aws

사용자 가이드

EventBridge 스케줄러

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

EventBridge 스케줄러: 사용자 가이드

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

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

Table of Contents

Amazon EventBridge 스케줄러란 무엇인가요?

Amazon EventBridge 스케줄러는 하나의 중앙 관리형 서비스에서 작업을 생성, 실행 및 관리할 수 있는 서버리스 스케줄러입니다. 확장성이 뛰어난 EventBridge 스케줄러를 사용하면 270개 이상의 AWS 서비스와 6,000개 이상의 API 작업을 호출할 수 있는 수백만 개의 작업을 예약할 수 있습니다. 인프라를 프로비저닝 및 관리하거나 여러 서비스와 통합할 필요 없이 EventBridge 스케줄러를 사용하면 대규모로 일정을 제공하고 유지 관리 비용을 줄일 수 있습니다.

EventBridge 스케줄러는 다운스트림 대상의 가용성에 따라 일정을 조정하는 내장 메커니즘을 통해 작업을 안정적으로 제공합니다. EventBridge 스케줄러를 사용하면 반복 패턴에 대해 cron 및 rate 표현식을 사용하여 일정을 만들거나 일회성 간접 호출을 구성할 수 있습니다. 전송을 위한 유연한 기간을 설정하고, 재시도 제한을 정의하고, 실패한 트리거의 최대 보존 시간을 설정할 수 있습니다.

주제

EventBridge 스케줄러의 주요 기능

EventBridge 스케줄러는 대상을 구성하고 일정을 조정하는 데 사용할 수 있는 다음과 같은 주요 기능을 제공합니다.

- 템플릿 기반 대상 — EventBridge 스케줄러는 Amazon SQS, Amazon SNS, Lambda 및 EventBridge를 사용하여 일반적인 API 작업을 수행할 수 있도록 템플릿 기반 대상을 지원합니다. 사전 정의된 대상이 있으면 EventBridge 스케줄러 콘솔, EventBridge 스케줄러 SDK 또는 AWS CLI를 사용하여 일정을 빠르게 구성할 수 있습니다.
- 범용 대상 - EventBridge 스케줄러는 일정에 따라 270개 이상의 AWS 서비스와 6,000개 이상의 API 작업을 대상으로 하는 사용자 지정 트리거를 생성하는 데 사용할 수 있는 범용 대상 파라미터(UTP)를 제공합니다. UTP를 사용하면 EventBridge 스케줄러 콘솔, EventBridge 스케줄러 SDK 또는 AWS CLI를 사용하여 사용자 지정 트리거를 구성할 수 있습니다.
- 유연한 기간 - EventBridge 스케줄러는 유연한 기간을 지원하므로 정확한 예정된 대상 호출이 필요하지 않은 사용 사례에 대해 일정을 분산하고 트리거의 안정성을 개선할 수 있습니다.
- 재시도 - EventBridge 스케줄러는 대상에 최소 한 번 이상의 이벤트 전송을 제공합니다. 즉, 최소 한 번 이상의 전송이 대상의 응답으로 성공하다는 의미입니다. EventBridge 스케줄러를 사용하면 실패한 작업에 대한 일정 재시도 횟수를 설정할 수 있습니다. EventBridge 스케줄러는 지연된 시도가 있더라도 실패한 작업을 재시도하여 일정의 안정성을 개선하고 대상을 사용할 수 있도록 합니다.

EventBridge 스케줄러 액세스

EventBridge 스케줄러 콘솔, EventBridge 스케줄러 SDK, AWS CLI를 통해, 또는 EventBridge 스케줄러 API를 직접 사용하여 EventBridge 스케줄러를 사용할 수 있습니다.

아마존 EventBridge 스케줄러 설정

EventBridge 스케줄러를 사용하려면 먼저 다음 단계를 완료해야 합니다.

주제

등록하기 AWS

계정이 없는 경우 다음 단계를 완료하여 계정을 만드세요. AWS 계정

가입하려면 AWS 계정

1. https://portal.aws.amazon.com/billing/signup을 여세요.
2. 온라인 지시 사항을 따르세요.

   등록 절차 중에는 전화를 받고 키패드로 인증 코드를 입력하는 과정이 있습니다.

   에 AWS 계정가입하면 AWS 계정 루트 사용자a가 생성됩니다. 루트 사용자에게는 계정의 모든 AWS 서비스 및 리소스 액세스 권한이 있습니다. 보안 모범 사례는 사용자에게 관리 액세스 권한을 할당하고, 루트 사용자만 사용하여 루트 사용자 액세스 권한이 필요한 작업을 수행하는 것입니다.

IAM 사용자를 생성합니다.

다음 옵션 중 하나를 선택하여 관리 사용자를 생성합니다.

관리자를 관리하는 방법 한 가지 선택	목적	By	다른 방법
IAM Identity Center에서 (권장)	단기 보안 인증 정보를 사용하여 AWS에 액세스합니다. 이는 보안 모범 사례와 일치합니다. 모범 사례에 대한 자세한 내용은 IAM 사용 설명서의 IAM 보안 모범 사례를 참조하세요.	AWS IAM Identity Center 사용 설명서의 시작하기 지침을 따르세요.	사용 AWS IAM Identity CenterAWS Command Line Interface 설명서에서 사용하도록 AWS CLI 구성하여 프로그래밍 액세스를 구성하십시오.
IAM에서 (권장되지 않음)	장기 보안 인증 정보를 사용하여 AWS에 액세스합니다.	IAM 사용 설명서의 첫 IAM 관리 사용자 및 사용자 그룹 만들기에 나온 지침을 따릅니다.	IAM 사용 설명서에 나온 IAM 사용자의 액세스 키 관리 단계를 수행하여 프로그래밍 방식의 액세스를 구성합니다.

관리형 정책 사용

이전 단계에서는 자격 증명을 사용하여 IAM 사용자를 설정하여 리소스에 액세스할 수 AWS 있도록 했습니다. 대부분의 경우 EventBridge Scheduler를 안전하게 사용하려면 Scheduler를 사용하는 EventBridge 데 필요한 권한만 있는 별도의 사용자, 그룹 또는 역할을 생성하는 것이 좋습니다. EventBridge 스케줄러는 일반적인 사용 사례에 대해 다음과 같은 관리형 정책을 지원합니다.

- the section called “AmazonEventBridgeSchedulerFullAccess”— 콘솔과 API를 사용하여 EventBridge 스케줄러에 대한 전체 액세스 권한을 부여합니다.
- the section called “AmazonEventBridgeSchedulerReadOnlyAccess”— 스케줄러에 대한 읽기 전용 액세스 권한을 EventBridge 부여합니다.

이전 단계에서 AdministratorAccess 정책을 연결한 것과 같은 방식으로 이러한 관리형 정책을 IAM 보안 주체에 연결할 수 있습니다. ID 기반 IAM 정책을 사용하여 EventBridge Scheduler에 대한 액

세스를 관리하는 방법에 대한 자세한 내용은 을 참조하십시오. the section called “자격 증명 기반 정책 사용”

실행 역할 설정

실행 역할은 사용자를 대신하여 다른 사용자와 상호 작용하기 위해 EventBridge Scheduler가 맡는 IAM 역할입니다. AWS 서비스 이 역할에 권한 정책을 추가하여 EventBridge 스케줄러에게 호출 대상을 위한 액세스 권한을 부여합니다.

콘솔을 사용하여 새 일정을 생성할 때 새 실행 역할을 생성할 수도 있습니다. 콘솔을 사용하는 경우 EventBridge Scheduler는 선택한 대상에 따라 권한을 가진 역할을 사용자 대신 생성합니다. EventBridge Scheduler가 사용자를 대신하여 역할을 생성하는 경우 역할의 신뢰 정책에는 사용자를 대신하여 역할을 수임할 수 있는 주도자를 제한하는 조건 키가 포함됩니다. 이렇게 하면 혼동될 수 있는 대리인 보안 문제를 방지할 수 있습니다.

다음 단계는 새 실행 역할을 만드는 방법과 EventBridge 스케줄러에 대상을 호출할 수 있는 액세스 권한을 부여하는 방법을 설명합니다. 이 항목에서는 일반적인 템플릿 기반 대상에 대한 권한을 설명합니다. 다른 대상에 대한 권한 추가에 대한 자세한 내용은 the section called “템플릿 기반 대상 사용” 섹션을 참조하세요.

를 사용하여 실행 역할을 만들려면 AWS CLI

1. 다음 역할 수임 JSON 정책을 복사하고 로컬에 `Scheduler-Execution-Role.json`로 저장하세요. 이 신뢰 정책을 통해 EventBridge Scheduler는 사용자를 대신하여 역할을 맡을 수 있습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "scheduler.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

> **Important**
>
> 프로덕션 환경에서 실행 역할을 설정하려면 혼동되는 부수적인 문제를 방지하기 위한 추가 보호 조치를 구현하는 것이 좋습니다. 자세한 내용과 예제 정책은 the section called “혼동된 대리자 방지” 섹션을 참조하세요.

2. AWS Command Line Interface (AWS CLI) 에서 다음 명령을 입력하여 새 역할을 생성합니다. *SchedulerExecutionRole*을 이 역할에 부여하려는 이름으로 바꾸세요.

```
$ aws iam create-role --role-name SchedulerExecutionRole --assume-role-policy-
document file://Scheduler-Execution-Role.json
```

성공하면 다음과 같은 결과가 출력됩니다.

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Scheduler-Execution-Role",
        "RoleId": "BR1L2DZK3K4CTL5ZF9EIL",
        "Arn": "arn:aws:iam::123456789012:role/SchedulerExecutionRole",
        "CreateDate": "2022-03-10T18:45:01+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "scheduler.amazonaws.com"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        }
    }
}
```

3. EventBridge Scheduler가 대상을 호출하도록 허용하는 새 정책을 만들려면 다음 공통 대상 중 하나를 선택합니다. JSON 권한 정책을 복사하여 로컬에 `.json` 파일로 저장합니다.

Amazon SQS – SendMessage

다음을 통해 EventBridge Scheduler는 계정의 모든 Amazon SQS 대기열에서 `sqs:SendMessage` 작업을 호출할 수 있습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "sqs:SendMessage"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

Amazon SNS – Publish

다음은 EventBridge Scheduler가 사용자 계정의 모든 Amazon SNS 주제에 대해 `sns:Publish` 작업을 호출할 수 있도록 하는 방법입니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "sns:Publish"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

Lambda – Invoke

다음을 통해 EventBridge 스케줄러는 계정의 모든 Lambda 함수에서 `lambda:InvokeFunction` 작업을 호출할 수 있습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

4. 다음 명령을 실행하여 새 권한 정책을 생성합니다. *PolicyName*를 이 정책에 부여하려는 이름으로 바꾸세요.

```
$ aws iam create-policy --policy-name PolicyName --policy-document file://
PermissionPolicy.json
```

성공하면 다음과 같은 결과가 출력됩니다. 정책 ARN을 기록합니다. 다음 단계에서 이 ARN을 사용하여 정책을 실행 역할에 연결합니다.

```
{
    "Policy": {
        "PolicyName": "PolicyName",
        "CreateDate": "2022-03-015T19:31:18.620Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "ZXR6A36LTYANPAI7NJ5UV",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:policy/PolicyName",
        "UpdateDate": "2022-03-015T19:31:18.620Z"
    }
}
```

5. 실행 역할에 정책을 연결하려면 다음 명령을 실행하세요. *your-policy-arn*을 이전 단계에서 생성한 정책의 ARN으로 변경합니다. *SchedulerExecutionRole*을 실행 역할의 이름으로 바꿉니다.

```
$ aws iam attach-role-policy --policy-arn your-policy-arn --role-
name SchedulerExecutionRole
```

이 `attach-role-policy` 작업은 명령줄에서 응답을 반환하지 않습니다.

대상 설정

EventBridge 스케줄러 일정을 생성하기 전에 스케줄에서 호출할 대상이 하나 이상 있어야 합니다. 기존 AWS 리소스를 사용하거나 새 리소스를 만들 수 있습니다. 다음 단계는 를 사용하여 새로운 표준 Amazon SQS 대기열을 생성하는 방법을 보여줍니다. AWS CloudFormation

새로운 Amazon SQS 대기열을 생성하려면

1. 다음 JSON AWS CloudFormation 템플릿을 복사하여 로컬에 다른 이름으로 저장합니다. `SchedulerTargetSQS.json`

```
{
   "AWSTemplateFormatVersion": "2010-09-09",
   "Resources": {
      "MyQueue": {
         "Type": "AWS::SQS::Queue",
         "Properties": {
            "QueueName": "MyQueue"
             }
         }
      },
   "Outputs": {
      "QueueName": {
         "Description": "The name of the queue",
         "Value": {
            "Fn::GetAtt": [
               "MyQueue",
               "QueueName"
            ]
         }
      },
      "QueueURL": {
         "Description": "The URL of the queue",
         "Value": {
            "Ref": "MyQueue"
```

```
         }
      },
      "QueueARN": {
         "Description": "The ARN of the queue",
         "Value": {
            "Fn::GetAtt": [
               "MyQueue",
               "Arn"
            ]
         }
      }
   }
}
```

2. 에서 다음 AWS CLI명령을 실행하여 Scheduler-Target-SQS.json 템플릿에서 AWS CloudFormation 스택을 생성합니다.

```
$ aws cloudformation create-stack --stack-name Scheduler-Target-SQS --template-body
 file://Scheduler-Target-SQS.json
```

성공하면 다음과 같은 결과가 출력됩니다.

```
{
    "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/Scheduler-
Target-SQS/1d2af345-a121-12eb-abc1-012e34567890"
}
```

3. 다음 명령을 실행하여 AWS CloudFormation 스택에 대한 요약 정보를 확인합니다. 이 정보에는 스택의 상태와 템플릿에 지정된 출력이 포함됩니다.

```
$ aws cloudformation describe-stacks --stack-name Scheduler-Target-SQS
```

성공할 경우, 명령은 Amazon SQS 대기열을 만들고 다음과 같은 출력을 반환합니다.

```
{
    "Stacks": [
        {
            "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/
Scheduler-Target-SQS/1d2af345-a121-12eb-abc1-012e34567890",
            "StackName": "Scheduler-Target-SQS",
            "CreationTime": "2022-03-17T16:21:29.442000+00:00",
            "RollbackConfiguration": {},
```

```
            "StackStatus": "CREATE_COMPLETE",
            "DisableRollback": false,
            "NotificationARNs": [],
            "Outputs": [
                {
                    "OutputKey": "QueueName",
                    "OutputValue": "MyQueue",
                    "Description": "The name of the queue"
                },
                {
                    "OutputKey": "QueueARN",
                    "OutputValue": "arn:aws:sqs:us-west-2:123456789012:MyQueue",
                    "Description": "The ARN of the queue"
                },
                {
                    "OutputKey": "QueueURL",
                    "OutputValue": "https://sqs.us-
west-2.amazonaws.com/123456789012/MyQueue",
                    "Description": "The URL of the queue"
                }
            ],
            "Tags": [],
            "EnableTerminationProtection": false,
            "DriftInformation": {
                "StackDriftStatus": "NOT_CHECKED"
            }
        }
    ]
}
```

이 안내서의 뒷부분에서는 의 값을 사용하여 대기열을 EventBridge 스케줄러의 대상으로 설정해 보겠습니다. QueueARN

다음 단계

설정 단계를 완료한 후에는 시작 안내서를 사용하여 첫 번째 EventBridge 스케줄러 스케줄러를 만들고 대상을 호출하십시오.

EventBridge 스케줄러 시작하기

이 항목에서는 새 EventBridge 스케줄러 일정 생성에 대해 설명합니다. EventBridge 스케줄러 콘솔, AWS Command Line Interface(AWS CLI) 또는 AWS SDK를 사용하여 템플릿 기반 Amazon SQS 대상이 포함된 일정을 생성할 수 있습니다. 그런 다음 로깅을 설정하고, 재시도를 구성하고, 실패한 작업에 대한 최대 보존 시간을 설정합니다. 일정을 만든 후에는 일정이 대상을 간접적으로 성공적으로 호출하고 대상 대기열에 메시지를 전송하는지 확인합니다.

> **Note**
>
> 이 설명서를 따르려면 the section called “자격 증명 기반 정책 사용”에 설명된 최소 필수 권한으로 IAM 사용자를 설정하는 것이 좋습니다. 사용자를 생성하고 구성한 후 다음 명령을 실행하여 액세스 자격 증명을 설정합니다. AWS CLI를 구성하려면 액세스 키 ID와 보안 액세스 키가 필요합니다.
>
> ```
> $ aws configure
> AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
> AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
> Default region name [None]: us-west-2
> Default output format [None]: json
> ```
>
> 자격 증명을 설정하는 다양한 방법에 대한 자세한 내용은 버전 2용 AWS Command Line Interface 사용 설명서의 구성 설정 및 우선 순위를 참조하세요.

주제

필수 조건

이 섹션의 단계를 시도하기 전에 다음을 수행해야 합니다.

- 설정에 설명된 작업을 완료합니다.

EventBridge 스케줄러 콘솔을 사용하여 일정 생성

콘솔을 사용하여 새 일정을 생성하려면

1. AWS Management Console에 로그인한 다음 https://us-west-2.console.aws.amazon.com/scheduler/home?region=us-west-2#home 링크를 선택하여 EventBridge 콘솔의 EventBridge 스케줄러 섹션을 엽니다.

   > **Note**
   >
   > AWS Management Console의 리전 선택기를 사용하여 AWS 리전을 전환할 수 있습니다.

2. 일정 페이지에서 일정 생성을 선택합니다.
3. 일정 세부 정보 지정 페이지의 일정 이름 및 설명 섹션에서 다음을 수행합니다.

   a. 일정 이름에 일정의 이름을 입력합니다. 예: **`MyTestSchedule`**

   b. 선택 사항인 설명에 일정에 대한 설명을 입력합니다. 예: **`My first schedule`**.

   c. 일정 그룹의 경우 드롭다운 옵션에서 일정 그룹을 선택합니다. 이전에 일정 그룹을 만든 적이 없는 경우 일정에 맞는 `default` 그룹을 선택할 수 있습니다. 새 일정 그룹을 만들려면 콘솔 설명에서 나만의 일정 만들기 링크를 선택합니다. 일정 그룹을 사용하여 일정 그룹에 태그를 추가합니다.

4. 일정 패턴 섹션에서 다음을 수행합니다.

   a. 발생에서 다음 패턴 옵션 중 하나를 선택합니다. 구성 옵션은 선택한 패턴에 따라 달라집니다.

   - 일회성 일정 – 일회성 일정은 사용자가 지정하는 날짜와 시간에 한 번만 대상을 간접적으로 호출합니다.

     날짜 및 시간에 유효한 날짜를 `YYYY/MM/DD` 형식으로 입력합니다. 그런 다음 24시간 `hh:mm` 형식으로 타임스탬프를 지정합니다. 마지막으로 드롭다운 옵션에서 시간대를 선택합니다.

   - 반복 일정 – 반복 일정은 cron 표현식 또는 rate 표현식을 사용하여 지정한 속도로 대상을 간접적으로 호출합니다.

cron 표현식을 사용하여 일정을 구성하려면 Cron 기반 일정을 선택합니다. rate 표현식을 사용하려면 속도 기반 일정을 선택하고 값에 양수를 입력한 다음 드롭다운 옵션에서 단위를 선택합니다.

Cron 및 Rate 표현식 사용에 대한 자세한 내용은 일정 유형 섹션을 참조하세요.

b. 유연한 기간에서 끄기를 선택하여 옵션을 끄거나 드롭다운 목록에서 미리 정의된 기간 중 하나를 선택합니다. 예를 들어, 15분을 선택하고 1시간에 한 번씩 대상을 간접적으로 호출하도록 반복 일정을 설정하면 일정은 매시간 시작 후 15분 이내에 실행됩니다.

5.

> **Note**
>
> 일회성 일정에서는 유연한 기간 기능을 사용할 수 없습니다.

이전 단계에서 반복 일정을 선택한 경우 시간대 섹션에서 시간대를 지정하고 선택적으로 일정의 시작 날짜 및 시간, 종료 날짜 및 시간을 설정합니다. 시작 날짜가 없는 반복 일정은 생성되어 사용 가능한 즉시 시작됩니다. 종료 날짜가 없는 반복 일정은 계속해서 해당 대상을 무기한으로 간접적으로 호출합니다.

6. 다음(Next)을 선택합니다.
7. 대상 선택 페이지에서 다음을 수행합니다.

   a. 템플릿 기반 대상을 선택하고 대상 API를 선택합니다. 이 예제에서는 Amazon SQS **SendMessage** 템플릿 기반 대상을 선택합니다.

   b. SendMessage 섹션의 SQS 대기열의 경우 드롭다운 목록에서 `arn:aws:sqs:us-west-2:123456789012:TestQueue` 같은 기존 Amazon SQS 대기열 ARN을 선택합니다. 새 대기열을 생성하려면 새 SQS 대기열 생성을 선택하여 Amazon SQS 콘솔로 이동합니다. 대기열 생성을 완료한 후 EventBridge 스케줄러 콘솔로 돌아가서 드롭다운을 새로 고치십시오. 새 대기열 ARN이 나타나고 선택할 수 있습니다.

   c. 대상에는 EventBridge 스케줄러가 대상에 전달하려는 페이로드를 입력합니다. 이 예에서는 대상 대기열에 다음 메시지를 보냅니다. **`Hello, it's EventBridge Scheduler.`**

8. 다음을 선택한 다음 설정 - 옵션 페이지에서 다음을 수행하십시오.
9.

   a. 일정 상태 섹션의 일정 활성화에서 스위치를 사용하여 기능을 켜거나 끕니다. EventBridge 스케줄러는 기본적으로 일정을 활성화합니다.

b. 일정 완료 후 조치 섹션에서 일정 완료 후 EventBridge 스케줄러가 수행하는 작업을 구성합니다.

- 일정을 자동으로 삭제하려면 삭제를 선택합니다. 일회성 일정의 경우 일정이 대상을 한 번 간접적으로 호출한 후에 발생합니다. 반복 일정의 경우 이는 일정의 마지막으로 계획된 간접 호출 이후에 발생합니다. 자동 삭제에 대한 자세한 내용은 the section called “일정 완료 후 삭제” 섹션을 참조하세요.
- 일정이 완료된 후 EventBridge 스케줄러가 어떤 작업도 취하지 않도록 하려면 없음을 선택하거나 값을 선택하지 마십시오.

c. 재시도 정책 및 DLQ(Dead Letter Queue) 섹션에서 재시도 정책에 대해 재시도를 켜서 일정에 대한 재시도 정책을 구성합니다. 재시도 정책을 사용하면 일정이 대상을 간접적으로 호출하지 못할 경우 EventBridge 스케줄러가 일정을 다시 실행합니다. 구성된 경우 일정에 대한 최대 보존 기간과 재시도 횟수를 설정해야 합니다.

d. 최대 이벤트 수명 - 선택 사항에 EventBridge 스케줄러가 처리되지 않은 이벤트를 보관해야 하는 최대 시간과 분을 입력합니다.

> **Note**
>
> 최댓값은 24시간입니다.

e. 최대 재시도 횟수에는 대상이 오류를 반환할 경우 EventBridge 스케줄러가 일정을 재시도하는 최대 횟수를 입력합니다.

> **Note**
>
> 최댓값은 185회입니다.

f. DLQ(Dead Letter Queue)의 경우 다음 옵션 중에서 선택합니다.

- 없음 - DLQ를 구성하지 않으려면 이 옵션을 선택합니다.
- 내 AWS 계정의 Amazon SQS 대기열을 DLQ로 선택 - 이 옵션을 선택한 다음 드롭다운 목록에서 대기열 ARN을 선택하고 일정을 생성하는 AWS 계정와 동일한 DLQ를 구성합니다.
- 다른 AWS 계정의 Amazon SQS 대기열을 DLQ로 지정 — 이 옵션을 선택한 다음 대기열이 다른 AWS 계정에 있는 경우 구성한 대기열의 ARN을 DLQ로 입력합니다. 이 옵션을 사용하려면 대기열의 정확한 ARN을 입력해야 합니다.

g. 고객 관리형 키를 사용하여 대상 입력을 암호화하려면 암호화 섹션에서 암호화 설정 사용자 지정(고급)을 선택합니다. 이 옵션을 선택하는 경우 기존 KMS 키 ARN을 입력하거나 AWS

KMS 키 생성을 선택하여 AWS KMS 콘솔로 이동합니다. EventBridge 스케줄러가 저장 데이터를 암호화하는 방법에 대한 자세한 내용은 the section called “저장된 데이터 암호화” 섹션을 참조하세요.

h. 권한에서 기존 역할 사용을 선택한 다음, 설정 절차 중에 생성한 역할을 드롭다운 목록에서 선택합니다. 또한 IAM 콘솔로 이동을 선택하여 새 역할을 생성할 수도 있습니다.

EventBridge 스케줄러가 새 실행 역할을 생성하도록 하려면 이 일정에 대한 새 역할 생성을 선택합니다. 그런 다음 역할 이름을 입력합니다. 이 옵션을 선택하면 EventBridge 스케줄러가 템플릿 기반 대상에 필요한 필수 권한을 역할에 연결합니다.

10. 다음(Next)을 선택합니다.
11. 일정 검토 및 생성 페이지에서 일정의 세부 정보를 검토합니다. 각 섹션에서 편집을 선택하여 해당 단계로 돌아가서 세부 정보를 편집합니다.
12. 일정 생성을 선택하여 새 일정 생성을 완료합니다. 일정 페이지에서 새 일정과 기존 일정 목록을 볼 수 있습니다. 상태 열에서 새 일정이 활성화됨 상태인지 확인합니다.
13. 일정에 따라 Amazon SQS 대상이 간접적으로 호출되는지 확인하려면 Amazon SQS 콘솔을 열고 다음을 수행하십시오.

a. 대기열 목록에서 대상 대기열을 선택합니다.

b. [메시지 전송 및 수신(Send and receive messages)]을 선택합니다.

c. 메시지 보내기 및 받기 페이지의 메시지 수신에서 메시지 폴링을 선택하여 일정에 따라 대상 대기열로 보낸 테스트 메시지를 검색합니다.

AWS CLI를 사용하여 일정을 생성합니다.

다음 예제는 AWS CLI 명령 `create-schedule`을 사용하여 템플릿 기반 Amazon SQS 대상이 포함된 EventBridge 스케줄러 일정을 생성하는 방법을 보여줍니다. 다음 파라미터의 자리 표시자 값을 사용자의 정보로 바꿉니다.

- --일정 – 일정 이름을 입력합니다.
- RoleArn — 일정과 연결할 실행 역할의 ARN을 입력합니다.
- Arn — 대상의 ARN을 입력합니다. 이 경우 대상은 Amazon SQS 대기열입니다.
- 입력 - EventBridge 스케줄러가 대상 대기열에 전달하는 메시지를 입력합니다.

