- `databrew` – the section called “AWS Glue DataBrew”
- `datasync` – the section called “AWS DataSync”
- `directconnect` – the section called “AWS Direct Connect”
- `ecs-sc` – the section called “Amazon Elastic Container Service”
- `events` – the section called “아마존 EventBridge”
- `marketplace-deployment` – the section called “AWS Marketplace”
- `opsworks-cm` – the section called “AWS OpsWorks for Chef Automate”
- `rds` – the section called “Amazon RDS”
- `redshift` – the section called “Amazon Redshift”
- `sqlworkbench` – the section called “Amazon Redshift 쿼리 편집기 v2”

다른 AWS 서비스에서 관리하는 비밀을 찾으려면 관리 암호 찾기를 참조하십시오.

비밀을 사용하는 서비스의 전체 목록은 을 참조하십시오비밀을 사용하는 서비스.

AWSAWS Secrets Manager 시크릿을 사용하는 서비스

다음 각 기능이 Secrets Manager와 AWS 서비스 통합되는 방법에 대한 정보를 확인하십시오.

- AWS App Runner 사용 방법 AWS Secrets Manager
- AWS App2Container의 사용 방식 AWS Secrets Manager
- 사용 방법 AWS AppConfigAWS Secrets Manager
- 아마존이 AppFlow 사용하는 방법 AWS Secrets Manager
- AWS AppSync 사용 방법 AWS Secrets Manager
- Amazon Athena의 AWS Secrets Manager활용 방식
- Amazon Aurora가 사용하는 방법 AWS Secrets Manager
- 사용 방법 AWS CodeBuildAWS Secrets Manager
- Amazon 데이터 파이어호스가 사용하는 방법 AWS Secrets Manager
- 사용 방법 AWS DataSyncAWS Secrets Manager
- 아마존이 DataZone 사용하는 방법 AWS Secrets Manager
- 사용 방법 AWS Direct Connect : AWS Secrets Manager
- AWS Directory Service 사용 방법 AWS Secrets Manager
- Amazon DocumentDB(MongoDB와 호환)의 AWS Secrets Manager활용 방식
- 사용 AWS Elastic Beanstalk 방법 AWS Secrets Manager
- Amazon Elastic 컨테이너 레지스트리를 사용하는 방법 AWS Secrets Manager
- Amazon Elastic Container Service
- 아마존이 ElastiCache 사용하는 방법 AWS Secrets Manager
- AWS Elemental Live 사용 방법 AWS Secrets Manager
- 사용 방법 AWS Elemental MediaConnectAWS Secrets Manager
- AWS Elemental MediaConvert 사용 방법 AWS Secrets Manager
- 사용 방법 AWS Elemental MediaLiveAWS Secrets Manager
- AWS Elemental MediaPackage 사용 방법 AWS Secrets Manager
- 사용 방법 AWS Elemental MediaTailorAWS Secrets Manager
- Amazon EMR에서 Secrets Manager를 사용하는 방법

- 아마존이 EventBridge 사용하는 방법 AWS Secrets Manager
- Amazon AWS Secrets Manager FSx가 비밀을 사용하는 방법
- AWS Glue DataBrew 사용 방법 AWS Secrets Manager
- AWS Glue Studio에서 사용하는 방법 AWS Secrets Manager
- 사용 방법 AWS IoT SiteWiseAWS Secrets Manager
- Amazon Kendra가 사용하는 방법 AWS Secrets Manager
- Amazon Kinesis Video Streams가 사용하는 방법 AWS Secrets Manager
- 사용 방법 AWS Launch WizardAWS Secrets Manager
- Amazon Lookout for Metrics의 AWS Secrets Manager활용 방식
- 아마존 매니지드 Grafana가 사용하는 방법 AWS Secrets Manager
- 사용 방법 AWS Managed ServicesAWS Secrets Manager
- Amazon Managed Streaming for Apache Kafka의 AWS Secrets Manager활용 방식
- Apache 에어플로우용 Amazon 관리형 워크플로를 사용하는 방법 AWS Secrets Manager
- AWS Marketplace
- AWS Migration Hub 사용 방법 AWS Secrets Manager
- AWS Panorama Secrets Manager를 사용하는 방법
- AWS ParallelCluster 사용 방법 AWS Secrets Manager
- Amazon Q가 Secrets Manager를 사용하는 방법
- AWS OpsWorks for Chef Automate 사용 방법 AWS Secrets Manager
- 아마존이 QuickSight 사용하는 방법 AWS Secrets Manager
- Amazon RDS
- 아마존 Redshift의 사용 방법 AWS Secrets Manager
- Amazon Redshift 쿼리 편집기 v2
- 아마존이 SageMaker 사용하는 방법 AWS Secrets Manager
- 사용 방법 AWS Schema Conversion ToolAWS Secrets Manager
- AWS Toolkit for JetBrains 사용 방법 AWS Secrets Manager
- 시크릿을 AWS Transfer Family 사용하는 AWS Secrets Manager 방법
- AWS Wickr가 비밀을 사용하는 방법 AWS Secrets Manager

AWS App Runner 사용 방법 AWS Secrets Manager

AWS App Runner 소스 코드 또는 컨테이너 이미지를 AWS 클라우드의 확장 가능하고 안전한 웹 애플리케이션으로 직접 배포할 수 있는 빠르고 간단하며 비용 효율적인 방법을 제공하는 AWS 서비스입니다. 새로운 기술을 배우거나, 사용할 컴퓨팅 서비스를 결정하거나, AWS 리소스를 프로비저닝하고 구성하는 방법을 몰라도 됩니다.

App Runner를 사용하면 서비스를 만들거나 서비스 구성을 업데이트할 때 보안 암호와 구성을 서비스의 환경 변수로 참조할 수 있습니다. 자세한 내용은 AWS App Runner 개발자 가이드의 환경 변수 참조 및 환경 변수 관리를 참조하세요.

AWS App2Container의 사용 방식 AWS Secrets Manager

AWS App2Container 온프레미스 데이터 센터 또는 가상 머신에서 실행되는 애플리케이션을 Amazon ECS, Amazon EKS 또는 에서 관리하는 컨테이너에서 실행되도록 지원하는 명령줄 도구입니다. AWS App Runner

App2Container는 Secrets Manager를 사용하여 원격 명령을 실행하기 위해 작업자 머신을 애플리케이션 서버에 연결하는 데 필요한 자격 증명을 관리합니다. 자세한 내용은 App2Container 사용 설명서의 AWS App2Container의 암호 관리를 참조하십시오.AWS

사용 방법 AWS AppConfigAWS Secrets Manager

AWS AppConfig 애플리케이션 구성을 생성, 관리 및 신속하게 배포하는 데 사용할 수 있는 기능입니다. AWS Systems Manager 구성에는 Secrets Manager에 저장된 자격 증명 데이터 또는 기타 중요한 정보가 포함될 수 있습니다. 자유 형식 구성 프로필을 만들 때 Secrets Manager를 구성 데이터의 소스로 선택할 수 있습니다. 자세한 내용은AWS AppConfig 사용 설명서의 Creating a freeform configuration profile(자유 형식 구성 프로필 생성)을 참조하세요. 자동 교체가 설정된 시크릿을 AWS AppConfig 처리하는 방법에 대한 자세한 내용은 AWS AppConfig 사용 설명서의 Secrets Manager 키 로테이션을 참조하십시오.

아마존이 AppFlow 사용하는 방법 AWS Secrets Manager

AppFlow Amazon은 Salesforce와 같은 서비스형 소프트웨어 (SaaS) 애플리케이션과 Amazon Simple Storage Service (Amazon S3) 및 Amazon Redshift와 같은 서비스형 소프트웨어 (SaaS) 애플리케이션 간에 데이터를 안전하게 교환할 수 있게 해주는 완전 관리형 통합 서비스입니다. AWS 서비스

AppFlowAmazon에서는 SaaS 애플리케이션을 소스 또는 대상으로 구성할 때 연결을 생성합니다. 인증 토큰, 사용자 이름, 암호 등 SaaS 애플리케이션에 연결하는 데 필요한 정보가 여기에 포함됩니다. Amazon은 접두사가 붙은 `appflow` Secrets Manager의 관리 비밀에 연결 데이터를 AppFlow 저장합니다. 암호 저장 비용은 Amazon 요금에 포함되어 AppFlow 있습니다. 자세한 내용은 Amazon AppFlow 사용 설명서의 AppFlow Amazon에서의 데이터 보호를 참조하십시오.

AWS AppSync 사용 방법 AWS Secrets Manager

AWS AppSync 애플리케이션 개발자가 Amazon DynamoDB 및 HTTP API를 비롯한 여러 소스의 데이터를 결합할 수 있도록 강력하고 확장 가능한 GraphQL 인터페이스를 제공합니다. AWS Lambda

AWS AppSync CLI 명령을 `rds execute-statement`사용하여 시크릿의 자격 증명을 사용하여 Amazon RDS에 연결합니다. 자세한 내용은 AWS AppSync 개발자 가이드의 자습서: Aurora Serverless를 참조하세요.

Amazon Athena의 AWS Secrets Manager활용 방식

Amazon Athena는표준 SQL을 사용하여 Amazon Simple Storage Service(S3)에 있는 데이터를 직접 간편하게 분석할 수 있는 대화형 쿼리 서비스입니다.

Amazon Athena 데이터 원본 커넥터는 Secrets Manager 보안 암호로 Athena 연합 쿼리 기능을 사용해 데이터를 쿼리할 수 있습니다. 자세한 내용은 Amazon Athena 사용 설명서의 Amazon Athena 페더레이션 쿼리 사용을 참조하세요.

Amazon Aurora가 사용하는 방법 AWS Secrets Manager

Amazon Aurora는 MySQL 및 PostgreSQL과 호환되는 완전 관리형 관계형 데이터베이스 엔진입니다.

Aurora의 마스터 사용자 자격 증명을 관리하기 위해 Aurora에서 관리 암호를 생성할 수 있습니다. 보안 암호에 대한 요금이 청구됩니다. 또한 Aurora는 이러한 자격 증명의 교체도 관리합니다. 자세한 내용은 Amazon Aurora 사용 설명서의 Amazon Aurora 및 AWS Secrets Manager을(를) 사용한 암호 관리를 참조하세요.

다른 Aurora 자격 증명에 대해서는 을 참조하십시오. the section called “데이터베이스 보안 암호 생성”

Amazon RDS 데이터 API를 호출할 때 Secrets Manager의 보안 암호를 사용하여 데이터베이스에 대한 자격 증명을 전달할 수 있습니다. 자세한 내용을 알아보려면 Amazon Aurora 사용 설명서의 Aurora Serverless에 데이터 API 사용을 참조하세요.

Amazon RDS 쿼리 편집기를 사용하여 데이터베이스에 연결하면 데이터 베이스에 대한 보안 인증 정보를 Secrets Manager에 저장할 수 있습니다. 자세한 내용은 Amazon RDS 사용 설명서의 쿼리 편집기 사용을 참조하세요.

사용 방법 AWS CodeBuildAWS Secrets Manager

AWS CodeBuild 클라우드의 완전 관리형 빌드 서비스입니다. CodeBuild 소스 코드를 컴파일하고, 단위 테스트를 실행하고, 배포할 준비가 된 아티팩트를 생성합니다.

Secrets Manager를 사용하여 프라이빗 레지스트리 자격 증명을 저장할 수 있습니다. 자세한 내용은 CodeBuild AWS CodeBuild 사용 설명서의 AWS Secrets Manager 샘플이 포함된 프라이빗 레지스트리를 참조하십시오.

Amazon 데이터 파이어호스가 사용하는 방법 AWS Secrets Manager

Amazon Data Firehose를 사용하여 다양한 스트리밍 대상으로 실시간 스트리밍 데이터를 전송할 수 있습니다. 대상에 자격 증명 또는 키가 필요한 경우 Firehose는 런타임에 Secrets Manager에서 암호를 검색하여 대상에 연결합니다. 자세한 내용은 Amazon Data Firehose 개발자 안내서의 Amazon Data AWS Secrets Manager Firehose에서 인증을 참조하십시오.

사용 방법 AWS DataSyncAWS Secrets Manager

AWS DataSync 스토리지 시스템과 서비스 간의 데이터 이동을 간소화, 자동화 및 가속화하는 온라인 데이터 전송 서비스입니다. DataSync Discovery를 사용하면 마이그레이션을 가속화할 수 있습니다. AWS

온프레미스 스토리지 시스템에 대한 정보를 수집하기 위해 DataSync Discovery는 스토리지 시스템의 관리 인터페이스에 대한 자격 증명을 사용합니다. DataSync 이러한 자격 증명을 접두사와 함께 `datasync` Secrets Manager 관리 비밀에 저장합니다. 보안 암호에 대한 요금이 청구됩니다. 자세한 내용은 사용 AWS DataSync 설명서의 DataSync Discovery에 온프레미스 스토리지 시스템 추가를 참조하십시오.

아마존이 DataZone 사용하는 방법 AWS Secrets Manager

DataZone Amazon은 데이터를 카탈로그, 검색, 관리, 공유 및 분석할 수 있는 데이터 관리 서비스입니다. 작업을 사용하여 크롤링되는 Amazon Redshift 클러스터의 테이블 및 뷰의 데이터 자산을 사용할

수 있습니다. AWS Glue 크롤러 Amazon Redshift에 연결하려면 Secrets Manager 시크릿으로 아마존 DataZone 자격 증명을 제공해야 합니다. 자세한 내용은 Amazon 사용 DataZone 설명서의 새 AWS Glue 연결을 사용하여 Amazon Redshift 데이터베이스의 데이터 소스 생성을 참조하십시오.

사용 방법 AWS Direct Connect : AWS Secrets Manager

AWS Direct Connect 표준 이더넷 광섬유 케이블을 통해 내부 네트워크를 특정 AWS Direct Connect 위치에 연결합니다. 이 연결을 통해 대중에게 직접 연결되는 가상 인터페이스를 만들 수 있습니다. AWS 서비스

AWS Direct Connect 접두사가 있는 관리 암호에 연결 연결 키 이름과 연결 연결 키 쌍 (CKN/CAK 쌍)을 저장합니다. `directconnect` 보안 비용은 요금에 포함되어 있습니다. AWS Direct Connect암호를 업데이트하려면 Secrets Manager AWS Direct Connect 대신 를 사용해야 합니다. 자세한 내용은 AWS Direct Connect 사용 설명서 Associate a MACsec CKN/CAK with a LAG(MACsec CKN/CAK와 LAG 연결)을 참조하세요.

AWS Directory Service 사용 방법 AWS Secrets Manager

AWS Directory Service Microsoft Active Directory (AD) 를 다른 AWS 서비스와 함께 사용할 수 있는 여러 가지 방법을 제공합니다. 보안 인증에 보안 암호를 사용하여 Amazon EC2 인스턴스를 디렉터리에 조인할 수 있습니다. 자세한 정보는 AWS Direct Connect 사용 설명서의 다음 섹션을 참조하세요:

- Linux EC2 인스턴스를 관리형 AWS Microsoft AD 디렉터리에 원활하게 조인합니다.
- AD Connector 디렉터리에 Linux EC2 인스턴스 원활하게 조인
- Simple AD 디렉터리에 Linux EC2 인스턴스 원활하게 조인

Amazon DocumentDB(MongoDB와 호환)의 AWS Secrets Manager 활용 방식

Amazon DocumentDB에서 사용자는 암호와 함께 클러스터에 대해 인증합니다. AWS Secrets Manager을 이용하면 코드의 암호를 포함해 하드 코딩된 자격 증명을 Secrets Manager에서 프로그래밍 방식으로 보안 암호를 검색하도록 하는 API 호출로 바꿀 수 있습니다. 자세한 내용은 the section called “데이터베이스 보안 암호 생성” 및 Amazon DocumentDB 개발자 가이드의 Amazon DocumentDB 사용자 관리를 참조하세요.

사용 AWS Elastic Beanstalk 방법 AWS Secrets Manager

를 사용하면 애플리케이션을 실행하는 인프라에 대해 알 필요 없이 AWS 클라우드에서 애플리케이션을 빠르게 배포하고 관리할 수 있습니다. AWS Elastic Beanstalk Elastic Beanstalk는 Dockerfile에 명시된 이미지를 빌드하거나 원격 Docker 이미지를 가져와 Docker 환경을 시작할 수 있습니다. Elastic Beanstalk는 개인 리포지토리를 호스팅하는 온라인 레지스트리를 사용하여 인증하는 데 Secrets Manager 보안 암호를 사용합니다. 자세한 내용은 AWS Elastic Beanstalk 개발자 안내서에서 Docker 구성을 참조하세요.

Amazon Elastic 컨테이너 레지스트리를 사용하는 방법 AWS Secrets Manager

Amazon Elastic 컨테이너 레지스트리 (Amazon ECR) 는 AWS 안전하고 확장 가능하며 신뢰할 수 있는 관리형 컨테이너 이미지 레지스트리 서비스입니다. Docker CLI를 사용하거나 선호하는 클라이언트를 사용하여 이미지를 리포지토리로 푸시 및 풀링할 수 있습니다. Amazon ECR 프라이빗 레지스트리에서 캐시하려는 이미지가 포함된 각 업스트림 레지스트리에 대해 풀스루 캐시 규칙을 생성해야 합니다. 인증이 필요한 업스트림 레지스트리의 경우 보안 인증 정보를 Secrets Manager 보안 암호로 저장해야 합니다. Amazon ECR 또는 Secrets Manager 콘솔에서 Secrets Manager 보안 암호를 생성할 수 있습니다. 자세한 내용은 Amazon ECR 사용 설명서의 풀스루 캐시 규칙 생성을 참조하십시오.

Amazon Elastic Container Service

Amazon Elastic Container Service(Amazon ECS)는 컨테이너 애플리케이션을 쉽게 배포, 관리 및 확대할 수 있도록 도와주는 완전 관리형 컨테이너 오케스트레이션 서비스입니다. Secrets Manager 보안 암호를 참조하여 민감한 데이터를 컨테이너에 주입할 수 있습니다. 자세한 내용은 Amazon Elastic Container Service 개발자 안내서의 다음 페이지를 참조하세요.

- 자습서: Secrets Manager 보안 암호를 사용해 민감한 데이터 지정
- 애플리케이션을 통해 프로그래밍 방식으로 보안 암호 검색
- 환경 변수를 통해 보안 암호 검색
- 로깅 구성을 위한 보안 암호 검색

Amazon ECS는 컨테이너용 Windows File Server 볼륨용 FSx를 지원합니다. Amazon ECS는 Secrets Manager 보안 암호에 저장된 보안 인증 정보를 사용하여 Active Directory에 도메인 가입을 하고 FSx for Windows File Server 파일 시스템을 연결합니다. 자세한 내용은 Amazon Elastic Container Service

개발자 안내서의 자습서: Amazon ECS에서 FSx for Windows File Server 파일 시스템 사용 및 FSx for Windows File Server 볼륨을 참조하세요.

레지스트리 자격 증명과 함께 Secrets Manager 암호를 사용하여 AWS 인증이 필요한 외부 프라이빗 레지스트리의 컨테이너 이미지를 참조할 수 있습니다. 자세한 내용을 알아보려면 Amazon Elastic Container Service 개발자 안내서의 태스크에 대한 프라이빗 레지스트리 인증을 참조하세요.

Amazon ECS Service Connect를 사용하는 경우, Amazon ECS는 Secrets Manager의 관리 암호를 사용하여 AWS Private Certificate Authority TLS 인증서를 저장합니다. 암호 저장 비용은 Amazon ECS 요금에 포함되어 있습니다. 암호를 업데이트하려면 Secrets Manager 대신 Amazon ECS를 사용해야 합니다. 자세한 내용은 Amazon Elastic 컨테이너 서비스 개발자 안내서의 Service Connect를 통한 TLS를 참조하십시오.

아마존이 ElastiCache 사용하는 방법 AWS Secrets Manager

ElastiCache 에서는 역할 기반 액세스 제어 (RBAC) 라는 기능을 사용하여 클러스터를 보호할 수 있습니다. 이러한 보안 인증 정보는 Secrets Manager에 저장할 수 있습니다. Secrets Manager는 이런 유형의 보안 암호에 대한 교체 템플릿을 제공합니다. 자세한 내용은 Amazon ElastiCache 사용 설명서의 사용자 암호 자동 교체를 참조하십시오.