```
$ aws scheduler create-schedule --name sqs-templated-schedule --schedule-expression
 'rate(5 minutes)' \
```

```
--target '{"RoleArn": "ROLE_ARN", "Arn": "QUEUE_ARN", "Input": "TEST_PAYLOAD" }' \
--flexible-time-window '{ "Mode": "OFF"}'
```

EventBridge 스케줄러 SDK를 사용하여 일정 생성

다음 예제에서는 EventBridge 스케줄러 SDK를 사용하여 템플릿 기반 Amazon SQS 대상이 포함된 EventBridge 스케줄러 일정을 생성합니다.

Example Python SDK

```
import boto3
scheduler = boto3.client('scheduler')

flex_window = { "Mode": "OFF" }

sqs_templated = {
    "RoleArn": "<ROLE_ARN>",
    "Arn": "<QUEUE_ARN>",
    "Input": "Message for scheduleArn: '<aws.scheduler.schedule-arn>', scheduledTime:
 '<aws.scheduler.scheduled-time>'"
}

scheduler.create_schedule(
    Name="sqs-python-templated",
    ScheduleExpression="rate(5 minutes)",
    Target=sqs_templated,
    FlexibleTimeWindow=flex_window)
```

Example Java SDK

```
package com.example;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.scheduler.SchedulerClient;
import software.amazon.awssdk.services.scheduler.model.*;


public class MySchedulerApp {

    public static void main(String[] args) {

        final SchedulerClient client = SchedulerClient.builder()
```

```
                .region(Region.US_WEST_2)
                .build();

        Target sqsTarget = Target.builder()
                .roleArn("<ROLE_ARN>")
                .arn("<QUEUE_ARN>")
                .input("Message for scheduleArn: '<aws.scheduler.schedule-arn>',
 scheduledTime: '<aws.scheduler.scheduled-time>'")
                .build();

        CreateScheduleRequest createScheduleRequest = CreateScheduleRequest.builder()
                .name("<SCHEDULE NAME>")
                .scheduleExpression("rate(10 minutes)")
                .target(sqsTarget)
                .flexibleTimeWindow(FlexibleTimeWindow.builder()
                        .mode(FlexibleTimeWindowMode.OFF)
                        .build())
                .build();

        client.createSchedule(createScheduleRequest);
        System.out.println("Created schedule with rate expression and an Amazon SQS
 templated target");
    }
}
```

다음 단계

- 콘솔, AWS CLI 또는 EventBridge 스케줄러 SDK를 사용하여 일정을 관리하는 방법에 대한 자세한 내용은 일정 관리 섹션을 참조하세요.
- 템플릿 기반 대상 구성 방법과 범용 대상 파라미터의 사용에 대한 자세한 내용은 대상 관리 섹션을 참조하세요.
- EventBridge 스케줄러 데이터 유형과 API 작업에 대한 자세한 내용은 EventBridge 스케줄러 API 참조를 참조하세요.

EventBridge 스케줄러의 일정 유형

다음 주제에서는 Amazon EventBridge 스케줄러가 지원하는 다양한 일정 유형과 EventBridge 스케줄러가 일광 절약 시간제를 처리하는 방법과 다양한 시간대의 일정 지정에 대해 설명합니다. 일정을 구성할 때 속도 기반, cron 기반 및 일회성 일정이라는 세 가지 일정 유형 중에서 선택할 수 있습니다.

속도 기반 일정과 cron 기반 일정 모두 반복 일정입니다. 구성하려는 일정 유형에 대한 일정 표현식을 사용하고 EventBridge 스케줄러가 표현식을 평가하는 시간대를 지정하여 모든 반복 일정 유형을 구성합니다.

일회성 일정은 대상을 한 번만 간접적으로 호출하는 일정입니다. EventBridge 스케줄러가 일정을 평가하는 시간, 날짜 및 시간대를 지정하여 일회성 일정을 구성할 수 있습니다.

> **Note**
>
> EventBridge 스케줄러의 모든 일정 유형은 60초 정밀도로 대상을 간접적으로 호출합니다. 즉, 일정을 `1:00`에 실행하도록 설정하면 `1:00:00~1:00:59`에서 대상 API가 간접적으로 호출됩니다.

다음 섹션을 사용하여 각 반복 일정 유형에 대한 일정 표현식을 구성하는 방법과 EventBridge 스케줄러에서 일회성 일정을 설정하는 방법에 대해 알아보십시오.

주제

속도 기반 일정

속도 기반 일정은 일정에 지정한 시작 날짜 이후에 시작되며 일정 종료 날짜까지 사용자가 정의한 일반 속도로 실행됩니다. 속도 기반 일정을 사용하여 가장 일반적인 반복 예약 사용 사례를 설정할 수 있습니다. 예를 들어 15분마다, 2시간에 한 번 또는 5일에 한 번씩 대상을 간접적으로 호출하는 일정을 원

한다면 속도 기반 일정을 사용하여 이를 달성할 수 있습니다. rate 표현식을 사용하여 속도 기반 일정을 구성합니다.

속도 기반 일정의 경우 `StartDate` 속성을 사용하여 일정의 첫 번째 발생을 설정합니다. 속도 기반 일정에 대해 `StartDate`를 제공하지 않으면 일정에서 즉시 대상을 간접적으로 호출하기 시작합니다.

Rate 표현식에는 다음과 같이 공백으로 구분된 두 개의 필수 필드가 있습니다.

구문

```
rate(value unit)
```

value

양수.

unit

일정에서 대상을 간접적으로 호출할 시간 단위입니다.

유효한 입력: `minutes` | `hours` | `days`

예제

다음 예제는 AWS CLI `create-schedule` 명령과 함께 rate 표현식을 사용하여 속도 기반 일정을 구성하는 방법을 보여줍니다. 이 예제에서는 5분마다 실행되는 일정을 생성하고 템플릿 기반 `SqsParameters` 대상 유형을 사용하여 Amazon SQS 대기열에 메시지를 전송합니다.

이 예제에서는 `--start-date` 파라미터 값을 설정하지 않으므로, 일정을 생성하고 활성화한 직후에 일정에서 대상을 간접적으로 호출하기 시작합니다.

```
$ aws scheduler create-schedule --schedule-expression 'rate(5 minutes)' --
name schedule-name \
--target '{"RoleArn": "role-arn", "Arn": "QUEUE_ARN", "Input": "TEST_PAYLOAD" }' \
--flexible-time-window '{ "Mode": "OFF"}'
```

Cron 기반 일정

cron 표현식은 선택한 특정 시간에 실행되는 세분화된 반복 일정을 생성합니다. EventBridge 스케줄러는 UTC(협정 표준시) 또는 일정을 생성할 때 지정한 시간대로 cron 기반 일정을 구성할 수 있도록 지원

합니다. Cron 기반 일정을 사용하면 일정 실행 시기와 빈도를 더 잘 제어할 수 있습니다. EventBridge 스케줄러의 rate 표현식 중 하나에서 지원되지 않는 사용자 지정 반복 일정이 필요한 경우 cron 기반 일정을 사용하십시오. 예를 들어 오전 8시에 실행되는 cron 기반 일정을 생성할 수 있습니다. 매월 첫째 월요일(PST) cron 표현식을 사용하여 cron 기반 일정을 구성합니다.

cron 표현식은 다음과 같이 분, 시간, 일, 월, 요일 및 선택적 필드인 연도 등 공백으로 구분된 다섯 개의 필수 필드로 구성됩니다.

구문

```
cron(minutes hours day-of-month month day-of-week year)
```

필드	값	와일드카드
Minutes	0~59	, - * /
시간	0~23	, - * /
날짜	1~31	, - * ? / L W
월	1-12 또는 JAN-DEC	, - * /
요일	1-7 또는 SUN-SAT	, - * ? L #
연도	1970~2199	, - * /

와일드카드

- ,(쉼표) 와일드카드는 추가 값을 포함합니다. ‘월’ 필드에서 JAN, FEB, MAR은 1월, 2월, 3월을 포함한다는 의미입니다.
- -(대시) 와일드카드는 범위를 지정합니다. ‘일’ 필드에서 1-15는 지정된 달의 1일에서 15일까지 포함한다는 의미입니다.
- *(별표) 와일드카드는 필드의 모든 값을 포함합니다. ‘시간’ 필드에서 *는 모든 시간을 포함한다는 의미입니다. '날짜' 및 '요일' 필드 모두에서 *를 사용할 수 없습니다. 필드 중 하나에 사용할 경우 다른 하나에는 반드시 ?를 사용해야 합니다.
- /(슬래시) 와일드카드로 증분을 지정합니다. 예를 들어, '분' 필드에 1/10을 입력하면 지정한 시간의 1분부터 시작해서 매 10분 간격을 지정할 수 있습니다(즉, 11분, 21분, 31분 등).

- ?(물음표) 와일드카드는 any를 지정합니다. '날짜' 필드에 7을 입력하고 일주일 중 어느 날이라도 괜찮다면 '요일' 필드에는 ?을 입력합니다.
- '날짜' 또는 '요일' 필드에서 L 와일드카드로 해당 월 또는 주의 마지막 날을 지정할 수 있습니다.
- '날짜' 필드에서는 **W** 와일드카드로 어떤 한 평일을 지정할 수 있습니다. 예를 들어 '날짜' 필드에 **3W**를 입력하면 해당 월의 세 번째 평일에 가장 가까운 날을 지정할 수 있습니다.
- '요일' 필드의 # 와일드카드는 그 달에 속한 정해진 요일의 특정 인스턴스를 지정합니다. 예를 들어, 3#2는 그 달의 두 번째 화요일입니다. 3은 각 주의 셋째 날이므로 화요일을 나타내고 2는 그 달의 두 번째 해당 요일입니다.

> **Note**
>
> '#' 문자를 사용하는 경우 요일(day-of-week) 필드에 하나의 표현식만 정의할 수 있습니다. 예를 들어 `"3#1,6#3"`은(는) 두 개의 표현식으로 해석되기 때문에 유효하지 않습니다.

예제

다음 예제는 AWS CLI `create-schedule` 명령과 함께 cron 표현식을 사용하여 cron 기반 일정을 구성하는 방법을 보여줍니다. 이 예제는 2022년부터 2023년까지 매월 마지막 금요일 오전 10시 15분(UTC+0)에 실행되는 일정을 생성하고 템플릿 기반 `SqsParameters` 대상 유형을 사용하여 Amazon SQS 대기열에 메시지를 전송합니다.

```
$ aws scheduler create-schedule --schedule-expression "cron(15 10 ? * 6L 2022-2023)" --
name schedule-name \
--target '{"RoleArn": "role-arn", "Arn": "QUEUE_ARN", "Input": "TEST_PAYLOAD" }' \
--flexible-time-window '{ "Mode": "OFF"}'
```

일회성 일정

일회성 일정은 유효한 날짜와 타임스탬프를 사용하여 사용자가 지정하는 날짜와 시간에 한 번만 대상을 간접적으로 호출합니다. EventBridge 스케줄러는 UTC(협정 표준시) 또는 일정을 생성할 때 지정한 시간대로 일정을 구성할 수 있도록 지원합니다.

> **Note**
>
> 일회성 일정은 실행을 완료하고 대상을 간접적으로 호출한 후에도 여전히 계정 할당량에 계산됩니다. 일회성 스케줄이 실행을 완료한 후에는 해당 일정을 삭제하는 것이 좋습니다.

at 표현식을 사용하여 일회성 일정을 구성합니다. at 표현식은 다음과 같이 EventBridge 스케줄러가 일정을 간접적으로 호출하도록 하려는 날짜 및 시간으로 구성됩니다.

구문

```
at(yyyy-mm-ddThh:mm:ss)
```

일회성 일정을 구성하면 EventBridge 스케줄러는 사용자가 일정에 지정한 `StartDate` 및 `EndDate`를 무시합니다.

예제

다음 예제는 AWS CLI `create-schedule` 명령과 함께 at 표현식을 사용하여 일회성 일정을 구성하는 방법을 보여줍니다. 이 예제는 2022년 11월 20일 오후 1시(UTC-8)에 한 번 실행되는 일정을 생성하고 템플릿 기반 `SqsParameters` 대상 유형을 사용하여 Amazon SQS 대기열에 메시지를 전송합니다.

```
$ aws scheduler create-schedule --schedule-expression "at(2022-11-20T13:00:00)" --
name schedule-name \
--target '{"RoleArn": "role-arn", "Arn": "QUEUE_ARN", "Input": "TEST_PAYLOAD" }' \
--schedule-expression-timezone "America/Los_Angeles"
--flexible-time-window '{ "Mode": "OFF"}'
```

EventBridge 스케줄러의 시간대

EventBridge 스케줄러는 사용자가 지정하는 모든 시간대에 cron 기반 일정 및 일회성 일정을 구성할 수 있도록 지원합니다. EventBridge 스케줄러는 IANA(Internet Assigned Numbers Authority)에서 유지 관리하는 시간대 데이터베이스를 사용합니다.

AWS CLI를 사용하면 EventBridge 스케줄러가 `--schedule-expression-timezone` 파라미터를 사용하여 일정을 평가하도록 하려는 시간대를 설정할 수 있습니다. 예를 들어 다음 명령은 매일 오전 8시 30분에 미국/뉴욕에서 템플릿 기반 Amazon SQS `SendMessage` 대상을 간접적으로 호출하는 cron 기반 일정을 생성합니다.

```
$ aws scheduler create-schedule --schedule-expression "cron(30 8 * * ? *)" --name
 schedule-in-est \
    --target '{"RoleArn": "role-arn", "Arn": "QUEUE_ARN", "Input": "This schedule runs
 in the America/New_York time zone." }' \
    --schedule-expression-timezone "America/New_York"
```

```
    --flexible-time-window '{ "Mode": "OFF"}'
```

EventBridge 스케줄러의 일광 절약 시간제

EventBridge 스케줄러는 일광 절약 시간제에 맞게 일정을 자동으로 조정합니다. 봄에 시간이 앞으로 당겨질 때 cron 표현식이 존재하지 않는 날짜 및 시간에 해당하는 경우 일정 간접 호출을 건너뛰게 됩니다. 가을에 시간이 뒤로 당겨질 때 일정은 한 번만 실행되며 간접 호출을 반복하지 않습니다. 다음 간접 호출은 지정된 날짜 및 시간에 정상적으로 발생합니다.

EventBridge 스케줄러는 일정을 생성할 때 지정한 시간대에 따라 일정을 조정합니다. 미국/뉴욕에서 일정을 구성한 경우 해당 시간대의 시간이 변경되면 일정이 조정되고, 미국/로스앤젤레스의 일정은 서부 해안의 시간이 변경되면 3시간 후에 조정됩니다.

days를 단위로 사용하는 속도 기반 일정의 경우(예: `rate(1 days)`) days는 24시간을 기준으로 합니다. 즉, 일광 절약 시간제로 인해 하루가 23시간으로 단축되거나 25시간으로 연장되는 경우에도 EventBridge Scheduler는 일정을 마지막으로 간접 호출한 지 24시간 후에 rate 표현식을 평가합니다.

> **Note**
>
> 현지 규칙 및 규정에 따라 일부 시간대는 일광 절약 시간제를 준수하지 않습니다. 일광 절약 시간제를 준수하지 않는 시간대로 일정을 생성하는 경우 EventBridge 스케줄러는 일정을 조정하지 않습니다. 일광 절약 시간 조정은 협정 세계시(UTC) 일정에는 적용되지 않습니다.

예

미국/로스앤젤레스에서 cron 표현식(예: `cron(30 2 * * ? *)`)을 사용하여 일정을 만드는 시나리오를 생각해 보십시오. 이 일정은 지정된 시간대로 매일 오전 2시 30분에 실행됩니다.

- 봄에는 앞으로 — 봄철에 시간이 오전 1시 59분에서 오전 3시로 앞으로 바뀌면 EventBridge 스케줄러는 해당 날짜에 일정 간접 호출을 건너뛰고 다음 날 정상적으로 일정 실행을 재개합니다.
- 가을에는 뒤로 – 가을에 시간이 오전 2시 59분에서 오전 2시로 뒤로 바뀌는 경우 EventBridge스케줄러는 변동이 발생하기 전 오전 2시 30분에 일정을 한 번만 실행하며, 시간 변동 후 오전 2시 30분에 일정 간접 호출을 다시 반복하지 않습니다.

일정 관리

일정은 Amazon EventBridge 스케줄러를 사용하여 생성, 구성 및 관리하는 주요 리소스입니다.

모든 일정에는 일정 실행 시기와 빈도를 결정하는 일정 표현식이 있습니다. EventBridge 스케줄러는 속도, cron 및 일회성 일정이라는 세 가지 유형의 일정을 지원합니다. 다양한 일정 유형에 대한 자세한 내용은 일정 유형 섹션을 참조하세요.

일정을 생성할 때 해당 일정이 간접적으로 호출할 대상을 구성합니다. 대상은 일정이 실행될 때마다 EventBridge 스케줄러가 사용자를 대신하여 호출하는 API 작업입니다. EventBridge Scheduler는 두 가지 유형의 대상을 지원합니다. 템플릿 기반 대상은 핵심 서비스 그룹에서 공통 API 작업을 직접적으로 호출하고, 다른 하나는 270개 이상의 서비스에서 6,000개 이상의 작업을 직접적으로 호출하는 데 사용할 수 있는 범용 대상 파라미터(UTP)입니다. 대상 구성에 대한 자세한 내용은 대상 관리 섹션을 참조하세요.

EventBridge 스케줄러가 이벤트를 대상에 성공적으로 전달할 수 없는 경우 재시도 정책과 DLQ(Dead Letter Queue)라는 두 가지 기본 메커니즘을 사용하여 일정이 실패를 처리하는 방법을 구성합니다. 재시도 정책은 EventBridge 스케줄러가 실패한 이벤트를 재시도해야 하는 횟수와 처리되지 않은 이벤트를 보관하는 기간을 결정합니다. DLQ는 EventBridge 스케줄러가 재시도 정책이 소진된 후 실패한 이벤트를 전송하는 데 사용하는 표준 Amazon SQS 대기열입니다. DLQ를 사용하여 일정 또는 다운스트림 대상과 관련된 문제를 해결할 수 있습니다. 자세한 정보는 the section called “배달 못한 편지 대기열 구성” 섹션을 참조하세요.

이 섹션에서는 콘솔, AWS CLI 및 EventBridge 스케줄러 SDK를 사용하여 EventBridge 스케줄러 일정을 관리하는 예를 찾을 수 있습니다.

주제

일정 상태 변경

EventBridge 스케줄러 일정에는 활성화와 비활성화라는 두 가지 상태가 있습니다. 다음 예제에서는 `UpdateSchedule`을 사용하여 5분마다 실행되고 Lambda 대상을 간접적으로 호출하는 일정을 비활성화합니다.

`UpdateSchedule`을 사용할 때는 모든 필수 파라미터를 제공해야 합니다. EventBridge 스케줄러는 사용자의 일정을 사용자가 제공한 정보로 대체합니다. 이전에 설정한 파라미터를 지정하지 않으면 `null`을 기본값으로 사용합니다.

Example AWS CLI

```
$ aws scheduler update-schedule --name lambda-universal --schedule-expression 'rate(5
 minutes)' \
--target '{"RoleArn": "ROLE_ARN", "Arn":"arn:aws:scheduler:::aws-sdk:lambda:invoke"
 "Input": "{\"FunctionName\":\"arn:aws:lambda:REGION:123456789012:function:HelloWorld
\",\"InvocationType\":\"Event\",\"Payload\":\"{\\\"message\\\":\\\"testing function\\
\"}\"}" }' \
--flexible-time-window '{ "Mode": "OFF"}' \
--state DISABLED
```

```
{
    "ScheduleArn": "arn:aws:scheduler:us-west-2:123456789012:schedule/default/lambda-
universal"
}
```

다음 예제에서는 Python SDK와 `UpdateSchedule` 작업을 사용하여 템플릿 기반 대상을 사용하여 Amazon SQS를 대상으로 하는 일정을 비활성화합니다.

Example Python SDK

```
import boto3
scheduler = boto3.client('scheduler')

sqs_templated = {
    "RoleArn": "<ROLE_ARN>",
    "Arn": "<QUEUE_ARN>",
    "Input": "{}"}

flex_window = { "Mode": "OFF" }
```

```
scheduler.update_schedule(Name="your-schedule",
    ScheduleExpression="rate(5 minutes)",
    Target=sqs_templated,
    FlexibleTimeWindow=flex_window,
    State='DISABLED')
```

유연한 기간 구성

유연한 기간으로 일정을 구성하면 EventBridge 스케줄러가 사용자가 설정한 기간 내에 대상을 간접적으로 호출합니다. 이는 대상의 정확한 예약 호출이 필요하지 않은 경우에 유용합니다. 기간을 유연하게 설정하면 대상의 간접 호출을 분산시켜 일정의 안정성을 높일 수 있습니다.

예를 들어 매시간 실행되는 일정에 대해 15분의 유연한 기간을 구성하면 예약된 시간보다 15분 후 내에 대상이 간접적으로 호출됩니다. 다음 AWS CLI와 EventBridge 스케줄러 SDK 예제에서는 `UpdateSchedule`을 사용하여 1시간에 한 번 실행되는 일정에 대해 15분의 유연한 기간을 설정합니다.

> **Note**
>
> 유연한 기간을 설정할지 여부를 지정해야 합니다. 이 옵션을 설정하지 않으려면 `OFF`를 지정합니다. 값을 `FLEXIBLE`로 설정하는 경우 일정을 실행할 최대 기간을 지정해야 합니다.

Example AWS CLI

```
$ aws scheduler update-schedule --name lambda-universal --schedule-expression 'rate(1
 hour)' \
--target '{"RoleArn": "ROLE_ARN", "Arn":"arn:aws:scheduler:::aws-sdk:lambda:invoke"
 "Input": "{\"FunctionName\":\"arn:aws:lambda:REGION:123456789012:function:HelloWorld
\",\"InvocationType\":\"Event\",\"Payload\":\"{\\\"message\\\":\\\"testing function\\
\"}\"}" }' \
--flexible-time-window '{ "Mode": "FLEXIBLE", "MaximumWindowInMinutes": 15} \
```

```
{
    "ScheduleArn": "arn:aws:scheduler:us-west-2:123456789012:schedule/lambda-universal"
}
```

Example Python SDK

```
import boto3
```

```
scheduler = boto3.client('scheduler')

sqs_templated = {
    "RoleArn": "<ROLE_ARN>",
    "Arn": "<QUEUE_ARN>",
    "Input": "{}"}

flex_window = { "Mode": "FLEXIBLE", "MaximumWindowInMinutes": 15}

scheduler.update_schedule(Name="your-schedule",
    ScheduleExpression="rate(1 hour)",
    Target=sqs_templated,
    FlexibleTimeWindow=flex_window)
```

일정에 대한 DLQ(Dead Letter Queue) 구성

Amazon EventBridge 스케줄러는 Amazon Simple Queue Service를 사용하여 DLQ(Dead Letter Queue)를 지원합니다. 일정이 대상을 간접적으로 호출하지 못하면 EventBridge 스케줄러는 간접적 호출 세부 정보 및 대상으로부터 수신한 모든 응답을 포함하는 JSON 페이로드를 사용자가 지정하는 Amazon SQS 표준 대기열로 전달합니다.

다음 주제에서는 이 JSON을 Dead Letter 이벤트라고 합니다. Dead Letter 이벤트를 사용하면 일정이나 목표와 관련된 문제를 해결할 수 있습니다. 일정에 맞게 재시도 정책을 구성하면 EventBridge 스케줄러는 사용자가 설정한 최대 재시도 횟수를 모두 사용한 Dead Letter 이벤트를 전달합니다.

다음 항목에서는 Amazon SQS 대기열을 일정에 맞는 DLQ로 구성하고, EventBridge 스케줄러가 Amazon SQS로 메시지를 전송하는 데 필요한 권한을 설정하고, DLQ에서 Dead Letter 이벤트를 수신하는 방법을 설명합니다.

주제

- Amazon SQS 대기열 생성
- 실행 역할 권한 설정
- DLQ(Dead Letter Queue) 지정
- dead-letter 이벤트 검색

Amazon SQS 대기열 생성

일정에 맞게 DLQ를 구성하기 전에 표준 Amazon SQS 대기열을 생성해야 합니다. 자세한 지침은 Amazon Simple Queue Service 개발자 안내서의 Amazon SQS 대기열 생성을 참조하세요.

> **Note**
>
> EventBridge 스케줄러는 FIFO 대기열을 스케줄의 DLQ로 사용하는 것을 지원하지 않습니다.

다음 AWS CLI 명령을 사용하여 표준 대기열을 생성할 수 있습니다.

```
$ aws sqs create-queue --queue-name queue-name
```

성공하면 결과에 QueueURL이 표시됩니다.

```
{
    "QueueUrl": "https://sqs.us-west-2.amazonaws.com/123456789012/scheduler-dlq-test"
}
```

대기열을 생성한 후 대기열 ARN을 기록해 둡니다. EventBridge 스케줄러 일정에 DLQ를 지정할 때 ARN이 필요합니다. 대기열 ARN은 Amazon SQS 콘솔에서 또는 get-queue-attributesAWS CLI 명령을 사용하여 찾을 수 있습니다.

```
$ aws sqs get-queue-attributes --queue-url your-dlq-url --attribute-names QueueArn
```

성공하면 결과에 대기열 ARN이 표시됩니다.

```
{
    "Attributes": {
        "QueueArn": "arn:aws:sqs:us-west-2:123456789012:scheduler-dlq-test"
    }
}
```

다음 섹션에서는 EventBridge 스케줄러가 Amazon SQS에 Dead Letter 이벤트를 전송할 수 있도록 일정 실행 역할에 필요한 권한을 추가합니다.

실행 역할 권한 설정

EventBridge 스케줄러가 Dead Letter 이벤트를 Amazon SQS로 전송하도록 하려면 일정 실행 역할에 다음과 같은 권한 정책이 필요합니다. 일정 실행 역할에 새 권한 정책을 연결하는 방법에 대한 자세한 내용은 실행 역할 설정을 참조하세요.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "sqs:SendMessage"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> EventBridge 스케줄러를 사용하여 Amazon SQS API 대상을 간접적으로 호출하는 경우 일정 실행 역할에 필요한 권한이 이미 연결되어 있을 수 있습니다.

다음 섹션에서는 EventBridge 스케줄러 콘솔을 사용하고 일정에 대한 DLQ를 지정합니다.

DLQ(Dead Letter Queue) 지정

DLQ를 지정하려면 EventBridge 스케줄러 콘솔 또는 AWS CLI를 사용하여 기존 일정을 업데이트하거나 새 일정을 생성하십시오.

Console

콘솔을 사용하여 DLQ를 지정하려면

1. AWS Management Console에 로그인한 다음 https://console.aws.amazon.com/scheduler/home 링크를 선택하여 EventBridge 콘솔의 EventBridge 스케줄러 섹션을 엽니다.
2. EventBridge 스케줄러 콘솔에서 새 일정을 만들거나 일정 목록에서 편집할 기존 일정을 선택합니다.
3. 설정 페이지에서 DLQ(Dead Letter Queue)에 대해 다음 중 하나를 수행하십시오.
   - 내 AWS 계정의 Amazon SQS 대기열을 DLQ로 선택을 선택한 다음 드롭다운 목록에서 해당 DLQ에 대한 대기열 ARN을 선택합니다.
   - 다른 AWS 계정의 Amazon SQS 대기열을 DLQ로 지정을 선택한 다음 DLQ에 대한 대기열 ARN을 입력합니다. 다른 AWS 계정에서 대기열을 선택하면 EventBridge 스케줄러 콘솔은 대기열 ARN을 드롭다운 목록에 표시할 수 없습니다.

4. 선택 사항을 검토한 다음 일정 생성 또는 일정 저장을 선택하여 DLQ 구성을 완료하십시오.
5. (선택 사항) 일정의 DLQ 세부 정보를 보려면 목록에서 일정 이름을 선택한 다음 일정 세부 정보 페이지에서 DLQ(Dead Letter Queue) 탭을 선택합니다.

AWS CLI

AWS CLI를 사용하여 기존 일정을 업데이트하려면

- update-schedule명령을 사용하여 일정을 업데이트하십시오. 이전에 생성한 Amazon SQS 대기열을 DLQ로 지정합니다. 필수 Amazon SQS 권한을 연결한 IAM 역할 ARN을 실행 역할로 지정합니다. 다른 모든 자리 표시자 값을 해당 정보로 바꿉니다.

```
$ aws scheduler update-schedule --name existing-schedule \
    --schedule-expression 'rate(5 minutes)' \
    --target '{"DeadLetterConfig": {"Arn": "DLQ_ARN"}, "RoleArn": "ROLE_ARN",
 "Arn":"QUEUE_ARN", "Input": "Hello world!" }' \
    --flexible-time-window '{ "Mode": "OFF"}'
```

AWS CLI를 사용하여 DLQ로 새 일정을 만들려면

- 일정을 생성하려면 create-schedule 명령을 사용합니다. 모든 자리 표시자 값을 해당 정보로 바꿉니다.

```
$ aws scheduler create-schedule --name new-schedule \
    --schedule-expression 'rate(5 minutes)' \
    --target '{"DeadLetterConfig": {"Arn": "DLQ_ARN"}, "RoleArn": "ROLE_ARN",
 "Arn":"QUEUE_ARN", "Input": "Hello world!" }' \
    --flexible-time-window '{ "Mode": "OFF"}'
```

다음 섹션에서는 AWS CLI를 사용하여 DLQ에서 dead-letter 이벤트를 수신합니다.

dead-letter 이벤트 검색

다음 그림과 같이 receive-message 명령을 사용하여 DLQ에서 dead-letter 이벤트를 검색합니다. `--max-number-of-messages` 속성을 사용하여 검색할 메시지 수를 설정할 수 있습니다.

```
$ aws sqs receive-message --queue-url your-dlq-url --attribute-names All --message-
attribute-names All --max-number-of-messages 1
```

성공한 경우 다음과 유사한 출력 화면이 표시됩니다.


```
{
    "Messages": [
        {
            "MessageId": "2aeg3510-fe3a-4f5a-ab6a-6906560eaf7e",
            "ReceiptHandle": "AQEBkNKTdOMrWgHKPoITRBwrPoK3eCSZIcZwVqCY0BZ
+FfTcORFpopJbtCqj36VbBTlHreM8+qM/m5jcwqSlAlGmIJO/hYmMgn/
+dwIty9izE7HnpvRhhEyHxbeTZ5V05RbeasYaBdNyi9WLcnAHviDh6MebLXXNWoFyYNsxdwJuG0f/
w3htX6r3dxpXvvFNPGoQb8ihY37+u0gtsbuIwhLtUSmE8rbldEEwiUfi3IJ1zEZpUS77n/k1GWrMrnYg0Gx/
BuaLzOrFi2F738XI/
Hnh45uv3ca6OYwS1ojPQ1LtX2URg1haV5884FYlaRvY8jRlpCZabTkYRTZKSXG5KNgYZnHpmsspii6JNkjitYVFKPo0H91w
            "MD5OfBody": "07adc3fc889d6107d8bb8fda42fe0573",
            "Body": "{\"MessageBody\":\"Hello, world!\",\"QueueUrl\":\"https://sqs.us-
west-2.amazonaws.com/123456789012/does-not-exist\"}",
            "Attributes": {
                "SenderId": "AROA2DZE3W4CTL5ZR7EIN:ff00212d8c453aaaae644bc6846d4723",
                "ApproximateFirstReceiveTimestamp": "1652499058144",
                "ApproximateReceiveCount": "2",
                "SentTimestamp": "1652490733042"
            },
            "MD5OfMessageAttributes": "f72c1d78100860e00403d849831d4895",
            "MessageAttributes": {
                "ERROR_CODE": {
                    "StringValue": "AWS.SimpleQueueService.NonExistentQueue",
                    "DataType": "String"
                },
                "ERROR_MESSAGE": {
                    "StringValue": "The specified queue does not exist for this wsdl
 version.",
                    "DataType": "String"
                },
                "EXECUTION_ID": {
                    "StringValue": "ad06616e51cdf74a",
                    "DataType": "String"
                },
                "EXHAUSTED_RETRY_CONDITION": {
                    "StringValue": "MaximumEventAgeInSeconds",
                    "DataType": "String"
                }
                "IS_PAYLOAD_TRUNCATED": {
                    "StringValue": "false",
                    "DataType": "String"
                },
```

```
                "RETRY_ATTEMPTS": {
                    "StringValue": "0",
                    "DataType": "String"
                },
                "SCHEDULED_TIME": {
                    "StringValue": "2022-05-14T01:12:00Z",
                    "DataType": "String"
                },
                "SCHEDULE_ARN": {
                    "StringValue": "arn:aws:scheduler:us-west-2:123456789012:schedule/
DLQ-test",
                    "DataType": "String"
                },
                "TARGET_ARN": {
                    "StringValue": "arn:aws:scheduler:::aws-sdk:sqs:sendMessage",
                    "DataType": "String"
                }
            }
        }
    ]
}
```

대상 호출이 실패한 가능한 원인을 식별하고 문제를 해결하는 데 도움이 되도록 dead-letter 이벤트의 다음 속성을 기록해 두십시오.

- **ERROR_CODE** - EventBridge 스케줄러가 대상의 서비스 API로부터 수신하는 오류 코드를 포함합니다. 위 예제에서 Amazon SQS에서 반환한 오류 코드는 `AWS.SimpleQueueService.NonExistentQueue`입니다. EventBridge 스케줄러 문제로 인해 일정이 대상을 간접적으로 호출하지 못하는 경우 대신 `AWS.Scheduler.InternalServerError` 오류 코드가 표시됩니다.
- **ERROR_MESSAGE** - EventBridge 스케줄러가 대상의 서비스 API로부터 수신하는 오류 메시지를 포함합니다. 위 예제에서 Amazon SQS에서 반환한 오류 메시지는 `The specified queue does not exist for this wsdl version`입니다. EventBridge 스케줄러 문제로 인해 일정이 실패하는 경우 대신 `Unexpected error occurred while processing the request` 오류 코드가 표시됩니다.
- **TARGET_ARN** - 스케줄에서 간접적으로 호출하는 대상의 ARN(다음 서비스 ARN 형식: `arn:aws:scheduler:::aws-sdk:`*`service`*`:`*`apiAction`*)입니다.
- **EXHAUSTED_RETRY_CONDITION** - 이벤트가 DLQ에 전달된 이유를 나타냅니다. 이 속성은 대상 API의 오류가 영구적인 오류가 아니라 재시도 가능한 오류인 경우 나타납니다. 이 속성에는 일정에 대해 구성한 최대 재시도 횟수를 초과한 후 EventBridge 스케줄러가 DLQ로 전송한 경우 이 속성에

`MaximumRetryAttempts` 값이 포함될 수 있고, 이벤트가 일정에 구성한 최대 기간보다 오래되었는데도 전송에 실패한 경우 이 속성에 `MaximumEventAgeInSeconds` 값이 포함될 수 있습니다.

위 예제에서는 오류 코드와 오류 메시지를 기반으로 일정에 지정한 대상 대기열이 존재하지 않음을 확인할 수 있습니다.

일정 삭제

자동 삭제를 구성하거나 개별 일정을 수동으로 삭제하여 일정을 삭제할 수 있습니다. 다음 항목을 통해 두 가지 방법을 모두 사용하여 일정을 삭제하는 방법과 한 가지 방법을 다른 방법 대신 선택하는 이유를 알아보십시오.

주제

일정 완료 후 삭제

EventBridge 스케줄러에서 일정 리소스를 개별적으로 관리할 필요가 없도록 하려면 일정 완료 후 자동 삭제를 구성하십시오. 한 번에 수천 개의 일정을 만들고 필요에 따라 일정 수를 확장할 수 있는 유연성이 필요한 애플리케이션에서는 자동 삭제를 통해 지정된 지역의 일정 수에 해당하는 계정 할당량에 도달하지 않을 수 있습니다.

일정에 대한 자동 삭제를 구성하면 EventBridge 스케줄러는 마지막 대상의 간접 호출 후에 일정을 삭제합니다. 일회성 일정의 경우 일정이 대상을 한 번 간접적으로 호출한 후에 삭제가 발생합니다. rate 또는 cron 표현식으로 설정한 반복 일정의 경우 마지막 간접 호출 이후 일정이 삭제됩니다. 반복 일정의 마지막 간접 호출은 지정한 `EndDate`와 가장 가까운 시간에 발생하는 간접 호출입니다. 자동 삭제를 사용하여 일정을 구성하지만 `EndDate` 값을 지정하지 않은 경우 EventBridge 스케줄러는 일정을 자동으로 삭제하지 않습니다.

일정을 처음 생성할 때 자동 삭제를 설정하거나 기존 일정의 기본 설정을 업데이트할 수 있습니다. 다음 단계에서는 기존 예약에 자동 삭제를 구성하는 방법을 설명합니다.

AWS Management Console

1. https://console.aws.amazon.com/scheduler/에서 EventBridge 스케줄러 콘솔을 엽니다.
2. 일정 목록에서 편집하려는 일정을 선택한 다음 편집을 선택합니다.

3. 왼쪽 탐색 창에서 설정을 선택합니다.
4. 일정 완료 후 작업 섹션의 드롭다운 목록에서 삭제를 선택한 다음 변경 내용을 저장합니다.

AWS CLI

1. 새 프롬프트 창을 엽니다.
2. update-schedule AWS CLI 명령을 사용하여 다음과 같이 기존 일정을 업데이트합니다. 이 명령은 `--action-after-completion`을 `DELETE`로 설정합니다. 이 예제에서는 대상 구성을 JSON 파일에 로컬로 정의했다고 가정합니다. 일정을 업데이트하려면 대상과 기존 일정에 대해 구성하려는 기타 일정 파라미터를 제공해야 합니다.

   이 일정은 시간 당 간접 호출 1회의 속도를 가진 반복 일정입니다. 따라서 `--action-after-completion` 파라미터를 설정할 때 종료 날짜를 지정합니다.

```
$ aws scheduler update-schedule --name schedule-name
 \
--action-after-completion 'DELETE' \
--schedule-expression 'rate(1 hour)' \
--end-date '2024-01-01T00:00:00'
--target file://target-configuration.json \
--flexible-time-window '{ "Mode": "OFF"}' \
```

수동 삭제

일정이 더 이상 필요하지 않으면 DeleteSchedule 작업으로 삭제할 수 있습니다.

Example AWS CLI

```
$ aws scheduler delete-schedule --name your-schedule
```

Example Python SDK

```
import boto3
scheduler = boto3.client('scheduler')

scheduler.delete_schedule(Name="your-schedule")
```

다음 단계

- Lambda 및 Step Functions의 템플릿 기반 대상을 구성하는 방법에 대한 자세한 내용과 범용 대상 파라미터 사용에 대한 자세한 내용은 대상 관리 섹션을 참조하세요.
- EventBridge 스케줄러 데이터 유형과 API 작업에 대한 자세한 내용은 EventBridge 스케줄러 API 참조를 참조하세요.

일정 그룹 관리

일정 그룹은 일정을 구성하는 데 사용하는 Amazon EventBridge 스케줄러 리소스입니다.

AWS 계정에는 `default` 스케줄러 그룹이 함께 제공됩니다. 새 일정을 `default` 그룹 또는 직접 만들고 관리하는 일정 그룹과 연결할 수 있습니다. AWS 계정에서 최대 500개의 일정 그룹을 만들 수 있습니다. EventBridge 스케줄러를 사용하면 태그를 적용하여 개별 일정 대신 일정 그룹을 구성할 수 있습니다.

태그는 대소문자를 구분하는 키와 사용자가 정의하는 대소문자를 구분하는 값으로 구성된 레이블입니다. 태그를 생성하여 용도, 소유자 또는 환경 기준으로 일정을 분류할 수 있습니다. 예를 들어, `environment:production` 태그를 사용하여 일정이 속한 환경을 식별할 수 있습니다.

> **Important**
>
> 개인 식별 정보(PII)나 기타 기밀 정보 또는 민감한 정보를 태그에 추가하지 않습니다. 청구를 비롯한 여러 AWS 서비스에서 태그에 액세스할 수 있습니다. 태그는 개인 데이터나 민감한 데이터에 사용하기 위한 것이 아닙니다.

일정 그룹에는 활성 및 삭제라는 두 가지 상태가 있을 수 있습니다.

그룹을 처음 생성하면 그룹이 기본적으로 `ACTIVE`입니다. `ACTIVE` 그룹에 일정을 추가할 수 있습니다. 그룹을 삭제하면 EventBridge 스케줄러가 관련 일정 삭제를 완료할 때까지 상태가 `DELETING`으로 변경됩니다. EventBridge 스케줄러가 그룹에서 일정을 삭제한 후에는 해당 그룹을 사용자 계정에서 더 이상 사용할 수 없습니다.

다음 항목을 사용하여 일정 그룹을 만들고 여기에 태그를 적용하십시오. 또한 일정을 그룹에 연결하고 마지막으로 그룹을 삭제합니다.

주제

- 일정 그룹 생성
- 일정 그룹 삭제
- 관련 리소스

일정 그룹 생성

일정 그룹과 태그 지정을 사용하여 공통 목적을 공유하거나 동일한 환경에 속하는 일정을 구성할 수 있습니다. 다음 단계에서는 새 일정 그룹을 작성하고 태그를 사용하여 레이블을 지정합니다. 그런 다음 새 일정을 해당 그룹에 연결합니다.

> **Note**
>
> 그룹을 만든 후에는 해당 그룹에서 일정을 제거하거나 일정을 다른 그룹에 연결할 수 없습니다. 일정을 처음 만들 때만 일정을 그룹과 연결할 수 있습니다.

1단계: 새 일정 그룹 생성

다음 주제에서는 새 일정 그룹을 작성하고 이 그룹에 `environment:development` 태그를 사용하여 레이블을 지정하는 방법에 대해 설명합니다.

AWS Management Console

AWS Management Console를 사용하여 새 그룹을 생성하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/events/에서 Amazon EventBridge 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 일정 그룹을 선택합니다.
3. 일정 그룹 페이지에서 일정 그룹 생성을 선택합니다.
4. 일정 그룹 세부 정보 섹션의 이름에 그룹 이름을 입력합니다. 예: **`TestGroup`**.
5. 태그 섹션에서 다음을 수행합니다.
   a. Add new tag(새 태그 추가가)를 선택합니다.
   b. 키에는 이 키에 할당하려는 이름을 입력합니다. 이 자습서에서 이 일정 그룹이 속한 환경에 레이블을 지정하려면 **`environment`**를 입력하십시오.
   c. 값(선택 사항)에 키에 할당할 값을 입력합니다. 이 자습서에서는 환경 키에 **`development`** 값을 입력합니다.

   > **Note**
   >
   > 그룹을 생성한 후 그룹에 추가 태그를 추가할 수 있습니다.

6. 종료하려면 일정 그룹 생성을 선택합니다. 새 그룹이 일정 그룹 목록에 표시됩니다.
7. (선택 사항) 그룹을 편집하거나 그룹 태그를 관리하려면 새 그룹의 확인란을 선택하고 편집을 선택합니다.

> **Note**
>
> `default` 일정 그룹은 편집할 수 없습니다.

AWS CLI

AWS CLI를 사용하여 새 그룹을 생성하려면

1. 새 명령 프롬프트 창을 엽니다.
2. AWS Command Line Interface(AWS CLI)에서 다음 `create-schedule-group` 명령을 입력하여 새 그룹을 생성합니다. 이 명령은 태그가 `environment:development` 하나인 그룹을 생성합니다. 이 태그 또는 유사한 태그 지정 시스템을 사용하여 일정 그룹이 속한 환경에 따라 일정 그룹에 레이블을 지정할 수 있습니다.

   일정 이름과 태그 키 및 값을 사용자 정보로 바꾸십시오.

```
$ aws scheduler create-schedule-group --name TestGroup --tags
 Key=environment,Value=development
```

기본적으로 새 그룹은 `ACTIVE` 상태입니다. 이제 새로 만든 그룹에 새 일정을 연결할 수 있습니다.

2단계: 일정을 그룹과 연결하기

다음 단계를 사용하여 이전 단계에서 만든 그룹에 새 일정을 연결합니다.

AWS Management Console

AWS Management Console를 사용하여 일정을 그룹과 연결하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/events/에서 Amazon EventBridge 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 일정을 선택합니다.
3. 일정 테이블에서 일정 만들기를 선택하여 새 일정을 만듭니다.

4. 일정 세부 정보 지정 페이지의 일정 그룹에 대해 드롭다운 목록에서 새 그룹의 이름을 선택합니다. 예를 들어, TestGroup을 선택합니다.
5. 일정 패턴, 대상, 설정을 지정한 다음 일정 검토 및 저장 페이지에서 선택 사항을 검토하십시오. 새 일정 구성에 대한 자세한 내용은 시작하기 섹션을 참조하세요.
6. 일정을 완료하고 저장하려면 일정 저장을 선택합니다.

AWS CLI

AWS CLI를 사용하여 일정을 그룹과 연결하려면

1. 새 명령 프롬프트 창을 엽니다.
2. AWS Command Line Interface(AWS CLI)에서 `create-schedule` 명령을 입력합니다. 이렇게 하면 일정이 생성되고 이전 단계에서 만든 `sqs-test-schedule`이라는 그룹과 일정이 연결됩니다. 이 일정은 템플릿 기반 Amazon SQS 대상 유형을 사용하여 SendMessage 작업을 간접적으로 호출합니다. 일정 이름, 대상 및 그룹 이름을 사용자 정보로 대체하십시오.