AWS Elemental Live 사용 방법 AWS Secrets Manager

AWS Elemental Live 브로드캐스트 및 스트리밍 전송을 위한 라이브 출력을 생성할 수 있는 실시간 비디오 서비스입니다.

AWS Elemental Live 비밀 ARN을 사용하여 Secrets Manager로부터 암호화 키가 포함된 암호를 가져옵니다. Elemental 라이브는 암호화 키를 사용하여 비디오를 암호화/해독합니다. 자세한 내용은 Elemental Live 사용 설명서의 런타임 시 전송 MediaConnect 방식을 참조하십시오. AWS Elemental Live

사용 방법 AWS Elemental MediaConnectAWS Secrets Manager

AWS Elemental MediaConnect 방송사 및 기타 프리미엄 비디오 제공업체가 라이브 비디오를 안정적으로 수집하여 내부 또는 외부의 여러 대상에 쉽게 배포할 수 있도록 하는 서비스입니다. AWS 클라우드 AWS 클라우드

정적 키 암호화를 사용하여 소스, 출력, 권한 부여를 보호할 수 있으며 암호화 키를 AWS Secrets Manager에 저장할 수 있습니다. 자세한 내용은 사용 설명서의 정적 키 암호화를 참조하십시오 AWS Elemental MediaConnect.AWS Elemental MediaConnect

AWS Elemental MediaConvert 사용 방법 AWS Secrets Manager

AWS Elemental MediaConvert 모든 규모의 미디어 라이브러리를 보유한 콘텐츠 소유자 및 배포자에게 확장 가능한 비디오 처리를 제공하는 파일 기반 비디오 처리 서비스입니다. 칸타르 워터마크를 MediaConvert 인코딩하는 데 사용하려면 Secrets Manager를 사용하여 칸타르 자격 증명을 저장합니다. 자세한 내용은 사용자 안내서의 출력에서 AWS Elemental MediaConvert 오디오 워터마킹을 위한 Kantar 사용을 참조하십시오.AWS Elemental MediaConvert

사용 방법 AWS Elemental MediaLiveAWS Secrets Manager

AWS Elemental MediaLive 브로드캐스트 및 스트리밍 전송을 위한 라이브 출력을 생성할 수 있는 실시간 비디오 서비스입니다. 조직에서 AWS Elemental MediaLive 또는 AWS Elemental MediaConnect와 함께 AWS Elemental Link 장치를 사용하는 경우 장치를 배포하고 장치를 구성해야 합니다. 자세한 내용은 MediaLive 사용 설명서의 신뢰할 수 있는 MediaLive 개체로 설정을 참조하십시오.

AWS Elemental MediaPackage 사용 방법 AWS Secrets Manager

AWS Elemental MediaPackage 에서 실행되는 just-in-time 비디오 패키징 및 원본 생성 서비스입니다. AWS 클라우드 MediaPackage를 사용하면 매우 안전하고 확장 가능하며 신뢰할 수 있는 비디오 스트림을 다양한 재생 디바이스 및 CDN(콘텐츠 전송 네트워크)에 전달할 수 있습니다 자세한 내용은 사용 설명서의 CDN 인증을 위한 Secrets Manager 액세스를 참조하십시오.AWS Elemental MediaPackage

사용 방법 AWS Elemental MediaTailorAWS Secrets Manager

AWS Elemental MediaTailor 에서 실행되는 확장 가능한 광고 삽입 및 채널 어셈블리 서비스입니다. AWS 클라우드

MediaTailor 소스 위치에 대한 Secrets Manager 액세스 토큰 인증을 지원합니다. Secrets Manager 액세스 토큰 인증을 사용하면 Secrets Manager 암호를 MediaTailor 사용하여 오리진에 대한 요청을 인증합니다. 자세한 내용은 AWS Elemental MediaTailor 사용 설명서의 AWS Secrets Manager 액세스 토큰 인증 구성을 참조하십시오.

Amazon EMR에서 Secrets Manager를 사용하는 방법

Amazon EMR은 Apache Hadoop 및 Apache Spark와 같은 빅 데이터 프레임워크를 실행하여 방대한 양의 데이터를 처리하고 분석할 수 있도록 간소화하는 플랫폼입니다. AWS 이러한 프레임워크와 함께 Apache Hive 및 Apache Pig와 같은 관련 오픈 소스 프로젝트를 사용하는 경우, 분석용 데이터와 비즈니스 인텔리전스 워크로드를 처리할 수 있습니다. 또한 Amazon EMR을 사용하여 대량의 데이터를

Amazon S3 및 Amazon DynamoDB 같은 다른 데이터 스토어 및 데이터베이스로 변환하고 다른 AWS 데이터 스토어 및 데이터베이스에서 데이터를 이동할 수 있습니다.

Amazon EC2에서 실행되는 Amazon EMR에서 Secrets Manager를 사용하는 방법

Amazon EMR에 클러스터를 만들 때 Secrets Manager의 보안 암호를 사용하여 클러스터에 응용 프로그램 구성 데이터를 제공할 수 있습니다. 자세한 내용은 Amazon EMR 관리 안내서의 Secrets Manager에 중요 데이터 보관을 참조하세요.

또한 Secrets Manager를 사용하여 EMR Notebook을 생성하고 프라이빗 Git 기반 레지스트리 보안 인증 정보를 저장할 수 있습니다. 자세한 내용은 Amazon EMR 관리 가이드의 Amazon EMR에 Git 기반 리포지토리 추가를 참조하세요.

EMR Serverless에서 Secrets Manager를 사용하는 방법

EMR Serverless는 분석 애플리케이션 운영을 간소화하는 서버리스 런타임 환경을 제공하므로 클러스터를 구성하거나 최적화, 보호, 운영할 필요가 없습니다.

EMR 서버리스 구성에 데이터를 저장한 다음 보안 ID를 EMR 서버리스 구성에서 사용할 수 있습니다. AWS Secrets Manager 이렇게 하면 민감한 구성 데이터를 일반 텍스트로 전달하여 외부 API에 노출하지 않아도 됩니다.

자세한 내용은 Amazon EMR Serverless 사용 설명서에서 EMR Serverless를 이용한 데이터 보호를 위한 Secrets Manager를 참조하세요.

아마존이 EventBridge 사용하는 방법 AWS Secrets Manager

EventBridge Amazon은 애플리케이션을 다양한 소스의 데이터와 연결하는 데 사용할 수 있는 서버리스 이벤트 버스 서비스입니다.

Amazon EventBridge API 대상을 생성할 때 접두사와 함께 `events` Secrets Manager의 관리 비밀에 해당 연결을 EventBridge 저장합니다. 보안 암호 저장 비용은 API 대상 사용 요금에 포함됩니다. 암호를 업데이트하려면 Secrets Manager EventBridge 대신 를 사용해야 합니다. 자세한 내용은 Amazon EventBridge 사용 설명서의 API 대상을 참조하십시오.

Amazon AWS Secrets Manager FSx가 비밀을 사용하는 방법

Amazon FSx for Windows File Server는 완전한 네이티브 Windows 파일 시스템이 지원하는 완전관리형 Microsoft Windows 파일 서버를 제공합니다. 파일 공유를 생성하거나 관리할 때 비밀의 자격 증명을

전달할 수 있습니다. AWS Secrets Manager 자세한 내용은 Amazon FSx for Windows File Server 사용 설명서의 파일 공유 및 Amazon FSx로 파일 공유 구성 마이그레이션을 참조하세요.

AWS Glue DataBrew 사용 방법 AWS Secrets Manager

AWS Glue DataBrew 코드를 작성하지 않고도 데이터를 정리하고 정규화하는 데 사용할 수 있는 시각적 데이터 준비 도구입니다. DataBrew에서는 일련의 데이터 변환 단계를 레시피라고 합니다. AWS Glue DataBrew Secrets Manager 시크릿에 저장된 암호화 키를 사용하여 데이터세트의 개인 식별 정보 (PII) 를 변환하기 위한 `DETERMINISTIC_DECRYPTDETERMINISTIC_ENCRYPT`, 및 `CRYPTOGRAPHIC_HASH`레시피 단계를 제공합니다. DataBrew 기본 비밀번호를 사용하여 암호화 키를 저장하는 경우, 는 접두사를 포함하는 관리 DataBrew 비밀번호를 생성합니다. `databrew` 암호 저장 비용은 사용 DataBrew 요금에 포함되어 있습니다. 암호화 키를 저장할 새 암호를 생성하는 경우 접두사를 `AwsGlueDataBrew` 사용하여 암호가 DataBrew 생성됩니다. 보안 암호에 대한 요금이 청구됩니다.

AWS Glue Studio에서 사용하는 방법 AWS Secrets Manager

AWS Glue Studio ETL (추출, 변환, 로드) 작업을 쉽게 만들고 실행하고 모니터링할 수 있는 그래픽 인터페이스입니다. AWS Glue에서 Elasticsearch Spark 커넥터를 구성하여 Amazon OpenSearch 서비스를 ETL (추출, 변환, 로드) 작업을 위한 데이터 스토어로 사용할 수 있습니다. AWS Glue Studio OpenSearch 클러스터에 연결하려면 Secrets Manager에서 시크릿을 사용할 수 있습니다. 자세한 내용은 AWS Glue 개발자 안내서의 자습서: Elasticsearch용 AWS 글루 커넥터 사용을 참조하십시오.

사용 방법 AWS IoT SiteWiseAWS Secrets Manager

AWS IoT SiteWise 산업 장비의 데이터를 대규모로 수집, 모델링, 분석 및 시각화할 수 있는 관리형 서비스입니다. AWS IoT SiteWise 콘솔을 사용하여 게이트웨이를 만들 수 있습니다. 그런 다음 게이트웨이에 연결된 데이터 원본, 로컬 서버 또는 산업 장비를 추가합니다. 소스에 인증이 필요할 경우 보안 암호를 사용하여 인증합니다. 자세한 내용은 AWS IoT SiteWise 사용 설명서의 데이터 소스 인증 구성을 참조하세요.

Amazon Kendra가 사용하는 방법 AWS Secrets Manager

Amazon Kendra는 사용자가 자연어 처리 및 고급 검색 알고리즘을 사용하여 구조화되지 않은 데이터와 구조화된 데이터를 검색할 수 있도록 해주는 매우 정확하고 지능적인 검색 서비스입니다.

해당 데이터베이스의 자격 증명을 포함한 보안 암호를 지정하여 데이터베이스에 저장된 문서를 인덱싱할 수 있습니다. 자세한 내용은 Amazon Kendra 사용 설명서의 데이터베이스 데이터 소스 사용을 참조하세요.

Amazon Kinesis Video Streams가 사용하는 방법 AWS Secrets Manager

Amazon Kinesis Video Streams를 사용하여 고객의 온프레미스 IP 카메라에 연결하고, 카메라의 비디오를 로컬로 녹화 및 저장하며, 장기간 저장, 재생 및 분석 처리를 위해 비디오를 클라우드로 스트리밍할 수 있습니다. IP 카메라에서 미디어를 녹화하고 업로드하려면 Kinesis Video Streams Edge Agent를 AWS IoT Greengrass에 배포합니다. 카메라로 스트리밍되는 미디어 파일에 액세스하는 데 필요한 보안 인증 정보를 Secrets Manager 보안 암호에 저장합니다. 자세한 내용은 Amazon Kinesis Video Streams 개발자 안내서에서 AWS IoT Greengrass에 Amazon Kinesis Video Streams Edge Agent 배포를 참조하세요.

사용 방법 AWS Launch WizardAWS Secrets Manager

AWS Launch Wizard for Active Directory는 AWS 클라우드 애플리케이션 모범 사례를 적용하여 새 Active Directory 인프라를 설정하거나 기존 인프라 (또는 온프레미스) 에 도메인 컨트롤러를 추가하는 과정을 안내하는 서비스입니다. AWS 클라우드

AWS Launch Wizard 도메인 컨트롤러를 Active Directory에 가입시키려면 Secrets Manager에 도메인 관리자 자격 증명을 추가해야 합니다. 자세한 내용은 AWS Launch Wizard 사용 설명서의 Active AWS Launch Wizard Directory용 설정을 참조하십시오.

Amazon Lookout for Metrics의 AWS Secrets Manager활용 방식

Amazon Lookout for Metrics는 데이터에서 이상을 찾고 근본 원인을 파악하여 신속하게 조치를 취하는 서비스입니다. Amazon Redshift 또는 Amazon RDS를 Lookout for Metrics 감지기의 데이터 원본으로 사용할 수 있습니다. 데이터 원본을 구성하는 데 데이터베이스 암호가 포함된 보안 암호를 사용할 수 있습니다. 자세한 내용은 Amazon Lookout for Metrics 개발자 가이드의 Lookout for Metrics와 함께 Amazon RDS 사용 및 Lookout for Metrics와 함께 Amazon Redshift 사용을 참조하세요.

아마존 매니지드 Grafana가 사용하는 방법 AWS Secrets Manager

Amazon Managed Grafana는 여러 소스의 운영 지표, 로그 및 추적을 즉시 쿼리, 상관 관계 파악 및 시각화하는 데 사용할 수 있는 완전 관리형의 안전한 데이터 시각화 서비스입니다. Amazon Redshift를

데이터 소스로 사용하는 경우 시크릿을 사용하여 Amazon Redshift 자격 증명을 제공할 수 있습니다. AWS Secrets Manager 자세한 내용은 Amazon Managed Grafana 사용자 안내서의 Amazon Redshift 구성 섹션을 참조하십시오.

사용 방법 AWS Managed ServicesAWS Secrets Manager

AWS Managed Services AWS 인프라의 지속적인 관리를 제공하는 엔터프라이즈 서비스입니다. AMS 셀프 서비스 프로비저닝 (SSP) 모드는 AMS 관리 계정의 네이티브 AWS 서비스 및 API 기능에 대한 전체 액세스를 제공합니다. AMS에서 Secrets Manager에 대한 액세스를 요청하는 방법에 대한 자세한 내용은 AMS Advanced User Guide에서 AWS Secrets Manager (AMS Self-Service Provisioning)을 참조하세요.

Amazon Managed Streaming for Apache Kafka의 AWS Secrets Manager활용 방식

Amazon Managed Streaming for Apache Kafka(Amazon MSK)는 Apache Kafka를 사용하여 스트리밍 데이터를 처리하는 애플리케이션의 구축 및 실행을 위해 사용할 수 있는 완전관리형 서비스입니다. AWS Secrets Manager을 사용하여 저장되고 보호되는 사용자 이름 및 암호를 사용하여 Amazon MSK 클러스터에 대한 액세스를 제어할 수 있습니다. 자세한 내용은 Amazon Managed Streaming for Apache Kafka 개발자 안내서의 AWS Secrets Manager를 사용한 사용자 이름 및 암호 인증을 참조하세요.

Apache 에어플로우용 Amazon 관리형 워크플로를 사용하는 방법 AWS Secrets Manager

Apache Airflow용 Amazon Managed Workflow는 Apache Airflow용 관리형 오케스트레이션 서비스로, 이를 통해 클라우드에서 대규모 end-to-end 데이터 파이프라인을 쉽게 설정하고 운영할 수 있습니다.

Secrets Manager 보안 암호를 사용하여 Apache Airflow 연결을 구성할 수 있습니다. 자세한 내용은 Apache Airflow용 Amazon 관리형 워크플로 사용 설명서의 Secrets Manager 암호를 사용한 Apache Airflow 연결 구성 및 Apache Airflow 변수에 대한 비밀 키 사용을 참조하십시오. AWS Secrets Manager

AWS Marketplace

AWS Marketplace Quick Launch를 사용하면 라이선스 키와 함께 소프트웨어를 AWS Marketplace 배포합니다. AWS Marketplace 사용자 계정에 라이센스 키를 Secrets Manager의 관리 암호로 저장합

니다. 보안 정보 저장 비용은 요금에 포함되어 AWS Marketplace있습니다. 암호를 업데이트하려면 Secrets Manager AWS Marketplace 대신 를 사용해야 합니다. 자세한 내용은 AWS Marketplace 판매자 설명서의 빠른 실행 구성을 참조하세요.

AWS Migration Hub 사용 방법 AWS Secrets Manager

AWS Migration Hub 여러 AWS 도구 및 파트너 솔루션에서 마이그레이션 작업을 추적할 수 있는 단일 위치를 제공합니다.

AWS Migration Hub Orchestrator는 서버 및 엔터프라이즈 애플리케이션을 마이그레이션하는 작업을 단순화하고 자동화합니다. AWS Migration Hub Orchestrator 는 소스 서버에 대한 연결 정보에 보안 암호를 사용합니다. 자세한 내용은 AWS Migration Hub 오케스트레이터 사용 설명서에서 사용 방법을 참조하세요:

- SAP 애플리케이션을 다음으로 마이그레이션하십시오. NetWeaver AWS
- Amazon EC2에서 애플리케이션 리호스팅

Migration Hub 전략 권장 사항은 애플리케이션 변환을 실행할 수 있는 경로를 제공할 마이그레이션 및 현대화 전략 권장 사항을 제공합니다. 전략 권장 사항은 연결 정보에 대한 보안 암호를 사용하여 SQL Server 데이터베이스를 분석할 수 있습니다. 자세한 내용은 전략 권장 사항 데이터베이스 분석을 참조하세요.

AWS Panorama Secrets Manager를 사용하는 방법

AWS Panorama 온-프레미스 카메라 네트워크에 컴퓨터 비전을 제공하는 서비스입니다. AWS Panorama 어플라이언스를 등록하고, 해당 소프트웨어를 업데이트하고, 애플리케이션에 애플리케이션을 배포하는 데 사용합니다. 비디오 스트림을 애플리케이션의 데이터 소스로 등록할 때 스트림이 암호로 보호되는 경우 해당 스트림에 대한 자격 증명을 Secrets Manager 시크릿에 AWS Panorama 저장합니다. 자세한 내용은 AWS Panorama 개발자 안내서의 AWS Panorama에서 카메라 스트림 관리를 참조하세요.

AWS ParallelCluster 사용 방법 AWS Secrets Manager

AWS ParallelCluster HPC (고성능 컴퓨팅) 클러스터를 배포하고 관리하는 데 사용할 수 있는 오픈 소스 클러스터 관리 도구입니다. AWS 클라우드 AWS 관리형 Microsoft AD (Active Directory) 와 통합된 환경을 포함하는 다중 사용자 환경을 만들 수 있습니다. AWS ParallelCluster 는 Secrets Manager 암호

를 AWS ParallelCluster 사용하여 Active Directory에 대한 로그인을 검증합니다. 자세한 내용은 AWS ParallelCluster 사용 설명서에서 Active Directory 통합을 참조하세요.

Amazon Q가 Secrets Manager를 사용하는 방법

Amazon Q가 데이터 소스에 액세스할 수 있도록 인증하려면 Secrets Manager 암호를 사용하여 Amazon Q에 데이터 소스 액세스 자격 증명을 제공해야 합니다. 콘솔을 사용하는 경우 새 암호를 생성하거나 기존 암호를 사용할 수 있습니다. 자세한 내용은 Amazon Q 개발자 안내서의 개념 - 인증을 참조하십시오.

AWS OpsWorks for Chef Automate 사용 방법 AWS Secrets Manager

AWS OpsWorks Puppet Enterprise 또는 AWS OpsWorks for Chef Automate를 사용하여 OpsWorks 클라우드 엔터프라이즈에서 애플리케이션을 구성하고 운영할 수 있도록 지원하는 구성 관리 서비스입니다.

에서 AWS OpsWorks CM새 서버를 만들면 OpsWorks CM은 접두사와 함께 `opsworks-cm` Secrets Manager 관리 암호에 서버 정보를 저장합니다. 시크릿 비용은 요금에 AWS OpsWorks포함되어 있습니다. 자세한 내용은 AWS OpsWorks 사용 설명서의 AWS Secrets Manager과 통합을 참조하세요.