```
$ aws scheduler create-schedule --name sqs-test-schedule --schedule-expression
 'rate(5 minutes)' \
--target '{"RoleArn": "ROLE_ARN", "Arn": "QUEUE_ARN", "Input": "TEST_PAYLOAD" }'
 \
--group-name TestGroup
--flexible-time-window '{ "Mode": "OFF"}'
```

이제 새 일정이 TestGroup 일정 그룹과 연결됩니다.

일정 그룹 삭제

다음에서 AWS Management Console 및 AWS Command Line Interface를 사용하여 일정 그룹을 삭제하는 방법을 알아볼 수 있습니다. 그룹을 삭제하면 EventBridge 스케줄러가 그룹 내의 모든 일정을 삭제하기 전까지는 그룹이 DELETING 상태가 됩니다. EventBridge 스케줄러가 그룹에서 일정을 삭제한 후에는 해당 그룹을 사용자 계정에서 더 이상 사용할 수 없습니다.

> **Note**
>
> 그룹을 만든 후에는 해당 그룹에서 일정을 제거하거나 일정을 다른 그룹에 연결할 수 없습니다. 일정을 처음 만들 때만 일정을 그룹과 연결할 수 있습니다.

AWS Management Console

AWS Management Console를 사용하여 그룹을 삭제하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/events/에서 Amazon EventBridge 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 일정 그룹을 선택합니다.
3. 일정 그룹 페이지에서 현재 AWS 리전의 기존 그룹 목록에서 삭제하려는 그룹을 찾습니다. 찾고 있는 그룹이 보이지 않으면 다른 AWS 리전을 선택하세요.

> **Note**
>
> 기본 그룹은 삭제하거나 편집할 수 없습니다.

4. 삭제하려는 그룹의 확인란을 선택합니다.
5. 삭제를 선택합니다.
6. 일정 그룹 삭제 대화 상자에서 그룹 이름을 입력하여 선택을 확인한 다음 삭제를 선택합니다.
7. 일정 그룹 목록에서 상태 열이 변경되어 그룹이 현재 삭제 중임을 나타냅니다. EventBridge 스케줄러가 그룹과 관련된 모든 일정을 삭제할 때까지 그룹은 이 상태를 유지합니다.
8. 목록을 새로 고치고 그룹이 삭제되었는지 확인하려면 새로 고침 아이콘을 선택합니다.

AWS CLI

AWS CLI를 사용하여 그룹을 삭제하려면

1. 새 명령 프롬프트 창을 엽니다.
2. AWS Command Line Interface(AWS CLI)에서 다음 `delete-schedule-group` 명령을 입력하여 일정 그룹을 삭제합니다. `--name`의 값을 사용자 정보로 바꾸십시오.

```
$ aws scheduler delete-schedule-group --name TestGroup
```

   성공한 경우 이 AWS CLI 작업은 응답을 반환하지 않습니다.

3. 그룹이 `DELETING` 상태에 있는지 확인하려면 다음 `get-schedule-group` 명령을 실행합니다.

```
$ aws scheduler get-schedule-group --name TestGroup
```

성공적으로 실행되면, 다음과 비슷한 출력이 표시됩니다.

```
{
    "Arn": "arn:aws::scheduler:us-west-2:123456789012:schedule-group/TestGroup",
    "CreationDate": "2023-01-01T09:00:00.000000-07:00",
    "LastModificationDate": "2023-01-01T09:00:00.000000-07:00",
    "Name": "TestGroup",
    "State": "DELETING"
}
```

EventBridge 스케줄러는 그룹과 관련된 일정을 삭제한 후 그룹을 삭제합니다. `get-schedule-group`을 다시 실행하면 `ResourceNotFoundException` 응답을 받게 됩니다.

```
An error occurred (ResourceNotFoundException) when calling the GetScheduleGroup
 operation: Schedule group TestGroup does not exist.
```

관련 리소스

일정 그룹에 대한 자세한 내용은 다음 리소스를 참조하세요.

- EventBridge 스케줄러 API 참조의 CreateScheduleGroup 작업.
- EventBridge 스케줄러 API 참조의 DeleteScheduleGroup 작업.

대상 관리

다음 항목에서는 EventBridge 스케줄러에서 템플릿 기반 및 범용 대상을 사용하는 방법을 설명하고 EventBridge 스케줄러의 범용 대상 파라미터를 사용하여 구성할 수 있는 지원되는 AWS 서비스 목록을 제공합니다.

템플릿 기반 대상은 Amazon SQS, Lambda 및 Step Functions와 같은 핵심 AWS 서비스 그룹 전반에 걸친 공통 API 작업 세트입니다. 예를 들어, 함수 ARN을 제공하여 Lambda의 Invoke API 작업을 대상으로 지정하거나 Amazon SQS의 `SendMessage` 작업을 대상의 대기열 ARN으로 지정할 수 있습니다.

범용 대상은 여러 AWS 서비스에 대해 더 광범위한 API 작업을 간접적으로 호출할 수 있는 사용자 지정 가능한 파라미터 세트입니다. 예를 들어 EventBridge 스케줄러의 범용 대상 파라미터(UTP)를 사용하면 `CreateQueue` 작업을 사용하여 새 Amazon SQS 대기열을 생성할 수 있습니다.

템플릿 기반 대상이나 범용 대상을 구성하려면 일정에 대상으로 구성한 API 작업을 직접적으로 호출할 수 있는 권한이 있어야 합니다. 일정의 실행 역할에 필요한 권한을 연결하여 이 작업을 수행할 수 있습니다. 예를 들어 Amazon SQS의 `SendMessage` 작업을 대상으로 지정하려면 실행 역할에 `sqs:SendMessage` 작업을 수행할 권한을 부여해야 합니다. 대부분의 경우 대상 서비스가 지원하는 AWS 관리형 정책을 사용하여 필요한 권한을 추가할 수 있습니다. 하지만 자체 고객 관리형 정책을 만들거나 실행 역할에 연결된 기존 정책에 인라인 권한을 추가할 수도 있습니다. 다음 항목에서는 템플릿 기반 대상 유형과 범용 대상 유형 모두에 권한을 추가하는 예를 보여줍니다.

일정의 실행 역할 설정에 대한 자세한 내용은 the section called “실행 역할 설정” 섹션을 참조하세요.

주제

템플릿 기반 대상 사용

템플릿 타겟은 Amazon SQS, Lambda 및 Step Functions와 같은 핵심 AWS 서비스 그룹 전반에 걸친 일반적인 API 작업 세트입니다. 예를 들어, 함수 ARN을 제공하여 Lambda의 `Invoke` 작업을 대상으로 지정하거나 대기열 ARN을 사용하여 Amazon SQS의 `SendMessage` 작업을 대상으로 지정할 수 있습니다. 템플릿 기반 대상을 구성하려면 일정의 실행 역할에 대상 API 작업을 수행할 수 있는 권한도 부여해야 합니다.

Scheduler AWS CLI SDK나 EventBridge Scheduler SDK 중 하나를 사용하여 템플릿 대상을 프로그래밍 방식으로 구성하려면 실행 역할의 ARN, 대상 리소스의 ARN, EventBridge Scheduler가 대상에 전달할 선택적 입력, 일부 템플리트 대상의 경우 해당 대상에 대한 추가 구성 옵션이 포함된 고유한 매개변수 세트를 지정해야 합니다. 템플릿이 있는 대상 리소스에 ARN을 지정하면 EventBridge Scheduler는 해당 서비스에 대해 지원되는 API 작업을 호출하려는 것으로 자동으로 가정합니다. EventBridge Scheduler가 서비스의 다른 API 작업을 대상으로 지정하도록 하려면 대상을 범용 대상으로 구성해야 합니다.

다음은 EventBridge Scheduler가 지원하는 모든 템플릿 대상 및 해당하는 경우 각 대상의 고유한 관련 매개 변수 집합의 전체 목록입니다. 각 매개 변수 집합의 링크를 선택하면 EventBridge Scheduler API 참조의 필수 및 선택적 필드를 볼 수 있습니다.

- CodeBuild – `StartBuild`
- CodePipeline – `StartPipelineExecution`
- Amazon ECS – `RunTask`
  - 파라미터: `EcsParameters`
- EventBridge – `PutEvents`
  - 파라미터: `EventBridgeParameters`
- Amazon Inspector – `StartAssessmentRun`
- Kinesis – `PutRecord`
  - 파라미터: `KinesisParameters`
- Firehose — **`PutRecord`**
- Lambda – `Invoke`
- SageMaker – `StartPipelineExecution`
  - 파라미터: `SageMakerPipelineParameters`
- Amazon SNS – `Publish`
- Amazon SQS – `SendMessage`
  - 파라미터: `SqsParameters`
- Step Functions – `StartExecution`

다음 예제를 사용하여 다양한 템플릿 기반 대상을 구성하는 방법과 설명된 각 대상에 필요한 IAM 권한을 알아보십시오.

Amazon SQS **SendMessage**

Example 실행 역할에 대한 권한 정책

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "sqs:SendMessage"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

Example AWS CLI

```
$ aws scheduler create-schedule --name sqs-templated --schedule-expression 'rate(5
 minutes)' \
--target '{"RoleArn": "ROLE_ARN", "Arn":"QUEUE_ARN", "Input": "Message for scheduleArn:
 '<aws.scheduler.schedule-arn>', scheduledTime: '<aws.scheduler.scheduled-time>'" }' \
--flexible-time-window '{ "Mode": "OFF"}'
```

Example Python SDK

```
import boto3
scheduler = boto3.client('scheduler')

flex_window = { "Mode": "OFF" }

sqs_templated = {
    "RoleArn": "<ROLE_ARN>",
    "Arn": "<QUEUE_ARN>",
    "Input": "Message for scheduleArn: '<aws.scheduler.schedule-arn>', scheduledTime:
 '<aws.scheduler.scheduled-time>'"
}

scheduler.create_schedule(
    Name="sqs-python-templated",
    ScheduleExpression="rate(5 minutes)",
```

```
    Target=sqs_templated,
    FlexibleTimeWindow=flex_window)
```

Example Java SDK

```
package com.example;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.scheduler.SchedulerClient;
import software.amazon.awssdk.services.scheduler.model.*;


public class MySchedulerApp {

    public static void main(String[] args) {

        final SchedulerClient client = SchedulerClient.builder()
                .region(Region.US_WEST_2)
                .build();

        Target sqsTarget = Target.builder()
                .roleArn("<ROLE_ARN>")
                .arn("<QUEUE_ARN>")
                .input("Message for scheduleArn: '<aws.scheduler.schedule-arn>',
 scheduledTime: '<aws.scheduler.scheduled-time>'")
                .build();

        CreateScheduleRequest createScheduleRequest = CreateScheduleRequest.builder()
                .name("<SCHEDULE NAME>")
                .scheduleExpression("rate(10 minutes)")
                .target(sqsTarget)
                .flexibleTimeWindow(FlexibleTimeWindow.builder()
                        .mode(FlexibleTimeWindowMode.OFF)
                        .build())
                .build();

        client.createSchedule(createScheduleRequest);
        System.out.println("Created schedule with rate expression and an Amazon SQS
 templated target");
    }
}
```

Lambda **Invoke**

Example 실행 역할에 대한 권한 정책

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

Example AWS CLI

```
$ aws scheduler create-schedule --name lambda-templated-schedule --schedule-expression
 'rate(5 minutes)' \
--target '{"RoleArn": "ROLE_ARN", "Arn":"FUNCTION_ARN", "Input": "{ \"Payload\":
 \"TEST_PAYLOAD\" }" }' \
--flexible-time-window '{ "Mode": "OFF"}'
```

Example Python SDK

```
import boto3
scheduler = boto3.client('scheduler')

flex_window = { "Mode": "OFF" }

lambda_templated = {
    "RoleArn": "<ROLE_ARN>",
    "Arn": "<LAMBDA_ARN>",
    "Input": "{ 'Payload': 'TEST_PAYLOAD' }"}
}

scheduler.create_schedule(
    Name="lambda-python-templated",
    ScheduleExpression="rate(5 minutes)",
    Target=lambda_templated,
```

```
    FlexibleTimeWindow=flex_window)
```

Example Java SDK

```
package com.example;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.scheduler.SchedulerClient;
import software.amazon.awssdk.services.scheduler.model.*;


public class MySchedulerApp {

    public static void main(String[] args) {

        final SchedulerClient client = SchedulerClient.builder()
                .region(Region.US_WEST_2)
                .build();

        Target lambdaTarget = Target.builder()
                .roleArn("<ROLE_ARN>")
                .arn("<Lambda ARN>")
                .input("{ 'Payload': 'TEST_PAYLOAD' }")
                .build();

        CreateScheduleRequest createScheduleRequest = CreateScheduleRequest.builder()
                .name("<SCHEDULE_NAME>")
                .scheduleExpression("rate(10 minutes)")
                .target(lambdaTarget)
                .flexibleTimeWindow(FlexibleTimeWindow.builder()
                        .mode(FlexibleTimeWindowMode.OFF)
                        .build())
                .clientToken("<Token GUID>")
                .build();

        client.createSchedule(createScheduleRequest);
        System.out.println("Created schedule with rate expression and Lambda templated
 target");
    }
}
```

Step Functions **`StartExecution`**

Example 실행 역할에 대한 권한 정책

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "states:StartExecution"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

Example AWS CLI

```
$ aws scheduler create-schedule --name sfn-templated-schedule --schedule-expression
 'rate(5 minutes)' \
--target '{"RoleArn": "ROLE_ARN", "Arn":"STATE_MACHINE_ARN", "Input": "{ \"Payload\":
 \"TEST_PAYLOAD\" }" }' \
--flexible-time-window '{ "Mode": "OFF"}'
```

Example Python SDK

```
import boto3
scheduler = boto3.client('scheduler')

flex_window = { "Mode": "OFF" }

sfn_templated= {
    "RoleArn": "<ROLE_ARN>",
    "Arn": "<STATE_MACHINE_ARN>",
    "Input": "{ 'Payload': 'TEST_PAYLOAD' }"
}

scheduler.create_schedule(Name="sfn-python-templated",
    ScheduleExpression="rate(5 minutes)",
    Target=sfn_templated,
    FlexibleTimeWindow=flex_window)
```

Example Java SDK

```
package com.example;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.scheduler.SchedulerClient;
import software.amazon.awssdk.services.scheduler.model.*;


public class MySchedulerApp {

    public static void main(String[] args) {

        final SchedulerClient client = SchedulerClient.builder()
                .region(Region.US_WEST_2)
                .build();

        Target stepFunctionsTarget = Target.builder()
                .roleArn("<ROLE_ARN>")
                .arn("<STATE_MACHINE_ARN>")
                .input("{ 'Payload': 'TEST_PAYLOAD' }")
                .build();

        CreateScheduleRequest createScheduleRequest = CreateScheduleRequest.builder()
                .name("<SCHEDULE_NAME>")
                .scheduleExpression("rate(10 minutes)")
                .target(stepFunctionsTarget)
                .flexibleTimeWindow(FlexibleTimeWindow.builder()
                        .mode(FlexibleTimeWindowMode.OFF)
                        .build())
                .clientToken("<Token GUID>")
                .build();

        client.createSchedule(createScheduleRequest);
        System.out.println("Created schedule with rate expression and Step Function
 templated target");
    }
}
```

범용 대상 사용

범용 대상은 여러 AWS 서비스에 대해 더 광범위한 API 작업을 간접적으로 호출할 수 있는 사용자 지정 가능한 파라미터 세트입니다. 예를 들어 범용 대상 파라미터(UTP)를 사용하면 `CreateQueue` 작업을 사용하여 새 Amazon SQS 대기열을 생성할 수 있습니다.

AWS CLI 또는 EventBridge 스케줄러 SDK 중 하나를 사용하여 일정에 대한 범용 대상을 구성하려면 다음 정보를 지정해야 합니다.

- RoleArn - 대상에 사용할 실행 역할의 ARN입니다. 지정하는 실행 역할에는 일정에서 대상으로 지정할 API 작업을 호출할 수 있는 권한이 있어야 합니다.
- Arn — 대상으로 지정하려는 API 작업을 포함한 전체 서비스 ARN으로 형식은 `arn:aws:scheduler:::aws-sdk:`*`service`*`:`*`apiAction`*과 같습니다.

  예를 들어, Amazon SQS의 경우 지정한 서비스 이름은 `arn:aws:scheduler:::aws-sdk:`**`sqs:sendMessage`**입니다.
- 입력 - EventBridge 스케줄러가 대상 API로 전송하는 요청 파라미터를 사용하여 사용자가 지정하는 올바른 형식의 JSON입니다. `Input`에서 설정하는 JSON의 파라미터와 모양은 일정이 간접적으로 호출하는 서비스 API에 따라 결정됩니다. 이 정보를 찾으려면 대상으로 지정하려는 서비스에 대한 API 참조를 참조하세요.

지원되지 않는 옵션

EventBridge 스케줄러는 일반적인 `GET` 작업과 같이 다음 접두사 목록으로 시작하는 읽기 전용 API 작업을 지원하지 않습니다.

```
get
describe
list
poll
receive
search
scan
query
select
read
lookup
discover
validate
```

```
batchGet
batchDescribe
batchRead
transactGet
adminGet
adminList
testMigration
retrieve
testConnection
translateDocument
isAuthorized
isAuthorizedWithToken
invokeModel
```

예를 들어 GetQueueUrl API 작업에 대한 서비스 ARN은 arn:aws:scheduler:::aws-sdk:sqs:**getQueueURL**과 같습니다. API 작업은 get 접두사로 시작하므로 EventBridge 스케줄러는 이 대상을 지원하지 않습니다. 마찬가지로 작업이 list 접두사로 시작하기 때문에 Amazon MQ 작업 ListBrokers는 대상으로 지원되지 않습니다.

범용 대상을 사용하는 예

일정 Input 필드에 전달하는 파라미터는 간접적으로 호출하려는 서비스 API가 수락하는 요청 파라미터에 따라 달라집니다. 예를 들어, Lambda Invoke를 대상으로 지정하려면 AWS Lambda API 참조에 나열된 파라미터를 설정할 수 있습니다. 여기에는 Lambda 함수로 전달할 수 있는 선택적 JSON 페이로드가 포함됩니다.

다양한 API에 설정할 수 있는 파라미터를 결정하려면 해당 서비스의 API 참조를 참조하세요. Lambda Invoke와 마찬가지로 일부 API는 요청 본문 페이로드뿐만 아니라 URI 파라미터도 허용합니다. 이러한 경우 일정 Input에 URI 경로 파라미터와 JSON 페이로드를 지정합니다.

다음 예는 범용 대상을 사용하여 Lambda, Amazon SQS 및 Step Functions에서 일반적인 API 작업을 간접적으로 호출하는 방법을 보여줍니다.

Example Lambda

```
$ aws scheduler create-schedule --name lambda-universal-schedule --schedule-expression
 'rate(5 minutes)' \
--target '{"RoleArn": "ROLE_ARN", "Arn":"arn:aws:scheduler:::aws-sdk:lambda:invoke"
 "Input": "{\"FunctionName\":\"arn:aws:lambda:REGION:123456789012:function:HelloWorld
\",\"InvocationType\":\"Event\",\"Payload\":\"{\\\"message\\\":\\\"testing function\\
\"}\"}" }' \
```

```
--flexible-time-window '{ "Mode": "OFF"}'
```

Example Amazon SQS

```
import boto3
scheduler = boto3.client('scheduler')

flex_window = { "Mode": "OFF" }

sqs_universal= {
    "RoleArn": "<ROLE_ARN>",
    "Arn": "arn:aws:scheduler:::aws-sdk:sqs:sendMessage",
    "Input": "{\"MessageBody\":\"My message\",\"QueueUrl\":\"<QUEUE_URL>\"}"}
}

scheduler.create_schedule(
    Name="sqs-sdk-test",
    ScheduleExpression="rate(5 minutes)",
    Target=sqs_universal,
    FlexibleTimeWindow=flex_window)
```

Example Step Functions

```
package com.example;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.scheduler.SchedulerClient;
import software.amazon.awssdk.services.scheduler.model.*;


public class MySchedulerApp {

    public static void main(String[] args) {

        final SchedulerClient client = SchedulerClient.builder()
                .region(Region.US_WEST_2)
                .build();

        Target stepFunctionsUniversalTarget = Target.builder()
                .roleArn("<ROLE_ARN>")
                .arn("arn:aws:scheduler:::aws-sdk:sfn:startExecution")
                .input("{\"Input\":\"{}\",\"StateMachineArn\":\"<STATE_MACHINE_ARN>
\"}")
```

```
                .build();

        CreateScheduleRequest createScheduleRequest = CreateScheduleRequest.builder()
                .name("<SCHEDULE_NAME>")
                .scheduleExpression("rate(10 minutes)")
                .target(stepFunctionsUniversalTarget)
                .flexibleTimeWindow(FlexibleTimeWindow.builder()
                        .mode(FlexibleTimeWindowMode.OFF)
                        .build())
                .clientToken("<Token GUID>")
                .build();

        client.createSchedule(createScheduleRequest);
        System.out.println("Created schedule with rate expression and Step Function
 universal target");
    }
}
```

컨텍스트 속성 추가

대상에 전달하는 페이로드에서 다음 키워드를 사용하여 일정에 대한 메타데이터를 수집하십시오. EventBridge 스케줄러는 스케줄이 대상을 간접적으로 호출할 때 각 키워드를 해당 값으로 대체합니다.

- **<aws.scheduler.schedule-arn>** - 일정의 ARN입니다.
- **<aws.scheduler.scheduled-time>** - 일정에서 대상을 간접적으로 호출하도록 지정한 시간(예: 2022-03-22T18:59:43Z)입니다.
- **<aws.scheduler.execution-id>** - EventBridge 스케줄러가 각 대상 호출에 할당하는 고유 ID(예: d32c5kddcf5bb8c3)입니다.
- **<aws.scheduler.attempt-number>** - 현재 호출에 대한 시도 횟수를 식별하는 카운터(예: 1)입니다.

이 예제에서는 5분마다 실행되고 Amazon SQS SendMessage 작업을 범용 대상으로 간접적으로 호출하는 일정을 생성하는 방법을 보여줍니다. 메시지 본문에는 schedule-time의 값이 포함됩니다.

Example AWS CLI

```
$ aws scheduler create-schedule --name your-schedule \
    --schedule-expression 'rate(5 minutes)' \
    --target '{"RoleArn": "ROLE_ARN", \
```

```
        "Arn": "arn:aws:scheduler:::aws-sdk:sqs:sendMessage", \
        "Input": "{\"MessageBody\":\"<aws.scheduler.scheduled-time>\",\"QueueUrl\":
\"https://sqs.us-west-2.amazonaws.com/123456789012/scheduler-cli-test\"}"}' \
    --flexible-time-window '{ "Mode": "OFF"}'
```

Example Python SDK

```
import boto3
scheduler = boto3.client('scheduler')

sqs_universal= {
    "RoleArn": "<ROLE_ARN>",
    "Arn": "arn:aws:scheduler:::aws-sdk:sqs:sendMessage",
    "Input": "{\"MessageBody\":\"<aws.scheduler.scheduled-time>\",\"QueueUrl\":
\"https://sqs.us-west-2.amazonaws.com/123456789012/scheduler-cli-test\"}"
}

flex_window = { "Mode": "OFF" }


scheduler.update_schedule(Name="your-schedule",
    ScheduleExpression="rate(5 minutes)",
    Target=sqs_universal,
    FlexibleTimeWindow=flex_window)
```

다음 단계

EventBridge 스케줄러 데이터 유형 및 API 작업에 대한 자세한 내용은 EventBridge 스케줄러 API 참조를 참조하세요.

Amazon EventBridge 스케줄러의 보안

클라우드 AWS 보안이 최우선 과제입니다. AWS 고객은 가장 보안에 민감한 조직의 요구 사항을 충족하도록 구축된 데이터 센터 및 네트워크 아키텍처를 활용할 수 있습니다.

보안은 기업과 기업 간의 공동 책임입니다. AWS 공동 책임 모델은 이 사항을 클라우드의 보안 및 클라우드 내 보안으로 설명합니다.

- 클라우드 보안 - AWS 클라우드에서 AWS 서비스를 실행하는 인프라를 보호하는 역할을 합니다 AWS 클라우드. AWS 또한 안전하게 사용할 수 있는 서비스를 제공합니다. Amazon EventBridge Scheduler에 적용되는 규정 준수 프로그램에 대해 자세히 알아보려면 규정 준수 프로그램별 범위 내 서비스 규정 준수 참조하십시오.
- 클라우드에서의 보안 — 귀하의 책임은 사용하는 AWS 서비스에 따라 결정됩니다. 또한 귀하는 귀사의 데이터의 민감도, 귀사의 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다.

이 설명서는 EventBridge Scheduler를 사용할 때 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다. 다음 항목에서는 보안 및 규정 준수 목표를 충족하도록 EventBridge 스케줄러를 구성하는 방법을 보여줍니다. 또한 EventBridge Scheduler 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스를 사용하는 방법도 알아봅니다.

주제

- Amazon EventBridge 스케줄러에 대한 액세스 관리
- Amazon EventBridge 스케줄러에서의 데이터 보호
- Amazon EventBridge 스케줄러에 대한 규정 준수 검증
- Amazon EventBridge 스케줄러의 레질리언스
- Amazon EventBridge 스케줄러의 인프라 보안

Amazon EventBridge 스케줄러에 대한 액세스 관리

AWS Identity and Access Management (IAM) 은 관리자가 리소스에 대한 액세스를 안전하게 제어할 수 AWS 서비스 있도록 도와줍니다. AWS IAM 관리자는 Scheduler 리소스를 사용할 EventBridge 수 있는 인증 (로그인) 및 권한 부여 (권한 보유) 를 받을 수 있는 사용자를 제어합니다. IAM은 추가 비용 AWS 서비스 없이 사용할 수 있습니다.

주제

고객

스케줄러에서 수행하는 작업에 따라 AWS Identity and Access Management (IAM) 사용 방식이 다릅니다. EventBridge

서비스 사용자 - EventBridge 스케줄러 서비스를 사용하여 작업을 수행하는 경우 관리자가 필요한 자격 증명과 권한을 제공합니다. 더 많은 EventBridge 스케줄러 기능을 사용하여 작업을 수행함에 따라 추가 권한이 필요할 수 있습니다. 액세스 권한 관리 방식을 이해하면 적절한 권한을 관리자에게 요청할 수 있습니다. EventBridge 스케줄러의 기능에 액세스할 수 없는 경우 을 참조하십시오. Amazon EventBridge 스케줄러 자격 증명 및 액세스 문제 해결

서비스 관리자 — 회사에서 EventBridge 스케줄러 리소스를 담당하는 경우 스케줄러에 EventBridge 대한 전체 액세스 권한이 있을 수 있습니다. 서비스 사용자가 액세스해야 하는 EventBridge 스케줄러 기능과 리소스를 결정하는 것은 여러분의 몫입니다. 그런 다음, IAM 관리자에게 요청을 제출하여 서비스 사용자의 권한을 변경해야 합니다. 이 페이지의 정보를 검토하여 IAM의 기본 개념을 이해합니다. 회사에서 EventBridge Scheduler와 함께 IAM을 사용하는 방법에 대한 자세한 내용은 을 참조하십시오. EventBridge 스케줄러와 IAM의 작동 방식

IAM 관리자 - IAM 관리자인 경우 Scheduler에 대한 액세스를 관리하기 위한 정책을 작성하는 방법에 대해 자세히 알아보는 것이 좋습니다. EventBridge IAM에서 사용할 수 있는 EventBridge 스케줄러 ID 기반 정책의 예를 보려면 을 참조하십시오. 자격 증명 기반 정책 사용

자격 증명을 통한 인증

인증은 ID 자격 증명을 AWS 사용하여 로그인하는 방법입니다. IAM 사용자로 인증 (로그인 AWS) 하거나 IAM 역할을 맡아 인증 (로그인) 해야 합니다. AWS 계정 루트 사용자

ID 소스를 통해 제공된 자격 증명을 사용하여 페더레이션 ID로 로그인할 수 있습니다. AWS AWS IAM Identity Center (IAM ID 센터) 사용자, 회사의 싱글 사인온 인증, Google 또는 Facebook 자격 증명이 페더레이션 ID의 예입니다. 페더레이션형 ID로 로그인할 때 관리자가 이전에 IAM 역할을 사용하여 ID 페더레이션을 설정했습니다. 페더레이션을 사용하여 액세스하는 경우 AWS 간접적으로 역할을 맡게 됩니다.

사용자 유형에 따라 AWS Management Console 또는 AWS 액세스 포털에 로그인할 수 있습니다. 로그인에 대한 자세한 내용은 AWS 로그인 사용 설명서의 내 로그인 방법을 참조하십시오. AWS AWS 계정

AWS 프로그래밍 방식으로 액세스하는 경우 자격 증명을 사용하여 요청에 암호화 방식으로 서명할 수 있는 소프트웨어 개발 키트 (SDK) 와 명령줄 인터페이스 (CLI) 를 AWS 제공합니다. AWS 도구를 사용하지 않는 경우 요청에 직접 서명해야 합니다. 권장 방법을 사용하여 직접 요청에 서명하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 AWS API 요청 서명을 참조하십시오.

사용하는 인증 방법에 상관없이 추가 보안 정보를 제공해야 할 수도 있습니다. 예를 들어, AWS 계정의 보안을 강화하기 위해 다단계 인증 (MFA) 을 사용할 것을 권장합니다. 자세한 내용은 AWS IAM Identity Center 사용 설명서의 다중 인증 및 IAM 사용자 설명서의 AWS에서 다중 인증(MFA) 사용을 참조합니다.

AWS 계정 루트 사용자

계정을 AWS 계정만들 때는 먼저 계정의 모든 AWS 서비스 리소스에 대한 완전한 액세스 권한을 가진 하나의 로그인 ID로 시작합니다. 이 ID를 AWS 계정 루트 사용자라고 하며, 계정을 만들 때 사용한 이메일 주소와 비밀번호로 로그인하여 액세스할 수 있습니다. 일상적인 작업에 루트 사용자를 사용하지 않을 것을 강력히 권장합니다. 루트 사용자 보안 인증 정보를 보호하고 루트 사용자만 수행할 수 있는 태스크를 수행하는 데 사용하세요. 루트 사용자로 로그인해야 하는 태스크의 전체 목록은 IAM 사용자 안내서의 루트 사용자 보안 인증이 필요한 태스크를 참조하세요.

페더레이션 자격 증명

가장 좋은 방법은 관리자 액세스가 필요한 사용자를 비롯한 수동 AWS 서비스 사용자가 ID 공급자와의 페더레이션을 사용하여 임시 자격 증명을 사용하여 액세스하도록 하는 것입니다.

페더레이션 ID는 기업 사용자 디렉토리, 웹 ID 공급자, Identity Center 디렉터리의 사용자 또는 ID 소스를 통해 제공된 자격 증명을 사용하여 액세스하는 AWS 서비스 모든 사용자를 말합니다. AWS Directory Service페더레이션 ID에 AWS 계정액세스하면 이들이 역할을 맡고 역할은 임시 자격 증명을 제공합니다.

중앙 집중식 액세스 관리를 위해 AWS IAM Identity Center을 사용하는 것이 좋습니다. IAM Identity Center에서 사용자 및 그룹을 생성하거나 자체 ID 소스의 사용자 및 그룹 집합에 연결하고 동기화하여

모든 사용자 및 애플리케이션에서 사용할 수 있습니다. AWS 계정 IAM Identity Center에 대한 자세한 내용은AWS IAM Identity Center 사용 설명서에서 IAM Identity Center란 무엇입니까?를 참조하세요.

IAM 사용자 및 그룹

IAM 사용자는 단일 사용자 또는 애플리케이션에 대한 특정 권한을 AWS 계정 가진 사용자 내 자격 증명입니다. 가능하면 암호 및 액세스 키와 같은 장기 자격 증명이 있는 IAM 사용자를 생성하는 대신 임시 자격 증명을 사용하는 것이 좋습니다. 하지만 IAM 사용자의 장기 자격 증명이 필요한 특정 사용 사례가 있는 경우 액세스 키를 교체하는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 장기 보안 인증이 필요한 사용 사례의 경우 정기적으로 액세스 키 교체를 참조하세요.

IAM 그룹은 IAM 사용자 컬렉션을 지정하는 자격 증명입니다. 사용자는 그룹으로 로그인할 수 없습니다. 그룹을 사용하여 여러 사용자의 권한을 한 번에 지정할 수 있습니다. 그룹을 사용하면 대규모 사용자 집합의 권한을 더 쉽게 관리할 수 있습니다. 예를 들어, IAMAdmins라는 그룹이 있고 이 그룹에 IAM 리소스를 관리할 권한을 부여할 수 있습니다.

사용자는 역할과 다릅니다. 사용자는 한 사람 또는 애플리케이션과 고유하게 연결되지만, 역할은 해당 역할이 필요한 사람이라면 누구나 수임할 수 있습니다. 사용자는 영구적인 장기 자격 증명을 가지고 있지만, 역할은 임시 보안 인증만 제공합니다. 자세한 정보는 IAM 사용 설명서의 IAM 사용자를 만들어야 하는 경우(역할이 아님)를 참조하세요.

IAM 역할

IAM 역할은 특정 권한을 가진 사용자 AWS 계정 내의 자격 증명입니다. IAM 사용자와 유사하지만, 특정 개인과 연결되지 않습니다. 역할을 AWS Management Console 전환하여 에서 일시적으로 IAM 역할을 맡을 수 있습니다. AWS CLI 또는 AWS API 작업을 호출하거나 사용자 지정 URL을 사용하여 역할을 수임할 수 있습니다. 역할 사용 방법에 대한 자세한 정보는 IAM 사용 설명서의 IAM 역할 사용을 참조하세요.

임시 보안 인증 정보가 있는 IAM 역할은 다음과 같은 상황에서 유용합니다.

- 페더레이션 사용자 액세스 - 페더레이션 아이덴티티에 권한을 부여하려면 역할을 생성하고 해당 역할의 권한을 정의합니다. 연동 자격 증명이 인증되면 역할이 연결되고 역할에 정의된 권한이 부여됩니다. 페더레이션 역할에 대한 자세한 내용은 IAM 사용 설명서의 타사 자격 증명 공급자의 역할 만들기를 참조하세요. IAM Identity Center를 사용하는 경우 권한 세트를 구성합니다. 인증 후 자격 증명이 액세스할 수 있는 항목을 제어하기 위해 IAM Identity Center는 권한 세트를 IAM의 역할과 연관짓습니다. 권한 세트에 대한 자세한 내용은AWS IAM Identity Center 사용 설명서의 권한 세트를 참조하세요.
- 임시 IAM 사용자 권한 - IAM 사용자 또는 역할은 IAM 역할을 수임하여 특정 태스크에 대한 다양한 권한을 임시로 받을 수 있습니다.

- 크로스 계정 액세스 - IAM 역할을 사용하여 다른 계정의 사용자(신뢰할 수 있는 보안 주체)가 내 계정의 리소스에 액세스하도록 허용할 수 있습니다. 역할은 계정 간 액세스를 부여하는 기본적인 방법입니다. 그러나 일부 AWS 서비스경우에는 역할을 프록시로 사용하는 대신 정책을 리소스에 직접 연결할 수 있습니다. 크로스 계정 액세스를 위한 역할과 리소스 기반 정책의 차이점을 알아보려면 IAM 사용 설명서의 IAM 역할과 리소스 기반 정책의 차이를 참조하세요.
- 서비스 간 액세스 — 일부는 다른 AWS 서비스서비스의 기능을 AWS 서비스 사용합니다. 예컨대, 어떤 서비스에서 호출을 수행하면 일반적으로 해당 서비스는 Amazon EC2에서 애플리케이션을 실행하거나 Amazon S3에 객체를 저장합니다. 서비스는 호출하는 보안 주체의 권한을 사용하거나, 서비스 역할을 사용하거나, 또는 서비스 연결 역할을 사용하여 이 작업을 수행할 수 있습니다.
  - 순방향 액세스 세션 (FAS) — IAM 사용자 또는 역할을 사용하여 작업을 수행하는 경우 보안 AWS 주체로 간주됩니다. 일부 서비스를 사용하는 경우 다른 서비스에서 다른 작업을 시작하는 작업을 수행할 수 있습니다. FAS는 전화를 거는 주체의 권한을 다운스트림 AWS 서비스서비스에 AWS 서비스 요청하기 위한 요청과 결합하여 사용합니다. FAS 요청은 다른 서비스 AWS 서비스 또는 리소스와의 상호 작용이 필요한 요청을 서비스가 수신한 경우에만 이루어집니다. 이 경우 두 작업을 모두 수행할 수 있는 권한이 있어야 합니다. FAS 요청 시 정책 세부 정보는 전달 액세스 세션을 참조하세요.
  - 서비스 역할 - 서비스 역할은 서비스가 사용자를 대신하여 태스크를 수행하기 위해 맡는 IAM 역할입니다. IAM 관리자는 IAM 내에서 서비스 역할을 생성, 수정 및 삭제할 수 있습니다. 자세한 정보는 IAM 사용자 설명서의 AWS 서비스에 대한 권한을 위임할 역할 생성을 참조합니다.
  - 서비스 연결 역할 — 서비스 연결 역할은 에 연결된 서비스 역할의 한 유형입니다. AWS 서비스서비스는 사용자를 대신하여 작업을 수행하기 위해 역할을 수임할 수 있습니다. 서비스 연결 역할은 사용자에게 AWS 계정 표시되며 해당 서비스가 소유합니다. IAM 관리자는 서비스 링크 역할의 권한을 볼 수 있지만 편집은 할 수 없습니다.
- Amazon EC2에서 실행되는 애플리케이션 — IAM 역할을 사용하여 EC2 인스턴스에서 실행되고 API 요청을 AWS CLI 하는 애플리케이션의 임시 자격 증명을 관리할 수 있습니다. AWS 이는 EC2 인스턴스 내에 액세스 키를 저장할 때 권장되는 방법입니다. EC2 인스턴스에 AWS 역할을 할당하고 모든 애플리케이션에서 사용할 수 있게 하려면 인스턴스에 연결된 인스턴스 프로필을 생성합니다. 인스턴스 프로파일에는 역할이 포함되어 있으며 EC2 인스턴스에서 실행되는 프로그램이 임시 보안 인증을 얻을 수 있습니다. 자세한 정보는 IAM 사용 설명서의 IAM 역할을 사용하여 Amazon EC2 인스턴스에서 실행되는 애플리케이션에 권한 부여를 참조하세요.

IAM 역할을 사용할지 또는 IAM 사용자를 사용할지를 알아보려면 IAM 사용자 설명서의 IAM 역할(사용자 대신)을 생성하는 경우를 참조합니다.

정책을 사용한 액세스 관리

정책을 생성하고 이를 AWS ID 또는 리소스에 AWS 연결하여 액세스를 제어할 수 있습니다. 정책은 ID 또는 리소스와 연결될 때 AWS 해당 권한을 정의하는 객체입니다. AWS 주도자 (사용자, 루트 사용자 또는 역할 세션) 가 요청할 때 이러한 정책을 평가합니다. 정책에서 권한은 요청이 허용되거나 거부되는 지를 결정합니다. 대부분의 정책은 JSON 문서로 AWS 저장됩니다. JSON 정책 문서의 구조와 콘텐츠에 대한 자세한 정보는 IAM 사용 설명서의 JSON 정책 개요를 참조하세요.

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

기본적으로, 사용자와 역할에는 어떠한 권한도 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다. 그런 다음 관리자가 IAM 정책을 역할에 추가하고, 사용자가 역할을 수임할 수 있습니다.

IAM 정책은 작업을 수행하기 위해 사용하는 방법과 상관없이 작업에 대한 권한을 정의합니다. 예를 들어, `iam:GetRole`태스크를 허용하는 정책이 있다고 가정합니다. 해당 정책을 사용하는 사용자는 AWS Management Console, AWS CLI, 또는 AWS API에서 역할 정보를 가져올 수 있습니다.

ID 기반 정책

ID 기반 정책은 IAM 사용자, 사용자 그룹 또는 역할과 같은 자격 증명에 연결할 수 있는 JSON 권한 정책 문서입니다. 이러한 정책은 사용자와 역할이 어떤 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는 지를 제어합니다. 자격 증명 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성을 참조하세요.

자격 증명 기반 정책은 인라인 정책 또는 관리형 정책으로 한층 더 분류할 수 있습니다. 인라인 정책은 단일 사용자, 그룹 또는 역할에 직접 포함됩니다. 관리형 정책은 내 여러 사용자, 그룹 및 역할에 연결할 수 있는 독립형 정책입니다. AWS 계정관리형 정책에는 AWS 관리형 정책과 고객 관리형 정책이 포함됩니다. 관리형 정책 또는 인라인 정책을 선택하는 방법을 알아보려면 IAM 사용 설명서의 관리형 정책과 인라인 정책의 선택을 참조하세요.

리소스 기반 정책

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 리소스 기반 정책의 예는 IAM 역할 신뢰 정책과 Amazon S3 버킷 정책입니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 정책이 연결된 리소스의 경우 정책은 지정된 보안 주체가 해당 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는 지를 정의합니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다. 보안 주체에는 계정, 사용자, 역할, 연동 사용자 등이 포함될 수 있습니다. AWS 서비스

리소스 기반 정책은 해당 서비스에 있는 인라인 정책입니다. IAM의 AWS 관리형 정책은 리소스 기반 정책에 사용할 수 없습니다.

액세스 제어 목록(ACLs)

액세스 제어 목록(ACLs)은 어떤 보안 주체(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는 지를 제어합니다. ACLs는 JSON 정책 문서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

ACL을 지원하는 서비스의 예로는 아마존 S3와 아마존 VPC가 있습니다. AWS WAF ACL에 대해 자세히 알아보려면 Amazon Simple Storage Service 개발자 안내서의 액세스 제어 목록(ACL) 개요를 참조하세요.

기타 정책 타입

AWS 일반적이지 않은 추가 정책 유형을 지원합니다. 이러한 정책 타입은 더 일반적인 정책 타입에 따라 사용자에게 부여되는 최대 권한을 설정할 수 있습니다.

- 권한 경계 – 권한 경계는 보안 인증 기반 정책에 따라 IAM 엔터티(IAM 사용자 또는 역할)에 부여할 수 있는 최대 권한을 설정하는 고급 기능입니다. 개체에 대한 권한 경계를 설정할 수 있습니다. 그 결과로 얻는 권한은 엔터티의 자격 증명 기반 정책과 그 권한 경계의 교집합입니다. `Principal` 필드에서 사용자나 역할을 보안 주체로 지정하는 리소스 기반 정책은 권한 경계를 통해 제한되지 않습니다. 이러한 정책 중 하나에 포함된 명시적 거부는 허용을 재정의합니다. 권한 경계에 대한 자세한 정보는 IAM 사용자 설명서의 IAM 엔터티에 대한 권한 경계를 참조합니다.
- 서비스 제어 정책 (SCP) - SCP는 조직 또는 조직 단위 (OU) 에 대한 최대 권한을 지정하는 JSON 정책입니다. AWS Organizations AWS Organizations 사업체가 소유한 여러 AWS 계정 개를 그룹화하고 중앙에서 관리하는 서비스입니다. 조직에서 모든 기능을 활성화할 경우 서비스 제어 정책(SCP)을 임의의 또는 모든 계정에 적용할 수 있습니다. SCP는 구성원 계정의 엔티티 (각 엔티티 포함) 에 대한 권한을 제한합니다. AWS 계정 루트 사용자조직 및 SCP에 대한 자세한 정보는AWS Organizations 사용 설명서의 SCP 작동 방식을 참조하세요.
- 세션 정책 – 세션 정책은 역할 또는 연합된 사용자에 대해 임시 세션을 프로그래밍 방식으로 생성할 때 파라미터로 전달하는 고급 정책입니다. 결과적으로 얻는 세션의 권한은 사용자 또는 역할 자격 증명 기반 정책의 교차 및 세션 정책입니다. 또한 권한을 리소스 기반 정책에서 가져올 수도 있습니다. 이러한 정책 중 하나에 포함된 명시적 거부는 허용을 재정의합니다. 자세한 정보는 IAM 사용자 설명서의 세션 정책을 참조합니다.

여러 정책 타입

여러 정책 타입이 요청에 적용되는 경우 결과 권한은 이해하기가 더 복잡합니다. 여러 정책 유형이 관련되어 있을 때 요청을 허용할지 여부를 AWS 결정하는 방법을 알아보려면 IAM 사용 설명서의 정책 평가 로직을 참조하십시오.

EventBridge 스케줄러와 IAM의 작동 방식

IAM을 사용하여 스케줄러에 대한 액세스를 관리하기 전에 EventBridge 스케줄러와 함께 사용할 수 있는 IAM 기능에 대해 알아보세요. EventBridge

Amazon EventBridge 스케줄러와 함께 사용할 수 있는 IAM 기능

IAM 특성	EventBridge 스케줄러 지원
ID 기반 정책	예
리소스 기반 정책	아니요
정책 작업	예
정책 리소스	예
정책 조건 키(서비스별)	예
ACLs	아니요
ABAC(정책 내 태그)	부분
임시 보안 인증	예
보안 주체 권한	예
서비스 역할	예
서비스 연결 역할	아니요

EventBridge 스케줄러 및 기타 AWS 서비스가 대부분의 IAM 기능과 어떻게 작동하는지 자세히 알아보려면 IAM 사용 설명서의 IAM과 함께 작동하는AWS 서비스를 참조하십시오.

스케줄러에 대한 자격 증명 기반 정책 EventBridge

ID 기반 정책 지원	예

자격 증명 기반 정책은 IAM 사용자, 사용자 그룹 또는 역할과 같은 자격 증명에 연결할 수 있는 JSON 권한 정책 문서입니다. 이러한 정책은 사용자와 역할이 어떤 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는 지를 제어합니다. 자격 증명 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성을 참조하세요.

IAM 자격 증명 기반 정책을 사용하면 허용되거나 거부되는 작업와 리소스뿐 아니라 작업이 허용되거나 거부되는 조건을 지정할 수 있습니다. 자격 증명 기반 정책에서는 보안 주체가 연결된 사용자 또는 역할에 적용되므로 보안 주체를 지정할 수 없습니다. JSON 정책에서 사용하는 모든 요소에 대해 알아보려면 IAM 사용 설명서의 IAM JSON 정책 요소 참조를 참조하세요.

스케줄러의 ID 기반 정책 예제 EventBridge

EventBridge 스케줄러 ID 기반 정책의 예를 보려면 을 참조하십시오. 자격 증명 기반 정책 사용

스케줄러 내의 리소스 기반 정책 EventBridge

리소스 기반 정책 지원	아니요

리소스 기반 정책은 리소스에 연결하는 JSON 정책 문서입니다. 리소스 기반 정책의 예는 IAM 역할 신뢰 정책과 Amazon S3 버킷 정책입니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 정책이 연결된 리소스의 경우 정책은 지정된 보안 주체가 해당 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는 지를 정의합니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다. 보안 주체에는 계정, 사용자, 역할, 연동 사용자 등이 포함될 수 있습니다. AWS 서비스

계정 간 액세스를 활성화하려는 경우 전체 계정이나 다른 계정의 IAM 엔터티를 리소스 기반 정책의 보안 주체로 지정할 수 있습니다. 리소스 기반 정책에 크로스 계정 보안 주체를 추가하는 것은 트러스트 관계 설정의 절반밖에 되지 않는다는 것을 유념하세요. 보안 주체와 리소스가 다른 AWS 계정경우 신뢰할 수 있는 계정의 IAM 관리자는 보안 주체 개체 (사용자 또는 역할) 에게 리소스에 액세스할 수 있는 권한도 부여해야 합니다. 개체에 자격 증명 기반 정책을 연결하여 권한을 부여합니다. 하지만 리소스 기반 정책이 동일 계정의 보안 주체에 액세스를 부여하는 경우 추가 자격 증명 기반 정책이 필요하지 않습니다. 자세한 정보는 IAM 사용 설명서의 IAM 역할과 리소스 기반 정책의 차이를 참조하세요.

스케줄러에 대한 EventBridge 정책 조치

정책 작업 지원	예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

JSON 정책의 `Action`요소는 정책에서 액세스를 허용하거나 거부하는 데 사용할 수 있는 태스크를 설명합니다. 정책 작업은 일반적으로 관련 AWS API 작업과 이름이 같습니다. 일치하는 API 작업이 없는 권한 전용 작업 같은 몇 가지 예외도 있습니다. 정책에서 여러 작업이 필요한 몇 가지 작업도 있습니다. 이러한 추가 작업을 일컬어 종속 작업이라고 합니다.

연결된 작업을 수행할 수 있는 권한을 부여하기 위한 정책에 작업을 포함하십시오.

EventBridge 스케줄러 작업 목록을 보려면 서비스 권한 부여 참조의 Amazon EventBridge Scheduler에서 정의한 작업을 참조하십시오.

EventBridge 스케줄러의 정책 작업은 작업 앞에 다음 접두사를 사용합니다.

```
scheduler
```

단일 문에서 여러 작업을 지정하려면 다음과 같이 쉼표로 구분합니다.