아마존이 QuickSight 사용하는 방법 AWS Secrets Manager

QuickSight Amazon은 분석, 데이터 시각화 및 보고에 사용할 수 있는 클라우드 규모의 비즈니스 인텔리전스 (BI) 서비스입니다. Amazon에서는 다양한 데이터 소스를 사용할 수 QuickSight 있습니다. Secrets Manager 암호에 데이터베이스 자격 증명을 저장하는 경우 Amazon은 이러한 암호를 사용하여 데이터베이스에 연결할 QuickSight 수 있습니다. 자세한 내용은 Amazon 사용 QuickSight 설명서의 QuickSight Amazon에서 데이터베이스 자격 증명 대신 AWS Secrets Manager 암호 사용을 참조하십시오.

Amazon RDS

Amazon Relational Database Service(Amazon RDS)는 AWS 클라우드에서 관계형 데이터베이스를 더 쉽게 설치, 운영 및 확장할 수 있는 웹 서비스입니다.

Aurora를 포함한 Amazon RDatabase Service (Amazon RDS) 의 마스터 사용자 자격 증명을 관리하기 위해 Amazon RDS에서 관리 암호를 생성할 수 있습니다. 보안 암호에 대한 요금이 청구됩니다. 또한

Amazon RDS는 이러한 보안 인증 정보의 교체를 관리합니다. 자세한 내용은 Amazon RDS 사용 설명서의 Amazon RDS 및 AWS Secrets Manager을 사용한 암호 관리를 참조하세요.

다른 Amazon RDS 보안 인증에 대한 자세한 내용은 the section called “데이터베이스 보안 암호 생성”을 참조하세요.

Amazon RDS 쿼리 편집기를 사용하여 데이터베이스에 연결하면 데이터 베이스에 대한 보안 인증 정보를 Secrets Manager에 저장할 수 있습니다. 자세한 내용은 Amazon RDS 사용 설명서의 쿼리 편집기 사용을 참조하세요.

아마존 Redshift의 사용 방법 AWS Secrets Manager

Amazon Redshift는 클라우드에서 완전히 관리되는 페타바이트급 데이터 웨어하우스 서비스입니다.

Amazon Redshift의 관리자 자격 증명을 관리하기 위해 Amazon Redshift는 사용자를 대신하여 관리 암호를 생성할 수 있습니다. 보안 암호에 대한 요금이 청구됩니다. Amazon Redshift는 또한 이러한 자격 증명의 교체를 관리합니다. 자세한 내용은 Amazon Redshift 관리 안내서의 AWS Secrets Manager을(를) 사용하는 Amazon Redshift 관리자 암호 관리를 참조하세요.

다른 Amazon Redshift 보안 인증의 경우, the section called “데이터베이스 보안 암호 생성”을(를) 참조하세요.

Amazon Redshift 데이터 API를 호출할 때 Secrets Manager의 보안 암호를 사용하여 클러스터에 대한 자격 증명을 전달할 수 있습니다. 자세한 내용은 Amazon Redshift 데이터 API 사용을 참조하세요.

Amazon RDS 쿼리 편집기를 사용하여 데이터베이스에 연결하면 Amazon Redshift는 사용자 보안 인증 정보를 `redshiftqueryeditor` 접두사가 붙은 Secrets Manager 보안 암호에 저장할 수 있습니다. 보안 암호에 대한 요금이 청구됩니다. 자세한 정보는 Amazon Redshift 관리 안내서의 쿼리 편집기를 사용하여 데이터베이스 쿼리를 참조하세요.

쿼리 편집기 v2에 대해서는 the section called “Amazon Redshift 쿼리 편집기 v2” 섹션을 참조하세요.

Amazon Redshift 쿼리 편집기 v2

Amazon Redshift 쿼리 편집기 v2는 Amazon Redshift 데이터 웨어하우스에서 쿼리를 작성하고 실행하는 데 사용하는 웹 기반 SQL 클라이언트 애플리케이션입니다. Amazon Redshift 쿼리 편집기 v2를 사용하여 데이터베이스에 연결하는 경우 Amazon Redshift는 접두사가 붙은 Secrets Manager의 관리 암호에 자격 증명을 저장할 수 있습니다. `sqlworkbench` 보안 암호 저장 비용은 Amazon Redshift 사용 요금에 포함됩니다. 보안 암호를 업데이트하려면 Secrets Manager 대신 Amazon Redshift를 사용해야 합니다. 자세한 정보는 Amazon Redshift 관리 안내서의 쿼리 편집기 v2 작업을 참조하세요.

이전 쿼리 편집기에 관해서는 the section called “Amazon Redshift” 섹션을 참조하세요.

아마존이 SageMaker 사용하는 방법 AWS Secrets Manager

SageMaker 완전 관리형 기계 학습 서비스입니다. 를 SageMaker 통해 데이터 과학자와 개발자는 기계 학습 모델을 쉽고 빠르게 구축 및 교육한 다음 프로덕션 준비가 완료된 호스팅 환경에 직접 배포할 수 있습니다. 탐색 및 분석에 필요한 데이터 원본에 대한 쉬운 액세스를 위해 내장형 Jupyter 작성 노트북 인스턴스를 제공하기 때문에 서버를 관리할 필요가 없습니다.

Git 리포지토리를 Jupyter Notebook 인스턴스와 연결하여 노트북 인스턴스를 중지 또는 삭제하더라도 유지되는 소스 제어 환경에 노트북을 저장할 수 있습니다. Secrets Manager를 사용하여 프라이빗 리포지토리 자격 증명을 관리할 수 있습니다. 자세한 내용은 Amazon 개발자 안내서의 Git 리포지토리와 SageMaker Amazon 노트북 인스턴스 연결을 참조하십시오. SageMaker

Databricks에서 데이터를 가져오기 위해 Data Wrangler는 Secrets Manager에 JDBC URL을 저장합니다. 자세한 내용은 Databricks 에서 데이터 가져오기(JDBC)를 참조하세요.

Snowflake에서 데이터를 가져오기 위해 Data Wrangler는 Secrets Manager 보안 암호에 자격 증명을 저장합니다. 자세한 내용은 Snowflake에서 데이터 가져오기를 참조하세요.

사용 방법 AWS Schema Conversion ToolAWS Secrets Manager

AWS Schema Conversion Tool (AWS SCT) 를 사용하여 기존 데이터베이스 스키마를 한 데이터베이스 엔진에서 다른 데이터베이스 엔진으로 변환할 수 있습니다. 관계형 OLTP 스키마 또는 데이터 웨어하우스 스키마를 변환할 수 있습니다. 변환된 스키마는 Amazon Relational Database Service(RDS) MySQL, MariaDB, Oracle, SQL Server, PostgreSQL DB, Amazon Aurora DB 클러스터 또는 Amazon Redshift 클러스터에 적합합니다. 변환된 스키마는 Amazon Elastic Compute Cloud 인스턴스의 데이터베이스에서 사용하거나 S3 버킷에 데이터로 저장할 수도 있습니다.

데이터베이스 스키마를 변환할 때 저장한 데이터베이스 자격 증명을 사용할 AWS SCT 수 AWS Secrets Manager있습니다. 자세한 내용은 사용 설명서의 AWS SCT 사용자 AWS Secrets Manager 인터페이스에서 사용을 참조하십시오.AWS Schema Conversion Tool

AWS Toolkit for JetBrains 사용 방법 AWS Secrets Manager

의 통합 개발 환경 (IDE) 을 위한 오픈 소스 AWS Toolkit for JetBrains 플러그인입니다. JetBrains 개발자가 도구 키트를 사용하여 AWS를 사용하는 서버리스 애플리케이션을 쉽게 개발, 디버깅, 배포할 수 있습니다. 도구 키트를 사용하여 Amazon Redshift 클러스터에 연결할 때 Secrets Manager 보안 암

호를 사용하여 인증할 수 있습니다. 자세한 내용은 AWS Toolkit for JetBrains 사용 설명서의 Amazon Redshift 클러스터에 대한 액세스를 참조하세요.

시크릿을 AWS Transfer Family 사용하는 AWS Secrets Manager 방법

AWS Transfer Family 스토리지 서비스 간에 파일을 전송하고 AWS 스토리지 서비스에서 파일을 전송할 수 있는 보안 전송 서비스입니다.

Transfer Family는 이제 Applicability Statement 2(AS2) 프로토콜을 사용하는 서버에 대해 기본 인증 사용을 지원합니다. 새 Secrets Manager 보안 암호를 생성하거나 보안 인증 정보에 대한 기존 보안 암호를 선택할 수 있습니다. 자세한 내용은 AWS Transfer Family 사용 설명서의 AS2 커넥터에 대한 기본 인증을 참조하세요.

Transfer Family 사용자를 인증하려면 ID AWS Secrets Manager 제공자로 사용할 수 있습니다. 자세한 내용은 사용 설명서의AWS Transfer Family 사용자 지정 ID 공급자 사용 및 블로그 문서 암호 인증 AWS Transfer Family 사용을 AWS Secrets Manager참조하십시오.

Transfer Family가 워크플로를 통해 처리하는 파일에 프리티 굿 프라이버시(PGP) 암호 해독을 사용할 수 있습니다. 워크플로 단계에서 암호 해독을 사용하려면 Secrets Manager에서 관리하는 PGP 키를 제공해야 합니다. 자세한 내용은AWS Transfer Family 사용 설명서의 Generate and manage PGP keys(PGP 키 생성 및 관리)를 참조하세요.

AWS Wickr가 비밀을 사용하는 방법 AWS Secrets Manager

AWS Wickr 조직 및 정부 기관이 그룹 메시징, 음성 및 화상 통화, 파일 공유, 화면 공유 등을 통해 one-to-one 안전하게 통신할 수 있도록 지원하는 end-to-end 암호화된 서비스입니다. Wickr 데이터 보존 봇을 사용하여 워크플로를 자동화할 수 있습니다. 봇이 액세스할 수 있게 AWS 서비스되면 Secrets Manager 암호를 만들어 봇 자격 증명을 저장해야 합니다. 자세한 내용은AWS Wickr 관리 가이드에서 데이터 보존 봇 시작을 참조하십시오.

AWS Secrets Manager VPC 엔드포인트 사용

가능한 경우 대부분의 인프라를 퍼블릭 인터넷에서 액세스할 수 없는 프라이빗 네트워크에서 실행하는 것이 좋습니다. 인터페이스 VPC 엔드포인트를 생성하여 VPC와 Secrets Manager 간에 프라이빗 연결을 설정할 수 있습니다. 인터페이스 엔드포인트는 인터넷 게이트웨이, NAT 디바이스, VPN 연결 또는 AWS Direct Connect 연결 없이 비공개로 Secrets Manager API에 액세스할 수 있도록 지원하는 AWS PrivateLink 기술로 구동됩니다. VPC의 인스턴스는 Secrets Manager API와 통신하는 데 퍼블릭 IP 주소를 필요로 하지 않습니다. VPC와 Secrets Manager 간의 트래픽은 AWS 네트워크를 벗어나지 않습니다. 자세한 내용은 Amazon VPC 사용 설명서의 인터페이스 VPC 엔드포인트(AWS PrivateLink)를 참조하세요.

Secret Manager가 Lambda 교체 함수를 사용하여 암호를 교체할 때(예: 데이터베이스 자격 증명이 포함된 암호) Lambda 함수는 데이터베이스와 Secret Manager 모두에게 요청을 합니다. 콘솔을 사용하여 자동 교체를 설정하면 Secrets Manager가 데이터베이스와 동일한 VPC에 Lambda 함수를 생성합니다. Lambda 교체 함수에서 Secrets Manager로의 요청이 Amazon 네트워크를 벗어나지 않도록 동일한 VPC의 Secrets Manager 엔드포인트를 생성하는 것이 좋습니다.

엔드포인트에 프라이빗 DNS를 사용하도록 설정하는 경우, 리전에 대한 기본 DNS 이름(예: `secretsmanager.us-east-1.amazonaws.com`)을 사용하여 Secrets Manager에 API 요청을 할 수 있습니다. 자세한 내용은 Amazon VPC 사용 설명서의 인터페이스 엔드포인트를 통해 서비스 액세스 섹션을 참조하세요.

권한 정책에 조건을 포함시켜 Secrets Manager에 대한 요청이 VPC 액세스에서 나오도록 할 수 있습니다. 자세한 설명은 the section called “예: 권한 및 VPC” 섹션을 참조하세요.

AWS CloudTrail 로그를 사용하여 VPC 엔드포인트를 통해 보안 암호 사용을 감사할 수 있습니다.

Secrets Manager에 대한 VPC 엔드포인트를 생성하려면

1. Amazon VPC 사용 설명서의 인터페이스 엔드포인트 생성을 참조하십시오. 서비스 이름으로 com.amazonaws.*region*.secretsmanager를 사용합니다.
2. 엔드포인트에 대한 액세스를 제어하려면 엔드포인트 정책을 사용하여 VPC 엔드포인트에 대한 액세스 제어를 참조하십시오.

공유 서브넷

공유하는 서브넷의 VPC 엔드포인트는 생성, 설명, 수정 또는 삭제할 수 없습니다. 그러나 공유하는 서브넷의 VPC 엔드포인트를 사용할 수는 있습니다. VPC 공유에 대한 자세한 내용은 Amazon Virtual Private Cloud 사용 설명서에서 다른 계정과 VPC 공유를 참조하세요.

AWS CloudFormation에서 AWS Secrets Manager 보안 암호 생성

보안 암호 생성에 나와 있는 것처럼 CloudFormation 템플릿의 `AWS::SecretsManager::Secret` 리소스를 사용하여 CloudFormation 스택에 보안 암호를 생성할 수 있습니다.

Amazon RDS 또는 Aurora에 대한 관리자 암호를 생성하려면 `AWS::RDS::DBCluster`에서 `ManageMasterUserPassword`를 사용하는 것이 좋습니다 . 그러면 Amazon RDS가 암호를 생성하고 자동으로 교체를 관리합니다. 자세한 내용은 관리형 교체 섹션을 참조하세요.

Amazon Redshift 및 Amazon DocumentDB 자격 증명에 대해서는 먼저 Secrets Manager에서 생성한 암호로 보안 암호를 생성한 다음 동적 참조를 사용하여 새 데이터베이스의 자격 증명으로 사용할 보안 암호에서 사용자 이름과 암호를 검색합니다. 다음으로 `AWS::SecretsManager::SecretTargetAttachment` 리소스를 사용하여 Secrets Manager가 보안 암호를 교체하는 데 필요한 보안 암호에 데이터베이스에 관한 세부 정보를 추가합니다. 마지막으로 자동 교체를 켜려면 `AWS::SecretsManager::RotationSchedule` 리소스를 사용하고 교체 함수와 일정을 제공합니다. 다음 예를 참조하세요.

- Amazon Redshift 자격 증명을 사용하여 보안 암호 생성
- Amazon DocumentDB 자격 증명을 사용하여 보안 암호 생성

보안 암호에 리소스 정책을 연결하려면 `AWS::SecretsManager::ResourcePolicy` 리소스를 사용합니다.

AWS CloudFormation을 사용한 리소스 생성에 대한 자세한 내용은 AWS CloudFormation 사용 설명서의 템플릿 기본 사항 알아보기를 참조하세요. AWS Cloud Development Kit (AWS CDK)도 사용할 수 있습니다. 자세한 내용은 AWS Secrets Manager 라이브러리 구성을 참조하세요.

AWS CloudFormation으로 AWS Secrets Manager 보안 암호 생성

이 예제에서는 **`CloudFormationCreatedSecret-`*`a1b2c3d4e5f6`*** 이라는 이름의 보안 암호를 생성합니다. 보안 암호 값은 다음 JSON으로, 보안 암호를 생성할 때 생성된 32자 암호를 포함합니다.

```
{
    "password": "EXAMPLE-PASSWORD",
```

```
    "username": "saanvi"
}
```

이 예제에서는 다음 CloudFormation 리소스를 사용합니다.

- AWS::SecretsManager::Secret

AWS CloudFormation을 사용한 리소스 생성에 대한 자세한 내용은 AWS CloudFormation 사용 설명서의 템플릿 기본 사항 알아보기를 참조하세요.

JSON

```
{
    "Resources": {
        "CloudFormationCreatedSecret": {
            "Type": "AWS::SecretsManager::Secret",
            "Properties": {
                "Description": "Simple secret created by AWS CloudFormation.",
                "GenerateSecretString": {
                    "SecretStringTemplate": "{\"username\": \"saanvi\"}",
                    "GenerateStringKey": "password",
                    "PasswordLength": 32
                }
            }
        }
    }
}
```

YAML

```
Resources:
  CloudFormationCreatedSecret:
    Type: 'AWS::SecretsManager::Secret'
    Properties:
      Description: Simple secret created by AWS CloudFormation.
      GenerateSecretString:
        SecretStringTemplate: '{"username": "saanvi"}'
        GenerateStringKey: password
        PasswordLength: 32
```

AWS CloudFormation을 사용하여 자동 교체되는 AWS Secrets Manager 보안 암호 및 Amazon RDS MySQL DB 인스턴스 생성

Amazon RDS 또는 Aurora에 대한 관리자 암호를 만들려면 `AWS::RDS::DBCluster`의 마스터 암호에 대한 Secrets Manager 암호 생성 예에 나와 있는 것처럼 `ManageMasterUserPassword`를 사용하는 것이 좋습니다. 그러면 Amazon RDS가 암호를 생성하고 자동으로 교체를 관리합니다. 자세한 내용은 관리형 교체 섹션을 참조하세요.

다음을 사용하여 AWS Secrets Manager 시크릿과 Amazon Redshift 클러스터를 생성합니다. AWS CloudFormation

Amazon Redshift용 관리자 암호를 생성하려면 및 의 예제를 사용하는 것이 좋습니다. `AWS::Redshift::ClusterAWS::RedshiftServerless::Namespace`

다음을 사용하여 AWS Secrets Manager 시크릿과 Amazon DocumentDB 인스턴스를 생성합니다. AWS CloudFormation

이 예제에서는 보안 암호를 생성하고, 보안 암호의 자격 증명을 사용자와 암호로 사용하는 Amazon DocumentDB 인스턴스를 생성합니다. 보안 암호에는 보안 암호에 액세스할 수 있는 사용자를 정의하는 리소스 기반 정책이 연결되어 있습니다. 또한 템플릿은 교체 함수 템플릿에서 Lambda 교체 함수를 생성하고 매월 첫째 날 오전 8시에서 오전 10시 사이에 보안 암호를 자동 교체하도록 구성합니다. 보안 모범 사례로서 인스턴스는 Amazon VPC에 있습니다.

이 예제에서는 Secrets Manager에 다음과 같은 CloudFormation 리소스를 사용합니다.

- `AWS::SecretsManager::Secret`
- `AWS::SecretsManager::SecretTargetAttachment`
- `AWS::SecretsManager::RotationSchedule`

를 사용하여 AWS CloudFormation리소스를 생성하는 방법에 대한 자세한 내용은 AWS CloudFormation 사용 설명서의 Learn 템플릿 기본 사항을 참조하십시오.