```
"Action": [
      "scheduler:action1",
      "scheduler:action2"
]
```

와일드카드(*)를 사용하여 여러 작업을 지정할 수 있습니다. 예를 들어, `List`라는 단어로 시작하는 모든 태스크를 지정하려면 다음 태스크를 포함합니다.

```
"Action": [
   "scheduler:List*"
]
```

스케줄러용 정책 리소스 EventBridge

정책 리소스 지원	예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지 지정할 수 있습니다.

`Resource` JSON 정책 요소는 작업이 적용되는 하나 이상의 개체를 지정합니다. 문장에는 `Resource`또는 `NotResource`요소가 반드시 추가되어야 합니다. 모범 사례에 따라 Amazon 리소스 이름(ARN)을 사용하여 리소스를 지정합니다. 리소스 수준 권한이라고 하는 특정 리소스 타입을 지원하는 작업에 대해 이 작업을 수행할 수 있습니다.

작업 나열과 같이 리소스 수준 권한을 지원하지 않는 작업의 경우, 와일드카드(*)를 사용하여 해당 문이 모든 리소스에 적용됨을 나타냅니다.

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

EventBridge 스케줄러 리소스 유형 및 ARN 목록을 보려면 서비스 인증 참조의 Amazon EventBridge Scheduler에서 정의한 리소스를 참조하십시오. 각 리소스의 ARN을 지정할 수 있는 작업에 대해 알아보려면 Amazon EventBridge Scheduler에서 정의한 작업을 참조하십시오.

EventBridge 스케줄러 ID 기반 정책의 예를 보려면 을 참조하십시오. 자격 증명 기반 정책 사용

스케줄러의 정책 조건 키 EventBridge

서비스별 정책 조건 키 지원	예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지 지정할 수 있습니다.

`Condition` 요소(또는 `Condition` 블록)를 사용하면 정책이 발효되는 조건을 지정할 수 있습니다. `Condition` 요소는 옵션입니다. 같거나 작음과 같은 조건 연산자를 사용하여 정책의 조건을 요청의 값과 일치시키는 조건식을 생성할 수 있습니다.

한 문에서 여러 `Condition`요소를 지정하거나 단일 `Condition`요소에서 여러 키를 지정하는 경우 AWS 는 논리적 `AND`태스크를 사용하여 평가합니다. 단일 조건 키에 여러 값을 지정하는 경우 는 논리적 `OR` 연산을 사용하여 조건을 AWS 평가합니다. 명문의 권한을 부여하기 전에 모든 조건을 충족해야 합니다.

조건을 지정할 때 자리 표시자 변수를 사용할 수도 있습니다. 예를 들어, IAM 사용자에게 IAM 사용자 이름으로 태그가 지정된 경우에만 리소스에 액세스할 수 있는 권한을 부여할 수 있습니다. 자세한 정보는 IAM 사용 설명서의 IAM 정책 요소: 변수 및 태그를 참조하세요.

AWS 글로벌 조건 키 및 서비스별 조건 키를 지원합니다. 모든 AWS 글로벌 조건 키를 보려면 IAM 사용 AWS 설명서의 글로벌 조건 컨텍스트 키를 참조하십시오.

EventBridge 스케줄러 조건 키 목록을 보려면 서비스 권한 부여 참조의 Amazon EventBridge Scheduler의 조건 키를 참조하십시오. 조건 키를 사용할 수 있는 작업 및 리소스를 알아보려면 Amazon EventBridge Scheduler에서 정의한 작업을 참조하십시오.

EventBridge 스케줄러 ID 기반 정책의 예를 보려면 을 참조하십시오. 자격 증명 기반 정책 사용

스케줄러의 ACL EventBridge

ACL 지원	아니요

액세스 제어 목록(ACLs)은 어떤 보안 주체(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는 지를 제어합니다. ACLs는 JSON 정책 문서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

ABAC (스케줄러 포함) EventBridge

ABAC(정책 내 태그) 지원	부분

속성 기반 액세스 제어(ABAC)는 속성을 기반으로 권한을 정의하는 권한 부여 전략입니다. AWS에서는 이러한 속성을 태그라고 합니다. IAM 개체 (사용자 또는 역할) 및 여러 AWS 리소스에 태그를 첨부할 수 있습니다. ABAC의 첫 번째 단계로 개체 및 리소스에 태그를 지정합니다. 그런 다음 보안 주체의 태그가 액세스하려는 리소스의 태그와 일치할 때 작업을 허용하도록 ABAC 정책을 설계합니다.

ABAC는 빠르게 성장하는 환경에서 유용하며 정책 관리가 번거로운 상황에 도움이 됩니다.

태그를 기반으로 액세스를 제어하려면 `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` 또는 `aws:TagKeys` 조건 키를 사용하여 정책의 조건 요소에 태그 정보를 제공합니다.

서비스가 모든 리소스 타입에 대해 세 가지 조건 키를 모두 지원하는 경우, 값은 서비스에 대해 예입니다. 서비스가 일부 리소스 타입에 대해서만 세 가지 조건 키를 모두 지원하는 경우, 값은 부분적입니다.

ABAC에 대한 자세한 정보는 IAM 사용 설명서의 ABAC란 무엇인가요?를 참조하세요. ABAC 설정 단계가 포함된 자습서를 보려면 IAM 사용 설명서의 속성 기반 액세스 제어(ABAC) 사용을 참조하세요.

스케줄러에서 임시 자격 증명 사용 EventBridge

임시 보안 인증 지원	예

임시 자격 증명을 사용하여 로그인하면 작동하지 AWS 서비스 않는 것도 있습니다. 임시 자격 증명을 사용하는 방법을 AWS 서비스 비롯한 추가 정보는 IAM 사용 설명서의 IAM과AWS 서비스 연동되는 내용을 참조하십시오.

사용자 이름과 암호를 제외한 다른 방법을 AWS Management Console 사용하여 로그인하면 임시 자격 증명을 사용하는 것입니다. 예를 들어 회사의 SSO (Single Sign-On) 링크를 AWS 사용하여 액세스하는 경우 이 프로세스에서 자동으로 임시 자격 증명을 생성합니다. 또한 콘솔에 사용자로 로그인한 다음 역할을 전환할 때 임시 보안 인증을 자동으로 생성합니다. 역할 전환에 대한 자세한 정보는 IAM 사용 설명서의 역할로 전환(콘솔)을 참조하세요.

또는 API를 사용하여 임시 자격 증명을 수동으로 생성할 수 있습니다 AWS CLI . AWS 그런 다음 해당 임시 자격 증명을 사용하여 액세스할 수 AWS있습니다. AWS 장기 액세스 키를 사용하는 대신 임시 자격 증명을 동적으로 생성할 것을 권장합니다. 자세한 정보는 IAM의 임시 보안 인증 섹션을 참조하세요.

스케줄러에 대한 서비스 간 보안 주체 권한 EventBridge

전달 액세스 세션(FAS) 지원	예

IAM 사용자 또는 역할을 사용하여 작업을 수행하는 AWS경우 사용자는 보안 주체로 간주됩니다. 일부 서비스를 사용하는 경우 다른 서비스에서 다른 작업을 시작하는 작업을 수행할 수 있습니다. FAS는 전화를 거는 주체의 권한을 다운스트림 서비스에 AWS 서비스 요청하라는 요청과 결합하여 사용합니다. AWS 서비스 FAS 요청은 다른 서비스 AWS 서비스 또는 리소스와의 상호 작용이 필요한 요청을 서비스가 수신한 경우에만 이루어집니다. 이 경우 두 작업을 모두 수행할 수 있는 권한이 있어야 합니다. FAS 요청 시 정책 세부 정보는 전달 액세스 세션을 참조하세요.

스케줄러의 서비스 역할 EventBridge

서비스 역할 지원	예

서비스 역할은 서비스가 사용자를 대신하여 작업을 수행하는 것으로 가정하는 IAM 역할입니다. IAM 관리자는 IAM 내에서 서비스 역할을 생성, 수정 및 삭제할 수 있습니다. 자세한 정보는 IAM 사용자 설명서의 AWS 서비스에 대한 권한을 위임할 역할 생성을 참조합니다.

> **Warning**
>
> 서비스 역할의 권한을 변경하면 EventBridge 스케줄러 기능이 작동하지 않을 수 있습니다. EventBridge 스케줄러가 이에 대한 지침을 제공하는 경우에만 서비스 역할을 편집하십시오.

스케줄러의 서비스 연결 역할 EventBridge

서비스 연결 역할 지원	아니요

서비스 연결 역할은 에 연결된 서비스 역할 유형입니다. AWS 서비스서비스는 사용자를 대신하여 작업을 수행하기 위해 역할을 수임할 수 있습니다. 서비스 연결 역할은 사용자에게 AWS 계정 표시되며 해당 서비스가 소유합니다. IAM 관리자는 서비스 링크 역할의 권한을 볼 수 있지만 편집은 할 수 없습니다.

서비스 연결 역할 생성 또는 관리에 대한 자세한 내용은 IAM으로 작업하는AWS 서비스 단원을 참조하세요. 서비스 연결 역할 열에서 Yes가 포함된 서비스를 테이블에서 찾습니다. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 Yes(네) 링크를 선택합니다.

자격 증명 기반 정책 사용

기본적으로 사용자와 역할에는 EventBridge Scheduler 리소스를 만들거나 수정할 권한이 없습니다. 또한 AWS Management Console, AWS Command Line Interface (AWS CLI) 또는 AWS API를 사용하여 작업을 수행할 수도 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다. 그런 다음 관리자가 IAM 정책을 역할에 추가하고, 사용자가 역할을 맡을 수 있습니다.

이러한 예제 JSON 정책 문서를 사용하여 IAM ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성을 참조하세요.

각 리소스 유형의 ARN 형식을 비롯하여 EventBridge 스케줄러가 정의하는 작업 및 리소스 유형에 대한 자세한 내용은 서비스 인증 참조의 Amazon EventBridge Scheduler용 작업, 리소스 및 조건 키를 참조하십시오.

주제

정책 모범 사례

ID 기반 정책은 누군가가 사용자 계정에서 EventBridge Scheduler 리소스를 생성, 액세스 또는 삭제할 수 있는지 여부를 결정합니다. 이 작업으로 인해 AWS 계정에 비용이 발생할 수 있습니다. ID 기반 정책을 생성하거나 편집할 때는 다음 지침과 권장 사항을 따르세요.

- AWS 관리형 정책으로 시작하여 최소 권한 권한으로 이동 — 사용자와 워크로드에 권한을 부여하려면 여러 일반적인 사용 사례에 권한을 부여하는 AWS 관리형 정책을 사용하세요. 해당 내용은 에서 사용할 수 있습니다. AWS 계정사용 사례에 맞는 AWS 고객 관리형 정책을 정의하여 권한을 더 줄이는 것이 좋습니다. 자세한 정보는 IAM 사용 설명서의 AWS managed policies(관리형 정책) 또는 AWS managed policies for job functions(직무에 대한 관리형 정책)를 참조하세요.
- 최소 권한 적용 – IAM 정책을 사용하여 권한을 설정하는 경우 태스크를 수행하는 데 필요한 권한만 부여합니다. 이렇게 하려면 최소 권한으로 알려진 특정 조건에서 특정 리소스에 대해 수행할 수 있는 작업을 정의합니다. IAM을 사용하여 권한을 적용하는 방법에 대한 자세한 정보는 IAM 사용 설명서에 있는 Policies and permissions in IAM(IAM의 정책 및 권한)을 참조하세요.
- IAM 정책의 조건을 사용하여 액세스 추가 제한 – 정책에 조건을 추가하여 작업 및 리소스에 대한 액세스를 제한할 수 있습니다. 예를 들어 SSL을 사용하여 모든 요청을 전송해야 한다고 지정하는 정책 조건을 작성할 수 있습니다. 예를 AWS 서비스들어 특정 작업을 통해 서비스 작업을 사용하는 경우 조건을 사용하여 서비스 작업에 대한 액세스 권한을 부여할 수도 AWS CloudFormation있습니다. 자세한 정보는 IAM 사용 설명서의 IAM JSON 정책 요소: 조건을 참조하세요.
- IAM Access Analyzer를 통해 IAM 정책을 검증하여 안전하고 기능적인 권한 보장 – IAM Access Analyzer에서는 IAM 정책 언어(JSON)와 모범 사례가 정책에서 준수되도록 신규 및 기존 정책을 검증합니다. IAM Access Analyzer는 100개 이상의 정책 확인 항목과 실행 가능한 추천을 제공하여 안전하고 기능적인 정책을 작성하도록 돕습니다. 자세한 정보는 IAM 사용 설명서의 IAM Access Analyzer 정책 검증을 참조하tpdy.
- 멀티 팩터 인증 (MFA) 필요 - IAM 사용자 또는 루트 사용자가 필요한 시나리오가 있는 경우 추가 보안을 위해 AWS 계정 MFA를 활성화하십시오. API 작업을 직접 호출할 때 MFA가 필요하면 정

책에 MFA 조건을 추가합니다. 자세한 정보는 IAM 사용 설명서의 Configuring MFA-protected API access(MFA 보호 API 액세스 구성)를 참조하세요.

IAM의 모범 사례에 대한 자세한 내용은 IAM 사용 설명서의 IAM의 보안 모범 사례를 참조하세요.

EventBridge 스케줄러 권한

IAM 주체 (사용자, 그룹 또는 역할) 가 EventBridge Scheduler에서 일정을 생성하고 콘솔 또는 API를 통해 EventBridge 스케줄러 리소스에 액세스하려면 보안 주체의 권한 정책에 일련의 권한이 추가되어야 합니다. 보안 주체의 직무에 따라 이러한 권한을 구성할 수 있습니다. 예를 들어 기존 일정 목록을 보기 위해 EventBridge 스케줄러 콘솔만 사용하는 사용자나 역할은 API 작업을 호출하는 데 필요한 권한이 없어도 됩니다. `CreateSchedule` 권한이 가장 적은 액세스만 제공하도록 ID 기반 권한을 조정하는 것이 좋습니다.

다음 목록은 EventBridge Scheduler의 리소스와 해당하는 지원되는 작업을 보여줍니다.

- Schedule
  - `scheduler:ListSchedules`
  - `scheduler:GetSchedule`
  - `scheduler:CreateSchedule`
  - `scheduler:UpdateSchedule`
  - `scheduler:DeleteSchedule`
- 일정 그룹
  - `scheduler:ListScheduleGroups`
  - `scheduler:GetScheduleGroup`
  - `scheduler:CreateScheduleGroup`
  - `scheduler:DeleteScheduleGroup`
  - `scheduler:ListTagsForResource`
  - `scheduler:TagResource`
  - `scheduler:UntagResource`

EventBridge 스케줄러 권한을 사용하여 자체 고객 관리형 정책을 만들어 스케줄러와 함께 EventBridge 사용할 수 있습니다. 또한 다음 섹션에 설명된 AWS 관리형 정책을 사용하여 자체 정책을 관리할 필요 없이 일반적인 사용 사례에 필요한 권한을 부여할 수 있습니다.

AWS EventBridge 스케줄러의 관리형 정책

AWS AWS 생성 및 관리하는 독립형 IAM 정책을 제공하여 많은 일반적인 사용 사례를 해결합니다. 관리형 또는 미리 정의된 정책은 일반 사용 사례에 필요한 권한을 부여하므로 어떤 권한이 필요한지 조사할 필요가 없습니다. 자세한 내용은 IAM 사용자 설명서의 AWS 관리형 정책을 참조하세요. 계정의 사용자에게 연결할 수 있는 다음과 같은 AWS 관리형 정책은 스케줄러에만 해당됩니다. EventBridge

- the section called “AmazonEventBridgeSchedulerFullAccess”— 콘솔 및 API를 사용하여 EventBridge 스케줄러에 대한 전체 액세스 권한을 부여합니다.
- the section called “AmazonEventBridgeSchedulerReadOnlyAccess”— 스케줄러에 대한 읽기 전용 액세스 권한을 EventBridge 부여합니다.

AmazonEventBridgeSchedulerFullAccess

AmazonEventBridgeSchedulerFullAccess관리형 정책은 스케줄 및 스케줄 그룹에 대한 모든 EventBridge 스케줄러 작업을 사용할 수 있는 권한을 부여합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "scheduler:*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::*:role/*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "scheduler.amazonaws.com"
                }
            }
        }
    ]
}
```

`AmazonEventBridgeSchedulerReadOnlyAccess`

AmazonEventBridgeSchedulerReadOnlyAccess 관리형 정책은 일정 및 일정 그룹에 대한 세부 정보를 볼 수 있는 읽기 전용 권한을 부여합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "scheduler:ListSchedules",
                "scheduler:ListScheduleGroups",
                "scheduler:GetSchedule",
                "scheduler:GetScheduleGroup",
                "scheduler:ListTagsForResource"
            ],
            "Resource": "*"
        }
    ]
}
```

스케줄러에 대한 EventBridge 고객 관리형 정책

다음 예를 사용하여 EventBridge Scheduler에 대한 고유한 고객 관리형 정책을 만들 수 있습니다. 고객 관리형 정책을 사용하면 보안 주체의 직무에 따라 팀의 애플리케이션 및 사용자에게 필요한 작업 및 리소스에 대한 권한만 부여할 수 있습니다.

주제

- 예제: CreateSchedule
- 예제: GetSchedule
- 예제: UpdateSchedule
- 예제: DeleteScheduleGroup

예제: `CreateSchedule`

새 일정을 생성할 때는 EventBridge Scheduler의 데이터를 암호화할지 아니면 고객 관리 키를 사용하여 암호화할지 선택합니다. AWS 소유 키

다음 정책은 보안 주체가 일정을 생성하고 AWS 소유 키를 사용하여 암호화를 적용할 수 있도록 허용합니다. 를 AWS 소유 키사용하면 AWS Key Management Service (AWS KMS) 에서 리소스를 AWS 관리하므로 상호 작용하는 데 추가 권한이 필요하지 않습니다. AWS KMS

```
{
    "Version": "2012-10-17",
    "Statement":
    [
        {
            "Action":
            [
                "scheduler:CreateSchedule"
            ],
            "Effect": "Allow",
            "Resource":
            [
                "arn:aws:scheduler:us-west-2:123456789012:schedule/my-group/my-
schedule-name"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::123456789012:role/*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "scheduler.amazonaws.com"
                }
            }
        }
    ]
}
```

다음 정책을 사용하여 보안 주체가 일정을 만들고 암호화에 AWS KMS 고객 관리 키를 사용하도록 허용하세요. 고객 관리 키를 사용하려면 보안 주체가 계정의 AWS KMS 리소스에 액세스할 수 있는 권한이 있어야 합니다. 이 정책은 Scheduler의 데이터를 암호화하는 데 사용할 지정된 단일 KMS 키에 대한 EventBridge 액세스 권한을 부여합니다. 또는 와일드카드(*) 문자를 사용하여 계정의 모든 키 또는 지정된 이름 패턴과 일치하는 하위 집합에 대한 액세스 권한을 부여할 수 있습니다.

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

```
    [
        {
            "Action":
            [
                "scheduler:CreateSchedule"
            ],
            "Effect": "Allow",
            "Resource":
            [
                "arn:aws:scheduler:us-west-2:123456789012:schedule/my-group/my-
schedule-name"
            ]
        },
        {
            "Action":
            [
                "kms:DescribeKey",
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Effect": "Allow",
            "Resource":
            [
                "arn:aws:kms:us-west-2:123456789012:key/my-key-id"
            ],
            "Conditions": {
                "StringLike": {
                    "kms:ViaService": "scheduler.amazonaws.com",
                    "kms:EncryptionContext:aws:scheduler:schedule:arn":
 "arn:aws:scheduler:us-west-2:123456789012:schedule/my-group/my-schedule-name"
            }
        }
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::123456789012:role/*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "scheduler.amazonaws.com"
                }
            }
        }
    ]
```

```
}
```

예제: **GetSchedule**

다음 정책을 사용하여 보안 주체가 일정에 대한 정보를 가져오도록 허용하십시오.

```
{
    "Version": "2012-10-17",
    "Statement":
    [
        {
            "Action":
            [
                "scheduler:GetSchedule"
            ],
            "Effect": "Allow",
            "Resource":
            [
                "arn:aws:scheduler:us-west-2:123456789012:schedule/my-group/my-
schedule-name"
            ]
        }
    ]
}
```

예제: **UpdateSchedule**

다음 정책을 사용하여 보안 주체가 `scheduler:UpdateSchedule` 작업을 직접적으로 호출하여 일정을 업데이트하도록 허용하십시오. 과 마찬가지로 정책은 스케줄에서 암호화에 사용하는 키 AWS KMS AWS 소유 키 또는 고객 관리 키를 사용하는지 여부에 따라 달라집니다. `CreateSchedule` 로 구성된 일정의 AWS 소유 키경우 다음 정책을 사용하십시오.

```
{
    "Version": "2012-10-17",
    "Statement":
    [
        {
            "Action":
            [
                "scheduler:UpdateSchedule"
            ],
            "Effect": "Allow",
```

```
            "Resource":
            [
                "arn:aws:scheduler:us-west-2:123456789012:schedule/my-group/my-
schedule-name"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::123456789012:role/*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "scheduler.amazonaws.com"
                }
            }
        }
    ]
}
```

고객 관리형 키로 구성된 일정의 경우 다음 정책을 사용하십시오. 이 정책에는 보안 주체가 계정의 AWS KMS 리소스에 액세스할 수 있는 추가 권한이 포함되어 있습니다.

```
{
    "Version": "2012-10-17",
    "Statement":
    [
        {
            "Action":
            [
                "scheduler:UpdateSchedule"
            ],
            "Effect": "Allow",
            "Resource":
            [
                "arn:aws:scheduler:us-west-2:123456789012:schedule/my-group/my-
schedule-name
        },
        {
            "Action":
            [
                "kms:DescribeKey",
                "kms:GenerateDataKey",
                "kms:Decrypt"
```

```
            ],
            "Effect": "Allow",
            "Resource":
            [
                "arn:aws:kms:us-west-2:123456789012:key/my-key-id"
            ],
            "Conditions": {
                "StringLike": {
                    "kms:ViaService": "scheduler.amazonaws.com",
                    "kms:EncryptionContext:aws:scheduler:schedule:arn":
 "arn:aws:scheduler:us-west-2:123456789012:schedule/my-group/my-schedule-name"
            }
        }
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::123456789012:role/*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "scheduler.amazonaws.com"
                }
            }
        }
    ]
}
```

예제: **`DeleteScheduleGroup`**

다음 정책을 사용하여 보안 주체가 일정 그룹을 삭제하도록 허용하십시오. 그룹을 삭제하면 해당 그룹과 연결된 일정도 삭제됩니다. 그룹을 삭제하는 보안 주체는 해당 그룹과 연결된 일정도 삭제할 수 있는 권한이 있어야 합니다. 이 정책은 보안 주체에게 지정된 일정 그룹의 `scheduler:DeleteScheduleGroup` 작업과 그룹 내 모든 일정을 직접적으로 호출할 수 있는 권한을 부여합니다.

> **Note**
>
> EventBridge Scheduler는 개별 일정에 대한 리소스 수준 권한 지정을 지원하지 않습니다. 예를 들어, 다음 설명은 유효하지 않으므로 정책에 포함해서는 안 됩니다. `"Resource": "arn:aws:scheduler:us-west-2:123456789012:schedule/my-group/my-schedule-name"`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "scheduler:DeleteSchedule",
            "Resource": "arn:aws:scheduler:us-west-2:123456789012:schedule/my-group/*"
        },
        {
            "Effect": "Allow",
            "Action": "scheduler:DeleteScheduleGroup",
            "Resource": "arn:aws:scheduler:us-west-2:123456789012:schedule/my-group"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::123456789012:role/*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "scheduler.amazonaws.com"
                }
            }
        }
    ]
}
```

AWS 관리형 정책 업데이트

변경 사항	설명	날짜
the section called “AmazonEventBridgeSchedulerFullAccess” – 새 관리형 정책	EventBridge Scheduler는 사용자에게 일정 및 일정 그룹을 포함한 모든 리소스에 대한 전체 액세스 권한을 부여하는 새로운 관리형 정책에 대한 지원을 추가합니다.	2022년 11월 10일
the section called “AmazonEventBridgeS	EventBridge Scheduler는 사용자에게 일정 및 일정 그룹을 포함한 모든 리소스에 대한 읽기	2022년 11월 10일

변경 사항	설명	날짜
chedulerReadOnlyAccess ” – 새 관리형 정책	전용 액세스 권한을 부여하는 새로운 관리형 정책에 대한 지원을 추가합니다.	
EventBridge 스케줄러가 변경 내용 추적을 시작했습니다.	EventBridge Scheduler는 AWS 관리형 정책의 변경 사항을 추적하기 시작했습니다.	2022년 11월 10일

혼동된 대리자 방지

혼동된 대리자 문제는 작업을 수행할 권한이 없는 엔터티가 권한이 더 많은 엔터티에게 작업을 수행하도록 강요할 수 있는 보안 문제입니다. 에서 AWS크로스 서비스 사칭으로 인해 대리인 문제가 발생할 수 있습니다. 교차 서비스 가장은 한 서비스(직접 호출하는 서비스)가 다른 서비스(직접 호출되는 서비스)를 직접 호출할 때 발생할 수 있습니다. 직접 호출하는 서비스는 다른 고객의 리소스에 대해 액세스 권한이 없는 방식으로 작동하게 권한을 사용하도록 조작될 수 있습니다. 이를 방지하기 위해 AWS 에서는 계정의 리소스에 대한 액세스 권한이 부여된 서비스 보안 주체를 사용하여 모든 서비스에 대한 데이터를 보호하는 데 도움이 되는 도구를 제공합니다.