JSON

```
{
   "AWSTemplateFormatVersion":"2010-09-09",
```

```
    "Transform":"AWS::SecretsManager-2020-07-23",
    "Resources":{
       "TestVPC":{
          "Type":"AWS::EC2::VPC",
          "Properties":{
             "CidrBlock":"10.0.0.0/16",
             "EnableDnsHostnames":true,
             "EnableDnsSupport":true
          }
       },
       "TestSubnet01":{
          "Type":"AWS::EC2::Subnet",
          "Properties":{
             "CidrBlock":"10.0.96.0/19",
             "AvailabilityZone":{
                "Fn::Select":[
                   "0",
                   {
                      "Fn::GetAZs":{
                         "Ref":"AWS::Region"
                      }
                   }
                ]
             },
             "VpcId":{
                "Ref":"TestVPC"
             }
          }
       },
       "TestSubnet02":{
          "Type":"AWS::EC2::Subnet",
          "Properties":{
             "CidrBlock":"10.0.128.0/19",
             "AvailabilityZone":{
                "Fn::Select":[
                   "1",
                   {
                      "Fn::GetAZs":{
                         "Ref":"AWS::Region"
                      }
                   }
                ]
             },
             "VpcId":{
```

```
            "Ref":"TestVPC"
         }
      }
   },
   "SecretsManagerVPCEndpoint":{
      "Type":"AWS::EC2::VPCEndpoint",
      "Properties":{
         "SubnetIds":[
            {
               "Ref":"TestSubnet01"
            },
            {
               "Ref":"TestSubnet02"
            }
         ],
         "SecurityGroupIds":[
            {
               "Fn::GetAtt":[
                  "TestVPC",
                  "DefaultSecurityGroup"
               ]
            }
         ],
         "VpcEndpointType":"Interface",
         "ServiceName":{
            "Fn::Sub":"com.amazonaws.${AWS::Region}.secretsmanager"
         },
         "PrivateDnsEnabled":true,
         "VpcId":{
            "Ref":"TestVPC"
         }
      }
   },
   "MyDocDBClusterRotationSecret":{
      "Type":"AWS::SecretsManager::Secret",
      "Properties":{
         "GenerateSecretString":{
            "SecretStringTemplate":"{\"username\": \"someadmin\",\"ssl\": true}",
            "GenerateStringKey":"password",
            "PasswordLength":16,
            "ExcludeCharacters":"\"@/\\"
         },
         "Tags":[
            {
```

```
                  "Key":"AppName",
                  "Value":"MyApp"
               }
            ]
         }
      },
      "MyDocDBCluster":{
         "Type":"AWS::DocDB::DBCluster",
         "Properties":{
            "DBSubnetGroupName":{
               "Ref":"MyDBSubnetGroup"
            },
            "MasterUsername":{
               "Fn::Sub":"{{resolve:secretsmanager:
${MyDocDBClusterRotationSecret}::username}}"
            },
            "MasterUserPassword":{
               "Fn::Sub":"{{resolve:secretsmanager:
${MyDocDBClusterRotationSecret}::password}}"
            },
            "VpcSecurityGroupIds":[
               {
                  "Fn::GetAtt":[
                     "TestVPC",
                     "DefaultSecurityGroup"
                  ]
               }
            ]
         }
      },
      "DocDBInstance":{
         "Type":"AWS::DocDB::DBInstance",
         "Properties":{
            "DBClusterIdentifier":{
               "Ref":"MyDocDBCluster"
            },
            "DBInstanceClass":"db.r5.large"
         }
      },
      "MyDBSubnetGroup":{
         "Type":"AWS::DocDB::DBSubnetGroup",
         "Properties":{
            "DBSubnetGroupDescription":"",
            "SubnetIds":[
```

```
            {
               "Ref":"TestSubnet01"
            },
            {
               "Ref":"TestSubnet02"
            }
         ]
      }
   },
   "SecretDocDBClusterAttachment":{
      "Type":"AWS::SecretsManager::SecretTargetAttachment",
      "Properties":{
         "SecretId":{
            "Ref":"MyDocDBClusterRotationSecret"
         },
         "TargetId":{
            "Ref":"MyDocDBCluster"
         },
         "TargetType":"AWS::DocDB::DBCluster"
      }
   },
   "MySecretRotationSchedule":{
      "Type":"AWS::SecretsManager::RotationSchedule",
      "DependsOn":"SecretDocDBClusterAttachment",
      "Properties":{
         "SecretId":{
            "Ref":"MyDocDBClusterRotationSecret"
         },
         "HostedRotationLambda":{
            "RotationType":"MongoDBSingleUser",
            "RotationLambdaName":"MongoDBSingleUser",
            "VpcSecurityGroupIds":{
               "Fn::GetAtt":[
                  "TestVPC",
                  "DefaultSecurityGroup"
               ]
            },
            "VpcSubnetIds":{
               "Fn::Join":[
                  ",",
                  [
                     {
                        "Ref":"TestSubnet01"
                     },
```

```
                        {
                           "Ref":"TestSubnet02"
                        }
                     ]
                  ]
               }
            },
            "RotationRules":{
              "Duration": "2h",
              "ScheduleExpression": "cron(0 8 1 * ? *)"
            }
         }
      }
   }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::SecretsManager-2020-07-23
Resources:
  TestVPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: 10.0.0.0/16
      EnableDnsHostnames: true
      EnableDnsSupport: true
  TestSubnet01:
    Type: AWS::EC2::Subnet
    Properties:
      CidrBlock: 10.0.96.0/19
      AvailabilityZone: !Select
        - '0'
        - !GetAZs
          Ref: AWS::Region
      VpcId: !Ref TestVPC
  TestSubnet02:
    Type: AWS::EC2::Subnet
    Properties:
      CidrBlock: 10.0.128.0/19
      AvailabilityZone: !Select
        - '1'
        - !GetAZs
```

```
          Ref: AWS::Region
      VpcId: !Ref TestVPC
  SecretsManagerVPCEndpoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      SubnetIds:
        - !Ref TestSubnet01
        - !Ref TestSubnet02
      SecurityGroupIds:
        - !GetAtt TestVPC.DefaultSecurityGroup
      VpcEndpointType: Interface
      ServiceName: !Sub com.amazonaws.${AWS::Region}.secretsmanager
      PrivateDnsEnabled: true
      VpcId: !Ref TestVPC
  MyDocDBClusterRotationSecret:
    Type: AWS::SecretsManager::Secret
    Properties:
      GenerateSecretString:
        SecretStringTemplate: '{"username": "someadmin","ssl": true}'
        GenerateStringKey: password
        PasswordLength: 16
        ExcludeCharacters: '"@/\'
      Tags:
        - Key: AppName
          Value: MyApp
  MyDocDBCluster:
    Type: AWS::DocDB::DBCluster
    Properties:
      DBSubnetGroupName: !Ref MyDBSubnetGroup
      MasterUsername: !Sub '{{resolve:secretsmanager:
${MyDocDBClusterRotationSecret}::username}}'
      MasterUserPassword: !Sub '{{resolve:secretsmanager:
${MyDocDBClusterRotationSecret}::password}}'
      VpcSecurityGroupIds:
        - !GetAtt TestVPC.DefaultSecurityGroup
  DocDBInstance:
    Type: AWS::DocDB::DBInstance
    Properties:
      DBClusterIdentifier: !Ref MyDocDBCluster
      DBInstanceClass: db.r5.large
  MyDBSubnetGroup:
    Type: AWS::DocDB::DBSubnetGroup
    Properties:
      DBSubnetGroupDescription: ''
```

```
      SubnetIds:
        - !Ref TestSubnet01
        - !Ref TestSubnet02
  SecretDocDBClusterAttachment:
    Type: AWS::SecretsManager::SecretTargetAttachment
    Properties:
      SecretId: !Ref MyDocDBClusterRotationSecret
      TargetId: !Ref MyDocDBCluster
      TargetType: AWS::DocDB::DBCluster
  MySecretRotationSchedule:
    Type: AWS::SecretsManager::RotationSchedule
    DependsOn: SecretDocDBClusterAttachment
    Properties:
      SecretId: !Ref MyDocDBClusterRotationSecret
      HostedRotationLambda:
        RotationType: MongoDBSingleUser
        RotationLambdaName: MongoDBSingleUser
        VpcSecurityGroupIds: !GetAtt TestVPC.DefaultSecurityGroup
        VpcSubnetIds: !Join
          - ','
          - - !Ref TestSubnet01
            - !Ref TestSubnet02
      RotationRules:
        Duration: 2h
        ScheduleExpression: cron(0 8 1 * ? *)
```

Secrets Manager의 AWS CloudFormation 사용 방식

콘솔을 사용하여 교체를 활성화하면 Secrets Manager는 AWS CloudFormation을 사용하여 교체를 위한 리소스를 생성합니다. 그 과정에서 새 교체 함수를 생성하면 AWS CloudFormation은 적절한 교체 함수 템플릿을 기반으로 `AWS::Serverless::Function`을 생성합니다. 그런 다음 AWS CloudFormation은 보안 암호에 대한 교체 함수 및 교체 규칙을 설정하는 `RotationSchedule`을 설정합니다. 자동 교체를 활성화한 후 배너에서 스택 보기(View stack)를 선택하여 AWS CloudFormation 스택을 볼 수 있습니다.

자동 교체 활성화에 대한 정보는 보안 암호 교체를 참조하세요.

에서 AWS Secrets Manager 시크릿 생성 AWS Cloud Development Kit (AWS CDK)

CDK 앱에서 비밀을 생성, 관리 및 검색하려면 `ResourcePolicy`, `RotationSchedule`, `Secret`, `SecretRotation`, `SecretTargetAttachment` 구성이 들어 있는 AWS Secrets Manager 구성 라이브러리를 이용하면 됩니다.

CDK 애플리케이션에서 시크릿을 사용하는 좋은 방법은 먼저 콘솔이나 CLI를 사용하여 시크릿을 생성한 다음 CDK 애플리케이션으로 시크릿을 가져오는 것입니다.

예를 들어 참조할 섹션:

- 보안 암호 생성
- 보안 암호 가져오기
- 보안 암호 검색
- 암호 사용 권한 부여
- 보안 암호 교체
- 데이터베이스 보안 암호 교체
- 다른 리전으로 보안 암호 복제

CDK 에 대한 자세한 내용은 AWS Cloud Development Kit (AWS CDK) v2 개발자 안내서를 참조하세요.

모니터 AWS Secrets Manager 시크릿

AWS Secrets Manager 비밀을 감시하고, 문제 발생 시 보고하고, 적절한 경우 자동 조치를 취할 수 있는 모니터링 도구를 제공합니다. 예기치 않은 사용 또는 변경을 조사해야 하는 경우 로그를 사용하여 원치 않는 변경 사항을 취소할 수 있습니다. 또한 보안 암호의 부적절한 사용 및 보안 암호 삭제 시도에 대한 자동 검사를 설정할 수 있습니다.

주제

- 로 AWS Secrets Manager 이벤트 기록 AWS CloudTrail
- AWS Secrets Manager 아마존으로 모니터링하기 CloudWatch
- Amazon과 AWS Secrets Manager 이벤트 맞추기 EventBridge
- 삭제 예정인 AWS Secrets Manager 비밀에 대한 액세스 시기 모니터링
- 다음을 사용하여 규정 준수를 위한 AWS Secrets Manager 비밀 모니터링 AWS Config
- Secrets Manager 비용 모니터링

로 AWS Secrets Manager 이벤트 기록 AWS CloudTrail

AWS CloudTrail Secrets Manager에 대한 모든 API 호출을 이벤트로 기록합니다. 여기에는 Secrets Manager 콘솔에서의 호출과 순환 및 비밀 버전 삭제를 위한 기타 여러 이벤트가 포함됩니다. Secrets Manager 레코드의 로그 항목 목록은 을 참조하십시오CloudTrail 출품작.

CloudTrail 콘솔을 사용하여 지난 90일간의 기록된 이벤트를 볼 수 있습니다. Secrets Manager의 이벤트를 포함하여 AWS 계정의 이벤트를 지속적으로 기록하려면 Amazon S3 버킷으로 로그 파일을 전송하는 트레일을 생성하십시오. CloudTrail AWS 계정의 트레일 생성을 참조하십시오. 여러 AWS 계정 및 에서 CloudTrail 로그 파일을 CloudTrail 수신하도록 구성할 수도 AWS 리전있습니다.

CloudTrail 로그에서 수집된 데이터를 추가로 분석하고 이에 따라 조치를 취하도록 다른 AWS 서비스를 구성할 수 있습니다. CloudTrail 로그와의AWS 서비스 통합을 참조하십시오. Amazon S3 버킷에 새 로그 파일을 CloudTrail 게시할 때 알림을 받을 수도 있습니다. 에 대한 Amazon SNS 알림 구성을 참조하십시오 CloudTrail.

CloudTrail 로그에서 Secrets Manager 이벤트를 검색하려면 (콘솔)

1. https://console.aws.amazon.com/cloudtrail/ 에서 CloudTrail 콘솔을 엽니다.
2. 콘솔이 이벤트가 발생한 지역을 가리키는지 확인하십시오. 콘솔에는 선택한 지역에서 발생한 이벤트만 표시됩니다. 콘솔 오른쪽 상단의 드롭다운 목록에서 지역을 선택합니다.

3. 왼쪽 탐색 창에서 Event history(이벤트 기록)을 선택합니다.
4. 필터 기준 및/또는 시간 범위를 선택하면 원하는 이벤트를 찾는 데 도움이 됩니다. 예:

   a. 모든 Secrets Manager 이벤트를 보려면 조회 속성에서 이벤트 소스를 선택합니다. 그런 다음 이벤트 소스 입력에서 **secretsmanager.amazonaws.com**을 선택합니다.

   b. 시크릿에 대한 모든 이벤트를 보려면 조회 속성에서 리소스 이름을 선택합니다. 그런 다음 리소스 이름 입력에 암호 이름을 입력합니다.
5. 추가 세부 정보를 보려면 이벤트 옆에 있는 확장 화살표를 선택하십시오. 사용 가능한 모든 정보를 보려면 이벤트 보기를 선택합니다.

AWS CLI

Example CloudTrail 로그에서 Secrets Manager 이벤트 검색

다음 `lookup-events` 예에서는 Secrets Manager 이벤트를 검색합니다.

```
aws cloudtrail lookup-events \
    --region us-east-1 \
    --lookup-attributes
 AttributeKey=EventSource,AttributeValue=secretsmanager.amazonaws.com
```

AWS CloudTrail Secrets Manager용 엔트리

AWS Secrets Manager 모든 Secrets Manager 작업과 순환 및 삭제와 관련된 기타 이벤트에 대한 항목을 AWS CloudTrail 로그에 기록합니다. 이러한 이벤트를 설정하는 방법에 대한 자세한 내용은 Secrets Manager 이벤트와 매칭하세요 EventBridge 섹션을 참조하세요.

로그 항목 유형

- Secrets Manager 작업에 대한 로그 항목
- 삭제할 로그 항목
- 복제를 위한 로그 항목
- 교체에 대한 로그 항목

Secrets Manager 작업에 대한 로그 항목

Secrets Manager 작업을 호출하여 생성되는 이벤트에는 `"detail-type": ["AWS API Call via CloudTrail"]`이(가) 있습니다.

> **Note**
>
> 2024년 2월 이전에 일부 Secrets Manager 운영에서는 비밀 ARN에 대해 “arn” 대신 “Arn”이 포함된 이벤트를 보고했습니다. 자세한 내용은 AWS re:Post를 참조하세요.

다음은 사용자 또는 서비스가 API, SDK 또는 CLI를 통해 Secrets Manager 작업을 호출할 때 생성되는 CloudTrail 항목입니다.

BatchGetSecretValue

작업에 의해 생성됩니다. BatchGetSecretValue 보안 암호 검색에 대한 정보는 비밀 찾기를 참조하세요.

CancelRotateSecret

CancelRotateSecret작업에 의해 생성됩니다. 교체에 대한 정보는 보안 암호 교체를 참조하세요.

CreateSecret

CreateSecret작업에 의해 생성됩니다. 보안 암호 생성에 대한 정보는 보안 암호 생성 및 관리를 참조하세요.

DeleteResourcePolicy

DeleteResourcePolicy작업에 의해 생성됩니다. 권한에 대한 정보는 인증 및 액세스 제어를 참조하세요.

DeleteSecret

DeleteSecret작업에 의해 생성됩니다. 보안 암호 삭제에 대한 정보는 the section called “보안 암호 삭제”를 참조하세요.

DescribeSecret

DescribeSecret작업에 의해 생성됩니다.

GetRandomPassword

GetRandomPassword작업에 의해 생성됩니다.

GetResourcePolicy

GetResourcePolicy작업에 의해 생성됩니다. 권한에 대한 정보는 인증 및 액세스 제어를 참조하세요.

GetSecretValue

GetSecretValue및 BatchGetSecretValue작업에 의해 생성됩니다. 보안 암호 검색에 대한 정보는 비밀 찾기를 참조하세요.

ListSecrets

ListSecrets작업에 의해 생성됩니다. 보안 암호 목록에 대한 정보는 the section called “보안 암호 찾기”를 참조하세요.

ListSecretVersionIds

ListSecretVersionIds작업에 의해 생성됩니다.

PutResourcePolicy

PutResourcePolicy작업에 의해 생성됩니다. 권한에 대한 정보는 인증 및 액세스 제어를 참조하세요.

PutSecretValue

PutSecretValue작업에 의해 생성됩니다. 보안 암호 업데이트에 대한 정보는 the section called “보안 암호 수정”을 참조하세요.

RemoveRegionsFromReplication

RemoveRegionsFromReplication작업에 의해 생성됩니다. 보안 암호 복제에 대한 정보는 지역 간 비밀 복제를 참조하세요.

ReplicateSecretToRegions

ReplicateSecretToRegions작업에 의해 생성됩니다. 보안 암호 복제에 대한 정보는 지역 간 비밀 복제를 참조하세요.

RestoreSecret

RestoreSecret작업에 의해 생성됩니다. 삭제된 보안 암호 복원에 대한 정보는 the section called “보안 암호 복원”을 참조하세요.

RotateSecret

RotateSecret작업에 의해 생성됩니다. 교체에 대한 정보는 보안 암호 교체를 참조하세요.

StopReplicationToReplica

StopReplicationToReplica작업에 의해 생성됩니다. 보안 암호 복제에 대한 정보는 지역 간 비밀 복제를 참조하세요.

TagResource

TagResource작업에 의해 생성됩니다. 보안 암호 태그 지정에 대한 정보는 the section called “ 보안 암호 태그 지정”을 참조하세요.

UntagResource

UntagResource작업에 의해 생성됩니다. 보안 암호 태그 해제에 대한 정보는 the section called “ 보안 암호 태그 지정”을 참조하세요.

UpdateSecret

UpdateSecret작업에 의해 생성됩니다. 보안 암호 업데이트에 대한 정보는 the section called “보안 암호 수정”을 참조하세요.

UpdateSecretVersionStage

UpdateSecretVersionStage작업에 의해 생성됩니다. 버전 단계에 대한 정보는 the section called “시크릿 버전”을 참조하세요.

ValidateResourcePolicy

ValidateResourcePolicy작업에 의해 생성됩니다. 권한에 대한 정보는 인증 및 액세스 제어를 참조하세요.

삭제할 로그 항목

Secrets Manager는 Secrets Manager 작업을 위한 이벤트 외에도 삭제와 관련된 다음과 같은 이벤트를 생성합니다. 이러한 이벤트에는 `"detail-type": ["AWS Service Event via CloudTrail"]`이(가) 포함되어 있습니다.

CancelSecretVersionDelete

Secrets Manager 서비스에서 생성됩니다. 버전이 있는 암호에 대해 `DeleteSecret`을 호출한 다음 나중에 `RestoreSecret`을 호출하면 Secrets Manager는 복원된 각 보안 암호 버전에 대해 이 이벤트를 로그합니다. 삭제된 보안 암호 복원에 대한 정보는 the section called “보안 암호 복원”을 참조하세요.

EndSecretVersionDelete

암호 버전이 삭제되면 Secrets Manager 서비스에서 생성됩니다. 자세한 정보는 the section called “보안 암호 삭제”을 참조하세요.

StartSecretVersionDelete

Secrets Manager가 암호 버전 삭제를 시작하면 Secrets Manager 서비스에서 생성됩니다. 보안 암호 삭제에 대한 정보는 the section called “보안 암호 삭제”를 참조하세요.

SecretVersionDeletion

Secrets Manager가 더 이상 사용되지 않는 보안 암호 버전 삭제를 시작하면 Secrets Manager 서비스에서 생성됩니다. 자세한 내용은 보안 암호 버전을 참조하세요.

복제를 위한 로그 항목

Secrets Manager는 Secrets Manager 작업을 위한 이벤트 외에도 복제와 관련된 다음과 같은 이벤트를 생성합니다. 이러한 이벤트에는 `"detail-type": ["AWS Service Event via CloudTrail"]`이(가) 포함되어 있습니다.

ReplicationFailed

복제가 실패하면 Secrets Manager 서비스에서 생성됩니다. 보안 암호 복제에 대한 정보는 지역 간 비밀 복제를 참조하세요.

ReplicationStarted

Secrets Manager가 보안 암호 복제를 시작하면 Secrets Manager 서비스에서 생성됩니다. 보안 암호 복제에 대한 정보는 지역 간 비밀 복제를 참조하세요.

ReplicationSucceeded

보안 암호가 성공적으로 복제되면 Secrets Manager 서비스에서 생성됩니다. 보안 암호 복제에 대한 정보는 지역 간 비밀 복제를 참조하세요.

교체에 대한 로그 항목

Secrets Manager는 Secrets Manager 작업을 위한 이벤트 외에도 교체와 관련된 다음과 같은 이벤트를 생성합니다. 이러한 이벤트에는 `"detail-type": ["AWS Service Event via CloudTrail"]`이(가) 포함되어 있습니다.

RotationStarted

Secrets Manager가 암호 교체를 시작하면 Secrets Manager 서비스에서 생성됩니다. 교체에 대한 정보는 보안 암호 교체를 참조하세요.

RotationAbandoned

Secrets Manager가 교체 시도를 중단하고 기존 보안 암호 버전에서 AWSPENDING 라벨을 제거하면 Secrets Manager 서비스에서 생성됩니다. Secrets Manager는 교체 중에 보안 암호의 새 버전을 생성하면 교체를 중단합니다. 교체에 대한 정보는 보안 암호 교체를 참조하세요.

RotationFailed

교체가 실패하면 Secrets Manager 서비스에서 생성됩니다. 교체에 대한 정보는 the section called “ 교체 문제 해결”를 참조하세요.

RotationSucceeded

암호가 성공적으로 교체되면 Secrets Manager 서비스에서 생성됩니다. 교체에 대한 정보는 보안 암호 교체를 참조하세요.

TestRotationStarted

Secrets Manager에서 즉시 교체가 예약되지 않은 보안 암호에 대한 교체 테스트를 시작하면 Secrets Manager 서비스에서 생성됩니다. 교체에 대한 정보는 보안 암호 교체를 참조하세요.

TestRotationSucceeded

Secrets Manager에서 즉시 교체가 예약되지 않은 보안 암호에 대한 교체 테스트를 성공적으로 마치면 Secrets Manager 서비스에서 생성됩니다. 교체에 대한 정보는 보안 암호 교체를 참조하세요.

TestRotationFailed

Secrets Manager에서 즉시 교체가 예약되지 않은 보안 암호에 대한 교체 테스트한 후 교체에 실패하면 Secrets Manager 서비스에서 생성됩니다. 교체에 대한 정보는 the section called “ 교체 문제 해결”를 참조하세요.

AWS Secrets Manager 아마존으로 모니터링하기 CloudWatch

CloudWatchAmazon을 사용하면 AWS 서비스를 모니터링하고 메트릭이 변경될 때 알려주는 경보를 생성할 수 있습니다. CloudWatch 이러한 통계를 15개월 동안 보관하므로 기록 정보에 액세스하고 웹 애플리케이션 또는 서비스 성능을 더 잘 파악할 수 있습니다. 의 경우 삭제 표시된 암호를 포함하여 계정에 있는 암호의 수와 콘솔을 통한 호출을 포함하여 Secrets Manager에 대한 API 호출을 모니터링할 수 있습니다. AWS Secrets Manager메트릭을 모니터링하는 방법에 대한 자세한 내용은 사용 CloudWatch 설명서의 CloudWatch 메트릭 사용을 참조하십시오.

Secrets Manager 지표를 찾으려면

1. CloudWatch 콘솔의 지표에서 모든 지표를 선택합니다.

2. 지표 검색 상자에 를 입력합니다`secret`.
3. 다음을 따릅니다.
   - 계정의 비밀 개수를 SecretsManager 모니터링하려면 AWS/를 선택한 다음 선택하십시오 SecretCount. 이 지표는 매시간 게시됩니다.
   - 콘솔을 통한 호출을 포함하여 Secrets Manager에 대한 API 호출을 모니터링하려면 사용량 > AWS 리소스별을 선택한 다음 모니터링할 API 호출을 선택합니다. Secrets Manager API 목록은 Secrets Manager 작업을 참조하십시오.
4. 다음을 따릅니다.
   - 지표의 그래프를 생성하려면 Amazon CloudWatch 사용 설명서의 그래프 지표를 참조하십시오.
   - 이상 징후를 감지하려면 Amazon 사용 설명서의 CloudWatch 예외 항목 탐지 사용을 참조하십시오. CloudWatch
   - 지표에 대한 통계를 가져오려면 Amazon CloudWatch 사용 설명서의 지표에 대한 통계 가져오기를 참조하십시오.

CloudWatch 알람

지표 값이 변경될 때 Amazon SNS 메시지를 전송하여 CloudWatch 경보의 상태가 변경되도록 하는 경보를 생성할 수 있습니다. 계정의 비밀 `ResourceCount` 개수인 Secrets Manager 지표에 경보를 설정할 수 있습니다. 경보를 설정할 수도 있습니다. 경보는 지정한 기간 동안 지표를 감시하고 일정 기간 동안 지정된 임계값을 기준으로 측정치를 기준으로 조치를 수행합니다. 경보는 지속적인 상태 변경에 대한 조치만 호출합니다. CloudWatch 경보는 단순히 특정 상태에 있다는 이유만으로 조치를 호출하지 않습니다. 경보는 지정된 기간 동안 상태가 변경되고 유지되어야 합니다.

자세한 내용은 사용 설명서의 Amazon CloudWatch 경보 사용 및 이상 탐지에 기반한 CloudWatch 경보 생성을 참조하십시오. CloudWatch

특정 임계값을 주시하다가 해당 임계값이 충족될 때 알림을 전송하거나 조치를 취하도록 경보를 설정할 수도 있습니다. 자세한 내용은 Amazon CloudWatch 사용 설명서를 참조하십시오.

Amazon과 AWS Secrets Manager 이벤트 맞추기 EventBridge

EventBridgeAmazon에서는 CloudTrail 로그 항목의 Secrets Manager 이벤트를 매칭할 수 있습니다. 이러한 이벤트를 찾은 다음 새로 생성된 이벤트를 대상으로 전송하여 조치를 취하도록 EventBridge 규칙을 구성할 수 있습니다. Secrets Manager가 기록하는 CloudTrail 항목 목록은 을 참조하십시

오CloudTrail 출품작. 설정 EventBridge 지침은 사용 설명서의 EventBridgeEventBridge 시작하기를 참조하십시오.

모든 변경 사항을 지정된 암호와 매칭

> **Note**
>
> 일부 Secrets Manager 이벤트는 대소문자가 다른 암호의 ARN을 반환하므로 둘 이상의 작업과 매칭되는 이벤트 패턴에서 ARN으로 암호를 지정하려면 `arn` 와(과) `aRN` 키를 모두 포함해야 할 수 있습니다. 자세한 내용은 AWS re:Post를 참조하세요.

다음 예제는 암호 변경에 대한 로그 항목과 일치하는 EventBridge 이벤트 패턴을 보여줍니다.

```
{
    "source": ["aws.secretsmanager"],
    "detail-type": ["AWS API Call via CloudTrail"],
    "detail": {
        "eventSource": ["secretsmanager.amazonaws.com"],
        "eventName": ["DeleteResourcePolicy", "PutResourcePolicy", "RotateSecret",
 "TagResource", "UntagResource", "UpdateSecret"],
        "responseElements": {
            "arn": ["arn:aws:secretsmanager:us-west-2:012345678901:secret:mySecret-
a1b2c3"]
        }
    }
}
```

암호 값이 교체될 때의 이벤트 매칭

다음 예제는 수동 업데이트 또는 자동 교체로 인해 발생하는 암호 값 변경에 대한 CloudTrail 로그 항목을 일치시키는 EventBridge 이벤트 패턴을 보여줍니다. 이러한 이벤트 중 일부는 Secrets Manager 작업에서 생성되고 일부는 Secrets Manager 서비스에서 생성되므로 두 이벤트 모두에 대해 `detail-type`을(를) 포함해야 합니다.

```
{
    "source": ["aws.secretsmanager"],
    "$or": [
        { "detail-type": ["AWS API Call via CloudTrail"] },
        { "detail-type": ["AWS Service Event via CloudTrail"] }
```

```
    ],
    "detail": {
        "eventSource": ["secretsmanager.amazonaws.com"],
        "eventName": ["PutSecretValue", "UpdateSecret", "RotationSucceeded"]
    }
}
```

삭제 예정인 AWS Secrets Manager 비밀에 대한 액세스 시기 모니터링

Amazon CloudWatch Logs와 Amazon Simple Notification Service (Amazon SNS) 를 함께 사용하여 삭제 대기 중인 비밀에 액세스하려는 모든 시도를 알리는 경보를 생성할 수 있습니다. AWS CloudTrail이러한 경보로부터 알림을 받으면 보안 암호 삭제를 취소하여, 정말로 삭제할지 판단할 시간을 확보하고 싶을 수 있습니다. 조사를 통해 필요하다고 판단되면 보안 암호를 복원할 수 있습니다. 또는 사용자가 사용할 새 암호 정보로 사용자를 업데이트해야 할 수 있습니다.

다음 절차는 `GetSecretValue` 작업 요청으로 인해 특정 오류 메시지가 CloudTrail 로그 파일에 기록될 때 알림을 받는 방법을 설명합니다. 경보를 트리거하지 않고 보안 암호에 대해 다른 API 작업을 수행할 수 있습니다. 이 CloudWatch 경보는 개인 또는 애플리케이션이 오래된 자격 증명을 사용하고 있음을 나타낼 수 있는 사용량을 탐지합니다.

이 절차를 시작하기 전에 AWS Secrets Manager API 요청을 모니터링하려는 AWS 리전 및 계정을 CloudTrail 켜야 합니다. 지침은 AWS CloudTrail 사용 설명서의 첫 번째 추적 생성을 참조하세요.

1단계: CloudWatch Logs에 CloudTrail 로그 파일 전송을 구성합니다.

CloudTrail 로그 파일을 CloudWatch Logs로 전달하도록 구성해야 합니다. 이렇게 하면 CloudWatch Logs가 Secrets Manager API 요청을 모니터링하여 삭제 보류 중인 비밀을 검색할 수 있습니다.

로그에 대한 CloudTrail 로그 파일 전송을 구성하려면 CloudWatch

1. https://console.aws.amazon.com/cloudtrail/ 에서 CloudTrail 콘솔을 엽니다.
2. 상단 탐색 표시줄에서 비밀을 AWS 리전 모니터링할 항목을 선택합니다.
3. 왼쪽 탐색 창에서 Trails를 선택한 다음 구성할 트레일의 이름을 선택합니다. CloudWatch
4. 트레일 구성 페이지에서 CloudWatch 로그 섹션까지 아래로 스크롤한 다음 편집 아이콘 ()
   
   을 선택합니다.

5. New or existing log group(새 또는 기존 로그 그룹)에 로그 그룹 이름을 입력합니다(예: **`CloudTrail/MyCloudWatchLogGroup`**).
6. IAM 역할의 경우 CloudTrail_ CloudWatchLogs _Role이라는 기본 역할을 사용할 수 있습니다. 이 역할에는 CloudTrail 이벤트를 로그 그룹에 전달하는 데 필요한 권한이 포함된 기본 역할 정책이 있습니다.
7. 계속을 선택하여 구성을 저장합니다.
8. 계정의 API 활동과 관련된 CloudTrail 이벤트를 CloudWatch Logs log group 페이지로 전달할 것에서 허용을 선택합니다.AWS CloudTrail

2단계: CloudWatch 알람 생성

Secrets Manager `GetSecretValue` API 작업에서 삭제 보류 중인 비밀에 대한 액세스를 요청할 때 알림을 받으려면 CloudWatch 경보를 생성하고 알림을 구성해야 합니다.

CloudWatch 경보를 만들려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔에 로그인합니다.
2. 상단 내비게이션 바에서 비밀을 모니터링하려는 AWS 지역을 선택합니다.
3. 왼쪽 탐색 창에서 로그를 선택합니다.
4. 로그 그룹 목록에서 이전 절차에서 생성한 로그 그룹 옆의 확인란을 선택합니다 (예: CloudTrail/)MyCloudWatchLogGroup. 그런 다음 [Create Metric Filter]를 선택합니다.
5. [Filter Pattern]에 다음을 입력하거나 붙여 넣습니다.

   ```
   { $.eventName = "GetSecretValue" && $.errorMessage = "*secret because it was marked
    for deletion*" }
   ```

   [Assign Metric]을 선택합니다.
6. [Create Metric Filter and Assign a Metric] 페이지에서 다음을 수행합니다.

   a. 지표 네임스페이스에 **`CloudTrailLogMetrics`**를 입력합니다.

   b. 지표 이름에 **`AttemptsToAccessDeletedSecrets`**를 입력합니다.

   c. 고급 지표 설정 표시를 선택한 후 필요에 따라 지표 값에 **`1`**을 입력합니다.

   d. 필터 생성을 선택합니다.
7. 필터 상자에서 [Create Alarm]을 선택합니다.
8. [Create Alarm] 창에서 다음과 같이 실행합니다.

a. Name에 **AttemptsToAccessDeletedSecretsAlarm**를 입력합니다.

b. 다음 경우 항상:의 결과 값:에서 **>=**을 선택하고 **1**을 입력합니다.

c. [Send notification to:] 옆에서 다음 중 하나를 실시합니다.

- 새로운 Amazon SNS 주제를 생성해 사용하려면 새 목록(New list)을 선택한 후 새 주제 이름을 입력합니다. 메일 주소 목록(Email list):에 이메일 주소를 하나 이상 입력합니다. 쉼표로 구분하여 두 개 이상의 이메일 주소를 입력할 수 있습니다.
- 기존 Amazon SNS 주제를 사용하려면 사용할 주제의 이름을 선택합니다. 목록이 없으면 목록 선택을 선택합니다.

d. 경보 생성을 선택합니다.

3단계: CloudWatch 알람 테스트

경보를 테스트하려면 보안 암호를 생성한 다음 삭제를 예약합니다. 그런 다음 보안 암호 값을 검색해 봅니다. 잠시 후 경보에 구성한 주소로 이메일이 수신됩니다. 삭제하도록 예정된 보안 암호 사용에 대해 알려줍니다.

다음을 사용하여 규정 준수를 위한 AWS Secrets Manager 비밀 모니터링 AWS Config

를 사용하여 비밀을 AWS Config 평가하여 표준을 준수하는지 확인할 수 있습니다. AWS Config 규칙을 사용하여 비밀에 대한 내부 보안 및 규정 준수 요구 사항을 정의합니다. 그러면 AWS Config 규칙을 준수하지 않는 비밀을 식별할 수 있습니다. 또한 비밀 메타데이터, 순환 구성, 비밀 암호화에 사용되는 KMS 키, Lambda 순환 함수, 비밀과 관련된 태그의 변경 사항을 추적할 수 있습니다.

변경 사항을 AWS Config 알리도록 구성할 수 있습니다. 자세한 내용은 Amazon SNS 주제로 AWS Config 보내는 알림을 참조하십시오.

조직 내에 여러 곳에 비밀이 있는 경우 해당 구성 AWS 계정 및 AWS 리전 규정 준수 데이터를 집계할 수 있습니다. 자세한 내용은 다중 계정 다중 지역 데이터 집계를 참조하십시오.

비밀이 규정을 준수하는지 여부를 평가하려면

- AWS Config 규칙을 사용한 리소스 평가의 지침을 따르고 다음 규칙 중 하나를 선택하세요.
  - `secretsmanager-secret-unused` — 보안 암호를 지정된 일수 내에 액세스했는지 확인합니다.

- secretsmanager-using-cmk— 에서 생성한 키 AWS 관리형 키 aws/secretsmanager 또는 고객 관리 키를 사용하여 비밀이 암호화되는지 확인합니다 AWS KMS.
- secretsmanager-rotation-enabled-check — Secrets Manager에 저장된 보안 암호에 대해 교체가 구성되었는지 확인합니다.
- secretsmanager-scheduled-rotation-success-check— 마지막으로 성공한 교체가 설정된 교체 주기 내에 있는지 확인합니다. 최소 확인 주기는 매일입니다.
- secretsmanager-secret-periodic-rotation - 보안 암호를 지정된 일수 내에 교체했는지 확인합니다.

Secrets Manager 비용 모니터링

Amazon을 사용하여 예상 AWS Secrets Manager 요금을 CloudWatch 모니터링할 수 있습니다. 자세한 내용은 CloudWatch 사용 설명서의 예상 AWS 요금을 모니터링하기 위한 청구 경보 생성을 참조하십시오.

비용을 모니터링할 수 있는 또 다른 옵션은 AWS 비용 이상 탐지입니다. 자세한 내용은 Cost Management User Guide의 AWS 비용 예외 항목 탐지를 통한 비정상적 지출 감지를 참조하십시오.AWS

Secrets Manager 사용 모니터링에 대한 자세한 내용은 the section called “모니터: CloudWatch” 및 을 참조하십시오the section called “다음과 같이 로그하십시오. AWS CloudTrail ”.

AWS Secrets Manager 요금에 대한 자세한 내용은 을 참조하십시오the section called “요금”.

규정 준수 검증 대상 AWS Secrets Manager

Secrets Manager를 사용할 때의 규정 준수 책임은 데이터의 민감도, 회사의 규정 준수 목표, 관련 법률 및 규정에 따라 결정됩니다. AWS 규정 준수에 도움이 되는 다음 리소스를 제공합니다.

- 보안 및 규정 준수 빠른 시작 안내서: 이 배포 안내서에서는 아키텍처 고려 사항에 관해 설명하고 AWS에서 보안 및 규정 준수에 중점을 둔 기본 환경을 배포하기 위한 단계를 제공합니다.
- HIPAA 보안 및 규정 준수를 위한 설계 백서 — 이 백서는 기업이 HIPAA 준수 애플리케이션을 개발하는 데 사용할 AWS 수 있는 방법을 설명합니다.
- AWS 규정 준수 리소스 규정 — 이 통합 문서 및 가이드 모음은 해당 산업 및 지역에 적용될 수 있습니다.
- AWS Config으로 리소스 구성이 내부 관행, 업계 지침 및 규정을 준수하는 정도를 평가할 수 있습니다. 자세한 정보는 the section called “규정 준수를 위한 비밀 모니터링”을 참조하세요.
- AWS Security Hub보안 업계 표준 및 모범 사례를 준수하는지 확인하는 데 도움이 AWS 되는 내부 보안 상태를 종합적으로 보여줍니다. Security Hub를 사용하여 Secrets Manager 리소스를 평가하는 방법에 대한 자세한 내용은AWS Security Hub 사용 설명서의 AWS Secrets Manager 제어를 참조하세요.
- IAM Access Analyzer는 외부 엔터티가 보안 암호에 액세스할 수 있도록 허용하는 정책의 조건 문을 비롯해 정책을 분석합니다. 자세한 내용은 Access Analyzer를 사용하여 액세스 미리 보기를 참조하세요.
- AWS Systems Manager은 Secrets Manager에 대해 미리 정의된 실행서를 제공합니다. 자세한 내용은 Secrets Manager에 대한 Systems Manager Automation 실행서 참조를 참조하세요.
- 를 사용하여 타사 감사 보고서를 다운로드할 수 AWS Artifact있습니다. 자세한 내용은 의 보고서 https://docs.aws.amazon.com/artifact/latest/ug/downloading-documents.html 참조하십시오 AWS Artifact.

규정 준수 표준

AWS Secrets Manager 다음 표준에 대한 감사를 거쳤으며 규정 준수 인증을 받아야 하는 경우 솔루션에 포함될 수 있습니다.

- HIPAA — AWS HIPAA 적격 서비스를 포함하도록 건강 보험 양도 및 책임법 (HIPAA) 규정 준수 프로그램을 확대했습니다. AWS Secrets Manager 비즈니스 제휴 계약 (BAA) 을 체결한 경우 Secrets Manager를 사용하여 HIPAA 준수 애플리케이션을 구축할 수 있습니다. AWS AWS 의료 정보의 처

리 및 보관에 활용할 AWS 수 있는 방법에 대해 자세히 알아보고 싶은 고객을 위해 HIPAA에 초점을 맞춘 백서를 제공합니다. 자세한 내용은 HIPAA 규정 준수를 참조하세요.