스케줄 실행 역할의 aws:SourceArn및 aws:SourceAccount글로벌 조건 컨텍스트 키를 사용하여 EventBridge Scheduler가 다른 서비스에 리소스에 액세스할 수 있도록 허용하는 권한을 제한하는 것이 좋습니다. 하나의 리소스만 교차 서비스 액세스와 연결되도록 허용하려는 경우 aws:SourceArn을 사용하세요. 해당 계정의 모든 리소스가 교차 서비스 사용과 연결되도록 허용하려는 경우 aws:SourceAccount을 사용하세요.

혼동된 대리자 문제로부터 보호하는 가장 효과적인 방법은 리소스의 전체 ARN이 포함된 aws:SourceArn 전역 조건 컨텍스트 키를 사용하는 것입니다. arn:aws:scheduler:*:*123456789012*:schedule-group/*your-schedule-group* 조건은 개별 일정 그룹으로 범위가 지정됩니다.

리소스의 전체 ARN을 모르거나 여러 리소스를 지정하는 경우, ARN의 알 수 없는 부분에 대해 와일드카드 문자(*)를 포함한 aws:SourceArn 글로벌 조건 컨텍스트 키를 사용합니다. 예를 들면 arn:aws:scheduler:*:*123456789012*:schedule-group/*입니다.

의 값은 이 조건의 범위를 지정할 EventBridge 스케줄러 스케줄 그룹 aws:SourceArn ARN이어야 합니다.

> **Important**
>
> `aws:SourceArn` 명령문의 범위를 특정 일정이나 일정 이름 접두사로 지정하지 마십시오. 지정하는 ARN은 일정 그룹이어야 합니다.

다음 예는 사용자 실행 역할 신뢰 정책에서 `aws:SourceArn` 및 `aws:SourceAccount` 전역 조건 컨텍스트 키를 사용하여 혼동된 대리자 문제를 방지하는 방법을 보여줍니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "scheduler.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012",
                    "aws:SourceArn": "arn:aws:scheduler:us-
west-2:123456789012:schedule-group/your-schedule-group"
                }
            }
        }
    ]
}
```

Amazon EventBridge 스케줄러 자격 증명 및 액세스 문제 해결

다음 정보를 사용하면 EventBridge Scheduler 및 IAM으로 작업할 때 발생할 수 있는 일반적인 문제를 진단하고 해결하는 데 도움이 됩니다.

주제

- 저는 스케줄러에서 작업을 수행할 권한이 없습니다. EventBridge
- 저는 IAM을 수행할 권한이 없습니다. PassRole
- 외부 사용자가 내 EventBridge 스케줄러 리소스에 액세스할 AWS 계정 수 있도록 허용하고 싶습니다.

저는 스케줄러에서 작업을 수행할 권한이 없습니다. EventBridge

작업을 수행할 수 있는 권한이 없다는 오류가 수신되면 작업을 수행할 수 있도록 정책을 업데이트해야 합니다.

다음 예제 오류는 mateojackson IAM 사용자가 콘솔을 사용하여 가상 *my-example-widget* 리소스에 대한 세부 정보를 보려고 하지만 가상 scheduler:*GetWidget* 권한이 없을 때 발생합니다.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 scheduler:GetWidget on resource: my-example-widget
```

이 경우 Mateo의 정책은 scheduler:*GetWidget* 작업을 사용하여 *my-example-widget* 리소스에 액세스하도록 허용하도록 업데이트해야 합니다.

도움이 필요하면 AWS 관리자에게 문의하세요. 관리자는 로그인 자격 증명을 제공한 사람입니다.

저는 IAM을 수행할 권한이 없습니다. PassRole

작업을 수행할 권한이 없다는 오류가 발생하는 경우 EventBridge Scheduler에 역할을 전달할 수 있도록 정책을 업데이트해야 합니다. iam:PassRole

일부 AWS 서비스 서비스에서는 새 서비스 역할 또는 서비스 연결 역할을 만드는 대신 기존 역할을 해당 서비스에 전달할 수 있습니다. 이렇게 하려면 사용자가 서비스에 역할을 전달할 수 있는 권한을 가지고 있어야 합니다.

다음 예제 오류는 이라는 IAM 사용자가 콘솔을 사용하여 Scheduler에서 작업을 marymajor 수행하려고 할 때 발생합니다. EventBridge 하지만 작업을 수행하려면 서비스 역할이 부여한 권한이 서비스에 있어야 합니다. Mary는 서비스에 역할을 전달할 수 있는 권한을 가지고 있지 않습니다.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

이 경우 Mary가 iam:PassRole작업을 수행할 수 있도록 Mary의 정책을 업데이트해야 합니다.

도움이 필요하면 관리자에게 문의하세요. AWS 관리자는 로그인 자격 증명을 제공한 사람입니다.

외부 사용자가 내 EventBridge 스케줄러 리소스에 액세스할 AWS 계정 수 있도록 허용하고 싶습니다.

다른 계정의 사용자 또는 조직 외부의 사람이 리소스에 액세스할 때 사용할 수 있는 역할을 생성할 수 있습니다. 역할을 수임할 신뢰할 수 있는 사람을 지정할 수 있습니다. 리소스 기반 정책 또는 액세스 제

어 목록(ACL)을 지원하는 서비스의 경우 이러한 정책을 사용하여 다른 사람에게 리소스에 대한 액세스 권한을 부여할 수 있습니다.

자세히 알아보려면 다음을 참조하세요.

- EventBridge 스케줄러가 이러한 기능을 지원하는지 여부를 알아보려면 을 참조하십시오. EventBridge 스케줄러와 IAM의 작동 방식
- 소유한 리소스의 액세스 권한을 AWS 계정 부여하는 방법을 알아보려면 IAM 사용 설명서의 다른 AWS 계정 IAM 사용자에게 액세스 권한 제공을 참조하십시오.
- 제3자에게 리소스에 대한 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 타사 AWS 계정AWS 계정 소유에 대한 액세스 제공을 참조하십시오.
- ID 페더레이션을 통해 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 외부에서 인증된 사용자에게 액세스 권한 제공(자격 증명 연동)을 참조하세요.
- 크로스 계정 액세스를 위한 역할과 리소스 기반 정책 사용의 차이점을 알아보려면 IAM 사용 설명서의 IAM 역할과 리소스 기반 정책의 차이를 참조하세요.

Amazon EventBridge 스케줄러에서의 데이터 보호

AWS 공동 책임 모델 Amazon EventBridge Scheduler의 데이터 보호에 적용됩니다. 이 모델에 설명된 대로, AWS 는 모든 모델을 실행하는 글로벌 인프라를 보호할 책임이 있습니다. AWS 클라우드사용자는 인프라에서 호스팅되는 콘텐츠를 관리해야 합니다. 사용하는 AWS 서비스 의 보안 구성과 관리 작업에 대한 책임도 사용자에게 있습니다. 데이터 프라이버시에 대한 자세한 내용은 데이터 프라이버시 FAQ를 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은AWS 보안 블로그에서 AWS 공동 책임 모델 및 GDPR 블로그 게시물을 참조하세요.

데이터 보호를 위해 AWS 계정 자격 증명을 보호하고 AWS IAM Identity Center OR AWS Identity and Access Management (IAM) 을 사용하여 개별 사용자를 설정하는 것이 좋습니다. 이렇게 하면 개별 사용자에게 자신의 직무를 충실히 이행하는 데 필요한 권한만 부여됩니다. 또한 다음과 같은 방법으로 데이터를 보호하는 것이 좋습니다.

- 각 계정에 멀티 팩터 인증 설정(MFA)을 사용하세요.
- SSL/TLS를 사용하여 리소스와 통신할 수 있습니다. AWS TLS 1.2는 필수이며 TLS 1.3를 권장합니다.
- 를 사용하여 API 및 사용자 활동 로깅을 설정합니다. AWS CloudTrail
- 포함된 모든 기본 보안 제어와 함께 AWS 암호화 솔루션을 사용하십시오 AWS 서비스.

- Amazon S3에 저장된 민감한 데이터를 검색하고 보호하는 데 도움이 되는 Amazon Macie와 같은 고급 관리형 보안 서비스를 사용하세요.
- 명령줄 인터페이스 또는 API를 AWS 통해 액세스할 때 FIPS 140-2로 검증된 암호화 모듈이 필요한 경우 FIPS 엔드포인트를 사용하십시오. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 FIPS(Federal Information Processing Standard) 140-2를 참조하세요.

고객의 이메일 주소와 같은 기밀 정보나 중요한 정보는 태그나 이름 필드와 같은 자유 양식 필드에 입력하지 않는 것이 좋습니다. 여기에는 콘솔, API 또는 SDK를 AWS 서비스 사용하여 EventBridge 스케줄러 또는 기타 작업을 수행하는 경우가 포함됩니다. AWS CLI AWS 이름에 사용되는 태그 또는 자유 형식 텍스트 필드에 입력하는 모든 데이터는 청구 또는 진단 로그에 사용될 수 있습니다. 외부 서버에 URL을 제공할 때 해당 서버에 대한 요청을 검증하기 위해 보안 인증 정보를 URL에 포함시켜서는 안 됩니다.

주제

저장된 데이터 암호화

이 섹션에서는 Amazon EventBridge Scheduler가 저장된 데이터를 암호화하고 복호화하는 방법을 설명합니다. 저장 데이터는 EventBridge 스케줄러와 서비스의 기본 구성 요소에 저장된 데이터입니다. EventBridge Scheduler는 AWS Key Management Service (AWS KMS) 와 통합되어 를 사용하여 데이터를 암호화하고 해독합니다. AWS KMS key EventBridge 스케줄러는 두 가지 유형의 KMS 키, 즉 고객 관리 키를 지원합니다. AWS 소유 키

> **Note**
>
> EventBridge 스케줄러는 대칭 암호화 KMS 키 사용만 지원합니다.

AWS 소유 키 AWS 서비스가 여러 계정에서 사용하기 위해 소유하고 관리하는 KMS 키입니다. AWS AWS 소유 키 EventBridge Scheduler에서 사용하는 데이터는 AWS 계정에 저장되지 않지만 EventBridge Scheduler는 이를 사용하여 데이터와 리소스를 보호합니다. 기본적으로 EventBridge Scheduler는 소유 키를 사용하여 모든 데이터를 암호화하고 복호화합니다. AWS 사용자 AWS 소유 키 또는 액세스 정책을 관리할 필요가 없습니다. EventBridge Scheduler가 데이터 보호를 AWS 소유

키 위해 사용하는 경우 수수료가 발생하지 않으며, 사용량은 계정 할당량에 포함되지 않습니다. AWS KMS

고객 관리 키는 AWS 계정에 저장되며 사용자가 만들고 소유하고 관리하는 KMS 키입니다. 특정 사용 사례에서 EventBridge Scheduler에서 데이터를 보호하는 암호화 키를 제어하고 감사해야 하는 경우 고객 관리 키를 사용할 수 있습니다. 고객 관리형 키를 선택하면 키 정책을 관리해야 합니다. 고객 관리형 키는 매달 요금이 발생하며, 프리 티어를 초과해서 사용한 만큼 요금이 부과됩니다. 고객 관리형 키를 사용하는 것도 AWS KMS 할당량의 일부로 계산됩니다. 요금에 대한 자세한 내용은 AWS Key Management Service 요금을 참조하세요.

주제

- 암호화 아티팩트
- KMS 키 관리
- CloudTrail 이벤트 예제

암호화 아티팩트

다음 표에는 EventBridge Scheduler가 저장 시 암호화하는 다양한 유형의 데이터와 각 범주에서 지원하는 KMS 키 유형이 설명되어 있습니다.

데이터 유형	설명	AWS 소유 키	고객 관리형 키
페이로드(최대 256KB)	대상에 전달되도록 일정을 구성할 때 일정의 `TargetInput` 파라미터에 지정하는 데이터입니다.	지원	지원
식별자 및 상태	일정의 고유 이름 및 상태(활성화, 비활성화).	지원	지원되지 않음
Scheduling configuration(예약 구성)	스케줄링 표현식(예: 반복 일정에 대한 rate 또는 cron 표현식, 일회성 간접 호출의 타임스탬프, 일정의 시작	지원	지원되지 않음

데이터 유형	설명	AWS 소유 키	고객 관리형 키
	날짜, 종료 날짜 및 시간대).		
대상 구성	대상의 Amazon 리소스 이름(ARN) 및 기타 대상 관련 구성 세부 정보.	지원	지원되지 않음
간접 호출 및 실패 동작 구성	유연한 기간 구성, 일정의 재시도 정책, 전달 실패에 사용되는 DLQ(Dead Letter Queue) 세부 정보.	지원	지원되지 않음

EventBridge Scheduler는 이전 표에 설명된 대로 대상 페이로드를 암호화하고 해독할 때만 고객 관리 키를 사용합니다. 고객 관리 키를 사용하도록 선택하면 EventBridge Scheduler는 페이로드를 두 번 암호화하고 해독합니다. 한 번은 기본값을 AWS 소유 키사용하고 다른 한 번은 지정한 고객 관리 키를 사용합니다. 다른 모든 데이터 유형의 경우 EventBridge Scheduler는 AWS 소유 키 기본값만 사용하여 저장된 데이터를 보호합니다.

다음 the section called “KMS 키 관리” 섹션을 통해 Scheduler에서 고객 관리 키를 사용하기 위해 IAM 리소스와 키 정책을 관리하는 방법을 알아보십시오. EventBridge

KMS 키 관리

필요에 따라 고객 관리 키를 제공하여 스케줄이 대상으로 전송하는 페이로드를 암호화하고 해독할 수 있습니다. EventBridge 스케줄러는 페이로드를 최대 256KB까지 암호화하고 복호화합니다. 고객 관리형 키를 사용하면 매달 요금이 발생하며, 프리 티어를 초과해서 요금이 부과됩니다. 고객 관리형 키를 사용하는 것도 AWS KMS 할당량의 일부로 계산됩니다. 요금에 대한 자세한 내용은 AWS Key Management Service 요금을 참조하세요.

EventBridge 스케줄러는 보안 주체와 연결된 IAM 권한을 사용하여 일정을 생성하여 데이터를 암호화합니다. 즉, Scheduler API를 호출하는 사용자 또는 역할에 필요한 AWS KMS 관련 권한을 연결해야 합니다 EventBridge . 또한 EventBridge Scheduler는 리소스 기반 정책을 사용하여 데이터를 복호화합니다. 즉, 일정과 관련된 실행 역할에도 데이터 암호 해독 시 API를 호출하는 데 필요한 AWS KMS 관련 권한이 있어야 합니다. AWS KMS

> **Note**
>
> EventBridge Scheduler는 임시 권한에 대한 권한 부여 사용을 지원하지 않습니다.

다음 섹션을 통해 Scheduler에서 고객 관리 AWS KMS 키를 사용하는 데 필요한 키 정책 및 필수 IAM 권한을 관리하는 방법을 알아보십시오. EventBridge

주제

- IAM 권한 추가
- 키 정책 관리

IAM 권한 추가

고객 관리형 키를 사용하려면 일정을 생성하는 자격 증명 기반 IAM 보안 주체에 다음 권한을 추가하고 일정에 연결하는 실행 역할을 추가해야 합니다.

고객 관리형 키에 대한 자격 증명 기반 권한

일정을 생성할 때 EventBridge Scheduler API를 호출하는 모든 보안 주체 (사용자, 그룹 또는 역할) 와 관련된 권한 정책에 다음 AWS KMS 작업을 추가해야 합니다.

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

                # Required to pass the execution role
                "iam:PassRole",

                "kms:DescribeKey",
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
    ]
```

```
}
```

- **kms:DescribeKey** - 제공한 키가 대칭 암호화 KMS 키인지 확인하는 데 필요합니다.
- **kms:GenerateDataKey**— EventBridge Scheduler가 클라이언트 측 암호화를 수행하는 데 사용하는 데이터 키를 생성하는 데 필요합니다.
- **kms:Decrypt**— EventBridge Scheduler가 암호화된 데이터와 함께 저장하는 암호화된 데이터 키의 암호를 해독해야 합니다.

고객 관리형 키에 대한 실행 역할 권한

EventBridge 스케줄의 실행 역할 권한 정책에 다음 작업을 추가하여 데이터를 해독할 때 Scheduler에 액세스하여 AWS KMS API를 호출하도록 허용해야 합니다.

```
{
    "Version": "2012-10-17",
    "Statement" : [
    {
      "Sid" : "Allow EventBridge Scheduler to decrypt data using a customer managed
 key",
      "Effect" : "Allow",
      "Action" : [
        "kms:Decrypt"

      ],
      "Resource": "arn:aws:kms:your-region:123456789012:key/your-key-id"
    }
  ]
}
```

- **kms:Decrypt**— EventBridge Scheduler가 암호화된 데이터와 함께 저장하는 암호화된 데이터 키의 암호를 해독해야 합니다.

새 일정을 생성할 때 EventBridge Scheduler 콘솔을 사용하여 새 실행 역할을 만들면 EventBridge Scheduler는 필요한 권한을 실행 역할에 자동으로 추가합니다. 하지만 기존 실행 역할을 선택한 경우 고객 관리형 키를 사용할 수 있으려면 역할에 필요한 권한을 추가해야 합니다.

키 정책 관리

를 사용하여 AWS KMS고객 관리 키를 생성하는 경우 기본적으로 키에는 스케줄의 실행 역할에 대한 액세스를 제공하는 다음과 같은 키 정책이 적용됩니다.

```
{
    "Id": "key-policy-1",
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Provide required IAM Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:root"
            },
            "Action": "kms:*",
            "Resource": "*"
        }
    ]
}
```

선택적으로 실행 역할에 대한 액세스만 제공하도록 키 정책의 범위를 제한할 수 있습니다. 고객 관리 키를 EventBridge 스케줄러 리소스에서만 사용하려는 경우 이 방법을 사용할 수 있습니다. 다음 키 정책 예제를 사용하여 키를 사용할 수 있는 EventBridge 스케줄러 리소스를 제한하세요.

```
{
    "Id": "key-policy-2",
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Provide required IAM Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::695325144837:root"
            },
            "Action": "kms:*",
            "Resource": "*"
        },
        {
            "Sid": "Allow use of the key",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:role/schedule-execution-role"
```

```
            },
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "*"
        }
    ]
}
```

CloudTrail 이벤트 예제

AWS CloudTrail 모든 API 호출 이벤트를 캡처합니다. 여기에는 EventBridge Scheduler가 고객 관리 키를 사용하여 데이터를 복호화할 때마다 발생하는 API 호출이 포함됩니다. 다음 예제는 고객 관리 키를 사용하여 `kms:Decrypt` 작업을 수행하는 EventBridge Scheduler를 보여주는 CloudTrail 이벤트 항목을 보여줍니다.

```
{
  "eventVersion": "1.08",
  "userIdentity": {
      "type": "AssumedRole",
      "principalId": "ABCDEABCD1AB12ABABAB0:70abcd123a123a12345a1aa12aa1bc12",
      "arn": "arn:aws:sts::123456789012:assumed-role/execution-
role/70abcd123a123a12345a1aa12aa1bc12",
      "accountId": "123456789012",
      "accessKeyId": "ABCDEFGHI1JKLMNOP2Q3",
      "sessionContext": {
          "sessionIssuer": {
              "type": "Role",
              "principalId": "ABCDEABCD1AB12ABABAB0",
              "arn": "arn:aws:iam::123456789012:role/execution-role",
              "accountId": "123456789012",
              "userName": "execution-role"
          },
          "webIdFederationData": {},
          "attributes": {
              "creationDate": "2022-10-31T21:03:15Z",
              "mfaAuthenticated": "false"
          }
      }
    },
    "eventTime": "2022-10-31T21:03:15Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
```