- PIC 참여 기관 — AWS Secrets Manager 서비스 제공업체 레벨 1의 결제 카드 산업 규정 준수 증명 (PCI) 데이터 보안 표준 (DSS) 버전 3.2를 보유하고 있습니다. AWS 제품 및 서비스를 사용하여 카드 소지자 데이터를 저장, 처리 또는 전송하는 고객은 자체 PCI AWS Secrets Manager DSS 규정 준수 인증을 관리할 때 사용할 수 있습니다. PCI AWS 컴플라이언스 패키지의 사본을 요청하는 방법을 포함하여 PCI DSS에 대한 자세한 내용은 PCI DSS 레벨 1을 참조하십시오.
- ISO — AWS Secrets Manager ISO/IEC 27001, ISO/IEC 27017, ISO/IEC 27018 및 ISO 9001에 대한 규정 준수 인증을 성공적으로 완료했습니다. 자세한 정보는 ISO 27001, ISO 27017, ISO 27018, ISO 9001을 참조하세요.
- AICPA SOC — 시스템 및 조직 제어 (SOC) 보고서는 Secrets Manager가 주요 규정 준수 제어 및 목표를 달성하는 방법을 보여주는 독립적인 타사 검토 보고서입니다. 이 보고서의 목적은 귀하와 감사자가 운영 및 규정 준수를 지원하기 위해 수립된 AWS 제어 체계를 이해하는 데 도움이 되는 것입니다. 자세한 내용은 SOC 규정 준수를 참조하세요.
- FedRAMP — 연방 위험 및 권한 관리 프로그램 (FedRAMP) 은 클라우드 제품 및 서비스에 대한 보안 평가, 권한 부여 및 지속적 모니터링에 대한 표준화된 접근 방식을 제공하는 정부 차원의 프로그램입니다. FedRAMP 프로그램은 또한 GovCloud 동서부의 서비스 및 지역에 대한 잠정 승인을 제공하고 정부 또는 규제 대상 데이터를 소비합니다. 자세한 내용은 FedRAMP 규정 준수를 참조하세요.
- 국방부 — 국방부 (DoD) 클라우드 컴퓨팅 보안 요구 사항 가이드 (SRG) 는 클라우드 서비스 제공업체 (CSP) 가 DoD 잠정 인증을 획득하여 DoD 고객에게 서비스를 제공할 수 있도록 하는 표준화된 평가 및 권한 부여 프로세스를 제공합니다. 자세한 내용은 DoD SRG 리소스를 참조하세요.
- IRAP — 정보 보안 등록 평가 프로그램 (IRAP) 을 통해 호주 정부 고객은 적절한 통제가 마련되어 있는지 확인하고 호주 사이버 보안 센터 (ACSC) 에서 작성한 호주 정부 정보 보안 매뉴얼 (ISM) 의 요구 사항을 해결하기 위한 적절한 책임 모델을 결정할 수 있습니다. 자세한 내용은 IRAP 리소스를 참조하세요.
- OSPAR — Amazon Web Services (AWS) 는 아웃소싱 서비스 제공업체의 감사 보고서 (OSPAR) 인증을 획득했습니다. AWS 아웃소싱 서비스 공급자의 통제 목표 및 절차에 관한 ABS (싱가포르 은행협회) 지침 (ABS 지침) 을 준수하는 것은 싱가포르 금융 서비스 업계가 설정한 클라우드 서비스 공급자에 대한 높은 기대치를 충족하려는 고객의 AWS 노력을 보여줍니다. 자세한 내용은 OSPAR 리소스를 참조하세요.

AWS Secrets Manager의 보안

AWS에서는 클라우드 보안을 가장 중요하게 생각합니다. AWS 고객은 보안에 가장 보안에 민감한 조직의 요구 사항에 부합하도록 구축된 데이터 센터 및 네트워크 아키텍처의 혜택을 누릴 수 있습니다.

고객과 AWS가 보안 책임을 공유합니다. 공동 책임 모델은 이 사항을 클라우드 내 보안 및 클라우드의 보안으로 설명합니다.

- 클라우드의 보안 - AWS는 AWS 클라우드에서 AWS 서비스를 실행하는 인프라를 보호합니다. AWS는 또한 안전하게 사용할 수 있는 서비스를 제공합니다. 타사 감사자는 정기적으로 AWS 규정 준수 프로그램의 일환으로 보안 효과를 테스트하고 검증합니다. AWS Secrets Manager에 적용되는 규정 준수 프로그램에 대한 자세한 내용은 규정 준수 프로그램 제공 범위 내의 AWS 서비스를 참조하세요.
- 클라우드 내 보안 – AWS 서비스에서 고객의 책임을 결정합니다. 또한 귀하는 데이터의 민감도, 회사 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다.

더 많은 리소스를 보려면 보안 원칙 - AWS Well-Architected 프레임워크를 참조하세요.

주제

- AWS Secrets Manager 보안 암호 저장 시 AWS CLI 사용으로 발생 가능한 위험 줄이기
- AWS Secrets Manager의 데이터 보호
- 의 비밀 암호화 및 복호화 AWS Secrets Manager
- AWS Secrets Manager의 인프라 보안
- 내에서의 레질리언스 AWS Secrets Manager
- 포스트 양자 TLS

AWS Secrets Manager 보안 암호 저장 시 AWS CLI 사용으로 발생 가능한 위험 줄이기

AWS Command Line Interface(AWS CLI)를 사용하여 AWS 작업을 호출하려면 명령 셸에 해당 명령을 입력합니다. 예를 들어 Windows 명령 프롬프트나 Windows PowerShell 또는 Bash나 Z shell 등을 사용할 수 있습니다. 이러한 명령 셸 대부분에는 생산성을 높이기 위해 설계된 기능이 포함되어 있습니다. 하지만, 이 기능은 보안 암호의 보안을 약화시키는 데 사용될 수 있습니다. 예를 들어 대부분의 셸에서

위쪽 화살표 키를 사용하면 마지막으로 입력한 명령을 볼 수 있습니다. 명령 기록 기능은 보호되지 않는 세션에 액세스한 누군가가 악용할 수 있습니다. 또한 백그라운드에서 작동하는 다른 기능이 명령 파라미터에 액세스할 수 있습니다. 이러한 기능은 모두 작업을 보다 효율적으로 수행하도록 지원하기 위한 것입니다. 이러한 위험을 줄이기 위해서는 다음 단계를 수행해야 합니다.

- 콘솔에서 자리를 비우는 경우에는 항상 컴퓨터를 잠급니다.
- 필요 없거나 더 이상 사용하지 않는 콘솔 유틸리티는 제거하거나 비활성화합니다.
- 셸 또는 원격 액세스 프로그램(사용하는 경우)에서 입력한 명령을 로깅하지 않도록 합니다.
- 셸 명령 기록에서 캡처하지 않는 파라미터를 전달하는 기술을 사용합니다. 다음 예는 텍스트 파일에 보안 암호 텍스트를 입력한 후 이 파일을 AWS Secrets Manager 명령으로 전달하고 바로 폐기하는 방법을 보여줍니다. 이는 일반적인 셸 기록에 보안 암호 텍스트가 캡처되지 않는다는 것을 의미합니다.

  다음 예에는 일반적인 Linux 명령이 나와 있습니다. 사용 중인 셸에 약간 다른 명령이 필요할 수 있습니다.

```
$ touch secret.txt
        # Creates an empty text file
$ chmod go-rx secret.txt
        # Restricts access to the file to only the user
$ cat > secret.txt
        # Redirects standard input (STDIN) to the text file
ThisIsMyTopSecretPassword^D
        # Everything the user types from this point up to the CTRL-D (^D) is saved in
 the file
$ aws secretsmanager create-secret --name TestSecret --secret-string file://
secret.txt       # The Secrets Manager command takes the --secret-string parameter
 from the contents of the file
$ shred -u secret.txt
        # The file is destroyed so it can no longer be accessed.
```

이러한 명령을 실행한 후에는 위쪽 및 아래쪽 화살표를 사용해 명령 기록을 스크롤하고 보안 암호 텍스트가 임의의 행에 표시되지 않았는지 확인할 수 있어야 합니다.

> **Important**
>
> 기본적으로 명령 기록 버퍼 크기를 먼저 1로 줄이지 않으면 Windows에서는 동일한 기능을 수행할 수 없습니다.

Windows 명령 프롬프트를 1개의 명령 기록 버퍼만 갖도록 구성하려면

1. 관리자 명령 프롬프트(관리자로 실행)를 엽니다.
2. 왼쪽 위에 있는 아이콘을 선택한 다음 속성을 선택합니다.
3. 옵션 탭에서 버퍼 크기 및 버퍼 수를 둘 다 **1**로 설정하고 확인을 선택합니다.
4. 기록에 표시하지 않으려는 명령을 입력해야 할 때마다 다음과 같은 다른 명령 하나를 즉시 입력합니다.

   ```
   echo.
   ```

   그러면 민감한 명령이 플러시됩니다.

Windows 명령 프롬프트 셸에서 SysInternals SDelete 도구를 다운로드하고 다음과 유사한 명령을 사용할 수 있습니다.

```
C:\> echo. 2> secret.txt
        # Creates an empty file
C:\> icacls secret.txt /remove "BUILTIN\Administrators" "NT AUTHORITY/SYSTEM" /
inheritance:r   # Restricts access to the file to only the owner
C:\> copy con secret.txt /y
        # Redirects the keyboard to text file, suppressing prompt to overwrite
THIS IS MY TOP SECRET PASSWORD^Z
      # Everything the user types from this point up to the CTRL-Z (^Z) is saved in the
 file
C:\> aws secretsmanager create-secret --name TestSecret --secret-string file://
secret.txt      # The Secrets Manager command takes the --secret-string parameter from
 the contents of the file
C:\> sdelete secret.txt
        # The file is destroyed so it can no longer be accessed.
```

AWS Secrets Manager의 데이터 보호

AWS 공동 책임 모델은 AWS Secrets Manager의 데이터 보호에 적용됩니다. 이 모델에서 설명하는 것처럼 AWS는 모든 AWS 클라우드를 실행하는 글로벌 인프라를 보호할 책임이 있습니다. 이 인프라에서 호스팅되는 콘텐츠에 대한 제어를 유지하는 것은 사용자의 책임입니다. 이 콘텐츠에는 사용하는 AWS 서비스 서비스의 보안 구성과 관리 작업이 포함돼 있습니다. 데이터 프라이버시에 대한 자세한 내용은 데이터 프라이버시 FAQ를 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 Identity and Access Management(IAM)를 사용하여 개별 사용자 계정을 설정하는 것이 좋습니다. 이러한 방식에서는 각 사용자에게 자신의 직무를 충실히 이행하는 데 필요한 권한만 부여됩니다. 또한 다음과 같은 방법으로 데이터를 보호하는 것이 좋습니다.

- 각 계정에 다중 인증(MFA)을 사용합니다.
- SSL/TLS를 사용하여 AWS 리소스와 통신합니다. Secrets Manager는 모든 리전에서 TLS 1.2 및 1.3을 지원합니다. Secrets Manager는 TLS (PQTLS) 네트워크 암호화 프로토콜에 대한 하이브리드 포스트 양자 키 교환 옵션도 지원합니다.
- IAM 보안 주체와 연결되어 있는 액세스 키 ID 및 비밀 액세스 키를 사용하여 Secrets Manager에 대한 프로그래밍 방식 요청에 서명합니다. 또는 AWS Security Token Service(AWS STS)를 사용하여 임시 보안 자격 증명을 생성하여 요청에 서명할 수 있습니다.
- AWS CloudTrail로 API 및 사용자 활동 로깅을 설정합니다. the section called “다음과 같이 로그하십시오. AWS CloudTrail ” 섹션을 참조하세요.
- 명령줄 인터페이스 또는 API를 통해 AWS에 액세스할 때 FIPS 140-2 검증된 암호화 모듈이 필요한 경우 FIPS 엔드포인트를 사용합니다. the section called “Secrets Manager 엔드포인트” 섹션을 참조하세요.
- AWS CLI를 사용하여 Secrets Manager에 액세스하는 경우 the section called “AWS Secrets Manager 보안 암호 저장 시 AWS CLI 사용으로 발생 가능한 위험 줄이기”.

유휴 시 암호화

Secrets Manager는 AWS Key Management Service(AWS KMS)를 통한 암호화를 사용하여, 저장된 데이터의 기밀성을 보호합니다.AWS KMS에서는 많은 AWS 서비스에서 사용되는 키 스토리지 및 암호화 서비스를 제공합니다. Secrets Manager의 모든 보안 암호는 고유한 데이터 키로 암호화됩니다. 각 데이터 키는 KMS 키에 의해 보호됩니다. 계정에 대한 Secrets Manager AWS 관리형 키로 기본 암호화를 사용하거나, AWS KMS에서 고유한 고객 관리형 키를 생성할 수도 있습니다. 고객 관리형 키를 사용하면 KMS 키 활동에 대한 권한 부여를 보다 세부적으로 제어할 수 있습니다. 자세한 내용은 the section called “보안 암호 암호화 및 복호화” 섹션을 참조하세요.

전송 중 데이터 암호화

Secrets Manager는 전송 중인 데이터를 암호화하기 위해 안전한 프라이빗 엔드포인트를 제공합니다. 안전한 프라이빗 엔드포인트를 사용하면 AWS에서 Secrets Manager에 대한 API 요청의 무결성을 보호할 수 있습니다. AWS에서 API 호출자는 X.509 인증서 및/또는 Secrets Manager 보안 액세스 키를

사용해 호출에 서명해야 합니다. 이 요구 사항은 Signature 버전 4 서명 프로세스(Sigv4)에 명시되어 있습니다.

이AWS Command Line Interface(AWS CLI) 또는 AWS SDK 중 하나를 사용하여 AWS에 대한 호출을 생성할 경우 사용할 액세스 키를 구성합니다. 그런 다음 이러한 도구는 자동으로 액세스 키를 사용하여 요청에 서명합니다. the section called “AWS Secrets Manager 보안 암호 저장 시 AWS CLI 사용으로 발생 가능한 위험 줄이기” 섹션을 참조하세요.

인터네트워크 트래픽 개인 정보 보호

AWS에는 알려진 네트워크 경로와 프라이빗 네트워크 경로를 통해 트래픽을 라우팅할 때 개인 정보를 보호하는 옵션이 있습니다.

서비스와 온프레미스 클라이언트 및 애플리케이션 간의 트래픽

프라이빗 네트워크와 AWS Secrets Manager 사이에 두 연결 옵션이 있습니다.

- AWS Site-to-Site VPN 연결. 자세한 내용은 AWS Site-to-Site VPN이란 무엇입니까?를 참조하세요.
- AWS Direct Connect 연결. 자세한 내용은 AWS Direct Connect란 무엇입니까?를 참조하세요.

같은 리전에 있는 AWS 리소스 사이의 트래픽

Secrets Manager와 AWS의 API 클라이언트 간 트래픽의 보안을 유지하려면 AWS PrivateLink를 설정해 Secrets Manager API 엔드포인트에 비공개로 액세스할 수 있습니다.

암호화 키 관리

Secrets Manager에서 보호된 보안 암호 데이터의 새 버전을 암호화해야 하는 경우 Secrets Manager는 AWS KMS에 대한 요청을 보내 KMS 키에서 새 데이터 키를 생성합니다. Secrets Manager는 봉투 암호화(envelope encryption)에 이 데이터 키를 사용합니다. Secrets Manager는 암호화된 데이터 키를 암호화된 보안 암호와 함께 저장합니다. Secrets Manager는 보안 암호를 복호화해야 할 때 AWS KMS를 요청하여 데이터 키를 복호화합니다. 그런 다음 Secrets Manager는 복호화된 데이터 키를 사용하여 암호화된 보안 암호를 복호화합니다. Secrets Manager는 데이터 키를 암호화되지 않은 형식으로 저장하지 않으며 가능한 한 빨리 메모리에서 키를 제거합니다. 자세한 내용은 the section called “보안 암호 암호화 및 복호화” 섹션을 참조하세요.

의 비밀 암호화 및 복호화 AWS Secrets Manager

Secrets Manager는 AWS KMS 키와 데이터 키가 포함된 봉투 암호화를 사용하여 각 비밀 값을 보호합니다. 시크릿의 시크릿 값이 변경될 때마다 Secrets Manager는 이를 보호하기 AWS KMS 위해 새 데이터 키를 요청한다. KMS 키 아래에 암호화된 데이터 키는 보안 암호의 메타데이터에 저장합니다. 암호를 해독하기 위해 Secrets Manager는 먼저 KMS 키를 사용하여 암호화된 데이터 키를 해독합니다. AWS KMS

Secrets Manager에서는 KMS를 사용하여 보안 암호 값을 직접 암호화하지 않습니다. 대신 KMS 키를 사용하여 256비트 고급 암호화 표준(AES) 대칭 데이터 키를 생성하고 암호화하며 데이터 키를 사용하여 보안 암호 값을 암호화합니다. Secrets Manager는 일반 텍스트 데이터 키를 사용하여 외부의 비밀 값을 암호화한 다음 메모리에서 제거합니다. AWS KMS데이터 키의 암호화된 사본을 암호의 메타데이터에 저장합니다.

주제

- 키 선택 AWS KMS
- 무엇을 암호화하나요?
- 암호화 및 복호화 프로세스
- KMS 키에 대한 권한
- Secrets Manager에서 KMS 키를 사용하는 방법
- AWS 관리형 키 (aws/secretsmanager)의 키 정책
- Secrets Manager 보안 암호 컨텍스트
- Secrets Manager와의 상호 작용을 모니터링하십시오. AWS KMS

키 선택 AWS KMS

암호를 생성할 때 AWS 계정 및 지역의 대칭 암호화 고객 관리 키를 선택하거나 Secrets Manager (`aws/secretsmanager`) AWS 관리형 키 에 사용할 수 있습니다. 를 선택했는데 아직 존재하지 않는 경우 Secrets Manager는 이를 생성하여 암호와 연결합니다. AWS 관리형 키 `aws/secretsmanager` 계정의 각 보안 암호에 대해 동일한 KMS 또는 다른 KMS 키를 사용할 수 있습니다. 암호 그룹의 키에 대한 사용자 지정 권한을 설정하거나 해당 키에 대한 특정 작업을 감사하려는 경우 다른 KMS 키를 사용할 수 있습니다. Secrets Manager는 대칭 암호화 KMS 키만 지원합니다. 외부 키 스토어에서 KMS 키를 사용하는 경우 요청이 AWS외부로 이동해야 하므로 KMS 키에 대한 암호화 작업이 더 오래 걸리고 신뢰성과 내구성이 떨어질 수 있습니다.

보안 암호의 암호화 키를 변경하는 방법에 대한 자세한 내용은 the section called “보안 암호에 대한 암호화 키 변경”을(를) 참조하세요.

암호화 키를 변경하면 Secrets Manager가 다시 암호화하고 AWSCURRENT 새 키로 AWSPREVIOUS 버전을 변경합니다. AWSPENDING Secrets Manager는 사용자가 비밀 정보에 접근할 수 없도록 기존 버전을 모두 이전 키로 암호화한 상태로 유지합니다. 즉, 이전 키 또는 새 키를 사용하여 AWSCURRENTAWSPENDING, 및 AWSPREVIOUS 버전을 해독할 수 있습니다.

새 암호화 키로만 암호를 AWSCURRENT 해독할 수 있도록 하려면 새 키로 새 버전의 암호를 만드십시오. 그러면 AWSCURRENT 보안 버전을 해독할 수 있으려면 새 키에 대한 권한이 있어야 합니다.

에 대한 권한을 AWS 관리형 키 aws/secretsmanager 거부하고 고객 관리 키로 암호를 암호화하도록 요구할 수 있습니다. 자세한 정보는 the section called “예: 암호를 암호화하기 위한 특정 AWS KMS 키 거부”을 참조하세요.

비밀과 연결된 KMS 키를 찾으려면 콘솔에서 비밀번호를 보거나 또는 를 호출하십시오 ListSecrets. DescribeSecret 시크릿이 AWS 관리형 키 for Secrets Manager (aws/secretsmanager) 와 연결된 경우 이러한 작업은 KMS 키 식별자를 반환하지 않습니다.

무엇을 암호화하나요?

Secrets Manager는 암호 값을 암호화하지만 다음 항목은 암호화하지 않습니다.

- 보안 암호 이름 및 설명
- 교체 설정
- 보안 암호와 연결된 KMS 키의 ARN
- 첨부된 모든 태그 AWS

암호화 및 복호화 프로세스

Secrets Manager는 다음 프로세스에 따라 보안 암호의 보안 암호 값을 암호화합니다.