```
    "awsRegion": "eu-north-1",
    "sourceIPAddress": "13.50.87.173",
    "userAgent": "aws-sdk-java/2.17.295 Linux/4.14.291-218.527.amzn2.x86_64 OpenJDK_64-
Bit_Server_VM/11.0.17+9-LTS Java/11.0.17 kotlin/1.3.72-release-468 (1.3.72) vendor/
Amazon.com_Inc. md/internal exec-env/AWS_ECS_FARGATE io/sync http/Apache cfg/retry-
mode/standard AwsCrypto/2.4.0",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:123456789012:key/2321abab-2110-12ab-a123-
a2b34c5abc67",
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "encryptionContext": {
            "aws:scheduler:schedule:arn": "arn:aws:scheduler:us-
west-2:123456789012:schedule/default/execution-role"
        }
    },
    "responseElements": null,
    "requestID": "request-id",
    "eventID": "event-id",
    "readOnly": true,
    "resources": [
        {
            "accountId": "123456789012",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:123456789012:key/2321abab-2110-12ab-a123-
a2b34c5abc67"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
      "tlsVersion": "TLSv1.3",
      "cipherSuite": "TLS_AES_256_GCM_SHA384",
      "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
  }
}
```

전송 중 암호화

EventBridge 스케줄러는 네트워크로 이동하는 전송 데이터를 암호화합니다. 전송 계층 보안 (TLS) 은 EventBridge 스케줄러 API 작업을 호출할 때와 스케줄러가 일정을 호출할 때 대상 API를 호출할 때 EventBridge 데이터를 암호화합니다. 기본적으로 EventBridge 스케줄러는 전송 데이터를 암호화할 때

TLS 1.2를 사용합니다. 전송 중 암호화를 구성할 필요가 없으며 스케줄러를 사용할 때 다른 TLS 버전을 선택할 수 없습니다. EventBridge

EventBridge 스케줄러 API 사용 - API 작업 (예:) 을 수행하는 경우 EventBridge 스케줄러는 요청 본문과 헤더를 포함한 전체 HTTP 요청을 암호화합니다. `CreateSchedule` EventBridge 또한 스케줄러는 API에서 수신하는 전체 응답 객체를 암호화합니다.

대상 API 사용 — EventBridge 스케줄러가 일정을 호출하면 일정을 생성할 때 지정한 대상 API를 호출합니다. 대상에 이벤트를 전달할 때 EventBridge Scheduler는 요청 본문과 모든 헤더는 물론 대상으로부터 받는 응답을 포함한 전체 요청을 암호화합니다.

Amazon EventBridge 스케줄러에 대한 규정 준수 검증

특정 규정 준수 프로그램의 범위 내에 AWS 서비스 있는지 알아보려면AWS 서비스 규정 준수 프로그램의AWS 서비스 범위별, 규정 참조하여 관심 있는 규정 준수 프로그램을 선택하십시오. 일반 정보는 AWS 규정 준수 프로그램AWS 보증 프로그램 규정AWS 참조하십시오.

를 사용하여 AWS Artifact타사 감사 보고서를 다운로드할 수 있습니다. 자세한 내용은 의 보고서 https://docs.aws.amazon.com/artifact/latest/ug/downloading-documents.html 참조하십시오 AWS Artifact.

사용 시 규정 준수 AWS 서비스 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 퀵 스타트 가이드 - 이 배포 가이드에서는 아키텍처 고려 사항을 설명하고 보안 및 규정 준수에 AWS 중점을 둔 기본 환경을 배포하기 위한 단계를 제공합니다.
- Amazon Web Services의 HIPAA 보안 및 규정 준수를 위한 설계 — 이 백서에서는 기업이 HIPAA 적격 애플리케이션을 만드는 AWS 데 사용할 수 있는 방법을 설명합니다.

> **Note**
>
> 모든 AWS 서비스 사람이 HIPAA 자격을 갖춘 것은 아닙니다. 자세한 내용은 HIPAA 적격 서비스 참조를 참조하십시오.

- AWS 규정 준수 리소스AWS — 이 워크북 및 가이드 모음은 해당 산업 및 지역에 적용될 수 있습니다.
- AWS 고객 규정 준수 가이드 — 규정 준수의 관점에서 공동 책임 모델을 이해하십시오. 이 가이드에서는 보안을 유지하기 위한 모범 사례를 AWS 서비스 요약하고 여러 프레임워크 (미국 표준 기술 연

구소 (NIST), 결제 카드 산업 보안 표준 위원회 (PCI), 국제 표준화기구 (ISO) 등) 에서 보안 제어에 대한 지침을 매핑합니다.

- AWS Config 개발자 안내서의 규칙을 사용하여 리소스 평가 — 이 AWS Config 서비스는 리소스 구성이 내부 관행, 업계 지침 및 규정을 얼마나 잘 준수하는지 평가합니다.
- AWS Security Hub— 이를 AWS 서비스 통해 내부 AWS보안 상태를 포괄적으로 파악할 수 있습니다. Security Hub는 보안 제어를 사용하여 AWS 리소스를 평가하고 보안 업계 표준 및 모범 사례에 대한 규정 준수를 확인합니다. 지원되는 서비스 및 제어 목록은 Security Hub 제어 참조를 참조하십시오.
- Amazon GuardDuty — 환경에 의심스럽고 악의적인 활동이 있는지 AWS 계정모니터링하여 워크로드, 컨테이너 및 데이터에 대한 잠재적 위협을 AWS 서비스 탐지합니다. GuardDuty 특정 규정 준수 프레임워크에서 요구하는 침입 탐지 요구 사항을 충족하여 PCI DSS와 같은 다양한 규정 준수 요구 사항을 해결하는 데 도움이 될 수 있습니다.
- AWS Audit Manager— 이를 AWS 서비스 통해 AWS 사용량을 지속적으로 감사하여 위험을 관리하고 규정 및 업계 표준을 준수하는 방법을 단순화할 수 있습니다.

Amazon EventBridge 스케줄러의 레질리언스

AWS 글로벌 인프라는 가용 영역을 중심으로 AWS 리전 구축됩니다. AWS 리전 물리적으로 분리되고 격리된 여러 가용 영역을 제공합니다. 이 가용 영역은 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.

가용 영역에 대한 AWS 리전 자세한 내용은 글로벌 인프라를 참조하십시오AWS .

AWS 글로벌 인프라 외에도 EventBridge Scheduler는 데이터 복원력 및 백업 요구 사항을 지원하는 데 도움이 되는 여러 기능을 제공합니다.

Amazon EventBridge 스케줄러의 인프라 보안

Amazon EventBridge Scheduler는 관리형 서비스로서 AWS 글로벌 네트워크 보안의 보호를 받습니다. AWS 보안 서비스 및 인프라 AWS 보호 방법에 대한 자세한 내용은 AWS 클라우드 보안을 참조하십시오. 인프라 보안 모범 사례를 사용하여 AWS 환경을 설계하려면 Security Pillar AWS Well-Architected Framework의 인프라 보호를 참조하십시오.

AWS 게시된 API 호출을 사용하여 네트워크를 통해 EventBridge 스케줄러에 액세스할 수 있습니다. 고객은 다음을 지원해야 합니다.

- 전송 계층 보안(TLS) TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- DHE(Ephemeral Diffie-Hellman) 또는 ECDHE(Elliptic Curve Ephemeral Diffie-Hellman)와 같은 완전 전송 보안(PFS)이 포함된 암호 제품군 Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

또한 요청은 액세스 키 ID 및 IAM 주체와 관련된 비밀 액세스 키를 사용하여 서명해야 합니다. 또는 AWS Security Token Service(AWS STS)를 사용하여 임시 보안 인증을 생성하여 요청에 서명할 수 있습니다.

Amazon EventBridge 스케줄러에 대한 모니터링 및 지표

Amazon EventBridge 스케줄러 및 다른 AWS 솔루션의 신뢰성, 가용성 및 성능을 유지하려면 모니터링이 중요합니다. AWS는 EventBridge 스케줄러를 모니터링하고, 이상이 있을 때 이를 보고하고, 필요한 경우 자동 조치를 취할 수 있도록 다음과 같은 모니터링 도구를 제공합니다.

- Amazon CloudWatch는 AWS에서 실행하는 AWS 리소스와 애플리케이션을 실시간으로 모니터링합니다. 지표를 수집 및 추적하고, 사용자 지정 대시보드를 생성할 수 있으며, 지정된 지표가 지정한 임계값에 도달하면 사용자에게 알리거나 조치를 취하도록 경보를 설정할 수 있습니다. 자세한 내용은 Amazon CloudWatch 사용 설명서를 참조하세요.
- AWS CloudTrail은 직접 수행하거나 AWS 계정을 대신하여 수행한 API 호출 및 관련 이벤트를 캡처하고 지정한 Amazon S3 버킷에 로그 파일을 전송합니다. 어떤 사용자 및 계정이 AWS를 호출했는지, 어떤 소스 IP 주소에 호출이 이루어졌는지, 언제 호출이 발생했는지 확인할 수 있습니다. 자세한 내용은 AWS CloudTrail 사용 설명서를 참조하세요.

주제

Amazon CloudWatch를 사용하여 Amazon EventBridge 스케줄러 모니터링

원시 데이터를 수집하여 읽기 가능하며 실시간에 가까운 지표로 처리하는 CloudWatch를 통해 Amazon EventBridge 스케줄러를 모니터링할 수 있습니다. EventBridge 스케줄러는 모든 일정에 대한 지표 집합과 연결된 DLQ(Dead Letter Queue)가 있는 일정에 대한 추가 지표 세트를 내보냅니다. 일정에 맞게 DLQ를 구성하면 일정이 재시도 정책을 모두 사용했을 때 EventBridge 스케줄러가 추가 지표를 게시합니다.

이러한 통계는 15개월간 보관되므로 기록 정보를 보고 일정에 오류가 발생하는 이유를 더 잘 파악하고 근본적인 문제를 해결할 수 있습니다. 특정 임계값을 주시하다가 해당 임계값이 충족될 때 알림을 전송하거나 조치를 취하도록 경보를 설정할 수도 있습니다. 자세한 내용은 Amazon CloudWatch 사용 설명서를 참조하세요.

주제

- 측정기준
- 지표 보기
- 지표 목록

약관

네임스페이스

네임스페이스는 AWS 서비스의 CloudWatch 지표를 위한 컨테이너입니다. EventBridge 스케줄러의 네임스페이스는 `AWS/Scheduler`입니다.

CloudWatch 지표

CloudWatch 지표는 CloudWatch에 고유한 시간 순서별 데이터 포인트 세트를 나타냅니다.

차원

측정기준은 지표의 자격 증명에 속하는 이름/값 페어입니다.

Unit

통계에는 측정 단위가 포함되어 있습니다. EventBridge 스케줄러의 경우 단위에는 개수가 포함됩니다.

측정기준

이 섹션에서는 CloudWatch에서 EventBridge 스케줄러 지표에 사용되는 CloudWatch 차원 그룹화에 대해 설명합니다.

차원	설명
일정 그룹	CloudWatch를 사용하여 지표를 보려는 일정의 그룹입니다. 아직 그룹을 만들지 않은 경우, EventBridge 스케줄러는 사용자의 일정을 `default` 그룹과 연결합니다.

지표 보기

이 섹션에서는 CloudWatch에서 특정 EventBridge 스케줄러 일정의 성능 지표에 액세스하는 방법을 설명합니다.

차원에 대한 성능 지표를 보려면

1. CloudWatch 콘솔에서 지표 페이지를 엽니다.
2. AWS 리전 선택기를 사용하여 일정의 리전을 선택합니다.
3. 스케줄러 네임스페이스를 선택합니다.
4. 모든 지표 탭에서 차원(예: 일정 그룹 지표)을 선택합니다. 선택한 리전에서 생성한 모든 일정에 대한 지표를 보려면 계정 지표를 선택합니다.
5. 차원에 대한 CloudWatch 지표를 선택합니다. 예를 들어 InvocationAttemptCount 또는 InvocationDropedCount를 선택한 다음 그래프 검색을 선택합니다.
6. 그래프로 표시된 지표 탭을 선택하여 EventBridge 스케줄러 지표의 성능 지표를 확인합니다.

지표 목록

다음 표에는 모든 EventBridge 스케줄러 일정에 대한 지표와 DLQ를 구성한 일정에 대한 추가 지표가 나열되어 있습니다.

모든 일정에 대한 지표

네임스페이스	지표	Unit	설명
AWS/Scheduler	InvocationAttemptCount	개수	모든 간접 호출 시도에 대해 내보냅니다. 이 지표를 사용하여 EventBridge 스케줄러가 일정의 간접 호출을 시도하는지 확인하고 간접 호출이 계정 할

네임스페이스	지표	Unit	설명
			당량에 가까워지는 시점을 확인할 수 있습니다.
AWS/Scheduler	TargetErrorCount	개수	EventBridge 스케줄러가 대상 API를 호출한 후 대상이 예외를 반환할 때 내보냅니다. 이를 사용하여 대상으로의 전송이 실패하는 시점을 확인할 수 있습니다.
AWS/Scheduler	TargetErrorThrottledCount	개수	대상의 API 제한으로 인해 대상 간접 호출이 실패할 때 내보냅니다. 기본 원인이 EventBridge 스케줄러에서 수행한 대상 API 제한 직접 호출인 경우 전송 실패를 진단하는 데 사용합니다.

네임스페이스	지표	Unit	설명
AWS/Scheduler	InvocationThrottleCount	개수	EventBridge 스케줄러가 설정한 서비스 할당량을 초과하여 EventBridge 스케줄러가 대상 간접 호출을 조절할 때 내보냅니다. 이를 사용하여 EventBridge 스케줄러 할당량을 초과한 시점을 확인할 수 있습니다. 서비스 할당량에 대한 자세한 정보는 할당량 섹션을 참조하세요.

네임스페이스	지표	Unit	설명
AWS/Scheduler	InvocationDroppedCount	개수	일정의 재시도 정책이 소진된 후 EventBridge 스케줄러가 대상을 간접적으로 호출하는 시도를 중지할 때 내보냅니다. 재시도 정책에 대한 자세한 내용은 EventBridge 스케줄러 API 참조의 RetryPolicy를 참조하세요.

DLQ를 사용하는 일정에 대한 지표

네임스페이스	지표	Unit	설명
AWS/Scheduler	InvocationsSentToDeadLetterCount	개수	일정의 DLQ에 성공적으로 전달될 때마다 발생합니다. 이를 사용하여 이벤트가 DLQ로 전송되는 시기를 결정한 다음, 일정의 DLQ로 전달

네임스페이스	지표	Unit	설명
			된 이벤트에서 실패 원인을 파악하는 데 도움이 되는 추가 세부 정보를 확인하십시오.

네임스페이스	지표	Unit	설명
AWS/Scheduler	InvocationsFailedToBeSentToDeadLetterCount	개수	EventBridge 스케줄러가 DLQ에 이벤트를 전달할 수 없을 때 발생합니다. 이 두 가지 지표를 사용하여 EventBridge 스케줄러가 DLQ에 이벤트를 전송할 수 없는 이유를 파악하고 DLQ 구성을 수정하여 문제를 해결하십시오. 다음은 DLQ로 지정한 Amazon SQS 대기열이 존재하지 않는 경우의 InvocationsFailedToBeSentToDeadLetterCount_<error_code> 지표 예제입니다. InvocationsFailedT
AWS/Scheduler	InvocationsFailedToBeSentToDeadLetterCount_<error_code>	개수	

네임스페이스	지표	Unit	설명
			oBeSentToDeadLetterCount_ **AWS.SimpleQueueService.NonExistentQueue**
AWS/Scheduler	InvocationsSentToDeadLetterCount_Truncated_MessageSizeExceeded	개수	DLQ로 전송된 이벤트의 페이로드가 Amazon SQS에서 허용하는 최대 크기를 초과하고 EventBridge 스케줄러가 일정의 Input 속성에 지정한 페이로드를 잘라낼 때 발생합니다.

AWS CloudTrail을 사용하여 Amazon EventBridge 스케줄러 API 직접 호출 로깅

Amazon CloudWatch Events는 사용자, 역할 또는 AWS 서비스가 EventBridge 스케줄러에서 수행한 작업의 레코드를 제공하는 AWS CloudTrail 서비스와 통합됩니다. CloudTrail은 EventBridge 스케줄러에 대한 모든 API 직접 호출을 이벤트로 캡처합니다. 캡처되는 호출에는 EventBridge 스케줄러 콘솔에서 수행한 직접 호출과 EventBridge 스케줄러 API 작업에 대한 코드 직접 호출이 포함됩니다. 추적을 생성하면 EventBridge 스케줄러의 이벤트를 비롯한 CloudTrail 이벤트를 Amazon S3 버킷에 지속적으로 전달할 수 있습니다. 추적을 구성하지 않은 경우에도 CloudTrail 콘솔의 이벤트 기록에서 최신 이벤트를 볼 수 있습니다. CloudTrail에서 수집한 정보를 사용하여 EventBridge 스케줄러에 수행된 요청, 요

청이 수행된 IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다.

CloudTrail에 대한 자세한 내용은 AWS CloudTrail 사용 설명서를 참조하세요.

CloudTrail의 EventBridge 스케줄러 정보

CloudTrail은 계정 생성 시 AWS 계정에서 사용되도록 설정됩니다. 활동이 EventBridge 스케줄러에서 발생하면, 해당 활동이 이벤트 기록의 다른 AWS 서비스 이벤트와 함께 CloudTrail 이벤트에 기록됩니다. AWS 계정에서 최신 이벤트를 확인, 검색 및 다운로드할 수 있습니다. 자세한 내용은 CloudTrail 이벤트 기록을 사용하여 이벤트 보기를 참조하세요.

EventBridge 스케줄러의 이벤트를 포함하여 AWS 계정에 이벤트를 지속적으로 기록하려면 추적을 생성합니다. CloudTrail은 추적을 사용하여 Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. 콘솔에서 추적을 생성하면 기본적으로 모든 AWS 리전에 추적이 적용됩니다. 추적은 AWS 파티션에 있는 모든 리전의 이벤트를 로깅하고 지정된 Amazon S3 버킷으로 로그 파일을 전송합니다. 또는 CloudTrail 로그에서 수집된 이벤트 데이터를 추가 분석 및 처리하도록 다른 AWS 서비스를 구성할 수 있습니다. 자세한 내용은 다음 자료를 참조하세요.

- 추적 생성 개요
- CloudTrail 지원 서비스 및 통합
- CloudTrail에 대한 Amazon SNS 알림 구성
- 여러 지역에서 CloudTrail 로그 파일 받기 및 여러 계정에서 CloudTrail 로그 파일 받기

모든 EventBridge 스케줄러 작업은 CloudTrail에서 기록되며 Amazon EventBridge 스케줄러 API 참조에 설명되어 있습니다. 예를 들어 `CreateSchedule`, `UpdateSchedule`, `DeleteSchedule` 작업을 호출하면 CloudTrail 로그 파일에 항목이 생성됩니다.

모든 이벤트 및 로그 항목에는 요청을 생성한 사용자에 대한 정보가 들어 있습니다. 자격 증명 정보를 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청을 루트로 했는지 아니면 AWS Identity and Access Management(IAM) 사용자 자격 증명으로 했는지.
- 역할 또는 페더레이션 사용자에 대한 임시 보안 자격 증명을 사용하여 요청이 생성되었는지 여부.
- 다른 AWS 서비스에서 요청했는지 여부.

자세한 내용은 CloudTrail userIdentity 요소를 참조하세요.

EventBridge 스케줄러 로그 파일 항목 이해

추적이란 지정한 Amazon S3 버킷에 이벤트를 로그 파일로 입력할 수 있게 하는 구성입니다. CloudTrail 로그 파일에는 하나 이상의 로그 항목이 포함될 수 있습니다. 이벤트는 모든 소스의 단일 요청을 나타내며 요청된 작업, 작업 날짜와 시간, 요청 파라미터 등에 대한 정보를 포함합니다. CloudTrail 로그 파일은 퍼블릭 API 호출의 주문 스택 트레이스가 아니므로 특정 순서로 표시되지 않습니다.

아마존 스케줄러 할당량 EventBridge

AWS 계정에는 각 서비스에 대한 기본 할당량 (이전에는 한도라고 함) 이 있습니다. AWS 다르게 표시되지 않는 한 리전별로 각 할당량이 적용됩니다. 일부 할당량에 대한 증가를 요청할 수 있으며 일부 할당량은 늘릴 수 없습니다.

EventBridge 스케줄러의 할당량을 보려면 Service Quotas 콘솔을 엽니다. 탐색 창에서 AWS 서비스를 선택한 다음 스케줄러를 선택합니다. EventBridge

할당량 증가을 요청하려면 Service Quotas 사용 설명서의 할당량 증가 요청을 참조하십시오. Service Quotas에서 아직 할당량을 사용할 수 없는 경우 한도 증가 양식을 사용합니다.

> **Note**
>
> EventBridge 스케줄러의 `CreateScheduleUpdateSchedule`,`GetSchedule`, 및 TPS (초당 `DeleteSchedule` 트랜잭션) 할당량은 최대 수천 TPS까지 조정할 수 있습니다. 간접 호출 제한 할당량은 최대 수만 TPS까지 조정할 수 있습니다.

AWS 계정에는 스케줄러와 관련된 할당량이 다음과 같습니다. EventBridge

명칭	기본값	조정 가능	설명
CreateSchedule 요청 속도	지원되는 각 지역: 50	예	초당 최대 CreateSchedule 요청 이 할당량에 도달하면 EventBridge Scheduler는 남은 기간 동안 이 작업에 대한 요청을 거부합니다.
CreateScheduleGroup 요청 속도	지원되는 각 리전: 10개	예	초당 최대 CreateScheduleGroup 요청 이 할당량에 도달하면 EventBridge Scheduler는 남은 기간

명칭	기본값	조정 가능	설명
			동안 이 작업에 대한 요청을 거부합니다.
DeleteSchedule 요청 속도	지원되는 각 지역: 50	예	초당 최대 DeleteSchedule 요청 이 할당량에 도달하면 EventBridge Scheduler는 남은 기간 동안 이 작업에 대한 요청을 거부합니다.
DeleteScheduleGroup 요청 속도	지원되는 각 리전: 10개	예	초당 최대 DeleteScheduleGroup 요청 이 할당량에 도달하면 EventBridge Scheduler는 남은 기간 동안 이 작업에 대한 요청을 거부합니다.
GetSchedule 요청 속도	지원되는 각 지역: 50	예	초당 최대 GetSchedule 요청 이 할당량에 도달하면 EventBridge Scheduler는 남은 기간 동안 이 작업에 대한 요청을 거부합니다.
GetScheduleGroup 요청 속도	지원되는 각 리전: 10개	예	초당 최대 GetScheduleGroup 요청 이 할당량에 도달하면 EventBridge Scheduler는 남은 기간 동안 이 작업에 대한 요청을 거부합니다.

명칭	기본값	조정 가능	설명
간접 호출은 초당 트랜잭션의 한도를 조절합니다.	지원되는 각 리전: 500	예	호출은 정의된 대상에 전달되는 스케줄 페이로드입니다. 한도에 도달하면 간접 호출이 제한됩니다. 즉, 간접 호출이 계속 발생하지만 지연됩니다.
ListScheduleGroups 요청 속도	지원되는 각 리전: 10개	예	초당 최대 ListScheduleGroups 요청 이 할당량에 도달하면 EventBridge Scheduler는 남은 기간 동안 이 작업에 대한 요청을 거부합니다.
ListSchedules 요청 속도	지원되는 각 지역: 50	예	초당 최대 ListSchedules 요청 이 할당량에 도달하면 EventBridge Scheduler는 남은 기간 동안 이 작업에 대한 요청을 거부합니다.
ListTagsForResource 요청 속도	지원되는 각 리전: 10개	예	스케줄러 리소스와 연결된 태그를 나열합니다.
스케줄 그룹 수	지원되는 각 리전: 500	예	지역별 최대 스케줄 그룹 수

명칭	기본값	조정 가능	설명
스케줄 수	지원되는 각 리전: 1,000,000	예	지역별 최대 일정 수. 이 할당량에는 실행이 완료된 1회성 일정이 포함됩니다. 일회성 스케줄이 실행을 완료하고 대상을 간접적으로 호출한 후에는 해당 일정을 삭제하는 것이 좋습니다.
TagResource 요청 속도	지원되는 각 리전: 1	예	지정된 스케줄러 리소스에 하나 이상의 태그(키-값 쌍)를 할당합니다.
UntagResource 요청 속도	지원되는 각 리전: 1	예	지정된 스케줄러 리소스에서 하나 이상의 태그를 제거합니다.
UpdateSchedule 요청 속도	지원되는 각 지역: 50	예	초당 최대 UpdateSchedule 요청 이 할당량에 도달하면 EventBridge Scheduler는 남은 기간 동안 이 작업에 대한 요청을 거부합니다.

EventBridge 스케줄러의 할당량 및 서비스 엔드포인트에 대한 자세한 내용은 일반 참조 안내서의 Amazon EventBridge Scheduler 엔드포인트 및 할당량을 참조하십시오. AWS

EventBridge 스케줄러 사용 설명서에 대한 문서 기록

다음 표에서는 EventBridge 스케줄러에 대한 문서 릴리스를 소개합니다.

변경 사항	설명	날짜
실행 역할 변경 및 혼동된 대리자 방지	이 업데이트에서는 역할의 권한 정책에 혼동된 대리자 방지를 구현할 때 일정 그룹 리소스에 실행 역할이 적용되는 방식에 대한 변경 사항을 설명합니다. • the section called “혼동된 대리자 방지”	2023년 9월 7일
완료 후 일정 자동 삭제	EventBridge 스케줄러는 자동 삭제를 지원합니다. 자동 삭제를 구성하면 EventBridge 스케줄러는 마지막으로 계획된 간접 호출 이후에 일정을 삭제합니다. • the section called “일정 완료 후 삭제”	2023년 8월 2일
범용 대상 사용에 대한 주제가 업데이트되었습니다.	EventBridge 스케줄러가 대상으로 지정하고 통합할 수 있는 지원 서비스 목록을 업데이트했습니다. 이 업데이트에는 지원되지 않는 GET API 작업 목록도 포함되어 있으며 범용 대상 예제에 대한 개선 사항과 설명서 전반에 대한 기타 사소한 개선 사항이 포함되어 있습니다.	2023년 3월 17일

	• the section called “범용 대상 사용”	
시작일이 없는 속도 기반 일정에 대한 정보가 업데이트되었습니다.	`StartDate` 를 지정하지 않은 경우 EventBridge 스케줄러가 속도 기반 일정을 처리하는 방법에 대한 정보가 추가되었습니다. • the section called “속도 기반 일정”	2023년 3월 17일
스케줄러 그룹 관리에 관한 새 주제	EventBridge 스케줄러를 사용하여 스케줄러 그룹을 생성하는 방법에 대한 새 장을 추가했습니다. 이 장을 통해 그룹을 만들고, 그룹에 일정을 추가하고, EventBridge 스케줄러 리소스를 보다 쉽게 관리 및 모니터링할 수 있도록 태그를 적용하고, 마지막으로 그룹을 삭제하는 방법을 알아보십시오. • 일정 그룹 관리	2023년 3월 17일
일광 절약 시간제 및 시간대에 관한 새 주제	EventBridge 스케줄러가 일광 절약 시간을 처리하는 방법과 다양한 시간대에서 일정을 생성하는 방법을 설명하는 새 섹션이 추가되었습니다. • the section called “일광 절약 시간제” • the section called “시간대”	2022년 11월 17일

지표에 관한 새 주제	EventBridge 스케줄러가 CloudWatch에 게시하는 지표를 설명하는 새 주제를 추가했습니다. 이러한 지표를 사용하여 간접 호출 실패를 모니터링하고 일정 관련 문제를 해결하는 방법을 이해할 수 있습니다. • the section called “CloudWatch를 사용한 모니터링”	2022년 11월 15일
최초 릴리스	EventBridge 스케줄러 사용 설명서의 첫 번째 릴리스.	2022년 11월 10일

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.