1. Secrets Manager는 암호의 KMS 키 ID와 256비트 AES 대칭 키를 요청하여 AWS KMS GenerateDataKey작업을 호출합니다. AWS KMS 일반 텍스트 데이터 키와 KMS 키로 암호화된 해당 데이터 키의 사본을 반환합니다.
2. Secrets Manager는 일반 텍스트 데이터 키와 고급 암호화 표준 (AES) 알고리즘을 사용하여 외부의 비밀 값을 암호화합니다. AWS KMS사용한 후에는 가능한 빨리 메모리에서 일반 텍스트 키를 제거합니다.

3. Secrets Manager는 보안 암호 값을 복호화할 수 있도록 보안 암호의 메타데이터에 암호화된 데이터 키를 저장합니다. 하지만 Secrets Manager API 중에서 어떤 것도 암호화된 보안 암호나 암호화된 데이터 키를 반환하지 않습니다.

암호화된 암호 값을 해독하려면:

1. AWS KMS Secrets Manager는 암호 해독 작업을 호출하고 암호화된 데이터 키를 전달합니다.
2. AWS KMS KMS 키를 암호로 사용하여 데이터 키를 해독합니다. 그런 다음 일반 텍스트 데이터 키를 반환합니다.
3. Secrets Manager는 일반 텍스트 데이터 키를 사용하여 보안 암호 값을 복호화합니다. 그런 다음 가능한 빨리 메모리에서 데이터 키를 제거합니다.

KMS 키에 대한 권한

Secrets Manager가 암호화 작업에서 KMS 키를 사용하는 경우 보안 암호 값에 액세스하거나 업데이트하는 사용자를 대신하여 KMS 키가 작동합니다. IAM 정책 또는 키 정책에서 권한을 부여할 수 있습니다. 다음과 같은 Secrets Manager 작업에는 AWS KMS 권한이 필요합니다.

- CreateSecret
- GetSecretValue
- PutSecretValue
- UpdateSecret
- ReplicateSecretToRegions

Secrets Manager에서 시작된 요청에만 KMS 키를 사용하도록 허용하려면
권한 정책에서 kms: ViaService 조건 키를 값과 함께 사용할 수 있습니다.
`secretsmanager.<Region>.amazonaws.com`

암호화 작업에 대한 KMS 키 사용 조건으로서 암호화 컨텍스트에서 키나 값을 사용할 수도 있습니다. 예를 들면 IAM 또는 키 정책 문서에서 문자열 조건 연산자를 사용하거나 권한 부여에서 권한 부여 제약을 사용할 수 있습니다. KMS 키 부여 전파에는 최대 5분이 걸릴 수 있습니다. 자세한 내용은 을 참조하십시오. CreateGrant

Secrets Manager에서 KMS 키를 사용하는 방법

Secrets Manager는 KMS 키를 사용하여 다음과 같은 AWS KMS 작업을 호출합니다.

GenerateDataKey

AWS KMS GenerateDataKeySecrets Manager는 다음과 같은 Secrets Manager 작업에 대한 응답으로 작업을 호출합니다.

- CreateSecret— 새 암호에 암호 값이 포함된 경우 Secrets Manager는 새 데이터 키를 요청하여 암호화합니다.
- PutSecretValue— Secrets Manager는 지정된 비밀 값을 암호화하기 위해 새 데이터 키를 요청합니다.
- ReplicateSecretToRegions— 복제된 암호를 암호화하기 위해 Secrets Manager는 복제본 리전의 KMS 키에 대한 데이터 키를 요청합니다.
- UpdateSecret— 암호 값이나 KMS 키를 변경하는 경우 Secrets Manager는 새 데이터 키를 요청하여 새 암호 값을 암호화합니다.

이 RotateSecret작업은 비밀 값을 변경하지 않기 때문에 `GenerateDataKey` 호출되지 않습니다. 하지만 `RotateSecret`에서 호출하는 Lambda 함수가 보안 암호 값을 변경하는 경우 `PutSecretValue` 작업에 대한 호출로 `GenerateDataKey` 요청이 트리거됩니다.

Decrypt

Secrets Manager는 다음 Secrets Manager 작업에 대한 응답으로 Decrypt 작업을 호출합니다.

- GetSecretValueand BatchGetSecretValue— Secrets Manager는 비밀 값을 호출자에게 반환하기 전에 암호를 해독합니다. 암호화된 비밀 값을 해독하기 위해 Secrets Manager는 AWS KMS 암호 해독 작업을 호출하여 암호에 있는 암호화된 데이터 키를 복호화합니다. 그런 다음, 일반 텍스트 데이터 키를 사용하여 암호화된 암호 값을 해독합니다. 배치 명령의 경우 Secrets Manager는 해독된 키를 재사용할 수 있으므로 일부 호출에서만 `Decrypt` 요청이 발생합니다.
- PutSecretValue그리고 UpdateSecret— 대부분의 NAS `UpdateSecret` 요청은 작업을 `PutSecretValue` 트리거하지 않습니다. `Decrypt` 하지만 `PutSecretValue` 또는 `UpdateSecret` 요청이 보안 암호의 기존 버전에서 보안 암호 값을 변경하려 할 경우 Secrets Manager는 기존 보안 암호 값을 복호화하고 요청의 보안 암호 값과 비교하여 동일한지 확인합니다. 이 작업을 통해 Secrets Manager 작업의 idempotent가 유지됩니다. 암호화된 비밀 값을 해독하기 위해 Secrets Manager는 AWS KMS 암호 해독 작업을 호출하여 암호에 있는 암호화된 데이터 키를 복호화합니다. 그런 다음, 일반 텍스트 데이터 키를 사용하여 암호화된 암호 값을 해독합니다.
- ReplicateSecretToRegions— Secrets Manager는 복제본 리전의 KMS 키를 사용하여 보안 값을 다시 암호화하기 전에 먼저 기본 리전의 보안 값을 복호화합니다.

암호화

Secrets Manager는 다음 Secrets Manager 작업에 대한 응답으로 Encrypt 작업을 호출합니다.

- UpdateSecret— KMS 키를 변경하면 Secrets Manager는 AWSCURRENTAWSPREVIOUS, 및 AWSPENDING 비밀 버전을 보호하는 데이터 키를 새 키로 다시 암호화합니다.
- ReplicateSecretToRegions— Secrets Manager는 복제본 리전의 KMS 키를 사용하여 복제 중에 데이터 키를 다시 암호화합니다.

DescribeKey

Secrets Manager는 사용자가 Secrets Manager 콘솔에서 시크릿을 생성하거나 편집할 때 KMS 키를 나열할지 여부를 결정하는 DescribeKey작업을 호출합니다.

KMS 키에 대한 액세스 검증

보안 암호와 연결된 KMS 키를 설정하거나 변경할 때 Secrets Manager는 지정된 KMS 키를 사용하여 GenerateDataKey 및 Decrypt 작업을 호출합니다. 이러한 호출은 발신자가 해당 작업에 대해 KMS 키를 사용할 수 있는 권한이 있는지 확인합니다. Secrets Manager는 이러한 작업의 결과를 무시하며, 암호화된 작업에서 사용하지 않습니다.

이러한 요청에서는 SecretVersionId 키 암호화 컨텍스트 값이 RequestToValidateKeyAccess이므로 이와 같은 검증 호출을 식별할 수 있습니다.

> **Note**
>
> 예전에는 Secrets Manager 검증 호출에 암호화 컨텍스트가 포함되지 않았습니다. 이전 AWS CloudTrail 로그에서 암호화 컨텍스트가 없는 통화를 찾을 수 있습니다.

AWS 관리형 키 (aws/secretsmanager)의 키 정책

Secrets Manager (aws/secretsmanager) 의 AWS 관리형 키 키 정책은 Secrets Manager가 사용자를 대신하여 요청하는 경우에만 지정된 작업에 KMS 키를 사용할 권한을 사용자에게 부여합니다. 키 정책에서는 사용자가 KMS 키를 직접 사용하도록 허용하지 않습니다.

이 키 정책은 모든 AWS 관리형 키의 정책처럼 서비스에 의해 설정됩니다. 키 정책은 변경할 수 없지만 언제든지 볼 수 있습니다. 자세한 내용은 키 정책 보기를 참조하세요.

키 정책의 정책 설명문은 다음 효과를 갖습니다.

- 계정 내 사용자가 Secrets Manager를 통해 대신 요청할 때만 암호화 작업에 대해 KMS 키를 사용할 수 있도록 합니다. `kms:ViaService` 조건 키는 이 제한을 강제 적용합니다.
- 사용자가 KMS 키 속성을 보고 권한 부여를 취소할 수 있도록 허용하는 IAM 정책을 해당 AWS 계정에서 생성할 수 있도록 허용합니다.
- Secrets Manager는 권한 부여를 사용하여 KMS 키에 대한 액세스 권한을 얻지는 않지만 정책은 Secrets Manager로 하여금 사용자를 대신하여 KMS 키에 대한 권한 부여를 생성하도록 하고 계정으로 하여금 Secrets Manager가 KMS 키를 사용하도록 허용하는 모든 권한 부여를 취소하도록 합니다. 이러한 정책은 AWS 관리형 키용 정책 문서의 표준 요소입니다.

다음은 Secrets Manager의 예제를 AWS 관리형 키 위한 주요 정책입니다.

```
{
  "Id": "auto-secretsmanager-2",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Allow access through AWS Secrets Manager for all principals in the
 account that are authorized to use AWS Secrets Manager",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "*"
        ]
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:CreateGrant",
        "kms:DescribeKey"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "111122223333",
          "kms:ViaService": "secretsmanager.us-west-2.amazonaws.com"
        }
      }
    },
    {
```

```
      "Sid": "Allow access through AWS Secrets Manager for all principals in the
  account that are authorized to use AWS Secrets Manager",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "*"
        ]
      },
      "Action": "kms:GenerateDataKey*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "111122223333"
        },
        "StringLike": {
          "kms:ViaService": "secretsmanager.us-west-2.amazonaws.com"
        }
      }
    },
    {
      "Sid": "Allow direct access to key metadata to the account",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:root"
        ]
      },
      "Action": [
        "kms:Describe*",
        "kms:Get*",
        "kms:List*",
        "kms:RevokeGrant"
      ],
      "Resource": "*"
    }
  ]
}
```

Secrets Manager 보안 암호 컨텍스트

암호화 컨텍스트는 보안되지 않은 임의의 데이터를 포함하는 키-값 페어 세트입니다. 데이터 암호화 요청에 암호화 컨텍스트를 포함하면 암호화 컨텍스트가 암호화된 데이터에 AWS KMS 암호화된 방식으로 바인딩됩니다. 따라서 동일한 암호화 컨텍스트로 전달해야 이 데이터를 해독할 수 있습니다.

Secrets Manager는 다음 GenerateDataKey예와 같이 Secrets Manager와 암호 해독 요청에서 암호와 버전을 식별하는 두 개의 이름-값 쌍이 있는 암호화 컨텍스트를 사용합니다. AWS KMS이름은 달라지지 않지만, 결합된 암호화 컨텍스트 값은 각 암호 값마다 다릅니다.

```
"encryptionContext": {
    "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-
a1b2c3",
    "SecretVersionId": "EXAMPLE1-90ab-cdef-fedc-ba987SECRET1"
}
```

암호화 컨텍스트를 사용하여 감사 기록 및 로그 (예: Amazon CloudWatch Logs) 에서 이러한 암호화 작업을 식별하고 정책 및 권한 부여의 승인 조건으로 사용할 수 있습니다. AWS CloudTrail

Secrets Manager 암호화 컨텍스트는 이름-값 페어 두 개로 구성됩니다.

- SecretARN - 첫 번째 이름-값 페어는 보안 암호를 식별합니다. 키는 SecretARN입니다. 이 값은 암호의 Amazon 리소스 이름(ARN)입니다.

  ```
  "SecretARN": "ARN of an Secrets Manager secret"
  ```

  예를 들어, 보안 암호 ARN이 arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-a1b2c3이면 암호화 컨텍스트는 다음 페어를 포함합니다.

  ```
  "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-
  a1b2c3"
  ```

- SecretVersionId— 두 번째 이름-값 쌍은 암호의 버전을 식별합니다. 키는 SecretVersionId입니다. 이 값은 버전 ID입니다.

  ```
  "SecretVersionId": "<version-id>"
  ```

  예를 들어, 보안 암호의 버전 ID가 EXAMPLE1-90ab-cdef-fedc-ba987SECRET1이면 암호화 컨텍스트에는 다음 페어가 포함됩니다.

  ```
  "SecretVersionId": "EXAMPLE1-90ab-cdef-fedc-ba987SECRET1"
  ```

암호에 대한 KMS 키를 설정하거나 변경하면 Secrets Manager는 요청을 GenerateDataKey보내고 암호 AWS KMS 해독하여 호출자가 이러한 작업에 KMS 키를 사용할 권한이 있는지 확인합니다. 응답을 무시하므로 암호 값에 응답을 사용하지 않습니다.

이러한 검증 요청에서 SecretARN의 값은 암호의 실제 ARN이지만, SecretVersionId 값은 다음 예시 암호화 컨텍스트에서 나와 있는 것처럼 RequestToValidateKeyAccess입니다. 이 특수 값은 로그 및 감사 내역에서 검증 요청을 식별하는 데 도움이 됩니다.

```
"encryptionContext": {
    "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-
a1b2c3",
    "SecretVersionId": "RequestToValidateKeyAccess"
}
```

> **Note**
>
> 예전에는 Secrets Manager 검증 호출에 암호화 컨텍스트가 포함되지 않았습니다. 이전 로그에서 암호화 컨텍스트가 없는 통화를 찾을 수 있습니다. AWS CloudTrail

Secrets Manager와의 상호 작용을 모니터링하십시오. AWS KMS

Amazon CloudWatch Logs를 사용하여 AWS CloudTrail Secrets Manager가 사용자를 AWS KMS 대신하여 보내는 요청을 추적할 수 있습니다. 보안 암호 사용 모니터링에 대한 자세한 내용은 보안 암호 모니터링 섹션을 참조하세요.

GenerateDataKey

시크릿에서 시크릿 값을 생성하거나 변경하면 Secrets Manager는 시크릿의 KMS 키를 AWS KMS 지정하는 GenerateDataKey요청을 보냅니다.

GenerateDataKey 작업을 기록하는 이벤트는 다음 예시 이벤트와 유사합니다. 이 요청은 secretsmanager.amazonaws.com에 의해 호출됩니다. 파라미터로는 보안 암호용 KMS 키의 Amazon 리소스 이름(ARN), 256비트 키를 요구하는 키 지정자, 그리고 보안 암호와 버전을 식별하는 암호화 컨텍스트가 있습니다.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
```

```
        "principalId": "AROAIGDTESTANDEXAMPLE:user01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/user01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-05-31T23:23:41Z"
            }
        },
        "invokedBy": "secretsmanager.amazonaws.com"
    },
    "eventTime": "2018-05-31T23:23:41Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "secretsmanager.amazonaws.com",
    "userAgent": "secretsmanager.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
        "keySpec": "AES_256",
        "encryptionContext": {
            "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-
secret-a1b2c3",
            "SecretVersionId": "EXAMPLE1-90ab-cdef-fedc-ba987SECRET1"
        }
    },
    "responseElements": null,
    "requestID": "a7d4dd6f-6529-11e8-9881-67744a270888",
    "eventID": "af7476b6-62d7-42c2-bc02-5ce86c21ed36",
    "readOnly": true,
    "resources": [
        {
            "ARN": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "accountId": "111122223333",
            "type": "AWS::KMS::Key"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

Decrypt

암호의 보안 값을 가져오거나 변경하면 Secrets Manager는 암호화된 데이터 키를 복호화하기 AWS KMS 위해 암호 해독 요청을 에 보냅니다. 배치 명령의 경우 Secrets Manager는 해독된 키를 재사용할 수 있으므로 일부 호출에서만 `Decrypt` 요청이 발생합니다.

`Decrypt` 작업을 기록하는 이벤트는 다음 예시 이벤트와 유사합니다. 사용자는 테이블에 액세스하는 AWS 계정의 보안 주체입니다. 매개변수에는 암호화된 테이블 키 (암호문 블럽) 와 테이블과 계정을 식별하는 암호화 컨텍스트가 포함됩니다. AWS AWS KMS 암호문에서 KMS 키의 ID를 가져옵니다.


```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AROAIGDTESTANDEXAMPLE:user01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/user01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-05-31T23:36:09Z"
            }
        },
        "invokedBy": "secretsmanager.amazonaws.com"
    },
    "eventTime": "2018-05-31T23:36:09Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "secretsmanager.amazonaws.com",
    "userAgent": "secretsmanager.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-
secret-a1b2c3",
            "SecretVersionId": "EXAMPLE1-90ab-cdef-fedc-ba987SECRET1"
        }
    },
    "responseElements": null,
    "requestID": "658c6a08-652b-11e8-a6d4-ffee2046048a",
    "eventID": "f333ec5c-7fc1-46b1-b985-cbda13719611",
```

```
    "readOnly": true,
    "resources": [
        {
            "ARN": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "accountId": "111122223333",
            "type": "AWS::KMS::Key"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

암호화

암호와 연결된 KMS 키를 변경하면 Secrets Manager는 새 키로 AWSCURRENTAWSPREVIOUS, 및 AWSPENDING 비밀 버전을 다시 암호화하라는 AWS KMS 암호화 요청을 에 보냅니다. 보안 암호를 다른 리전에 복제하는 경우에도 Secrets Manager가 AWS KMS로 암호화 요청을 보냅니다.

Encrypt 작업을 기록하는 이벤트는 다음 예시 이벤트와 유사합니다. 사용자는 테이블에 액세스하는 AWS 계정의 보안 주체입니다.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AROAIGDTESTANDEXAMPLE:user01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/user01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "creationDate": "2023-06-09T18:11:34Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "secretsmanager.amazonaws.com"
    },
    "eventTime": "2023-06-09T18:11:34Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Encrypt",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "secretsmanager.amazonaws.com",
```

```
    "userAgent": "secretsmanager.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-east-2:111122223333:key/EXAMPLE1-f1c8-4dce-8777-
aa071ddefdcc",
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "encryptionContext": {
            "SecretARN": "arn:aws:secretsmanager:us-
east-2:111122223333:secret:ChangeKeyTest-5yKnKS",
            "SecretVersionId": "EXAMPLE1-5c55-4d7c-9277-1b79a5e8bc50"
        }
    },
    "responseElements": null,
    "requestID": "129bd54c-1975-4c00-9b03-f79f90e61d60",
    "eventID": "f7d9ff39-15ab-47d8-b94c-56586de4ab68",
    "readOnly": true,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:111122223333:key/EXAMPLE1-f1c8-4dce-8777-
aa071ddefdcc"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


AWS Secrets Manager의 인프라 보안

관리형 서비스인 AWS Secrets Manager은(는) AWS 글로벌 네트워크 보안으로 보호됩니다. AWS 보안 서비스와 AWS의 인프라 보호 방법에 대한 자세한 내용은 AWS 클라우드 보안을 참조하세요. 인프라 보안에 대한 모범 사례를 사용하여 AWS 환경을 설계하려면 보안 원칙 AWS Well-Architected Framework의 인프라 보호를 참조하세요.

AWS TLS를 사용하여 게시된 API를 통한 네트워크를 이용하여 Secrets Manager에 액세스합니다. 네트워크 위치에서 Secrets Manage API를 호출할 수 있습니다. 하지만 Secrets Manager는 원본 IP 주소 기반의 제한을 포함할 수 있는 리소스 기반 액세스 정책을 지원합니다. Secrets Manager 리소스 정책을 사용하여 대한 액세스를 제어할 수도 있습니다.특정 Virtual Private Cloud(VPC) 엔드포인트 또는 특정 VPC의 암호에 대한 액세스를 제어할 수도 있습니다. 그러면 AWS 네트워크의 특정 VPC에서만 주

어진 암호에 대한 네트워크 액세스가 효과적으로 분리됩니다. 자세한 내용은 VPC 엔드포인트 섹션을 참조하세요.

내에서의 레질리언스 AWS Secrets Manager

AWS 가용 영역 AWS 리전 및 가용 영역을 중심으로 글로벌 인프라를 구축합니다. AWS 리전 물리적으로 분리되고 격리된 여러 가용 영역을 제공하여 지연 시간이 짧고 처리량이 높으며 중복성이 높은 네트워킹으로 연결됩니다. 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.

복원력 및 재해 복구에 대한 자세한 내용은 안정성 기둥 - Well-Architected AWS 프레임워크를 참조하십시오.

가용 영역 AWS 리전 및 가용 영역에 대한 자세한 내용은 글로벌 인프라를 참조하십시오.AWS

포스트 양자 TLS

Secrets Manager는 전송 계층 보안(TLS) 네트워크 암호화 프로토콜에 대한 하이브리드 포스트 퀀텀 키 교환 옵션을 지원합니다. Secrets Manager API 엔드포인트에 연결할 때 이 TLS 옵션을 사용할 수 있습니다. 포스트 양자 알고리즘이 표준화되기 전에 이 기능을 제공하므로 이러한 키 교환 프로토콜이 Secrets Manager 호출에 미치는 영향을 테스트할 수 있습니다. 선택 사항인 이 하이브리드 포스트 양자 키 교환 기능은 적어도 우리가 현재 사용하는 TLS 암호화만큼 안전하며 보안 이점을 추가로 제공할 수도 있습니다. 그러나 현재 사용 중인 클래식 키 교환 프로토콜과 달리 지연 시간 및 처리량에 영향을 미칩니다.

미래에 있을 수 있는 공격으로부터 오늘날 암호화된 데이터를 보호하기 위해 AWS는 암호화 커뮤니티와 함께 양자 내성 또는 포스트 양자 알고리즘 개발에 참여하고 있습니다. Secrets Manager 엔드포인트에서 하이브리드 포스트 양자 키 교환 암호 제품군을 구현했습니다. 클래식 및 포스트 양자 요소를 결합한 이 하이브리드 암호 제품군을 통해 TLS 연결은 적어도 클래식 암호 제품군과 동등한 수준으로 강력해집니다. 그러나 하이브리드 암호 제품군의 성능 특성 및 대역폭 요구 사항은 클래식 키 교환 메커니즘의 해당 요구 사항과 다르기 때문에 API 호출에 대해 이 제품군을 테스트하는 것이 좋습니다.

Secrets Manager는 중국 리전을 제외한 모든 리전에서 PQTLS를 지원합니다.

하이브리드 포스트 양자 TLS 구성

1. Maven 종속성에 AWS 공통 런타임 클라이언트를 추가합니다. 사용 가능한 최신 버전을 사용하는 것이 좋습니다. 예를 들어 이 문은 2.20.0 버전을 추가합니다.

```
<dependency>
  <groupId>software.amazon.awssdk</groupId>
  <artifactId>aws-crt-client</artifactId>
  <version>2.20.0</version>
</dependency>
```

2. Java 2.x용 AWS SDK를 프로젝트에 추가하고 초기화합니다. HTTP 클라이언트에서 하이브리드 포스트 양자 암호 제품군을 활성화합니다.

```
SdkAsyncHttpClient awsCrtHttpClient = AwsCrtAsyncHttpClient.builder()
            .postQuantumTlsEnabled(true)
            .build();
```

3. Secrets Manager 비동기 클라이언트를 생성합니다.

```
SecretsManagerAsyncClient SecretsManagerAsync = SecretsManagerAsyncClient.builder()
            .httpClient(awsCrtHttpClient)
            .build();
```

   이제 Secrets Manager API 작업을 호출하면 호출은 하이브리드 포스트 양자 TLS를 사용하여 Secrets Manager 엔드포인트로 전송됩니다.

하이브리드 포스트 양자 TLS 사용하는 방법에 대한 자세한 내용은 다음을 참조하십시오.

- AWS SDK for Java 2.x 개발자 안내서 및 AWS SDK for Java 2.x 릴리스된 블로그 게시물.
- 새 오픈 소스 TLS 구현, 즉 s2n-tls 소개 및 s2n-tls 사용.
- 미국 국립 표준 기술 연구소 (NIST)에서의 포스트 양자 암호화.
- TLS(전송 계층 보안) 1.2에 대한 PQ KEM(하이브리드 포스트 양자 키 캡슐화 방법).

Secrets Manager용 포스트 양자 TLS는 중국을 제외한 모든 AWS 리전에서 사용할 수 있습니다.

문제 해결 AWS Secrets Manager

여기 정보를 사용하여 Secrets Manager 작업 시 발생할 수 있는 문제를 진단하고 수정하세요.

교체와 관련된 문제는 the section called “ 교체 문제 해결” 섹션을 참조하세요.

주제

- '액세스 거부' 메시지
- 임시 보안 자격 증명에 대한 “액세스 거부됨”이라는 메시지 발생
- 변경 사항이 경우에 따라 즉시 표시되지 않습니다.
- 보안 암호를 생성할 때 “비대칭 KMS 키로 데이터 키를 생성할 수 없습니다.”라는 메시지 발생
- AWS CLI 또는 AWS SDK 작업이 부분 ARN에서 내 비밀을 찾을 수 없습니다.
- 이 비밀은 AWS 서비스에서 관리하므로 업데이트하려면 해당 서비스를 사용해야 합니다.

'액세스 거부' 메시지

Secrets Manager와 같은 GetSecretValue API 호출을 CreateSecret 할 때는 해당 호출을 수행할 수 있는 IAM 권한이 있어야 합니다. 콘솔을 사용하면 콘솔이 사용자를 대신하여 동일한 API 호출을 수행하므로 IAM 권한도 있어야 합니다. 관리자는 IAM 사용자 또는 사용자가 속한 그룹에 IAM 정책을 연결하여 권한을 부여할 수 있습니다. 이러한 권한을 부여하는 정책 설명에 IP 주소 제한과 time-of-day 같은 조건이 포함되어 있는 경우 요청을 보낼 때 해당 요구 사항도 충족해야 합니다. IAM 사용자, 그룹 또는 역할에 대한 정책을 확인 또는 수정하는 방법은 IAM 사용 설명서에 있는 정책 작업 섹션을 참조하세요. Secrets Manager에 필요한 권한에 대한 자세한 정보는 인증 및 액세스 제어 섹션을 참조하세요.

AWS SDK를 사용하지 않고 API 요청에 수동으로 서명할 경우 올바르게 요청에 서명했는지 확인합니다.

임시 보안 자격 증명에 대한 “액세스 거부됨”이라는 메시지 발생

요청을 위해 사용 중인 IAM 사용자나 역할에 올바른 권한이 있는지 확인합니다. 임시 보안 자격 증명에 대한 권한은 IAM 사용자 또는 역할에서 파생됩니다. 이는 권한이 IAM 사용자 또는 역할에 부여된 권한으로 제한됨을 의미합니다. 임시 보안 자격 증명의 권한이 결정되는 방법에 대한 자세한 내용은 IAM 사용 설명서의 임시 보안 자격 증명을 위한 권한 제어 섹션을 참조하세요.

요청에 올바르게 서명했고 요청이 잘 구성되었는지 확인합니다. 자세한 내용은 선택한 SDK의 툴킷 설명서 또는 IAM 사용 설명서의 임시 보안 자격 증명을 사용하여 AWS 리소스에 대한 액세스 요청을 참조하십시오.

임시 보안 자격 증명이 만료되지 않았는지 확인합니다. 자세한 내용은 IAM 사용 설명서의 임시 보안 자격 증명 요청 섹션을 참조하세요.

Secrets Manager에 필요한 권한에 대한 자세한 정보는 인증 및 액세스 제어 섹션을 참조하세요.

변경 사항이 경우에 따라 즉시 표시되지 않습니다.

Secrets Manager는 최종 일관성이라는 분산 컴퓨팅 모델을 사용합니다. Secrets Manager (또는 다른 AWS 서비스) 에서 변경한 내용을 가능한 모든 엔드포인트에서 볼 수 있으려면 시간이 걸립니다. 일부 지연은 서버에서 서버로, 복제 영역에서 복제 영역으로, 전세계의 리전에서 리전으로 데이터를 보내는 데 걸리는 시간으로 인해 발생합니다. 또한 Secrets Manager는 성능 향상을 위해 캐싱을 사용하지만, 어떤 경우에는 이로 인해 시간이 더 걸릴 수 있습니다. 이러한 변화는 이전에 캐싱된 데이터가 끝날 때까지 가시화되지 않을 수 있습니다.

이러한 잠재적 지연을 고려하도록 전역 애플리케이션을 설계합니다. 또한 한 위치에서 변경한 내용이 다른 위치에서 즉시 보이지 않을 때조차도 예상대로 작동하는지 확인합니다.

최종 일관성이 일부 다른 AWS 서비스에 미치는 영향에 대한 자세한 내용은 다음을 참조하십시오.

- Amazon Redshift 데이터베이스 개발자 가이드의 데이터 일관성 관리
- Amazon Simple Storage Service 사용 설명서의 Amazon S3 데이터 일관성 모델
- AWS 빅 데이터 블로그에서 ETL 워크플로에 대해 Amazon S3 및 Amazon EMR 사용 시 일관성 유지
- Amazon EC2 API 참조의 Amazon EC2 최종 일관성

보안 암호를 생성할 때 “비대칭 KMS 키로 데이터 키를 생성할 수 없습니다.”라는 메시지 발생

Secrets Manager는 보안 암호와 연결된 대칭 암호화 KMS 키를 사용하여 각 보안 암호 값에 대한 데이터 키를 생성합니다. 비대칭 KMS 키를 사용할 수 없습니다. 비대칭 KMS 키 대신 대칭 암호화 KMS 키를 사용하고 있는지 확인하세요. 지침은 비대칭 KMS 키 식별 단원을 참조하세요.

AWS CLI 또는 AWS SDK 작업이 부분 ARN에서 내 비밀을 찾을 수 없습니다.

많은 경우 Secrets Manager는 전체 ARN이 아닌 ARN의 일부에서 보안 암호를 찾을 수 있습니다. 그러나, 보안 암호 이름이 하이픈 다음에 6자로 끝나는 경우 Secrets Manager는 ARN의 일부에서만 보안 암호를 찾지 못할 수 있습니다. 그 대신 전체 ARN 또는 암호 이름을 사용할 것을 권장합니다.

추가 세부 정보

Secrets Manager는 보안 암호 ARN이 고유한지 확인하기 위해 보안 암호 이름의 끝에 임의의 문자 6개를 포함합니다. 원래 보안 암호를 삭제한 후 동일한 이름으로 새 보안 암호를 생성하면 해당 문자로 인해 두 보안 암호의 ARN이 달라집니다. ARN이 다르기 때문에 이전 보안 암호에 액세스할 수 있는 사용자는 새 보안 암호에 자동으로 액세스할 수 없습니다.

Secrets Manager는 다음과 같이 리전, 계정, 보안 암호 이름, 하이픈 및 추가 6자를 사용하여 보안 암호에 대한 ARN을 구성합니다.

```
arn:aws:secretsmanager:us-east-2:111122223333:secret:SecretName-abcdef
```

보안 암호 이름이 하이픈과 6자로 끝나는 경우 ARN의 일부만 사용하면 Secrets Manager에 전체 ARN을 지정하는 것처럼 보일 수 있습니다. 예를 들어 ARN과 함께 이름이 `MySecret-abcdef`인 보안 암호가 있을 수 있습니다.

```
arn:aws:secretsmanager:us-east-2:111122223333:secret:MySecret-abcdef-nutBrk
```

보안 암호 ARN의 일부만 사용하는 다음 작업을 호출할 경우, Secret Manager가 해당 보안 암호를 찾지 못할 수 있습니다.

```
$ aws secretsmanager describe-secret --secret-id arn:aws:secretsmanager:us-
east-2:111122223333:secret:MySecret-abcdef
```

이 비밀은 AWS 서비스에서 관리하므로 업데이트하려면 해당 서비스를 사용해야 합니다.

보안 암호를 수정하려고 할 때 이 메시지가 나타나면 메시지에 나열된 관리 서비스를 사용해야만 보안 암호를 업데이트할 수 있습니다. 자세한 정보는 관리형 시크릿을 참조하세요.

보안 암호 관리자를 결정하기 위해 보안 암호 이름을 검토할 수 있습니다. 다른 서비스에서 관리하는 보안 암호는 해당 서비스의 ID가 접두사로 붙습니다. 또는 에서 describe-secret을 호출한 다음 필드를 검토하십시오. AWS CLI`OwningService`

AWS Secrets Manager 할당량

Secrets Manager 읽기 API는 TPS 할당량이 높고, 호출 빈도가 낮은 컨트롤 플레인 API는 TPS 할당량이 낮습니다. 10분마다 두 번 이상 지속적으로 `PutSecretValue` 또는 `UpdateSecret`을 호출하지 않는 것이 좋습니다. `PutSecretValue` 또는 `UpdateSecret`을 호출하여 보안 암호 값을 업데이트하면 Secrets Manager는 새 버전의 보안 암호를 생성합니다. Secrets Manager는 100개를 넘는 버전이 있을 때 레이블이 지정되지 않은 버전을 제거하지만 24시간 이내에 생성된 버전은 제거하지 않습니다. 10분마다 두 번 이상 보안 암호 값을 업데이트하면 Secrets Manager에서 제거하는 버전보다 더 많은 버전이 생성되고 보안 암호 버전 할당량에 도달하게 됩니다.

계정에서 여러 리전을 운영할 수 있으며 각 할당량은 각 리전에 따라 다릅니다.

한 AWS 계정의 한 애플리케이션이 다른 계정 소유의 보안 암호를 사용하는 경우를 교차 계정 요청이라고 합니다. 교차 계정 요청의 경우, Secrets Manager는 보안 암호를 소유한 계정이 아니라 요청하는 자격 증명의 계정을 제한합니다. 예를 들어 계정 A의 자격 증명이 계정 B의 보안 암호를 사용하는 경우 보안 암호 사용은 계정 A의 할당량에만 적용됩니다.

Secrets Manager 할당량

이름	기본값	조정 가능	설명
DeleteResourcePolicy, GetResourcePolicy, PutResourcePolicy, ValidateResourcePolicy API 요청의 합산 비율	각각 지원되는 리전: 초당 50개	아니오	DeleteResourcePolicy, GetResourcePolicy, PutResourcePolicy, ValidateResourcePolicy API 요청의 합산에 대한 초당 최대 트랜잭션 수입니다.
DescribeSecret 및 GetSecretValue API 요청의 합산 비율	각각 지원되는 리전: 초당 10,000개	아니오	DescribeSecret 및 GetSecretValue API 요청의 합산에 대한 초당 최대 트랜잭션 수입니다.

이름	기본값	조정 가능	설명
PutSecretValue, RemoveRegionsFromReplication, ReplicateSecretToRegion, StopReplicationToReplica, UpdateSecret, UpdateSecretVersionStage API 요청의 합산 비율	각각 지원되는 리전: 초당 50개	아니오	PutSecretValue, RemoveRegionsFromReplication, ReplicateSecretToRegion, StopReplicationToReplica, UpdateSecret, UpdateSecretVersionStage API 요청의 합산에 대한 초당 최대 트랜잭션 수입니다.
RestoreSecret API 요청의 합산 비율	각각 지원되는 리전: 초당 50개	아니오	RestoreSecret API 요청에 대한 초당 최대 트랜잭션 수입니다.
RotateSecret 및 CancelRotateSecret API 요청의 합산 비율	각각 지원되는 리전: 초당 50개	아니오	RotateSecret 및 CancelRoteSecret API 요청의 합산에 대한 초당 최대 트랜잭션 수입니다.
TagResource 및 UntagResource API 요청의 합산 비율	각각 지원되는 리전: 초당 50개	아니오	TagResource 및 UntagResource API 요청의 합산에 대한 초당 최대 트랜잭션 수입니다.
BatchGetSecretValue API 요청 비율	지원되는 리전별: 초당 100개	아니오	BatchGetSecretValue API 요청에 대한 초당 최대 트랜잭션 수입니다.
CreateSecret API 요청 비율	각각 지원되는 리전: 초당 50개	아니오	CreateSecret API 요청에 대한 초당 최대 트랜잭션 수입니다.

이름	기본값	조정 가능	설명
DeleteSecret API 요청 비율	각각 지원되는 리전: 초당 50개	아니오	DeleteSecret API 요청에 대한 초당 최대 트랜잭션 수입니다.
GetRandomPassword API 요청 비율	각각 지원되는 리전: 초당 50개	아니오	GetRandomPassword API 요청에 대한 초당 최대 트랜잭션 수입니다.
ListSecretVersionIds API 요청 비율	각각 지원되는 리전: 초당 50개	아니오	ListSecretVersionIds API 요청에 대한 초당 최대 트랜잭션 수입니다.
ListSecrets API 요청 비율	지원되는 리전별: 초당 100개	아니오	ListSecrets API 요청에 대한 초당 최대 트랜잭션 수입니다.
리소스 기반 정책 길이	각각 지원되는 리전: 20,480	아니오	보안 암호에 연결된 리소스 기반 권한 정책의 최대 문자 수입니다.
보안 암호 값 크기	각각 지원되는 리전: 65,536바이트	아니오	암호화된 보안 암호 값의 최대 크기입니다. 암호 값이 문자열인 경우 이는 암호 값에 허용되는 문자 수입니다.
암호	각각 지원되는 리전: 500,000	아니오	이 AWS 계정의 각 AWS 리전에 있는 보안 암호의 최대 수입니다.
보안 암호의 모든 버전에 연결된 스테이징 레이블	지원되는 각 리전: 20	아니오	보안 암호의 모든 버전에 연결된 스테이징 레이블의 최대 수입니다.

이름	기본값	조정 가능	설명
보안 암호별 버전	각 지원되는 리전: 100	아니오	보안 암호의 최대 버전 수입니다.

애플리케이션에 재시도 추가

AWS 클라이언트에서는 클라이언트 측에서 예기치 않은 문제로 인해 Secrets Manager 호출이 실패하는 것을 확인할 수 있습니다. 또는 Secrets Manager의 속도 제한으로 인해 호출이 실패할 수 있습니다. API 요청 할당량을 초과하면 Secrets Manager에서 요청을 제한합니다. Secrets Manager는 다른 경우라면 유효한 요청을 거부하고 throttling 오류를 반환합니다. 두 유형의 실패 모두 짧은 대기 기간 후에 다시 호출하는 것이 좋습니다. 이것을 백오프 및 재시도 전략이라고 부릅니다.

다음과 같은 오류가 발생할 경우 애플리케이션 코드에 재시도를 추가할 수 있습니다.

일시적 오류 및 예외

- `RequestTimeout`
- `RequestTimeoutException`
- `PriorRequestNotComplete`
- `ConnectionError`
- `HTTPClientError`

서비스 측 조절 및 제한 오류와 예외

- `Throttling`
- `ThrottlingException`
- `ThrottledException`
- `RequestThrottledException`
- `TooManyRequestsException`
- `ProvisionedThroughputExceededException`

- `TransactionInProgressException`
- `RequestLimitExceeded`
- `BandwidthLimitExceeded`
- `LimitExceededException`
- `RequestThrottled`
- `SlowDown`

재시도, 지수 백오프 및 지터에 대한 자세한 내용 및 예시 코드는 다음 리소스를 참조하세요.

- 지수 백오프 및 지터
- 시간 초과, 지터를 사용한 재시도 및 백오프
- AWS의 오류 재시도 횟수 및 지수 백오프.

문서 기록

다음 표에서는 의 마지막 릴리스 이후 이 설명서에서 변경된 주요 내용을 설명합니다 AWS Secrets Manager. 이 설명서에 대한 업데이트 알림을 받으려면 RSS 피드를 구독하면 됩니다.

변경 사항	설명	날짜
Secrets Manager가 AWS 관리형 정책으로 변경	이제 `SecretsManagerReadWrite` 관리형 정책에 `redshift-serverless` 권한이 포함됩니다. 자세한 내용은 다음 AWS 관리형 정책을 참조하십시오. AWS Secrets Manager	2024년 3월 12일

이전 업데이트

다음 표에는 2024년 2월 이전 AWS Secrets Manager User Guide의 각 릴리스에서 변경된 주요 내용이 설명되어 있습니다.

변경 사항	설명	날짜
정식 출시	이것은 Secrets Manager의 첫 공개 릴리스입니다.	2018년 4월 4일

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.